

A Thesis Submitted for the Degree of PhD at the University of Warwick

Permanent WRAP URL:

<http://wrap.warwick.ac.uk/164146>

Copyright and reuse:

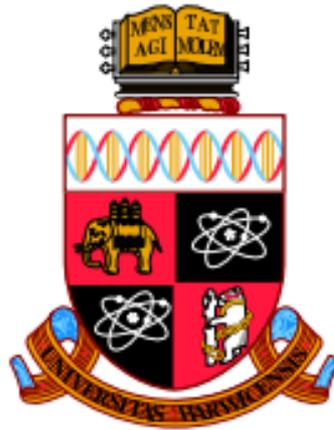
This thesis is made available online and is protected by original copyright.

Please scroll down to view the document itself.

Please refer to the repository record for this item for information to help you to cite it.

Our policy information is available from the repository home page.

For more information, please contact the WRAP Team at: wrap@warwick.ac.uk



Building Transformative Framework for Isolation and Mitigation of Quality Defects in Multi-Station Assembly Systems using Deep Learning

by

Sumit Sinha

Thesis

Submitted in partial fulfilment of the requirements

for the degree of

Doctor of Philosophy in Engineering

WMG

December 2021



Dedicated to my parents

Table of Contents

List of Figures.....	VI
List of Tables	X
List of Abbreviations	XII
List of Publications & Achievements	XIII
Declaration.....	XV
Acknowledgements	XVI
Abstract.....	XVII
1 Introduction.....	- 1 -
1.1 Background and Motivations under Industry 4.0	- 1 -
1.2 State-of-the-art Limitations and Research Requirements	- 8 -
1.3 Research Kernels and Roadmap	- 16 -
1.4 Thesis Organization	- 26 -
2 Literature Review	- 29 -
2.1 Modelling, Diagnosis and Correction of Assembly Systems	- 29 -
2.2 Deep Learning for 3D Object Classification and Segmentation.....	- 31 -
2.3 Deep Learning Applications in Manufacturing Fault Diagnosis	- 32 -
2.4 Soft Sensing for Industrial Processes.....	- 33 -
2.5 Bayesian Modelling for Uncertainty Quantification.....	- 36 -
2.6 Deep Reinforcement Learning for Control	- 37 -
2.7 Literature Assimilation and Knowledge Gaps Identification	- 38 -
3 Object Shape Error Response using Bayesian 3D Convolutional Neural Networks for Assembly Systems with Compliant Parts.....	- 42 -
3.1 Introduction.....	- 42 -
3.2 Problem Formulation	- 44 -
3.2.1 Object Shape Error Estimation in Manufacturing	- 44 -
3.2.2 3D Object Shape Error Voxelization	- 46 -
3.2.3 Uncertainty Estimation	- 47 -
3.3 Methods.....	- 48 -
3.3.1 Bayesian 3D CNN Model Architecture	- 48 -
3.3.2 Architecture Selection and Optimization.....	- 49 -
3.3.3 Model Training and Deployment.....	- 51 -
3.4 Case Study	- 54 -
3.4.1 Assembly Setup	- 54 -
3.4.2 Results.....	- 57 -
3.4.3 Benchmarking and Discussion.....	- 58 -
3.5 Conclusions.....	- 64 -
4 Root Cause Analysis of Multi-Station Assembly Systems with Non-Ideal Compliant Parts using Bayesian 3D U-Nets	- 65 -
4.1 Introduction.....	- 65 -
4.2 Problem Formulation	- 66 -
4.3 Methods.....	- 69 -
4.3.1 3D Object Shape Error Voxelization	- 69 -

4.3.2	Bayesian Neural Networks	- 70 -
4.3.3	Bayesian 3D U-Net Convolutional Architecture	- 71 -
4.3.4	Architecture Design Decisions	- 75 -
4.3.5	Closed-Loop Training and Deployment	- 77 -
4.3.6	Root Cause Analysis	- 81 -
4.4	Case Study	- 84 -
4.4.1	Multi-Station Assembly Setup	- 84 -
4.4.2	Interactions	- 88 -
4.4.3	Benchmarking: Process Parameter Estimation	- 89 -
4.4.4	Benchmarking: Uncertainty Quantification	- 91 -
4.4.5	Benchmarking: Object Shape Error Estimation	- 92 -
4.4.6	Object Shape Error Maps	- 93 -
4.5	Conclusions	- 94 -
5	Building a Scalable and Interpretable Bayesian Deep Learning Framework for Quality Control of Free Form Surfaces	- 96 -
5.1	Introduction	- 96 -
5.2	Related work on Scalability and Interpretability	- 97 -
5.3	Problem Formulation	- 98 -
5.4	Methods	- 99 -
5.4.1	Bayesian 3D U-Net Architecture	- 99 -
5.4.2	Closed-loop Sampling and Training	- 100 -
5.4.3	Uncertainty Guided Continual Learning	- 103 -
5.4.4	Transfer Learning	- 105 -
5.4.5	3D U-Net Architecture Interpretability	- 107 -
5.4.6	3D Gradient-Weighted Class Activation Maps	- 108 -
5.5	Case Study	- 111 -
5.5.1	Experimental Setup	- 111 -
5.5.2	Discussion: Scalability	- 114 -
5.5.3	Discussion: Interpretability	- 118 -
5.6	Conclusions	- 127 -
6	Building a Quality Correction Framework for Assembly Systems using Deep Reinforcement Learning	- 129 -
6.1	Introduction	- 129 -
6.2	Problem Formulation	- 131 -
6.3	Methodology: Object Shape Error Correction	- 137 -
6.3.1	Engineering Interpretation of CA Policies	- 139 -
6.3.2	NPI-Coefficient Selection	- 142 -
6.3.3	Deep Deterministic Policy Gradient (DDPG)	- 144 -
6.3.4	DDPG Actor-Critic Network Optimization & Training	- 146 -
6.3.5	Application Framework: OSEC Deployment	- 148 -
6.4	Case Study	- 150 -
6.4.1	Experimental Setup	- 150 -
6.4.2	Application Scenarios	- 152 -
6.4.3	Results	- 153 -
6.4.4	Benchmarking	- 158 -
6.5	Conclusions	- 161 -

7	Advanced Implementations and Industrial Applications	- 162 -
7.1	Physical Implementation.....	- 162 -
7.1.1	Industrial Demonstrator	- 162 -
7.1.2	Experimental Setup	- 163 -
7.1.3	VRM Validation.....	- 165 -
7.1.4	Results.....	- 166 -
7.2	Software Implementation.....	- 167 -
7.3	Industrial Applications.....	- 169 -
8	Conclusions and Future Work.....	- 171 -
8.1	Conclusions and Gaps Addressed.....	- 171 -
8.2	Contributions.....	- 177 -
8.3	Limitations	- 179 -
8.4	Future Work.....	- 181 -
	References.....	- 184 -
	Appendix A: Additional Background and Methods.....	- 215 -
	Appendix B: Software Implementation	- 219 -
	Appendix C: Published Journal Paper 1	- 224 -
	Appendix D: Published Journal Paper 2	- 236 -
	Appendix E: Under Review Journal Paper 3	- 258 -
	Appendix F: Under Review Journal Paper 4	- 274 -
	Appendix G: Industrial Demonstrator	- 295 -

List of Figures

Fig. 1.1. Body-in-white multi-station assembly.....	- 3 -
Fig. 1.2. Analysis and synthesis of MAS.....	- 18 -
Fig. 1.3. OSEC vs current approaches	- 22 -
Fig. 1.4. Research questions	- 26 -
Fig. 2.1. Modelling, diagnosis and correction of assembly systems.....	- 31 -
Fig. 2.2. Finite element models for assembly systems	- 31 -
Fig. 2.3. Deep learning-based 3D object classification and segmentation	- 32 -
Fig. 2.4. Deep learning applications in manufacturing fault diagnosis	- 34 -
Fig. 2.5. Deep learning for semiconductor defect detection and classification	- 34 -
Fig. 2.6. Deep learning for additive manufacturing design and quality evaluation.....	- 35 -
Fig. 2.7. Soft sensing for industrial processes	- 36 -
Fig. 2.8. Bayesian modelling for uncertainty quantification	- 37 -
Fig. 2.9. Deep reinforcement learning for control	- 38 -
Fig. 2.10. Literature assimilation and knowledge gap identification.....	- 41 -
Fig. 3.1. Object shape error propagation in assembly systems.....	- 46 -
Fig. 3.2. Bayesian 3D CNN model architecture of the OSER method.....	- 49 -
Fig. 3.3. Grid search for category one hyper-parameters	- 51 -
Fig. 3.4. Model training and deployment framework	- 54 -

Fig. 3.5. Assembly Process Parameters	- 55 -
Fig. 3.6. PCFR Stages of the Assembly Process.....	- 55 -
Fig. 3.7. MAE and R^2 across all process parameters	- 57 -
Fig. 3.8. Bayesian 3D CNN OSER Convergence.....	- 57 -
Fig. 3.9. Performance under different levels of fault collinearity and multiplicity-	59 -
Fig. 3.10. Convergence comparison for all benchmarking models	- 62 -
Fig. 3.11. Process Parameters Distribution Standard Deviations	- 63 -
Fig. 3.12. Process parameter distributions.....	- 64 -
Fig. 4.1. Multi-station assembly system	- 67 -
Fig. 4.2. Bayesian 3D U-Net Architecture.....	- 74 -
Fig. 4.3. Linking Bayesian 3D U-Net with MAS engineering	- 75 -
Fig. 4.4. Grid search results for U-Net architecture optimization	- 76 -
Fig. 4.5. Closed-loop training and deployment with RCA	- 80 -
Fig. 4.6. Cross-member assembly system configuration	- 85 -
Fig. 4.7. Cross-member assembly process parameters	- 86 -
Fig. 4.8. Model results and convergence	- 88 -
Fig. 4.9. Error (RMSE) vs uncertainty (IQR) comparison	- 92 -
Fig. 4.10. Actual and estimation comparison for upstream object shape errors....	- 94 -
Fig. 5.1. CLIP underpinned algorithm portfolio.....	- 110 -

Fig. 5.2. Assembly Cases.....	- 114 -
Fig. 5.3. Convergence comparison in all training scenarios	- 117 -
Fig. 5.4. Loss in performance vs improvement in convergence for CLIP.....	- 118 -
Fig. 5.5. Object Shape Error Voxelization.....	- 119 -
Fig. 5.6. 3D Grad-CAMs for various encoder levels.....	- 120 -
Fig. 5.7. 3D Grad-CAMs for various levels of the decoder.....	- 121 -
Fig. 5.8. Comparison of actual vs estimated upstream assemblies.....	- 121 -
Fig. 5.9. 3D Grad-CAMs for Attention at various levels of the decoder.....	- 122 -
Fig. 5.10. 3D Grad-CAMs for residual connection at various encoder levels.....	- 123 -
Fig. 5.11. Part Variation Root Cause	- 124 -
Fig. 5.12. Positioning Root Cause	- 125 -
Fig. 5.13. Clamping Root Cause.....	- 125 -
Fig. 5.14. Joining Root Cause.....	- 126 -
Fig. 5.15. Part Variation and Clamping Root Causes	- 127 -
Fig. 6.1. OSEC approach	- 138 -
Fig. 6.2. DDPG agent.....	- 148 -
Fig. 6.3. OSEC training and deployment framework	- 150 -
Fig. 6.4. Cross member assembly	- 151 -
Fig. 6.5. Training reward per episode	- 155 -

Fig. 6.6. Training steps per episode	- 155 -
Fig. 6.7. Scenario 1 policies.....	- 156 -
Fig. 6.8. Scenario 2 policies.....	- 156 -
Fig. 6.9. Scenario 3 policies.....	- 157 -
Fig. 6.10. Object shape error outputs for Scenario 1 policies.....	- 157 -
Fig. 6.11. Object shape error outputs for Scenario 2 policies.....	- 158 -
Fig. 6.12. Object shape error outputs for Scenario 3 policies.....	- 158 -
Fig. 7.1. CAE implementation of the demonstrator case study	- 164 -
Fig. 7.2. Physical implementation of the demonstrator case study.....	- 164 -
Fig. 7.3. 3D Scanner for point cloud data collection	- 165 -
Fig. 7.4. Demonstrator results.....	- 167 -
Fig. 8.1 Framework for isolation & mitigation of dimensional quality defects ..	- 176 -

List of Tables

Table 1.1: Research Roadmap	- 28 -
Table 3.1: Object detection & OSER comparison	- 51 -
Table 3.2: Benchmarking Results	- 61 -
Table 4.1. Fault Localization	- 82 -
Table 4.2. OSER-MAS methods, application and assumptions	- 84 -
Table 4.3: Process Parameters Description.....	- 87 -
Table 4.4: Benchmarking: Process Parameter Estimation	- 90 -
Table 4.5: Benchmarking: Object Shape Error Estimation	- 92 -
Table 4.6: Comparison of OSER (Chapter 3) & OSER-MAS.....	- 93 -
Table 5.1: Closed-Loop Sampling and Training Framework	- 103 -
Table 5.2: Continual Learning Framework.....	- 105 -
Table 5.3: Transfer Learning Framework.....	- 107 -
Table 5.4: Assembly Cases	- 114 -
Table 5.5: CLIP framework scalability performance	- 117 -
Table 5.6: Voxel Resolution Sensitivity Study.....	- 119 -
Table 6.1. NPI-Coefficient Selection.....	- 143 -
Table 6.2. Process Parameters Description.....	- 152 -
Table 6.3. OSEC Results	- 154 -

Table 6.4. Benchmarking..... - 160 -

Table 7.1. VRM Calibration Results..... - 166 -

List of Abbreviations

1D/2D/3D:	1/2/3 – Dimensional
AI:	Artificial Intelligence
APC:	Automatic Process Control
CAD:	Computer-Aided Design
CAE:	Computer-Aided Engineering
CA:	Correction Action(s)
CLIP:	Closed-loop In-Process
CNN:	Convolutional Neural Networks
DDPG:	Deep Deterministic Policy Gradient
E2E:	End-to-End
DRL:	Deep Reinforcement Learning
FEA:	Finite Element Analysis
FEM:	Finite Element Model
GPU:	Graphical Processing Unit
KPI:	Key Performance Indicator
MAS:	Multi-Station Assembly System
MTTR:	Mean-time-to-resolution
NPI:	New Product (or Production System) Introduction. The NPI is understood in the context of this thesis as new production system introduction which includes process design, production launch and full production
OSEC:	Object Shape Error Correction
OSEM:	Object Shape Error Modelling
OSER:	Object Shape Error Response
RC:	Root Cause(s)
RCA:	Root Cause Analysis; also called as RC isolation
SPC:	Statistical Process Control
VRM:	Variation Response Method
VSA:	Variation Simulation Analysis

List of Publications & Achievements

Journal Papers

Paper 1: S. Sinha, P. Franciosa, and D. Ceglarek, "Object Shape Error Response using Bayesian 3D Convolutional Neural Networks for Assembly Systems with Compliant Parts," vol. 17, no. 10, pp. 6676-6686, IEEE Transactions on Industrial Informatics, 2021, DOI: 10.1109/TII.2020.3043226.

<https://ieeexplore.ieee.org/document/9286512> | Also in Appendix C; contributing to Chapter 3.

Paper 2: S. Sinha, P. Franciosa, and D. Ceglarek, "Building a Scalable and Interpretable Bayesian Deep Learning Framework for Quality Control of Free Form Surfaces," IEEE Access, vol. 9, 2021, DOI: 10.1109/ACCESS.2021.3068867.

<https://ieeexplore.ieee.org/document/9386067> | Also in Appendix D; contributing to Chapter 5.

Paper 3: S. Sinha, P. Franciosa, and D. Ceglarek, "Root Cause Analysis of Multi-Station Assembly Systems with Non-Ideal Compliant Parts using Bayesian 3D U-Nets," in review, IEEE Transactions on Automation Science and Engineering, 2021.

Also in Appendix E; contributing to Chapter 4.

Paper 4: S. Sinha, P. Franciosa, and D. Ceglarek, "Building a Quality Correction Framework for Assembly Systems using Deep Reinforcement Learning," in review, Journal of Manufacturing Systems, 2021.

Also in Appendix F; contributing to Chapter 6.

Paper 5: P. Franciosa, M. Sokolov, S. Sinha, T. Sun, and D. Ceglarek, "Deep learning enhanced digital twin for Closed-Loop In-Process quality improvement," CIRP Annals, vol. 69, no. 1, pp. 369–372, 2020, DOI: 10.1016/j.cirp.2020.04.110.

<https://www.sciencedirect.com/science/article/abs/pii/S0007850620301323>

Conference Papers

S. Sinha, P. Franciosa, and D. Ceglarek, "Object Shape Error Correction using Deep Reinforcement Learning for Multi-station assembly systems," Proceedings of 19th IEEE International Conference on Industrial Informatics (INDIN), 2021.

<https://ieeexplore.ieee.org/document/9557359> .

S. Sinha, P. Franciosa and D. Ceglarek, "Object Shape Error Response using Bayesian 3D Convolutional Neural Networks for Assembly Systems with Compliant Parts," Proceedings of 18th IEEE International Conference on Industrial Informatics (INDIN), 2020, pp. 104-109, doi: 10.1109/INDIN45582.2020.9442245. <https://ieeexplore.ieee.org/document/9442245>

S. Sinha, E. Glorieux, P. Franciosa, and D. Ceglarek, "3D Convolutional Neural networks to estimate assembly process parameters using 3D point clouds," *Proceedings of SPIE*, Vol.

11059, Article number: 110590B, (SPIE, Bellingham, WA, 2019), doi: 10.1117/12.2526062, 24 - 27 June 2019, Munich, Germany.
<https://www.spiedigitallibrary.org/conference-proceedings-of-spie/11059/110590B/3D-convolutional-neural-networks-to-estimate-assembly-process-parameters-using/10.1117/12.2526062.short?SSO=1>

Conference Presentations

Object Shape Error Correction using Deep Reinforcement Learning for Multi-station assembly systems, 19th IEEE Conference of Industrial Informatics (INDIN), July 21-23, 2021, Palma de Mallorca, Spain.

Object Shape Error Response using Bayesian 3D Convolutional Neural Networks for Assembly System with Compliant Parts, 18th IEEE Conference of Industrial Informatics (INDIN), July 20-23, 2020, Coventry, UK.

3D Convolutional Neural networks to estimate assembly process parameters using 3D point-clouds, SPIE Optical Metrology, June 24-27, 2019, Munich, Germany

Software – Python Library Bayesian Deep Learning for Manufacturing

S. Sinha, P. Franciosa, and D. Ceglarek, "Bayesian deep learning for manufacturing," 2020. [Online]: Available. https://github.com/sumitsinha/Deep_Learning_for_Manufacturing
Also in Appendix B; contributing to Chapter 7.

Industrial Demonstrator – In-Process Quality Improvement (IPQI) EPSRC Project

S. Sinha, E. Glorieux, M. Babu, P. Franciosa, and D. Ceglarek, "Demonstrator: Inline Quality Monitoring with Root Cause Diagnosis," 2020. [Online]: Available. https://sumitsinha.github.io/Deep_Learning_for_Manufacturing/html/real_system_implementation.html

"In-Process Quality Improvement (IPQI) Project" 2020. [Online]: Available. https://warwick.ac.uk/fac/sci/wmg/research/materials/dlm/projects/ipqi_new/
Also in Appendix G; contributing to Chapter 7.

Workshops & Invited Talks

When CAE Simulation meets Artificial Intelligence: Deep Learning for Manufacturing, Catapult WMG Workshop, University of Warwick, UK, 2020. <https://warwick.ac.uk/fac/sci/wmg/mediacentre/wmgevents/getconnected-whencaemeetsai>

Bayesian Deep Learning for Manufacturing, Invited Talk – Jaguar Land Rover Machine Learning Workgroup, University of Warwick, UK, 2021.

Independent Annual Reviews

Dr Paul Jenkins (Computer Science & Statistics) and Prof Alan Chalmers (WMG)

Annual Review Year 2: "The student is making excellent progress. He has had his work accepted at a top journal and major conferences and is preparing further papers. The student gave a very detailed presentation of his work and is highly knowledgeable in his field. He has not been negatively affected by the lockdown and looks well on track to complete on time."

Annual Review Year 1 (upgrade): "The student gave a detailed presentation and was able to answer all questions in a clear and articulate manner. He has understood the potential risks in his project and has taken steps to mitigate them, e.g. using simulated data if real data not available."

Declaration

This thesis is my original work and submitted to the University of Warwick to support my application for the degree of Doctor of Philosophy. I declare that this thesis has not been submitted in any previous application for any degree. The work presented was carried out by the author, except where explicitly stated. The author has published parts of this thesis which are listed in the list of publications.

Acknowledgements

The journey towards my doctoral degree has been one of exponential learning and personal development. I would like to thank my supervisors Prof. Dariusz Ceglarek and Prof. Pasquale Franciosa, for guidance, support, motivation and suggestions to improve the quality of my research. Special thanks to Prof. Darek Ceglarek for his critical feedback on my paper drafts that helped present ideas in the best way possible. Special thanks to Prof. Pasquale Franciosa for providing the technical platform to enable my work. I feel this was the best supervisor combination any PhD candidate could have to help him perform to the best of his capabilities. I would also like to thank Prof. M. K. Tiwari for his motivation to pursue my research at WMG.

I would like to thank all members of the DLM group for their support and feedback over the years. Special thanks to Emile and Manoj, who helped me understand how challenging it is to implement solutions in the real world. A special thanks to the open-source deep learning community for providing open-source implementations and support that enabled swift implementation of the research work.

I would like to acknowledge the financial support received from the:

- WMG-IIT scholarship awarded by WMG, University of Warwick
- UK EPSRC EP/K019368/1: "Self-Resilient Reconfigurable Assembly Systems with In-process Quality Improvement" project
- UKRI open access fund

Pursuing a PhD in a new country along with a pandemic is a very challenging task. I would like to thank my friends Sakshi, Sana, Anubhav, Shubham, Anand, Sisir, Abhijeet, Asif and Yash, who helped and motivated me during difficult times. For all those whom I may have inadvertently left out in name, I extend my sincere thanks.

Saving the most important for last, I would like to thank my mother and father for having the strength to send their only child to a faraway country and then supporting and motivating me in every way possible to become the best version of myself. I am always indebted to their love and hope I make them proud will everything the future holds for me.

Abstract

The manufacturing industry is undergoing significant transformation towards electrification (e-mobility). This transformation has intensified critical development of new lightweight materials, structures and assembly processes supporting high volume and high variety production of Battery Electric Vehicles (BEVs). As new materials and processes get developed it is crucial to address quality defects detection, prediction, and prevention especially given that e-mobility products interlink quality and safety, for example, assembly of ‘live’ battery systems. These requirements necessitate the development of methodologies that ensure quality requirements of products are satisfied from Job 1. This means ensuring high *right-first-time* ratio during process design by reducing manual and ineffective trial-and-error process adjustments; and, then continuing this by maintaining *near zero-defect manufacturing* during production by reducing Mean-Time-to-Detection and Mean-Time-to-Resolution for critical quality defects. Current technologies for isolating and mitigating quality issues provide limited performance within complex manufacturing systems due to (i) limited modelling abilities and lack capabilities to leverage point cloud quality monitoring data provided by recent measurement technologies such as 3D scanners to isolate defects; (ii) extensive dependence on manual expertise to mitigate the isolated defects; and, (iii) lack of integration between data-driven and physics-based models resulting in limited industrial applicability, scalability and interpretability capabilities, hence constitute a significant barrier towards ensuring quality requirements throughout the product lifecycle.

The study develops a *transformative framework* that goes beyond improving the accuracy and performance of current approaches and overcomes fundamental barriers for isolation and mitigation of *product shape error* quality defects in multi-station assembly systems (MAS). The proposed framework is based on three methodologies which explore MAS: (i) *response* to quality defects by isolating process parameters (root causes (RCs)) causing unaccepted shape error defects; (ii) *correction* of the isolated RCs by determining corrective actions (CA) policy to mitigate unaccepted shape error defects; and, (iii) *training, scalability and interpretability* of (i) and (ii) by establishing closed-loop in-process (CLIP) capability that integrates in-line point cloud data, deep learning approaches of (i) and (ii) and physics-based models to provide comprehensive data-driven defect identification and RC isolation (causality analysis). The developed methodologies include:

(i) *Object Shape Error Response (OSER)* to isolate RCs within single- and multi-station assembly systems (*OSER-MAS*) by developing Bayesian 3D-convolutional neural network architectures that process point cloud data and are trained using physics-based models and have capabilities to relate complex product shape error patterns to RCs. It quantifies uncertainties and is applicable during the design phase when no quality monitoring data is available.

(ii) *Object Shape Error Correction (OSEC)* to generate CAs that mitigate RCs and simultaneously account for cost and quality key performance indicators (KPIs), MAS reconfigurability, and stochasticity by developing a deep reinforcement learning framework that estimates effective and feasible CAs without manual expertise.

(iii) *Closed-Loop In-Process (CLIP)* to enable industrial adoption of approaches (i) & (ii) by *firstly* enhancing the scalability by using (a) closed-loop training, and (b) continual/transfer learning. This is important as training deep learning models for a MAS is time-intensive and requires large amounts of labelled data; *secondly* providing interpretability and transparency for the estimated RCs that drive costly CAs using (c) 3D gradient-based class activation maps.

The methods are implemented as independent kernels and then integrated within a transformative framework which is further verified, validated, and benchmarked using industrial-scale automotive sheet metal assembly case studies such as car door and cross-member. They demonstrate 29% better performance for RC isolation and 40% greater effectiveness for CAs than current statistical and engineering-based approaches.

1 Introduction

1.1 Background and Motivations under Industry 4.0

The manufacturing industry is under tremendous pressure to consistently deliver high-quality products while ensuring high productivity and cost-efficiency in the face of increasing product variety and smaller batch sizes. Moreover, it must do so with ever-decreasing time-to-market. These requirements necessitate the development of methodologies that ensure quality requirements of products are satisfied from Job 1 by ensuring that quality issues are isolated and mitigated efficiently and effectively. This means ensuring right-first-time and continuing this requirement throughout the new product introduction (NPI) stages, namely design, pre-production, production launch and full production towards near zero-defect manufacturing. Therefore, the manufacturing industry needs a transformative framework [1]–[3] that can enable scalable digitalization and integration of *3D scanners data* [4], *artificial intelligence (AI) algorithms* [5], and *Computer-Aided Engineering (CAE) simulations* [6]. Currently, point cloud-based quality monitoring data captured by 3D scanners provides capability to monitor product 3D shape variations. However, the capability to leverage point cloud data to isolate and mitigate quality issues does not exist as current approaches are designed to work on few sampled points and are based on statistical analysis or engineering expertise and cannot leverage the granular shape information provided by high-dimensional point cloud data. Deep learning-based AI algorithms provide capabilities to extract actionable insights from high dimensional point cloud data and can potentially be leveraged to isolate and mitigate sources of quality issues to ensure high quality and productivity with cost-efficiency. However, while 3D scanners data supported by deep learning algorithms can provide comprehensive data-driven correlation analysis critical for isolation of quality issues, it is insufficient to determine and mitigate the causality or root cause(s) (RC(s)) of the quality issue. The determination of this causality further requires integrating data and physics-based CAE models, currently lacking in the industry. These gaps do not allow in-process [1] root cause analysis (RCA) or correction of quality issues necessary to prevent defects generation and their propagation in multi-stage manufacturing systems. Addressing

these gaps requires a framework integrating 3D scanners data, AI algorithms and CAE simulations. The framework must have the capability to leverage product/process data (for e.g., point cloud from 3D scanners) to isolate RCs of quality issues and further estimate corrective action(s) (CAs) that mitigate the sources of quality issues in an *end-to-end* manner. An end-to-end attribute of the aforementioned framework can be interpreted as the capability to transform high dimensional data collected from manufacturing systems into actionable insights such as RCs and CAs to mitigate quality issues without the need for any *manual expertise*.

Developing such an end-to-end framework enables the realization of various Industry 4.0 milestones such as *Zero-Defect-Manufacturing*, *Right-First-Time* and *Resilient Manufacturing* [7] which eventually drive towards scenarios where all quality requirements are met from Job 1. Industry 4.0 is referred to as the fourth industrial revolution that aims to enable autonomous decision-making processes by leveraging capabilities provided by advanced data collection and analysis technologies [1]. Zero-Defect-Manufacturing and Right-First-Time aim to prevent quality defects. Simultaneously, resilient manufacturing systems strategies integrate capabilities within the system to recover quickly from such defects within a stipulated amount of time. These milestones collectively drive towards manufacturing scenarios where all quality requirements of manufactured products are satisfied. However, achieving these milestones is challenging due to increased product variety and smaller batch sizes with ever-decreasing time-to-market. Traditionally dimensional quality has been the key criteria for automotive manufacturers to remain competitive, reduce quality costs and ensure high customer satisfaction. In new trends such as e-mobility (also known as Battery Electric Vehicle (BEV)) dimensional quality plays a much bigger role as it is directly linked with safety and electrical functionality. E.g., in automotive battery packs, there need to be no gaps in the battery enclosures to prevent water and chemical leakage, any defects within such systems can lead to safety issues. Secondly, battery packs consist of a large number of interconnected cells that require mechanical and electrical connections [8]. Any defects in the connectivity will cause electrical functional failures hence compromising the battery pack. These interdependencies lead to manufacturing processes requiring high right-first time during design and then zero-

defect manufacturing during production. Hence it is essential to develop closed-loop transformative frameworks for the realization of such stringent requirements.

Currently, numerous manufacturing applications require fully automatic robotic assembly stations, each with multiple stages involving positioning, clamping, fastening (joining), and release (PCFR) to be placed in the production line [9]–[11]. These are known as multi-station assembly systems (MAS). Each robotic assembly station comprises a robot with end effector holding joining head (i.e., laser welding, resistance spot welding (RSW), self-piercing riveting (SPR), friction stir welding (FSW), among others), and fixtures locating parts to be assembled. A typical production MAS line for automotive body assembly (Body-in-White (BIW)) consists of hundreds of stamping parts and components processed along 60-85 assembly stations with a production rate of 30-65 vehicles per hour with 3-5 varieties of vehicles being produced simultaneously on a single assembly line [9] [54]. An example of a BIW MAS is given in Fig. 1.1. All elements of the robotic station are affected by inherent variation caused by initial calibration, process deterioration, and routine maintenance, and as a result, might lead to diminished product quality. Diminished dimensional quality leads to increased risks of failure to deliver products at the required functional performance [12].

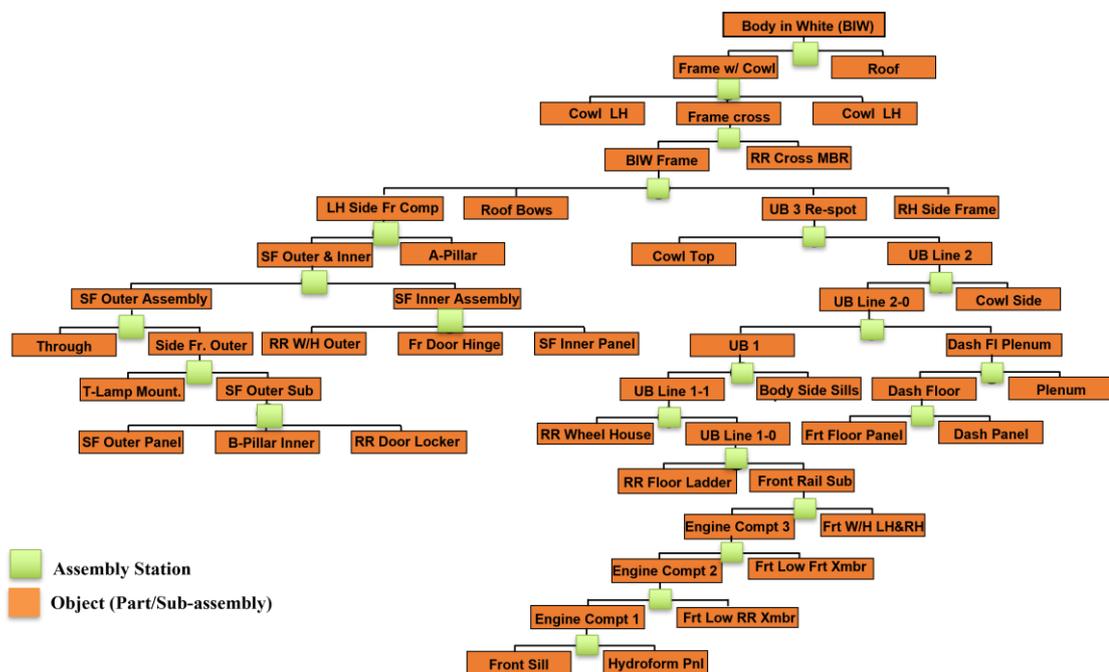


Fig. 1.1. Body-in-white multi-station assembly

Additionally, many products are made of deformable non-ideal parts [9], for which dimensional and geometric error/variation (*shape error*) is one of the leading quality problems. Two-thirds of all quality issues in the automotive, aerospace, and shipbuilding sectors occur due to dimensional variations [12]. Hence, it is essential to develop approaches to ensure that MAS can continue to produce high-quality products in the presence of non-ideal parts [9][13]. Non-ideal parts can be considered as *objects* characterizing a genuine manufactured part with dimensional and geometric variations [14] termed as *Object Shape Errors* [15]. The developed approaches should be capable of isolating and mitigating the RCs, i.e., sources of *object shape error* defects. This requires capabilities for accurate root cause analysis (RCA: isolation of sources of object shape error; also called as RC isolation) and furthermore, design and implementation of feasible and effective CAs for mitigation of sources of object shape error within MAS. *Hence, there is a need to develop a transformative framework for integrating 3D Scanners data, deep learning models and CAE simulations to ensure isolation and mitigation of object shape errors within MAS. This framework will be underpinned by approaches researched and developed as kernels through the work undertaken in this thesis. These approaches will have capabilities to map data collected from MAS to actionable insights such as RCs and CAs in an end-to-end manner. The main elements of the proposed framework with the underpinned approaches are discussed below.*

Recent advances in measurement technology such as 3D Optical Scanners [4] (please also see Appendix A for more details) that leverage multi-wave light technologies have enabled in-line measurement for dimensional quality of free form compliant surfaces such as sheet metal parts [16][17]. These provide high-resolution point cloud data on a relatively large coverage area of up to $500 \times 500 \text{ mm}$ in a short cycle time and a sufficient working distance [18]. However, currently there is no approach that can analyse high-dimensional point cloud data to obtain actionable insights related to quality issues. *Hence, it is crucial to develop an approach that can be trained and deployed in MAS with 3D Scanners and effectively extract local and global spatial features from the point cloud data and next leverage the features to isolate and mitigate*

sources of object shape error. This approach will be developed as a part of Chapter 3.

As markets get competitive in terms of product quality, production volume and costs, manufacturers aim to leverage recent developments in artificial intelligence. Deep neural networks (please see Appendix A for more details) have revolutionized data-intensive tasks that involve generating insights from high dimensional input data [19], [20]. Generally, 2D/3D Convolutional neural networks (CNN) are known to perform well for analysis of spatial data such as depth images [21], point clouds [22], mesh [23], and medical scans [24]. They have seen recent applications for tasks such as control systems [25], object detection [20], video analysis [26] and cancer detection [27]. Manufacturing is one of the major domains that has benefited from this development [28] [29]. However, currently, there are no deep learning approaches that can transform the shape error information into RCs or CAs corresponding to quality issues. Hence, *it is crucial to develop architectures that can utilize the spatial correlation present in point cloud or mesh data of non-ideal parts (objects) and build models that isolate RCs of object shape error. This approach will be developed as a part of Chapters 3 and 4.*

Computer-Aided Engineering (CAE) simulations enable modelling propagation of shape error in MAS through a combination of simulation and Finite Element Method (FEM) techniques [30]. These techniques have been known as Variation Simulation Analysis (VSA) [31][32]. These enable accurate estimation of variations (shape error) in MAS with non-ideal objects through high-fidelity multi-physics, first-principle models. Hence, they can potentially provide comprehensive datasets from the design phase of the MAS itself when no production data is available. The data includes various combinations of the shape error source input, i.e. process parameters of MAS and the corresponding estimated shape error output in the form of point clouds or mesh [6]. However, currently, there is no approach to effectively sample and build training datasets from CAE simulators. Hence, *it is crucial to develop an approach that can enable intelligent sampling from the CAE simulators and build comprehensive datasets essential to train models that enable isolation and mitigation of various sources of object shape error. This approach will be developed as a part of Chapters 3,4 and 5.*

Furthermore, manufacturing applications require that the RC isolation approaches will be supported by uncertainty quantification to provide an initial basis for cost/benefits analysis for potential CA implementation. Also, recent developments in explainable artificial intelligence [33] cautions making real-life decisions based on point estimates. Hence, from a probability theory perspective, artificial intelligence models should quantify the uncertainty in their predictions. Extensive work has been done on adding a Bayesian approach to neural networks that enables quantification of predictive uncertainty, prevents overfitting and requires comparatively lesser data to train [34]. Successful applications have been made in healthcare [24] and load forecasting [35]. Such models enable the segregation of the uncertainties into aleatoric and epistemic, the former quantifying the uncertainty due to uncontrollable factors such as system noise. In contrast, the latter quantifies uncertainties due to model structure and insufficient training data [36]. The manufacturing approaches for isolation and mitigation of RCs require uncertainty quantification due to shortcomings within CAE models, limited training data and noise within the incoming point cloud quality data. However, currently, no approach does the same, and hence, *it is crucial to incorporate this capability of uncertainty quantification when building models for RC isolation as they drive costly CAs. This approach will be developed as a part of Chapter 3.*

The determination of CAs to mitigate the isolated RCs is performed using two separate but interlinked steps (i) isolation of the RC using process and quality data and supported by statistical or deep learning approaches; and, (ii) determination and implementation of CA(s) to mitigate RCs which is currently based on experience and expertise to account for MAS constraints and stochasticity. The two steps are not seamlessly integrated in an end-to-end manner. As a result, the implementation of CAs can lead to high mean-time-to-resolution (MTTR) of quality defects that increases machine downtime, scrap and/or need for rework or repair, thereby, increasing the cost of quality and reducing productivity. Even the most experienced team is not able to take into consideration all stochasticities of manufacturing systems which directly affect mapping between RCs and CAs. At present there are no systematic data-driven analytical approaches that can link both steps into a single *end-to-end* approach, i.e., an approach that spans from quality data to RCs and to CAs via a single forward pass

without the need for experience and presence of human expert in the loop. This is primarily because currently, there are no CA approaches that can mathematically formulate the steps necessary to determine and implement CAs, thereby, resulting in CAs that are dependent on extensive manual expertise. Frequently, manually determined CAs cannot be implemented given the constraints of the system, such as ongoing production schedules or/and lack of clear cost/benefits analysis; this further motivates the need for end-to-end approaches that can transform manufacturing data to RCs and effective and feasible CAs that can be implemented within reasonable time and costs. This approach will be developed as a part of Chapter 6.

Reinforcement learning has performed well in various end-to-end tasks by training agents to optimize their control in an environment to obtain the maximum reward by learning state and action values and further learning optimal policies based on the values [37] [38]. Recently capabilities of deep neural networks in extracting features from high-dimensional image, video and text data and then building models to generalize across a wide variety of tasks have been integrated with reinforcement learning. The integration is known as deep reinforcement learning (DRL) [37]. It aims to integrate deep neural networks' feature extraction and generalization capabilities with sequential decision-making capabilities of reinforcement learning to build agents that can take optimal actions in complex environments with high-dimensional state or action spaces. To date, DRL has transformed end-to-end data-intensive tasks that involve taking actions based on high-dimensional data such as images and videos [37] and have achieved superior performance on complex games such as Atari and Go [37] [38]. Manufacturing has also benefitted from such approaches for scheduling and planning [39]. Various successful applications have been shown in robotics and control systems [40] [41]. These advances provide an opportunity to develop end-to-end approaches that can transform manufacturing data to RCs and effective and feasible CAs that can be implemented within reasonable time and costs. As such approaches currently, do not exist, *hence, it is crucial to develop approaches that leverage DRL and generate CAs that mitigate RCs of object shape errors while accounting for incoming stochasticity and MAS constraints. Additionally, ideally, such an approach should be applicable across the NPI lifecycle from design through production launch until full production.*

This approach will be developed as a part of Chapter 6.

The discussions above collectively drive the need for development of approaches integrating 3D Scanners, deep learning and reinforcement learning models and CAE simulators. These approaches should *isolate* (also known as diagnosis or perform RCA) and *mitigate* (also known as correction or CA) sources (also known as RCs) of object shape errors within MAS. While doing so, they should overcome various requirements and challenges currently not addressed by state-of-the-art methods and provide an end-to-end transformative framework that helps achieve conformance to quality requirements through all phases of the NPI and production lifecycle.

1.2 State-of-the-art Limitations and Research Requirements

Past approaches for RC isolation of shape error (dimensional and geometric variations) in single and multi-station assembly systems have seen much development in industrial applications and academic research. These models [42] can be grouped into two categories: (a) knowledge-based models; and (b) estimation-based data-driven models leveraging statistical and machine/deep learning techniques [43]. For single station systems, Apley and Shi [44] established a deviation transfer model based on process information such as fixture positioning and used least-squares to diagnose fault sources. Chang et al. [45] leveraged a linear model between shape error sources and measurement features followed by parameter estimation and statistical tests to diagnose shape error sources. Yu et al. [46] leveraged influence coefficients based on finite element modelling to establish shape errors between the sources of variation and measurements of flexible sheet metal parts. Further least-squares estimation was used to estimate errors in fixture positioning.

For MASs, Agarwal et al. [47] used regression models of sensor data. Zou et al. [48] proposed integrating BIC with LASSO variable selection. Shang et al. [49] proposed a Binary State Space Model (BSSM) for MASs to perform binary diagnosis. Jin et al. [50] proposed state-space and stream-of-variation (SoV) for multi-station shape error propagation of automotive assemblies. Ding et al. [51] extended the SoV method of assembly shape error of rigid parts using state-space considering different

fixture locating schemes. Leveraging the concept of SoV, various RCA models for MASs have been proposed. Ding et al. [52] [53] compared different variance estimation techniques and concluded a basis for method selection under different scenarios. Ceglarek and Prakash [54] proposed a shape error diagnosis based on enhanced piecewise least squares (EPLS) to detect and isolate collinear dimensional faults caused by fixture variation. Ceglarek and Shi [55] [56] employed pattern matching to diagnose fixtures based on principal component analysis. Liu and Hu [57] used designated component analysis for shape error diagnosis of flexible sheet metal parts. Various enhancements have been proposed using the knowledge of MASs [58] [59]. Given the ever-increasing complexity of MASs, increased computation capabilities and developments in machine learning, recently, RCA approaches [43] using machine learning have been proposed to overcome limitations of the above-stated methods such as linear approximations of the MAS. Du et al. [60] utilized artificial neural networks to monitor and identify process variability. Beruvides et al. [61] applied reinforcement learning to perform RCA. Bayesian Networks [62], [63] are seen as an alternative to solve small dataset problems and integrate process data and engineering knowledge.

Overall, the aforementioned approaches for RCA are approximately linear and are designed to work for a relatively small number of measurement points on each manufactured sub-assembly or final assembly (object). This significantly limits the application of 3D object shape error diagnosis in MAS. In MAS, final product accuracy and performance depend upon the accumulated performance of individual stations in the system. The quality of the subassemblies and the final assembly is influenced by (i) incoming non-ideal and deformable parts quality (dimensional accuracy); (ii) assembly process in each assembly station as determined by Positioning Clamping Fastening Release (PCFR) of each part and subassembly (object) in each assembly station as objects move through PCFR stages in each assembly station and then through all assembly stations of the MAS (shape error is being induced by all PCFR stages, controlled by fixturing and joining operations); (iii) product design as determined by object-to-object interactions within each station and further magnified between stations; and, (iv) MAS process as determined by station-to-station interactions due to re-orientation errors between stations (change of fixture locating layout between

stations).

The approaches for *RC isolation* of 3D object shape errors in MAS must have the capability to satisfy several *requirements* concerning:

- (i) *High data dimensionality* of a batch of 3D objects [64], which are defined by CAD (ideal parts) and point clouds (non-ideal parts) with millions of points for each part or subassembly (objects).
- (ii) *Non-linearity* due to compliant parts being constrained by assembly fixtures and part-to-part interactions [65][66].
- (iii) *Collinearities*, as many manufacturing systems, are ill-conditioned [67], with error patterns of crucial process parameters being near parallel, thus, yielding widely discrepant results.
- (iv) *High faults multiplicity* [68] as current near-zero-defects strategies require taking into consideration 6-sigma defects that lead to redefining defects from binary $\{0,1\}$, i.e., fault/no-fault, to continuous $[0,1]$, i.e., the fault being measured as a level of variation with dynamically changing threshold of acceptance, that significantly increases fault multiplicity.
- (v) *Uncertainty quantification* in the RCA output, as the identified RC frequently leads to costly corrective actions [69]. Therefore, it is crucial to enhance the RCA model by an uncertainty estimation of the predictions.
- (vi) *Dual data generation capability* by using metrology gages (3D scanners) and multi-physics-simulator needed for RCA model training. As the RCA model needs to be trained on a vast number of fault scenarios, which cannot be generated via physical systems; and the training needs to be done before the physical assembly systems are ready for production; there is a crucial need to generate data via high fidelity multi-physics simulator for training RCA model. Then, the RCA model will use point-cloud data of free-form surfaces obtained via a robotic 3D scanner when implemented in a physical system.
- (vii) *High dimensionality and heterogeneity* of process parameters [70], as MASs have a large number of process parameters inclusive of (a) real-valued parameters for non-ideal parts and fixturing/tooling (Positioning (P) and Clamping (C) stages of the assembly cycle); and, (b) binary parameters for joining operations (Fastening

(F) stage of the assembly cycle).

- (viii) *Fault Localization for RCA* [71] as MAS have multiple stations; hence, as compared to single-station assembly systems that required only isolating faulty process parameters within a single station [15], MAS require first localization of the assembly stations at fault followed by isolation of the faulty process parameter within the localized stations. Furthermore, faults originating in one station are propagated through all downstream stations. Hence for RCA within MAS, it is critical to localize the originating station of shape errors. The localized station can be further analysed to isolate a specific process parameter at fault. Methods for RCA of single-station assembly system cannot be adapted to perform RCA of MAS. Even if the methods assume all stations as a single station, the RC isolation will result in multiple process parameters being indicated as faulty rather than the specific process parameter at fault in the originating station. Hence RCA of MAS requires first localization of the faulty assembly station followed by isolating faulty process parameters within the localized station. This requires estimating shape error of sub-assemblies at the end of each upstream station of MAS as well as estimation of all process parameters within all stations of the MAS.
- (ix) *Scalability* as automotive MAS processes include hundreds of stamping parts and components, multiple stations with multiple stages in each station [9], namely, place-clamp-fasten-release (PCFR) to finish the final assembly product. The multiple variation sources in the MAS interact and accumulate in a non-additive manner.
- (x) *Interpretability* as the estimations of RCs will require insights into why the deep learning model made such estimates. Such interpretability insights are essential for contextualizing the RC estimated by the model. Additionally, the RC estimates will drive costly corrective actions; hence, model interpretability integrated with measures of uncertainty [15] are crucial requirements for effective CAs.

After the RCs are isolated in the next step, the CAs are determined. For performing CAs of the isolated RCs in MAS, currently, three groups of approaches can be taken into consideration. These approaches include:

(1) mapping manufacturing data to RC using mathematical models ($f^{RC}(\cdot)$) and then mapping RC to CAs based on human experience and expertise. The subset of process parameters y is identified as fault(s) and isolated as RC(s); and then, in the next step, CAs are generated as the quantitative negative of the RC ($CA = -RC = \Delta y_{RC=f^{RC}(y)}$);

(2) mapping manufacturing data directly to CAs using engineering feedback control models such as *statistical process control/automatic process control* (SPC/APC) that aim to eliminate process abnormalities (special causes) rather than to explicitly eliminate quality defects ($CA = \Delta y_{feedback}$); and,

(3) mapping manufacturing data directly to CAs by manually monitoring key product characteristics (KPCs) and implementing CAs as a change of process parameters affecting the key product characteristics which fail to comply with design requirements ($CA = \Delta y_{KPC}$).

The first group of approaches ($CA = -RC = \Delta y_{RC=f^{RC}(y)}$) estimate RCs using statistical models [42], based on quality data collected from the MAS. Human experts then use the outputs of these models to determine the CAs. The effectiveness of CAs for such approaches depends considerably on the *accuracy* and *fidelity* of the used RC isolation model. Accuracy can be measured as the ability of the model to discriminate between different combinations of RCs and *fidelity* can be measured as the ability of the model to precisely estimate the magnitude of the isolated RC. Previously, RC models leveraged a state space-based stream-of-variation model [51] of the MAS. However, the RC models have had limited applicability to complex high dimensional and nonlinear systems [72] due to the use of linear models between process parameters and measurements of dimensional product quality [73]. Currently, the CAs corresponding to the RCs are generated by human experts by assessing the estimated RCs against the MAS's constraints and stochasticity [71][74]. Mathematically, these CAs can be quantified as the negative of the mean value of RC distribution ($CA = -RC$). In certain cases, such CAs might not be feasible for implementation due to time constraints of available maintenance time window. Moreover, even in cases where

implementation is feasible, nonetheless, the cost consideration can be exorbitant particularly for MAS with limited reconfigurability. On the whole, the implementation of manually generated CAs might be ineffective due to the inherent level of stochasticity in the MAS caused by non-ideal part variations generated from upstream processes such as stamping, extrusions or/and casting. In case of stamping process, this inherent level of stochasticity is further magnified as the magnitude and pattern of incoming parts variations change between subsequent batches of parts (batch-to-batch variation in stamping). Hence, a manually determined CA, derived based on the experience of a previous batch, can fail if applied to subsequent batches with different variation patterns thereby, resulting in increased scrap and diminished productivity.

The *second group* of approaches ($CA = \Delta y_{feedback}$) determine CAs of quality defects and involve integrating SPC/APC approaches collectively known as engineering feedback control [75]–[79]. The SPC/APC approaches enable automatic CA generation; however, they assume that throughout the NPI and then production lifecycle, the *design nominals* of process parameters would continue to provide the best product quality and process performance in terms of costs and productivity. Hence, they generate CAs by attempting to revert any process parameter with special causes to the *design nominal*. The CAs are then given as feedback to process controllers, or as a maintenance task involving faulty equipment [80][81]. Thus, the SPC/APC methods aim to control special causes by identifying process parameters that have temporal patterns such as mean-shift or heteroskedasticity and/or are beyond the pre-specified control limits. As a result, these methods aim to correct the MAS by reverting process parameters with *special causes (assignable causes* [82]) rather than accounting for dynamically changing stochasticity, constraints and KPIs of the MAS. Various studies have shown that as MAS progresses through the NPI and production lifecycle, the design nominals and specified tolerances no longer provide optimal key performance indicators (KPIs) values [71] [74]. Moreover, Mannar et al. [71] observed that a number of product warranty faults occur when all the key process parameters are within the design tolerance windows. Such faults are often classified as no-fault-found/no-defect-found phenomenon, i.e., a product/subassembly is reported as faulty, however, the external tests do not detect any fault.

The no-fault-found phenomenon might be caused by issues in the design stage such as inaccurate design nominals and/or imprecise tolerance windows of key parameters (assembly process design). They may also be caused due to issues within the production stage such as imprecise deterioration of the production equipment (maintenance plan) or/and inaccurate and/or imprecise incoming parts (fabrication process). Mannar et al. [71] developed the Fault Region Localization (FRL) methodology to address no-fault-found issues by identifying and localising quality faults even if they occur inside the tolerance window caused by imprecise design tolerance boundaries. The FRL methodology combines warranty and production data to identify and localise in-tolerance faulty regions, i.e., low-yield regions within the tolerance. Additionally, Mannar et al. [74] proposed re-evaluating the manufacturing acceptance criteria or re-evaluating product tolerances to prevent delivering to customers defective products that fall into the in-tolerance faulty regions. Overall, the inaccurate design nominals and imprecise tolerance windows of key parameters can be the result of unmodelled interactions during product and process design given the limitations of simulation-based design methods [83].

Hence, the *second group* ($CA = -\Delta y_{feedback}$) and *third group* ($CA = -\Delta y_{y \rightarrow KPC}$) of approaches directly map data to CAs ($CA = -\Delta y$) assuming a fixed design nominal without systematically isolated RC. Then, CAs are estimated as the quantitative negative of the difference between process parameter deviation estimates and fixed design nominals. In scenarios where the design nominal does not give optimal KPI performance, the CAs determined by both categories of approaches may be ineffective in practice. This is especially critical during design and pre-production phases of NPI. Therefore, instead of using *design nominals* it might be necessary to estimate *functional nominal* for determining effective CAs based on the current stochasticity and interactions within a given MAS. *Functional nominals* can be interpreted as the values of process parameters that ensure the lowest process fallout rate [74] or/and maximize KPIs under the current MAS constraints and stochasticity.

As the aforesaid discussion underscores that CAs determined by the three groups of approaches may be inadequate in industrial applications of MAS. Therefore,

determining CAs within MAS should account for various requirements. These requirements are specifically addressing CA and they are on top of the discussed earlier (i)-(x) requirements for isolation of RCs. Therefore, the requirements for determining CAs are numbered consecutively from (xi)-(xiv) as below. Additionally, depending on the origin of the requirement, they are classified into *RC related* and *MAS related*. RC related requirements are induced due to considerations such as the estimated uncertainty and impact of RCs on KPIs, while MAS related requirements are induced due to the attributes of the MAS, such as constraints and stochasticity.

RC related requirements for determination of CAs include:

- (xi) *Uncertainty of isolated RCs*. The proposed RC isolation approaches account for various requirements (i)-(x). Under such requirements, RCs are uncertain, and hence, they are estimated as probability distributions that need to be taken into consideration when determining CA. The CA cannot be determined by leveraging only the mean of the RC distribution as it might lead to bias and be *ineffective* in practice for assembly processes with non-ideal parts and in scenarios involving multiple collinear defects. For example, in case of new e-mobility manufacturing, i.e., battery cell/module/pack assembly using emerging joining processes such as remote laser welding (RLW) [84], part-to-part gap variation affects the quality of welded joints and can be caused either due to fixture-induced errors or/and incoming part sudden variation change. The part-to-part gap variation must be corrected at each point of the welded seam and cannot be determined based on the mean of the part-to-part gap (RC). The RCs can be corrected for small gap variation by laser welding power adjustment as CA. Additionally, however, it requires addressing sudden variation change of incoming parts (as CA) if part-to-part variation is relatively large (>50% of part thickness).
- (xii) *KPI improvement* [85] as CAs should eliminate RCs subject to cost-benefit analysis, i.e., assessment of the CA impact on the final KPIs as related to cost and quality. Certain MAS may be fully automated, allowing for quick CA implementation without any major impact on the production schedule, while some MAS may be manually controlled, resulting in increased downtime and productivity. In such scenarios, an easily implementable CA can be used to continue

production till the next maintenance schedule but might result in increased scrap hence resulting in high cost of quality. Alternatively, implementing a more effective but time-consuming *infeasible* CA will need increased production downtime and will result in decreased productivity hence increasing the cost of production. Therefore, the final choice between multiple CAs that can be implemented is subject to a cost-benefit analysis while considering the KPIs of quality.

MAS related requirements for determination of CAs include:

- (xiii) *MAS design architecture* [71] can result in certain CAs being infeasible to implement in a given MAS because of process design and MAS reconfigurability limitations that constrain certain degrees of freedom for process parameter adjustments. Additionally, high costs or restricted time and frequency of CA implementation also limit the set of feasible CAs. For instance, certain CAs requires a long duration for implementation, but only a limited time window may be available for regular maintenance and CA implementation. These limitations need to be considered in the CA methodology as they might create scenarios wherein the implementation of a generated CA to eliminate a given RC may be *unjustifiable* in terms of cost-benefits analysis as in (xii).
- (xiv) *MAS inherent stochasticity* [9] as caused by incoming part variations (i.e., non-ideal parts due to inherent variation of upstream processes such as stamping, extrusions, etc.) and tooling variation (i.e., tooling calibration error, repeatability and reproducibility) might lead to certain CAs being *ineffective* in practice.

1.3 Research Kernels and Roadmap

Overall, the above challenges (i)-(xiv) related to RC isolation and CA estimation need to be overcome on two levels of modelling:

- (i) *forward analysis model* (Fig. 1.2) for the variation propagation in a forward manner using high fidelity CAE simulations (estimate shape error given process parameters). This is done by leveraging the Object Shape Error Modelling (OSEM) kernel (Fig. 1.2) that utilizes high fidelity multi-physics simulator of the MAS, called Variation

Response Method (VRM) [6]. The VRM can model and simulate the assembly process with non-ideal compliant parts constrained by assembly fixtures and part-to-part interactions. It also enables high-fidelity point-cloud data generation of 3D assembled products/subassemblies with error patterns obtained under different sets of process parameters. The VRM model accuracy was verified and validated for various assembly processes [6]. The OSEM kernel leverages high-fidelity VRM simulations to model the propagation of shape error as objects move through the MAS. The MAS consists of assembly stations, and each station consists of multiple stages, namely, positioning, clamping, fastening (joining) and release (PCFR). Each stage consists of process parameters that control various operations done in these stages. The system has non-ideal deformable parts (objects) that behave as free form surfaces. As these objects move through various stages and stations, various operations controlled by the process parameters are performed within the MAS. Additionally, the objects also interact with each other through part-to-part interactions. This leads to shape error in the object and is accumulated in a non-additive way causing the system to be inherently non-linear. OSEM aims to model this propagation of object shape error in MAS using Finite Element Analysis (FEA) and Variation Simulation Analysis (VSA) [86] based VRM simulator. Overall, OSEM, as enabled by VRM, can be considered as a Computer-Aided Engineering (CAE) based forward propagation model of the MAS and is leveraged for applications such as design optimization, tolerance analysis, process capability improvement and variation reduction [87].

(ii) *backward synthesis models* (Fig. 1.2) for the backward analysis of MAS that can relate shape errors backwards chaining to abnormal process parameters while fulfilling the aforementioned requirements (i)-(xiv). These models are essential for RC isolation and CA estimation applications to ensure that all quality requirements for the given products are met.

The overall research objectives of this thesis consist of formulating, developing and validating kernels, Object Shape Error Response (OSER), Object Shape Error Response for Multi-Station Assembly Systems (OSER-MAS), Closed-loop In Process (CLIP) and Object Shape Error Correction (OSEC). These kernels collectively aim to build synthesis approaches while ensuring all the aforementioned requirements are met.

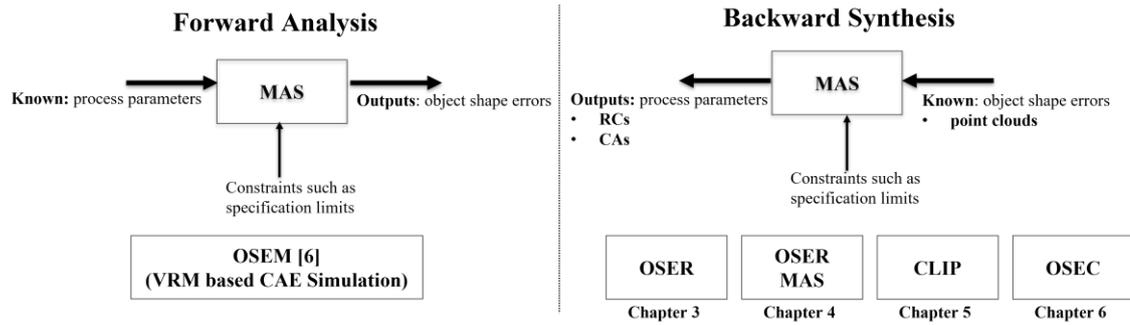


Fig. 1.2. Analysis and synthesis of MAS

The kernels and requirements fulfilled by each kernel are described as follows:

(1) *Object Shape Error Response (OSER)*: The OSER kernel aims to estimate a subset of real-valued process parameters related to positioning (P) and clamping (C) stages based on the object shape error of a single assembly station output. The OSER leverages Bayesian 3D Convolutional Neural Networks (CNN) trained in a closed-loop sampling and training framework based on the VRM data. This integration of a probabilistic neural network and CAE simulation provides a step towards understanding causality and having a safer, self-aware and interpretable solution, essential for RCA solutions in industrial setups. The kernel aims to fulfil a subset of the requirements as needed for a *single station assembly system* as follows:

- Requirements (i)-(iv) by developing a 3D deep learning approach. This kernel proposes a 3D CNN architecture that enables the extraction of spatial features from point cloud quality monitoring data representing object shape (dimensional and geometric) errors and models non-linear relationships between the extracted shape error features and process parameters. This approach has high performance for non-linear and ill-conditioned systems having high fault multiplicity.
- Requirement (v) by leveraging a Bayesian deep learning approach compared to traditional CNNs with deterministic weights. Bayesian CNNs leverage probability distributions over model weights and model outputs. The proposed approach estimates each model parameter as a distribution (epistemic uncertainty) while also modelling the output estimates as parameters of a multivariate distribution (aleatoric uncertainty). Such estimates involving the different types of uncertainties in model predictions

are crucial as they quantify when the model is 'randomly guessing' compared to making a confident prediction. Particularly within manufacturing environments, these uncertainty estimates integrate a degree of confidence within the estimates and hence, support the decision-maker in making a cost-effective selection of costly CAs.

- Requirement (vi) by making the developed Bayesian 3D CNN compatible with point cloud data obtained via either VRM or 3D scanners

(2) *Object Shape Error Response for Multi-station Assembly System (OSER-MAS)*: The OSER-MAS kernel builds on the capabilities of the OSER kernel to extend the scope of RC isolation from single- to multi-station assembly systems by extending the capabilities of the OSER 3D CNN architecture. The OSER-MAS can be leveraged for RC isolation of both single and multi-station assembly systems. It aims to estimate a heterogeneous set of process parameters consisting of real-valued process parameters related to non-ideal part variations, positioning (P), clamping (C) and fastening (F) stages and binary process parameters indicating the success of a joining in the fastening (F) stage. Additionally, the kernel also estimates object shape errors at the end of upstream stations which is critical to localize the assembly station at fault. The OSER-MAS leverages a Bayesian 3D U-Net trained in a closed-loop sampling and training framework based on the VRM. The Bayesian 3D U-Net is built based on the validation of 3D convolution capabilities provided by the OSER kernel. The kernel aims to fulfil a subset of the requirements as required for *MAS* as follows:

- Requirements (i)-(vi) by proposing a 3D fully convolutional 3D U-Net [88] based encoder-decoder network architecture. The encoder consists of probabilistic Bayesian Flipout layers [89], while the decoder consists of attention-based [90] layers. Overall, the model has three output heads, two at the end of the encoder and one at the end of the decoder. The model is trained using dual data generation capabilities by utilizing high fidelity VRM simulator of the assembly process [6] and 3D Optical scanners.
- Requirement (vii) by leveraging two output heads at the end of the encoder, one head estimates real-valued continuous process parameters as done in a

regression setting (for example, part variation and fixturing RCs). In contrast, the second head estimates categorical/binary process parameters as done in a multi-label classification setting (for example, SPR joining process RCs).

- Requirement (viii) by leveraging the proposed 3D U-Net-based architecture decoder to provide segmentation maps corresponding to the object shape error for previous (upstream) stations. Volumetric segmentation has seen immense development in the past few years. The initially proposed 2D U-Net architecture has been extended for 3D applications such as MRI images and kidney scans [91]. In U-Nets, 3D volume is mapped to a latent space via successive convolutional and pooling layers; this latent representation is then upsampled and convolved until it reaches the original volume's size generates a per-voxel segmentation. The proposed network extends this capability to estimate object shape errors for upstream stations of the MAS. Further, a methodology to perform RC isolation is proposed that leverages the upstream object shape error estimates for fault localization (i.e., determination of candidate station with potential RC) and the process parameters for RC isolation.

(3) *Closed-loop In-process (CLIP)*: The CLIP kernel aims to provide an approach to enable large-scale industrial adoption of proposed RC isolation approaches (OSER/OSER-MAS) by firstly enhancing the *scalability* as training deep learning models for each MAS is time-intensive and requires large amounts of labelled data and secondly providing *interpretability* and transparency for the estimated RCs that drive costly corrective actions (CAs) by integrating a CLIP underpinned *algorithm portfolio*. Scalability is enabled by closed-loop sampling and training, uncertainty guided continual learning [92] and transfer learning [29], enabling efficient and effective utilization of data provided by VRM. Interpretability is enabled by 3D gradient-based class activation maps [93] (3D Grad-CAMs) that are interpolated on the Computer-Aided Design (CAD) model leveraged in VRM, providing context on the values of process parameters estimated by the model. The kernel aims to fulfil a subset of the requirements as required for single/multi-station assembly

systems as follows:

- Requirement (ix) by developing a closed-loop training framework that leverages the epistemic uncertainty [15] estimates of the Bayesian 3D CNN based OSER or OSER-MAS model to intelligently sample from the process parameter hyperspace for faster convergence and hence, reduce the computation time bottleneck of the VRM simulations. Further, to exponentially enhance the scalability for high dimensional MAS and reduce VRM simulation time, a combined uncertainty guided continual learning [92] and transfer learning [29] algorithm portfolio are integrated with the closed-loop training framework. This enables the transfer of meta-knowledge from the trained model to a new MAS, and thus, each new MAS requires comparatively fewer training samples. Theoretically, given that the assembly system's multi-physics processes are similar within each station, the features extracted by spatial convolutional operations are transferable across different assemblies; thus, making transferability within the model essential for scalability. Given that models for different MAS would be trained sequentially, leveraging continual approaches reduces catastrophic forgetting. A model with continual learning capabilities can also account for the assembly system's dynamic nature and achieve lifelong learning. This is accomplished using uncertainty-guided continual learning [65] that leverages the Bayesian neural network parameter uncertainty to assign importance for each task, i.e., a particular assembly case study, thereby, enabling continual learning by updating less important parameters at a faster rate.
- Requirement (x) by developing an interpretability model which is based on 3D Gradient-weighted Class Activation Maps (3D Grad-CAMs) and entails: (a) linking elements of MAS model with functional elements of the 3D Bayesian U-Net model; and (b) leveraging 3D Grad-CAMs [93] that provide insights into the regions within the input that the model focuses on to estimate process parameters, i.e. RCs. These collectively provide the required interpretability for RCA.

(4) *Object Shape Error Correction (OSEC)*: The OSEC kernel aims to build an approach that can generate feasible and effective CAs. Within the approach, high-dimensional manufacturing data collected at the end of the MAS is first transformed to RCs, and then RCs are transformed into CAs. The proposed approach enables correction of object shape error quality defects for MAS with non-ideal parts. It leverages DRL, specifically deep deterministic policy gradient (DDPG) method, to optimize a novel reward function resulting in CA policies that generate sequential process parameter adjustments. A CA policy can be interpreted as a sequence (also referred to as an episode) of corrective adjustments of process parameters (CAs) that eventually lead to RC elimination and KPI maximization. The approach can be leveraged to generate CAs when the CAs estimated by current techniques cannot be directly transformed to CAs ($CA \neq -RC, CA \neq -\Delta y$). This is because the RCs map to CAs, which might be infeasible to implement due to KPI improvement and MAS constraints or rendered ineffective after implementation due to RC uncertainty or MAS stochasticity. The approach also provides capabilities (Fig. 1.3) to estimate alternative CAs for those classes of CAs that are infeasible or potentially ineffective due to NPI lifecycle requirements or assumptions such as a fixed design nominal. Where such a class of CAs does not exist, the CAs estimated by the planned approach correspond to the CAs estimated by current approaches ($CA = -RC, CA = -\Delta y$) which assume that all CAs are implementable, and the design nominal always maximizes KPI performance.

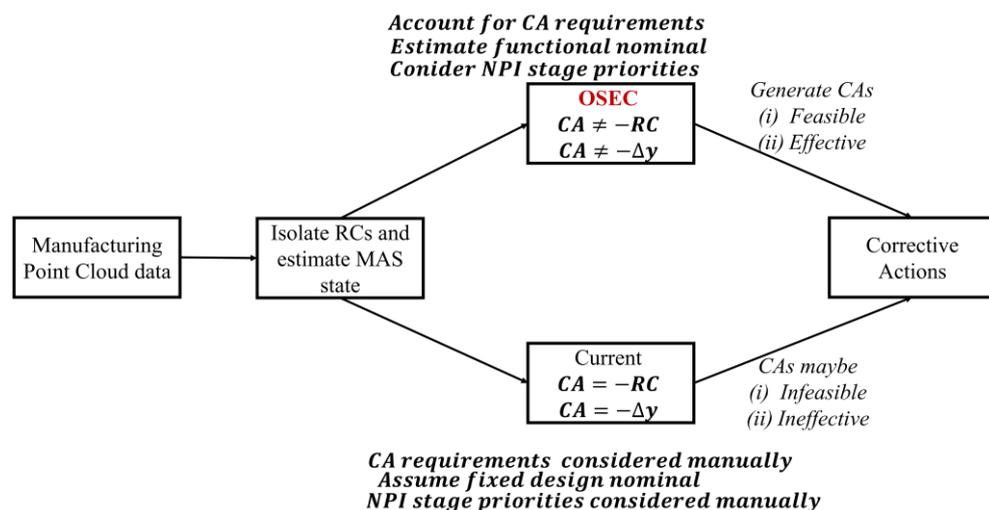


Fig. 1.3. OSEC vs current approaches

Overall the OSEC kernel aims to fulfil a subset of the requirements (xi, xii, xiii and xiv) as required for correction of single/multi-station assembly systems as follows:

- Formulating MAS correction as a Markov Decision Process (MDP) and proposing a *deep reinforcement learning (DRL) approach that can estimate CAs (actions) based on the RCs (state) provided as inputs while accounting for the four CA requirements* (reward) (xi) Uncertainty of isolated RCs, (xii) KPI improvement, (xiii) MAS design architecture and (xiv) MAS inherent stochasticity. This approach is developed with end-to-end capability, i.e., combines both estimation of: (i) RCs; and (ii) CAs. For this, first, high dimensional point cloud data which are collected at the end of the MAS (end-of-line sensing), are used as input to RC models (such as OSER and OSER-MAS developed in this thesis). Next, the isolated RCs are used as input to CA methodology to estimate optimal CAs as the final outputs.
- Leveraging *Deep Deterministic Policy Gradients (DDPG)* technique as the specific DRL method. Given that the output space for CAs consists of multi-dimensional real-valued process parameters adjustments, traditionally used action-value based methods cannot be used as they are limited to work on discrete action spaces [37]. Therefore, the proposed approach leverages DDPG as this can map states (RCs) to multi-dimensional real-valued actions (CAs) by approximating both the CA policy (i.e., sequence of CAs) and CA value function (i.e., overall reward of taking a CA) using deep neural networks. In the proposed CA approach, OSER is leveraged as the RC isolation model to estimate the state of the MAS, which is given as input to the DDPG agent. DDPG agent is constituted as the combination of the CA policy and CA value function, which takes as input the RCs and estimates the CA with maximum value.
- Training DDPG agent as stated above with *a novel reward function* that mathematically models all (xi)-(xiv) requirements that need to be accounted for the design and implementation of CAs in MAS. The reward generates positive values (*benefits*) when RCs are eliminated, and KPIs are improved. Simultaneously, it penalizes the DDPG agent (*costs*) for generating CAs that

are infeasible due to the cost and time needed for implementation. No reward is allocated when CAs do not improve KPIs due to the stochasticity present in the MAS or in scenarios when the uncertainty in isolate RCs are high. The proposed OSEC approach does not limit CAs to adjust process parameters to the fixed design nominals; but rather, it selects the CAs to adjust process parameters to the *functional nominals in order to maximize the KPIs*. This is in contrast to current SPC/APC approaches, which assume a fixed design nominal to generate CAs that adjust values of a subset of process parameters identified as abnormal due to temporal patterns or variations beyond the specified control limits.

- Parametrizing the reward function as stated above with *NPI-coefficients* to enable the application of the proposed method throughout all NPI stages, namely design, pre-production, production launch and full production. This is especially important as the importance and priorities of the aforementioned four requirements changes during each NPI stage. At design and pre-production stages of the NPI, there are fewer constraints on cost, number of changes and no constraints related to production/maintenance schedules, however, the improvement of KPIs is crucial, and thus, the flexibility of having functional nominals instead of fixed design nominals provide the necessary flexibility and thus, frequent CAs can be implemented to improve the KPIs. On the other hand, during production stage, the MAS becomes more '*rigid*' due to increased costs of change and limited time permitted for changes (maintenance scheduled breaks). Therefore, to account for the changing trade-offs and priorities, the method proposes to parametrize the reward function with *NPI-coefficients*. The NPI-coefficients include estimates for KPI importance, costs, system reconfigurability and sample size. The user can select the NPI-coefficients for a specific NPI stage which is then used to parametrize the reward function. Overall the proposed reward function simultaneously maximises KPIs by eliminating RCs while accounting for the MAS stochasticity and ensuring that the CAs are within the allowable MAS reconfigurability for executing corrections [94].

These kernels (OSER, OSER-MAS, CLIP and OSEC) collectively provide a principled

and comprehensive way for the scalable and interpretable synthesis of single/multi-station assembly systems and ensure milestones under Industry 4.0 such as *Zero-Defect-Manufacturing*, *Right-First-Time* and *Resilient Manufacturing* can be achieved. The proposed kernels and framework go beyond just improving the accuracy and performance of current models used for isolation and mitigation of quality defects that are driven by statistical learning and manual expertise. The proposed kernels integrated within a framework provide ‘transformative’ capabilities by:

- (i) addressing a novel problem of applying Bayesian deep learning on point cloud data for object shape error identification instead of object detection.
- (ii) integrating physics-based engineering models with deep learning approaches as stated in (i).
- (iii) leveraging the integrated approach in (ii) to provide capabilities for unsolved challenges such as ill-conditioning, fault-multiplicity, RC isolation with uncertainty quantification, learning at the design phase when no measurement data are available, scalability during training and interpretability during deployment.
- (iv) mathematically formulating mitigation requirements and using deep reinforcement learning to estimate mitigation decisions based on outputs of (iii) hence reducing the need for manual expertise.

Overall, the thesis aims to identify the requirements and then develop kernels for isolating and mitigating object shape errors within MAS by addressing various research questions. The presented research questions (RQ) provide a general, concise summary of the considered scientific research problem. The research questions in this thesis are:

RQ1: What are the relevant requirements of methods to enable isolation and mitigation of object shape error defects in assembly systems?

RQ2: How can the assembly system be modelled in a backward manner (synthesis) to enable isolation of RCs while fulfilling the requirements in RQ1?

RQ3: How can the answer in RQ2 be extended for multi-station assembly systems?

RQ4: How can the scalability and interpretability of the methods for RQ2 and RQ3 be enhanced to enable wider industrial adoption?

RQ4: What methods can be developed to perform effective and feasible CAs for the RCs of shape error defects isolated using the answer to RQ2 and RQ3?

Fig. 1.4 summarises the links between the research questions and the kernels developed to address the research questions.

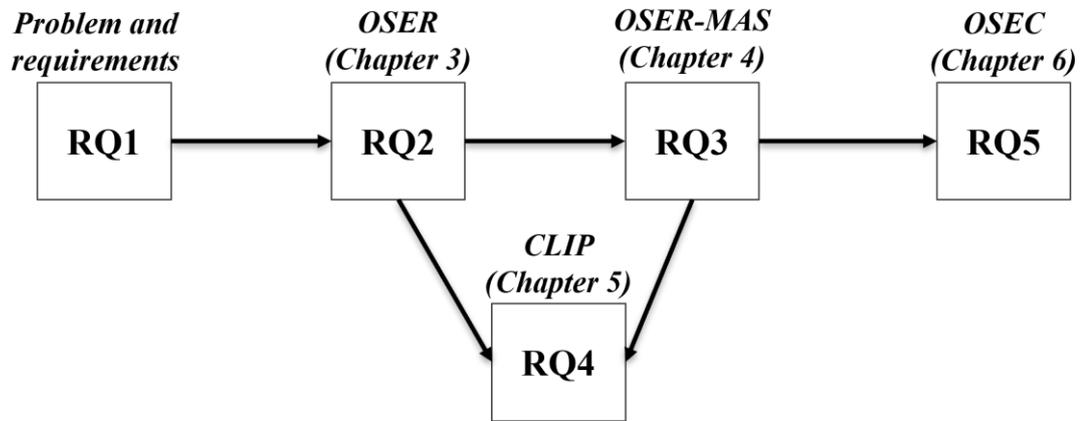


Fig. 1.4. Research questions

Table 1.1 provides a comprehensive research roadmap of problems, motivations, challenges, state-of-the-art limitations, objectives, methods and validations.

1.4 Thesis Organization

The overall thesis comprehensively describes the kernels in the proposed roadmap, their implementations, and the conclusions deduced. *In addition, the required background and the contributions for each kernel are also comprehensively discussed in the chapters.*

Chapter 2, titled "*Literature Review*", aims to close the gap between manufacturing assembly systems and deep learning and presents a review of the existing literature on manufacturing assembly systems, related work on applications of deep learning, soft sensing, Bayesian modelling and deep reinforcement learning. Finally, it outlines the knowledge gaps and links these to the aforementioned requirements.

Chapter 3, titled "*Object Shape Error Response using Bayesian 3D Convolutional Neural Networks for Assembly Systems with Compliant Parts*", discusses the OSER kernel, including problem formulation, methods, case study, conclusions and contributions.

Chapter 4, titled "*Root Cause Analysis of Multi-station Assembly Systems with Non-Ideal Compliant Parts using Bayesian 3D U-nets*", discusses the OSER-MAS

kernel, including problem formulation, methods, case study, conclusions and contributions.

Chapter 5, titled "*Building a Scalable and Interpretable Bayesian Deep Learning Framework for Quality Control of Free Form Surfaces*", discusses the CLIP kernel, including problem formulation, methods, case study, conclusions and contributions.

Chapter 6, titled "*Building a Quality Correction Framework for Assembly Systems using Deep Reinforcement Learning*", discusses the OSEC kernel, including problem formulation, methods, case study, conclusions and contributions.

Chapter 7, titled "*Advanced implementations and Industrial Applications*", discusses the software and physical implementation of the work, including an industrial demonstrator. The physical implementation validates the proposed kernels on physical data of free-form surfaces collected from 3D scanners. This chapter also summarizes various industrial applications of the proposed work

Chapter 8, titled "*Conclusions and Future Work*", summarises the kernels developed in this thesis. Further, it discusses contributions, limitations and future work.

Additional background and methods used within the thesis have been referenced and are later discussed in Appendix A. An overview of the software implementation on GitHub is included in Appendix B. The published journal papers for Chapters 3 and 5 are included in Appendix C and D, respectively. The journal papers under review for Chapters 4 and 6 are included in Appendix E and F, respectively. Appendix G includes the poster of the industrial demonstrator developed as a part of this thesis to showcase the capabilities to the industry.

Table 1.1: Research Roadmap

Problem	Motivations	Requirements	State of the art limitations	Objectives	Methodology (Kernels)	Applications	Contributions	Verification and Validation
Isolation & mitigation of object shape error defects in MAS	<i>Zero Defect Manufacturing</i>	High dimensional point cloud data from the 3D scanner	Work on sampled point data	Object shape error representation for point cloud data	OSER Kernel Object shape error formulation and representation	RCA for manufacturing processes (assembly, stamping, machining, additive manufacturing) leading to reduction of Cost of Quality	<u>Extending deep learning object detection to object shape error source estimation using 3D CNN based OSER and OSER-MAS</u>	Verification, validation & benchmarking against current RCA and machine/deep learning models for performance and convergence using case studies namely: 1. Single station door inner and hinge reinforcement assembly 2. Multi-station cross-member assembly Scalability study leveraging six case studies of different complexities Interpretability study for root causes Physical demonstrator for physical-system implementation using 3D scanner point cloud data Python-based software library named Deep Learning for Manufacturing Benchmarking using cross-member assembly case study
		Non-linearity due to compliant, non-ideal and deformable parts	Statistical multivariate models with linear approximations	Non-linear model	3D CNN architecture selection & optimization			
		Collinearities and fault multiplicity (six sigma requirements)	Consider a single fault multiple orthogonal fault	Model with high discriminative ability	Bayesian deep learning			
		Uncertainty quantification for costly corrective actions	No uncertainty quantification	Uncertainty quantification	Uncertainty based sampling using CAE simulations	Confidence in root cause estimates		
		Lack of datasets at initial design & production stages	Use limited available physical system data	Data augmentation using CAE simulations		Learning at early design stages		
	<i>Right First Time</i>	Heterogeneous set of process parameters including non-ideal part variations, fixturing and joining	Consider homogenous parameters with negligible interaction	Multi-task model	OSER-MAS Kernel 3D attention-based Bayesian U-Net architecture with multiple output heads	Simultaneous estimation of binary and six-sigma fault indicators	<u>Integration with CAE simulations for dual data generation</u>	
		Fault localization and RCA in MAS	Do not estimate shape errors at the upstream stations	Upstream shape error estimation	RCA workflow including fault identification, fault localization and fault isolation	Fault localization and RCA for MAS	<u>Bayesian learning approach for uncertainty segregation and quantification</u>	
				Estimation of heterogenous set of process parameters				
	<i>Reduction of manual expertise</i>	Scalability across various MAS	Exponential data and computational requirement for scaleup	Scalable approaches for computational feasibility	CLIP Kernel Algorithm portfolio with closed-loop training, continual learning, transfer learning and 3D gradient-based class activation maps with mesh interpolation	Efficient scaling to high-dimensional multi-station assembly processes	<u>CLIP diagnostic framework underpinned algorithm portfolio for scalability and interpretability</u>	
		Interpretability of estimated root causes	Black-box models	MAS interpretation for isolated RCs	OSEC Kernel Deep deterministic policy gradients (DDPG) agent with a reward function parameterized with NPI coefficients	Expert insights on model estimates integrating 'trust'	<u>OSEC framework for object shape error correction</u>	
Uncertainty of isolated RCs KPI improvement MAS design architecture MAS inherent stochasticity				Manual expertise-based models that assume a fixed design nominal				CAs that account for all MAS requirements
<i>Reduction of cost of quality and increased productivity</i>		Scalability across various MAS	Exponential data and computational requirement for scaleup	Scalable approaches for computational feasibility	CLIP Kernel Algorithm portfolio with closed-loop training, continual learning, transfer learning and 3D gradient-based class activation maps with mesh interpolation	Efficient scaling to high-dimensional multi-station assembly processes	<u>CLIP diagnostic framework underpinned algorithm portfolio for scalability and interpretability</u>	
	Interpretability of estimated root causes							Black-box models
<i>Reduced scrap and machine downtime</i>	Interpretability of estimated root causes	Black-box models	MAS interpretation for isolated RCs	OSEC Kernel Deep deterministic policy gradients (DDPG) agent with a reward function parameterized with NPI coefficients	Expert insights on model estimates integrating 'trust'	<u>OSEC framework for object shape error correction</u>		
							Uncertainty of isolated RCs KPI improvement MAS design architecture MAS inherent stochasticity	Manual expertise-based models that assume a fixed design nominal

2 Literature Review

The development of the aforementioned kernels in Chapter 1 requires cross-disciplinary research review across manufacturing and artificial intelligence (AI). The review should include a *deep* analysis of past work done in manufacturing assembly systems and a *wide* analysis of AI tools and successful AI applications across different areas to identify existing knowledge gaps. Hence, this chapter explores work done across diverse areas, including (1) *modelling and diagnosis of manufacturing assembly systems*, (2) *deep learning-based 3D object detection and segmentation*, (3) *deep learning applications in manufacturing*, (4) *soft sensing for industrial processes* (5) *Bayesian modelling for uncertainty quantification*, and (6) *deep reinforcement learning for control*. A review of research work is carried out for each area. Seminal research works are identified in each field, and a knowledge graph is constructed for similar papers. Connected papers [95] is used to construct the knowledge graphs, which aims to cluster similar papers by using co-citations of papers as the metric for similarity. Finally, all knowledge graphs are combined to identify *knowledge gaps* that are then linked to requirements (Section 1.2) that are addressed by the proposed kernels (Section 1.3) within the research.

2.1 Modelling, Diagnosis and Correction of Assembly Systems

Modelling, diagnosis and correction of assembly systems include various seminal works [12], [31], [50], [55], [58], [96]–[98] (in-depth analysis discussed in Section 1.2). The state-space model for compliant assembly [50] has served as one of the key works to lay the foundations for building shape error models for forward analysis and backward synthesis of assembly systems (Fig. 2.1). The forward analysis models have been further improved by integrating finite element modelling and part-to-part interactions effects [31], [99], [100]. This has also been driven by extensive development in building scalable finite element models [6], [101]–[104] that can effectively capture the multi-physics behind manufacturing processes [87] (Fig. 2.2). The backward synthesis models have been leveraged to perform tasks such as assembly diagnosis, diagnosability analysis [51], [55], [97], [105] and optimal sensor placement

studies [106]–[109]. The models have also been leveraged for tolerance analysis [110]–[112]. Fig. 2.1 summarizes the knowledge graph for key works related to modelling and diagnosis of assembly systems. *Overall, after a huge surge of works relevant to shape error analysis and synthesis in assembly systems, there has been a relevant stagnation in the field due to the limited capabilities offered by point-based data collected using coordinate measuring machines. Limitations on computation resources and lack of comprehensive datasets have been bottlenecks in building advanced data-driven models to analyse and synthesize such systems. Additionally, over the years, as the complexity of assembly systems has increased exponentially, the performance of approximately linear first-principle state-space models have diminished. Therefore, there is a need to build models that can account for the increasing complexity of assembly systems that require scalable integration of multi-physics and deep learning-based models while accounting for the limited availability of comprehensive datasets in manufacturing environments, especially during the design phase of the production cycle. Furthermore, 3D scanners are now increasingly used for data collection. These provide high resolution point cloud as compared to point-based data collected from coordinate measuring machines. Hence methods developed should fully utilize the granularity of information provided by point cloud data collected from 3D scanners.*

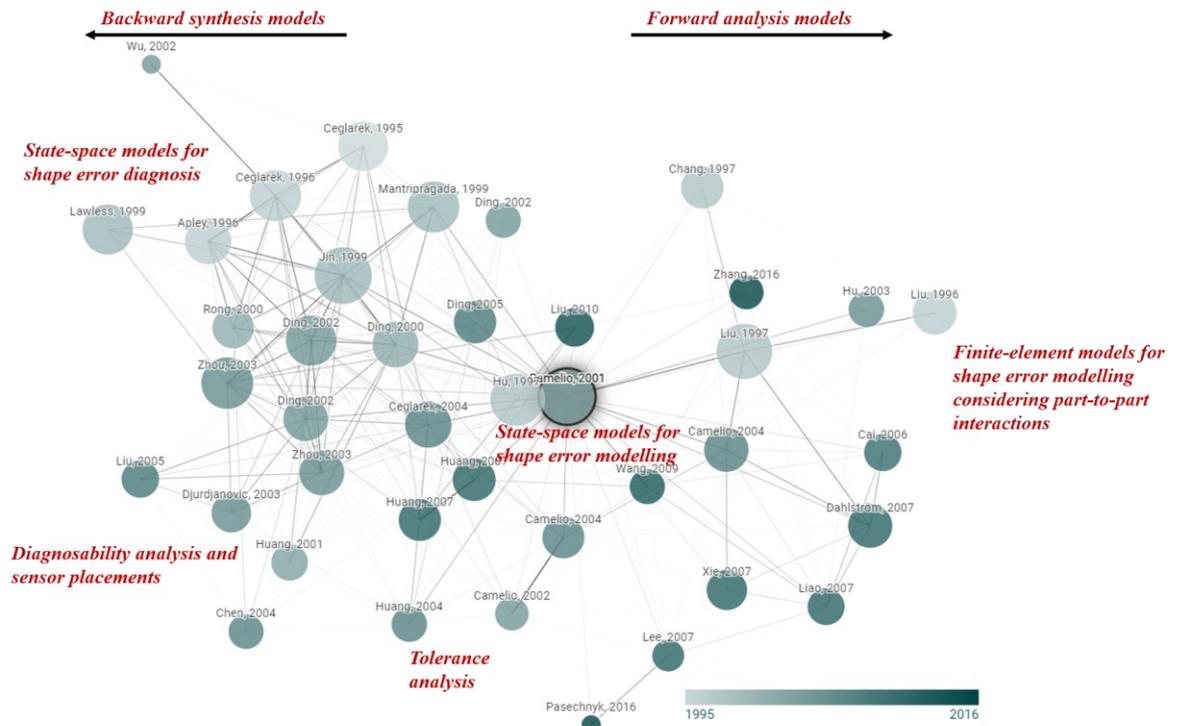


Fig. 2.1. Modelling, diagnosis and correction of assembly systems (circles denote papers; the size of the circle denotes the number of citations; the color of the circle denotes the year of publication, the links between circles shown similarity between papers)

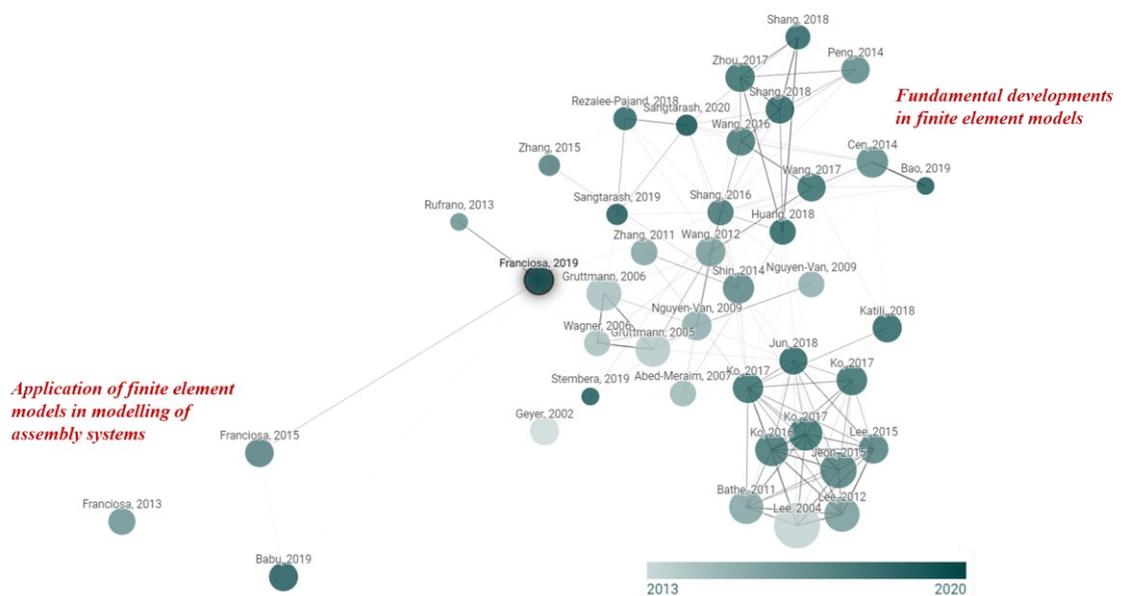


Fig. 2.2. Finite element models for assembly systems

2.2 Deep Learning for 3D Object Classification and Segmentation

Deep learning for computer vision has seen immense development over the years owing

to the availability of large datasets, GPU based computation resources and the development of CNN based architectures [113]. These enable 3D object classification, detection and segmentation using architectures specialized to work on voxelized or point cloud data [19], [20], [114]–[117] (Fig. 2.3). Further, the capabilities of such architectures to do high-level object classification and segmentation have been expanded to perform tasks requiring higher resolution, such as manufacturability analysis and shape transformation [117]–[119]. Such architectures have been leveraged to use image data to perform microstructure analysis to see the effect of material properties and process parameters [120], [121]. *There is a need to develop architectures for tasks requiring estimation of shape errors on objects (manufactured products) instead of traditional object detection. This enables isolation and mitigation of dimensional quality defects in manufacturing assembly systems using point cloud data obtained from 3D scanners.*



Fig. 2.3. Deep learning-based 3D object classification and segmentation

2.3 Deep Learning Applications in Manufacturing Fault Diagnosis

Deep learning has seen various applications for building fault diagnosis models within manufacturing [25], [122]–[125] (Fig. 2.4). CNNs have been used to analyse high dimensional data both temporally [126], [127] using 1D convolutions on time series data and spatially [128]–[130] using 2D/3D convolutions on images and point clouds data. Convolution operations enable the extraction of relevant features from the high

dimensional data input to estimate various categories of faults. The availability of industrial datasets such as motor bearing and locomotive datasets has aided in validating such approaches [131]. To further increase performance, research has also looked into using transfer learning [29], [132]–[135] from models trained on much larger datasets such as ImageNet [136]. This has allowed for extensive fault diagnosis applications in industrial environments with relatively small datasets. The semiconductor manufacturing industry (Fig. 2.5), which initially leveraged traditional spatial filtering and clustering models [137]–[140], has seen extensive applications of deep learning models [141]–[144] that leverage CNN based image classification and segmentation model to detect and classify semiconductor manufacturing defects. This has led to an overall 30% increase in the yield [142] due to higher productivity and lower scrap. Additive manufacturing is another area that has seen an extensive application of machine learning and deep learning algorithms for applications, including model design, in situ monitoring, and quality defect detection and evaluation [145] (Fig. 2.6). The previously used models for prototyping and 3D printing [146], [147] have been improved by integrating these approaches with machine/deep learning [148]–[151]. *The discussions indicate extensive applications of deep learning models in semiconductors and additive manufacturing for fault diagnosis applications. Further, the impact of using deep learning models and techniques such as transfer learning for fault diagnosis and correction in assembly systems in a scalable and interpretable manner needs exploration.*

2.4 Soft Sensing for Industrial Processes

Various key quality variables in industrial systems are difficult to measure online due to various reasons such as costs, access constraints and severe measuring environments [152]. Soft sensing provides an alternative by developing mathematical models that relate easily measurable variables to quality variables. Estimating these quality variables enables further monitoring, isolation and mitigation of quality-related root causes within the process. Initially, techniques (Fig. 2.7) such as latent factor analysis [153]–[155] and variants of principal component analysis [153], [156], [157] were leveraged to develop non-linear models that take into account the temporal dynamics within the process. The ability of deep learning models to extract high-level output

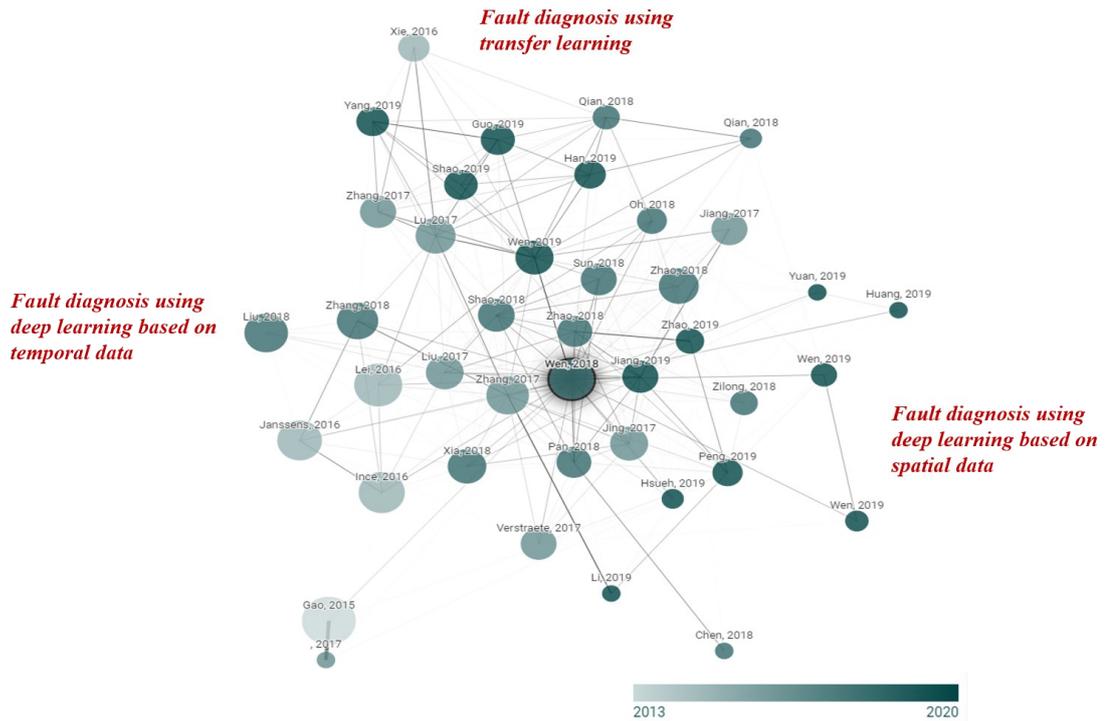


Fig. 2.4. Deep learning applications in manufacturing fault diagnosis

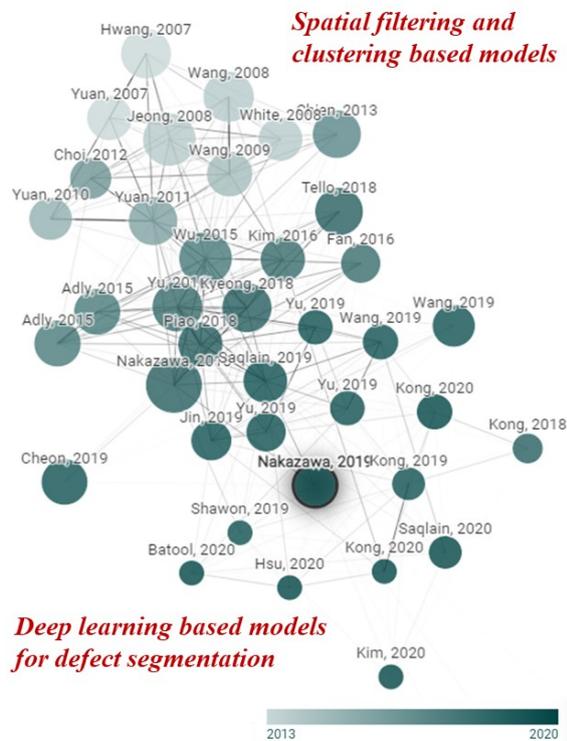


Fig. 2.5. Deep learning for semiconductor defect detection and classification

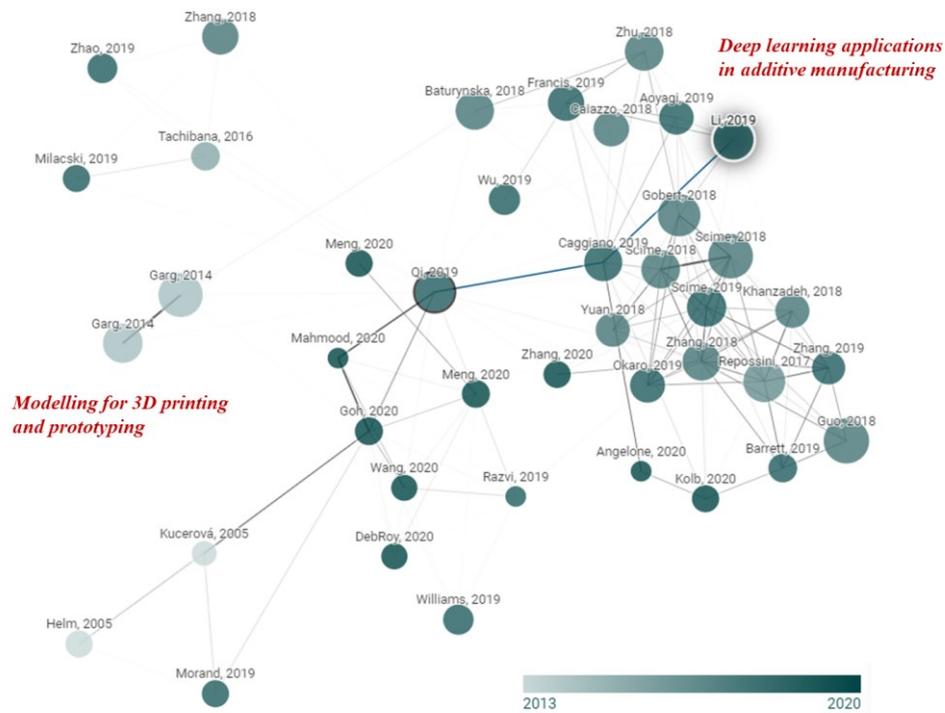


Fig. 2.6. Deep learning for additive manufacturing design and quality evaluation

related features from high dimensional data makes them an ideal choice for developing soft sensing models. Various approaches leveraging variants of deep auto-encoders [152], [158]–[161] have been proposed. The capability of neural networks to extract relevant features and approximate non-linear functions enables deep learning-based soft sensing models' superior performance. Although these models excel at feature extraction, they cannot be directly used for RCA applications. That would require them to have capabilities for causality analysis so that RCs can be isolated within the system. *Therefore, there is a need to extend soft sensing models to estimate process parameters that can be leveraged to isolate and mitigate RCs in assembly systems. In the context of manufacturing assembly systems, soft-sensing approaches need to be developed that can relate easily measurable point cloud data to quality-relevant process parameters related to non-ideal part variations, fixturing and joining.*

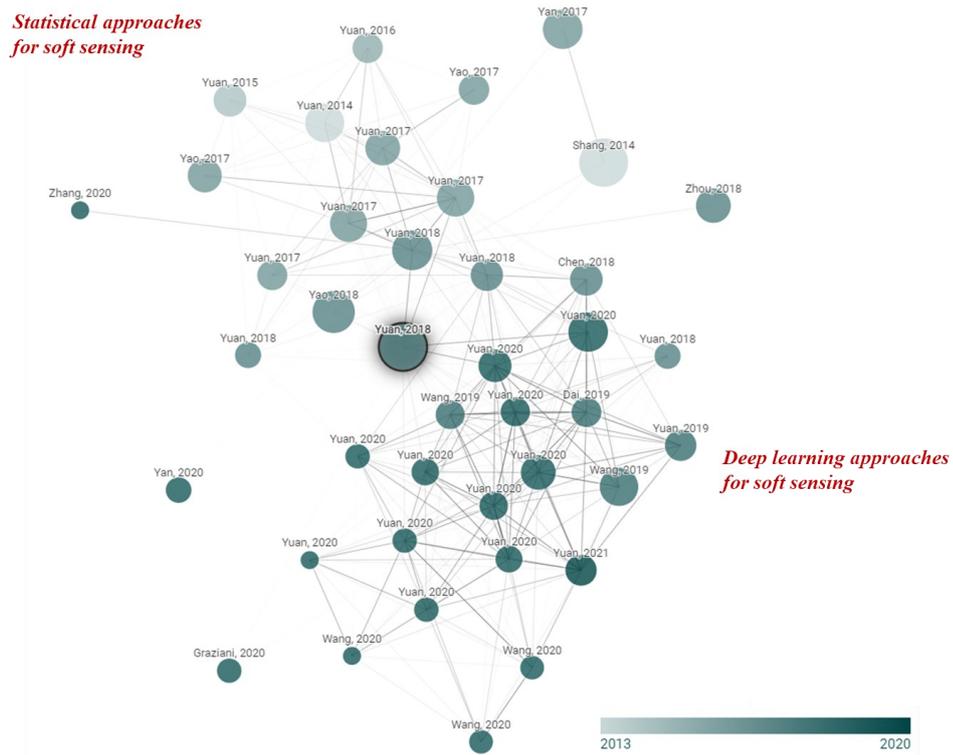


Fig. 2.7. Soft sensing for industrial processes

2.5 Bayesian Modelling for Uncertainty Quantification

Although very effective in extracting relevant information from high dimensional data and relating them to the desired outputs, deep learning cannot quantify the uncertainty while developing such complex relationships, which often results in the model giving overconfident predictions for out-of-distribution samples [162] [34]. Alternatively, Bayesian modelling provides a principled way to estimate uncertainty in the predictions by combining prior beliefs of a relationship with data likelihood to estimate the predictive posterior using Bayes theorem [163]. Recently much work has been done on combining deep learning with uncertainty modelling [34]. This provides a way to provide uncertainty estimates by modelling the weights with neural networks and the network outputs as probability distributions instead of point estimates [164]–[166]. This has been applied for estimating epistemic and aleatoric uncertainties in computer vision [36], [167], [168] (Fig. 2.8). *Such calibrated uncertainty estimates are crucial in RC isolation as they drive costly corrective actions. Therefore, models used for isolation need to be integrated with capabilities for uncertainty quantification. The uncertainty quantification techniques need to account for uncertainties due to system*

collinearities, model structure, limited data availability and sensor noise. This augments isolated RCs with calibrated confidence estimates.

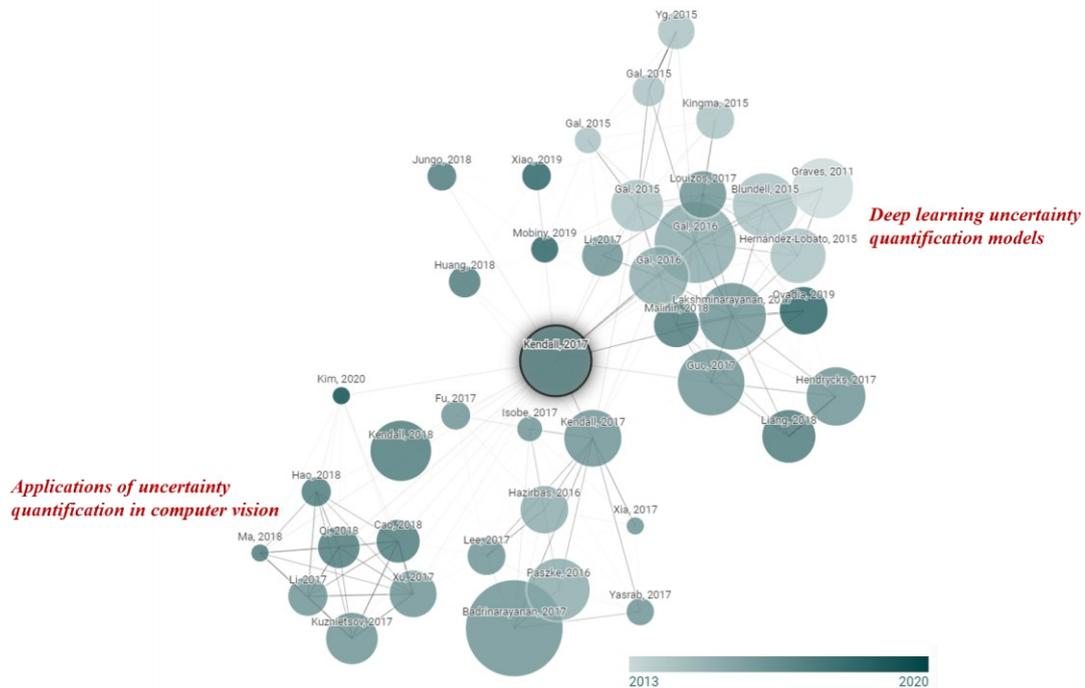


Fig. 2.8. Bayesian modelling for uncertainty quantification

2.6 Deep Reinforcement Learning for Control

Reinforcement learning has performed well in various complex sequential tasks by training agents to optimize their control in an environment to obtain the maximum amount of reward. Deep reinforcement learning integrates reinforcement learning concepts with deep learning by leveraging neural networks as effective function approximators. These approximators enable accurate estimation of value functions [37] or policy functions [169]. Various developments such as building better representations of value/policy functions and using target networks to stabilize training have allowed scaling of deep reinforcement learning for complex tasks [169]–[174]. Control of various processes and systems [175]–[179] has been the major area for application for reinforcement learning (Fig. 2.9). Deep reinforcement learning enables taking efficient control actions of various process parameters based on high-dimensional state estimates of the system. *The applications of these developments for correction of MAS have not been explored yet. A deep reinforcement learning framework is required that accounts*

for isolated root causes, MAS constraints and stochasticity to make optimal decisions on which process parameters to change within the MAS to mitigate object shape errors.

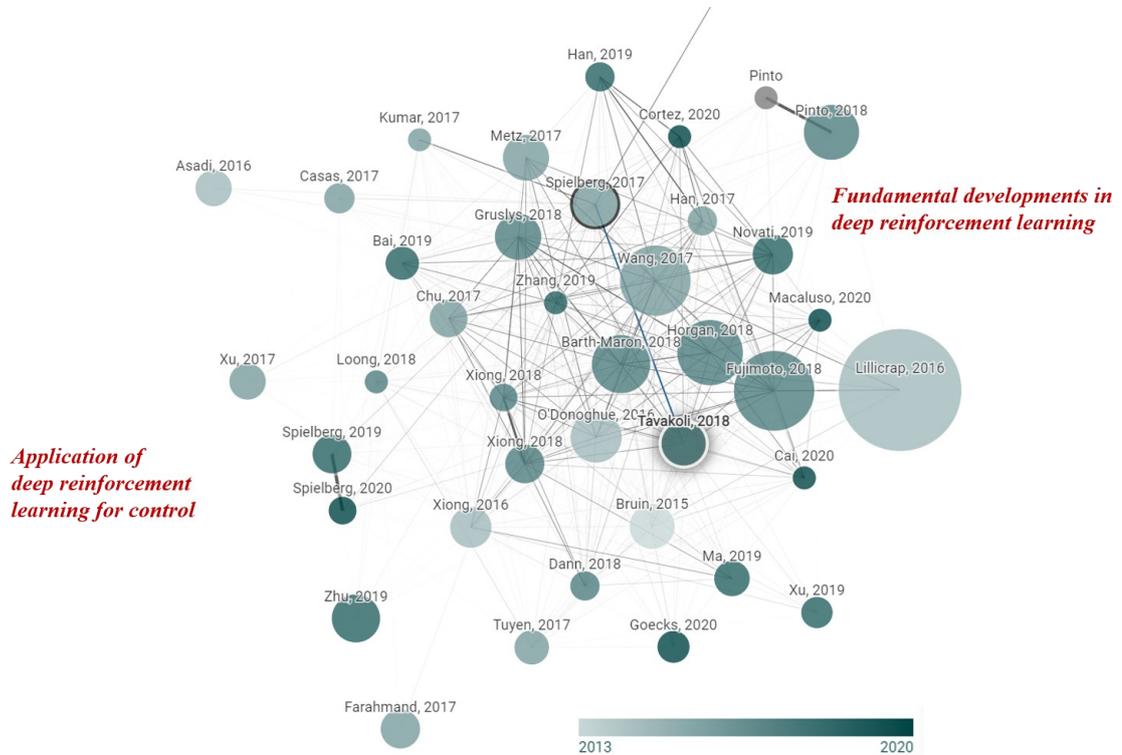


Fig. 2.9. Deep reinforcement learning for control

2.7 Literature Assimilation and Knowledge Gaps Identification

The discussions of the research areas are aggregated to identify the knowledge gaps addressed by the research (Fig. 2.10). The knowledge gaps are discussed in detail below. These are also linked with the requirements as outlined in Section 1.2:

- (1) The impact of the application of deep learning methodologies for the synthesis of MAS needs comprehensive research. Existing deep learning methodologies developed for low-resolution shape analysis (object detection, manufacturability analysis) need to be further developed to perform shape error analysis (object shape error estimation). The current capabilities of deep learning models to perform 3D object classification and segmentation must be further developed to estimate shape error on assembly objects and then relate them to RCs within MAS. These developed capabilities will address the current limitations of approaches used to isolate sources of dimensional quality defects within MAS and similar

manufacturing processes such as stamping, machining and additive manufacturing involving shape error analysis. This will enable the fulfilment of requirements (as described earlier in Section 1.2), namely (i) *High data dimensionality*, (ii) *Non-linearity*, (iii) *Collinearities*, (iv) *High faults multiplicity*, (vii) *High dimensionality and heterogeneity of process parameters* and (viii) *Fault localization*. The research addresses these requirements by developing the OSER and OSER-MAS kernels with deep learning capabilities.

- (2) Multi-physics CAE simulations have seen advancement and are currently used for forward analysis of MAS by integrating such models with machine learning-based surrogate models. The integration of such multi-physics simulators with deep learning approaches to build *synthesis* models is required to enable isolation of shape errors by RCs from the design phase itself. These developed capabilities will enable efficient training of the developed deep learning approaches and will drive towards determining causality of the identified defects. This will enable the fulfilment of requirement, namely (vi) *Dual data generation capability*. The research addresses this requirement by developing the OSER and OSER-MAS kernels with closed-loop sampling and training using a multi-physics CAE model known as VRM.
- (3) Principled approaches for uncertainty quantification during the RCA of MAS need to be explored. Capabilities of using Bayesian deep learning for uncertainty quantification and segregation need to be developed for to quantify uncertainty in isolated RCs. These developed capabilities quantify uncertainties due to measurement noise, limited data availability and further enable efficient sampling from CAE simulators. This will enable the fulfilment of requirement, namely (v) *Uncertainty quantification*. The research addresses this requirement by developing the OSER and OSER-MAS kernels with uncertainty quantification capabilities by leveraging Bayesian deep learning.
- (4) Successful industrial application of deep learning-based fault diagnosis approaches in semiconductor and additive manufacturing industries and various soft sensing approaches for indirect monitoring of quality relevant variables need to be explored to build an intelligent integrated algorithm portfolio enabling scalability and interpretability of approaches that enable isolation and mitigation of shape errors

within MAS. These capabilities will enable scalability for multi-variety products and large-scale multi-station assembly systems and further enable interpretability for transparency of RCs to drive costly CAs. This will enable the fulfilment of requirements, namely *(ix) Scalability and (x) Interpretability*. *The research addresses these requirements by developing the CLIP kernel underpinned by an algorithm portfolio with capabilities to enhance OSER and OSER-MAS kernels' scalability and interpretability.*

- (5) Correction of shape errors within MAS has been done by SPC/APC methods or manual approaches. However, given the uncertainties in isolated RCs, costs and constraints of implementing CAs, and stochasticity of MAS might lead to CAs being ineffective and infeasible in practice. Therefore, deep reinforcement learning capabilities need to be integrated with MAS engineering knowledge to learn CA policies that generate effective CAs that dynamically account for changes within MAS and estimate a functional nominal as compared to a predefined design nominal used by current approaches. These capabilities will mitigate shape errors by providing CAs across all stages of the NPI lifecycle. This will enable the fulfilment of CA related requirements, namely *(xi) Uncertainty of isolated RCs, (xii) KPI improvement, (xiii) MAS design architecture, (xiv) MAS inherent stochasticity*. *The research addresses this requirement by developing the OSEC kernel.*

These knowledge gaps and the corresponding requirements are addressed by the four kernels as described in Section 1.2. Further, the thesis comprehensively discusses the developed kernels that satisfy all the outlined requirements and close the corresponding knowledge gaps between manufacturing assembly systems and deep learning.

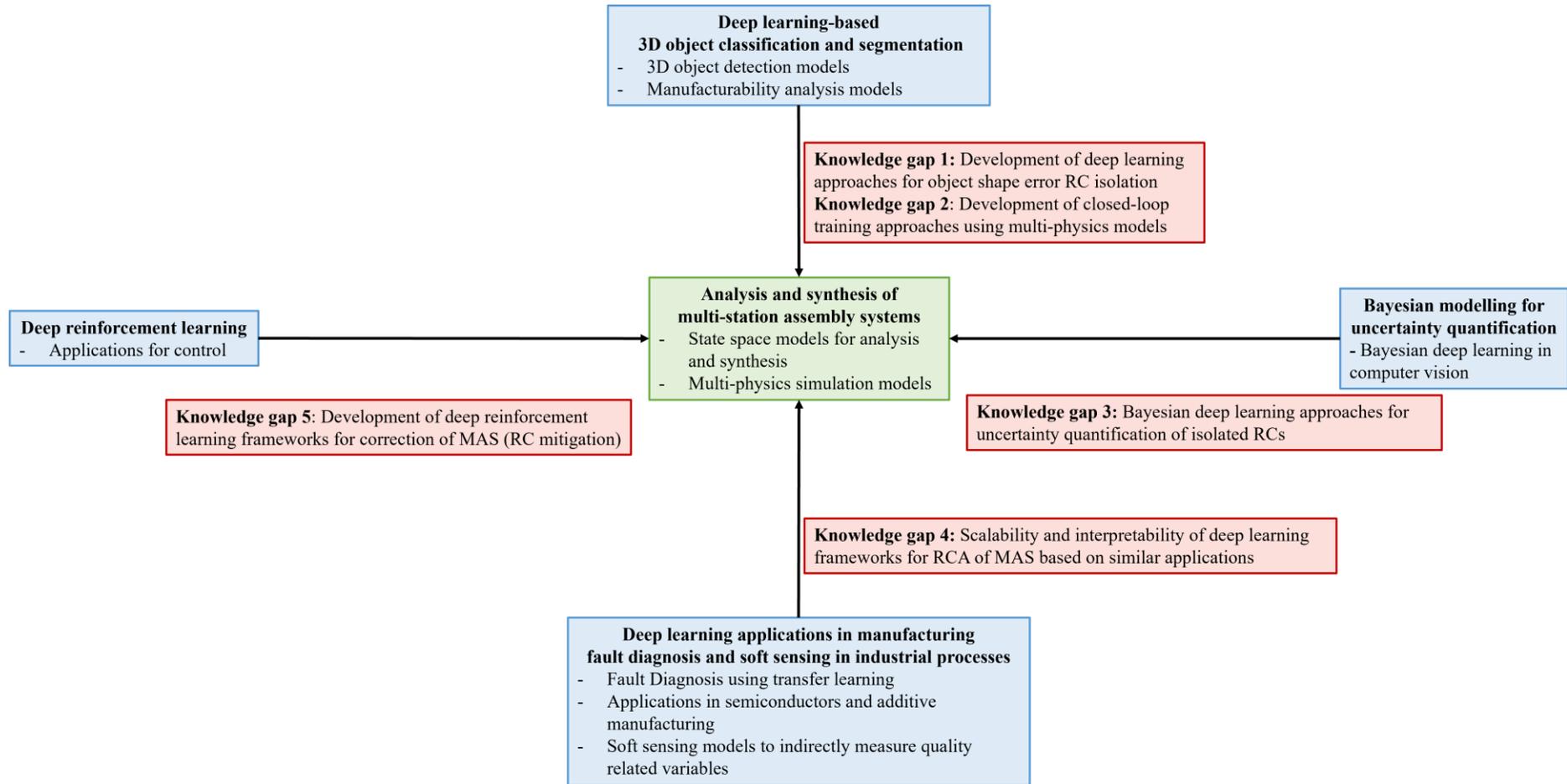


Fig. 2.10. Literature assimilation and knowledge gap identification

3 Object Shape Error Response using Bayesian 3D Convolutional Neural Networks for Assembly Systems with Compliant Parts¹

3.1 Introduction

The chapter proposes the novel OSER kernel to estimate the dimensional and geometric variation of assembled products and then relate these to process parameters, which can be interpreted as RCs of the object shape defects. The OSER approach leverages Bayesian 3D-Convolutional Neural Networks integrated with CAE simulations for RC isolation. Compared with the existing methods, the proposed approach (i) addresses a novel problem of applying deep learning for object shape error identification instead of object detection; (ii) overcomes fundamental performance limitations of current linear approaches for RCA of assembly systems that cannot be used on point cloud data; and, (iii) provides capabilities for unsolved challenges such as ill-conditioning, fault-multiplicity, RC prediction with uncertainty quantification and learning at design phase when no measurement data is available. Comprehensive benchmarking with existing machine learning models demonstrates superior performance with $R^2=0.98$ and Mean Absolute Error (MAE) = 0.05 mm, thus, improving RCA capabilities by 29%. The kernel fulfils requirements (i)-(vi) within the scope of single station assembly systems. The contributions of the chapter can be summarized as follows:

(1) Propose a *3D OSER methodology* based on a novel Bayesian 3D CNN architecture:

¹ Based on S. Sinha, P. Franciosa, and D. Ceglarek, "Object Shape Error Response using Bayesian 3D Convolutional Neural Networks for Assembly Systems with Compliant Parts," vol. 17, no. 10, pp. 6676-6686, IEEE Transactions on Industrial Informatics, 2021, DOI: 10.1109/TII.2020.3043226.

<https://ieeexplore.ieee.org/document/9286512>

it builds on current research in the area of 3D Object Detection [14] by expanding it to manufacturing systems where the key goal is not to detect the object but to estimate various shape error patterns present on the final object/product and relate these variation patterns to manufacturing process parameters variations within the system. This is the first work to propose an uncertainty enabled 3D CNN-based deep learning model for RCA of assembly systems.

- (2) Propose a *closed-loop framework for training and deployment* of the Bayesian 3D CNN model that leverages a Computer-Aided Engineering (CAE) simulator known as VRM [6] to emulate the multi-stage assembly system. The VRM performs sampling, which leverages the Bayesian 3D CNN's epistemic uncertainty estimates, thereby, reducing overall simulation and training time. Given that data availability within manufacturing systems is costly, scarce and the data can be highly skewed, the VRM functions as a physics-based Digital Twin for generating augmented data that is close to the physical system and can, therefore, be used to train the proposed model. The trained model can then be leveraged for RCA of assembly systems using point cloud scans obtained from 3D scanners.
- (3) Verify & validate the methodology on an *industrial, automotive door assembly process* made of compliant parts.
- (4) *Benchmark 3D OSER methodology* against three categories of methods that can be leveraged to estimate the dimensional and geometric variation of assembled products, namely, (i) current linear state-of-the-art RCA models; (ii) machine learning models in a multi-output regression setting; and (iii) deep learning models such as various types of CNNs and fully connected networks to highlight performance and the ability to fulfil the aforementioned six requirements.

The rest of the chapter is organised as follows; Section 3.2 formulates the object shape error estimation problem; Section 3.3 presents the proposed Bayesian 3D CNN architecture, the steps involved in architecture optimization and the overall steps required to train and deploy the model; Section 3.4 presents the industrial case study. Finally, conclusions are summarized in Section 3.5.

3.2 Problem Formulation

3.2.1 Object Shape Error Estimation in Manufacturing

Multi-stage assembly systems can be mathematically expressed as a state-space model where different states correspond to different configurations of the manufacturing system at a given stage [73]. The input is an object (set of parts to be assembled) entering the assembly process. Within the process, object shape errors can be introduced in any stages due to one or multiple variations in the process parameters and are further propagated through the stages (Fig. 3.1). Any object o at its design nominal shape is characterized by a set of nominal points $\mathbf{P}_o = \{\mathbf{p}_{ok}\}$, $k = 1, \dots, n_o$, where \mathbf{p}_{ok} is a vector consisting of the x,y and z coordinates of the kth input point and n_o represents the total number of points on object o . The object here represents a single subassembly that is assembled in a single station, which can be understood as a collective reference to all parts used in this assembly station. In practice, the points correspond to mesh nodes in the Computer-Aided Design model of the object when considering CAE simulations and actual points within the point cloud when considering the 3D scan of the object. $\mathbf{d}_o = \{\mathbf{d}_{ok}\}$ denotes the deviation of each point k after the nominal object o has gone through different stages of the process, \mathbf{d}_{ok} is a vector comprised of deviations of each point in x,y and z axes on object o . An assumption made in this chapter is that the assembly process has a single station which includes multiple stages $s = 1, \dots, 4$ involving objects/parts: positioning (P), clamping (C), fastening (F) and release (R). This assumption is relaxed in chapter 3. Stage $s = 0$ is used to represent the incoming part that includes deviations from the previous processes, such as part fabrication. As the object o goes through multiple stages, the set of points are represented as \mathbf{P}_o^s while \mathbf{d}_o^s represents the deviations.

As the primary goal of this research work is object shape error estimation hence, it extends the problem formulation in object detection, which only considers the set of points $\{\mathbf{p}_{ok}\}$ [20], by including deviations for each point $\{\mathbf{d}_{ok}\}$ as additional features. This adds the required discriminative ability in the data hence, enabling object shape error estimation. Thus, the object shape error for object o after stage s can be represented as:

$$\mathbf{x}_o^s = (\mathbf{P}_o^s, \mathbf{d}_o^s) \quad (3.1)$$

On the other hand, the set of all process parameters across all stages are denoted by \mathbf{y} where $\mathbf{y} = \{y_1, \dots, y_h\}$, h denotes the total number of process parameters. The deviation \mathbf{d}_o^s of points at each stage s for object o can be expressed as the sum of all deviations Δ_o^s accumulated in all stages from stage 0 up to stage s :

$$\mathbf{d}_o^s = \sum_{j=0}^s \Delta_o^j \quad (3.2)$$

where \mathbf{d}_o^0 represents the shape error of incoming object o caused by upstream manufacturing processes. After each stage s the actual points of the object o with error can be written as:

$$\mathbf{P}_o^s = \mathbf{P}_o + \mathbf{d}_o^s \quad (3.3)$$

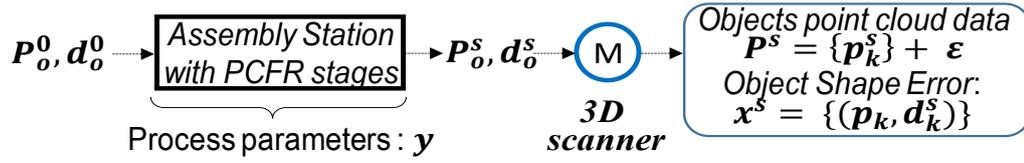
At the end of the final assembly stage $s = 4$ the object shape error data for the assembly $\mathbf{P}^{s=4}$ is collected and decomposed into the nominal points \mathbf{P} and their deviations $\mathbf{d}^{s=4}$ by using alignment techniques[64], where $\mathbf{P}^{s=4}$, $\mathbf{d}^{s=4}$ are now a collective reference to the set of all incoming objects that have been assembled. The measurement system error ε is considered to be negligible ($\varepsilon \approx 0$). The object with errors is represented as a point cloud of non-ideal part:

$$\mathbf{x}^s = \{(\mathbf{p}_k, \mathbf{d}_k^s)\} = (\mathbf{P}, \mathbf{d}^s) \quad (3.4)$$

where \mathbf{d}_k^s can be considered as features at each point \mathbf{p}_k .

The Bayesian 3D CNN model training aims to learn assembly process transfer function $f(\cdot)$ (equivalent to state transition matrix in [51]). The function $f(\cdot)$ is parametrized by weights and biases of a CNN that can accurately estimate the process parameters \mathbf{y} given the point cloud data of non-ideal \mathbf{x}^s parts collected from the system:

$$\mathbf{y} = f(\mathbf{x}^s) \quad (3.5)$$



PCFR: Position-Clamp-Fastening-Release

Fig. 3.1. Object shape error propagation in assembly systems

The high accuracy of the 3D CNN in estimating all assembly process parameters y provides the underlined capability of the OSER approach for high root cause (RC) isolability. Essentially within assembly systems, RCs are estimated as a subset of the estimated process parameters:

$$RC \subseteq y \quad (3.6)$$

Based on the requirements and the production phase of the assembly system, the exact definition of an RC may differ, but the critical requirement to conduct RCA under any definition is to accurately estimate all process parameters y . Hence, the proposed OSER approach aims to do the aforementioned by estimating $f(.)$ as specified in (3.5).

3.2.2 3D Object Shape Error Voxelization

In the presented OSER approach, the simulation output represented as mesh or point cloud data $\{(p_k, d_k^s)\}$ (3.4) is transformed to voxel grids $\{V_{u,v,w}\}$ with discrete voxel coordinates (u,v,w) in the following way: for all points $p_k = (x_k, y_k, z_k)$ that fall within a voxel grid $\{V_{u,v,w}\}$ the maximum value of $d_k = (\tilde{x}_k, \tilde{y}_k, \tilde{z}_k)$ across all points mapped to the same voxel, characterizes the features of the corresponding voxel grid and is represented as $\{V_{u,v,w,d}\}$. The voxelization techniques used in object detection [20] is applied to construct the initial voxel structure of the object, and for each unique object, the voxel features are characterized by the shape error d_k . The key difference is that in object detection, voxel grids are characterized by either binary voxels or voxels containing RGB values for each point instead of real values of shape error d_k as in the OSER approach. Although binary voxels, traditionally used in Object Detection, retain

the spatial structure, the granularity of voxelization required to discriminate between minor differences in the shape error will make the problem computationally infeasible and hence, limit performance. In the proposed approach, the nominal object is voxelized, and actual values of the shape error characterize each voxel \mathbf{d}_k . This is critical in representing the geometric variations with the required granularity for effective RCA. This efficiently retains all information about the spatial structure of the object and the components of object shape errors. Given the alignment ensures a fixed orientation, there is no need for data augmentation to achieve rotation invariance.

3.2.3 Uncertainty Estimation

Given the uncertainties of the system and the availability of only a limited dataset, a deterministic estimate of function $f(\cdot)$, as shown in (3.5), is not feasible. Hence, by leveraging Bayesian inference, a prior distribution can be allocated over the space of possible functions $p(f)$ which represents a prior belief of the possible functions $f(\cdot)$. Given a dataset, a likelihood $p(\mathbf{y}|f, \mathbf{x}^s)$ is defined to model the function from which the observation is generated; and, hence, given a dataset $(\mathbf{x}^s, \mathbf{y})$ the posterior distribution over the functions $p(f|\mathbf{x}^s, \mathbf{y})$ can be inferred. The function is characterized by model parameters ω represented by f^ω (weights and biases for neural networks) and the posterior over the function can be inferred by estimating the posterior over the parameters ω . In Bayesian Neural networks, this is achieved through Bayes-by-Backprop [164] and Flipout [89]. Given a dataset, the posterior can be written as:

$$p(\omega|\mathbf{x}^s, \mathbf{y}) = p(\mathbf{y}|\mathbf{x}^s, \omega) p(\omega)/p(\mathbf{y}|\mathbf{x}^s) \quad (3.7)$$

For complex models such as deep neural networks, it is not analytically possible to infer the true posterior for all model parameters $p(\omega|\mathbf{x}^s, \mathbf{y})$ hence, an approximating variational distribution $q_\theta(\omega)$ parametrized by θ , such as normal distribution, is used to approximate the posterior. This approach is known as variational inference [180]. The approximating distribution should be as close as possible to the true posterior, which is achieved by minimizing the Kullback-Leibler (KL) divergence with respect to θ :

$$KL(q_{\theta}(\omega)||p(\omega|\mathbf{x}^s, \mathbf{y})) = \int q_{\theta}(\omega) \log q_{\theta}(\omega)/p(\omega|\mathbf{x}^s, \mathbf{y}) d\omega \quad (3.8)$$

The posterior weight distribution parameters θ are obtained by minimizing the variational free energy cost function known as the expected lower bound (ELBO). ELBO includes sum of (i) Kullback-Leibler (KL) divergence (3.8) which measures the divergence between the true posterior and its variational approximation, and (ii) negative log-likelihood (NLL) of the dataset which measures the goodness-of-fit of the model. Using the estimated variational distribution $q^*_{\theta}(\omega)$ the process parameter distribution quantifying the uncertainties for a new data point \mathbf{x}^{s*} can be obtained using:

$$p(\mathbf{y}^*|\mathbf{x}^{s*}, \mathbf{x}^s, \mathbf{y}) \approx \int p(\mathbf{y}^*|\mathbf{x}^s, \omega)p(\omega|\mathbf{x}^s, \mathbf{y})d\omega =: q^*_{\theta}(\mathbf{y}^*|\mathbf{x}^{s*}) \quad (3.9)$$

3.3 Methods

3.3.1 Bayesian 3D CNN Model Architecture

Building on the work done on voxel-based approaches for 3D object detection such as VoxNet[20], the research proposes a Bayesian 3D CNN architecture to enable object shape error estimation. The 3D convolutions aggregate features from the input, which are then utilized by the fully connected layers and mapped to process parameters. The model consists of three 3D convolutional Flipout layers, a 3D max-pooling layer followed by three fully connected Flipout layers. The final layer estimates parameters of the predictive distribution for all process parameters. The convolution can be represented as:

$$v_{ab}^{xyz} = ReLU \left(\beta_{ab} + \sum_m \sum_{p=0}^{P_i-L_i} \sum_{q=0}^{Q_i-M_i} \sum_{r=0}^{R_i-N_i} w_{ab}^{pqr} v_{(a-1)m}^{(x+p)(y+q)(z+r)} \right) \quad (3.10)$$

where v_{ab}^{xyz} represents the layer output value at position (x, y, z) in the a^{th} layer and b^{th} feature map. ReLU is the Rectified Linear Unit activation function [113]. β_{ab} represents the bias; m represents no. of filters from the previous layer; (P_a, Q_a, R_a) and

(L_a, M_a, N_a) represent the kernel dimensions and stride lengths in the three directions, respectively; w_{ab}^{pqr} represents the weights of the connections. The convolution operation as in (3.11) is done consecutively for the three convolutional layers. In 3D max-pooling operation, the resolution of the feature map is reduced by taking the maximum value of the local neighbourhood of the layer outputs. Given the Bayesian framework, each parameter of the Bayesian 3D CNN model follows a distribution. In the case of neural networks, it is not feasible to assign informative priors; hence, non-informative prior distributions are placed over the model parameters. Each parameter is approximated using a variational inference approach, assuming that the posterior follows a normal distribution. The overall model has 1,997,286 trainable parameters. Output nodes have linear activation units. Fig. 3.2 shows the proposed Bayesian 3D CNN model architecture with annotated hyper-parameters.

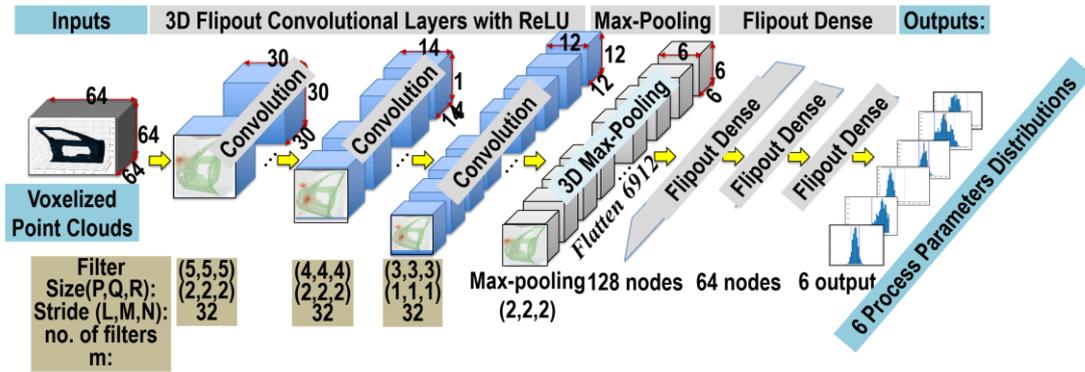


Fig. 3.2. Bayesian 3D CNN model architecture of the OSER method

3.3.2 Architecture Selection and Optimization

Hyper-parameters optimization for Bayesian 3D CNNs is done to maximize performance and eliminate architectures that are more likely to overfit. As this is computationally intensive hence, in order to perform optimization in a computationally feasible manner, the following steps were involved:

Step 1 – Set Baseline: VoxNet [14], a 3D CNN architecture used for object detection consisting of two 3D convolutional layers, one max-pooling layer and two fully connected layers, is set as the baseline. A dataset consisting of 1500 samples is generated to conduct k-fold cross-validation (k=6). The hyper-parameters are split into

two categories; Category one consists of the number of convolutional layers $N_C = \{2,3,4\}$ and number of dense layers $N_D = \{1,2,3\}$; Category two consisted of the number of filters in each 3D convolutional layer, filter size for each 3D convolutional layer and number of hidden units in each dense layer.

Step 2 – Grid Search for Category one Hyper-parameters: In this step, a grid search for category one hyper-parameters is conducted. Each selection is evaluated using k-fold cross-validation (Fig. 3.3). For computational feasibility, category two hyper-parameters are kept constant and equal to the VoxNet architecture values. $N_C = 3$ and $N_D = 3$ were obtained as the optimal hyper-parameters having the minimum cross-validation Mean Absolute Error (MAE) average of 0.08 mm.

Step 3 – Hyperband for Category Two Hyper-parameters: The optimal values for category one hyper-parameters are fixed, and further Hyperband [181] is leveraged to obtain the optimal values for category two hyper-parameters given its ability to speed up the random search process through adaptive resource allocation and early stopping.

Step 4 – Deterministic to Bayesian Model: The final step includes replacing the deterministic layer with Bayesian Flipout layers and then training using Bayes-By-Backprop. Various learning rates and prior distributions for the model weights were tested. Standard normal distribution provided the best balance between weight initialization and weight exploration, which was inferred by conducting an uncertainty vs error calibration study. The training hyper-parameters that provided the best uncertainty calibration and ensured that the model performance was more significant than or equal to the deterministic counterpart were selected as the final Bayesian 3D CNN architecture training hyper-parameters. The changes from the baseline architecture of Object Detection that enable the fulfilment of the aforementioned six requirements are summarized in Table 3.1.

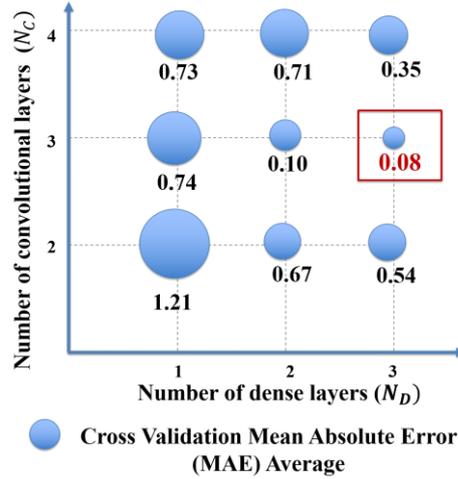


Fig. 3.3. Grid search for category one hyper-parameters

Table 3.1: Object detection & OSER comparison

Object Detection (VoxNet)	Object Shape Error Detection (OSER)	Rationale
Single-channel binary occupancy input voxel of dimension $32 \times 32 \times 32$	Multi-channel real-valued object shape error input voxel of dimension $64 \times 64 \times 64 \times 3$	High data dimensionality of shape error (\mathbf{P}, \mathbf{d}^s) (Requirement (i))
Two 3D Convolutional layers, one dense layer and categorical output layer	Three 3D Convolutional layers, three dense layers and real-valued output layer	Increased model capabilities to handle non-linearity, collinearity and high fault multiplicity (Requirement (ii),(iii),(iv))
Deterministic Layer	Bayesian Flipout Layer	Uncertainty quantification for corrective actions and closed-loop sampling for faster convergence (Requirement (v),(vi))

3.3.3 Model Training and Deployment

Training of the model is done in a closed-loop framework (discussed in detail in Chapter 5) using data generated by OSEM. The key motivation behind using a closed-loop framework instead of an open-loop framework is to minimize the bottle-neck computation, i.e., multi-physics simulation using the VRM model. Although this increases the number of training iterations, the overall time of VRM simulation and training is significantly reduced as fewer samples need to be generated. The key steps of the proposed framework are summarized below (Fig. 3.4):

Step 1 – Sampling: Process parameters \mathbf{y} are sampled from the allowable ranges. Latin Hypercube Sampling [182] is used to generate initial process parameter sample values, given it distributes samples optimally across the h –dimensional process parameter space by stratifying the possible ranges. The consecutive sets of samples are generated using Monte Carlo sampling based on the uncertainty $\sigma(\hat{\mathbf{y}})$ of the model.

Step 2 – VRM Simulation: The samples are used as input to the VRM to simulate the assembly process and generate the output mesh from which the point cloud and deviations of each point are extracted after the desired stage s of the assembly system as in (3.4) $\mathbf{x}^s = \{(\mathbf{p}_k, \mathbf{d}_k^s)\}$.

Step 3 – Model Training: The point cloud and deviation data of object shape errors along with the respective process parameters ($\mathbf{x}^s = \{(\mathbf{p}_k, \mathbf{d}_k^s)\}, \mathbf{y}$) are used for model training. Note that \mathbf{x}^s is voxelized $\{\mathbf{V}_{u,v,w,d}\}$ before it is used for training. Bayes-by-Backprop and Flipout are applied for model training. The loss function optimized while training comprises the sum of KL divergence for each layer (3.8) and the negative log-likelihood (3.11) of the predictive distribution.

$$-\ln L = 1/2 [\ln(|\Sigma|) + (\mathbf{y} - \boldsymbol{\mu})^T \Sigma^{-1} (\mathbf{y} - \boldsymbol{\mu}) + h \ln(2\pi)] \quad (3.11)$$

The KL divergence term quantifies the divergence between the true posterior and its variational approximation. After rearrangement one of the terms appearing in the KL divergence quantifies the role of the prior and has a regularization effect, hence preventing overfitting. Group normalization [183] with four groups is used after each convolutional layer. This also prevents overfitting and accounts for small minibatch size due to GPU memory size constraints and aids in stabilizing the training process. Weights of the network are initialized using a normal initializer [184]. The Adam method for stochastic optimization was used to optimize the loss function while training [185]. The initial learning rate is fine-tuned to $\alpha = 0.0005$, and monotonic KL annealing was leveraged to ensure the model initially learns the object shape error and process parameter relations before applying the KL penalty for uncertainty quantification. The learning rate fine-tuning, monotonic KL annealing and ReLU

activations prevent gradient vanishing. The predictive distribution is modelled as a multivariate normal \mathcal{N}_h with h components (same number of components as the number of process parameters h), $\mathbf{y} \sim \mathcal{N}_h(\boldsymbol{\mu}, \boldsymbol{\Sigma})$ where each component corresponds to a process parameter hence, the mean across all components of the multi-variate distribution corresponds to the set of process parameters \mathbf{y} . The distribution is assumed to have a diagonal covariance matrix $\boldsymbol{\Sigma}$. The scale parameters in the diagonal are assumed to be fixed since the noise has been assumed to be negligible.

After each iteration of training, the model is evaluated on the validation set. For evaluation, Monte Carlo (MC) sampling from the model is done, and the sample means $\bar{\mathbf{y}}$ and standard deviations $\sigma(\hat{\mathbf{y}})$ are estimated for each process parameter. $\sigma(\hat{\mathbf{y}})$ represents the epistemic uncertainty, while the fixed scale parameters of the predictive distribution represent the known aleatoric uncertainty [36]. Given the assumption of negligible measurement noise, aleatoric uncertainty is considered to be negligible, and hence, the overall uncertainty in the prediction can be assumed to be equal to epistemic uncertainty $\sigma(\hat{\mathbf{y}})$. This uncertainty is used for sampling in the next iteration.

Mean Absolute Error (MAE) between the model estimates $\hat{\mathbf{y}} = \bar{\mathbf{y}}$ and actual value \mathbf{y} across all process parameters h (3.12) is used as the metric for model performance evaluation given the ease of interpretation and given that the model outputs are continuous and real-valued. Training is stopped when MAE is below the required threshold ϵ . The threshold value for this metric is determined based on the quality requirements for a specific product as required by design tolerances and the accuracy of the measurement system. The model is trained within the measurement system accuracy. For example, automotive body assembly process tolerances are within [-1mm,1mm], and the 3D optical scanner used has a repeatability of 0.05 mm and accuracy within 0.15 mm.

$$MAE = \sum_h |\mathbf{y} - \hat{\mathbf{y}}|/h \quad (3.12)$$

Step 4 – Model Deployment: After training, the model can be deployed within an

existing system. The data collected from the 3D scanner P^s is aligned to obtain point cloud and deviations $x^s = \{p_k, d_k^s\}$ and then, voxelized $V_{u,v,w,d}$ before it can be given to the trained model for conducting RCA inference. Inferencing estimates the process parameters for a given x^s (3.5) using MC sampling from the trained model. Using these samples, process parameters (distribution mean) \bar{y} and their uncertainty (distribution standard deviation) $\sigma(\hat{y})$ can be estimated. The sample mean \bar{y} is considered as the model estimate \hat{y} , while $\sigma(\hat{y})$ quantifies the uncertainty. Further, the RCs can be inferred as a subset of \hat{y} (3.6). The work has been implemented using Python 3.7 and TensorFlow - GPU 2.0 [186] and TensorFlow-Probability 0.8. A python library, Bayesian Deep Learning for Manufacturing [187] has been developed to validate and replicate the results of the methodology. Both the data generation and evaluation of the OSER methodology have been done using VRM. Two Nvidia Tesla V100 32 GB GPUs are used for model training and deployment.

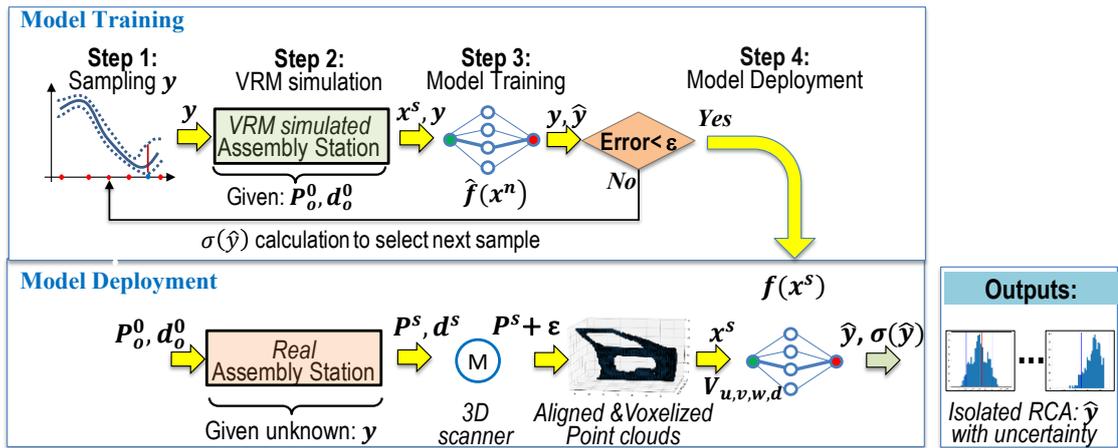


Fig. 3.4. Model training and deployment framework

3.4 Case Study

3.4.1 Assembly Setup

For verification and validation of the proposed OSER approach, an automotive assembly of two components, namely, the door inner and hinge reinforcement, are selected. The assembly setup and parameters are shown in Fig. 3.5. The assembly process is controlled by the six ($h = 6$) parametrized process parameters y_1, y_2, \dots, y_6 (depicted using yellow symbols in Fig. 3.5). Assembly parameters such as pin-hole,

pin-slot and NC blocks for the door inner are considered constant (depicted using green symbols in Fig. 3.5) and are not parameterized. Data is collected after stage $s = 4$. The point cloud is characterized by $n = 10841$ points, which are pre-processed and voxelized to $(u, v, w) = (64, 64, 64)$ voxel grids. The deviation features \mathbf{d} include deviations in all directions for all points $(\tilde{x}_k, \tilde{y}_k, \tilde{z}_k)$. The assembly consists of four stages (Fig. 3.6): Stage 1 involves positioning (i) the door inner on the pin-hole, pin-slot and the three NC blocks (not parameterized; marked in green in Fig. 3.6), (ii) hinge reinforcement using the pin-slot (y_1), pin-hole (y_2, y_3); Stage 2 comprises of clamping two parts together using three NC-Blocks with clamps (y_4, y_5 and y_6); Stage 3 involves fastening/joining of the two parts using self-piercing riveting (SPR); and, finally, Stage 4 involves releasing the clamps (y_4, y_5 and y_6) after the fastening is completed. The training range for all process parameters is $[-1 \text{ mm}, 1 \text{ mm}]$ while the validation range is $[-2 \text{ mm}, 2 \text{ mm}]$. Point cloud and deviation data $\{(p_k, d_k^4)\}$ are collected after release, i.e., Stage 4 (Fig. 3.6). The data is voxelized $\{V_{64,64,64,3}\}$ and used as model input and the process parameters y_1, y_2, \dots, y_6 are used as model outputs.

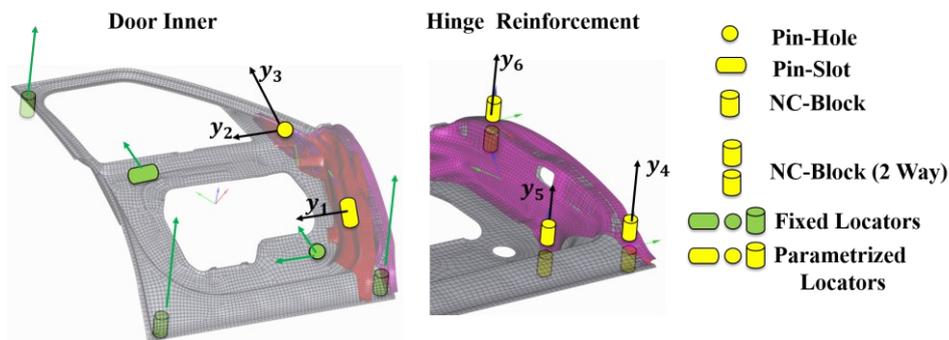


Fig. 3.5. Assembly Process Parameters

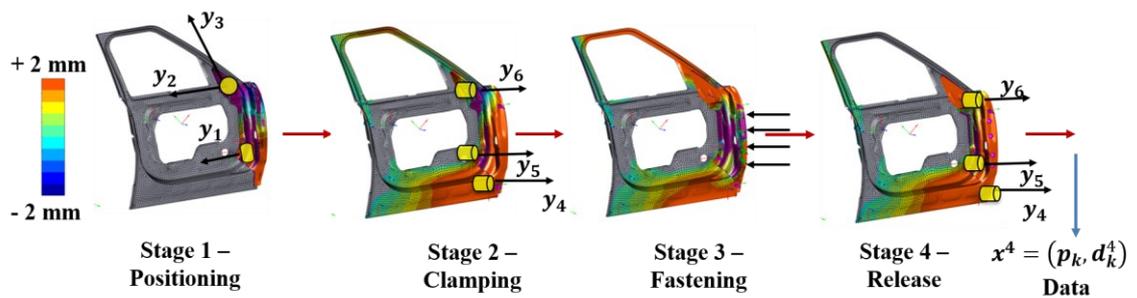


Fig. 3.6. PCFR Stages of the Assembly Process

After starting with 200 initial samples for model training, 200 samples are adaptively added during each iteration of the closed-loop training based on the uncertainty estimates. The model is trained on the combined dataset, including all previous samplings, to ensure no catastrophic forgetting (using 200 samples provided an optimal tradeoff between VRM simulation time and model training time). These samples and outputs are used for training the Bayesian 3D CNN model. The diagonal scale parameters for all process parameters in the covariance matrix are fixed at 0.001. A validation set of 300 samples is generated within the validation range, and after each iteration, the trained model is evaluated on the validation set. During evaluation for each of 300 samples, 1000 MC samples are drawn from the trained model. The sample means are considered as the estimate for the process parameters, while the sample standard deviations quantify the uncertainty for each process parameter for the given sample. RCs can be inferred from the process parameter estimates. The closed-loop training is stopped when the average MAE across all process parameters for the validation set is below the threshold selected to be 0.05 mm for automotive assembly applications as the impact of variations less than 0.05 mm is not detectable by the 3D scanner. After this, the model is ready for deployment with measurement data collected from 3D optical scanners followed by alignment and voxelization. For each measurement, MC samples from the trained Bayesian 3D CNN model can be drawn to estimate process parameter mean and standard deviations (uncertainty). Measurement data collection is done using WLS400A[4] mounted on an ABB robot.

In summary, the industrial assembly process selected for the case study consists of (i) high dimensionality point cloud (10841 points); (ii) non-linearity as induced by fixturing ($N-2-1$, where $N=6$), two compliant parts and part-to-part interactions (door inner to hinge reinforcement); (iii) collinearity induced by fixturing as locators: $y_4, y_5, \text{ and } y_6$ are within 5 degrees of collinearity (-3 to 2-degree deviation from axis y); and (iv) high fault multiplicity as we take into consideration 6-sigma defects at the level of variation within 3D scanner accuracy (<0.05 mm) that significantly increases fault multiplicity from zero to 6 process parameters manifesting errors (100% fault multiplicity). The door assembly requirements are: (1) Product: Design tolerances of door assembly: [-1.0, 1.0] [mm], (2) Process: Fixturing calibration and commissioning

is achieved within $[-0.1; 0.1]$ [mm], and (3) Shape error detection: Using the 3D optical scanner for measurement.

Key Performance Indicators (KPIs) used for assessment of the results are as follows: (i) Mean Absolute Error (MAE) < 0.05 mm and, (ii) $R^2 > 0.95$ for the model to have the capability to explain more than 95% variance in the process parameters under the assembly system Requirements (ii)-(iv).

3.4.2 Results

The KPIs of model performance are summarized for all y_1, \dots, y_6 in Fig. 3.7. The model convergence is shown in Fig. 3.8. The model converges with average MAE across all process parameters equal to 0.05 (below the required threshold) and average R^2 equal to 0.98 after 10 iterations of closed-loop training, which included a total of 2000 samples being generated adaptively. For validation purposes, this study trained both Bayesian 3D CNN and a deterministic version of the model, i.e., 3D CNN with the same architecture as in Fig. 3.2.

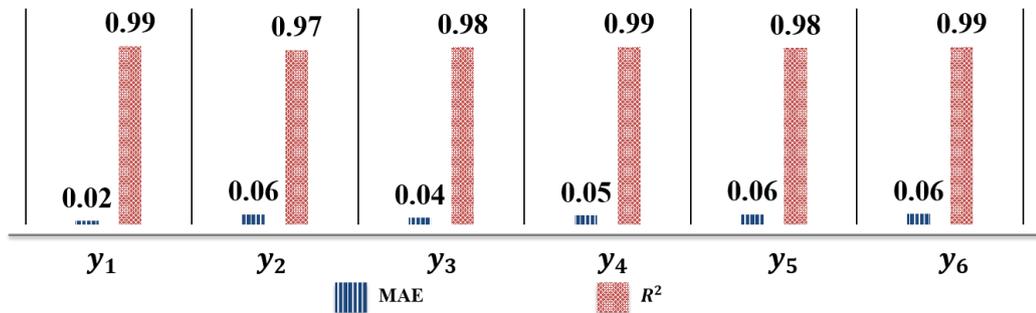


Fig. 3.7. MAE and R^2 across all process parameters

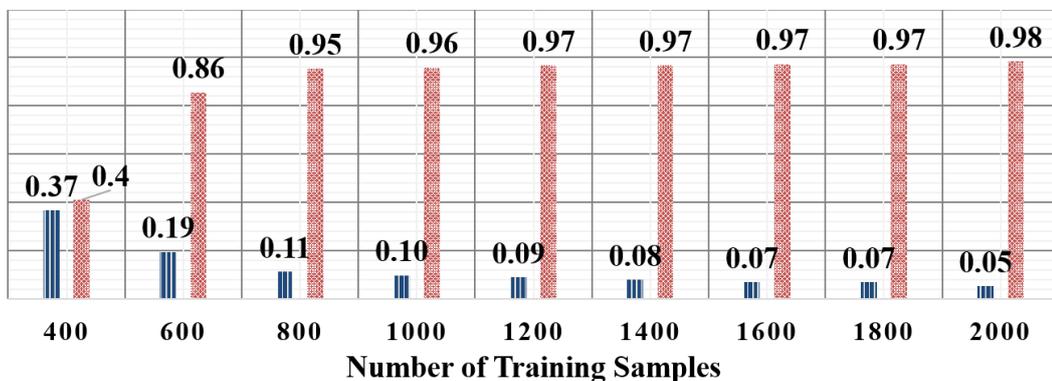


Fig. 3.8. Bayesian 3D CNN OSER Convergence

3.4.3 Benchmarking and Discussion

The benchmarking analysis is conducted by using the six requirements as listed in Section 1.2. The case study and results and analysis of collinearity, multiplicity and uncertainty, are used to demonstrate the capabilities of the proposed approach to fulfil the aforementioned requirements. The benchmarking analysis of the proposed 3D OSER approach is discussed on two levels:

1. *OSER vs currently used approaches at production phase when point cloud data is available* – The benchmarking is conducted for two scenarios: (a) RCA; and (b) RCA with uncertainty quantification;

RCA: as discussed in Section 1.2, the state-of-the-art models used for assembly process RCA, such as [54] [188], are linear and can be classified as regularized linear regression approaches (Table 3.2). Hence, their upper limit performance can be estimated by using regularized linear regression on all point deviations d within the point cloud. They also use a limited number of sampled points from the point cloud on a single part (less than 100 out of >10,000) which additionally limit their performance for assembly processes. The OSER methodology validation against the six requirements as presented in Section 1.2 is as follows. Requirement (i) is fulfilled by the proposed voxelization approach, which ensures that irrespective of the dimensionality of the point cloud, it is transformed into a sparse tensor of dimensions (64,64,64,3) which preserves information in terms of the spatial object structure and point deviation features. This also enables the application of OSER based models that require a regular data structure as input. Secondly, the model performance of the state-of-the-art regularized linear regression approaches is at $MAE = 0.41$ mm and $R^2 = 0.76$ (see Table 3.2), which is unsatisfactory as compared to the required $MAE < 0.05$, $R^2 > 0.95$. This is because the regularized linear regression model can explain only the linear variance in the system. By comparison, the proposed OSER model demonstrates good performance at $MAE = 0.05$, $R^2 = 0.98$, hence fulfilling Requirements (ii), (iii) and (iv). Fig. 3.9 compares the performance of regularized linear regression (i.e. upper limit for state-of-the-art approaches) with the proposed OSER approach under different levels of fault multiplicity and collinearity. For example, in scenarios 1, 2 and 3 (fault multiplicity up to 50%), both approaches have similar performance. However, in scenarios 4, 5 and 6,

as the fault multiplicity increases to 4, 5, and 6 parameters being simultaneously at fault, i.e., 100% of parameters, and with induced by collinear design relation between process parameters and input, the performance of linear model decreases while OSER approach exhibits performance above the required threshold ($R^2 > 0.95$).

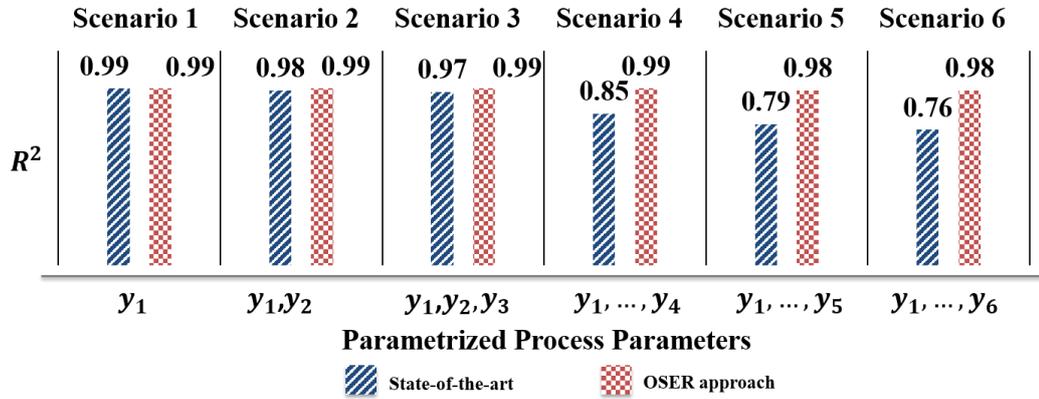


Fig. 3.9. Performance under different levels of fault collinearity and multiplicity (from single fault y_1 ; and two-fault y_1, y_2 ; ... up to all parameters being simultaneously at fault y_1, y_2, \dots, y_6)

The benchmarking also comprehensively assesses the OSER against existing deep learning and machine learning techniques [189] in ways that are not currently used for RCA of assembly processes (see Table 3.2). This chapter implemented these techniques and applied them for the aforementioned case study. CNN based deep learning methods were selected as they retain spatial information while learning, which is essential for object shape error estimation. Each model is compared in its ability to fulfil the aforementioned six requirements. Table 3.2 shows the implemented benchmark approaches and the results. The multi-view 2D CNN (MV-CNNs) [190] approach considers six 2D projections of the object shape error. Gridding is performed on each projection. Each projection has an input dimension of $64 \times 64 \times 3$. Each of the six projections is given as input to the MV- CNN consisting of six heads. These are pooled before the fully connected layers. Depth based CNNs [130] considers a single projection along the y-axis with dimension $64 \times 64 \times 4$. The first three channels consist of the shape error (\mathbf{d}), while the fourth channel consists of the y-coordinate of the nominal point-cloud \mathbf{P}_o . Both 2D CNNs based approaches have same hyperparameters as the OSER (but only considering 2D Convolutions and 2D Max-pooling). The deep dense neural network is given as input the flattened vector of shape

error $\mathbf{d} = Flatten \{(\tilde{x}_k, \tilde{y}_k, \tilde{z}_k)\}$. It consists of two hidden layers (1024, 512 nodes respectively) with ReLU activations and a linear output layer with 6 nodes. All machine learning models take as input a transformed input of $\mathbf{d} = Flatten \{(\tilde{x}_k, \tilde{y}_k, \tilde{z}_k)\}$. Principal component Analysis (PCA) is used for the transformation and reduced features explaining 99% of the variance are retained. Comparison for Requirement (i) is done based on the transformation used for input. Shape error voxelization retains information on the 3D structure and shape error features, while projection retains only 2D spatial structure. Flattening eliminates all information related to the spatial structure, while PCA also eliminates information explaining 1% of the variance. Comparison for Requirements (ii), (iii) and (iv) are done on performance attributes, namely, accuracy (MAE) and goodness-of-fit (R^2). Comparison for Requirement (v) is done on the ability to quantify and segregate uncertainties. Lastly, a comparison for requirement (vi) is done on overall training time, which is inclusive of the CAE simulation time and model training time. Although the proposed model has a higher model training time, the overall training time is significantly lesser due to leveraging the epistemic uncertainty to generate informative samples leading to faster convergence with only ~ 2000 samples. All other models are trained using random sampling until convergence. Fig. 3.10 summarizes the convergence of the entire set of benchmarking models. The hyper-parameters of the machine learning models were optimized using grid search. For statistical quantification of accuracy and goodness-of-fit, 20 runs of training and testing are conducted using a set of 4000 randomly sampled data points for training and 300 for validation within the validation range. The mean and standard deviation (SD) for each model-averaged across six process parameters have been reported. The proposed OSER model's model performance is significantly better in terms of accuracy and goodness of fit. The result from ANOVA followed by post-hoc Tukey-HSD test at 95% significance level considering two sources of variations (model type and process parameter) showed the differences to be statistically significant. This comes at the expense of increased model complexity.

Table 3.2: Benchmarking Results

	Models	Requirement (i)	Requirement (ii),(iii),(iv)				Requirement (v)	Requirement (vi)			
			Accuracy (MAE)		Goodness-of-fit (R^2)			Sampling	CAE Simulation Time (minutes)	Model Training Time (minutes)	Total Training Time (minutes)
			Mean	SD	Mean	SD					
Proposed	OSER (Bayesian 3D CNN)	3D Shape Error Voxelization	0.05	0.03	0.98	0.01	Aleatoric & Epistemic Uncertainty	Epistemic uncertainty sampling (2000 samples)	4400	424	4824
	OSER (3D CNN)	3D Shape Error Voxelization	0.05	0.01	0.98	0.009	No	Random Sampling (4000 samples)	8800	268	9068
Deep Learning	Multi-View 2D CNNs (MV-CNNs)	6 face projection and 2D gridding	0.08	0.02	0.94	0.01	No	Random Sampling (4000 samples)	8800	321	9121
	Depth Based 2D CNNs	1 face projection and 2D gridding	0.12	0.04	0.93	0.02	No	Random Sampling (3000 samples)	6600	248	6848
	Deep Dense Neural Networks	Flattening	0.28	0.09	0.91	0.07	No	Random Sampling (5000 samples)	11000	358	11358
Machine Learning	Gradient Boosted Trees	PCA	0.26	0.08	0.93	0.08	No	Random Sampling (3000 samples)	6600	120	6720
	Random Forests	PCA	0.29	0.09	0.92	0.08	No	Random Sampling (3000 samples)	6600	136	6736
	Support Vector Regression	PCA	0.38	0.09	0.85	0.1	No	Random Sampling (2500 samples)	5500	180	5680
Currently used linear approaches	Regularized Linear Regression	PCA	0.41	0.01	0.76	0.01	No	Random Sampling (1600 samples)	3300	10	3310

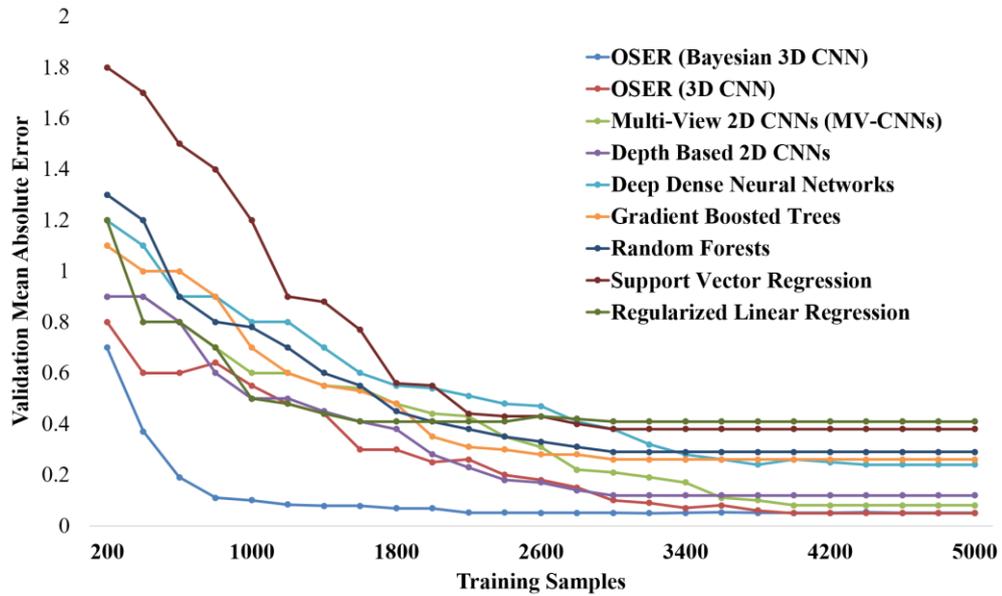


Fig. 3.10. Convergence comparison for all benchmarking models

RCA with Uncertainty Quantification: As discussed in Section 1.2, the identified RCA frequently leads to costly corrective actions conducted in the manufacturing environment. Therefore, it is crucial, especially for 6-sigma faults, to have a decision-driven RCA directed toward informing choices by uncertainty quantification of solving problems. The OSER methodology provides standard deviation of the predicted process parameter distributions $\sigma(\hat{y})$ that quantifies this uncertainty hence, fulfilling requirement (v). Although the performance of the OSER with 3D CNN and OSER with Bayesian 3D CNN models are similar, the latter can quantify and segregate the aleatoric and epistemic uncertainty while estimating the process parameters. Demonstration of the capability of the model in quantifying the uncertainty on unseen samples is done by evaluation on 500 samples within the training range [-1 mm, 1 mm] and 500 samples outside of the training range [-2 mm, 2 mm]. The standard deviation across all observations has been averaged and compared for each process parameter y_1, \dots, y_6 . Results are shown in Fig. 3.11. Fig. 3.12 represents the process parameter distribution for within-training, and out-of-training range prediction, the prediction standard deviation for the out-of-training sample is comparatively higher. The uncertainty is caused due to the non-linear impact of various process parameters in the output point cloud data. To further validate the model's capability to quantify uncertainty, the model is given as input out-of-distribution (OoD) samples from a different case study. The model is able to exhibit that it has not been trained for this case study by providing

uncertainty estimates approximately comparable to the error when estimating process parameters by random guessing. Additionally, the epistemic uncertainty estimates enable closed-loop training to reduce overall training time.

2. OSER vs approaches at design phase when NO point cloud data is available

– In manufacturing environments, the availability of a comprehensive dataset inclusive of all fault scenarios is not feasible, hence augmenting the dataset with high-fidelity multi-physics simulation enables training and deployment of deep learning approaches during the design phase of a new product/production system introduction. Given that the proposed OSER approach transforms the simulation mesh nodes output and scanned point cloud output to the same voxelized shape error compatible with 3D CNN, it enables this integration, hence fulfilling Requirement (vi). This provides the capability to model and simulate the assembly process and conduct system diagnosability and resilience analysis. Currently, no approaches providing this capability for object shaper error RCA at the design phase.

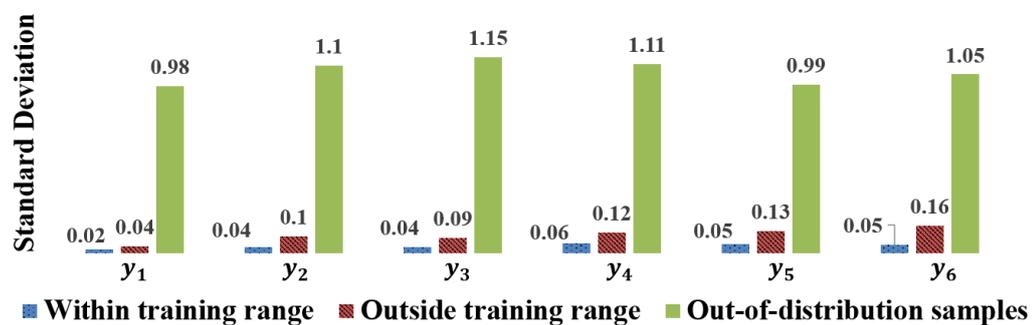


Fig. 3.11. Process Parameters Distribution Standard Deviations

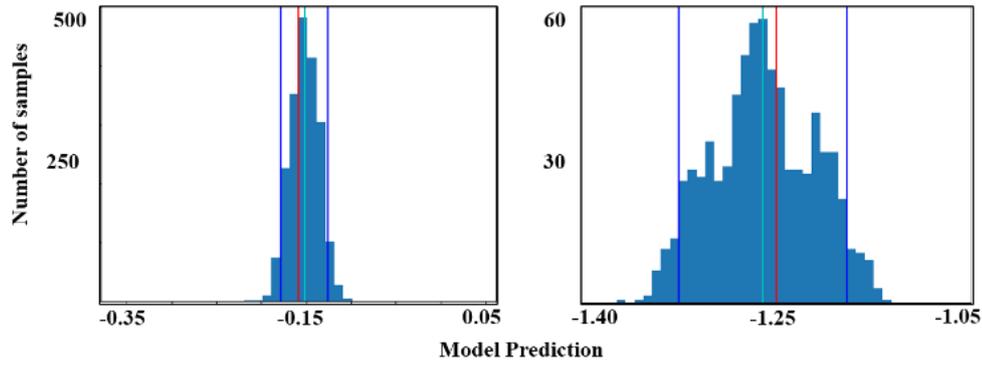


Fig. 3.12. Process parameter distributions

Left: Process parameter distribution histogram for within-training range sample for y^1 (*Actual Value* = -0.16); Right: Process parameter distribution histogram for out-of-training range sample for y^1 (*Actual Value* = -1.23), Note Red line: actual value; Green line: predictive mean; Blue lines: 95% confidence estimates assuming normally distributed predictive posterior.

3.5 Conclusions

This chapter presented an Object Shape Error Response (OSER) approach which is relevant to manufacturing industries where dimensional and geometric variations can be quantified as object shape errors. This is also relevant to robotics, computer-aided detection, stamping, machining, and additive manufacturing. RCA of dimensional variations for these applications translates to estimating object shape error patterns and relating them to process parameters. The proposed approach leverages a Bayesian 3D CNN model trained within a closed-loop framework using a multi-physics simulation (VRM) model to estimate shape errors and relate them to process parameters while quantifying uncertainty. This can then be deployed on physical data collected from 3D surface scanners, thereby, enabling more effective and efficient decision-making to control and correct manufacturing systems. The approach is benchmarked against state-of-the-art assembly RCA models and other machine learning models to highlight statistically significantly better model performance while fulfilling the manufacturing system design requirements. Leveraging such automated RCA models ensures early estimation and elimination of process variations before they become defects, improving the quality and productivity of the system by reducing scrap and machine downtime. This also eliminates the need for trial and error approaches for root causes analysis, often ineffective and inefficient.

4 Root Cause Analysis of Multi-Station Assembly Systems with Non-Ideal Compliant Parts using Bayesian 3D U-Nets²

4.1 Introduction

As shown in Chapter 3, the Bayesian 3D CNN approach of the OSER kernel effectively estimates sources of object shape error within single-station assembly systems. This chapter further proposes the novel OSER-MAS kernel to firstly localize assembly stations at fault, and further isolate RCs of dimensional and geometric product shape errors for *multi-station assembly systems*. The proposed OSER-MAS approach leverages Bayesian 3D U-Net based CNN architecture and is integrated with CAE simulations. The previously proposed 3D CNN architecture in OSER could estimate real-valued process parameters enabling it to isolate RCs in single-station assembly systems, while the 3D U-Net proposed in OSER-MAS approach can estimate real-valued and binary process parameters as well as upstream shape errors by using multiple output heads. This enables (i) fault localization, (ii) isolation of RCs with uncertainty quantification and (iii) learning during the design phase of assembly system when no measurement data is available. The approach relates the estimated shape errors to three categories of process parameters: (i) incoming parts variation; (ii) place-clamp-fasten-release (PCFR) assembly cycle; and, (iii) part-to-part contact chain. The approach overcomes fundamental limitations of current approaches by (i) addressing root cause analysis of dimensional and geometric defects for MAS with non-ideal parts using deep learning on point cloud data; and, (ii) providing capabilities to estimate upstream shape errors and process parameters. Benchmarking is done using an industrial, automotive cross-member MAS. The kernel fulfils requirements (i)-(vi) within the scope of multi-station assembly systems and also fulfils requirements (vii)

² Based on S. Sinha, P. Franciosa, and D. Ceglarek, " Root Cause Analysis of Multi-Station Assembly Systems with Non-Ideal Compliant Parts using Bayesian 3D U-Nets," in review, IEEE Transactions on Automation Science and Engineering, 2021.

and (viii). The contributions of the chapter can be summarized as follows:

- (1) Propose a novel *Bayesian 3D U-Net Architecture* with a *probabilistic encoder, an attention-based decoder* and multiple output heads that enable simultaneous estimation of (i) object shape error components of upstream shape errors, (ii) real-valued process parameters and (iii) binary process parameters. This serves as the critical capability to enable RCA of MAS.
- (2) Propose a novel methodology for RCA of MAS using the outputs of the Bayesian 3D U-net to sequentially perform (i) fault isolation, (ii) fault localization and (iii) fault identification.
- (3) *Verify and validate* the methodology by (i) linking the deep learning architecture with requirements of RC isolation in MAS; and (ii) implementing it on an industrial, automotive cross member assembly consisting of three stations, 25 binary and 123 real-valued process parameters and four non-ideal parts using a *closed-loop training* approach that leverages a CAE simulator known as VRM [6].
- (4) *Benchmark* the proposed methodology against (i) current state-of-the-art RCA models; (ii) traditional machine learning models in a multi-output regression or classification settings; and (iii) state-of-the-art deep learning models used for image segmentation on three levels, namely (i) *Process Parameter Estimation*; (ii) *Uncertainty Quantification*; and, (iii) *Object Shape Error Estimation* to highlight the ability to fulfil the aforementioned eight requirements.

The rest of the chapter is organized as follows; Section 4.2 formulates the object shape error estimation problem for multi-station assembly systems; Section 4.3 discusses the Bayesian 3D U-Net architecture and the overall closed-loop framework required to train and deploy the model; Section 4.4 presents the industrial case study, and finally, conclusions are summarized in Section 4.5.

4.2 Problem Formulation

Multistage manufacturing systems (Fig. 4.1) with a single assembly station have been represented as state-space models in Chapter 3. Each stage corresponds to operations, namely positioning, clamping, fastening and release (PCFR) (Fig. 4.1a). On the other

hand, MAS can be considered a process tree consisting of multiple interconnected stations (Fig. 4.1b). The process tree consists of a set of upstream station and the final station after which data is collected. The input to each station is a set of incoming parts (objects) that need to be assembled. Shape errors within the objects can be induced due to variation in any process parameter(s) within the upstream stations [51] and are propagated through the process tree to the final station.

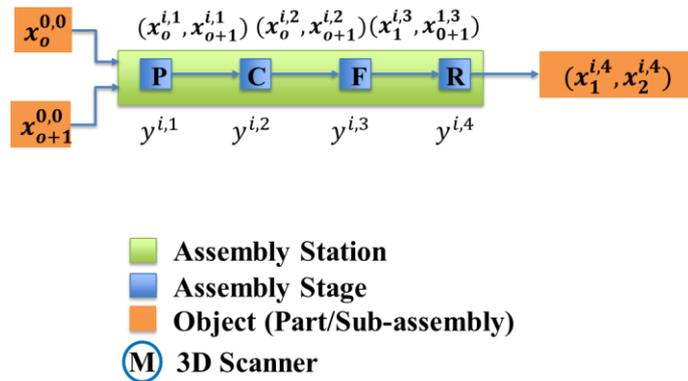


Fig. 4.1a. An assembly station with PCFR stages

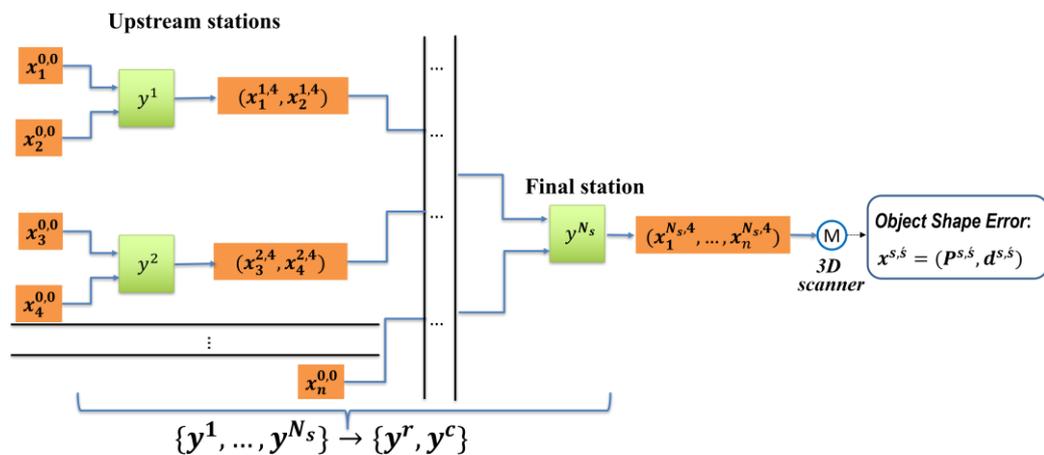


Fig. 4.1b. Process tree for multi-station assembly systems

Fig. 4.1. Multi-station assembly system

Previously the formulation of object shape error propagation in single station (multistage) assembly has been proposed in Chapter 3 that represents shape error x_o^s for object o after stage s as:

$$\mathbf{x}_o^s = (\mathbf{P}_o^s, \mathbf{d}_o^s) \quad (4.1)$$

Where \mathbf{P}_o^s represents set of nominal points and \mathbf{d}_o^s represents the three components (x, y and z) of shape error for each nominal point.

As the goal of this chapter is RCA for MAS, the chapter extends the problem formulation for single station assembly systems by considering multiple stations. Stations are represented by s : $s = 1, \dots, N_s$, where N_s represents the total number of stations within the system, and the previously represented stages are represented as \acute{s} : $\acute{s} = 1, 2, 3, 4$ as each station has four stages. Hence, for MAS object shape error for object o after station s and stage \acute{s} can be represented as:

$$\mathbf{x}_o^{s,\acute{s}} = (\mathbf{P}_o^{s,\acute{s}}, \mathbf{d}_o^{s,\acute{s}}) \quad (4.2)$$

$\mathbf{x}_o^{0,0}$ correspond to shape errors from upstream part variations which are also addressed as non-ideal part variations [9]. A set of objects (sub-assembly) after stage \acute{s} of station s is represented as $(\mathbf{x}_1^{s,\acute{s}}, \dots, \mathbf{x}_o^{s,\acute{s}})$.

Process parameters within a station and stage are represented as $\mathbf{y}^{s,\acute{s}}$, process parameters within a station is represented as \mathbf{y}^s , while the entire set of h process parameters can be represented as a vector \mathbf{y} . Potentially variation in each process parameter is a source of shape error and must be isolated as a root cause. In MAS, these process parameters are classified into three categories [191]: (a) Real-valued parameters of incoming parts (objects) variation as caused by upstream fabrication processes such as stamping, extrusions, etc.; (b) Real-valued process parameters related to PCFR stages of each assembly station. They represent any deviation from nominal in fixturing/tooling or joining operations; and (c) Binary joining-based process parameters in the fastening stage indicate the joint's success. The value is $\{1\}$ when joint is successfully completed or $\{0\}$ for an unsuccessful joint due to the excessive gap between objects to be joined or current failure in the tool. The overall vector of h process parameters \mathbf{y} can be split into a vector of r real-valued process parameters \mathbf{y}^r (category (a) and (b)) and c binary process parameters \mathbf{y}^c (category (c)).

For RCA of single station assembly systems performing RCA is equivalent to estimating process parameters of the station based on the shape error at the end of the station. Chapter 3 proposed using a Voxnet [20] based 3D CNN approach to estimate a function $f(\cdot)$ that could estimate only real-valued process parameters as a function of shape error at the end of all stages:

$$\mathbf{y}^r = f(\mathbf{x}^{s=4}) \quad (4.3)$$

In MAS, shape error is propagated through the upstream assembly stations and shape error data is collected only at the end of the final station $\mathbf{x}^{s=N_s, \acute{s}=4}$. To perform RCA in such scenarios, firstly, the RC(s) must be localized to the originating station, followed by which RC(s) can be isolated within the localized station. Hence, to comprehensively perform RCA, shape errors for upstream stations $[\mathbf{x}^{1,4}, \dots, \mathbf{x}^{N_s-1,4}]$ need to be estimated, which enable localization of the station and further binary and real-valued process parameters must be estimated to isolate RC(s) within the station. Therefore the chapter proposes a Bayesian 3D U-net model which can be trained to learn a function $f(\cdot)$ that takes as input the combined object shape error at the end of the system $\mathbf{x}^{N_s,4}$, i.e., after the final stage of the last station ($s = N_s, \acute{s} = 4$), and estimates the process parameters across the entire system and the object shape error for all objects at the end of the upstream stations:

$$[\mathbf{y}^r, \mathbf{y}^c, \mathbf{x}^{1,4}, \dots, \mathbf{x}^{N_s-1,4}] = f(\mathbf{x}^{N_s,4}) \quad (4.4)$$

Further, a methodology is proposed that leverages the outputs of $f(\cdot)$ to perform RCA.

4.3 Methods

4.3.1 3D Object Shape Error Voxelization

For the application of CNN based models to estimate $f(\cdot)$ as in (4.4), the shape error needs to be transformed to a regular structure; hence, the output shape error $\mathbf{x}_o^{s,\acute{s}} = (\mathbf{p}_o^{s,\acute{s}}, \mathbf{d}_o^{s,\acute{s}})$ is voxelized to voxel grids $\{\mathbf{V}_{u,v,w}\}$ with discrete voxel coordinates (u,v,w) in the following way: for all points $\mathbf{p}_k = (x_k, y_k, z_k)$ that fall within a voxel grid

$\{\mathbf{V}_{u,v,w}\}$ the maximum value of $\mathbf{d}_k = (\tilde{x}_k, \tilde{y}_k, \tilde{z}_k)$ across all points mapped to the same voxel, characterizes the features of the corresponding voxel grid and is represented as $\{\mathbf{V}_{u,v,w,d}\}$ [20] [192]. Similar processing is done for previous station shape errors $\mathbf{x}^{1,4}, \dots, \mathbf{x}^{N_s-1,4}$ that need to be estimated by the U-Net model. As opposed to traditional binary image 2D segmentation maps that are generated by U-Net models, the proposed model estimates real-valued multi-channel 3D shape error maps for all previous stations and all components of object shape error.

4.3.2 Bayesian Neural Networks

Given the uncertainties in the system and the availability of only a limited dataset, a deterministic estimate of function $f(\cdot)$, as shown in (4.4), is not feasible. Therefore, for uncertainty quantification, Bayesian learning enabled by variational inference is used to learn weight distributions instead of only point estimates. This is realized by using Bayes-by-Backprop [164], which integrates backpropagation with variational inference [180] to estimate a posterior distribution $q_\theta(\omega)$ which is parametrized by θ over the neural network weights based on the pre-specified prior $p(\omega)$. The posterior weight distribution parameters θ are obtained by minimizing the variational free energy cost function known as the expected lower bound (ELBO). ELBO includes sum of (i) Kullback-Leibler (KL) divergence (4.5) which is the divergence between the true posterior and its variational approximation, and (ii) negative log-likelihood (NLL) of the dataset which measures the goodness-of-fit of the model.

$$KL(q_\theta(\omega)||p(\omega|\mathbf{x}^s, \mathbf{y})) = \int q_\theta(\omega) \log q_\theta(\omega)/p(\omega|\mathbf{x}^s, \mathbf{y}) d\omega \quad (4.5)$$

While training in a stochastic manner using a mini-batch of examples, the Flipout [89] estimator is used to solve the challenge associated with similarly sampled weights within a mini-batch. Flipout achieves ideal variance reduction by sampling weights pseudo-independently for each example.

4.3.3 Bayesian 3D U-Net Convolutional Architecture

The Chapter extends encoder-decoder u-net architectures to design an architecture that can accurately estimate the outputs for $f(\cdot)$ as shown in (4.4) (Fig. 4.2). The architecture consists of an encoder, a decoder and two output heads at the encoder's end. The encoder consists of four levels (Fig. 4.2). Each encoder level consists of down-sampling kernels (Fig. 4.2 left). The first level takes as input the voxelized shape error ($x^{N_s,4}$) while subsequent levels take as input the output of the previous level. A 3D Max pooling is performed within the kernel, which is duplicated to a residual and encoding connection. A 3D convolution operation with a filter size of one and a stride length of one is performed within the residual connection. Two 3D convolutions (with ReLU activations) of filter size three and stride length one are performed within the encoding connection. These are further merged using element-wise addition followed by ReLU activation. The merged features are then duplicated and given as input into the corresponding level decoder and next level encoder. Overall, the encoder enables spatial correlation filtering, feature extraction, and non-linear transformations. Consecutive levels of the encoder extract more discriminating features from the high-resolution voxelized shape error input. The discriminative ability of the features increases at each consecutive encoder level. The link to the corresponding decoder enables transfer of features related to the part geometry, thus, enabling accurate estimation of upstream part shape error at the decoder's end. *From a MAS engineering perspective, the encoder can be considered as a feature extractor that enables the synthesis of the MAS, i.e. it extracts the necessary shape error features from the output of the MAS that can be leveraged to estimate process parameters and upstream shape errors.*

The decoder levels consist of up-sampling kernels (Fig. 4.2 right). Each level of the decoder takes two inputs, the encoder input from the corresponding level encoder and the decoder input from the previous decoder level. The decoder input is then up-sampled using upsampling and 3D convolution operations. The operations collectively upsample and add necessary detail into the decoder input. The upsampled features are duplicated and sent to the attention gate and feature concatenation layer. The attention gate [90] distills features from the corresponding encoder and then generates relevant

features that are concatenated with the upsampled features. The concatenated feature set is duplicated to the residual connection, and the decoder connection and similar operations as in the encoder layer are performed. The decoder output dimension is equal to the number of components of shape error multiplied by the number of upstream stations, and the voxel granularity of the output is equal to the input size. Each level of the decoder consolidates part geometry features (corresponding encoder input) with the shape error features (previous decoder input), enabling accurate estimation of upstream shape errors. In the context of object segmentation, the decoder outputs real-valued segmentation maps that estimate three components of shape errors for all upstream stations. *From a MAS engineering perspective, the decoder can be considered as a feature consolidator that enables the synthesis of real-valued segmentation maps that estimate upstream shape errors ($\mathbf{x}^{1,4}, \dots, \mathbf{x}^{N_s-1,4}$).*

The architecture consists of two output heads at the end of the encoder; one head estimates real-valued process parameters \mathbf{y}^r as done in a regression setting, while the second head estimates binary process parameters \mathbf{y}^c as done in a classification setting. The feature maps at the end of the decoder are given as input to both heads. 3D convolution and global-max pooling operations are performed within each head, followed by a Dense Flipout layer operation with 64 nodes and ReLU activation. The number of regression output nodes in the regression head is equal to the number of real-valued parameters. The number of classification output nodes is equal to the number of binary process parameters. *From a MAS engineering perspective, the output-heads can be considered enablers for RCA across multiple domains of process parameters, including (a) real-valued parameters for non-ideal parts and fixturing/tooling; and (b) binary parameters for joining operations.*

The attention gate (Fig. 4.2 right-down) at each level of the decoder enhances the architecture's ability to perform synthesis of MAS. As proposed by Oktay et al. [24], the soft-attention mechanism is used between corresponding levels of the encoder and decoder. Attention enables the model to be specific to local regions. The attention mechanism allows the model to reweight the input features based on a given set of features. The mechanism leverages a convolutional layer with the output having the

same dimensions as the number of incoming features. The outputs of the network go through a sigmoid activation to generate values that lie between 0 and 1 and are known as attention weights. These attention weights are multiplied by the input features, which reweights the features and helps the model be specific to certain input features. In the proposed 3D U-net, the attention weights are generated based on the input from the previous decoder level. These are then used to generate attention weights which are used to reweight input features coming from the corresponding encoder. This helps the model focus on particular parts/subassemblies as the incoming features about the geometry of the part/subassembly are reweighted based on the previous level decoder features. The attention gate increases model performance as it learns on which part/subassembly to focus on to estimate upstream shape errors accurately. *From a MAS engineering perspective, attention gates improve performance for upstream synthesis and provide enhanced and calibrated estimates of upstream shape errors.* Fig. 4.3 aims to summarize the link between the functionalities of different elements of the 3D U-net architecture and the MAS engineering challenges that they fulfil. The residual connections also enhance synthesis performance by eliminating the vanishing gradients problem using residual [193] or skip connections within down-sampling and upsampling kernels, ensuring gradient propagation. Finally, the architecture leverages Bayesian layers within the architecture. Flipout [89] layers within the encoder and Bayes-by-Backprop [164] which combines backpropagation with variational inference, are leveraged for uncertainty quantification. *From a MAS engineering perspective, uncertainty estimates integrate confidence measures within isolated RC(s) and drive costly corrective actions [15].* The previously proposed architecture (Chapter 3) for RCA of assembly systems only consisted of an encoder with a regression head limiting its application scope to single-station assembly with only real-valued process parameters. The proposed architecture includes multiple output heads and a decoder that broadens the OSER-MAS approach's application scope to *assembly systems with multiple stations with both real-valued and binary process parameters.*

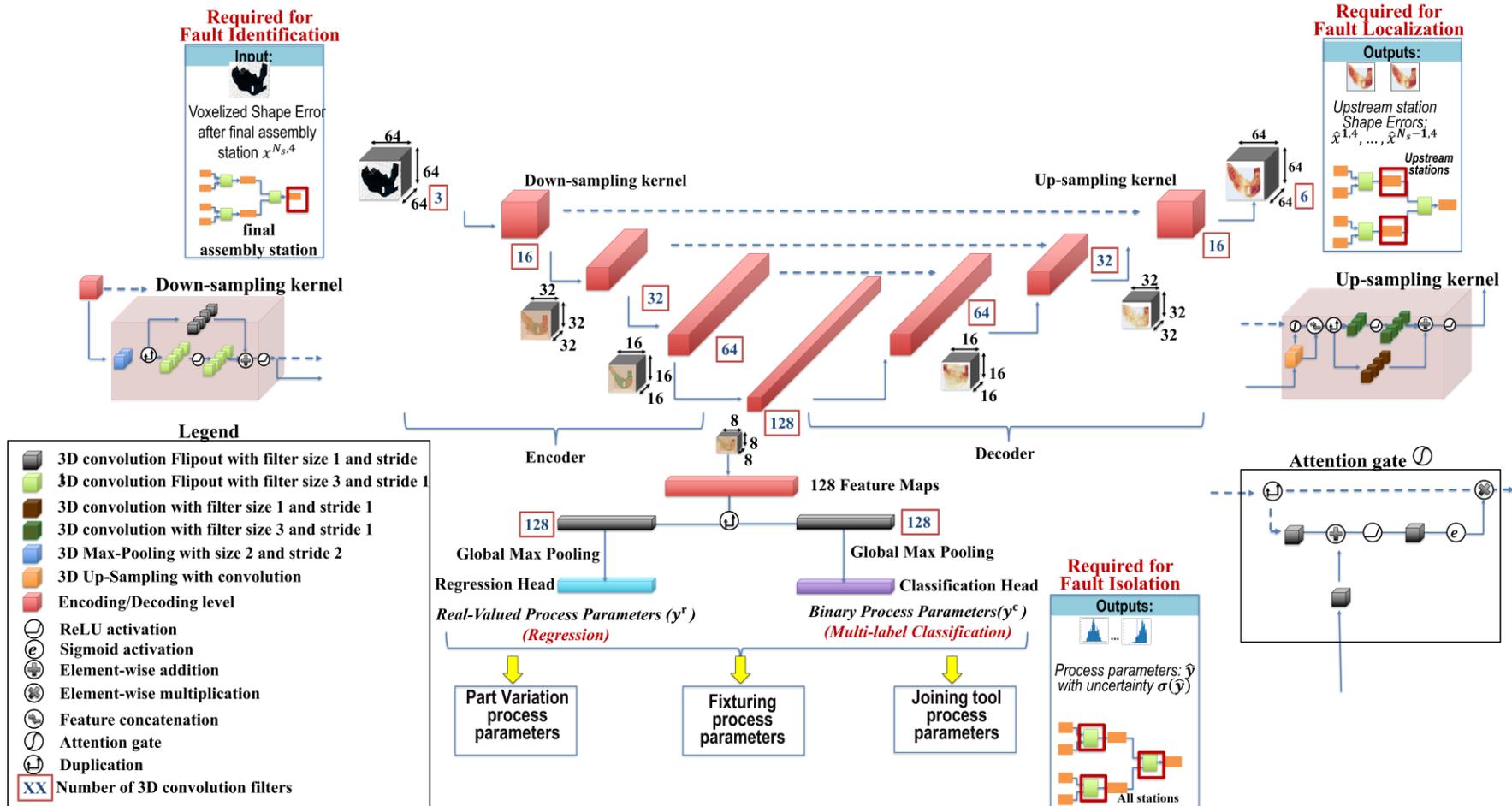


Fig. 4.2. Bayesian 3D U-Net Architecture
 left: down-sampling kernel, right: up-sampling kernel, right-down: attention gate

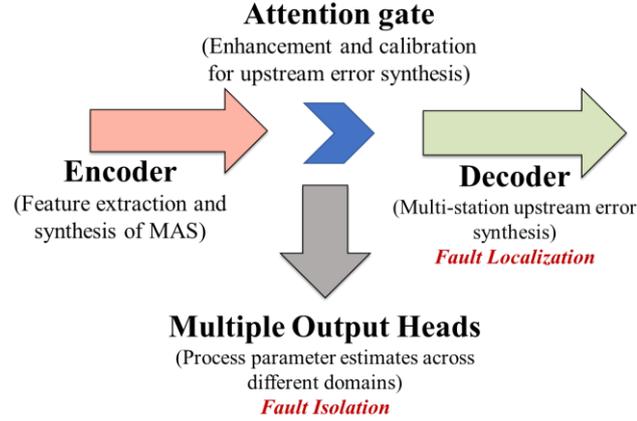


Fig. 4.3. Linking Bayesian 3D U-Net with MAS engineering

4.3.4 Architecture Design Decisions

The proposed architecture was designed by making key changes to the base architecture proposed for 2D biological image segmentation [88], which was later extended for 3D images [91] and further incorporated residual connections [194]. The design changes were done based on object shape error estimation requirements while accounting for standard guidelines of 3D U-Net architecture optimizations. Architecture hyperparameter optimization using grid search exhaustively searches for the best set of hyperparameters and has high computational costs [195]. Hence, to perform this optimization in a computationally feasible manner, critical decisions are taken sequentially. The first key decision involved the total depth (Number of levels) N_L and the number of filters in each level : $l = 1, 2 \dots, N_L$. The U-Net standards require the number of filters in each level to be constraint at values 2^{F+l} for the l^{th} level. Grid search (Fig. 4.4) was performed for $N_L = \{3, 4, 5\}$ and $F = \{2, 3, 4\}$ to determine the best configuration. $N_L = 4, F = 3$ gave the best performance (boxed in red) in terms of the minimum value of the weighted loss function. The weighted loss function includes the losses all the outputs of the network including real-valued (regression) and binary process parameters (classification) and upstream shape errors (real-valued segmentation) is discussed later in model training. The mathematical formulation of the loss function is discussed in Section 4.9. The next decision step involved experimenting with the filter size $F_s = \{3, 5, 7\}$. For computation feasibility, the filter size was kept universal across the network. $F_s = 3$ gave the best performance given the higher resolution and more granular feature extraction due to small filter size. Residual

connections were added in each level of the encoder and decoder, this increased performance as these help in efficient propagation of gradients in deep networks while model training. To further increase performance experimentation was done by adding the soft-attention mechanism as proposed by Oktay et al. [90] between corresponding levels on the encoder and decoder. The attention approach allows the model to be specific to local regions. In the context of shape error estimation of assemblies, this helps the model focus on particular parts/subassemblies in each station thereby, enabling fault localization. Adding of the attention gate provides a significant increase in accuracy of upstream stations shape error estimation $[x^{1,4}, \dots, x^{N_s-1,4}]$. The final step involved replacing standard layers in the encoder layers with Flipout layers for uncertainty quantification. As opposed to [196] which leverages Flipout layers in the decoder our architecture uses Flipout layers in the encoder as this enables uncertainty quantification for the process parameters that are estimated at the end of the encoder. For the regression and classification heads, experiments were done using global average pooling and global max pooling. Global Max pooling gave superior performance given the ability to extract the most significant feature. After this, the KL penalty for uncertainty quantification was added in the loss function as shown in (4.5), and the model was trained. Experiments are done with three priors in the Flipout layers, namely, standard normal, scale mixture and spike and slab. An uncertainty calibration study that involved a comparison of model error and model uncertainty was conducted to ensure that the model predicts high uncertainty for samples with high error and vice-versa. Standard Normal prior gave the most calibrated uncertainty measure compared to other priors and was selected as the prior distribution for all weights in the encoder.

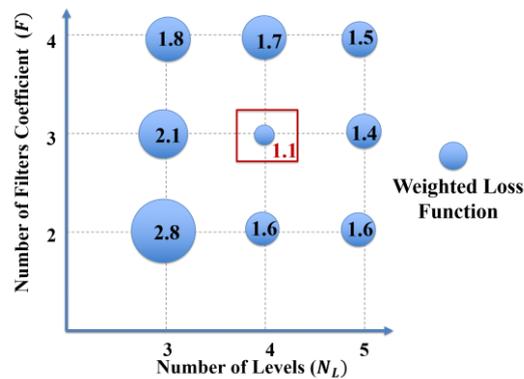


Fig. 4.4. Grid search results for U-Net architecture optimization $N_L = 4, F = 3$ (boxed in red) is selected as the best model architecture

4.3.5 Closed-Loop Training and Deployment

Training of the model to estimate posterior distribution parameters (ω) of weights and biases of $f(\cdot)$ is done in a closed-loop [191] (discussed in detail in Chapter 5) using data generated by VRM. The key motivation behind using a closed-loop framework instead of an open-loop framework is to minimize the computational bottleneck of the VRM simulator. Although this increases the number of iterations, the overall time of VRM simulation and training is significantly reduced as fewer samples need to be generated. The key steps of the proposed framework are summarized below (Fig. 4.5):

Step 1 – Sampling: Closed-loop sampling enables the dynamic and adaptive generation of training samples based on the uncertainty and error of the previous iterations while ensuring that the sample generation has a degree of randomness to prevent overfitting. Initially, Latin Hypercube Sampling (LHS) is used given it distributes samples optimally across the h –dimensional process parameter space by stratifying the possible ranges. Further sampling is done on the basis of the uncertainty and error of the previous iterations while ensuring that the sample generation has a degree of randomness to prevent overfitting.

Step 2 – VRM Simulation: The set of process parameters generated during sampling are used as input to the VRM to simulate the assembly process and generate the output mesh from which the shape errors are extracted after each desired station of the assembly system.

Step 3 – Model Training and Evaluation: The model is trained using a weighted sum of four loss functions to estimate process parameters, shape errors and uncertainty. Descriptions i) to iv) summarizes the loss value for each training sample, while training all these are averaged across the minibatch:

i) *Negative Log-likelihood* for the real-valued process parameters \mathbf{y}^r that are modelled using a multivariate normal with a diagonal matrix to quantify aleatoric uncertainty. Aleatoric uncertainty (known unknowns) quantifies uncertainty due to factors such as noise, while epistemic uncertainty (unknown unknowns) quantifies uncertainty due to

factors such as model structure and limited data availability [192].

$$L^1 = 1/2 [\ln(|\Sigma|) + (\mathbf{y}^r - \boldsymbol{\mu})^T \Sigma^{-1} (\mathbf{y}^r - \boldsymbol{\mu}) + h \ln(2\pi)] \quad (4.6)$$

ii) *Binary cross-entropy* for the binary process parameters \mathbf{y}^c

$$L^2 = \mathbf{y}^c \ln \widehat{\mathbf{y}}^c + (1 - \mathbf{y}^c) \ln(1 - \widehat{\mathbf{y}}^c) \quad (4.7)$$

iii) *Mean squared error* for the decoded object shape error

$$L^3 = \sum_{s=1,2,\dots,N_s-1} (\mathbf{x}^{s,4} - \widehat{\mathbf{x}}^{s,4})^2 \quad (4.8)$$

iv) *KL divergence* (L^4) as shown in (4.5) for all network weights and biases distribution parameters to quantify epistemic uncertainty. The final loss function (L) used while training is a weighted sum of all the above-specified loss functions:

$$L = \sum_{l=1}^4 w^l \cdot L^l \quad (4.9)$$

In usual practice, the values for these weights for the loss function are determined by empirical tuning, considering them as hyperparameters for optimization, but this can result in the requirement of exhaustive computational budget and time. The paper leverages the homoscedastic uncertainty for multi-task learning approach [168]. The approach estimates the weights during the training process without the need for any manual tuning. The approach aims to quantify the task-based uncertainty depending on the data and leverages the uncertainty estimate as the basis for weighting losses in a multi-task learning problem. The approach has been shown to provide superior performance in multi-task settings involving regression, classification and semantic segmentation as compared to performing each task individually. Leveraging this approach ensures that extensive fine-tuning of the weights is not required for training the model on different MAS. The loss function weights $\{w^1, w^2, w^3\}$ are generated using this approach. Monotonic KL annealing is used for w^4 to ensure that the model

first learns the necessary estimation capabilities for the process parameters and upstream shape errors before the KL penalty is applied for epistemic uncertainty quantification. This also stabilizes the training process. Adam optimizer [185] is used for training, and exponential decay of the learning rate is applied. A Minibatch size of 32 is used, and the model is trained for 300 epochs. Group normalization [183] with four groups is used after each convolutional layer. This prevents overfitting and accounts for small minibatch size due to GPU memory size constraints and aids in stabilizing the training process.

After each iteration of training, the model is evaluated on the validation set. For evaluation, Monte Carlo (MC) sampling from the model is done, and the sample means $\bar{\mathbf{y}}$ and standard deviations $\sigma(\hat{\mathbf{y}})$ are estimated for all real-valued and binary process parameters. $\sigma(\hat{\mathbf{y}})$ represents the epistemic uncertainty, while the fixed scale parameters of the predictive distribution for real-valued process parameters represent the known aleatoric uncertainty [36]. Given the assumption of negligible measurement noise, aleatoric uncertainty is considered to be negligible, and hence, the overall uncertainty in the prediction can be assumed to be equal to epistemic uncertainty $\sigma(\hat{\mathbf{y}})$. Similarly, means $\hat{\mathbf{x}}^{s,s}$ and standard deviations $\sigma(\hat{\mathbf{x}}^{s,s})$ of all object shape errors at upstream stations are calculated. The model is trained until the error is below the set threshold. The thresholds are set based on assembly requirements. For example, in the automotive body assembly process, the threshold is set considering tolerances within [-1mm,1mm]. The 3D optical scanner used has a repeatability of 0.05 mm and accuracy within 0.15 mm. During training, the object shape errors' uncertainty is estimated but is not considered for sampling as the VRM requires process parameters as input.

Final evaluation and comparison for all model outputs are made using the known actual values and the estimated means. Real-valued process parameters are evaluated on Mean Absolute Error (MAE) and R-squared (R^2). Binary process parameters are evaluated on Accuracy and Receiving Operating Characteristics – Area Under Curve (ROC-AUC). Given object shape errors are real-valued, they are also evaluated on MAE, R^2 and Root Mean Squared Error (RMSE).

Step 4 – Model Deployment: After training, the model can be deployed for an existing system. The product shape data is collected from the 3D scanner after the final station, and then the data is aligned to obtain the final shape error $x_o^{s,s} = (P_o^{s,s}, d_o^{s,s})$ and finally, voxelized $V_{u,v,w,d}$ before it can be given to the trained model for estimating process parameters and shape error from all upstream stations. Inferencing estimates the process parameters and shape errors for a given $x_o^{s,s}$ using MC sampling from the trained model. Using these samples, process parameters (distribution mean) \bar{y} and their uncertainty (distribution standard deviation) $\sigma(\hat{y})$ can be estimated. The sample mean \bar{y} is considered as the model estimate \hat{y} , while $\sigma(\hat{y})$ quantifies the uncertainty. Similarly, $\hat{x}^{s,s}$ estimates the shape error and $\sigma(\hat{x}^{s,s})$ the uncertainty in the estimated shape error. Further, fault localization and RCA can be done using these estimates. The work has been implemented using Python 3.7 and TensorFlow - GPU 2.1 [186] and TensorFlow Probability 0.9. A python library [187] has been developed to validate and replicate the results of the methodology. Both the data generation and evaluation of the OSER-MAS methodology have been done using VRM. Two Nvidia Tesla V100 32 GB GPUs are used for model training and deployment.

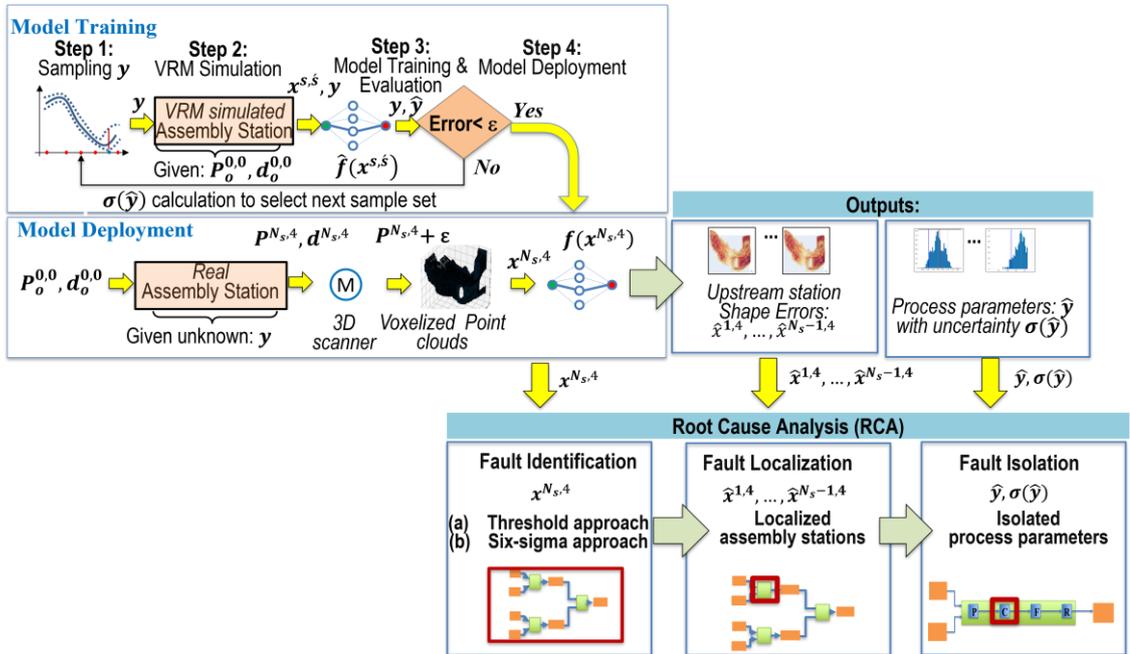


Fig. 4.5. Closed-loop training and deployment with RCA

For this Chapter, both the data generation and evaluation of the OSER-MAS methodology have been done using VRM. Although the VRM has been previously

verified and validated for the simulation of assembly systems, constraints in the computational budget might inversely hamper the data quality due to modelling assumptions such as mesh granularity and boundary conditions. In such situations, the data generated might not be representative of a physical assembly system. *Uncertainty quantification (req. (v)) capability* is crucial for the recognition of such situations as the model will indicate high uncertainty when the distribution of the physical data is different from the VRM data used while training. In such situations, the model should be fine-tuned using data collected from the physical MAS. The Bayesian nature of the proposed architecture enables effective fine-tuning using uncertainty-guided continual learning and ensures that the model can be continually adapted to changes in the MAS while minimizing catastrophic forgetting (discussed in detail in Chapter 5).

4.3.6 Root Cause Analysis

After model deployment, the measured point cloud data $[\mathbf{x}^{N_s,4}]$ and the estimates of the model $[\mathbf{y}^r, \mathbf{y}^c, \mathbf{x}^{1,4}, \dots, \mathbf{x}^{N_s-1,4}]$ are leveraged to perform RCA for MAS. Building on previous RCA models that leveraged point-based measurement systems [197], the chapter proposes three key steps, namely: (i) fault identification; (ii) fault localization; and, (iii) fault isolation. The proposed approach in Fig. 4.5 can be leveraged to estimate single or multiple faults within the MAS.

(i) *Fault Identification*: The first is identifying if the MAS is at fault. In practice, all systems are at fault due to inherent variations and tolerances. Significant faults are identified by analysing the output $[\mathbf{x}^{N_s,4}]$ of the MAS. Two approaches can be leveraged, namely: (a) *Threshold approach*: In this approach, a fault is identified if the measured point cloud is beyond a particular a set threshold. The thresholds are determined based on the tolerance limits of the assembled products. This approach is prone to false alarms as a single outlier in the output may indicate faults even when the no process parameter within the MAS is at fault. (b) *Six-sigma approach*: In this approach, a fault is identified by analyzing the statistics of a sample of assembled products. The mean and variance of the output $(\mathbf{x}^{N_s,4})$ for a sample of products is calculated, a mean shift or a change in variance (heteroskedasticity) can be identified as a fault based on the significance level used. This approach is robust to outliers and

can be leveraged for continuous improvement of the system.

(ii) *Fault Localization*: If the MAS has been identified at fault, the estimates of the upstream shape errors $[\mathbf{x}^{1,4}, \dots, \mathbf{x}^{N_s-1,4}]$ as obtained from the deployed model can be leveraged to localize stations within the MAS at fault. This entails identifying particular stations within the MAS process tree within which the object(s) (sub-assemblies) shape error becomes significant. The shape error estimates $[\mathbf{x}^{1,4}, \dots, \mathbf{x}^{N_s-1,4}]$ for all objects $o: o = 1, \dots, n$, are compared with the design nominal. If the shape error for the objects within the station exceeds the threshold (sub-assembly tolerances) at the end of a station but within the threshold for the previous upstream station, then the fault is localized to that particular station for the corresponding station and object. This is done for all stations. The set of localized stations at fault is denoted as \mathbf{s}_F . Table 4.1 summarizes the steps used to localize assembly stations at fault. Using the proposed bottom-up approach ensures that the fault is localized to the originating station. This is crucial as MAS involve re-orientation/re-positioning of objects between stations [51] [197]. Localizing using a bottom-up approach ensures that the station within which the fault originated is localized before the object was re-oriented in later stations.

Table 4.1. Fault Localization

<i>Algorithm 4.1: Fault Localization</i>	
Procedure:	
List of localization stations: $\mathbf{s}_F = \{\}$	
Repeat for all stations $i=0$ to N_s:	
Compare $\mathbf{x}^{i,4}$ to nominal design	
Compare $\mathbf{x}^{i+1,4}$ to nominal design	
If: $\mathbf{x}^{i,4}$ within tolerance and $\mathbf{x}^{i+1,4}$ is beyond tolerance	
Localize station $s = i + 1$	
$\mathbf{s}_F \leftarrow \mathbf{s}_F \cup \{s = i + 1\}$	

(iii) *Fault Isolation*: Fault isolation involves isolating which process parameters within localization stations \mathbf{s}_F are root causes and have a significant impact on the shape error of the final object (product). The set of process parameters (denoted a $\mathbf{y}^{s=\mathbf{s}_F}$) that lie within the localized stations \mathbf{s}_F are compared with the assembly process tolerances and isolated as RCs if they fulfil the below criteria (RC_K): (a) All real-valued process parameters \mathbf{y}^F within the set of localized process parameters $\mathbf{y}^{s=\mathbf{s}_F}$ that are beyond the

assembly process tolerances are isolated as RCs. (b) All binary process parameters \mathbf{y}^c within the localized stations that are estimated as failed (e.g. for joining tool failure $\mathbf{y}^c = 0$) are isolated as RCs. Overall RCs can be denoted as a subset of process parameters within localized stations that:

$$RC_{\kappa} \subseteq \mathbf{y}^{S=S_F} \quad (4.10)$$

Within fault isolation, it is also crucial to consider the uncertainty of the process parameters $\sigma(\hat{\mathbf{y}})$ isolated as RCs drive costly corrective actions. High values of uncertainty indicate the trained model is not confident about the process parameter estimates, which may be due to noise in the measurement systems (aleatoric uncertainty) or lack of similar instances within the training set (epistemic uncertainty) [15]. In such scenarios, it is crucial to consider expert system knowledge to get conclusive insights about the isolated RCs

The OSER-MAS methods, requirements, MAS application and assumptions are summarized in Table 4.2. The overall framework of OSER-MAS inclusive of closed-loop sampling, training and deployment and the RCA framework that leverages the outputs of the trained model to perform fault identification, fault localization and fault isolation is summarized in Fig. 4.5. In real-life applications, all process parameters within MAS have an inherent level of variation. The proposed approach for RCA using the estimates of the proposed architecture is crucial in differentiating benign faults (have no significant impact on the product shape error) from malignant faults (cause product shape error to go beyond assembly thresholds).

Another key consideration for RCA is the existence of a unique set of RCs. Given the high dimensional process parameter space, multiple shape error patterns may be caused by different combinations of RCs (known as collinear RCs [15]). Such scenarios will be very limited in the proposed approach. The input voxelized object shape error data is highly granular; hence, scenarios of different RCs giving the same voxelized object shape error are unlikely to happen. 3D CNN based approaches that leverage granular voxelized point cloud data have been shown to have high discriminative ability for various collinear RCs given the 3D deep learning capabilities

[15]. This contrasts with previously proposed statistical approaches that used simplified representations such as a vector of surface points or sensor readings (low granularity), leading to frequent scenarios when different RCs resulted in the same vector output. Additionally, uncertainty quantification enables the identification of such scenarios as a single shape error caused by multiple RCs would lead to RC distributions that are multi-modal or have high variance. During such scenarios, engineering expertise can be applied to isolate RCs accurately.

Table 4.2. OSER-MAS methods, application and assumptions

OSER-MAS method	Requirement	MAS Application	Assumption
3D object shape error voxelization	(i) High data dimensionality	Enables processing of 3D object shape error	Availability of point cloud data
Encoder	(ii) non-linearity, (iii) collinearities, (iv) high faults multiplicity	Incorporates capabilities for handling the non-linearity and interactions due to multiple stages and stations within MAS	Sufficient discriminative information within the incoming point cloud data and sufficient modelling capabilities within the architecture
Bayesian model	(v) Uncertainty quantification	Enables uncertainty qualification for isolated RC for costly corrective action and enabling closed-loop sampling	Ability of bayes-by-backprop and Flipout to model uncertainty using a normal distribution as prior for each model parameter
Closed-loop sampling	(vi) Dual data generation capabilities	Enables training of the proposed model using a closed-loop framework that generates samples based on uncertainty	Ability of VRM (CAE simulator) to simulate data that is close to a physical MAS
Multiple output heads	(vii) High dimensionality and heterogeneity	Enables estimation of real-valued and categorical RCs across part variations, fixturing and joining	Process parameters are either real-valued or binary
Decoder with attention gate	(viii) Fault localization for RCA	Enables estimation of upstream shape error required to localize faulty stations to enable RCA	Sufficient modelling capabilities within the decoder architecture and the soft attention mechanism
RCA framework	Fault identification, fault localization and fault isolation	Leverages the process parameter and upstream shape error estimates to perform RCA	Six sigma indicators sufficient for fault identification

4.4 Case Study

4.4.1 Multi-Station Assembly Setup

The selected cross member assembly consists of $N_s = 3$ stations and $n = 4$ non-ideal compliant parts, namely, pocket, pocket reinforcement, cross member and cross member reinforcement (Fig. 4.6). The assembly is controlled by $h = 148$ process

parameters (Fig. 4.7) \mathbf{y} including 123 real-valued parameters \mathbf{y}^r and 25 binary parameters \mathbf{y}^c . Table 4.3 summarizes the process parameter and their physical interpretation corresponding to the root causes of the multi-station assembly system. 11875 points characterize the point cloud of the final assembly. Shape error data is collected at the end of all three stations $\mathbf{x}^{1,4}, \mathbf{x}^{2,4}, \mathbf{x}^{3,4}$. The shape error after the final station ($\mathbf{x}^{3,4}$) is used as input while the process parameters \mathbf{y} and upstream stations shape errors $\mathbf{x}^{1,4}, \mathbf{x}^{2,4}$ are used as output. Before training all shape errors are pre-processed and voxelized to $(u, v, w, d) = (64, 64, 64, 3)$ voxel grids $\mathbf{V}_{64,64,64,3}$. The deviation features \mathbf{d} include deviations in all directions for all points $(\tilde{x}_k, \tilde{y}_k, \tilde{z}_k)$. For this particular case, the model has 123 output nodes in the regression head, 25 output nodes in the classification head, and decoder outputs shape errors with dimension $(64, 64, 64, 6)$, corresponding to the shape errors at the end of two upstream stations. The training range for all real-valued process parameters is $[-1 \text{ mm}, 1 \text{ mm}]$ during the validation and testing range is $[-2 \text{ mm}, 2 \text{ mm}]$.

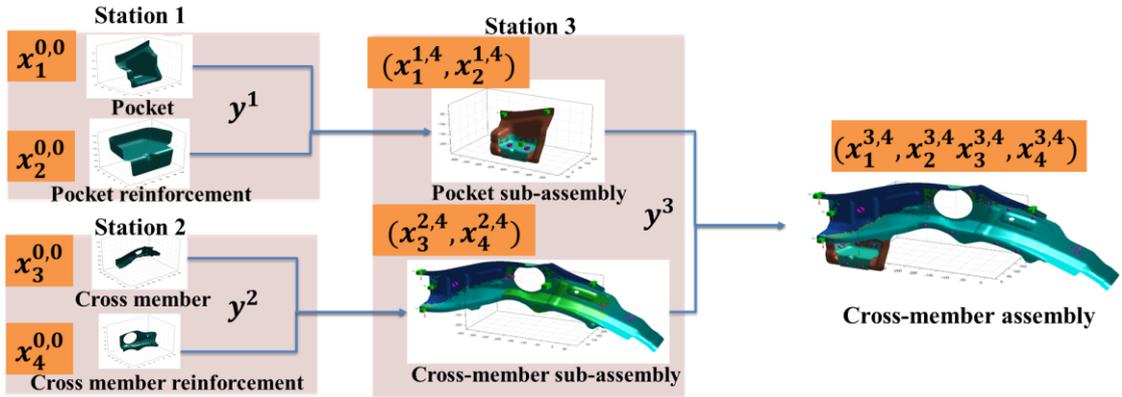


Fig. 4.6. Cross-member assembly system configuration

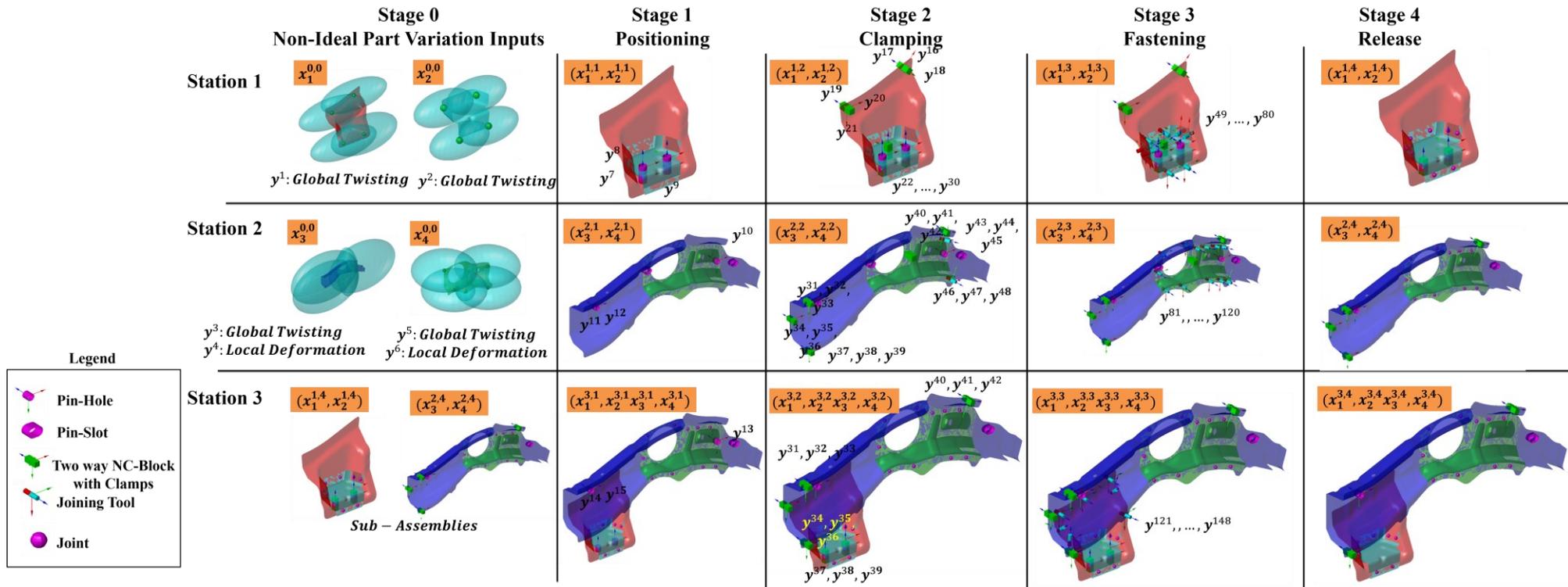


Fig. 4.7. Cross-member assembly process parameters

In summary, the industrial assembly process selected for case study consists of: (i) high dimensionality point cloud (11875 points); (ii) non-linearity induced by fixturing (N-2-1, N=6,9,7 for station 1, 2 and 3 respectively), transfer between stations, four non-ideal compliant parts and part-to-part interactions; (iii) collinearity induced by fixturing and re-orientation between the three stations; (iv) high fault multiplicity as 6-sigma defects are considered at the level of variation within 3D scanner accuracy (<0.05 mm) that significantly increases fault multiplicity from zero to 148 process parameters manifesting errors (100% fault multiplicity); (v) requirement for uncertainty quantification as the isolated root causes will drive costly corrective actions in high volume automotive manufacturing systems; (vi) requirement for dual data generation as a comprehensive real dataset of all possible root cause scenarios is not possible; (vii) high dimensional and heterogeneous set of process parameters as the assembly system consists of 123 real-valued and 25 binary process parameters; and, (viii) fault localization as the multi-station system requires estimation of shape error at the end of stations one and two to perform RCA.

Table 4.3: Process Parameters Description

Process Parameters		Interpretation of Root Causes in MAS
Part variation parameters y^1, \dots, y^6	6 real-valued (y^r) (substance)	These are root causes (RCs) caused by parameters variations of upstream fabrication process(es) such as stamping, etc. The process parameters have been consolidated to represent global variation patterns such as bending and twisting; and local deformations such as dents and wrinkles in the incoming parts $x^{0,0}$.
Placement (P) parameters y^7, \dots, y^{15}	9 real-valued (y^r) (infrastructure)	These are RCs represented by part placement errors (orientation/reorientation, stability). These RCs are caused by tooling installation and calibration error or tooling deterioration due to gradual worn out of fixture locators (pin-hole, pin-slot, NC block). Each station has three real-valued positioning process parameters.
Clamping (C), fastening (F) parameters y^{16}, \dots, y^{123}	108 real-valued (y^r) (infrastructure)	These are RCs caused by misalignment of clamps or fastening tools. Each has three real-valued parameters quantifying misalignment in x,y and z-direction.
Fastening/Joining (F) parameters y^{124}, \dots, y^{148}	25 binary (y^c) (infrastructure)	These are RCs caused by failure of achieving quality joint due to excessive gap (> 1 mm).

Model training is started with 5000 initial samples, 1000 samples are adaptively added during each iteration of the closed-loop training based on the uncertainty estimates, and the model is trained on the combined dataset including all previous samplings to ensure that there is no catastrophic forgetting. The diagonal scale parameters for all process parameters in the covariance matrix are fixed at 0.005, assuming a fixed level of noise hence, constant aleatoric uncertainty. A validation set of 300 samples is generated. After each iteration, the trained model is evaluated on the validation set. During evaluation for each of 300 samples, 100 MC samples are drawn from the trained model. The sample means are considered the estimate for the process parameters, while the sample Inter-Quartile Range (IQR) quantifies the epistemic uncertainty. The closed-loop training is stopped when the average MAE across all real-valued process parameters for the validation set is below the threshold, which is selected to be 0.3 mm and the accuracy of all binary process parameters is above 90%. Overall, ten trials are done for comparison of performance across all the requirements. The results and convergence of the closed-loop training are shown in Fig. 4.8.

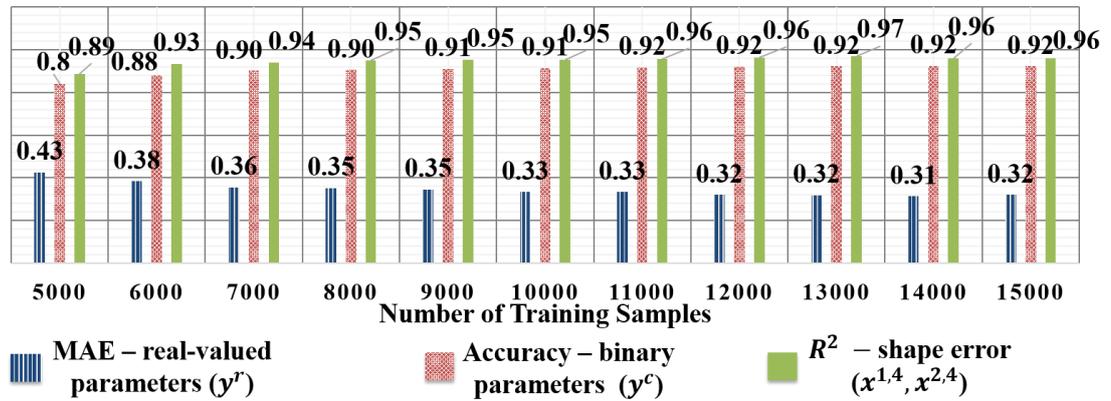


Fig. 4.8. Model results and convergence

4.4.2 Interactions

The above-mentioned requirements and the proposed solutions have various levels of interaction and hence, can be aggregated to understand the tradeoff between the proposed solutions further and conduct effective benchmarking for one or multiple requirements collectively. Requirement (i) drives to build an input data structure that can effectively retain all information regarding the object geometry and components of shape error. 3D Shape error voxelization retains all such information and also enables

convolution operations by building a regular 3D structure. Using simpler data structures, dimensionality reduction or downsampling points will negatively affect performance for requirements (ii), (iii), (iv) and (vii) as the global geometry information and local shape error granularity will be reduced. Requirements (ii), (iii), (iv) and (vii) can be collectively considered as an indicator of the complexity of the MAS. Complex MASs with more stations and process parameters will have higher non-linearity, collinearity and possible fault multiplicity within the system. Hence, benchmarking for all these requirements are done collectively as *process parameter estimation* capabilities against various machine/deep learning approaches used for regression and classification. Requirement (v) enables quantifying the uncertainties and is fulfilled using probabilistic weights in the network. The quality of the uncertainty estimates (calibration) is directly affected by MAS complexity, an increase in the number of process parameters increases uncertainty within various estimation ranges. Benchmarking for requirement (v) is done as *Uncertainty Quantification* for different prior distributions assuming the total number of process parameters are fixed. Benchmarking for requirement (vi) is done by comparing the number of samples required for convergence for each methodology. Finally, benchmarking for requirement (vii) that aims to estimate upstream shape errors is done as *Object Shape Error Estimation* against deep learning methods used for segmentation.

4.4.3 Benchmarking: Process Parameter Estimation

Process parameter estimation capabilities are benchmarked considering requirements (i) to (vii) against three categories of methods specifically, (1) CNN based Deep Learning approaches; (2) Traditional Machine Learning approaches in a multi-output regression or classification setting; and (3) currently used linear approaches. The results of the benchmarking are summarized in Table 4.4. Comparison for requirement (i) is done based on the data preprocessing technique used to handle the point cloud data dimensionality. Shape error voxelization retains information of the 3D structure and shape error features, while projection retains only 2D spatial structure. Flattening eliminates all information related to the spatial structure, while PCA also eliminates information explaining 1% of the variance. Comparison for requirements (ii), (iii), (iv) and (vii) are done on regression performance attributes, namely, accuracy (MAE) and

Table 4.4: Benchmarking: Process Parameter Estimation

		Requirement (i)	Requirements (ii),(iii),(iv) and (vii)								Req. (v)	Requirement (vi)			
	Models		MAE y^r		R^2 y^r		Accuracy y^c		AUC-ROC y^c		Uncertainty Estimates	Samples	CAE Simulation Time (minutes)	Training Time (minutes)	Total Time (minutes)
			Mean	SD	Mean	SD	Mean	SD	Mean	SD					
Proposed	OSER-MAS (Bayesian 3D U-Net)	3D Shape Error Voxelization	0.32	0.04	0.94	0.03	0.92	0.02	0.88	0.01	Standard Normal prior (corr. = 0.99)	Closed-loop (15000)	54000	1865	55865
	OSER-MAS (3D U-Net)	3D Shape Error Voxelization	0.32	0.01	0.94	0.02	0.91	0.01	0.87	0.01	No	30000	108000	1782	109782
Bayesian Deep Learning	OSER (Bayesian 3D CNN)	3D Shape Error Voxelization	0.35	0.05	0.91	0.03	0.88	0.01	0.84	0.02	Standard Normal prior (corr. = 0.91)	Closed-loop (16000)	57600	1802	59402
Deep Learning	PV CNN [114]	3D Shape Error Voxelization	0.36	0.02	0.90	0.03	0.86	0.01	0.81	0.03	No	30000	108000	1389	109389
	Pointnet++[198]	Uniformly downsampled set of points	0.37	0.01	0.91	0.04	0.84	0.02	0.80	0.02	No	30000	108000	1389	109389
	Multi-View 2D [190] CNNs (MV-CNNs)	6 face projection and 2D gridding	0.41	0.03	0.88	0.02	0.81	0.01	0.78	0.01	No	30000	108000	1389	109389
	Depth Based 2D CNNs [130]	1 face projection and 2D gridding	0.45	0.03	0.85	0.03	0.80	0.02	0.77	0.02	No	30000	108000	1321	109321
	Sparse Deep Neural Networks [199]	Flattening	0.62	0.02	0.78	0.01	0.78	0.03	0.65	0.03	No	30000	108000	1445	109445
Machine Learning	Boosted Trees	PCA	0.57	0.02	0.79	0.01	0.75	0.02	0.67	0.02	No	20000	72000	488	72488
	Random Forests	PCA	0.63	0.03	0.65	0.02	0.71	0.02	0.61	0.01	No	20000	72000	468	72468
	SVM/SVR	PCA	0.78	0.04	0.59	0.02	0.68	0.03	0.59	0.04	No	20000	72000	425	72425
Linear Approaches	Reg. Linear/Logistic Regression	PCA	0.81	0.01	0.58	0.01	0.65	0.02	0.62	0.01	No	20000	72000	58	72058

goodness-of-fit (R^2) for y^r ; and, classification performance metrics, namely, accuracy and AUC-ROC. Comparison for requirement (v) is done on the ability to quantify and segregate uncertainties. Lastly, a comparison for requirement (vi) is done on overall training time inclusive of the CAE simulation time and model training time. The OSER-MAS is able to perform better compared to all models. The OSER model also provides good results for these requirements as they overlap for single- and multi-station assembly systems. A two-sample t-test at 95% significance between the OSER-MAS and OSER model for metrics, namely R^2 ($p = 0.03$) and AUC-ROC ($p = 0.01$) demonstrates that the performance of the proposed OSER-MAS model is statistically significant

4.4.4 Benchmarking: Uncertainty Quantification

Benchmarking for quality of uncertainty estimates, i.e. requirement (v), is done using an error vs uncertainty calibration curve. RMSE is used as the error metric, and Inter-Quartile-Range (IQR) is used as the uncertainty metric given the non-normal nature of the estimated distributions. The RMSE for all observations between two consecutive thresholds is averaged. The correlation between the IQR and RMSE is also compared. Four priors for the weights and biases are considered: Standard Normal, Two Mixture GMM, Slab-Spike and Bernoulli. The Bernoulli prior method for uncertainty quantification is also known Monte Carlo (MC) dropout [44] and leverages dropout for approximate Bayesian inference.

As shown in Fig. 4.9, Standard Normal gives the most calibrated estimates for uncertainty with a correlation of 0.99. Both the OSER-MAS and OSER use standard normal priors, but given the Voxnet based 3D CNN architecture of the OSER approach, the correlation between IQR and RMSE is limited to 0.91, hence demonstrating the capability of the OSER-MAS model to provide more calibrated uncertainty estimates.

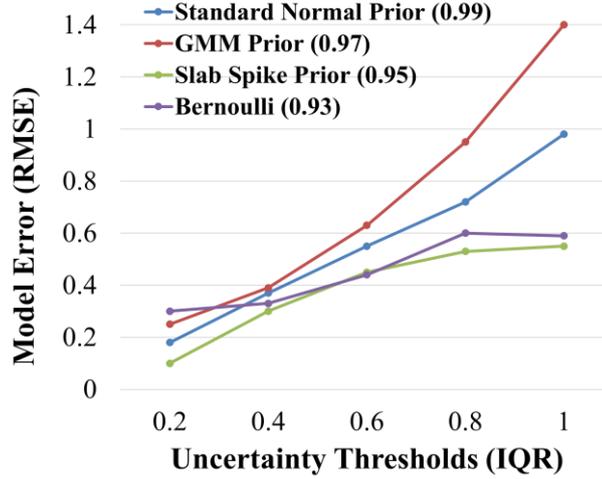


Fig. 4.9. Error (RMSE) vs uncertainty (IQR) comparison

4.4.5 Benchmarking: Object Shape Error Estimation

Benchmarking for requirement (viii), i.e., object shape error estimation for previous assembly stations, is done against other variants of 3D U-Nets and state-of-the-art methods used for semantic segmentation such as fully convolutional networks (FCN) [200], U-Net++, SegNet and DeepLab. Methods other than U-Net based approaches use a 2D projection and a VGG [201] based feature extractor. Given that shape errors are real-valued, regression-based metrics RMSE, MAE and R^2 are used to compare performance and have been summarized in Table 4.5. The proposed model performs better in shape error estimation than others given the attention link between corresponding levels of the encoder and decoder, which allows the model to be particular to parts/subassemblies within upstream shape errors.

Table 4.5: Benchmarking: Object Shape Error Estimation

Models	Requirement (viii) $x^{1,4}, x^{2,4}$						Parameters
	RMSE (mm)		MAE (mm)		R^2		
	Mean	SD	Mean	SD	Mean	SD	
OSER-MAS (3D Attention U-Net)	0.014	0.005	0.002	0.001	0.96	0.1	2.4 M
U-Net ++ [202]	0.21	0.02	0.09	0.03	0.93	0.1	1.8 M
3D U-Net without attention [194]	0.25	0.01	0.15	0.02	0.91	0.2	1.5 M
SegNet [203]	0.31	0.01	0.21	0.02	0.88	0.1	1.5M
DeepLab [204]	0.38	0.02	0.25	0.03	0.86	0.1	20.5M
Fully Convolutional Networks (FCN) [200]	0.44	0.02	0.33	0.03	0.84	0.3	1.6M

Additionally, a comparison between OSER and OSER-MAS kernels in terms of methodology capabilities and performance is summarized in Table 4.6.

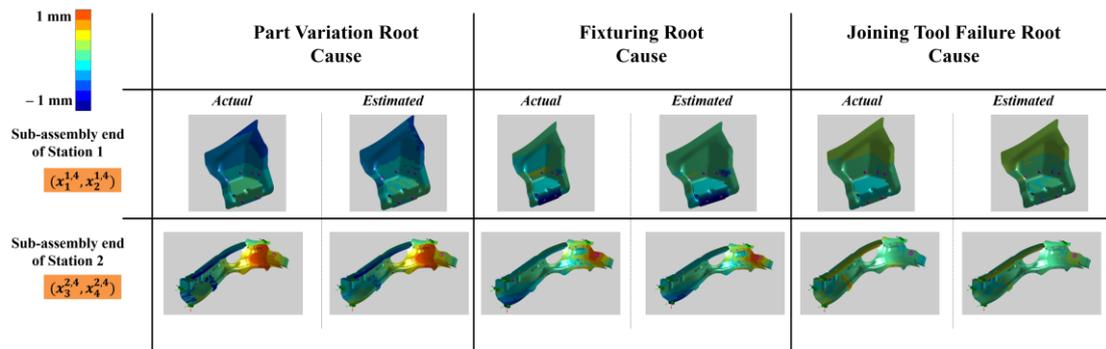
Table 4.6: Comparison of OSER (Chapter 3) & OSER-MAS

Requirements	Methodology		Performance	
	OSER	OSER-MAS	OSER	OSER-MAS
(i) <i>high data dimensionality</i>	3D Shape Error Voxelization	3D Shape Error Voxelization	(64*64*64*3) – voxels with shape error components	(64*64*64*3) – voxels with shape error components
(ii) <i>non-linearity</i>	Voxnet based CNN architecture	3D U-Net based CNN architecture	$MAE = 0.32\text{ mm}$ $R^2 = 0.94$ 83 % fault multiplicity	$MAE = 0.34\text{ mm}$ $R^2 = 0.91$ 100 % fault multiplicity
(iii) <i>collinearities</i>				
(iv) <i>high faults multiplicity</i>				
(v) <i>uncertainty quantification</i>	Bayes-by-backprop and Flipout	Bayes-by-backprop and Flipout	Calibration correlation = 0.91	Calibration correlation = 0.99
(vi) <i>dual data generation capabilities</i>	Uncertainty based sampling from CAE Simulation	Uncertainty based sampling from CAE Simulation	15000 samples for convergence	16000 samples for convergence
(vii) <i>High Dimensionality and heterogeneity of process parameters</i>	Single output head to estimate real-valued parameters	Multiple output heads to estimate real-valued and binary process parameters	N/A	$Accuracy = 0.92$ $AUC - ROC = 0.88$
(viii) <i>Fault Localization for RCA (Required for RCA of MAS)</i>	No	Attention-based decoder	N/A	$RMSE = 0.014\text{ mm}$ $MAE = 0.002\text{ mm}$ $R^2 = 0.96$

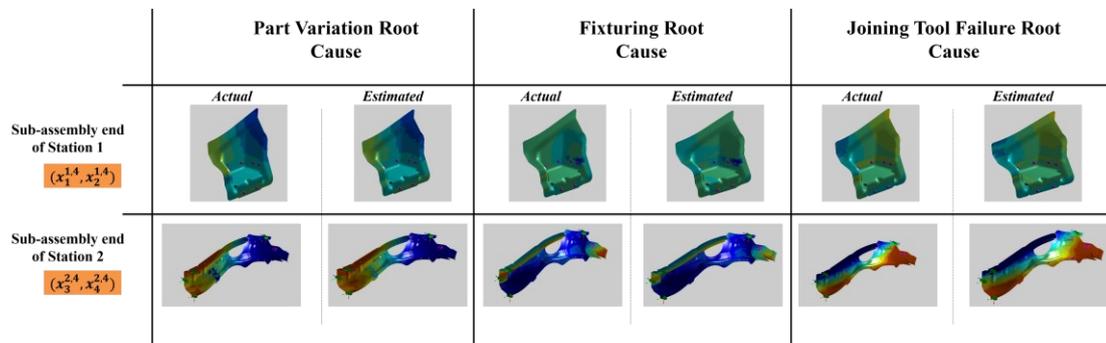
4.4.6 Object Shape Error Maps

Fig. 4.10 compares the actual and estimated upstream shape error of the proposed approach for various types of RCs (part variation RC: $y^1 = 0.5\text{ mm}$, fixturing RC: $y^{11} = 0 - 0.5\text{ mm}$, joining RC: $y^{124} = 0$) across all three components (x,y and z) of shape error. The proposed approach is able to accurately estimate all components of upstream shape errors in spite of all RCs being located in station 1. These shape errors enable the localisation of faulty assembly stations; the process parameters within these stations can be further analysed to isolate the particular process parameter (RC). The

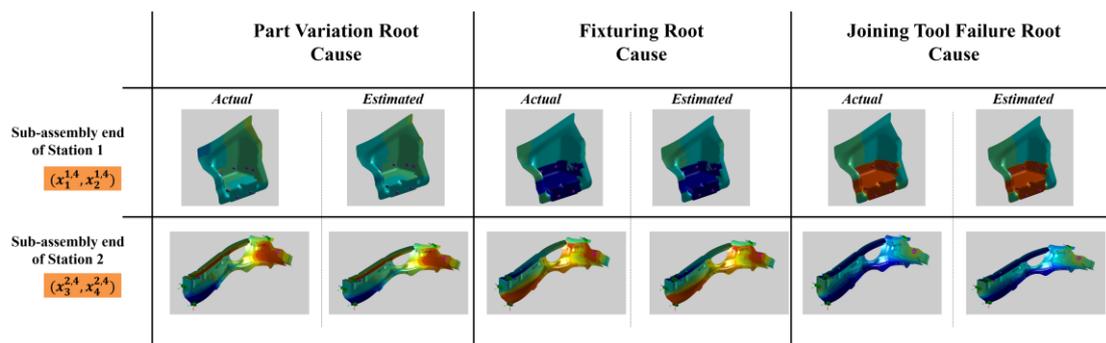
actual and estimated object shape errors at the end of upstream stations as obtained from the decoder of the OSER-MAS model are shown in Fig. 4.10.



Comparison of x-component of object shape error



Comparison of y-component of object shape error



Comparison of z-component of object shape error

Fig. 4.10. Actual and estimation comparison for upstream object shape errors

4.5 Conclusions

The chapter presented a 3D U-Net based methodology, Object Shape Error Response for Multi-station Assembly System (OSER-MAS), to isolate RC of dimensional and

geometric product shape errors. It leverages a Bayesian U-Net 3D CNN architecture and closed-loop training to enable RCA with uncertainty quantification and learning during the design phase of the assembly system in the absence of measurement data. Additionally, the approach enables fault isolation by estimation of a heterogeneous set of process parameters, which include (i) incoming non-ideal and deformable parts; (ii) place-clamp-fasten-release (PCFR) parameters in assembly cell; and, (iii) part-to-part contact chains. The approach also enables fault localization by estimation of shape errors for upstream assembly stations. The approach is benchmarked against several categories of deep learning and statistical models. Results demonstrated better performance, quicker convergence and unique uncertainty quantification.

5 Building a Scalable and Interpretable Bayesian Deep Learning Framework for Quality Control of Free Form Surfaces³

5.1 Introduction

As shown in Chapter 3 and Chapter 4, OSER and OSER-MAS models demonstrate high accuracy for 3D object shape error response to estimate dimensional and geometric quality defects in MAS. Deep learning-driven RCA is increasingly used for decision-making when planning corrective action of quality defects. However, given the current absence of scalability enabling models, training deep learning models for each MAS is exceedingly time-consuming. It requires large amounts of labelled data and multiple computational cycles. Understanding and interpreting how deep learning produces final predictions while quantifying various uncertainties also remains a fundamental challenge. In an effort to address these gaps, a novel CLIP diagnostic algorithm portfolio is proposed in this chapter, which simultaneously enhances scalability and interpretability of the OSER-MAS approaches to isolate root cause(s) of quality defects in MAS. The CLIP diagnostic approach shortens OSER-MAS model training time by developing: (i) closed-loop training to enable faster convergence for a single MAS by leveraging uncertainty estimates of the Bayesian 3D U-net model; and (ii) transfer/continual learning-based scalability model to transmit meta-knowledge from the trained model to a new MAS resulting in convergence using comparatively less training samples. Additionally, CLIP increases the transparency for quality-related root cause predictions by developing an interpretability model which is based on 3D Gradient-based Class Activation Maps (3D Grad-CAMs) and entails: (a) linking

³ Based on S. Sinha, P. Franciosa, and D. Ceglarek, "Building a Scalable and Interpretable Bayesian Deep Learning Framework for Quality Control of Free Form Surfaces," IEEE Access, vol. 9, 2021, DOI: 10.1109/ACCESS.2021.3068867.

<https://ieeexplore.ieee.org/document/9386067>

elements of MAS model with functional elements of the U-Net architecture; and, (b) relating features extracted by the architecture with elements of the MAS model and further with the object shape error patterns for root cause(s) that occur in MAS. Benchmarking studies are conducted using six automotive-MAS with varying complexities. Results highlight a reduction in up to 56% training samples with a loss in performance of up to 2.1%. The kernel fulfils requirements (ix)-(x) within the scope of single- and multi-station assembly systems. The contributions of the chapter can be summarized as follows:

- (1) A closed-loop training approach to enable faster convergence for a single MAS by leveraging uncertainty estimates of the Bayesian 3D U-net OSER-MAS model.
- (2) Uncertainty guided transfer/continual learning-based scalability model to transfer meta-knowledge from the trained model to a new MAS, and thus, each new MAS requires comparatively fewer training samples.
- (3) A 3D Grad-CAMs based interpretability model which links functional elements and features extracted by the 3D U-Net architecture with elements of the MAS model and further with the shape error patterns for root causes(s) that occur in MAS.
- (4) Verified and validated scalability and interpretability of the CLIP diagnostic framework underpinned algorithm portfolio using six different industrial automotive assemblies of varying complexities.

The rest of the chapter is organized as follows; Section 5.2 summarizes the related work on scalability and interpretability; Section 5.3 summarizes the object shape error estimation and RCA problem for MASs as done previously in Section 4.2, Section 5.4 discusses the methods for scalability and interpretability, Section 5.5 presents the industrial case studies and conclusions are summarized in Section 5.6.

5.2 Related work on Scalability and Interpretability

Scalability within manufacturing systems has been stated as a set of capabilities to provide transfer of knowledge and ideas from other engineering and management areas [205]. Scalability for algorithms to perform RCA of MASs translates into effectively leveraging the learning for one type of assembly system and then transferring this

learning in the form of features and relationships which can be relevant for another similar assembly system, and hence, can enable learning for the latter using an exponentially lesser amount of data and computation capabilities. Applications of using transfer learning techniques have been made for fault diagnosis [206]. Digital Twins [207] have also been proposed as a way to enable scalability. Successful applications of transfer learning across multiple domains [29], [135], [208]–[210] have enabled scalability in a sustainable manner that does not require exhaustive training data and computation capabilities. Recent works have also proposed that scalability should be life-long or continual and should not come at the expense of forgetting previous learning when new features or relationships are learnt for new systems [92], [211]–[214]. Hence, to enable scalability for MAS, methodologies that integrate transfer/continual learning with existing deep learning approaches for RCA are required. Furthermore, it is essential to ensure that training data and computation times do not become barriers to applying such models within industrial setups.

Interpretability has been another primary concern for applying deep learning-based RCA models within MAS as they do not provide the required context, trust and confidence within root cause estimates. When coupled with costly corrective actions driven by them, this lack of transparency results in such models not being adopted at scale. Various methodologies such as Gradient-based class activation maps that can integrate deep learning estimates with the required transparency have been proposed [215]–[217]. Bayesian deep learning [15] has been proposed to integrate confidence and uncertainty measures with root cause estimates. However, there is a need for frameworks within MAS to provide interpretability while accounting for context and confidence. Such frameworks enable trust in black-box deep learning models by providing interpretability on multiple levels.

5.3 Problem Formulation

The CLIP approach leverages the same formulation of MAS as done in Section 4.2. The object shape error for object o after station s and stage \acute{s} can be represented as:

$$\mathbf{x}_o^{s,\acute{s}} = (\mathbf{P}_o^{s,\acute{s}}, \mathbf{d}_o^{s,\acute{s}}) \quad (5.1)$$

The aim for CLIP is training the proposed 3D CNN (Chapter 3) or 3D U-Net (Chapter 4) model in a *scalable* and *interpretable* manner for each MAS to learn a function $f(\cdot)$ (as shown in (4.4)) that takes as input the combined object shape error at the end of the system $\mathbf{x}^{N_s,4}$, i.e., after the final stage of the last station (N_s), and estimates the process parameters across the entire system and the object shape error for all objects at the end of the previous stations:

$$[\mathbf{y}^r, \mathbf{y}^c, \mathbf{x}^{1,4}, \dots, \mathbf{x}^{N_s-1,4}] = f(\mathbf{x}^{N_s,4}) \quad (5.2)$$

As discussed in Section 4.3.6, the framework for RCA remains the same. The estimates of (5.2) are leveraged to perform RCA using three steps, namely: (i) *Fault Identification*, (ii) *Fault Localization* and (iii) *Fault Isolation*.

5.4 Methods

5.4.1 Bayesian 3D U-Net Architecture

As discussed in Section 4.3.3 for the estimation of $f(\cdot)$ as shown in (5.2), a Bayesian 3D U-Net model is leveraged (Fig. 4.2). This enables (i) estimation of a heterogeneous set of process parameters; (ii) estimation of upstream object shape errors; and (iii) quantification and segregation of uncertainties. The model is trained using a weighted loss function [218] that accounts for all the aforementioned outputs and uncertainty quantification. The architecture consists of four levels of the encoder and decoder. The end of the encoder consists of a regression and classification head. Each head contains one hidden Dense Flipout layer with 64 nodes and ReLU activation. The output nodes in each head are equal to the real-valued and binary process parameters. These heads enable the estimation of a heterogeneous set of process parameters. The end of the decoder estimates the upstream object shape errors. Given the use of Bayesian Flipout layers in the encoder and regression and classification heads, the architecture enables quantification and uncertainty segregation. The encoder-decoder model consists of seven key functional elements namely: (1a) Object shape error voxelization; (1b) Encoder with down-sampling kernels; (1c) Decoder with up-sampling kernels; (1d) Multiple output heads; (1e) Attention gate; (1f) Bayesian Flipout layers; and, (1g)

Residual connections. The interpretability of each element is discussed concerning the requirements of MAS. Overall the model takes as input the voxelized shape error after the final station $\mathbf{x}^{N_s,4} \rightarrow \{V_{u,v,w,d}\}$ and give as output the shape error after previous stations and the process parameters $[\mathbf{y}^r, \mathbf{y}^c, \mathbf{x}^{1,4}, \dots, \mathbf{x}^{N_s-1,4}]$. Uncertainties are estimated for each output value. The uncertainty estimates are crucial in driving costly corrective actions. They are segregated into epistemic and aleatoric uncertainties. The aleatoric uncertainty is estimated by considering that the outputs follow a multivariate normal distribution with a diagonal covariance matrix. The epistemic uncertainty is estimated by assuming each weight ω in the network follows a normal distribution with parameters $\theta_\omega = (\mu_\omega, \sigma_\omega)$, and then estimating the posterior parameters of the distributions. The chapter proposes to leverage these measures of uncertainty to build approaches that enable scalable learning. The overall epistemic uncertainty of the estimated process parameters $\sigma(\mathbf{y})$ and the uncertainty of the weights σ_ω can be further leveraged to build methods that enable scalable learning by leveraging closed-loop sampling from CAE simulators and further leverage uncertainty-guided continual learning [92] for effective learning that aids in transferring knowledge in between similar MASs, hence, enabling convergence using fewer training samples while also ensuring that there is no catastrophic forgetting of learning for previous MAS.

5.4.2 Closed-loop Sampling and Training

Closed-loop sampling enables the dynamic and adaptive generation of training samples based on the uncertainty and error of the previous training iterations while ensuring that the sample generation has a degree of randomness to prevent the repeated generation of similar samples (Table 5.1). This enables faster convergence to the optimal weights and biases distribution parameters of $f(\cdot)$ as shown in (5.2). Sampling is done using VRM [6] as the CAE simulator, which takes as input a set of process parameters (\mathbf{Y}) and outputs the object shape errors after the desired stage/station (\mathbf{X}). Initially, Latin Hypercube Sampling (LHS) [182] of the process parameters within the allowable ranges is done for input to the CAE simulation model to generate the test (\mathbf{Z}) and validation (\mathbf{V}) set.

$$\mathbf{Z} = (\mathbf{X}^Z, \mathbf{Y}^Z), \mathbf{V} = (\mathbf{X}^V, \mathbf{Y}^V) \quad (5.3)$$

While each element in \mathbf{X} is characterized by the object shape error $\mathbf{x}^{s,\hat{s}}$ (5.1) and each row in \mathbf{Y} consists of a vector of process parameters: $\mathbf{y} = \{\mathbf{y}^f, \mathbf{y}^c\}$. The initial training set \mathbf{T}^0 is also generated using LHS. This is used to train the proposed architecture $f(\cdot)$. After training, the inference is performed on the validation set to obtain the predictions $\hat{\mathbf{Y}}^V$ and uncertainty σ^V on the validation set (\mathbf{V})

$$[\hat{\mathbf{Y}}^V, \sigma^V] = f(\mathbf{X}^V) \quad (5.4)$$

For all samples, the absolute error is calculated for all h the process parameters:

$$E_h^V = |\hat{Y}_h^V - Y_h^V| \quad (5.5)$$

The error is summed up across all h process parameters for all samples to generate \mathbf{e}^V which is a column vector consisting of combined error for each sample within the validation set.

$$\mathbf{e}^V = \sum_h E_h^V \quad (5.6)$$

The error \mathbf{e}^V and uncertainty σ^V are normalized between [0,1] across all samples in the validation set. The normalized error ($\tilde{\mathbf{e}}^V$) and uncertainties ($\tilde{\sigma}^V$) are weighted to obtain the sampling importance metric $\boldsymbol{\tau}^V$ for each sample:

$$\boldsymbol{\tau}^V = w \cdot \tilde{\mathbf{e}}^V + (1 - w) \cdot \tilde{\sigma}^V \quad (5.7)$$

Samples are sorted based on the importance metric. The sorted set of importance metric $\check{\boldsymbol{\tau}}^V$ and process parameter $\check{\mathbf{Y}}^V$ (true values) is represented as $(\check{\boldsymbol{\tau}}^V, \check{\mathbf{Y}}^V)$. This sorting is done considering that samples having the highest importance, i.e., having the maximum sum of error and uncertainty would significantly contribute to model convergence than other samples. Based on the sorted importance metric $\check{\boldsymbol{\tau}}^V$ and corresponding process parameters $\check{\mathbf{Y}}^V$, the parameters of a sampling distribution are estimated – Gaussian Mixture Model (GMM) with a pre-specified number of mixtures is considered as the sampling distribution. Given the GMM has K mixtures, the sorted samples $(\check{\boldsymbol{\tau}}^V, \check{\mathbf{Y}}^V)$ are

subdivided into K blocks each block i^b having $dim(V)/K$ samples where $dim(V)$ represents the total number of samples within the validation set. Each block i^b with samples having importance metric $\check{\tau}_{i^b}^V$ and process parameters $\check{Y}_{i^b}^V$ are used to estimate distribution parameters for the i^b th mixture. To estimate the mixture component weights the importance metric for all samples within each block $\check{\tau}_{i^b}^V$ are summed up and further, these summed up values are normalized between $[0,1]$ across all blocks to represent mixture component weights ϕ_{i^b} . A multivariate normal distribution characterizes each mixture while the multi-variate normal component corresponds to the process parameters. The mean μ_{i^b} and covariance Σ_{i^b} of each mixture (block) is calculated by taking the mean and covariance of the samples within the block $\check{Y}_{i^b}^V$. Overall, the distribution parameters θ_{i^b} for the $i^{b^{th}}$ can be denoted as:

$$\theta_{i^b} \rightarrow \phi_{i^b}, \mu_{i^b} = mean(\check{Y}_{i^b}^V), \Sigma_{i^b} = covar(\check{Y}_{i^b}^V) \quad (5.8)$$

Where ϕ_{i^b} represent mixture component weight and μ_{i^b}, Σ_{i^b} represent the mean vector and covariance matrix, respectively, for the $i^{b^{th}}$ mixture. After estimating the distribution, T^i samples (outputs Y) are drawn from the GMM model and evaluated using the CAE simulator (VRM) to obtain inputs (X). These are then further leveraged to train the model. This ensures that samples are drawn considering the error and uncertainty while ensuring that the model does not overfit on the validation set. It should be noted that although the validation set is used for sampling, the model is never trained on the validation samples. After each training iteration is done, the model is tested on the test set (Z) to determine the model performance. The training and sampling are terminated either after (i) the performance on the test set (Z) reaches the required threshold ε or (ii) the maximum number of training iterations n^m is reached. The performance threshold can be decided based on the case study and application, and the maximum number of iterations is decided based on the CAE simulation budget.

Table 5.1: Closed-Loop Sampling and Training Framework

<i>Algorithm 5.1: Closed-loop sampling and training</i>	
Procedure:	
Generate and evaluate Testing (\mathbf{Z}) and Validation (\mathbf{V}) set	
Generate Initial Training Set (\mathbf{T}^0)	
Repeat $i = 0$ to n^m:	
Train model on training set (\mathbf{T}^i)	
Perform inference on validation (\mathbf{V}) set and obtain $\hat{\mathbf{Y}}^V, \sigma^V, \mathbf{E}_h^V$	
Estimate GMM parameters (ϕ, μ, Σ) using (5.6), (5.7) and (5.8)	
Sample from the GMM and evaluate using CAE simulator to obtain \mathbf{T}^s	
Add samples to training set $\mathbf{T}^{i+1} \leftarrow \mathbf{T}^i \cup \mathbf{T}^s$	
Stopping Checks:	
Error on test (\mathbf{Z}) set, below the threshold $e^e < \varepsilon$	
Maximum number of iterations reached $i = n^m$	

5.4.3 Uncertainty Guided Continual Learning

The closed-loop sampling approach enables faster convergence within one MAS but does not enable the trained $f(\cdot)$ to be used across different MASs. Leveraging the trained function $f(\cdot)$ across different MAS enables transfer of relevant knowledge (features) and enables convergence in comparatively lesser samples enabling scalability. Continual learning methods (also known as sequential/lifelong learning) aim to incrementally learn new tasks without forgetting previous tasks for which they have been trained. In the context of MASs, the scale-up starts from tasks or cases as simple as a coupon or Top-hat assembly to full-scale MAS such as automotive car door or cross member assemblies and can cumulatively consist of up to 100 assembly stations, each consisting of multiple stages. Each assembly case is treated as a task T_i , and continual learning is performed for a total of T_n tasks. Continual learning enables the transfer of the process parameter estimation capabilities of previous assembly cases to more complex assemblies while retaining the essential capabilities required for process parameter estimation of previous assemblies. The key to achieving continual learning requires assignment of importance to each neural network weight ω and further updating only non-important weights such that the model learns the new task without forgetting the previous task [92]. The approach leverages uncertainty guided continual learning because the weight uncertainty σ_ω of the Bayesian 3D U-Net model serves as an implicit measure of importance. Additionally, the ease of interpretation, strong mathematical foundation and good results on various datasets [92] motivate using such

a learning algorithm.

Given the use of Bayesian neural networks and a normal distribution parametrized by $\theta_\omega = (\mu_\omega, \sigma_\omega)$ for each weight ω within the network, the standard deviation of each weight distribution is leveraged as the metric for importance. To enable continual learning, the learning rate α for each parameter is updated by the corresponding importance Ω .

$$\alpha_\mu \leftarrow \alpha_\mu / \Omega_\mu, \alpha_\sigma \leftarrow \alpha_\sigma / \Omega_\sigma \quad (5.9)$$

The importance of the parameters is set to be inversely proportional to the standard deviation, which mathematically means that weights with higher standard deviation are less important and hence can be updated at a higher rate to learn new tasks. In comparison, weights with lower standard deviation are more important and hence should be updated at a lower rate to prevent catastrophic forgetting (performance loss) for the old tasks.

$$\Omega \propto 1/\sigma_\omega \quad (5.10)$$

Based on various empirical studies done by Ebrahimi et al. [92], the learning rate adaptations were determined as:

$$\Omega_\mu \leftarrow 1/\sigma_\omega, \Omega_\sigma \leftarrow 1 \quad (5.11)$$

The overall algorithm consisting of closed-loop sampling and continual learning to train the model on multiple tasks T_1, \dots, T_n (different assembly case studies) is shown in Table 5.2. After each task, the learning rates are updated as shown in (5.11). The number of output nodes within the model is kept equal to sum all process parameters across all cases studies. For each assembly case, the specific process parameter nodes output the values while other nodes corresponding to process parameters for other assembly cases are set to output a nominal fixed value (generally set to zero). Overall continual learning aims to learn process parameter estimation capabilities of each assembly case study incrementally while minimizing the forgetting for previous assembly case studies.

Table 5.2: Continual Learning Framework

Algorithm 5.2: Uncertainty based continual learning with closed-loop sampling

Procedure:
Setup all tasks (Assembly Case Studies)
Set initial learning rate $\alpha_\mu = \alpha_\sigma = \alpha_0$
Repeat for all tasks T_n:
Perform closed loop training for T_i^{th} task (<i>Algorithm 5.1</i>)
Update Learning Rates:
$\Omega_\mu \leftarrow 1/\sigma_\omega$
$\Omega_\sigma \leftarrow 1$
$\alpha_\mu \leftarrow \alpha_\mu/\Omega_\mu$
$\alpha_\sigma \leftarrow \alpha_\sigma/\Omega_\sigma$
end

5.4.4 Transfer Learning

Transfer learning is an effective method for transferring learning (process parameter estimation for MASSs) from one task to related tasks (between different assembly case studies) using fewer training samples. It hence acts as a critical enabler for scalability. This chapter leverages transfer learning and continual learning as a combined algorithm portfolio enabling scalability. The choice between them can be made based on training results and deployment performance.

Transfer learning is mathematically formalized [209][219] as a domain D which consists of features \mathbf{X} and a distribution over the feature space $P(\mathbf{X})$. In this case the domain entails process parameter estimation $f(\cdot)$ on a particular assembly case T_i with shape error features \mathbf{X} and distribution over the feature space as $P(\mathbf{X})$. Within Domain D task T is performed that constitutes learning a conditional distribution $P(\mathbf{Y}|\mathbf{X})$ to estimate process parameters \mathbf{Y}

$$D = \{\mathbf{X}, P(\mathbf{X})\} \quad (5.12)$$

The chapter aims to ‘transfer learn’ from the source domain (D_s) corresponding to a particular assembly case to a target domain (D_T), i.e. a similar assembly case to perform the same task of estimating $f(\cdot)$ (5.2) while accounting for differences between cases such that at least one of the elements between the domain and target are not the same:

$$(\mathbf{X}_S \neq \mathbf{X}_T), (P(\mathbf{X}_S) \neq P(\mathbf{X}_T)), (\mathbf{Y}_S \neq \mathbf{Y}_T), P(\mathbf{Y}_S|\mathbf{X}_S) \neq P(\mathbf{Y}_T|\mathbf{X}_T) \quad (5.13)$$

Considering the prior knowledge on the similarity of assembly cases studies, it can be estimated that: (i) $(\mathbf{X}_S \approx \mathbf{X}_T)$ – given that similar features [111] [220] need to be extracted from the object shape error data that include bends, twists, rotations, and translations.; (ii) $(P(\mathbf{X}_S) \approx P(\mathbf{X}_T))$ – similarly, the distribution around the input features is approximately the same; (iii) $\mathbf{Y}_S \neq \mathbf{Y}_T$ – the outputs for each study are different given a different number of process parameters are involved; and, (iv) $(P(\mathbf{Y}_S|\mathbf{X}_S) \neq P(\mathbf{Y}_T|\mathbf{X}_T))$ – the conditional distribution is also significantly changed given the change in the assembly system and the output. To account for (iii) the final layer of the network is replaced with nodes corresponding to the new set of process parameters; to account for (iv) the approach leverages standard protocols established in transfer learning to achieve transferability.

Based on past work on successful applications of transfer learning that involved using ImageNet data to aid Computer-Aided Detection [210], the fine-tuning transfer learning protocol is leveraged. The network weights are initialized using the weights of a network trained on the previous assembly case study. The whole network is then fine-tuned while keeping the learning rate of the convolutional layers α_c in the first two encoder levels ten times less than the rest of the network α_f . The regression and classification heads are replaced and reinitialized. The nodes in each of the heads are determined by the number of process parameters for the case study. Overall transfer learning (Table 5.3) aims to learn process parameter estimation capabilities of each assembly case study while ‘transferring’ knowledge from previous case studies. Although while doing this, the model can ‘forget’ estimation capabilities for previous case studies. Fig. 5.1 summarizes the overall framework for closed-loop sampling and training integrated with continual and transfer learning approaches.

Table 5.3: Transfer Learning Framework

<i>Algorithm 5.3: Transfer learning with closed-loop sampling</i>	
Procedure:	
Setup all tasks (Assembly Case Studies)	
Set initial learning rate α_0 for all layers	
Perform closed-loop training on 1 st task	
Set learning rate lower convolutional layers $\alpha_C \leftarrow \alpha_0/10$	
Set learning rate for all other layers $\alpha_F \leftarrow \alpha_0$	
Repeat for all tasks (2 to T_n):	
Replace output layer of neurons equal to the number of process parameters for the T_i^{th} task	
Perform closed-loop training for T_i^{th} task (<i>Algorithm 5.1</i>)	
Initialize network with trained weights	
end	

5.4.5 3D U-Net Architecture Interpretability

The implementation of deep learning models within industrial environments requires opening the black box and providing interpretability and causation on why the deep learning models can give superior performance compared to traditional linear or piecewise linear approaches traditionally used for RCA of MAS. The chapter proposes to do that on two levels:

(i) providing a link between MASs requirements and functional elements of the Bayesian 3D U-Net architecture of the OSER-MAS approach. This also aims to link the engineering challenges faced in RCA of MASs and the developments done within the OSER-MAS model to overcome these challenges.

(ii) leveraging 3D Grad-CAMs to interpret the features that are extracted by the architecture and then propagated through various encoder and decoder layers to be interpreted as root causes. To integrate high measures of confidence within the deep learning model estimates, it must be established that the input context $\mathbf{x}^{N_s,4}$ (object shape error) on which the model focuses should be directly related to the estimated output \mathbf{y} (process parameter or root cause), e.g. if the model estimates ‘part variation’ as a root cause, it should focus on the ‘part variation’ rather than other possible root causes such as ‘clamping’ or ‘positioning’. Clearly extracted semantics in convolutional layers integrates a much-needed measure of ‘trust’ within the root cause estimates.

5.4.6 3D Gradient-Weighted Class Activation Maps

3D Gradient-weighted class activation maps (3D Grad-CAMs) aim to visualize the input features that led to a particular output. In the context of MASs, this aims to localize key regions within the input shape error $\mathbf{x}^{Ns,4}$ that led to the estimation of a process parameter \mathbf{y} . This is estimated by taking a discriminative gradient of a particular process parameter y^m output with respect to the feature map of a selected convolutional layer within the 3D U-Net architecture. The map for a particular output process parameter y^m is represented as $\mathbf{L}_{grad-CAM}^{y^m}$ and can be calculated as a weighted sum of the features maps:

$$\mathbf{L}_{grad-CAM}^{y^m} = ReLU \sum_{f=1}^F \alpha_f^{y^m} \mathbf{A}^f \quad (5.14)$$

Where \mathbf{A}^f represents f feature maps ($f = 1, 2, \dots, F$) for the selected convolutional layer. ReLU represents the activation function which is rectified linear unit. The weights $\alpha_f^{y^m}$ are calculated by summing the gradients for each element within the feature map where a, b, c represent the dimensions of the 3D feature maps:

$$\alpha_f^{y^m} = \frac{1}{abc} \sum_{i=1}^a \sum_{j=1}^b \sum_{k=1}^c \frac{dy^m}{d\mathbf{A}_{i,j,k}^f} \quad (5.15)$$

The $\mathbf{L}_{grad-CAM}^{y^m}$ is interpolated to match the dimensions of the input voxelized object shape error. The overlay between the voxelized object shape error and interpolated 3D Grad-CAM provides interpretability information on what features/spatial regions within the shape error did the model focus on to estimate the selected process parameter y^m of interest. The interpolated 3D Grad-CAM is then transformed to a point-based shape error, and smoothing is done using a median filter for consistency across the mesh. These can be visualized to obtain regions within the shape error input that the neural network model focuses on to estimate the process parameter. Various limitations of Grad-CAM have been shown concerning the granularity of the generated

interpretability maps [221]. Hence further work is required in exploring other model-agnostic and model-specific approaches for interpretability and explainability [222].

Fig. 5.1 summarizes the integration of closed-loop training with continual/transfer learning-based scalability model and 3D Grad-CAMs based interpretability model. The effective integration of these models enhances the OSER-MAS model's diagnostic capabilities and enables scalable and interpretable RCA.

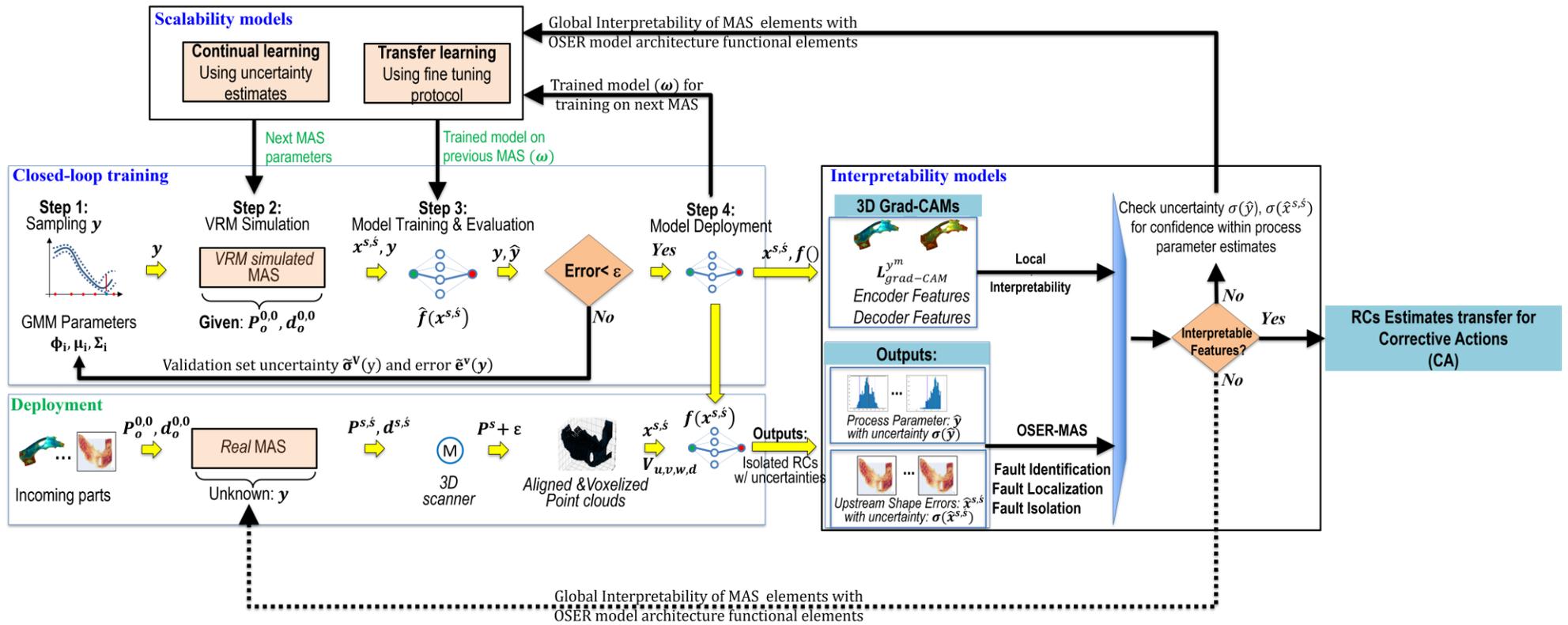


Fig. 5.1. CLIP underpinned algorithm portfolio

5.5 Case Study

5.5.1 Experimental Setup

Verification and validation is done using $T_n = 6$ tasks or assembly systems (Fig. 5.2, Table 5.4) with varying complexities ranging from a single part coupon level assembly to automotive, industrial multi-station assemblies. Each assembly case is considered a unique task. Continual or transfer learning is done sequentially for all case studies, as in the order mentioned below. The case studies include:

(1) *Flat Plate (Coupon) Assembly*: consists of $n = 1$ ideal compliant part with a flat 2D geometry. It involves $N_s = 1$ station and four stages (PCFR) and is controlled by $h = 7$ real-valued \mathbf{y}^f fixturing and joining based process parameters.

(2) *Top-Hat Assembly*: consists of a $n = 2$ ideal compliant parts with a simple 3D geometry. It involves $N_s = 1$ station and four stages (PCFR) and is controlled by $h = 17$ real-valued \mathbf{y}^f fixturing and joining based process parameters.

(3) *Door Halo Reinforcement Panel Assembly*: consists of $n = 1$ ideal compliant part with complex 3D geometry. It involves $N_s = 1$ station and two stages (PC) and is controlled by $h = 3$ real-valued \mathbf{y}^f fixturing based process parameters.

(4) *Door Inner and Hinge Reinforcement Assembly*: consists of $n = 2$ ideal compliant parts with a complex 3D geometry. It involves $N_s = 1$ station and four stages (PCFR) and is controlled by $h = 6$ real-valued \mathbf{y}^f fixturing based process parameters.

(5) and (6) *Cross Member Assembly*: consists of $n = 4$ non-ideal compliant parts with complex 3D geometry. It involves $N_s = 3$ stations, each with four stages (PCFR). Two sub-cases within this are considered: case (5) consisting of $h = 12$ real-valued \mathbf{y}^f part variation and fixturing based process parameters; and case (6) consisting of a heterogeneous (real-valued and binary) set of $h = 158$ process parameters, including 123 real-valued \mathbf{y}^f part variation, fixturing and joining based process parameters and

25 binary process parameters \mathbf{y}^c indicating the success of joining (Fig. 4.7).

For comparison and benchmarking of scalability, all five cases are analysed under $T_s = 4$ training scenarios are considered:

(i) *Random Sampling*: Involves randomly sampling from the CAE simulator within the allowable ranges for each process parameter. Each case study is trained on a re-initialized network with random weights. This also serves the baseline performance expectations.

(ii) *Closed-loop Sampling*: Involves training the five case studies using Algorithm 5.1 as shown in Table 5.1.

(iii) *Transfer Learning with Closed-loop sampling*: Involves training the five case studies sequentially using Algorithm 5.3 as shown in Table 5.3.

(iv) *Continual Learning with Closed-loop Sampling*: Involves training the five case studies sequentially using Algorithm 5.2 as shown in Table 5.2.

Interpretability is verified by considering the cross-member assembly (5) to provide links between MASs requirements and architecture functional elements and obtain 3D Grad-CAMs for key process parameter variations. While obtaining 3D Grad-CAMs, the weights and biases of the network are fixed at the mean values ($\omega = \mu_\omega$).

Before training, all shape errors are pre-processed and voxelized to $(u, v, w, d) = (64, 64, 64, 3)$ voxel grids $\mathbf{V}_{64,64,64,3}$. The deviation features \mathbf{d} include deviations in all directions for all points $(\tilde{x}_k, \tilde{y}_k, \tilde{z}_k)$. The shape error after the final station ($\mathbf{x}^{N_s, 4}$) is used as input while the process parameters \mathbf{y} and *upstream* stations shape errors $\mathbf{x}^{1, 4}, \dots, \mathbf{x}^{N_s-1, 4}$ are used as output. The model architecture hyperparameters were selected as proposed in the OSER-MAS approach. Training hyperparameters were optimized for all scenarios. Adam optimizer [185] is used for training in scenarios (i), (ii) and (iii). Initial learning rates $\alpha_0 = [0.1, 0.01, 0.001, 0.0001]$ were compared for scenario (i) and (ii), $\alpha_0 = 0.001$ gave

optimal performance in terms of error and convergence, $\alpha_0 = [0.1, 0.01]$ gave an inferior performance as compared to $\alpha_0 = [0.001, 0.0001]$, $\alpha_0 = 0.001$ was finally selected as the learning rate gives faster convergence between the two values. The same combinations were tested for scenario (iii) while under the constraint that $\alpha_C = \alpha_F/10$ given the fine-tuning protocol to ensure later layers learn at a faster rate as compared to initial layers. $\alpha_C = 0.0001$ and $\alpha_F = 0.001$ gave the most optimal performance. Scenario (iv) tested the initial learning rate for stochastic gradient descent (SGD), $\alpha_0 = 0.001$ gave the optimal performance. The learning rates were multiplied in each case by the weight uncertainty as described in Table 5.2. Minibatch sizes of 8, 16 and 32 were tested. Larger batch sizes could not be used given the high GPU memory requirements of 3D CNNs. Minibatch size of 32 gave the best performance. Smaller sizes such as 8 and 16 caused the training process to be unstable. The model is trained for 300 epochs. Group normalization [183] with four groups is used after each convolutional layer. This prevents overfitting and accounts for small minibatch size due to GPU memory size constraints and aids in stabilizing the training process. Optimization for training hyperparameters was done for case (1) (Flat Plate Assembly) to ensure computation feasibility. Given case studies (1) to (4) consist of a single station, the 3D U-Net model's decoder is not used as upstream shape error for previous station stations does not need to be estimated. Case studies (5) and (6) leverage the decoder to estimate shape error after upstream stations. The work has been implemented using Python 3.7 and TensorFlow - GPU 2.1 [186] and TensorFlow Probability 0.9. A python library named DLMFG [187] has been developed to validate and replicate the results of the methodology. Both the data generation and evaluation of the approaches have been done using VRM. Two Nvidia Tesla V100 32 GB GPUs are used for model training and deployment.

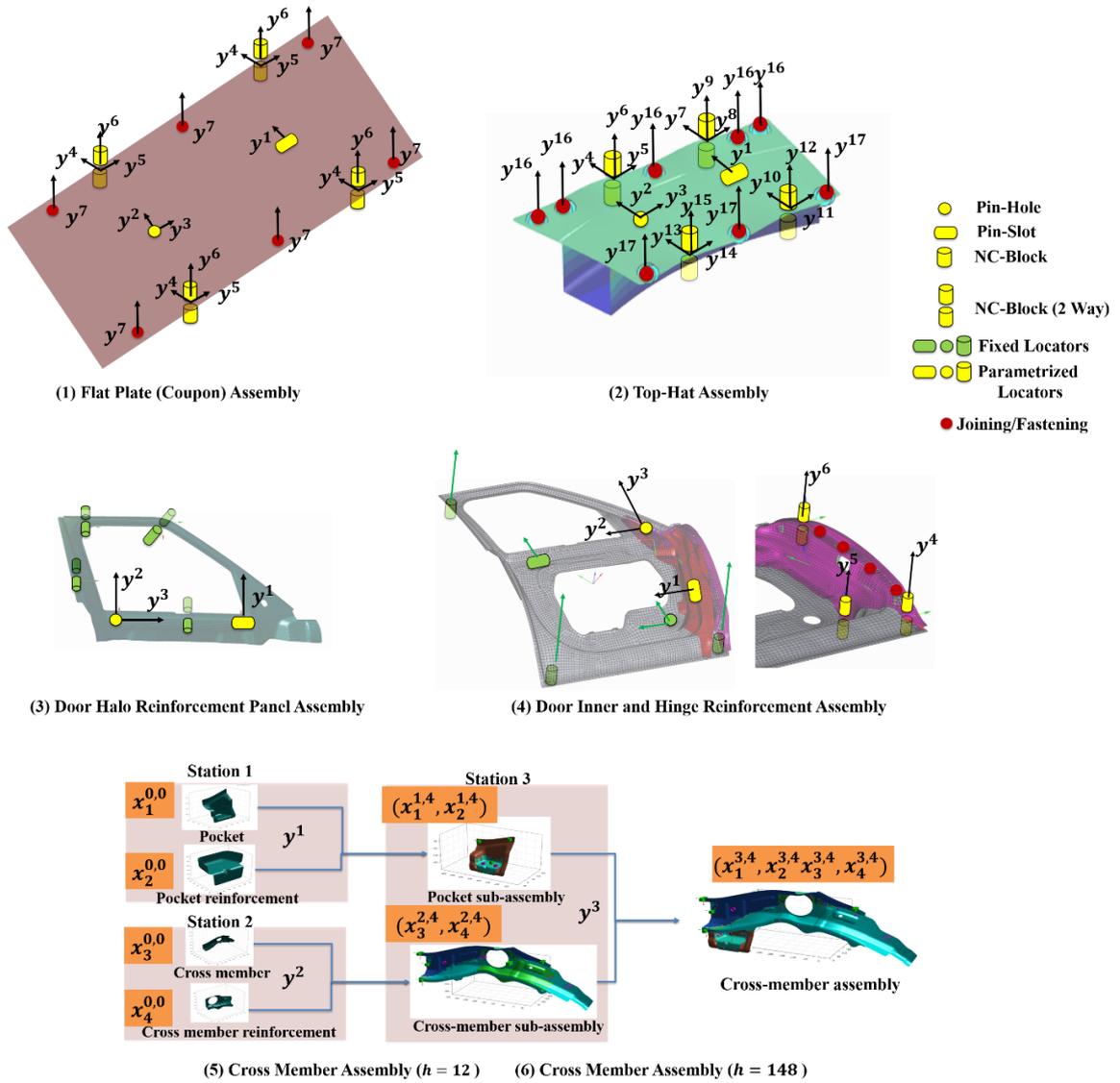


Fig. 5.2. Assembly Cases

Table 5.4: Assembly Cases

Case	Geometry	Input Parts	Stages	Stations	Process Parameters
(1) Flat Plate	2D	1 ideal part	4	1	7
(2) Top-Hat	3D	2 ideal parts	4	1	17
(3) Door Halo	3D	1 ideal parts	2	1	3
(4) Door Inner and Hinge	3D	2 ideal parts	4	1	6
(5) Cross Member	3D	4 non-ideal parts	12	3	12
(6) Cross Member	3D	4 non-ideal parts	12	3	148

5.5.2 Discussion: Scalability

The results for training assembly cases sequentially in the aforementioned scenario are

summarized in Fig. 5.3. Training in all scenarios is done until convergence. Model is considered converged when the performance metrics on the test set $E = (X^E, Y^E)$ are lesser than the threshold. R-Squared (R^2) ≥ 0.90 is considered as the convergence criteria for y^r and Receiving Operating Characteristics – Area Under Curve (ROC-AUC) ≥ 0.90 is considered as a convergence for y^c . The initial training set size T^0 is set to be 500 samples and $T^i = 100$ samples are added in each closed-loop iteration based on the estimated GMM parameters. Based on empirical tests, the pre-specified number of mixtures in the GMM is fixed at $K = 5$.

The results show that for low complexity cases such as Flat Plate (1) and Top-Hat (2) the effects of using closed-loop sampling with continual or transfer learning gives only minor reductions in training samples for converging. As the complexity of the assembly cases increases and the effect of pre-trained weights of continual and transfer learning become a significant reduction up to 50% in the number of required training samples for case (6) can be seen. This validates the need for scalable approaches required for training high-dimensional assembly cases while leveraging the pre-trained models on low-dimensional assembly cases.

The performance measure of using continual learning for all T_n tasks are done by comparing R^2 on the T_i th task when learning is performed only till T_i th task (R_{T_i, T_i}^2) and when learning is performed till the T_n th task (R_{T_i, T_n}^2). Catastrophic forgetting for each task (CF_{T_i}) is quantified as the difference between performance in the aforementioned situations:

$$CF_{T_i} = R_{T_i, T_n}^2 - R_{T_i, T_i}^2 \quad (5.16)$$

Negative value of CF_{T_i} means catastrophic forgetting of previous cases, while positive values mean that learning new tasks has improved the performance of previous tasks. Table V summarizes the results. Average Generalized Performance $\frac{1}{T_n} \sum_{T_i} R_{T_i, T_n}^2$ and Average Catastrophic Forgetting $\frac{1}{T_n} \sum_{T_i} CF_{T_i}$ are summarized and highlight generalized performance (95%) more significant than the required threshold while with

average forgetting of only up to 2.1%.

The convergence is measured by the number of samples (S^{T_s}) for training for the given training scenario T_s . Improvement in convergence Δ_{T_s} for a training scenario T_s is measured as the percentage difference in training samples required for convergence as compared to the baseline training scenario, which for this case is (i) random sampling

$$\Delta_{T_s} = \frac{S^{T_s} - S^{T_{s=(i)}}}{S^{T_{s=(i)}}} \quad (5.17)$$

Table 5.5 also summarizes the improvement in convergence for the CLIP framework, i.e. improvement in convergence when using training scenario (iv) Continual Learning with Closed-Loop Sampling $\Delta_{T_s=(iv)}$. Fig. 5.4 aims to compare loss in performance (CF_{T_i}) with improvement in convergence ($\Delta_{T_s=(iv)}$). Overall, across the six cases, the proposed CLIP framework provides 56% improvement scalability as quantified by improvement in convergence with a loss in performance of only 2.1% as quantified by catastrophic forgetting.

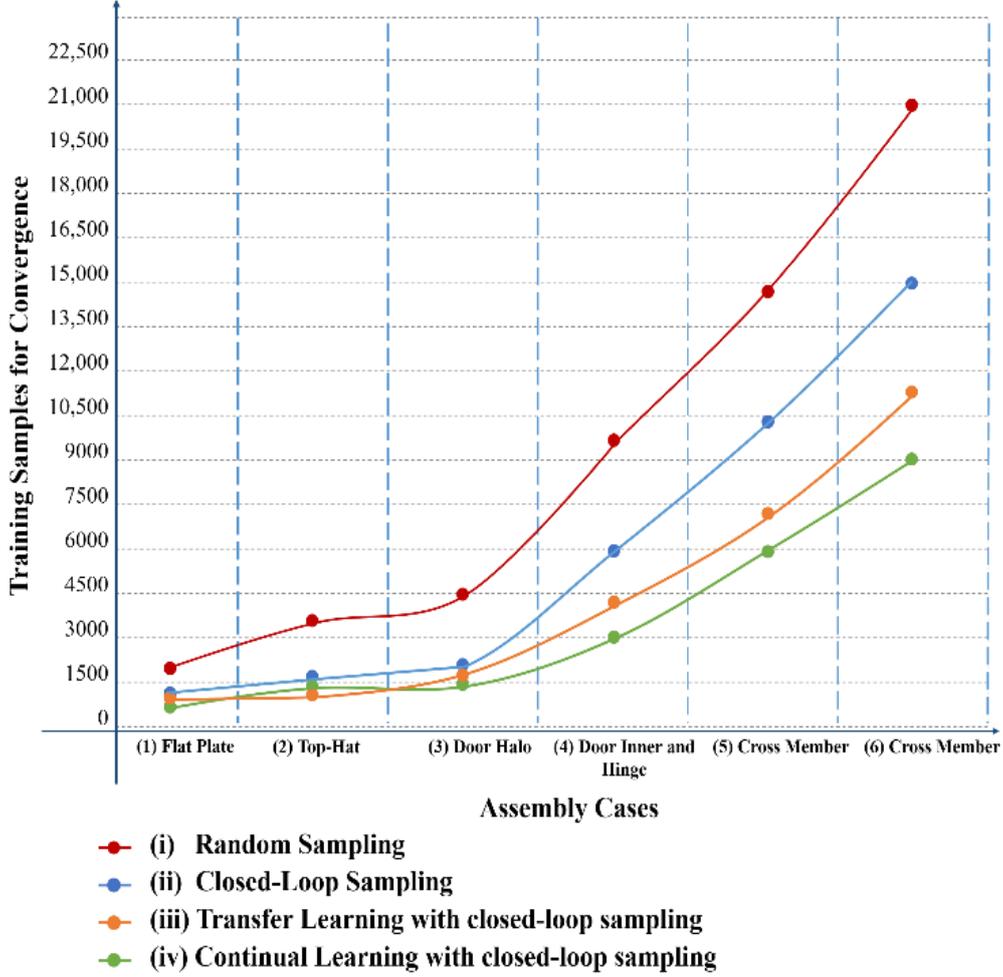


Fig. 5.3. Convergence comparison in all training scenarios

Table 5.5: CLIP framework scalability performance

Assembly Case	R_{T_i, T_n}^2		R_{T_i, T_i}^2		CF_{T_i}	$\Delta_{T_s=(iv)}$
	Mean	SD	Mean	SD		
(1) Flat Plate	0.93	0.01	0.96	0.02	-0.03	0.33
(2) Top-Hat	0.91	0.02	0.95	0.01	-0.04	0.57
(3) Door Halo Reinforcement	0.98	0.01	0.98	0.01	0.00	0.62
(4) Door Inner and Hinge	0.97	0.01	0.98	0.02	-0.01	0.68
(5) Cross Member	0.95	0.02	0.98	0.01	-0.03	0.59
(6) Cross Member	0.96	0.03	0.96	0.03	0.00	0.57

Average Generalized Performance = **0.95**

Average Catastrophic Forgetting = **-0.02**

Average Convergence Improvement = **0.56**

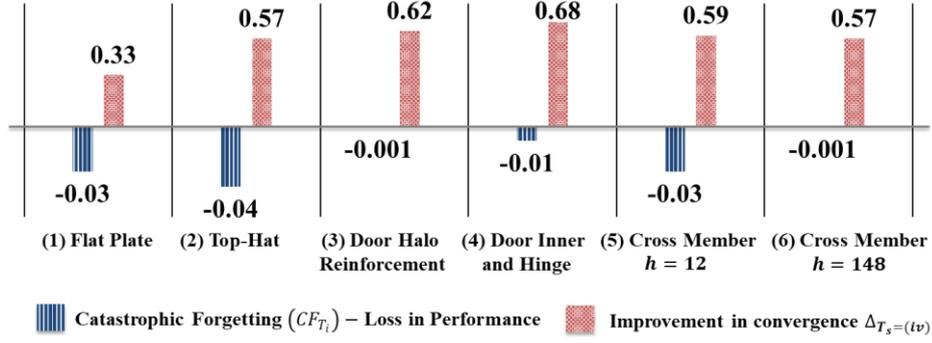


Fig. 5.4. Loss in performance vs improvement in convergence for CLIP

5.5.3 Discussion: Interpretability

The interpretability of the Bayesian 3D U-Net is done on two levels:

(1) By linking requirements of the MASs with functional elements of the Bayesian 3D U-Net architecture. These include:

(1a) *Object Shape Error Voxelization*: Shape error voxelization provides an intermediate 3D data structure linking mesh obtained from CAE simulation and point clouds obtained from 3D optical scanners. Voxelization ensures that both these data structures are converted to voxels and are compatible with 3D convolution operations fulfilling: Requirement (i) high data dimensionality; and, Requirement (vi) dual data generation capabilities. The voxels are multi-channels, with each channel corresponding to one component of shape error. The resolution of voxels depends on the required performance. Fig. 5.5 shows voxelization for one component of shape error at different resolutions. Low-resolution voxels capture global shape error patterns, as the resolution is increased, local shape error patterns are effectively captured. This also increases the discriminative capability required to differentiate between collinear shape error patterns, although this comes at a higher computational cost. Empirical studies have shown no significant increase in performance above $(64 \times 64 \times 64)$ for RCA of MAS. Case (6) Cross Member Assembly ($h = 148$) is used for a sensitivity study. Object Shape Error Reconstruction Error and performance is compared against voxel granularity. The reconstruction error is less than 1% given $(64 \times 64 \times 64)$ or more granular voxels. The model performance does not increase over $R^2 = 0.96$ even with voxels as granular as $(96 \times 96 \times 96)$. Additionally, with increased voxel sizes

above $(64 \times 64 \times 64)$, the minibatch size used during training has to be further reduced to 16 given GPU memory constraints, resulting in unstable model training and, hence, negatively impacts performance. Table 5.6 summarises the results of the sensitivity study.

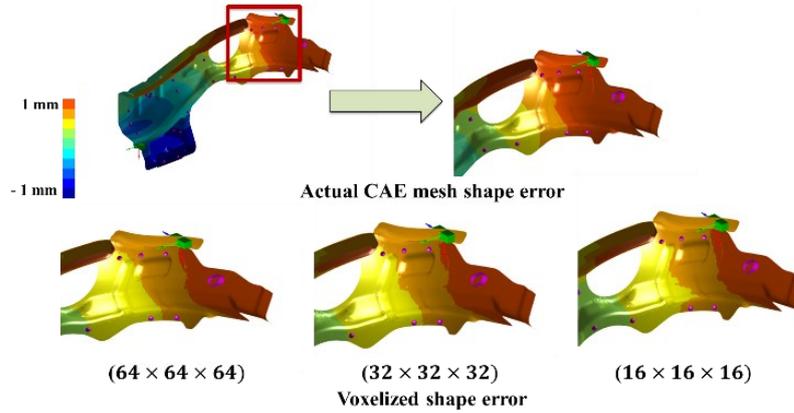


Fig. 5.5. Object Shape Error Voxelization

Table 5.6: Voxel Resolution Sensitivity Study

Voxel Resolution	Reconstruction Error	Performance (R^2)		Minibatch Size
		Mean	SD	
$(16 \times 16 \times 16)$	9.2%	0.85	0.01	64
$(32 \times 32 \times 32)$	3.8%	0.91	0.01	32
$(64 \times 64 \times 64)$	1.4%	0.96	0.02	32
$(80 \times 80 \times 80)$	1.2%	0.96	0.04	16
$(96 \times 96 \times 96)$	0.9%	0.95	0.05	16

(1b) *Encoder with Down Sampling Kernels*: As described earlier, the Bayesian 3D U-Net architecture consists of four levels of the encoder and decoder models (Fig. 4.2). Each level of encoding consists of the down-sampling kernel (see Down-sampling kernel in Fig. 4.2). The kernel consists of 3D Max pooling, which is duplicated to a residual connection and encoding connections. The residual connection consists of a 3D convolution with a filter size of one and a stride length of one in all three dimensions. The encoding connection consists of two 3D convolutions of filter size three and stride length one with ReLU activation in between. Then, the residual connection and the encoding connection are merged using element-wise addition. Finally, ReLU is applied before duplicating the output into the decoder input and next level encoder input. Overall, the down-sampling kernels in the encoder with consecutive 3D convolutions and pooling are essential for spatial correlation filtering,

feature extraction, and non-linear transformations. Consecutive levels of the decoder extract more discriminating features from the high-resolution voxelized shape error input. The discriminative ability of the features increases at each consecutive encoder level thus, enabling an accurate estimate of process parameters hence high root cause isolability. Each level of the encoder is also linked to the corresponding decoder. This enables the transfer of features related to the part geometry and enables accurate estimation of upstream part shape error at the end of the decoder. Fig. 5.6 highlights the features extraction capabilities of different levels of the encoder. In contrast, lower levels focus on the whole part, higher levels focus on the regions that contain the shape error. This enables the fulfilment of requirements: (ii) non-linearity; (iii) collinearities; and, (iv) high faults multiplicity. The 3D Grad-CAMs for encoder levels provides interpretability by visualizing the extracted shape error features. The transparency provided in the extracted shape error features enables interpretability on why a particular root cause was isolated.

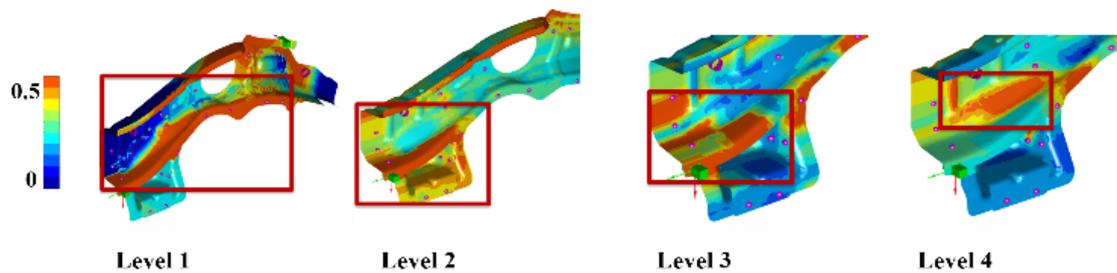


Fig. 5.6. 3D Grad-CAMs for various encoder levels

(1c) *Decoder with Up-sampling Kernels*: Each level of the decoder consists of the up-sampling kernel (see Up-sampling kernel in Fig. 4.2) and provides real-valued segmentation maps that estimate object shape error, i.e., the three components of deviation for each subassembly at the end of all upstream stations. Each level of the decoder consists of two input sources; the encoder input from the corresponding level encoder and the decoder input from the previous decoder level. The following operations are then performed: (i) up-sampling of the decoder input, which is duplicated and sent to the attention gate and feature concatenation layer; (ii) the attention gate [90] distills information from the encoder and then generates relevant features that are concatenated with the up-sampling output from (i); and, (iii) this concatenated feature set is duplicated to the residual connection, and the decoder connection and similar

operations as in the encoder layer are performed. The number of channels of the decoder output equal to the number of components of shape error multiplied by the number of upstream stations, while the granularity of the output is the same as the input voxel size. Various levels of the decoder aggregate features from the corresponding encoder and previous decoder. This integrates part geometry features (as provided by the encoder) with the shape error features (as provided by the previous decoder), enabling accurate estimation of upstream part shape error. Different levels of decoder reconstruct shape error within different regions of the part. This enables the fulfilment of requirement (viii) Fault Localization. Fig. 5.7 highlights the up-sampling capabilities of different levels of the decoder that enable estimation of object shape error of upstream assemblies. Level 2 reconstructs the pocket reinforcement subassembly features, while levels 3 and 4 reconstruct the cross-member reinforcement assembly features. The 3D Grad-CAMs for decoder levels provides interpretability into the stations within which the fault is localized. Fig. 5.8 compares the actual upstream assembly shape errors as estimated using CAE simulation with those estimated by the decoder output.

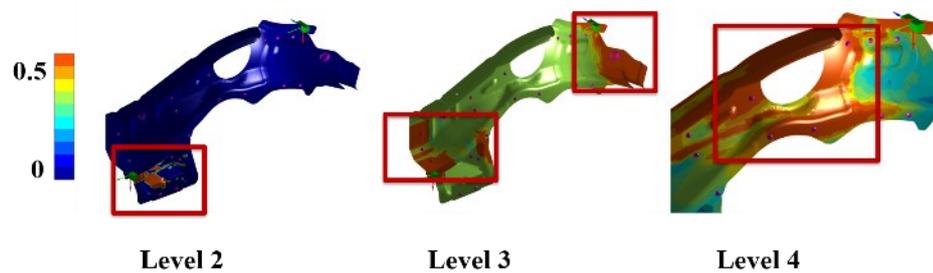


Fig. 5.7. 3D Grad-CAMs for various levels of the decoder

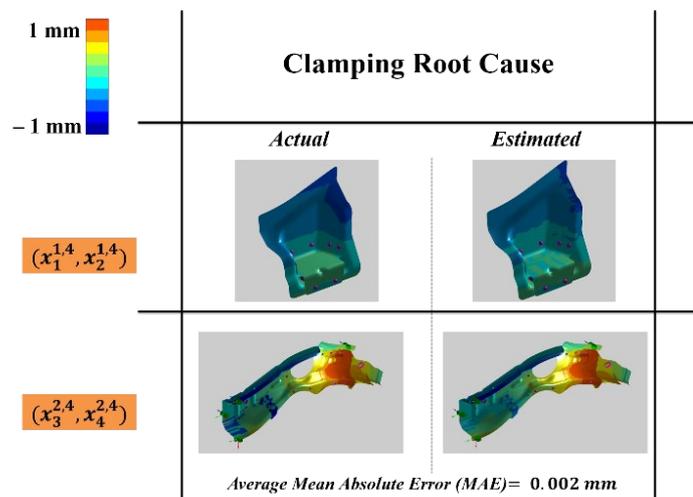


Fig. 5.8. Comparison of actual vs estimated upstream assemblies

(1d) *Attention Gate*: The soft-attention mechanism as proposed by Oktay et al. [90] is used between corresponding levels of the encoder and decoder. The attention approach (see attention gate in Fig. 4.2) allows the model to be specific to local regions. In the context of shape error estimation of assemblies, this helps the model focus on particular parts/subassemblies in each station. Adding of the attention gate increases in accuracy of upstream stations shape error estimation as the model learns where to look within the final assembly to estimate upstream sub-assemblies $[x^{1,4}, \dots, x^{N_s-1,4}]$. This decoder inclusive of attention gates improves performance for requirement (viii) Fault Localization. Fig. 5.9 shows 3D Grad-CAMs for areas of focus at different functions within the up-sampling kernel. Attention 3D Grad-CAMs enables interpretability by providing insights into the decoder's regions in estimating upstream shape errors.

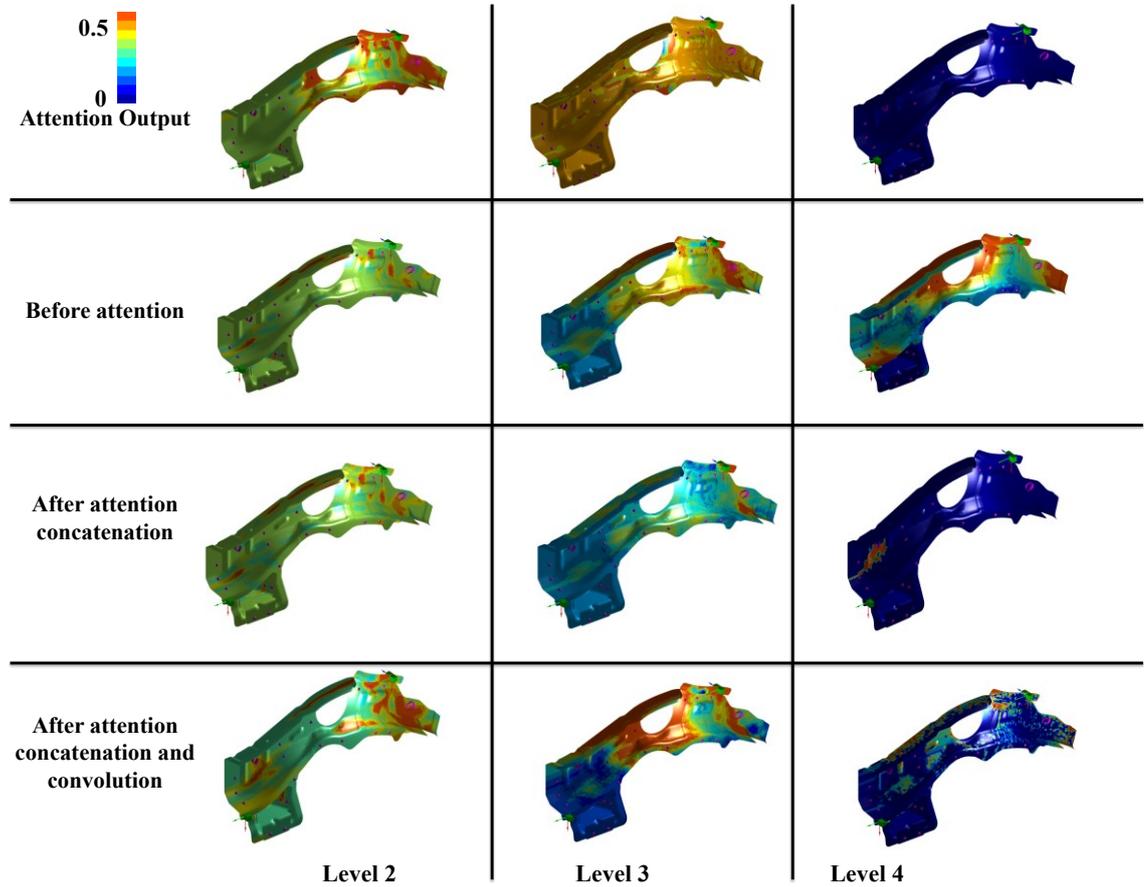


Fig. 5.9. 3D Grad-CAMs for Attention at various levels of the decoder

(1e) *Bayesian Flipout Layers*: Given the uncertainties in the system and the availability of only a limited dataset, a deterministic estimate of function $f(\cdot)$ as shown in (4.4) is not feasible. The Flipout [89] layers leveraged in the encoder enable uncertainty

quantification. These estimates of uncertainty integrate measures of confidence within isolated RC(s) and hence drive costly corrective actions [218]. This is realized by using Bayes-by-Backprop [164], which integrates backpropagation with variational inference [180] to estimate a posterior distribution $q_{\theta}(\omega)$ which is parametrized by θ over the neural network weights based on the pre-specified prior $p(\omega)$. This enables fulfillment of requirement (v) uncertainty quantification. The uncertainties are key elements of interpretability insights as they integrate confidence within the root cause estimates.

(1f) *Residual Connections*: Given the deep architecture of the model, vanishing gradients can be a significant issue. Hence residual [193] or skip connections are added within each down-sampling and up-sampling kernel that ensure effective prorogation of gradients by providing a skip route. Fig. 5.10 highlights the 3D Grad-CAMs for various stages of the residual connection as seen (highlighted in red rectangle) in level three the layer before the residual connection has negligible activations due to skipping of the layer while the gradients become significant after the addition of the residual. The residual connections improve performance for requirements: (ii) non-linearity, (iii) collinearities, and (iv) high faults multiplicity.

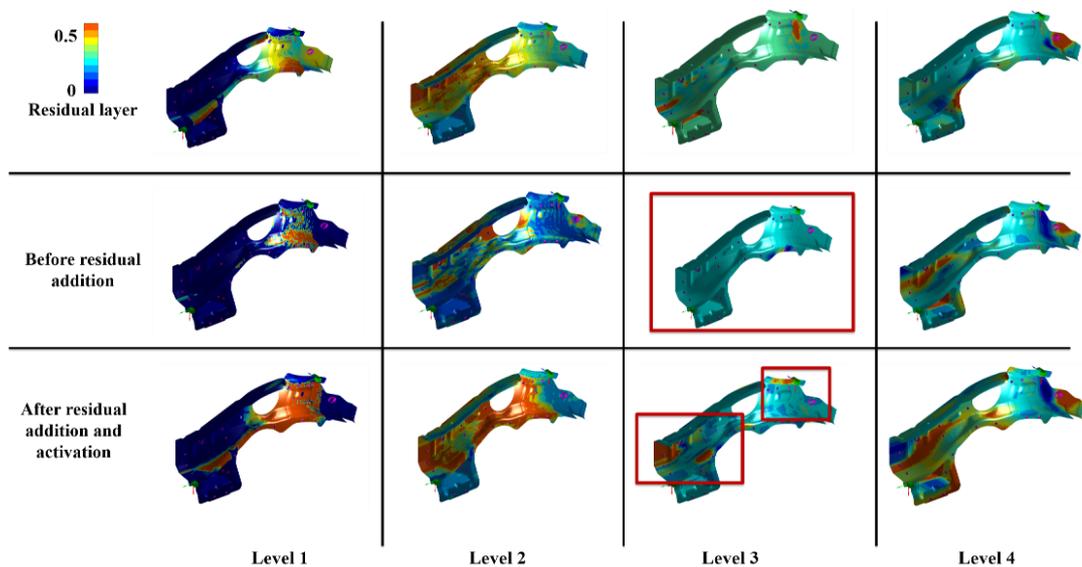


Fig. 5.10. 3D Grad-CAMs for residual connection at various encoder levels

(2) Using 3D Grad-CAMs to interpret the working of the architecture for different process parameter variations or root cause(s). The above 3D Grad-CAMs provide a *global* level of interpretability by linking functional elements of the architecture with

the requirements of the MAS. The next local level of interpretability aims to provide transparency into the 3D Grad-CAMs for various levels of the encoder for key root cause scenarios. This links the shape error features extracted by each encoder level to estimate that particular root cause. To interpret that the architecture is isolating a root cause correctly, the features extracted by various levels of the encoder should correspond to the shape error patterns caused by that root cause. To validate this, the cross member assembly (case (5)) is considered, and the working of the architecture is analysed for five key root cause(s) scenarios:

(2a) *Part Variation Root Cause*: This is caused due to variation in upstream fabrication processes is estimated as variation in $\mathbf{y}^m = \mathbf{y}^1 = 2 \text{ mm}$. Fig. 5.11 represents the output of the assembly given the incoming part (cross-member) $n = 3$ has part variation. The region marked in red depicts a bend in the part that is unique to a part variation root cause [111][220]. The 3D Grad-CAMs, as shown in Fig. 5.11 highlights, that the first encoder focuses around the entire part, the second encoder level can identify the edges near the bend, and the final encoder levels (three and four) can identify the region where the bend has occurred and hence accurately estimate \mathbf{y}^1 as a part variation root cause.

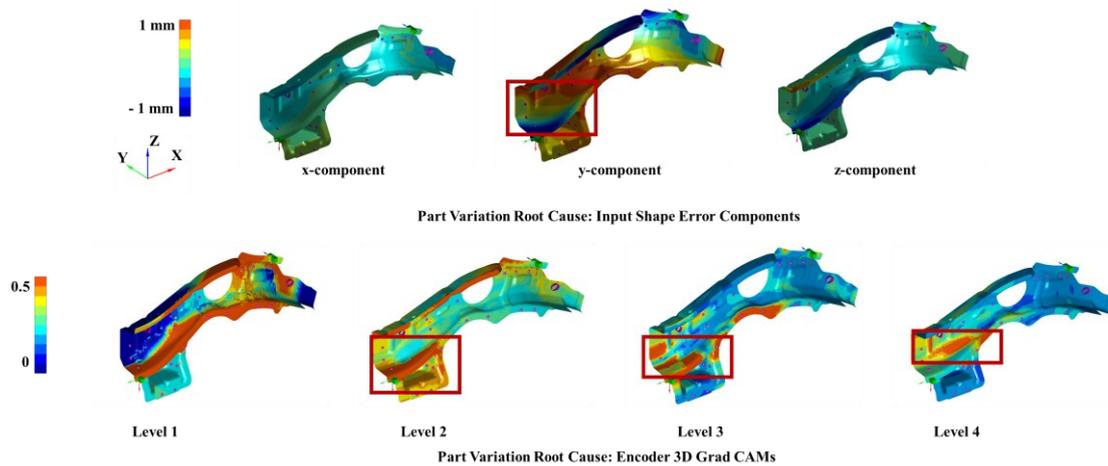


Fig. 5.11. Part Variation Root Cause

(2b) *Positioning Root Cause*: This is caused by tooling installation and calibration error, or tooling deterioration due to gradual wearing out of fixture locators and is estimated as variation in $\mathbf{y}^m = \mathbf{y}^5 = 1 \text{ mm}$. They affect the part placement, including

orientation/reorientation and stability. The 3D Grad-CAMs, as shown in Fig. 5.12, highlights that the encoder focuses around the entire part that has an error in orientation and estimates y^5 as the magnitude of the error.

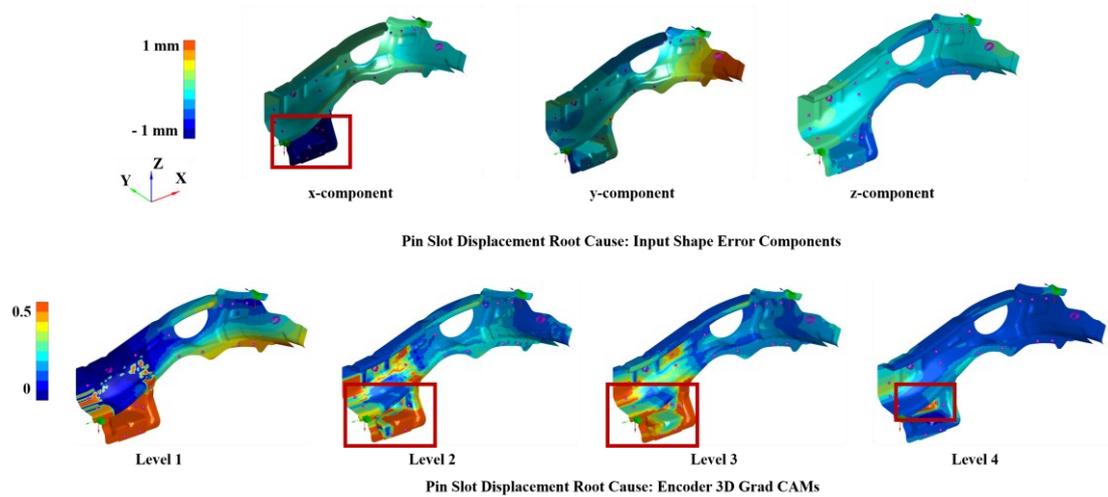


Fig. 5.12. Positioning Root Cause

(2c) *Clamping Root Cause*: This is caused by misalignment of the clamp in the y -direction and estimated as $y^m = y^{11} = 2 \text{ mm}$. They cause part bending of compliant parts. The 3D grad-CAMs, as shown in Fig. 5.13, highlights that the encoder can focus on the local bend pattern at the location of the clamp and estimate y^{11} as the magnitude of the clamp misalignment in the y -direction.

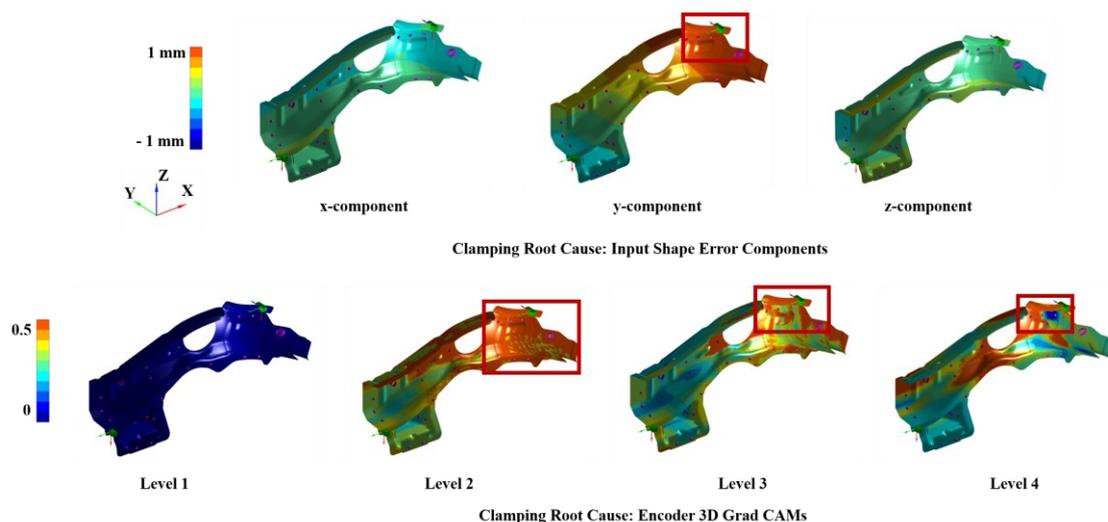


Fig. 5.13. Clamping Root Cause

(2d) *Joining Root Cause*: This is caused by misalignment of the joining tool (SPR) in

the y-direction and estimated as $\mathbf{y}^m = \mathbf{y}^{12} = 2 \text{ mm}$. They lead to a defective joint between the two assemblies. The 3D grad-CAMs as shown in Fig. 5.14 highlights that the first level of the encoder can focus on the region of defective joint and the later levels focus on the subassembly affected due to the defective joint and hence estimates \mathbf{y}^{12} as the magnitude of joining tool misalignment in the y-direction.

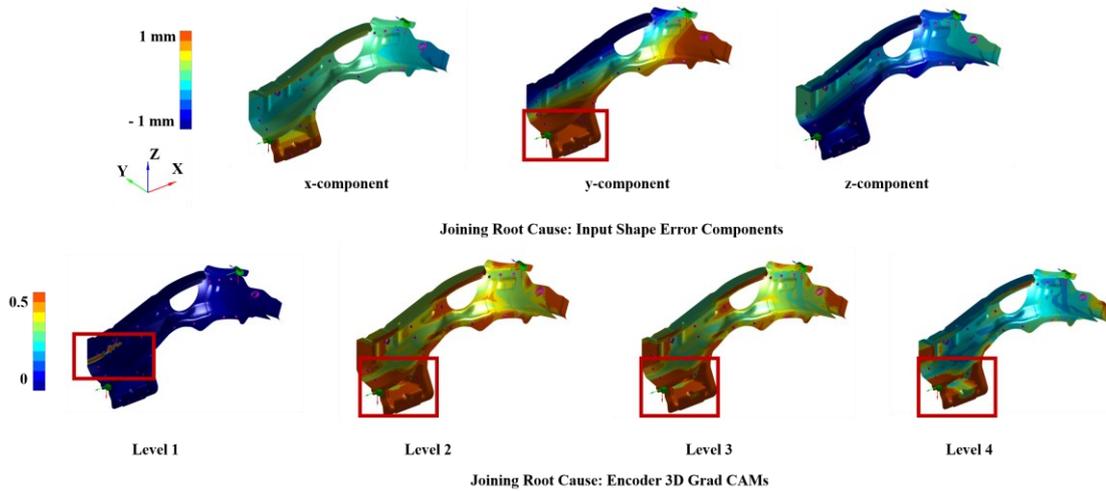


Fig. 5.14. Joining Root Cause

(2e) *Part Variation and Clamping Root Causes*: This is caused when there is an upstream part variation ($\mathbf{y}^m = \mathbf{y}^1 = 2\text{mm}$) and misalignment of the clamp in the y-direction ($\mathbf{y}^m = \mathbf{y}^{11} = 2\text{mm}$). These lead to multiple simultaneous bends across the assembly. The 3D Grad-CAMs, as shown in Fig. 5.15, highlights that various levels of decoder focus on all affected regions within the sub-assembly to estimate multiple root causes simultaneously. This capability is crucial in ensuring that deep learning models have high RCA capabilities even when all process parameters have variation and potentially are at fault (100% fault multiplicity). Such cases of high fault multiplicity cause various shape errors that are collinear (highly similar). The ability of architecture to simultaneously focus on multiple areas within the multi-channel voxelized input and localize various bends, twists and other shape error patterns which are potentially overlapping (interacting) and then relate them to the process parameter(s) causing it, makes it the ideal approach to do RCA of high-dimensional MASs with high fault multiplicity using granular 3D data structures such as mesh (CAE) or point clouds from 3D scanners.

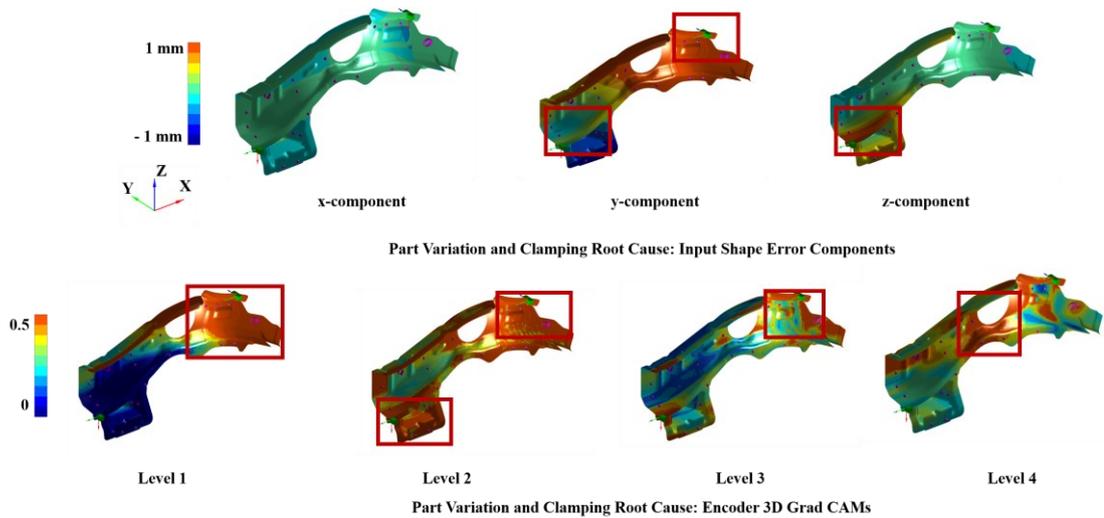


Fig. 5.15. Part Variation and Clamping Root Causes

5.6 Conclusions

The chapter proposed a novel *closed-loop in-process (CLIP)* algorithm portfolio to address the current limitations of scalability and interpretability. Scalability is enabled by leveraging closed-loop training integrated with uncertainty guided continual learning or transfer learning. The approach enables the effective transfer of knowledge through invariant features between MASs thereby, achieving quicker convergence with 56% fewer training samples. The overall loss in performance was limited to only 2.1 %, as quantified by average catastrophic forgetting (Table 5.5). Interpretability is enabled by leveraging 3D Grad-CAMs that provide insights into the functioning of crucial elements within the architecture and relate features extracted by the architecture to shape MAS's error features. The visual interpretability explanations and uncertainty estimates integrate confidence hence, enabling *trust in black-box* deep learning models.

Scalability and interpretability are vital challenges that must be solved to enable widespread adoption of deep learning methodologies in industrial environments. Essential industrial application entails RCA of assembly processes of discrete components made of sheet metal parts used in automotive, aerospace or consumer products industries. These applications will leverage the CLIP approach directly with the OSER approaches to enable scalable and interpretable root cause analysis. They will be especially beneficial for processes with a more significant number of parts and a larger number of assembly stations. The approach can also be leveraged for the

transfer of learning to different types of manufacturing processes such as stamping, machining and additive manufacturing that can be formulated using the proposed formulation of object shape error estimation for root cause analysis. This will lead to leveraging transfer and continual learning to other manufacturing processes that can be linked to assembly processes. Interpretability has been a significant barrier preventing the adoption and deployment of deep learning models in the industry. The interpretability elements proposed by the work aim to eliminate the barrier and integrate context and confidence to the estimates hence, enabling wider adoption.

6 Building a Quality Correction Framework for Assembly Systems using Deep Reinforcement Learning⁴

6.1 Introduction

Previous chapters have focused on building kernels to isolate RCs of object shape errors in MAS. This Chapter proposes a novel deep reinforcement learning-based kernel, OSEC, to mitigate RC of dimensional and geometric product shape errors by determining optimal CA(s). It leverages the DDPG algorithm to learn optimal process parameter correction policies based on isolated RCs and state estimates of MAS. These policies can be interpreted in engineering terms as sequential corrective adjustments of process parameters necessary to mitigate RCs of product shape errors. The approach has the capability to estimate adjustments of process parameters related to fixturing and joining while simultaneously accounting for (xi) Uncertainty of isolated RCs, (xii) Key Performance Indicator (KPI) improvement, (xiii) MAS design architecture; and, (xiv) MAS inherent stochasticity. The OSEC methodology leverages a novel reward function parametrised by user-defined NPI-coefficients that balance the trade-off between benefits and costs and enable application of the proposed approach throughout the NPI lifecycle. Benchmarking the methodology using an industrial, automotive cross-member assembly system demonstrated a 40% increase in corrective actions' effectiveness compared to current manual and SPC/APC based approaches. The contributions of the chapter can be summarized as follows:

(1) Proposed a novel deep reinforcement learning-based *OSEC approach* that leverages a DDPG agent to mitigate object shape errors by learning CA policies that can map

⁴ Based on Sinha, P. Franciosa, and D. Ceglarek, "Building a Quality Correction Framework for Assembly Systems using Deep Reinforcement Learning " in review, Journal of Manufacturing Systems, 2021.

isolated RCs and state of MAS to sequential process parameter adjustments (CA). The CA policy generates a functional nominal that simultaneously optimises for requirements such as RC uncertainty, KPI improvement, MAS design constraints and MAS inherent stochasticity.

(2) Proposed a quantitative formulation for MAS correction that mathematically accounts for the (a) benefits of CAs provided due to RC elimination and cost/quality KPI improvement, (b) the costs incurred due to MAS reconfigurability and adjustment constraints, (c) MAS stochasticity due to incoming part and tooling variations and (d) the NPI stages of the MAS. The formulation proposes a novel correction reward function parametrised by user-defined NPI-coefficients, enabling the fulfilment of all CA requirements and providing capabilities for mitigating object shape error across all stages of the NPI and production lifecycle.

(3) Verified and validated the methodology by implementing it on an industrial, automotive cross member assembly consisting of three stations, four non-ideal parts and twelve process parameters using a CAE simulator known as VRM [6].

(4) Conducted comprehensive benchmarking against the three groups of frameworks used for correction of MAS, that includes (a) first group of approaches that map data to RCs and to CAs based on manual expertise ($CA = -RC$), and (b) second and third group of approaches that directly map data to CAs ($CA = \Delta y$) such as SPC/APC assuming a fixed design nominal and do not consider the NPI stage. The benchmarking highlights superior performance in terms of higher quality KPIs (RC elimination and reduction of object shape error), lower costs (penalty due to MAS reconfigurability and adjustment constraints) and quicker convergence to optimal CAs under varying levels of stochasticity (part and tooling variations).

The rest of the chapter is organized as follows: Section 6.2 formulates the correction problem for MAS using the Markov decision process (MDP); Section 6.3 discusses the OSEC methodology; Section 6.4 presents the industrial case study; and, finally, conclusions are summarized in Section 6.5.

6.2 Problem Formulation

The application of OSEC requires the formulation of the correction problem as an MDP so that DDPG agent can be trained to perform CA. As stated in Chapter 4, the MAS (Fig. 4.1) can be represented as a process tree where different nodes correspond to stages within a single assembly station (Fig. 4.1a) or as stations within the assembly system (Fig. 4.1b). A station consists of four stages: positioning, clamping, fastening, and release (PCFR). Object shape errors can be induced in any station by variations in one or multiple process parameter(s). These errors are propagated and accumulate in a non-additive manner [72][51]. At any given state, the variation in the process parameter(s) from the nominal is a source of shape error and is known as the RC(s) of the shape errors. CAs involve adjustments of controllable/reconfigurable process parameters to mitigate RCs of object shape errors.

The object shape error response and RCA problem has been previously formulated in Chapter 4 (Section 4.2). Within the formulation, stations in the MAS are represented by $s: s= 1, \dots, N_s$, where N_s represents the total number of stations within the system and $\acute{s}: \acute{s} = 1,2,3,4$ represents the four stages within each station. $s \acute{=} 4$ represents the final stage of the assembly station. The MAS consists of n incoming parts also known as objects that need to be assembled. Any object $o: o = 1, \dots, n$ at its design nominal shape is characterised by a set of nominal points $\mathbf{P}_o = \{\mathbf{p}_{ok}\}$, $k = 1, \dots, n_o$, where \mathbf{p}_{ok} is a vector consisting of the coordinates of the k th input point and n_o represents the total number of points on object o . $\mathbf{d}_o = \{\mathbf{d}_{ok}\}$ denotes the deviation of each point k after the nominal object o has gone through different stages of the process, \mathbf{d}_{ok} is a vector comprised of deviations of each point in x, y and z axes on object o . As the object o goes through multiple stations and stages within the stations, the set of points are represented as $\mathbf{P}_o^{s,\acute{s}}$ while $\mathbf{d}_o^{s,\acute{s}}$ represents the corresponding shape error. Object shape error for object o after station s and stage \acute{s} can be represented as:

$$\mathbf{x}_o^{s,\acute{s}} = (\mathbf{P}_o^{s,\acute{s}}, \mathbf{d}_o^{s,\acute{s}}) \quad (6.1)$$

where $\mathbf{x}_o^{0,0} = (\mathbf{P}_o^{0,0}, \mathbf{d}_o^{0,0})$ represents the incoming non-ideal part inclusive of the part

variations. A subassembly of incoming parts after the stage \acute{s} of station \mathbf{s} is represented as the set of objects $(\mathbf{x}_1^{s,\acute{s}}, \dots, \mathbf{x}_o^{s,\acute{s}})$. The set of process parameters within the MAS are real-valued and represented by a vector \mathbf{y} for the total set \mathbf{h} of process parameters across the entire MAS. An assumption made in this chapter is that point cloud data is only collected at the end of MAS $\mathbf{x}^{N_s,4}$ not at the end of intermediate stations $\{\mathbf{x}^{1,4}, \dots, \mathbf{x}^{N_s-1,4}\}$. This is known as end-of-line sensing [17]. This is generally the case with 3D scanners as they are expensive and have a high measurement time. Installing scanners at end-of-line ensures products can be measured without any interruption in the production line.

OSER-MAS (Chapter 4) model learns a function using Bayesian 3D U-Net based Convolutional Neural Networks (CNN) and Computer-Aided Engineering (CAE) simulations that takes as input the combined object shape error at the end of the MAS $\mathbf{x}^{s=N_s, \acute{s}=4}$, i.e., after the last station of the MAS ($s = N_s$) and final stage ($\acute{s} = 4$) of the station, and estimates the process parameters across the entire system \mathbf{y} with uncertainties $\sigma(\mathbf{y})$ and the object shape error for all objects at the last stage ($\acute{s} = 4$) of all upstream stations ($s = 1, 2, \dots, N_s - 1$) collectively represented as $\{\mathbf{x}^{1,4}, \dots, \mathbf{x}^{N_s-1,4}\}$. Overall, the inputs and outputs of the RCA function $f(\cdot)$ can be represented as:

$$\{\mathbf{y}, \sigma(\mathbf{y}), \mathbf{x}^{1,4}, \dots, \mathbf{x}^{N_s-1,4}\} = OSER(\mathbf{x}^{N_s,4}) \quad (6.2)$$

The RCs are estimated as a subset of process parameters $\mathbf{RC} \subseteq \mathbf{y}$ and hence, are directly linked to the point cloud data by the OSER model. The model has capabilities to estimate one or multiple RCs. The Chapter assumes that the MAS is diagnosable (i.e., one or multiple RCs can be estimated based on input point cloud data collected) and the OSER model, i.e., the measurement data and model capability is sufficient such that all possible RCs (single RC or multiple RCs) can be discriminated. However, this may not always be the case as the data is collected only at the end of MAS ($\mathbf{x}^{N_s,4}$) (end-of-line sensing) and distributed sensing approaches that collect data at multiple stages within the MAS may need to be used to ensure all RC combinations can be discriminated. For notation consistency, the dimension of the \mathbf{RC} vector is the same as

the process parameter vector \mathbf{y} , i.e., the total of h process parameters and all process parameters that are not RCs are assumed at their nominal (considered zero for simplicity). This also translates into the process parameter uncertainty being equal to the RC estimation uncertainty $\sigma(\mathbf{y}) = \sigma(\mathbf{RC})$.

The Chapter extends the object shape error response formulation to leverage the outputs of the OSER RCA model as shown in (6.2) to define state, action and reward and formally formulate the correction problem as an MDP. Firstly, the state representation of the MAS \mathbf{s}_t^{MAS} for time step (t) is constructed for which the upstream shape errors $\{\mathbf{x}^{1,4}, \dots, \mathbf{x}^{N_s-1,4}\}$ are consolidated into a fixed-length vector $\mathbf{x}^{upstream}$ using a dimensionality reduction function $g(\cdot)$. The 3D CNN model as proposed in the OSER methodology (Chapter 3) is leveraged as $g(\cdot)$ as this enables the extraction of local and global features from the upstream shape errors. The output after the last hidden layer with 64 nodes is leveraged as the consolidated shape error vector with dimension 64 for each set of shape errors. This is concatenated across all $N_s - 1$ shape errors after all upstream stations and has a length of $d = 64 \times (N_s - 1)$

$$\mathbf{x}^{upstream} = g(\mathbf{x}^{1,4}, \dots, \mathbf{x}^{N_s-1,4}) \quad (6.3)$$

In practice, a time step (t) corresponds to a sample of \mathbf{s}_p products rather than a single product, i.e., after a sample of \mathbf{s}_p objects that need to be assembled have gone through the respective stations and stages. At the end of this time step, the state (\mathbf{s}_t^{MAS}) of the MAS is described by a consolidated set of estimated \mathbf{RC}_t , their estimation uncertainties $\sigma(\mathbf{RC}_t)$ and the consolidated set of all upstream shape errors $\mathbf{x}_t^{upstream}$:

$$\mathbf{s}_t^{MAS} = \{\mathbf{RC}_t, \sigma(\mathbf{RC}_t), \mathbf{x}_t^{upstream}\} \quad (6.4)$$

Where \mathbf{RC}_t is vector of dimension h corresponding to RCs within the MAS, $\sigma(\mathbf{RC}_t)$ is a vector of dimension h corresponding to the uncertainty in the respective RCs and $\mathbf{x}_t^{upstream}$ is a consolidated vector of all upstream shape errors as shown in (6.3). The \mathbf{RC}_t vector can also be augmented with other data collected from other sensors distributed across the MAS. This set is the *Markovian* representation of the MAS's

current state that contains all information required (i.e., access to earlier states of the MAS is not required) for the agent to take a CA.

Apart from the current state \mathbf{s}_t^{MAS} , the DDPG agent needs a *reward signal* to maximise while learning a CA policy (behaviour function that generates sequential process parameter adjustments) to estimate CA-value and determine optimal CAs (process parameter adjustments). A sequence of these CAs is known as an *episode*. The episode eventually terminates into states with negligible shape error. The reward should primarily account for the aforementioned four RC and MAS requirements (xi), (xii), (xiii), (xiv) and should be useable across all stages of the NPI lifecycle. The chapter formulates such as reward function by leveraging a set of NPI-coefficients δ .

The first part of the reward aims to quantify the *benefits* gained due to a CA taken at timestep t (after observing \mathbf{s}_p products). This part also ensures that RC requirements (r_δ^{RC} , determined based on the NPI stage and coefficients δ) are met and can be formulated as:

$$r_\delta^{RC} = -(1 - \sigma^{RC_t}) \frac{1}{k_n} \sum_{i=1}^{k_n} K^i (KPI_{nominal}^i - KPI_{t+sp}^i)^2 \quad (6.5)$$

r_δ^{RC} quantifies the loss of k_n KPIs deviating from their nominal $KPI_{nominal}$ across the system leveraging a Taguchi loss function [85]. Each KPI has a relevance coefficient K^i that accounts for the KPI importance and can be used as a measure for the sensitivity of the KPI to the overall cost or quality (cost of quality). The reward is discounted by a normalised uncertainty [0,1] factor σ^{RC_t} which is a summation of uncertainty in all process parameters $\sigma(\mathbf{RC}_t)$. σ^{RC_t} is 0 when uncertainty is below a minimum threshold and 1 when uncertainty is above the maximum threshold. This effectively translates into the reward being near zero for CAs taken when the estimated RCs are highly uncertain. Collectively this mathematically formulates req. (xi) *Uncertainty of isolated RCs and req. (xii) KPI improvement*.

The next part of the reward aims to quantify the *costs* incurred due to a CA

being taken at timestep t (after observing \mathbf{s}_p products). This part also ensures that MAS requirements (r_δ^{CA} determined based on the NPI stage and coefficients δ) are met and can be formulated as:

$$r_t^{CA} = -(1 - \sigma^{RC_t})\lambda \frac{1}{h} \sum_{j=1}^h C^j (y_{t+sp}^j - RC_t^j)^2 \quad (6.6)$$

r_t^{CA} quantifies the cost for violating req. (xiii) *MAS design architecture* by including a process parameter change penalty that penalises the agent from frequently changing h process parameters. The negative of this term ($-r_t^{CA}$) can also be interpreted as a *MAS requirement penalty* as it penalises the agent from taking frequent infeasible CAs that violate MAS reconfigurability constraints. Permitted reconfigurability of a MAS is measured by the degree of freedom for adjustments of various tools, the allowable ranges of adjustments and cost of one or multiple adjustments. For consistency, each degree of freedom (DOF) for a given tool (e.g. pin-hole, pin-slot, or NC-block) is considered a unique process parameter. One or multiple of these process parameters can be adjusted if selected as a CA. Each process parameter y^j can only vary between allowable limits $[y_{min}^j, y_{max}^j]$. Any adjustment to a process parameter is associated with a cost coefficient (C^j) which accounts for the costs of changing specific process parameters to correct RCs. Correcting RCs requires a sequence of CAs (episode). Each time a CA is performed within an episode, a cost proportional to the cost coefficient C^j is incurred. High values of C^j can be leveraged to constrain the agent from changing certain degrees of freedom for various process parameters. The overall impact of penalty for changing process parameters is controlled by a MAS permitted reconfigurability coefficient (λ), which can be used to limit the frequency of CAs. This accounts for MAS constraints such as permitted maintenance duration and frequency. The coefficient also enables trade-off between the benefits (RC related requirements) and costs (MAS related requirements) of the reward function based on the current MAS reconfigurability. To account for RC estimation uncertainty r_t^{CA} is also multiplied by the uncertainty factor.

The reward further accounts for req. (xiv) *MAS system inherent stochasticity*

due to incoming non-ideal part variations and tooling variations by allowing the agent to take CAs only after observing a maximum sample of products s_p products rather than a single product. This is crucial as there are varying levels of stochasticity due to batch-to-batch variation in incoming parts. CAs determined based on previous batch stochasticity may be ineffective if the stochasticity variation pattern and magnitude has changed in subsequent batches. The sample size can be determined based on the required confidence interval [9]. High values of s_p translate into increased confidence scores for MAS stochasticity but also into higher costs as more product samples would need to be measured. Overall, the reward function that the agent should maximise to determine CAs while accounting for the aforementioned requirements can be written as a combination of r_{δ}^{RC} and r_{δ}^{CA} as:

$$r_t = -(1 - \sigma^{RC_t}) \left(\frac{1}{s^p * k^n} \sum_{p=1}^{s^p} \sum_{i=1}^{k_n} K^i (KPI_{nominal}^i - KPI_{t+s^p}^{i,p})^2 + \lambda \frac{1}{h} \sum_{j=1}^h C^j (y_{t+s^p}^j - RC_t^j)^2 \right) \quad (6.7)$$

Lastly the set of NPI-coefficients δ is leveraged to account for the stage of the NPI lifecycle. The set δ consists of:

$$\delta = \{\mathbf{K}, \mathbf{C}, \lambda, s_p\} \quad (6.8)$$

This includes set of relevant KPIs importance (\mathbf{K}), adjustment costs of reconfigurable process parameters (\mathbf{C}), MAS permitted reconfigurability (λ), maximum available data sample (s_p). The NPI-coefficients enable continual tuning of the reward based on the stage of NPI lifecycle. As the MAS goes through the lifecycle the priorities of different requirements related to benefits and costs can change. Using a set of user-parametrizable NPI-coefficients ensures that the CA methodology can be leveraged throughout the lifecycle by fine-tuning the NPI-coefficients based on requirements of the current stage. The coefficients are leveraged to adapt the reward function for agents to learn robust CA policies across different stages of the NPI lifecycle. Such examples for adaption are discussed in later sections. Intuitively in the design stage NPI

coefficients are selected to focus primarily on improving quality KPIs while during production stages importance is given to both cost and quality KPIs.

Correction involves learning a deterministic CA policy $\pi(\cdot)$ which is a function that maps the current MAS state \mathbf{s}_t^{MAS} into a CA. The CA physically translates into a new set of process parameters (or functional nominal) \mathbf{y}_{t+s_p} , that can maximise reward r_t and eventually, eliminate the shape errors $\mathbf{x}^{Ns,4}$.

$$\mathbf{CA}_{t+s_p} = \mathbf{y}_{t+s_p} = \pi(\mathbf{s}_t^{MAS}) \quad (6.9)$$

A \mathbf{CA}_t taken by the DDPG agent corrects one or multiple process parameters within the allowable ranges that aim to maximize the overall reward.

$$\mathbf{CA}_t = \mathbf{y}_{t+s_p} \quad (6.10)$$

In summary, the correction of MASs can be formalised as a Markov Decision process where an environment E , i.e., MAS, can exist in a set of possible states. CAs are taken to correct the MAS to observe a real-valued reward $r_t(\mathbf{s}_t^{MAS}, \mathbf{CA}_t)$ that ensures that the aforementioned requirements are met. The Markovian property, i.e., *the effects of an action taken in a state, depends only on that state and not on historical states*, is applicable for the correction process. Within an episode of sequential CAs only the previous MAS state is considered while determining the CA. The overall reward $R_t^{s_t^{MAS}}$ for a state \mathbf{s}_t^{MAS} is defined as the cumulative discounted sum over the next T time steps where γ is the discount factor that discounts rewards in the future and gives higher weights to current rewards. As stated earlier, for correction of MAS, each timestep is a sample of s_p products.

$$R_t^{s_t^{MAS}} = \sum_{i=t}^T \gamma^{(i-t)} r(\mathbf{s}_t^{MAS}, \mathbf{CA}_i) \quad (6.11)$$

In practice, the MAS state and CAs are deterministic – each state and CA lead to a new state. The states are observable – the new state after a CA is known. The goal is to learn a CA policy π using DDPG that generates optimal CAs \mathbf{CA}_t based on the current MAS state \mathbf{s}_t^{MAS} that maximises the expected reward $R_t^{s_t^{MAS}}$.

6.3 Methodology: Object Shape Error Correction

The OSEC aims to enable shape error correction in an end-to-end manner by first

leveraging RCA models to use manufacturing data (point clouds) to isolate RCs and estimate MAS state and then map the isolated RC and MAS state into effective and feasible CAs using DDPG within MDP setup. The overall approach includes steps to:

(i) *Isolate RCs and estimate MAS state*: Involves point cloud data $\mathbf{x}^{S=N_s, \acute{S}=4}$ collection using 3D scanners (6.1). Further RCs, RC uncertainties $\sigma(\mathbf{RC}_t)$ and upstream shape errors $\mathbf{x}_t^{upstream}$ are estimated using the OSER model (6.2). These collectively represent the MAS state \mathbf{s}_t^{MAS} (6.4).

(ii) *Account for CA requirements (RC and MAS related)*: Involves defining the reward function r_t (6.7) subject to RC uncertainty, MAS corrections constraints (reconfigurability) and inherent stochasticity (part and process variation). Further allocation of NPI-coefficients $\delta = \{K, C, \lambda, s_p\}$ (6.8) is done based on the NPI and production stage to optimally trade-off the aforementioned requirements.

(iii) *Generate feasible and effective CAs*: The correction problem is formulated as an MDP as described in the previous section to bound the scope of possible CA policies $\pi(\mathbf{s}_t^{MAS})$ (6.9) that can be learnt and enable the application of DDPG. DDPG agent is leveraged as it can take as input real-valued multi-dimensional state estimates and output real-valued multi-dimensional CAs \mathbf{CA}_{t+s_p} (6.10). The DDPG architecture is optimized to enable the mapping of RCs to CAs based on the engineered reward function. This involves selecting the hyperparameters of the DDPG agent network architecture, matching the output layer nodes with the CA output dimension (i.e., adjustable process parameters) and selecting an exploration policy to comprehensively explore the performance of various CA policies. The DDPG agent is trained based on the engineered reward function to learn CA policies π (6.9) that can generate feasible and effective CAs. Finally, the trained DDPG agent is deployed to obtain CAs for elimination of RCs within MAS. The steps are summarized in Fig. 6.1.

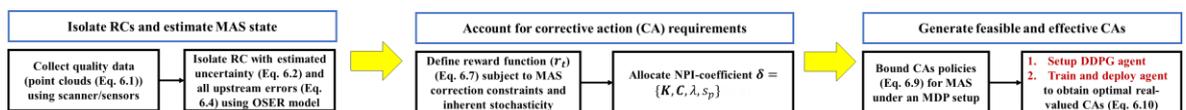


Fig. 6.1. OSEC approach

6.3.1 Engineering Interpretation of CA Policies

From a MAS engineering perspective, CAs are one or multiple process parameter adjustments generated sequentially by a CA policy that mitigates one or multiple RCs, leading to quality defects elimination and KPI improvement. Given a MAS with h process parameters \mathbf{y} and no RCs, all values of \mathbf{y} are at their design nominal $\check{\mathbf{y}}$ (considered 0 for simplicity). Generally, it is assumed that after designing the MAS, the MAS is a deterministic function $\phi: \mathbb{R}^h \rightarrow \mathbb{R}^{k^n}$ that maps h process parameters to k^n KPIs (KPIs = $\phi(\mathbf{y})$). The function ϕ is assumed to have a single optimum, i.e., nominal KPI performance when $\mathbf{y} = \check{\mathbf{y}}$. In the presence of RCs ($\mathbf{RC}_t = \mathbf{y}_t$) the value of \mathbf{y} is estimated by using various RCA models (Chapter 4) or process sensors. A CA, i.e., a process parameter adjustment, is determined to eliminate the RC. Hence in a general case (i.e., a single optimum for the ϕ exist, ϕ is a deterministic assembly function with no constraints on \mathbf{y}), there exists a *one-to-one mapping between CAs and RCs* such that RCs can be corrected and optimal KPI performance can be achieved by leveraging

$$CA = \check{\mathbf{y}} - \mathbf{RC}_t = -\mathbf{RC}_t \quad (6.12)$$

This correction policy is known as *Mean RC Estimates policy* π^0 which generates CAs, which are equivalent to the difference between the mean value of the RC \mathbf{RC}_t and design nominal $\check{\mathbf{y}}$. This policy is currently used in manual approaches and also in SPC/APC methods $CA = \check{\mathbf{y}} - \mathbf{y}_t$ that aim to revert the process parameters to the design nominal.

However, as outlined, CA and RC requirements pose various challenges in determining optimal CAs using π^0 policy mapping. Firstly, given various RCA models use an estimator function to estimate \mathbf{RC}_t , this can result in a biased and high variance estimate of the actual value of RCs. This requirement is known as *(xi) Uncertainty is isolated RC*. OSER leverages Bayesian deep learning to estimate a distribution on \mathbf{RC} : $[\mathbf{RC}_t, \sigma(\mathbf{RC}_t)]$. Generating CAs now requires mapping a distribution to process parameter adjustments (CAs). This means multiple CAs can now be mapped by drawing multiple samples from the RC distribution. Using only the mean of this distribution will give lower convergence and will require a higher number of sequential

process adjustments to converge to the optimal CA that revert the system to $\tilde{\mathbf{y}}$. Alternatively, the proposed OSEC approach learns a policy π^{OSEC} aims to account for this by factoring in an uncertainty factor which results in the reward $r_{\delta}^{RC} = -(1 - \sigma^{RC_t}) \frac{1}{k_n} \sum_{i=1}^{k_n} K^i (\phi(\mathbf{RC}_t)^i - \phi(\tilde{\mathbf{y}})^i)^2$. This ensures the agent is only rewarded for CAs when uncertainty in isolated RC is low; if the uncertainty in isolated RC is high, the agent is not rewarded irrespective of its CA. This introduces an ability to *wait* and observe more MAS states before the RC uncertainty increases and the respective CAs are assigned higher rewards.

The other three requirements state that the assumed design nominal $\tilde{\mathbf{y}}$ may not be the value for optimal performance of ϕ and hence a *functional nominal* $\tilde{\mathbf{y}}$ should be estimated, and then the CA should be determined as:

$$CA = \tilde{\mathbf{y}} - \mathbf{RC}_t \quad (6.13)$$

The goal of the OSEC CA policy π^{OSEC} is to also account for the remaining three requirements while estimating a functional nominal and then generate CAs. The requirements are accounted for as follows:

(xii) *KPI improvement*: In practice, MAS systems are ill-conditioned, there may not exist a unique value $\tilde{\mathbf{y}}$ that generates nominal performance for all KPIs. Multiple process parameter configurations can exist that would lead to local optimum for a subset of KPIs. OSEC aims to handle this by leveraging optimal relevant KPIs importance \mathbf{K} for all KPIs. This constructs a weighted sum of KPIs, which can be maximized by estimating the functional nominal $\tilde{\mathbf{y}}$ after which the CA can be determined using (6.13).

(xiii) *MAS design architecture* entails that not all process parameter adjustments can be implemented due to constraints in MAS reconfigurability, production schedules and adjustment costs that limit the feasible space of CAs. In such scenarios, the CA that reverts the MAS to the design nominal $\tilde{\mathbf{y}}$ may be infeasible to achieve in practice; hence a functional nominal $\tilde{\mathbf{y}}$ that achieves similar KPI performance must be determined to estimate CAs (6.13). OSEC aims to leverage NPI-coefficients, namely adjustment costs

of reconfigurable process parameters \mathbf{C} and MAS permitted reconfigurability λ to penalize the agent from learning CA policies that output infeasible CAs.

(xiv) *MAS inherent stochasticity* due to factors such as non-ideal part variations entails that a deterministic estimate of the MAS $\phi: \mathbb{R}^h \rightarrow \mathbb{R}^{k_n}$ will not exist. Hence a unique value for $\tilde{\mathbf{y}}$ cannot be determined. As ϕ varies due to change in batch-to-batch stochasticity, the optimal $\tilde{\mathbf{y}}$ will also vary, which will result in CAs being ineffective under different stochastic variations. OSEC aims to learn CA policies that dynamically account for other system information apart from RCs such as upstream shape errors by augmenting the state information with upstream shape errors ($\mathbf{x}^{upstream}$) such that the state is represented as $\mathbf{s}_t^{MAS} = [\mathbf{RC}_t, \boldsymbol{\sigma}(\mathbf{RC}_t), \mathbf{x}^{upstream}]$. It also aims first to observe a sample of \mathbf{s}_p products which provide estimates for current stochasticity and a confidence level before determining the functional nominal and the corresponding CA.

Overall policies π^0 and π^{OSEC} fundamentally differ on how CAs are generated. CAs generated by π^0 aims to transition the MAS from the current process parameters (RCs) to the design nominal, while CAs generated by the OSEC policy π^{OSEC} aims to transition the system to a functional nominal that is determined by maximizing a combination of KPI improvement and MAS requirement penalty function within the allowable ranges of the process parameters $[y_{min}, y_{max}]$:

$$\max_{\mathbf{y} \in [y_{min}, y_{max}]} \left(-(1 - \sigma^{\mathbf{y}_t}) \left(\frac{1}{s^p * k_n} \sum_{p=1}^{s^p} \sum_{i=1}^{k_n} K^i (\phi(\mathbf{RC}_t)^i - \phi(\tilde{\mathbf{y}})^{i,p})^2 + \lambda \frac{1}{h} \sum_{j=1}^h C^j (\mathbf{y}_{t+s^p}^j - \mathbf{RC}_t^j)^2 \right) \right) \quad (6.14)$$

If the effect of the aforementioned requirements is insignificant and the MAS requirement change penalty is zero ($\lambda=0$), then theoretically the functional nominal will be the same as the design nominal ($\tilde{\mathbf{y}} = \mathbf{y}$) and the policies will converge towards π^0 (6.12), i.e., restoring the MAS to the design nominal. Under conditions of RC uncertainty and MAS stochasticity the π^0 CA policy might perform poorly but after

convergence π^{OSEC} will always perform as good as π^0 . In summary CA policies must account for all these requirements related to benefits, costs, stochasticity and NPI stage to ensure effective and feasible CAs are implemented that simultaneously improve KPIs, reduce machine downtime and eliminate scrap in a cost-efficient manner.

6.3.2 NPI-Coefficient Selection

The first step before training DDPG agent is engineering the reward function (see (6.5), (6.6) and (6.7)) to account for MAS correction requirements. This involves selecting the values for the user-defined NPI-coefficients $\delta = \{K, C, \lambda, s_p\}$ based on MAS KPIs, reconfigurability constraints and stochasticity that are dependent on the stage of NPI and production lifecycle [1]. From a MAS engineering perspective, different stages of the lifecycle entail different levels of reconfigurability and stochasticity. E.g. during design and pre-production stages, the costs of correcting process parameters (CAs) are much lower than when this is done in production stages [1]. The chapter proposes guidelines for selecting NPI-coefficient δ values to optimally trade-off the importance of the requirements, costs and benefits through the design and production stages of the NPI lifecycle.

Design Stage – During design, the MAS permitted reconfigurability is set to zero ($\lambda = 0$) as the design is done virtually within CAE simulation models resulting in near-zero restrictions on process parameter adjustments. This sets the $r_t^{CA} = 0$ and effectively changes the reward function to a weighted sum of improvement in KPIs multiplied by the RC uncertainty, thus, only focusing on the benefits and not the costs. The CA policies learnt with this reward aim to simultaneously maximise KPIs and minimise sensitivity to RCs (uncertainty in isolated RCs) within the allowable system reconfigurability (process parameter ranges) hence maximising product quality, process *robustness* and process *resilience*. Previous approaches [9] focused mainly on maximising KPIs under a predefined level of system stochasticity due to incoming part variations and tooling variations hence only maximising product quality and process robustness. Small values of $s^p \approx [3,10]$ are leveraged to ensure that all the factors that induce stochastic variations in the MAS are considered. Large allowable ranges

$[y_{min}^j, y_{max}^j]$ are selected for all process parameters to allow for effective exploration of the design space. The adjustment costs of reconfigurable process parameters (\mathbf{C}) are not required given the MAS permitted reconfigurability is set to zero. The agent is then trained over various stochastic sources such as part variations, tooling variations and system noise to learn a CA policy that yields optimal process parameter adjustments across all stochastic variations of the MAS.

Production Stage – To account for the substantial costs for adjusting process parameters during production stages [1], the MAS permitted reconfigurability can be fine-tuned within ranges $\lambda = [0.1, 1]$ to allow for changes in process parameters only when the impact in KPI improvement is significant. Relatively larger values of $s^p \approx [50,100]$ are leveraged to account for inherent MAS stochasticity. The relevant KPIs importance (\mathbf{K}) and adjustment costs of reconfigurable process parameters (\mathbf{C}) can be selected based on the MAS design architecture and constraints. In this scenario, CAs for only those process parameters that have a low-cost coefficient C^j would be estimated. Process parameters that cannot be changed during production have high values of C^j . The allowable ranges $[y_{min}^j, y_{max}^j]$ for adjustable process parameters is limited based on the permissible design tolerance, which is relatively smaller than the ranges during design given the process constraints. The NPI-coefficients (δ) can be further fine-tuned such that CA policies achieve a good trade-off between RC requirements (benefits) (r_{τ}^{RC}) and MAS requirements (costs) (r_{τ}^{CA}). Table 6.1 summarizes the guidelines to select values of the NPI-coefficients based on the NPI stage. After selecting NPI-coefficients CA policies bound by the MDP assumption are learnt using DDPG and the engineered reward function.

Table 6.1. NPI-Coefficient Selection

<i>NPI-Coefficient</i>	<i>NPI Stage</i>	
	<i>Design</i>	<i>Production</i>
Relevant KPIs importance (\mathbf{K})	High importance to quality	Importance to both cost and quality
Adjustment costs of reconfigurable process parameters (\mathbf{C})	No adjustment costs	High adjustment costs based on MAS constraints
MAS permitted reconfigurability (λ)	High reconfigurability: $\lambda=0$	Limited reconfigurability: $\lambda= [0.1,1]$
Maximum available data sample (s_p)	Small sample sizes: $[3,5] - [5, 10]$	Large sample sizes: $[50,100]$

	Prototype batches available at design stage between early and more mature design	As a rule of thumb production volume between breaks; often equals to 2 hours of production
--	--	--

6.3.3 Deep Deterministic Policy Gradient (DDPG)

DDPG is a model-free, off-policy actor-critic algorithm [169] that leverages neural networks to learn approximate CA policies π and CA-value (Q-value) functions Q^π in multi-dimensional real-valued CA spaces. The algorithm leverages the deterministic policy gradient (DPG) framework and approximates the CA policy function using a neural network known as the actor and CA value function using a neural network known as the critic. The actor estimates a CA while the critic generates the corresponding Q-value (expected reward) for the CA. Key insights from Deep Q-Networks (DQN) [37], such as samples from a replay buffer to train the network off-policy minimising correlations and using target networks, are leveraged to give consistent targets during temporal difference updates. The DPG algorithm leverages an actor (μ) parametrised by weights and biases of a neural network (θ^μ) which generates the current CA policy by a learning a deterministic mapping from the current state (RCs) to the CAs $\mu(\mathbf{s}_t^{MAS}|\theta^\mu)$. The critic network takes as input the CA and the state $Q(\mathbf{s}_t^{MAS}, \mathbf{CA}_t|\theta^Q)$ to generate the value of the current CA policy.

The CA action-value (Q-value) function is used to describe the expected return after taking an CA \mathbf{CA}_t in state \mathbf{s}_t^{MAS} and following the given CA policy π :

$$Q^\pi(\mathbf{s}_t^{MAS}, \mathbf{CA}_t) = E_{r_i \geq t, s_i \geq t \sim E, CA_i \geq t \sim \pi} [R_t | \mathbf{s}_t^{MAS}, \mathbf{CA}_t] \quad (6.15)$$

The Q-value can also be represented recursively using the Bellman's equation. Given a target deterministic policy Bellman's equation can be expressed as:

$$Q^\mu(\mathbf{s}_t^{MAS}, \mathbf{CA}_t) = E_{r_t, \mathbf{s}_{t+s_p}^{MAS} \sim E} [r(\mathbf{s}_t^{MAS}, \mathbf{CA}_t) + \gamma Q^\mu(\mathbf{s}_{t+s_p}^{MAS}, \mu(\mathbf{s}_{t+s_p}^{MAS}))] \quad (6.16)$$

Given Q^μ is only dependent on the stochasticity of the environment; it is learnt off-policy using transitions generated from a different stochastic behaviour policy β . The

loss function minimised to optimise the parameters θ^Q of the critic is expressed as:

$$L(\theta^Q) = E_{s_t \sim \beta, CA_t \sim \beta, r_t \sim E} \left[(Q(s_t^{MAS}, CA_t | \theta^Q) - p_t)^2 \right] \quad (6.17)$$

The term is known as temporal differencing (TD) error, where

$$p_t = r(s_t^{MAS}, CA_t) + \gamma Q(s_{t+s_p}^{MAS}, \mu(s_{t+s_p}^{MAS}) | \theta^Q) \quad (6.18)$$

The loss function (6.17) is minimized iteratively to train the critic. Simultaneously after updating the critic parameters, the actor parameters θ^μ are updated to obtain policy π by applying the policy gradient theorem to the training objective J [169] and obtaining the gradient for actor parameters θ^μ as:

$$\nabla_{\theta^\mu} J = E_{s_t \sim \beta} \left[\nabla_{\theta^\mu} Q(s_t^{MAS}, CA_t | \theta^Q) \Big|_{s=s_t^{MAS}, a=\mu(s_t^{MAS} | \theta^\mu)} \right] \quad (6.19)$$

The parameter update while training the actor and critic is done by gradient descent based models such as the Adam [185], which assume that observations are independently and identically distributed (IID). This is not the case when MAS state, CA and reward samples are sequentially generated and temporally correlated. Various techniques are leveraged to stabilise the learning process. (i) A replay buffer [37] is leveraged, which consists of a finite-sized cache and stores sequential transitions from the MAS environment. A transition is quantified as a tuple of the MAS state, CA, reward and resulting state $(s_t^{MAS}, CA_t, r_t, s_{t+s_p}^{MAS})$. While training, a random minibatch is uniformly sampled from the buffer. This minibatch consists of temporally decorrelated samples and hence aids to stabilize training. As the replay buffer is full, the oldest transitions are discarded in a First In First Out (FIFO) setting. (ii) Target critic and actor networks [37] are also used to account for the divergence issue caused as the critic network $Q(s, a | \theta^Q)$ being updated is also used in calculating the target value. The weights of the target networks are constrained to update slowly compared to the learned networks:

$$\theta' \leftarrow \tau\theta + (1 - \tau)\theta' \quad (6.20)$$

where $\tau \ll 1$. Although this slows the learning process in terms of the propagations of the value estimations, there is a significant increase in learning stability and a decrease in the sensitivity to actor-critic network architecture hyperparameters.

Learning in multi-dimensional real-valued action spaces also requires an effective exploration of different CA policies that can generate feasible and effective CAs under the aforementioned requirements. Given the off-policy nature of DDPG, an exploration policy μ' independent of the learning algorithm is chosen:

$$\mu'(\mathbf{s}_t^{MAS}) = \mu(\mathbf{s}_t^{MAS} | \theta_t^\mu) + \mathcal{N} \quad (6.21)$$

Where \mathcal{N} is generated using an Ornstein-Uhlenbeck process [223] that generates temporally correlated noise, the generated noise is added to corrective action estimated by the actor-network, which is then leveraged for transition and generation of the next state. This serves as an effective exploration strategy β for CA policies. The exploration ensures that the agent can search and test various CAs, observe the generated rewards that account for MAS benefits, costs, stochasticity and NPI stage and finally learn CA policies that generate CAs that are effective and feasible given the current state of the MAS.

6.3.4 DDPG Actor-Critic Network Optimization & Training

The performance of DDPG to generate feasible and effective CAs is heavily dependent on careful hyperparameter optimization of the actor and critic networks (Fig. 6.2). The chapter uses the original architecture as a baseline [169] and optimises it to enable CA estimation. The actor network μ that estimates the CA policy function is optimized to contain two dense hidden layers with 128 and 64 hidden nodes with ReLU activations. The output layer of the actor is modified to have neurons equal to the \mathbf{CA}_t dimension, i.e., the number of controllable/adjustable process parameters \mathbf{y} . The output layer leverages Tanh activation to bound the CA within the process parameter allowable limits.

The optimised critic network Q (action-value network) consists of two heads with 128 and 64 nodes that input the state \mathbf{s}_t^{MAS} and action \mathbf{CA}_t . These are concatenated and given as input to the next layer with 64 nodes with ReLU activation. The output layer consists of a linear activation and estimates the CA value for the MAS state and CA $Q^\pi(\mathbf{s}_t^{MAS}, \mathbf{CA}_t)$ (6.15).

Further optimizations are done to enable robust and stable learning. Batch normalization is leveraged as this scales all features to have similar ranges and minimises covariance shift in the deep networks as each layer within the network receives a whitened input [169]. A replay buffer of size 10^5 is leveraged with a minibatch size of 64. This converts sequential samples from consecutive states into near independent and identically distributed samples, reducing divergence for actor-critic network parameters during training. Initializations of final layer weights were done within small ranges $[-3 \times 10^{-3}, 3 \times 10^{-3}]$ to ensure that initial CAs and values of CAs were near zero to enable quicker learning, especially during initial training stages. The MAS state vector \mathbf{s}_t^{MAS} is scaled between $[-1,1]$, and the reward r_t is scaled between $[0,1]$. Such scaling is crucial for the network to train comparatively faster and reduce the risk of divergence. The Adam [185] with learning rate $\alpha = 0.001$ is used for training. The discount factor is set to $\gamma = 0.5$ (6.18) to give higher importance to current CAs (myopic agent). $\tau = 0.01$ (6.20) is used for soft target updates. The action exploration temporally correlated noise is generated using the Ornstein-Uhlenbeck process with $\theta = 0.15$ and $\sigma = 0.2$ (6.21). Each episode is run until termination. *Episodes are terminated when the difference in KPI compared to the nominal is below a pre-determined threshold.* A maximum of 500 training runs are done for each episode to account for the computer-aided engineering (CAE) simulation [6] time and computational budget constraints. Testing is done after training for 200 episodes at an interval of 20 episodes. VRM [6] is used as the CAE environment for training. The work has been implemented using Python 3.7 and TensorFlow - GPU 2.1. A python library [187] has been developed to validate and replicate the results of the methodology. Two Nvidia Tesla V100 32 GB GPUs are used for model training and deployment.

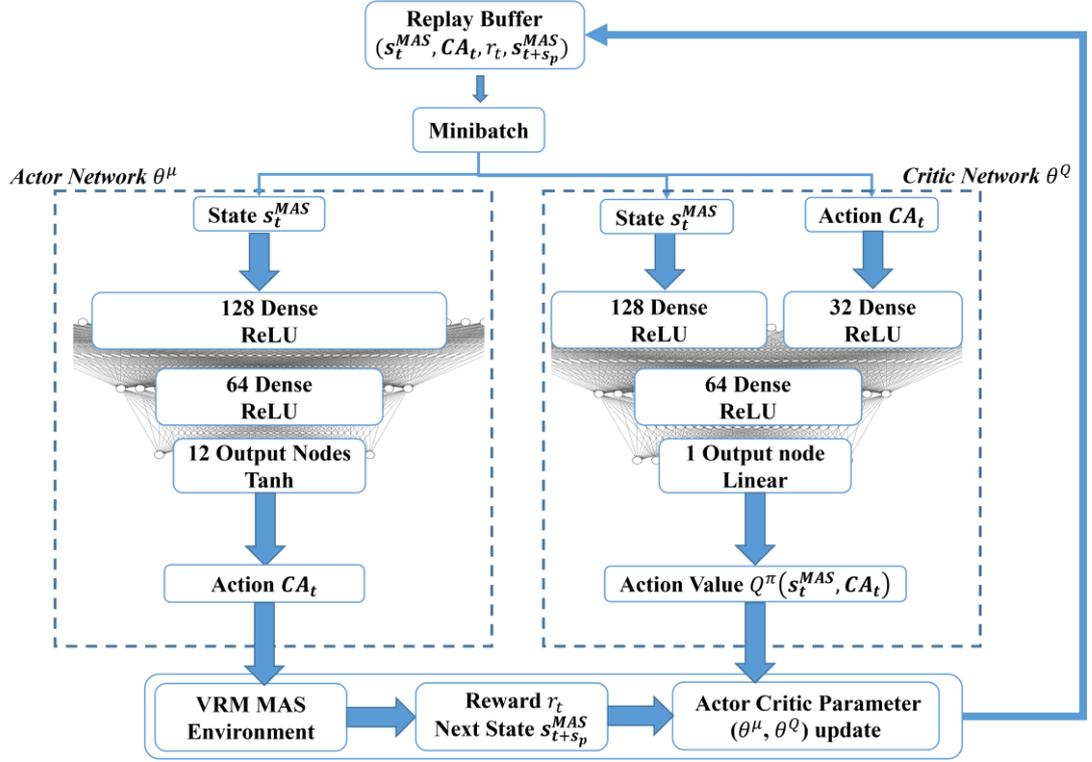


Fig. 6.2. DDPG agent

6.3.5 Application Framework: OSEC Deployment

After training, the actor of the DDPG agent is deployed within the MAS (Fig. 6.3). The overall steps within the deployment framework include:

- (i) *Product assembly*: the physical MAS assembles s_p products,
- (ii) *Collect point cloud data*: point cloud data for the products are collected at the end of the MAS using a 3D scanner and processed into object shape error data $\mathbf{x}^{s=N_s, \dot{s}=4}$ (6.1),
- (iii) *Isolate RC and estimate MAS state*: based on the object shape error data, the RCs are isolated and the MAS state \mathbf{s}_t^{MAS} is estimated using previously proposed OSER models (see (6.2), (6.3) and (6.4)),
- (iv) *Correct MAS using trained DDPG model*: the actor $\pi(\mathbf{s}_t^{MAS})$ (CA policy function) estimates feasible and effective CAs \mathbf{CA}_t (process parameter adjustment) (see (6.9) and (6.10)). The process parameter adjustment values are given as input to process controllers or maintenance is scheduled.

The CA policies generate new process parameter values known as the *functional nominal*, which can be interpreted as the value of process parameters that maximizes MAS KPIs given the current level of stochasticity while accounting for the constraints and costs. This may be different from the *design nominal* as this is learnt dynamically given the MAS's current state, while the design nominal is pre-determined based on limited modelling of all possible physical interactions. Current SPC/APC assume that the design nominal is permanently fixed and gives the best performance. CAs using these approaches only minimise the difference between current process parameters values and the design nominal. However, such CAs are frequently ineffective and infeasible to implement.

It should be noted that the agent *does not perform control actions*; instead, it behaves as an enabler by generating the process parameter adjustment (functional nominal) for the process controller/maintenance. This is also known as the process parameter setpoint and is generally assumed to be the design nominal in SPC/APC literature. Conventional controllers perform the task of going from the current process parameter to the new process parameter setpoint under process disturbances while minimising delay and overshoot. Using the controller also ensures an additional layer of safety and stability while correcting process parameters. Before deploying the agent in physical MAS, extensive testing should be carried out in the simulation environment (VRM) to ensure that CA policies have converged. During the initial stages, a human within-the correction loop can also be leveraged to ensure that the CA policy is safe and stable. The chapter leverages VRM as the simulation environment for both training and deployment. The overall framework for training and deployment of the DDPG based OSEC approach is summarized in Fig. 6.3.

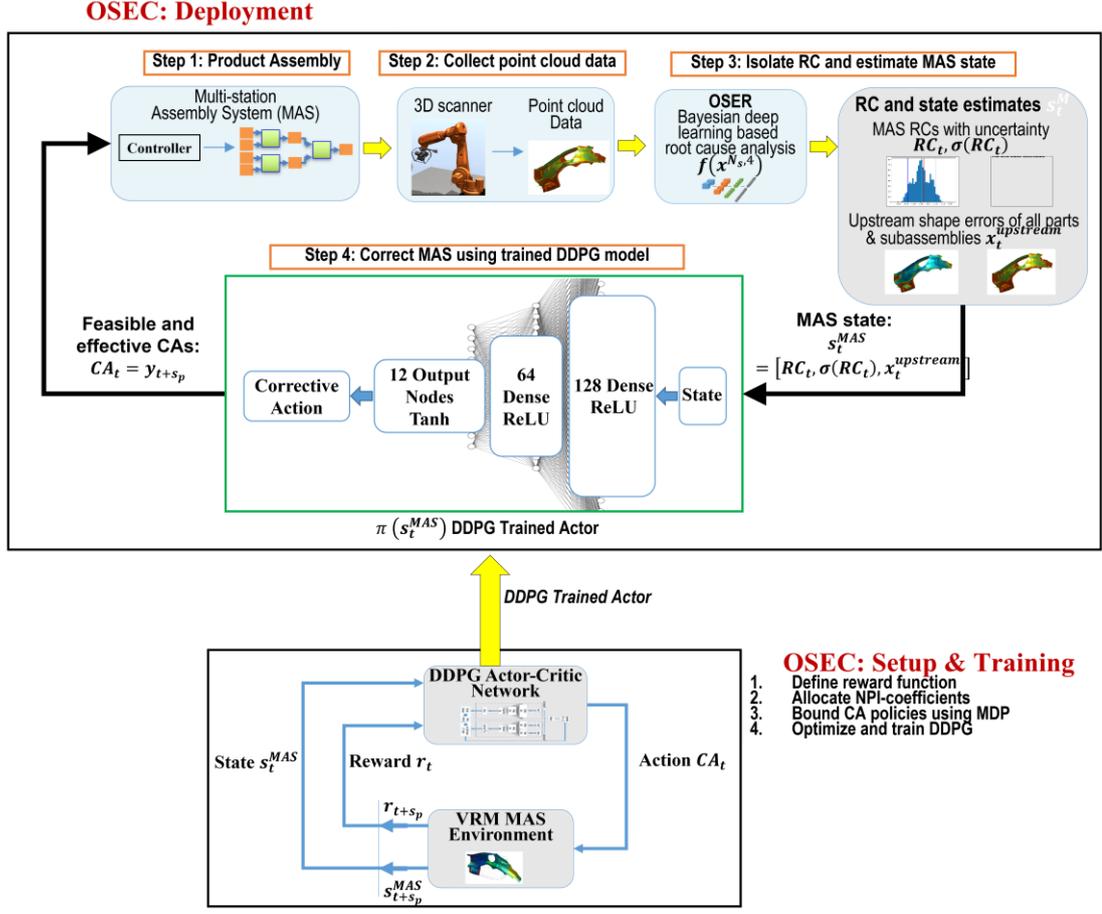


Fig. 6.3. OSEC training and deployment framework

6.4 Case Study

6.4.1 Experimental Setup

For verification and validation of the OSEC approach, an industrial, automotive cross member assembly is selected. It consists of $n = 4$ non-ideal compliant parts, namely, pocket, pocket reinforcement, cross member and cross member reinforcement (Fig. 6.4) and $N_s = 3$ stations, with each station consisting of four stages (Fig. 4.6). The assembly is controlled by $h = 12$ process parameters y^1, y^2, \dots, y^{12} (Table 6.2). The upstream shape errors are consolidated into a fixed-length vector $\mathbf{x}^{upstream}$ with dimensionality $d = 64 \times (N_s - 1) = 128$ (6.3). The overall state vector \mathbf{s}_t^{MAS} (6.4) consists of $\mathbf{x}^{upstream}$ and $\mathbf{RC}: \{y^1, y^2, \dots, y^{12}\}$ and has a dimensionality of $h + d = 140$. The CA space $\mathbf{CA}_t: \{y^5, y^6, \dots, y^{12}\}$ (6.10) has a dimensionality equal to the number of adjustable process parameters (8 process parameters). The CA policy learnt

for this case study will account for all the aforementioned requirements: (xi) *Uncertainty in isolated RC* as the state estimation OSER model will estimate uncertainties with the RCs (6.2); (xii) *KPIs improvement (benefits)* related to shape error of the final assembly; (xiii) *MAS system design architecture (costs)* as frequent changes are not permissible depending on the stage of the NPI cycle. Additionally, process parameters have different costs and range of allowed readjustments; (xiv) *MAS system inherent stochasticity* given the incoming part variations within non-ideal parts as induced by uncontrollable (*stochastic*) process parameters y^1, y^2, y^3, y^4 .

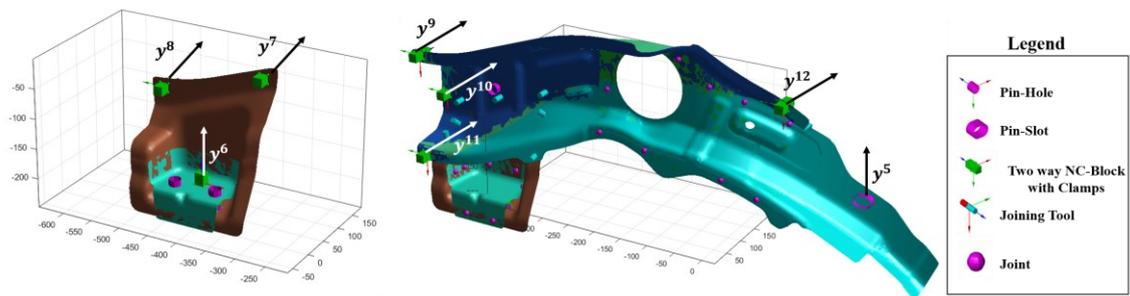


Fig. 6.4. Cross member assembly

Each training step consists of s^p samples being generated. Each episode is run until the maximum number of training steps are reached, or the terminal state is reached. Terminal state within each episode is defined when the mean absolute shape error of the final assembly across s^p samples are below the set threshold set as 0.05 mm, as variations more minor than this are not detectable by the 3D scanner. The goal within all scenarios is to learn CA policies that can adjust one or multiple process parameters sequentially and take the MAS from any state (i.e., state with one or multiple RCs) that has significant shape error to a terminal state (i.e., state with no RCs) where the shape error is below the threshold. An episode is terminated if the terminal state is not reached after 500 training steps, i.e., a total of $500 * s^p$ product samples. This termination criterion is introduced to limit the VRM CAE simulation time. As each computation can take up to five minutes for computation, allowing the training to run without a hard stopping criterion will require more time and computation resources.

Table 6.2. Process Parameters Description

Process Parameter	Description	Type
y^1, y^2, y^3, y^4	Part variation parameters	Uncontrollable (stochastic)
y^5	Pin-slot displacement in x	Controllable (action)
y^6	Clamp 1 displacement in y	Controllable (action)
y^7	Clamp 2 displacement in y	Controllable (action)
y^8	Clamp 3 displacement in z	Controllable (action)
y^9	Clamp 4 displacement in y	Controllable (action)
y^{10}	Clamp 5 displacement in y	Controllable (action)
y^{11}	Clamp 6 displacement in y	Controllable (action)
y^{12}	Clamp 7 displacement in y	Controllable (action)

6.4.2 Application Scenarios

As described in Section 6.3.2, different values of the NPI-coefficients are leveraged to engineer the reward function of the proposed OSEC DDPG approach for the correction of MAS. The chapter applies the framework for three scenarios:

(i) *Scenario 1 (Design Stage)* – To leverage OSEC during design stages, the MAS permitted reconfigurability $\lambda = 0$ is set to zero as the design is done within CAE simulations. The absolute value of shape errors after the final station is considered as the KPIs. A maximum sample size of $s^p = 10$ is leveraged. The reward function for this scenario can be written as:

$$r_t = -(1 - \sigma^{RC_t}) \left(\frac{1}{s^p * k^n} \sum_{p=1}^{s^p} \sum_{i=1}^{k^n} (|\mathbf{x}_{i,p}^{3,4}|)^2 \right) \quad (6.22)$$

The relevant KPI importance for shape error across all points is assumed constant and equal to 1. k^n consists of all points across the assembly, $k^n = n_1 + n_2 + n_3 + n_4 = 10841$. The process parameter ranges for CAs are set as [-4 mm, 4 mm] for all process parameters for efficient exploration of design space.

(ii) *Scenario 2 (production stage and medium reconfigurability)* – To leverage OSEC during production stages, the MAS permitted reconfigurability of $\lambda = 0.5$ and maximum sample size of $s^p = 30$ is leveraged. The reward function is:

$$r_t = -(1 - \sigma^{RC_t}) \left(\frac{1}{s^p * k_n} \sum_{p=1}^{s^p} \sum_{i=1}^{k_n} (|\mathbf{x}_{i,p}^{3,4}|) \right)^2 + \lambda \frac{1}{h} \sum_{j=1}^h C^j (y_{t+s^p}^j - RC_t^j)^2 \quad (6.23)$$

The process parameter ranges for CAs is set as [-1 mm, 1 mm] for all process parameters as the flexibility to change process parameters is limited during production stages. The adjustment costs for y^5 is set to $C^1 = 0.7$ while all others it is set to $C^j = 0.9$ $j = \{2,3 \dots,8\}$ as costs related to changing clamps are higher than adjustments for pin-slot. All other NPI-coefficients are the same as for Scenario 1.

(iii) *Scenario 3 (production stage and low reconfigurability)* – Further to leverage OSEC during production stages for MAS that cannot be reconfigured significantly, the MAS permitted reconfigurability is set to $\lambda = 1$ so that process parameters are only adjusted only when the effect on KPI is significant. A maximum sample size of $s^p = 40$ is leveraged to ensure changes are only done after observing a significantly large number of products. The process parameter ranges for CAs are set as [-0.5 mm, 0.5 mm] for all process parameters as these are permissible tolerance limits during production stages. The reward function and remaining NPI-coefficients are the same as for Scenario 2.

6.4.3 Results

Ten training trials are done for the actor-critic DDPG network, each with 500 episodes. To reduce sensitivity to initial parameters, all trails use the same seed for initialization. Testing is done for 200 episodes. The exploration policy is not used while testing. The agent can successfully learn optimal CA policies for all scenarios. The overall average metrics of the ten trials for all three scenarios are summarised in Table 6.3 and Fig. 6.5 summarises the normalised averaged reward for all ten trials while training in all three scenarios. Fig. 6.6 summarises the average training steps per episode while training for all ten trials. The failed episodes quantify the percentage of test episodes that did not go below the required threshold after 500 steps.

Table 6.3. OSEC Results

	Scenario 1		Scenario 2		Scenario 3	
NPI stage	<i>Design</i>		<i>Production</i>			
Reconfigurability	<i>High reconfigurability</i> $\lambda = 0$		<i>Medium reconfigurability</i> $\lambda = 0.5$		<i>Low reconfigurability</i> $\lambda = 1$	
	<i>Mean</i>	<i>SD</i>	<i>Mean</i>	<i>SD</i>	<i>Mean</i>	<i>SD</i>
Total training time	87.6 hrs	5 hrs	255.3 hrs	8 hrs	369.14 hrs	16 hrs
Average training steps per episode	57	8	61	12	74	20
% Failed episodes	2.1%	0.3%	4.3%	0.6%	5.1%	1.1%
Normalized test reward	0.97	0.05	0.79	0.08	0.68	0.12
Test mean absolute shape error	0.2 mm	0.05 mm	0.17 mm	0.04 mm	0.11 mm	0.03 mm

The results demonstrate how CA policies differ across different stages of the NPI lifecycle. CA policies during design (Scenario 1) converge faster with lesser number of CAs per episode as compared to CA policy during production (Scenario 2). This is because the MAS requirements are significant during production stages resulting in the requirement of higher number of CAs to reduce shape error below the required threshold. Further comparison between Scenario 2 and 3 demonstrates the difference in CA policies under different MAS permitted reconfigurability. The final reward for Scenario 3 is lower given the high penalty for MAS related requirements as the MAS permitted reconfigurability is minimum in Scenario 3. This also results in Scenario 3 having the highest failure rate of 5.1% due to the high MAS penalty (low reconfigurability) which prohibits the agent from frequently changing process parameters even during situations when the shape error is above the minimum threshold.

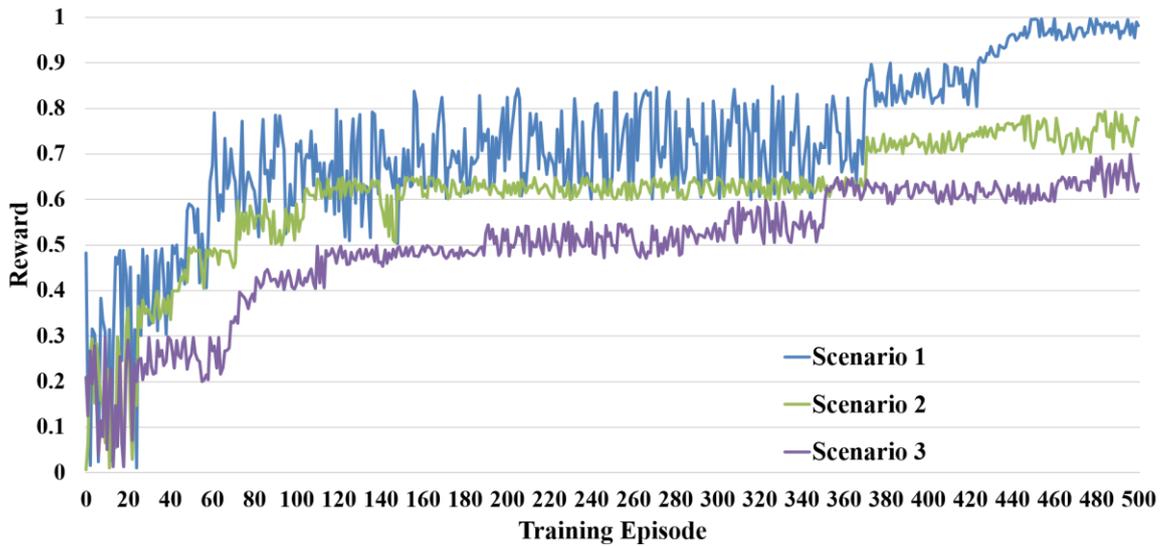


Fig. 6.5. Training reward per episode

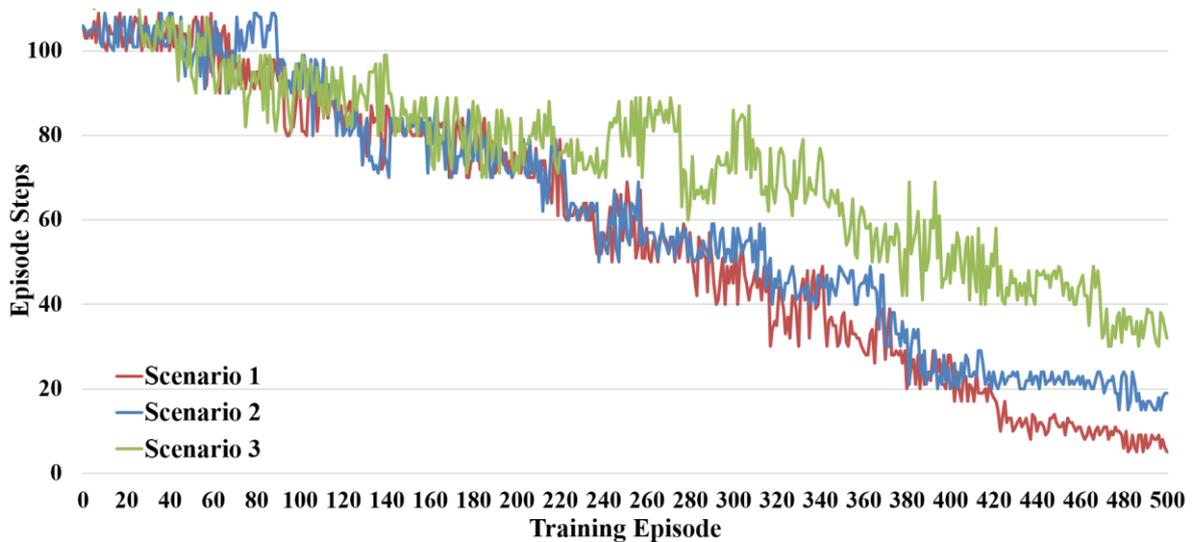


Fig. 6.6. Training steps per episode

The estimated CA policies provide the required CAs (process parameter adjustments) to significantly reduce shape errors. After training, the process parameter adjustments for CA policies are analysed across all three scenarios. The policies learnt in Scenario 1 converge to terminal states much faster due to zero MAS requirement penalty, while Scenarios 2 and 3 take more steps to converge to the terminal states given CAs have to be sequentially generated while ensuring that the MAS requirements costs do not dominate the KPI improvement and RC elimination benefits.

Figs. 6.7, 6.8 and 6.9 summarises the behaviour of agent when incoming stochasticity i.e. y^1, y^2, y^3, y^4 follow a normal distribution with $\mu = 0 \text{ mm}, \sigma = 0.1 \text{ mm}$. CA taken by the agents ensure that all process parameters y^5, \dots, y^{12} can converge at the functional nominal where the shape error is minimum, although with different number of steps and within different ranges. The object shape error outputs after different steps for the policies learnt for all scenarios are shown in Fig. 6.10, Fig. 6.11 and Fig. 6.12.

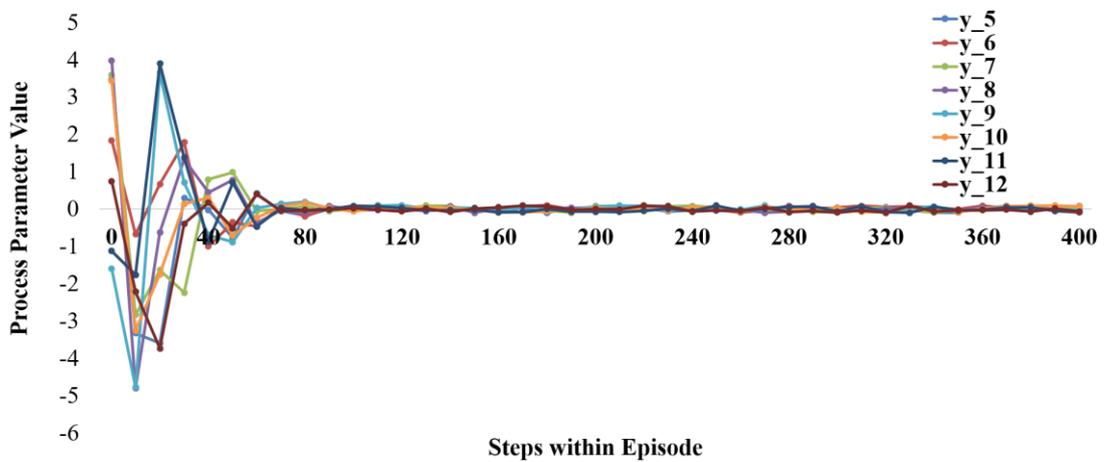


Fig. 6.7. Scenario 1 policies

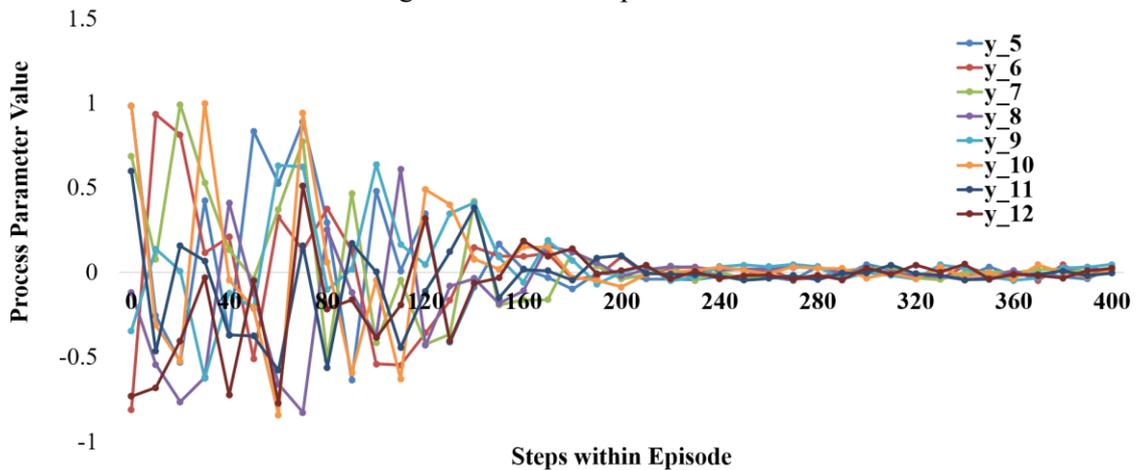


Fig. 6.8. Scenario 2 policies

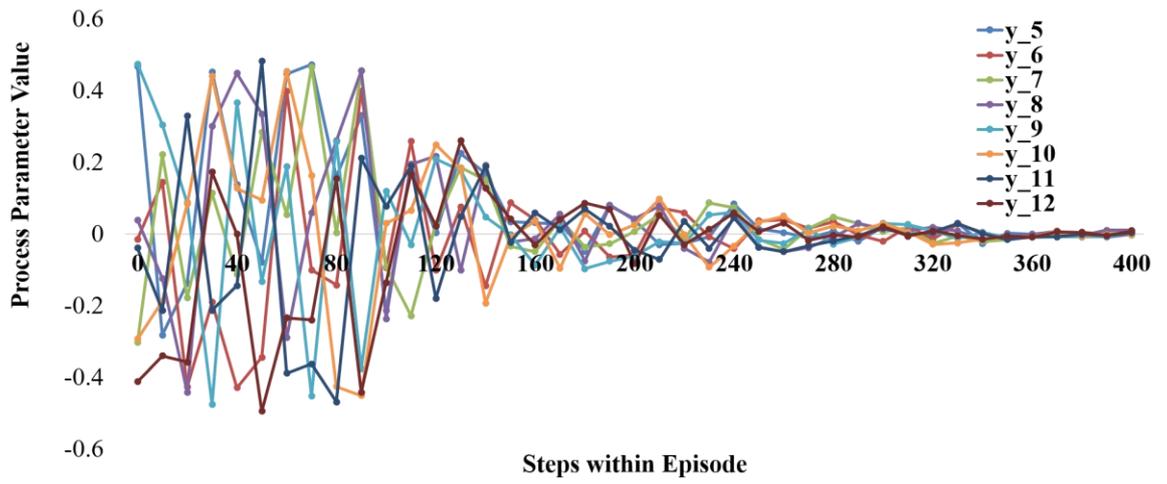


Fig. 6.9. Scenario 3 policies

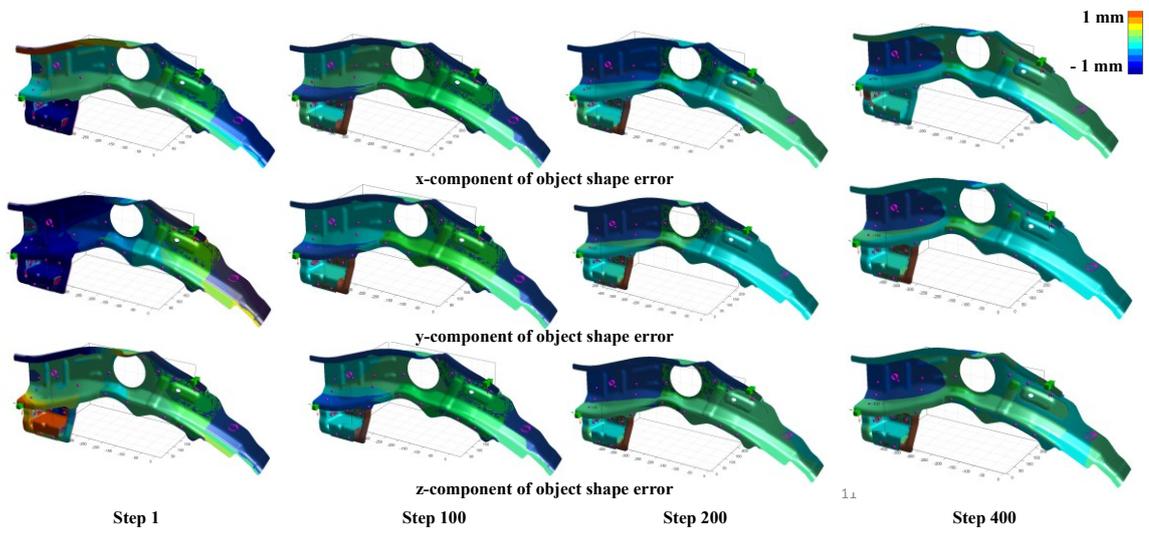


Fig. 6.10. Object shape error outputs for Scenario 1 policies

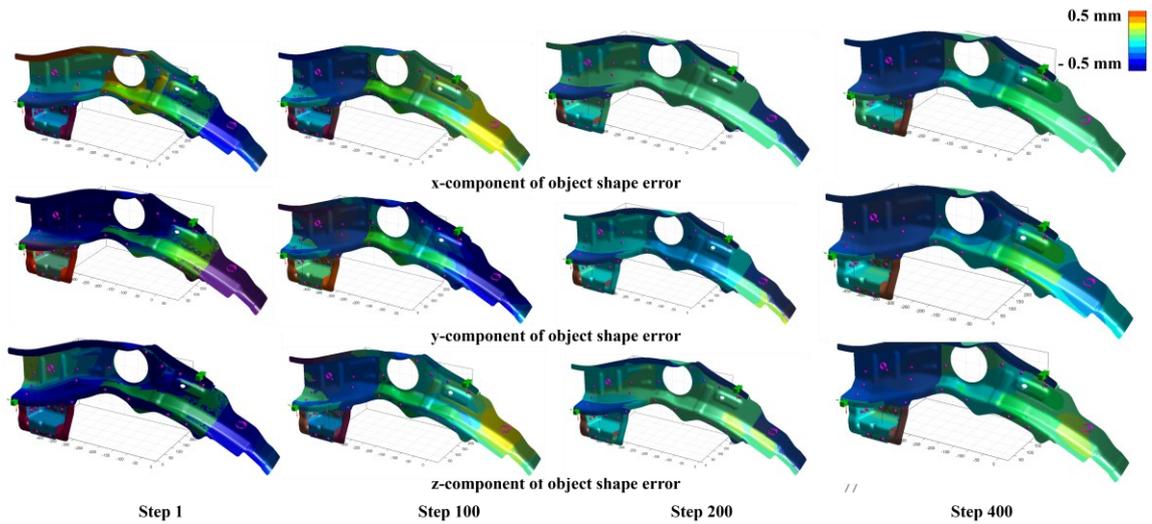


Fig. 6.11. Object shape error outputs for Scenario 2 policies

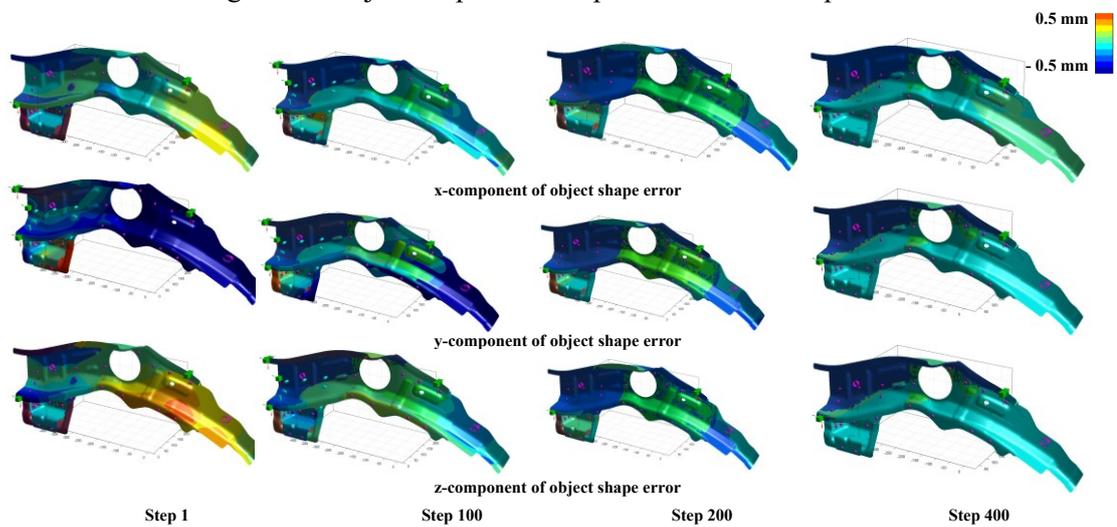


Fig. 6.12. Object shape error outputs for Scenario 3 policies

6.4.4 Benchmarking

Scenario 3 is selected as benchmarking scenario, given that the NPI-coefficients for the reward function selected for this scenario ensure all requirements are taken into account. This scenario consists of cases in which $RC \neq CAs$ given less MAS permitted reconfigurability ($\lambda = 1$) which translated into CAs not being feasible to implement. Additionally, varying levels of incoming part variations are considered which result in CAs that are generated using a fixed design nominal being ineffective. The benchmarking of the OSEC CA policy is done against three CA policies, namely:

(i) *Mean RC estimates π^0* – in this, the process parameters are changed based on the

estimated mean value of the process parameter \mathbf{y} (RC) by the OSER model (6.2). The difference between the design nominal and mean RC is estimated as the CA. This is case for first group of approaches that map data to RCs and to CAs based on manual expertise ($CA = -RC$)

(ii) *SPC/APC based approaches π^1 (DDPG with basic KPI improvement reward)* – in this, DDPG with the same architecture as proposed in the chapter is leveraged, but the reward function is limited to KPI improvement $r_t = -(\frac{1}{s^p * k^n} \sum_{p=1}^{s^p} \sum_{i=1}^{k^n} (|\mathbf{x}_{i,p}^{3,4}|)^2)$ (i.e., requirement (ii)). Other requirements such as RC uncertainty, MAS constraints and stochasticity are not considered. This case is the *upper bound* for second and third group of approaches that directly map data to CAs ($CA = \Delta y$) such as SPC/APC assuming a fixed design nominal and do not consider the NPI stage.

(iii) *OSEC π^{OSEC} (DDPG with reward engineering)* – in this, the proposed DDPG agent and the engineered reward function consisting of the NPI-coefficients (6.7) accounting for all requirements is leveraged to learn the policy. The difference between the functional nominal as estimated by the CA policy and the current value of the process parameter (RC) is estimated as the CA.

For the testing of all three CA policies, 300 episodes are run until the terminal criteria of shape error (KPI) being below the threshold or the maximum number of steps (computational budget) is reached. For each episode, the stochastic non-ideal part variation parameters y^1, y^2, y^3, y^4 follow a normal distribution with a randomly sampled mean between [-1 mm, 1 mm] for each episode while the standard deviation is always fixed at 0.1 mm. The initial values for all process parameters are randomly sampled between the allowable ranges [-0.5 mm, 0.5 mm]. Comparison is made for key benchmarking metrics, namely: (i) mean absolute object shape error (quality KPI) across all episodes, this quantifies the benefits of the CA policies; (ii) mean MAS requirement penalty across all episodes, this quantifies the costs (in terms of process parameter adjustments) of the CA policies to get the KPI below the required threshold; (iii) mean steps per episode this quantifies the converge as the number of sequential adjustments (CA) to get the KPI below the required threshold; and, (iv) percentage of

failed episodes that did not converge, i.e. shape error was still beyond the threshold after the maximum number of steps. The results are summarised in Table 6.4.

The OSEC policies perform better across all benchmarking metrics. The overall mean absolute shape error (quality KPI) is limited to 0.14 mm as compared to 1.2 mm for mean RC estimates change policy (first group of approaches) and 0.35 mm for SPC/APC based approaches (second and third group of approaches). The cost in terms of MAS requirement penalty is also limited to 0.25 mm for the OSEC CA policy as compared to 0.64 mm for SPC/APC approaches, highlighting the ability of the policy to estimate functional nominals that maximize KPI performance under the current level of stochasticity and are within the MAS reconfigurability constraints. The mean steps per episode is lower for SPC/APC CA policy as compared to OSEC CA policy given it assumes a fixed design nominal, but it has a high failure rate of 21.5%, which is caused by CAs that are ineffective when applied given the varying levels of stochasticity. The OSEC CA policy estimates a functional nominal under the given level of stochasticity hence resulting in a failure rate of only 6.2%. This ensures that the effective convergence of the CA policy under varying levels of stochasticity is better than other groups of approaches. Overall, the OSEC policies give a higher and robust performance by giving higher quality KPI improvement (shape error reduction) and lower MAS penalty costs while converging faster under varying levels of stochasticity.

Table 6.4. Benchmarking

Benchmarking Metric	Mean absolute shape error (Quality KPI: Object Shape Error)		Mean MAS penalty (Costs)		Mean steps per episode (Convergence)	% Failed episodes
	Mean (mm)	SD (mm)	Mean (mm)	SD (mm)		
Policy						
Mean RC estimates (π^0) (i) no DDPG (ii) CA = -RC	1.2	0.47	0.79	0.33	380	64%
SPC/APC based approaches (π^1) (i) DDPG agent with basic KPI improvement reward (ii) CA = Δy	0.35	0.11	0.64	0.24	230	21.5%
OSEC (π^{OSEC}) (i) DDPG agent with engineered reward (ii) CA = $\pi(s_t^{MAS})$	0.14	0.06	0.25	0.12	270	6.2%

6.5 Conclusions

The chapter presented a deep reinforcement learning-based methodology, Object Shape Error Correction (OSEC), for multi-station assembly systems, to mitigate root cause(s) of dimensional and geometric product shape errors. It leverages Deep Deterministic Policy Gradient and formulates a reward function parametrized by NPI-coefficients that enable mitigation of shape errors throughout the NPI and production lifecycle. It enables estimating real-valued corrective actions while accounting for root cause estimation uncertainty, KPI improvement, MAS design architecture and inherent stochasticity. Results and benchmarking demonstrated better performance in terms of quality and cost KPIs and quicker convergence under stochasticity when compared to current approaches for CAs. The OSEC is able to learn CA policies that adjust process parameters at a required frequency to optimally balance the benefits and cost priorities depending on the NPI stage. In cases of low reconfigurability, the CA policies need a greater number of steps to converge to ensure that reconfigurability constraints are not violated. The ability of CA policies to generate functional nominal ensures that when the stochasticity varies or the RC uncertainty is high, the corresponding effective CAs are generated. The CA approach is crucial for the enhancement of productivity by reducing machine downtime and scrap. The approach also considers six-sigma quality and process indicators ensuring that the CAs improve quality KPIs and minimize the costs incurred due to CA implementation.

7 Advanced Implementations and Industrial Applications

Previous chapters have comprehensively discussed each proposed kernel's formulation, methodology, verification, validation, and contributions. This chapter aims to discuss various implementations and applications of the kernels. The contributions of the chapter are as follows:

- (1) Develop a physical Industrial In-Process Quality Improvement (IPQI) demonstrator that provides validation for the proposed kernels *using real quality point cloud data collected from a 3D scanner*.
- (2) Develop a software implementation of the kernels as an open-source, fully documented python library termed *Bayesian Deep Learning for Manufacturing* (DLMFG) which aims to provide the required technical platform including datasets and code to enable reproducibility and further enable the advancement of the proposed research.
- (3) Discuss the kernel applications in *industrial manufacturing environments* and elaborate on the value created within these environments.

The rest of the chapter is organised as follows: Section 7.1 discusses the physical implementation of the work involving the formulation and development of an industrial demonstrator; Section 7.2 discusses the software implementation involving the python library; and finally, industrial applications are summarized in Section 7.3.

7.1 Physical Implementation

7.1.1 Industrial Demonstrator

As a crucial part of this research, an industrial demonstrator of the work titled '*Inline Quality Monitoring with Root-Cause Diagnosis* [224] was formulated and implemented. This demonstrator's objectives were to showcase inline quality monitoring with root cause diagnosis for a sheet metal assembly with non-ideal parts.

The incorporated digital methodologies use multi-physics simulation combined with data analytics, statistical modelling, optimisation and machine learning to minimise inspection cycle-time while maintaining fault diagnosis capabilities. This capability is crucial for systematic quality improvement within manufacturing systems in the automotive or aerospace industry. The demonstrator was also leveraged to verify and validate the proposed OSER and CLIP kernels using a physical assembly setup to generate object shape errors patterns similar to ones generated in MAS. Point cloud data collection is done using 3D scanners.

7.1.2 Experimental Setup

For verification and validation of the proposed OSER and CLIP kernels, an automotive non-ideal compliant part, the door halo was selected. The experimental setup and parameters are shown in Fig. 7.1. The assembly process is controlled by the four ($h = 4$) parametrized process parameters y^1, y^2, \dots, y^4 (depicted using yellow symbols in Fig. 7.1). These parameterise the corresponding clamp movement in the y-direction within the range of [-5 mm, 5 mm]. Parameters such as pin-hole and pin-slot for the door halo are considered constant (depicted using green symbols in Fig. 7.1) and are not parameterized. A physical implementation (Fig. 6.2) and a CAE implementation using VRM of the case study are done. Point cloud data in the physical implementation is collected using the 3D scanner (Fig. 7.3). Hexagon WLS400 mounted on an ABB robot was used as the data collection setup. CoreView AM and CoreView Teach were leveraged as the software setup enabling robot and scanner communication. The point cloud is characterized by $n = 1047$ key surface points which correspond to the mesh nodes for simulation in VRM. Both are pre-processed and voxelized to $(u, v, w) = (64, 64, 64)$ voxel grids for model training and testing. *The demonstrator was used to generate object shape error patterns similar to those generated within assembly systems.*

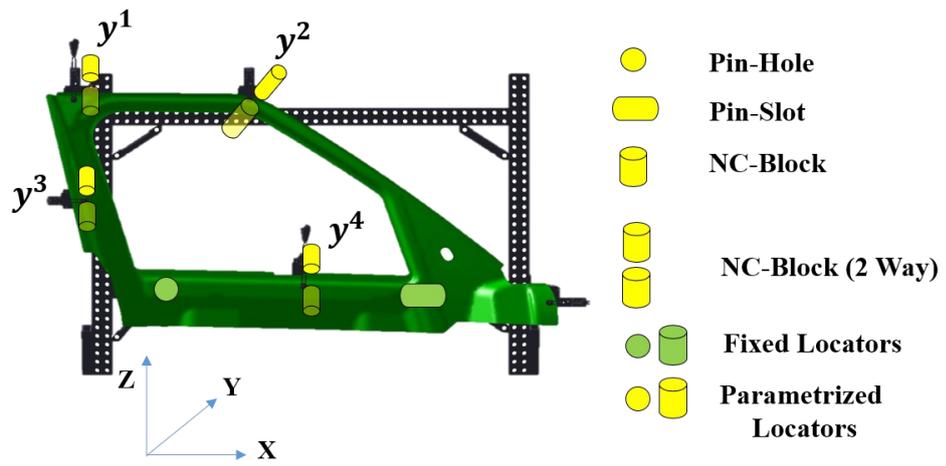


Fig. 7.1. CAE implementation of the demonstrator case study

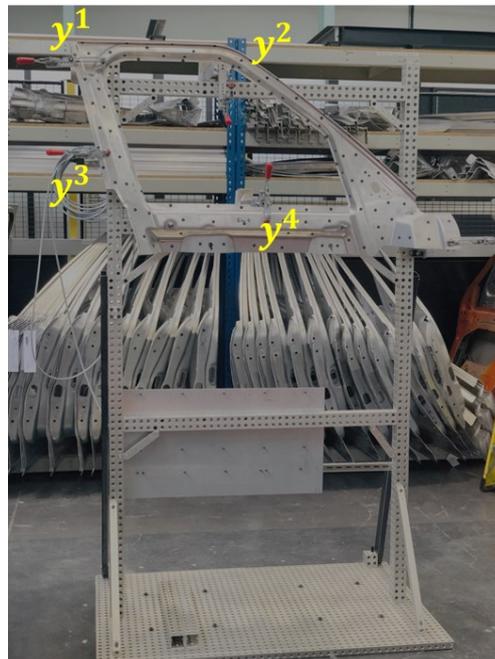


Fig. 7.2. Physical implementation of the demonstrator case study



Fig. 7.3. 3D Scanner for point cloud data collection

7.1.3 VRM Validation

VRM [6] was leveraged as the multi-physics CAE simulator. Before generating datasets for training and testing, the VRM model was calibrated to give maximum correlation between shape error data collected from the scanner and generated using VRM simulation. Grid search was leveraged to calibrate crucial parameters (soft spring constant and stiffness penalty) within VRM to ensure it generates data as close as possible to data collected from the 3D scanner. Nine cases for different given values of process parameters were selected for calibration (Table 7.1). Correlation for shape error for all 1047 points measured by the scanner and as simulated by VRM was used to find the best-calibrated model. The highest value of an average correlation of 0.93 across all nine test cases was observed after grid searching over crucial VRM parameters. This calibrated simulation model with the optimized VRM parameters was further used for training and testing OSER and CLIP kernels.

Table 7.1. VRM Calibration Results

y^1	y^2	y^3	y^4	Correlation
0	0	0	0	1
-5	0	0	0	0.89
5	0	0	0	0.89
0	-5	0	0	0.92
0	5	0	0	0.88
0	0	-5	0	0.97
0	0	5	0	0.98
0	0	0	-5	0.97
0	0	0	5	0.99

7.1.4 Results

The calibrated VRM model was used to generate 10,000 training samples, and 130 samples were collected using the 3D Scanner. This set of 130 samples were split into training (90) and test (30) sets. This combined dataset was used to train and test various models in four scenarios, namely:

(i) *Point-based multiple regression model* – This entails current state-of-the-art practices for assembly diagnosis [188]. A multiple regression model for each y is formulated. The independent variables include four critical points within the point cloud. The closest point corresponding to each clamp is selected as the critical points, given that they contain maximum information about the clamp movement. The coefficients are estimated using maximum likelihood estimation based on the 90 scanner training set samples. The trained model is then tested on 30 scanner test set samples.

(ii) *OSER trained only on CAE simulation data* – The Bayesian 3D CNN model is trained on 10,000 simulation samples and then tested on 30 scanner test samples

(iii) *OSER trained on simulated data and fine-tuned on scanner data using CLIP transfer learning framework* – The Bayesian 3D CNN model is trained on 10,000 simulation samples and then fine-tuned using the CLIP framework with a transfer learning algorithm. Ninety scanner train samples are used for fine-tuning. Testing is done on thirty scanner test samples.

(iv) *OSER trained on simulated data and find tuned on scanner data using CLIP continual learning framework* – The Bayesian 3D CNN model is trained on 10,000 simulation samples and then fine-tuned using the CLIP framework with continual learning algorithm. Ninety scanner train samples are used for fine-tuning. Testing is done on thirty scanner test samples.

The results for all scenarios are summarized in Fig. 7.4. OSER performed better than state-of-the-art linear/statistical models by 40%, with an $R^2 = 0.60$ as compared to $R^2 = 0.22$. Further fine-tuning the OSER model using CLIP based algorithm portfolio of continual and transfer learning further improved performance by 15% with an $R^2 = 0.75$, hence validating the integration of CAE simulated data and scanner data (physical data collected from assembly system) to train OSER framework. This demonstrator is currently showcased within the In-Process Quality Improvement (IPQI) cell within WMG, University of Warwick and is used to demonstrate the capabilities of integrating 3D scanners, deep learning and CAE simulation for applications such as RCA of MAS. The post of the demonstrator is shown in Appendix G.

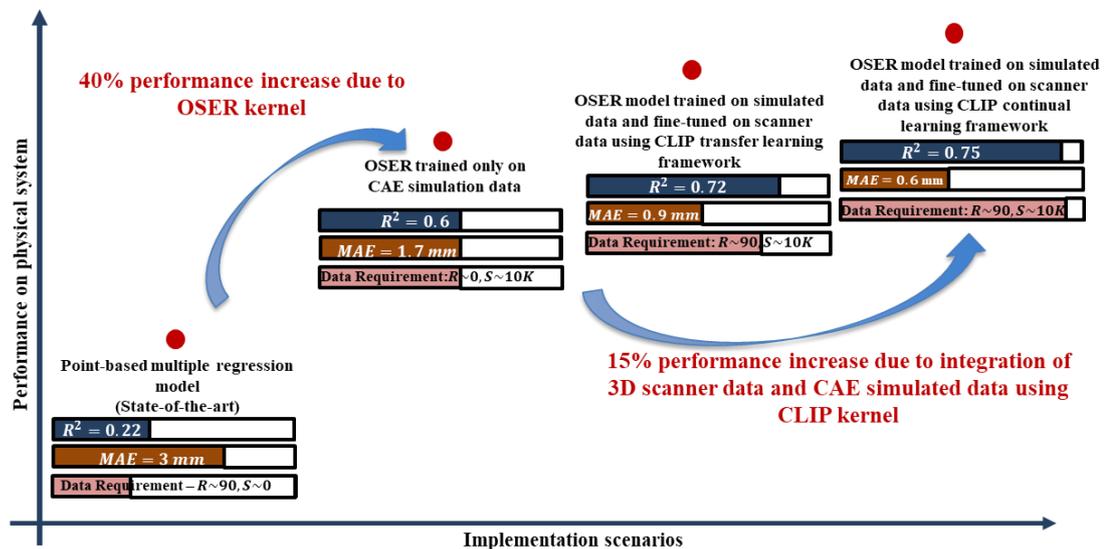


Fig. 7.4. Demonstrator results

7.2 Software Implementation

The importance of open-source code and datasets to enable reproducibility of any model is crucial to the research field's overall advancement. The impact can be seen in

research areas such as deep learning and artificial intelligence, where progress and advancement are exponentially faster than in other traditional research areas. This is owed to various libraries such as TensorFlow (Google) and PyTorch (Facebook) that are entirely open-sourced, and the availability of common benchmarking datasets to compare and validate methods. Additionally, the majority of the research groups and companies within these fields open-source their code and results. Similar open datasets and implementations are required given the increasing trend of AI applications in manufacturing [213].

The research develops and implements all the proposed kernels as an open-source Python and Matlab integrated library titled Bayesian Deep Learning for Manufacturing [187] that is hosted on Github. The library has been documented using Sphinx [225] and is hosted on Github pages. The VRM based OSEM kernel is implemented using a combination of Matlab and C++ integrated using OpenMP. The OSER, OSER-MAS, CLIP and OSEC kernels are implemented in python using a TensorFlow (TF) and Tensorflow-Probability (TFP) backend. The Matlab and Python kernels are integrated using the Matlab-Engine package and enable low latency data and variable transfer. The datasets are stored in a Postgres SQL database. The library also consists of Jupyter notebooks that perform exploratory data analysis. The datasets used for training, validation and testing are hosted publically on Zenodo [226]. Any new case study can be implemented by defining a set of configuration files. The library is programmed in an object-oriented manner to enable the easy addition of new features. Various novel features that are not a part of current state-of-the-art deep learning libraries are implemented from a software perspective that can be leveraged for other applications. They include:

(i) *Multi-task attention based Bayesian 3D U-Nets* – such networks can be leveraged for tasks requiring analysis of 3D data such as mesh, point clouds, depth images across domains such as manufacturing, healthcare and robotics.

(ii) *Uncertainty based closed-loop sampling* – can be leveraged to sample from high-fidelity physics simulations that are inherently computationally expensive. Using data sampled using such an algorithm potentially gives quicker convergence when training

machine or deep learning models than uniform sampling and LHS sampling.

(iii) *Uncertainty guided continual learning* – enables sequential decision making when learning needs to be done continually for systems that are constantly changing and evolving.

(iv) *Greedy sensor placement* – as the scale of the systems increases and the capabilities to do backward synthesis using data collected only at the end is not possible; hence greedy algorithms provide the following best location within the MAS to collect data by placing a sensor to increase the capabilities of backward synthesis.

(v) *3D mesh interpolated gradient-based class activation maps* – such implementations can be leveraged to obtain interpretability insights such as feature maps and gradient activations for any layer within the network, providing first-hand interpretability and implicit attention insights crucial for the application of black-box systems within industrial environments.

(vi) *CAE assembly environments* – aims to provide environments to train and validate various reinforcement learning algorithms on tasks that are inherently more complex than traditionally used platforms [37]. The agents in this setting should output multi-dimensional real-valued actions. Scaling such agents is currently one of the most researched areas in deep reinforcement learning [227], [228].

An overview of the software implementation on GitHub is given in Appendix B.

7.3 Industrial Applications

The developed and validated kernels provide capabilities for various industry applications listed below:

(1) *RC isolation within the scope of object shape error defects* – The OSER and OSER-MAS kernels extend traditional object detection to object shape error estimation and provide capabilities to estimate process parameters (RCs) that cause object shape errors in products. This essentially provides the capability to do RCA of shape

error defects for manufacturing applications such as assembly, stamping, machining and additive manufacturing. This capability leads to improvement in quality due to early detection and elimination of shape error sources. This also increases productivity due to a reduction in scrap and machine downtime.

- (2) *CA estimation within the scope of object shape error defects* – The OSEC kernel provides capabilities to mitigate RCs of object shape errors by generating effective and feasible CAs that can be implemented in the MAS while considering multiple objectives such as RC elimination, cost/quality KPI improvement, MAS reconfigurability constraints and stochasticity while considering the NPI stage. These CAs can then be mapped to CAs, maintenance schedules or standard process control adjustments without the need for manual expertise. It also enables the decoupling of complex correction problems into two sub-problems, namely (i) determining the optimal CA and (ii) implementing the optimal CA within the MAS.
- (3) *Explainable and interpretable decision making with confidence estimates* – The uncertainty estimates from the OSER kernel and interpretability insights from the CLIP kernel enable explainability and accountability while making costly decisions within manufacturing environments. The lack of interpretability has been one of the major roadblocks in adopting deep learning systems within industrial environments. These insights aim to integrate the much needed ‘trust’ factor within black-box deep learning models needed for actual deployment of such solutions, thus, providing a transparent alternative to manual expertise-based decision making.
- (4) *Scalable end-to-end transformative framework* – Overall, the four kernels aim to provide a broader impact by providing a transformative framework for integrating 3D scanners, deep learning and CAE simulations within industrial manufacturing environments so that milestones such as right first time, zero-defect manufacturing and resilient manufacturing can be realised.

8 Conclusions and Future Work

Previous chapters have comprehensively discussed the development of the proposed methodologies, their software implementation as kernels and demonstration using industrial case studies. The kernels were developed to help the manufacturing industry consistently deliver high-quality products while ensuring high productivity and cost-efficiency in the face of increasing product variety and smaller batch sizes by addressing key challenges for isolation and mitigation of dimensional quality defects. The kernels aimed to bridge gaps such as the inability of current statistical and engineering-based models to leverage quality monitoring data such as point clouds to isolate RCs, lack of uncertainty quantification for isolated RCs, lack of integration between data-driven and physics-based engineering models, inability of current models to enable scaling across different assembly systems, lack of transparency in the isolated RCs and extensive reliance on manual expertise to design CAs to mitigate isolated RCs. This chapter summarizes the overall findings of the research work. The rest of the chapter is organised as follows: Section 8.1 discusses the conclusions and the gaps addressed of the presented research; Section 8.2 discusses the knowledge contributions of the developed kernels; Section 8.3 discusses various limitations of the work; and, finally, the scope for future research is summarized in Section 8.4.

8.1 Conclusions and Gaps Addressed

Manufacturing has been driven forward by understanding the physics behind the process to analyse a given inputs' behaviour and estimate the outputs during process design and was supported by statistical analysis and operations research to understand the stochasticity of outputs such as quality and productivity during production. On the other hand, deep learning has been driven by availability of data with inputs and outputs of required task(s), and a comparatively lesser understanding of the model's working, however, with superb capabilities for learning complex relationships and patterns. *The presented research leveraged deep learning capabilities in analysing high dimensional manufacturing data, building complex I/O relationships to model product quality defects, and quantifying their uncertainties. The developed deep learning models were*

integrated with models that account for the physics of the manufacturing process to build kernels that enable the synthesis of manufacturing systems in a scalable and interpretable manner. These kernels, namely OSER, OSER-MAS, CLIP and OSEC, enable isolation and mitigation of object shape error-based quality defects within MAS. While doing so, they enable fulfilment of various requirements that have not been comprehensively addressed by current models, which can be summarized as (i) *High data dimensionality*, (ii) *Non-linearity*, (iii) *Collinearities*, (iv) *High faults multiplicity*, (v) *Uncertainty quantification*, (vi) *Dual data generation capability*, (vii) *High dimensionality and heterogeneity of process parameters*, (viii) *Fault localization for RCA*, (ix) *Scalability*, (x) *Interpretability*, (xi) *Uncertainty in isolated RC*, (xii) *KPI improvement*, (xiii) *MAS design architecture*, and (xiv) *MAS inherent stochasticity*. These challenges are solved by developing and validating various kernels that aim to fulfil the aforementioned requirements. The conclusions from the development and validation of the kernels as related to manufacturing application and scientific knowledge contribution are summarized below:

(1) Object Shape Error Response (OSER) (Fig. 8.1) leverages Bayesian 3D CNNs integrated with VRM in a closed-loop to estimate process parameters causing object shape errors within the final assembly. The trained model is then deployed in real MAS using point cloud quality data collected using 3D scanners. This fulfils requirement (i), (ii), (iii), (iv), (v) and (vi) within the scope of single-station assembly systems. The kernel closes the scientific gap between deep learning-based 3D object detection and object shape error estimation for manufacturing. Object shape error estimation enables the analysis of point cloud quality monitoring data, hence, provides capabilities for isolation of quality defects. The kernel also establishes a link between deep learning and physics-based models, driving towards a framework where causality of quality issues (RC) can be isolated rather than only a correlation between input quality data and output process parameters. Case studies demonstrate that the kernel can isolate RCs and quantify uncertainty in complex, non-linear MAS with high fault multiplicity, which current statistical and engineering-driven approaches are unable to do. From an industrial application point of view, the research establishes that point cloud generating data sources such

as 3D scanners can be implemented to isolate RCs, especially for assembly processes with compliant parts used in automotive, aerospace and consumer goods assembly processes where dimensional quality is the major source of quality issues and RC isolation translates into estimating shape error patterns and relating them to process parameters.

- (2) Object Shape Error Response for Multi-station Assembly Systems (OSER-MAS) (Fig. 8.1) leverages a multi-task Bayesian 3D U-net integrated with VRM to estimate process parameters and upstream shape errors at the end of upstream stations, which are leveraged to perform RCA using three steps, namely (i) fault identification, (ii) fault localization and (iii) fault isolation. This fulfils requirements (i), (ii), (iii), (iv), (v) and (vi) within the scope of MAS. Additionally, it also fulfils requirements (vii) and (viii), which are essential for RC isolation in MAS. The kernel widens the scope of the OSER kernel from single- to multi-station assembly systems hence, providing capabilities to perform RC isolation where quality variations are propagated through multiple upstream stations and point cloud quality data is only collected at the end of the system (end-of-line-sensing [13]). Case studies demonstrate that the accurate estimation of upstream shape errors enables localizing faulty assembly stations, which further enables isolation of RCs based on the estimated process parameters. From an industrial application perspective, the kernel establishes that performing RC isolation in MAS does not require placement of 3D scanner after each station or measurement of each subassembly. It can be used to inform the best placement of 3D scanners within the MAS to enable RCA for all possible combinations of RCs. However, the exact number of scanners may also depend on the structure and layout of the MAS and is a topic of future research.
- (3) Closed-loop In-Process (CLIP) (Fig. 8.1) leverages an integrated algorithm portfolio to enable scalability and interpretability of the OSER and OSER-MAS kernels. Scalability enabling algorithms include closed-loop training, uncertainty guided continual learning and transfer learning. Interpretability enabling algorithms include 3D gradient-based class activation maps. This fulfils requirements (ix) and (x). The kernel provides scientific validation regarding the ability to transfer learning between similar MAS; this is in line with previous engineering-based

approaches where rules for RC isolation for one assembly could be leveraged for RC isolation of other assembly systems. The case studies show a significant reduction in the number of training samples when leveraging a model previously trained on another assembly system. Industrially this translates into scalability as RC isolation models would be implemented across the whole production line. The kernel also links the isolated RC to shape error patterns within the point cloud quality data and hence, provides the required transparency that is needed to drive costly CAs. These scalability capabilities across multiple similar manufacturing systems and interpretability of isolated RCs mitigate the current gaps in the industrial adoption of deep learning-driven RC isolation models. The kernel is critical for automotive and other manufacturing sectors where the adoption of deep learning is blocked due to high computational requirements and the black-box nature of neural network models.

- (4) Object Shape Error Correction (OSEC) (Fig. 8.1) leverages deep reinforcement learning, specifically a DDPG actor-critic network that is trained using a VRM simulated MAS environment using a novel reward function parametrized by NPI-coefficients. It learns CA policies that mitigate sources of object shape error based on the isolated RCs and MAS state estimates. The policies simultaneously account for benefits, costs, constraints and stochasticity of the MAS. The approach can leverage CAs throughout all phases of the NPI cycle from assembly process design and launch until full production. This fulfils the requirement (xi), (xii), (xiii) and (xiv). The kernel mathematically formulates all requirements for correction of assembly systems, hence providing capabilities to mitigate sources of quality defects in a systematic and comprehensive manner; these requirements were previously accounted for using manual expertise, leading to decreased productivity and increased cost of quality. Such kernels are critical for high-volume automotive industries where quality deterioration, ineffective maintenance and machine downtime can lead to significant losses in terms of revenue, product performance and reputation. Case studies demonstrate that CA-related requirements such as MAS constraints and stochasticity become significant in transition from RC isolation to actionable corrections, hence, only isolating RCs is insufficient. A corresponding effective and feasible CA must be determined to mitigate quality

defects caused by the isolated RCs.

Further, the work is implemented physically as part of an *Industrial Demonstrator* to provide validation on real point cloud quality data collected from 3D scanners. A python library software termed Bayesian Deep Learning for Manufacturing is developed to further enable reproducibility of results and help the progress of the research work by open-sourcing code, data and algorithms. Overall, the four kernels provide a transformative framework by integrating MAS models supported by VRM simulator of assembly process, 3D scanners point cloud data of parts and subassembly shape errors, deep learning models, and Bayesian modelling to isolate and mitigate the sources of object shape errors, reducing and potentially eliminating geometric and dimensional variations in MAS. The overall research framework for isolation and mitigation of dimensional quality defects within MAS is summarized in Fig. 8.1.

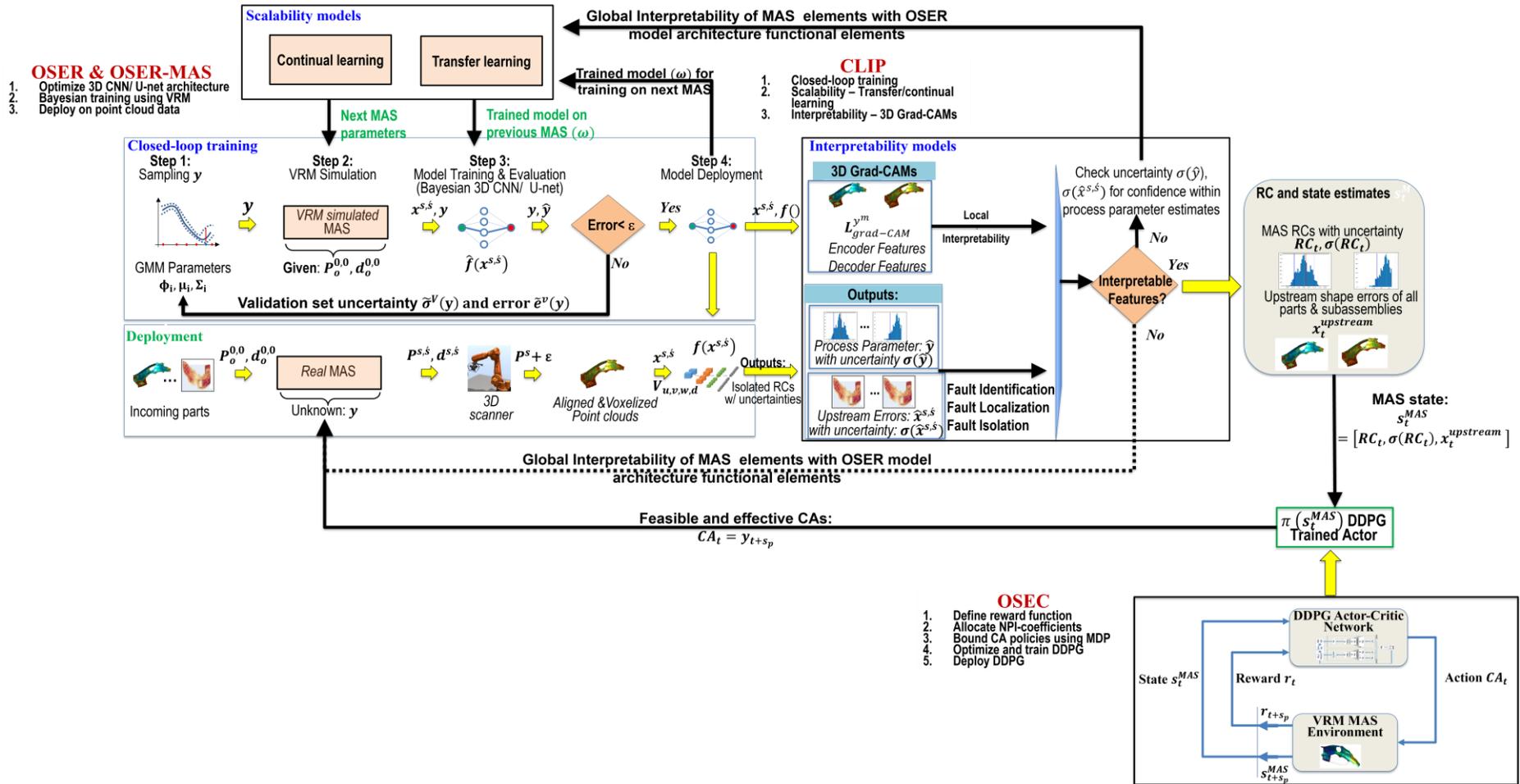


Fig. 8.1 Framework for isolation & mitigation of dimensional quality defects

8.2 Contributions

The developed kernels integrated within a single framework is much needed enabler for milestones such as *zero-defect manufacturing*, *right-first-time* and *resilient manufacturing*. The generalized problem formulation, novel methodology development, comprehensive verification and validation, and extensive benchmarking of the developed kernels address numerous limitations of state-of-the-art methods in isolating and mitigating object shape errors defects within the scope of MAS, enabling the advancement of the field at the intersection of deep learning and manufacturing assembly systems. The contributions of the developed kernels can be summarised as:

- (1) Novel 3D OSER and OSER-MAS kernels based on 3D CNN architectures such as multi-head Bayesian 3D U-Nets with a probabilistic encoder and attention-based decoder. The developed kernels expand the work done in deep learning for 3D object detection in manufacturing systems where the key goal is not to detect the object but to estimate various shape error patterns present on the final object/product and relate these variation patterns to manufacturing process parameters variations within the system. This is the first work to propose uncertainty enabled 3D CNN based Bayesian deep learning models for RCA of MAS that directly leverage point cloud data. Comprehensive benchmarking with existing machine learning models demonstrates superior performance with $R^2 = 0.98$ and $MAE = 0.05$ mm, thus, improving RCA capabilities by 29% as compared to current statistical approaches.
- (2) A novel CLIP kernel for training, scaling and interpreting the developed deep learning-driven kernels (OSER, OSER-MAS). The training is enabled by a closed-loop sampling framework that leverages a CAE simulator known as VRM [6]. Scalability is enabled by integrating continual/transfer learning algorithms which simultaneously reduce time-consuming computational cycles and the amount of labelled training data. Interpretability is enabled by applying 3D Grad-CAMs which link the extracted features by deep learning architecture to product shape error features of product assembled in a given MAS. The CLIP integrates deep learning approaches and physics-based models and provides capability for industrial adoption as the models can be trained from a design stage itself (when no

real data is available) in a scalable manner and provide interpretable causality driven RC isolation. Results highlight a reduction in up to 56% training samples when compared to random sampling/training from scratch methods with a loss in performance of up to 2.1%.

- (3) A novel OSEC kernel that leverages a deep reinforcement learning (DDPG) agent to enable effective and feasible CA estimation for the mitigation of RCs while accounting for uncertainty in isolated RC, cost and quality KPIs, MAS constraints (reconfigurability) and stochasticity. The agent enables the mitigation of object shape errors by estimating CA policies that estimate a functional nominal and can optimally adjust process parameters. Furthermore, the kernel is parametrized by NPI-coefficients that provides the capability to mitigate shape errors across all stages of the NPI lifecycle while allowing for the optimal tradeoff of changing priorities depending on the NPI stage. Current approaches do not mathematically formulate all CA requirements and rely on manual expertise to account for them while using simplifying assumptions such as a fixed design nominal. The OSEC policies enable mitigation of shape error with up to 40% better performance than current manual and SPC/APC based approaches with a failure rate of only 6.2%.
- (4) Formulation and implementation of an industrial demonstrator that provides verification and validation of the kernels using data collected from 3D scanners. Further, a python software is developed to verify and validate the methodologies using six case studies of varying complexities. The case studies include two automotive assembly processes from automotive industry: (i) door assembly process and (ii) cross-member assembly process.
- (5) Benchmarking methodologies against four categories of methods that can be leveraged to isolate and mitigate dimensional and geometric variation of assembled products, namely, (i) current approximately linear statistical/engineering-driven state-of-the-art RC isolation approaches; (ii) machine learning models in a multi-output regression or classification setting; (iii) deep learning models such as various types of CNNs and fully connected networks; and, (iv) SPC/APC and manual approaches used for CAs. This benchmarking collectively illustrates superior performance in terms of accuracy for isolating RCs, data required for training and effectiveness of estimated CAs and verifies the kernels' ability to fulfil the

aforementioned fourteen requirements. Currently, no frameworks exist that can provide capabilities to satisfy all requirements (i)-(xiv) as needed for isolation and mitigation of quality defects.

8.3 Limitations

The kernels developed in the thesis have many advantages compared to the state-of-the-art as discussed in the previous sections. Below the limitations of the developed kernels are discussed in the context of their key features, advantages and applications.

- (1) *Scope*: The work is done within the scope of sheet metal assembly systems considering process parameters related to non-ideal part variations, fixturing and joining. The fixtures are bounded by n-2-1 layout dedicated to sheet metal parts (deformable in one plane perpendicular to the part surface). The joining process is limited to 2-sided contact joining processes such as self-piercing riveting (SPR)/resistant spot welding (RSW) or 1-sided non-contact joining processes such as remote laser welding (RLW). The RCs and CAs correspond to only these sets of process parameters, i.e., part deformation, fixturing locating layout errors; and some related to joining (binary criterion of good joint/no joint). Defects caused due to variations in process parameters beyond the mentioned ones, such as the ones induced by heat (thermal deformation of parts or/and weld quality), phase change and/or fluid flow (weld quality), are outside the scope of the presented research and might affect the performance of the framework. However, the uncertainty quantification capabilities developed in this research can be used to identify such uncertainty as related to the above process parameters. In case of advances in the CAE simulator development (beyond the current capability of the VRM simulator used in this thesis) which will enhance abilities in modelling the impact of the currently unmodeled process parameters; the developed kernels can be used to generate a training dataset to train the kernels within the framework. However, the RCs and CAs kernels will need to be refined to include these new capabilities.
- (2) *Detectability*: The work considers only RCs that affect the shape error with a minimum error and resolution of 0.1 mm. The minimum resolution is also bounded by the accuracy and repeatability of the 3D Scanner used in this research in

collecting point cloud data. The hexagon WLS400A used to collect data for this research has an accuracy of 0.1 mm. This accuracy and resolution are sufficient for dimensional and geometry quality of automotive sheet metal assemblies but is not sufficient for applications such as surface roughness defect identification or for applications to different processes requiring higher accuracy (machined components, optical lenses, etc.). The developed kernels can be adopted to work with CAE simulators with higher fidelity and measurement systems with higher accuracy in order to train the proposed framework for such applications.

- (3) Physical System Validation: Both training and validation of the framework have been done using CAE simulation data as generated by the VRM simulator. Validation using point cloud data collected from the 3D Scanner has been done using a simplified physical demonstrator that generates object shape error patterns similar to ones generated in MAS. Further validation of the proposed framework within physical MAS is required to validate the performance capabilities of the framework as currently the proposed framework depends on the fidelity of the VRM simulator to emulate the physical system. Additionally, it will be advantageous to provide additional learning capability of the framework to provide feedback to the CAE (VRM) model for scenarios when it is unable to emulate the physical system accurately.
- (4) Uncertainty Estimation: The Bayesian deep learning model uses Mean Field Variational Inference (MFVI) [164] for uncertainty quantification on unseen data. This approximation may lead to poorly calibrated uncertainty estimates for RCs while scaling up as it assumes that the posterior of all weights within the neural network follow independent univariate normal distributions. Research in using more comprehensive priors for network weights such as multivariate normal distributions with full covariance matrix can be done to see the effect on uncertainty estimates for isolated RCs.
- (5) Changes after Deployment: The research assumes that the assembly system varies within the pre-set constraints on process parameters after model deployment. After deployment, any changes that are done to the MAS will impact the performance of the framework. However, they can be addressed through fine-tuning the models using uncertainty guided continual learning as discussed in the CLIP kernel.

8.4 Future Work

The framework and kernels presented in this thesis will lay the foundations for a much needed and industrially relevant field of study, which will combine manufacturing systems and artificial intelligence in a conceptually sound and practical manner. Critical topics for future research are discussed below:

- (1) *Bendlets*: Like capsule neural networks [229] that provide a principled way to formulate high-level objects mathematically, Bendlets can be formulated as an interpretable and transferable formulation of high-level object shape errors by quantitative modelling of invariant features [229] across different MAS. This would enable deeper integration between physics-based engineering models and deep representation learning. This could further enable building models with better performance on different MAS and then be potentially transferred across similar manufacturing systems such as stamping and machining. A framework can be developed that combines the CAE data and physical data to train a model that can go a step beyond isolating and mitigating RCs and also provide feedback to the CAE simulators to enable it to emulate the physical system more accurately. This capability can be used to address limitations (1) and (2) as they decouple the impact of different process parameters on the output shape error and provide a principled way to provide feedback from deep learning models to physics-based models.
- (2) *Physics-based closed-loop sampling*: Currently, the work leverages epistemic uncertainty to sample effectively from the CAE simulator. This can be done more efficiently while considering the physics of the behaviour of different process parameters within the MAS and hence, enable faster training of deep learning models. For example, the impact of particular process parameters related to pinhole displacement on the output shape error is linear hence requiring fewer training samples, while the impact of process parameters related to joining is non-linear, resulting in the requirement of a larger number of training samples. Integrating such prior engineering knowledge with epistemic uncertainty can potentially reduce the number of samples required for training. Additionally, the CAE simulator, used as data generator to emulate MAS during production, is currently based on Monte Carlo technique / Latin Hypercube Sampling (LHS) [182], which requires a

relatively large number of samples. The main drawbacks of MC-based methods have been widely recognized as: (i) very time consuming, especially for large scale problems; and (ii) poor computational accuracy at small to median sample sizes. There are other techniques such as pseudo-MC methods (for e.g., number-theory techniques (NT-net) which significantly reduce computational effort needed in simulation with higher accuracy yield [230].

- (3) Dynamic (online) learning: The continual learning framework can be extended to enable online learning for dynamically changing assembly systems. The changes can be quantified as concept drifts or covariate shifts. When detected, such changes would result in the model being fine-tuned using limited data collected from the system and large dataset collected from the CAE simulator. This would enable lifelong learning for dynamic manufacturing environments and would address limitation (5).
- (4) Self-supervised learning: Recent advances in using self-supervised learning [231][232] can be leveraged to learn latent concepts that can be transferred across various manufacturing systems using large scale datasets generated using multi-physics based simulators. These models can further be fine-tuned for downstream tasks such as regression and classification-based RCA and reinforcement learning-based CA models using exponentially lesser training samples.
- (5) Explainability and generalisability: The research leveraged 3D Grad-CAMs for explainability of the isolated RCs. These have been shown to have various limitations concerning the granularity of the generated interpretability maps [221]. The research needs to further consider other methods that provide local and global interpretability. Various model-agnostic approaches such as global model interpretation via recursive partitioning (GIRP), local interpretable model-agnostic explanation (LIME) and Shapley additive explanations (SHAP) and others need to be further explored [222] for a comprehensive understanding of the explainability and generalisability of the proposed approaches in isolating and mitigating quality defects. Model-specific methods [233] to deep learning also need to be explored to increase transparency and trust in the predictions. This will significantly aid large scale industrial adoption of deep learning approaches that drive costly manufacturing decisions.

(6) Preventive actions (PA): The corrective action framework can be extended to include preventive actions (PA) aimed at changing downstream process parameters to compensate for upstream shape variations.

References

- [1] P. Franciosa, M. Sokolov, S. Sinha, T. Sun, and D. Ceglarek, “Deep learning enhanced digital twin for Closed-Loop In-Process quality improvement,” *CIRP Ann.*, vol. 69, no. 1, pp. 369–372, Jan. 2020, doi: 10.1016/j.cirp.2020.04.110.
- [2] Q. Qi and F. Tao, “Digital Twin and Big Data Towards Smart Manufacturing and Industry 4.0: 360 Degree Comparison,” *IEEE Access*, vol. 6, pp. 3585–3593, Jan. 2018, doi: 10.1109/ACCESS.2018.2793265.
- [3] D. De Silva, S. Sierla, D. Alahakoon, E. Osipov, X. Yu, and V. Vyatkin, “Toward Intelligent Industrial Informatics: A Review of Current Developments and Future Directions of Artificial Intelligence in Industrial Applications,” *IEEE Industrial Electronics Magazine*, vol. 14, no. 2. Institute of Electrical and Electronics Engineers Inc., pp. 57–72, 01-Jun-2020, doi: 10.1109/MIE.2019.2952165.
- [4] “WLS400A | Hexagon Manufacturing Intelligence.” [Online]. Available: <https://www.hexagonmi.com/products/white-light-scanner-systems/hexagon-metrology-wls400a>. [Accessed: 26-Feb-2020].
- [5] W. G. Hatcher and W. Yu, “A Survey of Deep Learning: Platforms, Applications and Emerging Research Trends,” *IEEE Access*, vol. 6, pp. 24411–24432, Apr. 2018, doi: 10.1109/ACCESS.2018.2830661.
- [6] P. Franciosa, A. Palit, S. Gerbino, and D. Ceglarek, “A novel hybrid shell element formulation (QUAD+ and TRIA+): A benchmarking and comparative study,” *Finite Elem. Anal. Des.*, vol. 166, p. 103319, Nov. 2019, doi: 10.1016/j.finel.2019.103319.
- [7] W. J. Zhang and C. A. Van Luttervelt, “Toward a resilient manufacturing system,” *CIRP Ann. - Manuf. Technol.*, vol. 60, no. 1, pp. 469–472, Jan. 2011, doi: 10.1016/j.cirp.2011.03.041.

- [8] M. F. R. Zwicker, M. Moghadam, W. Zhang, and C. V. Nielsen, "Automotive battery pack manufacturing – a review of battery to tab joining," *J. Adv. Join. Process.*, vol. 1, p. 100017, Mar. 2020, doi: 10.1016/J.JAJP.2020.100017.
- [9] V. J. Shahi, A. Masoumi, P. Franciosa, and D. Ceglarek, "A quality-driven assembly sequence planning and line configuration selection for non-ideal compliant structures assemblies," *Int. J. Adv. Manuf. Technol.*, vol. 106, no. 1–2, pp. 15–30, Jan. 2020, doi: 10.1007/s00170-019-04294-w.
- [10] H. Wang and D. Ceglarek, "Quality-driven Sequence Planning and Line Configuration Selection for Compliant Structure Assemblies," *CIRP Ann.*, vol. 54, no. 1, pp. 31–35, Jan. 2005, doi: 10.1016/S0007-8506(07)60043-2.
- [11] H. Wang and D. Ceglarek, "Variation propagation modeling and analysis at preliminary design phase of multi-station assembly systems," *Assem. Autom.*, vol. 29, no. 2, pp. 154–166, 2009, doi: 10.1108/01445150910945606.
- [12] D. Ceglarek and J. Shi, "Dimensional variation reduction for automotive body assembly," *Manuf. Rev.*, vol. Vol. 8, pp. 139–154, 1995.
- [13] "ISO - ISO 17450-1:2011 - Geometrical product specifications (GPS) — General concepts — Part 1: Model for geometrical specification and verification." [Online]. Available: <https://www.iso.org/standard/53628.html>. [Accessed: 17-Mar-2021].
- [14] "ISO - ISO 10579:2010 - Geometrical product specifications (GPS) — Dimensioning and tolerancing — Non-rigid parts." [Online]. Available: <https://www.iso.org/standard/54707.html>. [Accessed: 31-Mar-2021].
- [15] S. Sinha, P. Franciosa, and D. Ceglarek, "Object Shape Error Response using Bayesian 3D Convolutional Neural Networks for Assembly Systems with Compliant Parts," *IEEE Trans. Ind. Informatics*, vol. 17, no. 10, pp. 6676–6686, 2021, doi: 10.1109/TII.2020.3043226.

- [16] A. Khan and D. Ceglarek, "Sensor Optimization for Fault Diagnosis in Multi-Fixture Assembly Systems With Distributed Sensing," *J. Manuf. Sci. Eng.*, vol. 122, no. 1, pp. 215–226, Feb. 2000, doi: 10.1115/1.538917.
- [17] M. Babu, P. Franciosa, and D. Ceglarek, "Spatio-Temporal Adaptive Sampling for effective coverage measurement planning during quality inspection of free form surfaces using robotic 3D optical scanner," *J. Manuf. Syst.*, vol. 53, pp. 93–108, Oct. 2019, doi: 10.1016/j.jmsy.2019.08.003.
- [18] E. Glorieux, P. Franciosa, and D. Ceglarek, "End-effector design optimisation and multi-robot motion planning for handling compliant parts," *Struct. Multidiscip. Optim.*, vol. 57, no. 3, pp. 1377–1390, Mar. 2018, doi: 10.1007/s00158-017-1798-x.
- [19] C. R. Qi *et al.*, "Pointnet: Deep learning on point sets for 3d classification and segmentation," *Proc. Comput. Vis. Pattern Recognit. (CVPR), IEEE*, vol. 1, no. 2, p. 4, 2017, doi: 10.1109/CVPR.2017.16.
- [20] D. Maturana and S. Scherer, "VoxNet: A 3D Convolutional Neural Network for real-time object recognition," in *IEEE International Conference on Intelligent Robots and Systems*, 2015, vol. 2015-Decem, pp. 922–928, doi: 10.1109/IROS.2015.7353481.
- [21] L. Ge, H. Liang, J. Yuan, and D. Thalmann, "Real-Time 3D Hand Pose Estimation with 3D Convolutional Neural Networks," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 41, no. 4, pp. 956–970, Apr. 2019, doi: 10.1109/TPAMI.2018.2827052.
- [22] A. Nguyen and B. Le, "3D point cloud segmentation: A survey," in *2013 6th IEEE Conference on Robotics, Automation and Mechatronics (RAM)*, 2013, pp. 225–230, doi: 10.1109/RAM.2013.6758588.
- [23] Z. Han, Z. Liu, J. Han, C.-M. Vong, S. Bu, and C. L. P. Chen, "Mesh

- Convolutional Restricted Boltzmann Machines for Unsupervised Learning of Features With Structure Preservation on 3-D Meshes,” *IEEE Trans. Neural Networks Learn. Syst.*, vol. 28, no. 10, pp. 2268–2281, Oct. 2017, doi: 10.1109/TNNLS.2016.2582532.
- [24] C. Leibig, V. Allken, M. S. Ayhan, P. Berens, and S. Wahl, “Leveraging uncertainty information from deep neural networks for disease detection,” *Sci. Rep.*, vol. 7, no. 1, p. 17816, Dec. 2017, doi: 10.1038/s41598-017-17876-z.
- [25] T. Wang, H. Gao, and J. Qiu, “A Combined Adaptive Neural Network and Nonlinear Model Predictive Control for Multirate Networked Industrial Process Control,” *IEEE Trans. Neural Networks Learn. Syst.*, vol. 27, no. 2, pp. 416–425, Feb. 2016, doi: 10.1109/TNNLS.2015.2411671.
- [26] Z. Gao *et al.*, “EEG-Based Spatio-Temporal Convolutional Neural Network for Driver Fatigue Evaluation,” *IEEE Trans. Neural Networks Learn. Syst.*, pp. 1–9, 2019, doi: 10.1109/TNNLS.2018.2886414.
- [27] F. Liao, M. Liang, Z. Li, X. Hu, and S. Song, “Evaluate the Malignancy of Pulmonary Nodules Using the 3-D Deep Leaky Noisy-or Network,” *IEEE Trans. Neural Networks Learn. Syst.*, pp. 1–12, 2019, doi: 10.1109/TNNLS.2019.2892409.
- [28] L. Monostori, “AI and machine learning techniques for managing complexity, changes and uncertainties in manufacturing,” *Eng. Appl. Artif. Intell.*, vol. 16, no. 4, pp. 277–291, Jun. 2003, doi: 10.1016/S0952-1976(03)00078-2.
- [29] S. Shao, S. McAleer, R. Yan, and P. Baldi, “Highly Accurate Machine Fault Diagnosis Using Deep Transfer Learning,” *IEEE Trans. Ind. Informatics*, vol. 15, no. 4, pp. 2446–2455, Apr. 2019, doi: 10.1109/TII.2018.2864759.
- [30] M. V. K. Chari and S. J. Salon, *the Finite Element Method*. Butterworth-Heinemann, 2000.

- [31] S. Charles Liu and S. Jack Hu, "Variation simulation for deformable sheet metal assemblies using finite element methods," *J. Manuf. Sci. Eng. Trans. ASME*, vol. 119, no. 3, pp. 368–374, Aug. 1997, doi: 10.1115/1.2831115.
- [32] P. Franciosa, S. Gerbino, and S. Patalano, "Variational modeling and assembly constraints in tolerance analysis of rigid part assemblies: Planar and cylindrical features," *Int. J. Adv. Manuf. Technol.*, vol. 49, no. 1–4, pp. 239–251, Jul. 2010, doi: 10.1007/s00170-009-2400-5.
- [33] W. Hao and D. Y. Yeung, "Towards Bayesian Deep Learning: A Framework and Some Existing Methods," *IEEE Trans. Knowl. Data Eng.*, vol. 28, no. 12, pp. 3395–3408, Dec. 2016, doi: 10.1109/TKDE.2016.2606428.
- [34] Y. Gal, "Uncertainty in deep learning," *Univ. Cambridge*, 2016.
- [35] M. Sun, T. Zhang, Y. Wang, G. Strbac, and C. Kang, "Using Bayesian Deep Learning to Capture Uncertainty for Residential Net Load Forecasting," *IEEE Trans. Power Syst.*, vol. 35, no. 1, pp. 188–201, Jan. 2020, doi: 10.1109/TPWRS.2019.2924294.
- [36] A. Kendall and Y. Gal, "What uncertainties do we need in bayesian deep learning for computer vision?," in *Advances in neural information processing systems*, 2017, pp. 5574–5584.
- [37] V. Mnih *et al.*, "Human-level control through deep reinforcement learning," *Nature*, vol. 518, no. 7540, pp. 529–533, Feb. 2015, doi: 10.1038/nature14236.
- [38] D. Silver *et al.*, "Mastering the game of Go with deep neural networks and tree search," *Nat. 2016 5297587*, vol. 529, no. 7587, pp. 484–489, Jan. 2016, doi: 10.1038/nature16961.
- [39] A. S. Xanthopoulos, A. Kiatipis, D. E. Koulouriotis, and S. Stieger, "Reinforcement Learning-Based and Parametric Production-Maintenance

- Control Policies for a Deteriorating Manufacturing System,” *IEEE Access*, vol. 6, pp. 576–588, Nov. 2017, doi: 10.1109/ACCESS.2017.2771827.
- [40] J. Xu, Z. Hou, W. Wang, B. Xu, K. Zhang, and K. Chen, “Feedback Deep Deterministic Policy Gradient with Fuzzy Reward for Robotic Multiple Peg-in-Hole Assembly Tasks,” *IEEE Trans. Ind. Informatics*, vol. 15, no. 3, pp. 1658–1667, Mar. 2019, doi: 10.1109/TII.2018.2868859.
- [41] C. Chen, M. Cui, F. Li, S. Yin, and X. Wang, “Model-Free Emergency Frequency Control Based on Reinforcement Learning,” *IEEE Trans. Ind. Informatics*, vol. 17, no. 4, pp. 2336–2346, Apr. 2021, doi: 10.1109/TII.2020.3001095.
- [42] J. Shi and S. Zhou, “Quality control and improvement for multistage systems: A survey,” *IIE Trans.*, vol. 41, no. 9, pp. 744–753, Jul. 2009, doi: 10.1080/07408170902966344.
- [43] Y. Liu, R. Sun, and S. Jin, “A survey on data-driven process monitoring and diagnostic methods for variation reduction in multi-station assembly systems,” *Assembly Automation*, vol. 39, no. 4. Emerald Group Publishing Ltd., pp. 727–739, 02-Sep-2019, doi: 10.1108/AA-10-2018-0174.
- [44] D. W. Apley and J. Shi, “Diagnosis of Multiple Fixture Faults in Panel Assembly,” *J. Manuf. Sci. Eng.*, vol. 120, no. 4, p. 793, Nov. 2008, doi: 10.1115/1.2830222.
- [45] M. Chang and D. C. Gossard, “Computational method for diagnosis of variation-related assembly problems,” *Int. J. Prod. Res.*, vol. 36, no. 11, pp. 2985–2995, 1998, doi: 10.1080/002075498192247.
- [46] K. G. Yu, S. Jin, and X. M. Lai, “Fixture variation diagnosis of compliant assembly using sensitivity matrix,” *J. Shanghai Jiaotong Univ.*, 2009, doi: 10.1007/s12204-009-0707-x.

- [47] R. Agrawal, J. F. Lawless, and R. J. Mackay, "Analysis of variation transmission in manufacturing processes - Part II," *J. Qual. Technol.*, vol. 31, no. 2, pp. 143–154, 1999, doi: 10.1080/00224065.1999.11979911.
- [48] C. Zou, W. Jiang, and F. Tsung, "A lasso-based diagnostic framework for multivariate statistical process control," *Technometrics*, vol. 53, no. 3, pp. 297–309, Aug. 2011, doi: 10.1198/TECH.2011.10034.
- [49] Y. Shang, F. Tsung, and C. Zou, "Statistical process control for multistage processes with binary outputs," *IIE Trans. (Institute Ind. Eng.)*, vol. 45, no. 9, pp. 1008–1023, Sep. 2013, doi: 10.1080/0740817X.2012.723839.
- [50] J. Jin and J. Shi, "State Space Modeling of Sheet Metal Assembly for Dimensional Control," *J. Manuf. Sci. Eng.*, vol. 121, no. 4, p. 756, Nov. 1999, doi: 10.1115/1.2833137.
- [51] Y. Ding, D. Ceglarek, and J. Shi, "Fault Diagnosis of Multistage Manufacturing Processes by Using State Space Approach," *J. Manuf. Sci. Eng.*, vol. 124, no. 2, p. 313, May 2002, doi: 10.1115/1.1445155.
- [52] Y. Ding, J. Shi, and D. Ceglarek, "Diagnosability Analysis of Multi-Station Manufacturing Processes," 2002, doi: 10.1115/1.1435645.
- [53] Y. Ding, S. Zhou, and Y. Chen, "A comparison of process variation estimators for in-process dimensional measurements and control," *J. Dyn. Syst. Meas. Control. Trans. ASME*, vol. 127, no. 1, pp. 69–79, Mar. 2005, doi: 10.1115/1.1870041.
- [54] D. Ceglarek and P. Prakash, "Enhanced piecewise least squares approach for diagnosis of ill-conditioned multistation assembly with compliant parts," *Proc. Inst. Mech. Eng. Part B J. Eng. Manuf.*, vol. 226, no. 3, pp. 485–502, Mar. 2012, doi: 10.1177/0954405411423458.

- [55] D. Ceglarek and J. Shi, "Fixture Failure Diagnosis for Autobody Assembly Using Pattern Recognition," *J. Eng. Ind.*, vol. 118, no. 1, p. 55, Feb. 1996, doi: 10.1115/1.2803648.
- [56] D. Ceglarek and J. Shi, "Fixture Failure Diagnosis for Sheet Metal Assembly with Consideration of Measurement Noise," *J. Manuf. Sci. Eng.*, vol. 121, no. 4, p. 771, Nov. 1999, doi: 10.1115/1.2833142.
- [57] Y. G. Liu and S. J. Hu, "Assembly Fixture Fault Diagnosis Using Designated Component Analysis," *J. Manuf. Sci. Eng.*, vol. 127, no. 2, p. 358, May 2005, doi: 10.1115/1.1852572.
- [58] D. Ceglarek, J. Shi, and S. M. Wu, "A Knowledge-Based Diagnostic Approach for the Launch of the Auto-Body Assembly Process," *J. Eng. Ind.*, vol. 116, no. 4, p. 491, Nov. 1994, doi: 10.1115/1.2902133.
- [59] S. Du and L. Xi, "Fault diagnosis in assembly processes based on engineering-driven rules and PSOSAEN algorithm," *Comput. Ind. Eng.*, vol. 60, no. 1, pp. 77–88, Feb. 2011, doi: 10.1016/j.cie.2010.10.001.
- [60] S. Du, J. Lv, and L. Xi, "An integrated system for on-line intelligent monitoring and identifying process variability and its application," *Int. J. Comput. Integr. Manuf.*, vol. 23, no. 6, pp. 529–542, 2010, doi: 10.1080/09511921003667730.
- [61] G. Beruvides, A. Villalonga, P. Franciosa, D. Ceglarek, and R. E. Haber, "Fault pattern identification in multi-stage assembly processes with non-ideal sheet-metal parts based on reinforcement learning architecture," in *Procedia CIRP*, 2018, vol. 67, pp. 601–606, doi: 10.1016/j.procir.2017.12.268.
- [62] S. Dey and J. A. Stori, "A Bayesian network approach to root cause diagnosis of process variations," *Int. J. Mach. Tools Manuf.*, vol. 45, no. 1, pp. 75–91, Jan. 2005, doi: 10.1016/j.ijmachtools.2004.06.018.

- [63] Y. Liu and S. Jin, "Application of Bayesian networks for diagnostics in the assembly process by considering small measurement data sets," *Int. J. Adv. Manuf. Technol.*, vol. 65, no. 9–12, pp. 1229–1237, Jun. 2013, doi: 10.1007/s00170-012-4252-7.
- [64] G. Arvanitis, A. S. Lalos, and K. Moustakas, "Robust and Fast 3D Saliency Mapping for Industrial Modeling Applications," *IEEE Trans. Ind. Informatics*, pp. 1–1, 2020, doi: 10.1109/TII.2020.3003455.
- [65] L. Da Xu, C. Wang, Z. Bi, and J. Yu, "AutoAssem: An automated assembly planning system for complex products," *IEEE Trans. Ind. Informatics*, vol. 8, no. 3, pp. 669–678, 2012, doi: 10.1109/TII.2012.2188901.
- [66] T. Phoomboplab and D. Ceglarek, "Process Yield Improvement Through Optimum Design of Fixture Layouts in 3D Multistation Assembly Systems," *J. Manuf. Sci. Eng.*, vol. 130, no. 6, pp. 0610051–06100517, Dec. 2008, doi: 10.1115/1.2977826.
- [67] Q. Rong, J. Shi, and D. Ceglarek, "Adjusted Least Squares Approach for Diagnosis of Ill-Conditioned Compliant Assemblies," *J. Manuf. Sci. Eng.*, vol. 123, no. 3, p. 453, Aug. 2001, doi: 10.1115/1.1365116.
- [68] F. Adly *et al.*, "Simplified subspaced regression network for identification of defect patterns in semiconductor wafer maps," *IEEE Trans. Ind. Informatics*, vol. 11, no. 6, pp. 1267–1276, Dec. 2015, doi: 10.1109/TII.2015.2481719.
- [69] G. A. Susto, A. Schirru, S. Pampuri, S. McLoone, and A. Beghi, "Machine learning for predictive maintenance: A multiple classifier approach," *IEEE Trans. Ind. Informatics*, vol. 11, no. 3, pp. 812–820, Jun. 2015, doi: 10.1109/TII.2014.2349359.
- [70] X. Wang, L. T. Yang, L. Song, H. Wang, L. Ren, and J. Deen, "A Tensor-based Multi-Attributes Visual Feature Recognition Method for Industrial Intelligence,"

IEEE Trans. Ind. Informatics, pp. 1–1, Jun. 2020, doi: 10.1109/tii.2020.2999901.

- [71] K. Mannar, D. Ceglarek, F. Niu, and B. Abifaraj, “Fault region localization: Product and process improvement based on field performance and manufacturing measurements,” *IEEE Trans. Autom. Sci. Eng.*, vol. 3, no. 4, pp. 423–439, Oct. 2006, doi: 10.1109/TASE.2006.880526.
- [72] S. Sinha, E. Glorieux, P. Franciosa, and D. Ceglarek, “3D convolutional Neural networks to estimate assembly process parameters using 3D point-clouds,” in *Proceedings of SPIE*, 2019, vol. 11059, doi: 10.1117/12.2526062.
- [73] W. Huang, J. Lin, Z. Kong, and D. Ceglarek, “Stream-of-variation (SOVA) modeling - Part II: A generic 3D variation model for rigid body assembly in multistation assembly processes,” *J. Manuf. Sci. Eng. Trans. ASME*, vol. 129, no. 4, pp. 832–842, Jan. 2007, doi: 10.1115/1.2738953.
- [74] K. Mannar and D. Ceglarek, “Functional capability space and optimum process adjustments for manufacturing processes with in-specs failure,” *IIE Trans. (Institute Ind. Eng.)*, vol. 42, no. 2, pp. 95–106, Feb. 2010, doi: 10.1080/07408170902789027.
- [75] G. E. P. Box, A. Luceño, and M. del C. Paniagua-Quiñones, “Statistical control by monitoring and adjustment,” p. 333, 2009.
- [76] F. Tsung, J. Shi, and C. F. J. Wu, “Joint Monitoring of PID-Controlled Processes,” <https://doi.org/10.1080/00224065.1999.11979926>, vol. 31, no. 3, pp. 275–285, 2018, doi: 10.1080/00224065.1999.11979926.
- [77] J. Krüger *et al.*, “Innovative control of assembly systems and lines,” *CIRP Ann.*, vol. 66, no. 2, pp. 707–730, Jan. 2017, doi: 10.1016/j.cirp.2017.05.010.
- [78] M. Metzger and G. Polaków, “A survey on applications of agent technology in industrial process control,” *IEEE Transactions on Industrial Informatics*, vol. 7,

no. 4, pp. 570–581, Nov-2011, doi: 10.1109/TII.2011.2166781.

- [79] L. J. Wells and J. A. Camelio, “A bio-inspired approach for self-correcting compliant assembly systems,” *J. Manuf. Syst.*, vol. 32, no. 3, pp. 464–472, Jul. 2013, doi: 10.1016/J.JMSY.2013.03.002.
- [80] R. van den Berg, E. Lefeber, and K. Rooda, “Modeling and control of a manufacturing flow line using partial differential equations,” *IEEE Trans. Control Syst. Technol.*, vol. 16, no. 1, pp. 130–136, Jan. 2008, doi: 10.1109/TCST.2007.903085.
- [81] D. Mourtzis and E. Vlachou, “A cloud-based cyber-physical system for adaptive shop-floor scheduling and condition-based maintenance,” *J. Manuf. Syst.*, vol. 47, pp. 179–198, Apr. 2018, doi: 10.1016/J.JMSY.2018.05.008.
- [82] L. C. Alwan and H. V. Roberts, “Time-series modeling for statistical process control,” *J. Bus. Econ. Stat.*, vol. 6, no. 1, pp. 87–95, 1988, doi: 10.1080/07350015.1988.10509640.
- [83] Y. Wang, X. Yue, R. Tuo, J. H. Hunt, and J. Shi, “Effective Model Calibration via Sensible Variable Identification and Adjustment, with Application to Composite Fuselage Simulation,” *Ann. Appl. Stat.*, vol. 14, no. 4, pp. 1759–1776, Dec. 2019.
- [84] D. Ceglarek *et al.*, “Rapid deployment of remote laser welding processes in automotive assembly systems,” *CIRP Ann. - Manuf. Technol.*, vol. 64, no. 1, pp. 389–394, 2015, doi: 10.1016/J.CIRP.2015.04.119.
- [85] J. Antony, “Simultaneous optimisation of multiple quality characteristics in manufacturing processes using Taguchi’s quality loss function,” *Int. J. Adv. Manuf. Technol.*, vol. 17, no. 2, pp. 134–138, 2001, doi: 10.1007/s001700170201.

- [86] J. Camelio, S. J. Hu, and D. Ceglarek, “Modeling Variation Propagation of Multi-Station Assembly Systems with Compliant Parts,” *J. Mech. Des. Trans. ASME*, vol. 125, no. 4, pp. 673–681, 2003, doi: 10.1115/1.1631574.
- [87] P. Franciosa and D. Ceglarek, “Hierarchical synthesis of multi-level design parameters in assembly system,” *CIRP Ann. - Manuf. Technol.*, vol. 64, no. 1, pp. 149–152, 2015, doi: 10.1016/j.cirp.2015.04.028.
- [88] O. Ronneberger, P. Fischer, and T. Brox, “U-net: Convolutional networks for biomedical image segmentation,” in *Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)*, 2015, vol. 9351, pp. 234–241, doi: 10.1007/978-3-319-24574-4_28.
- [89] Y. Wen, P. Vicol, J. Ba, D. Tran, and R. Grosse, “Flipout: Efficient pseudo-independent weight perturbations on mini-batches,” in *6th International Conference on Learning Representations, ICLR 2018 - Conference Track Proceedings*, 2018.
- [90] O. Oktay *et al.*, “Attention u-net: Learning where to look for the pancreas,” *arXiv Prepr. arXiv1804.03999*, 2018.
- [91] Ö. Çiçek, A. Abdulkadir, S. S. Lienkamp, T. Brox, and O. Ronneberger, “3D U-net: Learning dense volumetric segmentation from sparse annotation,” 2016, vol. 9901 LNCS, pp. 424–432, doi: 10.1007/978-3-319-46723-8_49.
- [92] S. Ebrahimi, M. Elhoseiny, T. Darrell, and M. Rohrbach, “Uncertainty-guided Continual Learning with Bayesian Neural Networks,” Jun. 2019.
- [93] R. R. Selvaraju, M. Cogswell, A. Das, R. Vedantam, D. Parikh, and D. Batra, “Grad-cam: Visual explanations from deep networks via gradient-based localization,” in *Proceedings of the IEEE international conference on computer vision*, 2017, pp. 618–626.

- [94] M. Bortolini, F. G. Galizia, and C. Mora, "Reconfigurable manufacturing systems: Literature review and research trend," *J. Manuf. Syst.*, vol. 49, pp. 93–106, Oct. 2018, doi: 10.1016/J.JMSY.2018.09.005.
- [95] "Connected Papers | Find and explore academic papers." [Online]. Available: <https://www.connectedpapers.com/>. [Accessed: 19-Mar-2021].
- [96] R. Mantripragada and D. E. Whitney, "Modeling and controlling variation propagation in mechanical assemblies using state transition models," *IEEE Trans. Robot. Autom.*, vol. 15, no. 1, pp. 124–140, 1999, doi: 10.1109/70.744608.
- [97] J. F. Lawless, R. J. Mackay, and J. A. Robinson, "Analysis of variation transmission in manufacturing processes - Part I," *J. Qual. Technol.*, vol. 31, no. 2, pp. 131–142, 1999, doi: 10.1080/00224065.1999.11979910.
- [98] S. C. Liu, S. J. Hu, and T. C. Woo, "Tolerance analysis for sheet metal assemblies," *J. Mech. Des. Trans. ASME*, vol. 118, no. 1, pp. 62–67, 1996, doi: 10.1115/1.2826857.
- [99] T. Zhang and J. Shi, "Stream of Variation Modeling and Analysis for Compliant Composite Part Assembly-Part I: Single-Station Processes," *J. Manuf. Sci. Eng. Trans. ASME*, vol. 138, no. 12, 2016, doi: 10.1115/1.4033231.
- [100] S. Dahlstrom and L. Lindkvist, "Variation simulation of sheet metal assemblies using the method of influence coefficients with contact modeling," *J. Manuf. Sci. Eng. Trans. ASME*, vol. 129, no. 3, pp. 615–622, 2007, doi: 10.1115/1.2714570.
- [101] Y. Shang, S. Cen, and M. J. Zhou, "8-node unsymmetric distortion-immune element based on Airy stress solutions for plane orthotropic problems," *Acta Mech.*, vol. 229, no. 12, pp. 5031–5049, 2018, doi: 10.1007/s00707-018-2291-3.

- [102] F. Gruttmann and W. Wagner, "A linear quadrilateral shell element with fast stiffness computation," *Comput. Methods Appl. Mech. Eng.*, vol. 194, no. 39–41, pp. 4279–4300, 2005, doi: 10.1016/j.cma.2004.11.005.
- [103] Y. Zhang, H. Zhou, J. Li, W. Feng, and D. Li, "A 3-node flat triangular shell element with corner drilling freedoms and transverse shear correction," *Int. J. Numer. Methods Eng.*, vol. 86, no. 12, pp. 1413–1434, 2011, doi: 10.1002/nme.3109.
- [104] S. Cen, Y. Shang, C. F. Li, and H. G. Li, "Hybrid displacement function element method: A simple hybrid-Trefftz stress element method for analysis of Mindlin-Reissner plate," *Int. J. Numer. Methods Eng.*, vol. 98, no. 3, pp. 203–234, 2014, doi: 10.1002/nme.4632.
- [105] D. W. Apley and J. Shi, "Diagnosis of multiple fixture faults in panel assembly," *J. Manuf. Sci. Eng. Trans. ASME*, vol. 120, no. 4, pp. 793–801, 1998, doi: 10.1115/1.2830222.
- [106] Y. Chen, J. Jin, and J. Shi, "Integration of dimensional quality and locator reliability in design and evaluation of multi-station body-in-white assembly processes," *IIE Trans. (Institute Ind. Eng.)*, vol. 36, no. 9, pp. 827–839, 2004, doi: 10.1080/07408170490473015.
- [107] C. Liu, Y. Ding, and Y. Chen, "Optimal coordinate sensor placements for estimating mean and variance components of variation sources," *IIE Trans. (Institute Ind. Eng.)*, vol. 37, no. 9, pp. 877–889, 2005, doi: 10.1080/07408170590969889.
- [108] D. Djurdjanovic and J. Ni, "Dimensional errors of fixtures, locating and measurement datum features in the stream of variation modeling in machining," *J. Manuf. Sci. Eng. Trans. ASME*, vol. 125, no. 4, pp. 716–730, 2003, doi: 10.1115/1.1621424.

- [109] S. Zhou, Y. Chen, Y. Ding, and J. Shi, “Diagnosability study of multistage manufacturing processes based on linear mixed-effects models,” *Technometrics*, vol. 45, no. 4, pp. 312–325, 2003, doi: 10.1198/004017003000000131.
- [110] J. A. Camelio, S. J. Hu, and D. J. Ceglarek, “Impact of fixture design on sheet metal assembly variation,” *Proc. ASME Des. Eng. Tech. Conf.*, vol. 3, pp. 133–140, 2002, doi: 10.1115/detc2002/dfm-34167.
- [111] W. Huang and D. Ceglarek, “Mode-based decomposition of part form error by discrete-cosine-transform with implementation to assembly and stamping system with compliant parts,” *CIRP Ann. - Manuf. Technol.*, vol. 51, no. 1, pp. 21–26, Jan. 2002, doi: 10.1016/S0007-8506(07)61457-7.
- [112] J. A. Camelio, S. J. Hu, and D. Ceglarek, “Impact of fixture design on sheet metal assembly variation,” *J. Manuf. Syst.*, vol. 23, no. 3, pp. 182–193, 2004, doi: 10.1016/S0278-6125(05)00006-3.
- [113] A. Krizhevsky, I. Sutskever, and G. E. Hinton, “ImageNet Classification with Deep Convolutional Neural Networks.” pp. 1097–1105, 2012.
- [114] Z. Liu, H. Tang, Y. Lin, and S. Han, “Point-voxel CNN for efficient 3d deep learning,” *arXiv*, 2019.
- [115] Y. Xiao, Y. Ma, Q. Huang, and J. Zhang, “Deep Multi-level Feature Learning on Point Sets for 3D Object Recognition,” *DEStech Trans. Comput. Sci. Eng.*, no. csse, 2018, doi: 10.12783/dtcse/csse2018/24500.
- [116] C. Wang, M. Cheng, F. Sohel, M. Bennamoun, and J. Li, “NormalNet: A voxel-based CNN for 3D object classification and retrieval,” *Neurocomputing*, vol. 323, pp. 139–147, 2019, doi: 10.1016/j.neucom.2018.09.075.
- [117] S. Ghadai, X. Y. Lee, A. Balu, S. Sarkar, and A. Krishnamurthy, “Multi-level 3D CNN for learning multi-scale spatial features,” *IEEE Comput. Soc. Conf.*

- Comput. Vis. Pattern Recognit. Work.*, vol. 2019-June, pp. 1152–1156, 2019, doi: 10.1109/CVPRW.2019.00150.
- [118] S. Ghadai, A. Balu, S. Sarkar, and A. Krishnamurthy, “Learning localized features in 3D CAD models for manufacturability analysis of drilled holes,” *Comput. Aided Geom. Des.*, vol. 62, pp. 263–275, 2018, doi: 10.1016/j.cagd.2018.03.024.
- [119] K. G. Lore, D. Stoecklein, M. Davies, B. Ganapathysubramanian, and S. Sarkar, “Deep Action Sequence Learning for Causal Shape Transformation,” 2016.
- [120] B. S. S. Pokuri, S. Ghosal, A. Kokate, S. Sarkar, and B. Ganapathysubramanian, “Interpretable deep learning for guided microstructure-property explorations in photovoltaics,” *npj Comput. Mater.*, vol. 5, no. 1, 2019, doi: 10.1038/s41524-019-0231-y.
- [121] R. Singh, V. Shah, B. Pokuri, S. Sarkar, B. Ganapathysubramanian, and C. Hegde, “Physics-aware Deep Generative Models for Creating Synthetic Microstructures,” *arXiv*, 2018.
- [122] F. Jia, Y. Lei, J. Lin, X. Zhou, and N. Lu, “Deep neural networks: A promising tool for fault characteristic mining and intelligent diagnosis of rotating machinery with massive data,” *Mech. Syst. Signal Process.*, vol. 72–73, pp. 303–315, 2016, doi: 10.1016/j.ymsp.2015.10.025.
- [123] L. Wen, X. Li, L. Gao, and Y. Zhang, “A New Convolutional Neural Network-Based Data-Driven Fault Diagnosis Method,” *IEEE Trans. Ind. Electron.*, vol. 65, no. 7, pp. 5990–5998, 2018, doi: 10.1109/TIE.2017.2774777.
- [124] L. Wen, L. Gao, and X. Li, “A new deep transfer learning based on sparse auto-encoder for fault diagnosis,” *IEEE Trans. Syst. Man, Cybern. Syst.*, vol. 49, no. 1, pp. 136–144, 2019, doi: 10.1109/TSMC.2017.2754287.

- [125] O. Janssens *et al.*, “Convolutional Neural Network Based Fault Detection for Rotating Machinery,” *J. Sound Vib.*, vol. 377, pp. 331–345, 2016, doi: 10.1016/j.jsv.2016.05.027.
- [126] M. Xia, T. Li, L. Xu, L. Liu, and C. W. De Silva, “Fault Diagnosis for Rotating Machinery Using Multiple Sensors and Convolutional Neural Networks,” *IEEE/ASME Trans. Mechatronics*, vol. 23, no. 1, pp. 101–110, 2018, doi: 10.1109/TMECH.2017.2728371.
- [127] T. Ince, S. Kiranyaz, L. Eren, M. Askar, and M. Gabbouj, “Real-Time Motor Fault Detection by 1-D Convolutional Neural Networks,” *IEEE Trans. Ind. Electron.*, vol. 63, no. 11, pp. 7067–7075, 2016, doi: 10.1109/TIE.2016.2582729.
- [128] L. Wen, X. Li, X. Li, and L. Gao, “A new transfer learning based on VGG-19 network for fault diagnosis,” *Proc. 2019 IEEE 23rd Int. Conf. Comput. Support. Coop. Work Des. CSCWD 2019*, pp. 205–209, 2019, doi: 10.1109/CSCWD.2019.8791884.
- [129] H. Gao, B. Cheng, J. Wang, K. Li, J. Zhao, and D. Li, “Object Classification Using CNN-Based Fusion of Vision and LIDAR in Autonomous Vehicle Environment,” *IEEE Trans. Ind. Informatics*, vol. 14, no. 9, pp. 4224–4230, Sep. 2018, doi: 10.1109/TII.2018.2822828.
- [130] J. Han, H. Chen, N. Liu, C. Yan, and X. Li, “CNNs-Based RGB-D saliency detection via cross-view transfer and multiview fusion,” *IEEE Trans. Cybern.*, vol. 48, no. 11, pp. 3171–3183, Nov. 2018, doi: 10.1109/TCYB.2017.2761775.
- [131] Y. Lei, F. Jia, J. Lin, S. Xing, and S. X. Ding, “An Intelligent Fault Diagnosis Method Using Unsupervised Feature Learning Towards Mechanical Big Data,” *IEEE Trans. Ind. Electron.*, vol. 63, no. 5, pp. 3137–3147, 2016, doi: 10.1109/TIE.2016.2519325.

- [132] W. Qian, S. Li, and J. Wang, “A New Transfer Learning Method and its Application on Rotating Machine Fault Diagnosis Under Variant Working Conditions,” *IEEE Access*, vol. 6, pp. 69907–69917, 2018, doi: 10.1109/ACCESS.2018.2880770.
- [133] B. Yang, Y. Lei, F. Jia, and S. Xing, “An intelligent fault diagnosis approach based on transfer learning from laboratory bearings to locomotive bearings,” *Mech. Syst. Signal Process.*, vol. 122, pp. 692–706, 2019, doi: 10.1016/j.ymsp.2018.12.051.
- [134] L. Guo, Y. Lei, S. Xing, T. Yan, and N. Li, “Deep Convolutional Transfer Learning Network: A New Method for Intelligent Fault Diagnosis of Machines with Unlabeled Data,” *IEEE Trans. Ind. Electron.*, vol. 66, no. 9, pp. 7316–7325, 2019, doi: 10.1109/TIE.2018.2877090.
- [135] R. Zhang, H. Tao, L. Wu, and Y. Guan, “Transfer Learning with Neural Networks for Bearing Fault Diagnosis in Changing Working Conditions,” *IEEE Access*, vol. 5, pp. 14347–14357, 2017, doi: 10.1109/ACCESS.2017.2720965.
- [136] J. Deng, W. Dong, R. Socher, L.-J. Li, Kai Li, and Li Fei-Fei, “ImageNet: A large-scale hierarchical image database,” 2010, pp. 248–255, doi: 10.1109/cvpr.2009.5206848.
- [137] J. Y. Hwang and W. Kuo, “Model-based clustering for integrated circuit yield enhancement,” *Eur. J. Oper. Res.*, vol. 178, no. 1, pp. 143–153, 2007, doi: 10.1016/j.ejor.2005.11.032.
- [138] C. H. Wang, “Recognition of semiconductor defect patterns using spatial filtering and spectral clustering,” *Expert Syst. Appl.*, vol. 34, no. 3, pp. 1914–1923, 2008, doi: 10.1016/j.eswa.2007.02.014.
- [139] C. H. Wang, “Separation of composite defect patterns on wafer bin map using support vector clustering,” *Expert Syst. Appl.*, vol. 36, no. 2 PART 1, pp. 2554–

2561, 2009, doi: 10.1016/j.eswa.2008.01.057.

- [140] T. Yuan, W. Kuo, and S. J. Bae, “Detection of spatial defect patterns generated in semiconductor fabrication processes,” *IEEE Trans. Semicond. Manuf.*, vol. 24, no. 3, pp. 392–403, 2011, doi: 10.1109/TSM.2011.2154870.
- [141] T. Nakazawa and D. V. Kulkarni, “Wafer map defect pattern classification and image retrieval using convolutional neural network,” *IEEE Trans. Semicond. Manuf.*, vol. 31, no. 2, pp. 309–314, 2018, doi: 10.1109/TSM.2018.2795466.
- [142] T. Nakazawa and D. V. Kulkarni, “Anomaly detection and segmentation for wafer defect patterns using deep Convolutional Encoder-Decoder Neural Network Architectures in Semiconductor Manufacturing,” *IEEE Trans. Semicond. Manuf.*, vol. 32, no. 2, pp. 250–256, 2019, doi: 10.1109/TSM.2019.2897690.
- [143] S. Cheon, H. Lee, C. O. Kim, and S. H. Lee, “Convolutional Neural Network for Wafer Surface Defect Classification and the Detection of Unknown Defect Class,” *IEEE Trans. Semicond. Manuf.*, vol. 32, no. 2, pp. 163–170, 2019, doi: 10.1109/TSM.2019.2902657.
- [144] Y. Kim, D. Cho, and J. H. Lee, “Wafer Map Classifier using Deep Learning for Detecting Out-of-Distribution Failure Patterns,” *Proc. Int. Symp. Phys. Fail. Anal. Integr. Circuits, IPFA*, vol. 2020-July, 2020, doi: 10.1109/IPFA49335.2020.9260877.
- [145] X. Qi, G. Chen, Y. Li, X. Cheng, and C. Li, “Applying Neural-Network-Based Machine Learning to Additive Manufacturing: Current Applications, Challenges, and Future Perspectives,” *Engineering*, vol. 5, no. 4, pp. 721–729, 2019, doi: 10.1016/j.eng.2019.04.012.
- [146] A. Garg, K. Tai, and M. M. Savalani, “State-of-the-art in empirical modelling of rapid prototyping processes,” *Rapid Prototyp. J.*, vol. 20, no. 2, pp. 164–178,

2014, doi: 10.1108/RPJ-08-2012-0072.

- [147] A. Garg, K. Tai, C. H. Lee, and M. M. Savalani, “A hybrid M5'-genetic programming approach for ensuring greater trustworthiness of prediction ability in modelling of FDM process,” *J. Intell. Manuf.*, vol. 25, no. 6, pp. 1349–1365, 2014, doi: 10.1007/s10845-013-0734-1.
- [148] Z. Li, Z. Zhang, J. Shi, and D. Wu, “Prediction of surface roughness in extrusion-based additive manufacturing with machine learning,” *Robot. Comput. Integr. Manuf.*, vol. 57, pp. 488–495, 2019, doi: 10.1016/j.rcim.2019.01.004.
- [149] C. Gobert, E. W. Reutzel, J. Petrich, A. R. Nassar, and S. Phoha, “Application of supervised machine learning for defect detection during metallic powder bed fusion additive manufacturing using high resolution imaging,” *Addit. Manuf.*, vol. 21, pp. 517–528, 2018, doi: 10.1016/j.addma.2018.04.005.
- [150] L. Scime and J. Beuth, “Anomaly detection and classification in a laser powder bed additive manufacturing process using a trained computer vision algorithm,” *Addit. Manuf.*, vol. 19, pp. 114–126, 2018, doi: 10.1016/j.addma.2017.11.009.
- [151] Z. Zhu, N. Anwer, Q. Huang, and L. Mathieu, “Machine learning in tolerancing for additive manufacturing,” *CIRP Ann.*, vol. 67, no. 1, pp. 157–160, 2018, doi: 10.1016/j.cirp.2018.04.119.
- [152] X. Yuan, J. Zhou, B. Huang, Y. Wang, C. Yang, and W. Gui, “Hierarchical Quality-Relevant Feature Representation for Soft Sensor Modeling: A Novel Deep Learning Strategy,” *IEEE Trans. Ind. Informatics*, vol. 16, no. 6, pp. 3721–3730, Jun. 2020, doi: 10.1109/TII.2019.2938890.
- [153] X. Yuan, Z. Ge, and Z. Song, “Locally weighted kernel principal component regression model for soft sensing of nonlinear time-variant processes,” *Ind. Eng. Chem. Res.*, vol. 53, no. 35, pp. 13736–13749, 2014, doi: 10.1021/ie4041252.

- [154] X. Yuan, Y. Wang, C. Yang, W. Gui, and L. Ye, “Probabilistic density-based regression model for soft sensing of nonlinear industrial processes,” *J. Process Control*, vol. 57, pp. 15–25, 2017, doi: 10.1016/j.jprocont.2017.06.002.
- [155] L. Yao and Z. Ge, “Locally weighted prediction methods for latent factor analysis with supervised and semisupervised process data,” *IEEE Trans. Autom. Sci. Eng.*, vol. 14, no. 1, pp. 126–138, 2017, doi: 10.1109/TASE.2016.2608914.
- [156] X. Yuan, L. Ye, L. Bao, Z. Ge, and Z. Song, “Nonlinear feature extraction for soft sensor modeling based on weighted probabilistic PCA,” *Chemom. Intell. Lab. Syst.*, vol. 147, pp. 167–175, 2015, doi: 10.1016/j.chemolab.2015.08.014.
- [157] X. Yuan, Z. Ge, B. Huang, Z. Song, and Y. Wang, “Semisupervised JITL framework for nonlinear industrial soft sensing based on locally semisupervised weighted PCR,” *IEEE Trans. Ind. Informatics*, vol. 13, no. 2, pp. 532–541, 2017, doi: 10.1109/TII.2016.2610839.
- [158] X. Yuan, B. Huang, Y. Wang, C. Yang, and W. Gui, “Deep Learning-Based Feature Representation and Its Application for Soft Sensor Modeling with Variable-Wise Weighted SAE,” *IEEE Trans. Ind. Informatics*, vol. 14, no. 7, pp. 3235–3243, 2018, doi: 10.1109/TII.2018.2809730.
- [159] X. Yuan, S. Qi, and Y. Wang, “Stacked Enhanced Auto-Encoder for Data-Driven Soft Sensing of Quality Variable,” *IEEE Trans. Instrum. Meas.*, vol. 69, no. 10, pp. 7953–7961, 2020, doi: 10.1109/TIM.2020.2985614.
- [160] X. Yuan, C. Ou, Y. Wang, C. Yang, and W. Gui, “Deep quality-related feature extraction for soft sensing modeling: A deep learning approach with hybrid VW-SAE,” *Neurocomputing*, vol. 396, pp. 375–382, 2020, doi: 10.1016/j.neucom.2018.11.107.
- [161] X. Yan, J. Wang, and Q. Jiang, “Deep relevant representation learning for soft sensing,” *Inf. Sci. (Ny)*, vol. 514, pp. 263–274, 2020, doi:

10.1016/j.ins.2019.11.039.

- [162] B. Lakshminarayanan, A. Pritzel, and C. Blundell, “Simple and scalable predictive uncertainty estimation using deep ensembles,” *Adv. Neural Inf. Process. Syst.*, vol. 2017-Decem, pp. 6403–6414, 2017.
- [163] Y. Gal and Z. Ghahramani, “Dropout as a Bayesian approximation: Representing model uncertainty in deep learning,” in *international conference on machine learning*, 2016, pp. 1050–1059.
- [164] C. Blundell, J. Cornebise, K. Kavukcuoglu, and D. Wierstra, “Weight uncertainty in neural networks,” *32nd Int. Conf. Mach. Learn. ICML 2015*, vol. 2, pp. 1613–1622, May 2015.
- [165] Y. Gal and Z. Ghahramani, “On Modern Deep Learning and Variational Inference,” *Nips*, pp. 1–9, 2015.
- [166] C. Guo, G. Pleiss, Y. Sun, and K. Q. Weinberger, “On calibration of modern neural networks,” *34th Int. Conf. Mach. Learn. ICML 2017*, vol. 3, pp. 2130–2143, 2017.
- [167] A. Kendall, V. Badrinarayanan, and R. Cipolla, “Bayesian segnet: Model uncertainty in deep convolutional encoder-decoder architectures for scene understanding,” *Br. Mach. Vis. Conf. 2017, BMVC 2017*, 2017, doi: 10.5244/c.31.57.
- [168] R. Cipolla, Y. Gal, and A. Kendall, “Multi-task Learning Using Uncertainty to Weigh Losses for Scene Geometry and Semantics,” *Proc. IEEE Comput. Soc. Conf. Comput. Vis. Pattern Recognit.*, pp. 7482–7491, 2018, doi: 10.1109/CVPR.2018.00781.
- [169] T. P. Lillicrap *et al.*, “Continuous control with deep reinforcement learning,” in *4th ICLR*, 2016.

- [170] G. Barth-Maron *et al.*, “Distributed distributional deterministic policy gradients,” *arXiv*, 2018.
- [171] Z. Wang *et al.*, “Sample efficient actor-critic with experience replay,” *5th Int. Conf. Learn. Represent. ICLR 2017 - Conf. Track Proc.*, 2017.
- [172] A. Tavakoli, F. Pardo, and P. Kormushev, “Action branching architectures for deep reinforcement learning,” *32nd AAAI Conf. Artif. Intell. AAAI 2018*, pp. 4131–4138, 2018.
- [173] S. Fujimoto, H. Van Hoof, and D. Meger, “Addressing Function Approximation Error in Actor-Critic Methods,” *35th Int. Conf. Mach. Learn. ICML 2018*, vol. 4, pp. 2587–2601, 2018.
- [174] D. Horgan *et al.*, “Distributed prioritized experience replay,” *arXiv*, 2018.
- [175] S. Spielberg, P. Kumar, and B. Gopaluni, “Process Control using Deep Reinforcement Learning,” *Preprint*, no. 1, pp. 1–6, 2017.
- [176] S. S. Pon Kumar, A. Tulsyan, B. Gopaluni, and P. Loewen, “A Deep Learning Architecture for Predictive Control,” *IFAC-PapersOnLine*, vol. 51, no. 18, pp. 512–517, 2018, doi: 10.1016/j.ifacol.2018.09.373.
- [177] S. Spielberg, A. Tulsyan, N. P. Lawrence, P. D. Loewen, and R. B. Gopaluni, “Deep reinforcement learning for process control: A primer for beginners,” *arXiv*, 2020, doi: 10.1002/aic.16689.
- [178] K. Asadi and J. D. Williams, “Sample-efficient Deep Reinforcement Learning for Dialog Control,” 2016.
- [179] Y. Ma, W. Zhu, M. G. Benton, and J. Romagnoli, “Continuous control of a polymerization system with deep reinforcement learning,” *J. Process Control*, vol. 75, pp. 40–47, 2019, doi: 10.1016/j.jprocont.2018.11.004.

- [180] D. M. Blei, A. Kucukelbir, and J. D. McAuliffe, “Variational Inference: A Review for Statisticians,” *J. Am. Stat. Assoc.*, vol. 112, no. 518, pp. 859–877, Jan. 2016, doi: 10.1080/01621459.2017.1285773.
- [181] L. Li, K. Jamieson, G. DeSalvo, A. Rostamizadeh, and A. Talwalkar, “Hyperband: A novel bandit-based approach to hyperparameter optimization,” *J. Mach. Learn. Res.*, vol. 18, no. 1, pp. 6765–6816, 2017.
- [182] M. Stein, “Large sample properties of simulations using latin hypercube sampling,” *Technometrics*, vol. 29, no. 2, pp. 143–151, 1987, doi: 10.1080/00401706.1987.10488205.
- [183] Y. Wu and K. He, “Group normalization,” in *Proceedings of the European conference on computer vision (ECCV)*, 2018, pp. 3–19.
- [184] X. Glorot and Y. Bengio, “Understanding the difficulty of training deep feedforward neural networks,” in *Proceedings of the thirteenth international conference on artificial intelligence and statistics*, 2010, pp. 249–256.
- [185] D. P. Kingma and J. L. Ba, “Adam: A method for stochastic optimization,” in *3rd International Conference on Learning Representations, ICLR 2015 - Conference Track Proceedings*, 2015.
- [186] GoogleResearch, “TensorFlow: Large-scale machine learning on heterogeneous systems,” *Google Res.*, 2015, doi: 10.1207/s15326985ep4001.
- [187] S. Sinha, P. Franciosa, and D. Ceglarek, “Bayesian Deep Learning for Manufacturing,” 2020. [Online]. Available: https://github.com/sumitsinha/Deep_Learning_for_Manufacturing.
- [188] K. Bastani, B. Barazandeh, and Z. Kong, “Fault Diagnosis in Multistation Assembly Systems Using Spatially Correlated Bayesian Learning Algorithm,” *J. Manuf. Sci. Eng.*, vol. 140, no. 3, p. 031003, Dec. 2017, doi:

10.1115/1.4038184.

- [189] F. Pedregosa *et al.*, “Scikit-learn: Machine Learning in Python,” *J. Mach. Learn. Res.*, vol. 12, pp. 2825–2830, 2011.
- [190] H. Su, S. Maji, E. Kalogerakis, and E. Learned-Miller, “Multi-view Convolutional Neural Networks for 3D Shape Recognition,” in *2015 IEEE International Conference on Computer Vision (ICCV)*, 2015, pp. 945–953, doi: 10.1109/ICCV.2015.114.
- [191] S. Sinha, P. Franciosa, and D. Ceglarek, “Building a Scalable and Interpretable Bayesian Deep Learning Framework for Quality Control of Free Form Surfaces,” *IEEE Access*, vol. 9, pp. 50188–50208, 2021, doi: 10.1109/ACCESS.2021.3068867.
- [192] S. Sinha, P. Franciosa, and D. Ceglarek, “Object Shape Error Response using Bayesian 3D Convolutional Neural Networks for Assembly Systems with Compliant Parts,” in *IEEE Transactions on Industrial Informatics*, 2020, p. pp 104–109, doi: 10.1109/TII.2020.3043226.
- [193] K. He, X. Zhang, S. Ren, and J. Sun, “Deep residual learning for image recognition,” in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2016, pp. 770–778.
- [194] F. Milletari, N. Navab, and S. A. Ahmadi, “V-Net: Fully convolutional neural networks for volumetric medical image segmentation,” in *Proceedings - 2016 4th International Conference on 3D Vision, 3DV 2016*, 2016, pp. 565–571, doi: 10.1109/3DV.2016.79.
- [195] J. Bergstra, J. B. Ca, and Y. B. Ca, “Random Search for Hyper-Parameter Optimization Yoshua Bengio,” 2012.
- [196] T. LaBonte, C. Martinez, and S. A. Roberts, “We Know Where We Don’t Know:

3D Bayesian CNNs for Credible Geometric Uncertainty,” Oct. 2019.

- [197] Z. Kong, D. Ceglarek, and W. Huang, “Multiple Fault Diagnosis Method in Multistation Assembly Processes Using Orthogonal Diagonalization Analysis,” *J. Manuf. Sci. Eng.*, vol. 130, no. 1, p. 011014, Feb. 2008, doi: 10.1115/1.2783228.
- [198] C. R. Qi, L. Yi, H. Su, and L. J. Guibas, “PointNet++: Deep Hierarchical Feature Learning on Point Sets in a Metric Space,” Jun. 2017.
- [199] F. W. Qin, J. Bai, and W. Q. Yuan, “Research on intelligent fault diagnosis of mechanical equipment based on sparse deep neural networks,” *J. Vibroengineering*, vol. 19, no. 4, pp. 2439–2455, Jun. 2017, doi: 10.21595/jve.2017.17146.
- [200] J. Long, E. Shelhamer, and T. Darrell, “Fully convolutional networks for semantic segmentation,” in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2015, pp. 3431–3440.
- [201] K. Simonyan and A. Zisserman, “Very deep convolutional networks for large-scale image recognition,” *arXiv Prepr. arXiv1409.1556*, 2014.
- [202] Z. Zhou, M. M. Rahman Siddiquee, N. Tajbakhsh, and J. Liang, “Unet++: A nested u-net architecture for medical image segmentation,” in *Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)*, 2018, vol. 11045 LNCS, pp. 3–11, doi: 10.1007/978-3-030-00889-5_1.
- [203] V. Badrinarayanan, A. Kendall, and R. Cipolla, “SegNet: A Deep Convolutional Encoder-Decoder Architecture for Image Segmentation,” *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 39, no. 12, pp. 2481–2495, Dec. 2017, doi: 10.1109/TPAMI.2016.2644615.

- [204] L. C. Chen, G. Papandreou, I. Kokkinos, K. Murphy, and A. L. Yuille, “DeepLab: Semantic Image Segmentation with Deep Convolutional Nets, Atrous Convolution, and Fully Connected CRFs,” *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 40, no. 4, pp. 834–848, Apr. 2018, doi: 10.1109/TPAMI.2017.2699184.
- [205] G. Putnik *et al.*, “Scalability in manufacturing systems design and operation: State-of-the-art and future developments roadmap,” *CIRP Ann. - Manuf. Technol.*, vol. 62, no. 2, pp. 751–774, Jan. 2013, doi: 10.1016/j.cirp.2013.05.002.
- [206] P. Wang and R. X. Gao, “Transfer learning for enhanced machine fault diagnosis in manufacturing,” *CIRP Ann.*, vol. 69, no. 1, pp. 413–416, Jan. 2020, doi: 10.1016/j.cirp.2020.04.074.
- [207] Y. Xu, Y. Sun, X. Liu, and Y. Zheng, “A Digital-Twin-Assisted Fault Diagnosis Using Deep Transfer Learning,” *IEEE Access*, vol. 7, pp. 19990–19999, 2019, doi: 10.1109/ACCESS.2018.2890566.
- [208] J. Yosinski, J. Clune, Y. Bengio, and H. Lipson, “How transferable are features in deep neural networks?” pp. 3320–3328, 2014.
- [209] Y. Bengio, “Deep learning of representations for unsupervised and transfer learning,” in *Proceedings of ICML workshop on unsupervised and transfer learning*, 2012, pp. 17–36.
- [210] H. C. Shin *et al.*, “Deep Convolutional Neural Networks for Computer-Aided Detection: CNN Architectures, Dataset Characteristics and Transfer Learning,” *IEEE Trans. Med. Imaging*, vol. 35, no. 5, pp. 1285–1298, May 2016, doi: 10.1109/TMI.2016.2528162.
- [211] J. Wang and C. Zhao, “Mode-cloud data analytics based transfer learning for soft sensor of manufacturing industry with incremental learning ability,” *Control Eng. Pract.*, vol. 98, p. 104392, May 2020, doi:

10.1016/j.conengprac.2020.104392.

- [212] F. Feng, R. H. M. Chan, X. Shi, Y. Zhang, and Q. She, “Challenges in Task Incremental Learning for Assistive Robotics,” *IEEE Access*, vol. 8, pp. 3434–3441, 2020, doi: 10.1109/ACCESS.2019.2955480.
- [213] G. A. Tahir and C. K. Loo, “An open-ended continual learning for food recognition using class incremental extreme learning machines,” *IEEE Access*, vol. 8, pp. 82328–82346, 2020, doi: 10.1109/ACCESS.2020.2991810.
- [214] H. J. Song and S. B. Park, “Enriching translation-based knowledge graph embeddings through continual learning,” *IEEE Access*, vol. 6, pp. 60489–60497, 2018, doi: 10.1109/ACCESS.2018.2874656.
- [215] A. Bielski and T. Trzcinski, “Understanding Multimodal Popularity Prediction of Social Media Videos With Self-Attention,” *IEEE Access*, vol. 6, pp. 74277–74287, 2018, doi: 10.1109/ACCESS.2018.2884831.
- [216] H. Y. Chen and C. H. Lee, “Vibration Signals Analysis by Explainable Artificial Intelligence (XAI) Approach: Application on Bearing Faults Diagnosis,” *IEEE Access*, vol. 8, pp. 134246–134256, 2020, doi: 10.1109/ACCESS.2020.3006491.
- [217] L. Zhao, Y. Zeng, P. Liu, and G. He, “Band selection via explanations from convolutional neural networks,” *IEEE Access*, vol. 8, pp. 56000–56014, 2020, doi: 10.1109/ACCESS.2020.2981475.
- [218] S. Sinha, P. Franciosa, and D. Ceglarek, “Root Cause Analysis of Multi-Station Assembly Systems with Non-Ideal Compliant Parts using Bayesian 3D U-Nets,” *IEEE Trans. Autom. Sci. Eng.*, 2021.
- [219] H. Yosinski, J. Clune, J. Bengio, Y., & Lipson, “How transferable are features in deep neural networks?,” in *Advances in neural information processing systems*, 2014, pp. 3320–3328.

- [220] W. Huang, J. Liu, V. Chalivendra, D. Ceglarek, Z. Kong, and Y. Zhou, “Statistical modal analysis for variation characterization and application in manufacturing quality control,” <https://doi.org/10.1080/0740817X.2013.814928>, vol. 46, no. 5, pp. 497–511, May 2014, doi: 10.1080/0740817X.2013.814928.
- [221] H. G. Ramaswamy and others, “Ablation-cam: Visual explanations for deep convolutional network via gradient-free localization,” in *Proceedings of the IEEE/CVF Winter Conference on Applications of Computer Vision*, 2020, pp. 983–991.
- [222] A. Adadi and M. Berrada, “Peeking Inside the Black-Box: A Survey on Explainable Artificial Intelligence (XAI),” *IEEE Access*, vol. 6, pp. 52138–52160, Sep. 2018, doi: 10.1109/ACCESS.2018.2870052.
- [223] G. E. Uhlenbeck and L. S. Ornstein, “On the theory of the Brownian motion,” *Phys. Rev.*, vol. 36, no. 5, pp. 823–841, Sep. 1930, doi: 10.1103/PhysRev.36.823.
- [224] “IPQI.” [Online]. Available: https://warwick.ac.uk/fac/sci/wmg/research/materials/dlm/projects/ipqi_new/. [Accessed: 26-Mar-2021].
- [225] “Overview — Sphinx documentation.” [Online]. Available: <https://www.sphinx-doc.org/en/master/>. [Accessed: 26-Mar-2021].
- [226] S. Sinha, P. Franciosa, and D. Ceglarek, “Point Cloud Object Shape Error Datasets for Root Cause Analysis of Multi-Station Assembly Systems,” Dec. 2020, doi: 10.5281/ZENODO.4537219.
- [227] G. Matheron, N. Perrin, and O. Sigaud, “The problem with DDPG: understanding failures in deterministic environments with sparse rewards,” *arXiv*, Nov. 2019.

- [228] L. Metz, J. Ibarz, N. Jaitly, and J. Davidson, “Discrete sequential prediction of continuous actions for deep rl,” *arXiv*, 2017.
- [229] S. Sabour, N. Frosst, and G. E. Hinton, “Dynamic routing between capsules,” in *arXiv*, 2017, pp. 3859–3869.
- [230] W. Huang, D. Ceglarek, and Z. Zhou, “Tolerance analysis for design of multistage manufacturing processes using number-theoretical net method (NT-net),” *Int. J. Flex. Manuf. Syst.*, vol. 16, no. 1, pp. 65–90, Jan. 2004, doi: 10.1023/B:FLEX.0000039173.07009.8a.
- [231] A. Vaswani *et al.*, “Attention is All You Need,” in *Proceedings of the 31st International Conference on Neural Information Processing Systems*, 2017, pp. 6000–6010.
- [232] A. Dosovitskiy *et al.*, “An Image is Worth 16x16 Words: Transformers for Image Recognition at Scale,” Oct. 2020.
- [233] W. Samek, T. Wiegand, and K.-R. Müller, “Explainable Artificial Intelligence: Understanding, Visualizing and Interpreting Deep Learning Models,” Aug. 2017.
- [234] D. E. Whitney, *Mechanical assemblies: their design, manufacture, and role in product development*, vol. 1. Oxford university press New York, 2004.
- [235] E. Savio, L. De Chiffre, and R. Schmitt, “Metrology of freeform shaped parts,” *CIRP Ann. - Manuf. Technol.*, vol. 56, no. 2, pp. 810–835, Jan. 2007, doi: 10.1016/j.cirp.2007.10.008.
- [236] H. A. M. Daanen and F. B. Ter Haar, “3D whole body scanners revisited,” *Displays*, vol. 34, no. 4. Elsevier, pp. 270–275, 01-Oct-2013, doi: 10.1016/j.displa.2013.08.011.

- [237] I. Goodfellow, Y. Bengio, and A. Courville, *Deep learning*, vol. 29, no. 7553. MIT press Cambridge, 2016.
- [238] R. S. Sutton and A. G. Barto, "Reinforcement learning: An introduction." Cambridge, MA: MIT Press, 2011.
- [239] H. Dong, Z. Ding, and S. Zhang, *Deep reinforcement learning: Fundamentals, research and applications*. Springer Singapore, 2020.
- [240] A. Papadopoulos, P. Korus, and N. Memon, "Hard-Attention for Scalable Image Classification," Feb. 2021.

Appendix A: Additional Background and Methods

This chapter aims to provide an aggregated review of various broader concepts leveraged to develop the proposed kernels. Firstly, an additional background is given for assembly systems and optical measurement systems. Secondly, an overview of methods including deep neural networks, deep reinforcement learning and attention gates are provided.

Assembly Systems

Assembly system [234] refers to processes that enable aggregation of various parts to form an assembly that can be leveraged to perform specific functions. The research focuses on the assembly of non-ideal compliant parts (3D free-form surfaces also referred to as objects) used extensively in automotive, aerospace and shipbuilding manufacturing industries. Variations of process parameters within the assembly system lead to shape error within the objects. A detailed description of single station assembly systems and multi-station assembly systems are given in section 3.2.1 and section 4.2, respectively. *The research aims to develop kernels that enable isolation and mitigation of these sources of object shape errors within the scope of single and multi-station assembly systems.*

Optical Measurement Systems

Measurement systems are crucial for providing dimensional and geometric quality information for parts, sub-assemblies and products (collectively addressed as *objects*) within manufacturing assembly systems. These can be implemented off-line or in-line to estimate the shape error on each object by comparing it with the nominal CAD geometry of the object. Optical measurement systems allow non-contact measurement enabling fast acquisition of a large amount of data (points). Various kinds of optical measurement systems [235] as (i) point sensor, (ii) line sensor or (iii) area sensor, can be leveraged to obtain data of different granularity. A point-based sensor aims to

measure a single or collection of point(s) on an object. While area sensors are also known as 3D scanners [236], can be leveraged to measure all points within a field of view. 3D scanners enable collecting granular high dimensional point cloud data that provide comprehensive information about the shape error of measured objects. Such data acquisition sensors require simultaneous development of models that can effectively leverage this data for quality improvement. Existing approaches are designed to isolate and mitigate object shape errors using only point-based data. *The research aims to develop kernels that can leverage point cloud data obtained from 3D scanners to isolate and mitigate sources of object shape error within manufacturing assembly systems.*

Deep Neural Networks

Learning a relationship between variables requires modelling a function that inputs a set of variables and performs transformations to output the desired variable. The quality of the relationship learning is dependent on the transformations used within the function. Linear regression is the most straightforward statistical tool to perform this using a linear transformation function. To generalize this further for non-linear cases, fixed basis functions are used that perform non-linear transformations on the input variables, followed by which linear regression can be done on these transformed variables. This forms the basis for methods such as wavelets and polynomial basis functions. Further, this constraint of using fixed basis functions can be relaxed, and parameterized basis functions can be leveraged. These learn the values of these function parameters based on a set of known input and output variables. Deep neural networks aim to build a network of these basis functions by using hierarchical layers that perform transformations on the previous layer's output. Each layer is considered a building block, and the modularity in the composition of blocks embodies the versatility of deep neural networks. These blocks can be adapted (for e.g., using convolutions) to process high dimensional input variables such as images, point clouds, sequences to estimate the desired outputs. *The research aims to leverage this capability of deep neural networks to develop kernels (OSER, OSER-MAS) that relate high dimensional object shape errors to process parameters enabling applications like isolation and mitigation*

of root causes of shape error. An extensive literature on deep learning [237] can be referred to for further details.

Deep Reinforcement Learning

Reinforcement learning has performed well in various complex sequential tasks by training agents to optimize their control in an environment to obtain the maximum amount of reward. However, as the task complexity increases, such as high-dimensional sensory inputs with unmanageable state-space in environments, the ability of these agents to perform optimal control decreases. Agents need to have the capabilities to derive efficient representations and generalize experience to new instances. Deep neural networks have excelled in extracting features from high-dimensional image, video and text data and then build models to generalize across a wide variety of tasks. Deep reinforcement learning aims to integrate the feature extraction and generalisation capabilities of deep neural networks with sequential decision-making capabilities of reinforcement learning to build agents that can perform optimal control in complex environments with large state or action spaces. This integration is done by replacing value functions and policy functions with deep neural networks [37]. The replacement enables the agent to generalize the value or policy function across large state spaces. *The proposed OSEC framework leverages DDPG [169] that parametrizes both policy (actor) and action-value (critic) function with neural networks.* An extensive literature on reinforcement learning [238] and deep reinforcement learning [239] can be referred to for further details.

Attention Gates

Attention gates (AG) are leveraged in computer vision and NLP domains for caption generation, classification, and machine translation. Attention mechanism [231] are of two categories and can be leveraged while training model parameters. (i) *Hard attention* mechanisms [240] such as iterative region proposal are not differentiable and have to be trained alternatively using reinforcement learning techniques. (ii) *Soft attention* mechanisms [231] are differentiable and can be trained using backpropagation. This has been further extended to self-attention [231] to remove dependency on external

gating information. AG enable progressive feature suppression within irrelevant spatial regions in the context of spatial feature extraction [63] as required for fault localisation in assembly systems. Attention coefficients identify salient shape error regions and further preserve and enhance those features to enable high performance in fault localisation. This enables the transfer of necessary geometric shape error features and disambiguation of irrelevant geometric information simultaneously. Further, this is concatenated within the up-sampling kernel to estimate upstream shape errors in MAS. *The proposed 3D U-Net architecture (Chapter 4) incorporates these attention gates to enhance shape error and geometric features that pass through skip connections between the encoder and decoder's corresponding levels.*

Appendix B: Software Implementation

Python library: Bayesian Deep Learning for Manufacturing

Code:

https://github.com/sumitsinha/Deep_Learning_for_Manufacturing

Documentation:

https://sumitsinha.github.io/Deep_Learning_for_Manufacturing/html/index.html

Bayesian Deep Learning for Manufacturing 2.0 (dlmfg)



Object Shape Error Response (OSER)

Digital Lifecycle Management - In Process Quality Improvement (IPQI)

DOI [10.1117/12.2526062](https://doi.org/10.1117/12.2526062) License [MIT](#) website [up](#) Maintained? [yes](#) status [beta](#) Documentation [WIP](#) PyPI [WIP](#) keras [tensorflow](#)

Overview

The open source **Bayesian Deep Learning for Manufacturing (dlmfg) Library** is built using a TensorFlow, TensorFlow Probability and Keras back end to build:

- Bayesian deep learning models such as **Bayesian 3D Convolutional Neural Network** and **Bayesian 3D U-net** to enable **root cause analysis** in Manufacturing Systems.
- Deep reinforcement learning models such as **Deep Deterministic Policy Gradients** to enable **control and correction** in Manufacturing Systems.

The library can be used across various domains such as assembly systems, stamping, additive manufacturing and milling where the key problem is **Object Shape Error Detection and Estimation**. The library is build using Object Oriented Programming to enable extension and contribution from other related disciplines within the artificial intelligence community as well as the manufacturing community.

Video

[A Video for the work can be found here](#)

Dataset

[Computer Aided Engineering Dataset for model training and testing using Variation Response Method \(VRM\) can be found here](#)

Published Work

The published works can be found below:

- [Building a Scalable and Interpretable Bayesian Deep Learning Framework for Quality Control of Free Form Surfaces](#)
- [Object Shape Error Response using Bayesian 3D Convolutional Neural Networks for Assembly Systems with Compliant Parts](#)
- [A novel hybrid shell element formulation \(QUAD+ and TRIA+\): A benchmarking and comparative study](#)
- [3D convolutional neural networks to estimate assembly process parameters using 3D point-clouds](#)
- [Deep learning enhanced digital twin for closed loop in-process quality improvement](#)

Two follow up papers are currently under review are expected by April 2021

Documentation

The complete documentation and other ongoing research can be found here: [Documentation and Research](#).

Highlights and New Additions

1. [Bayesian 3D U-Net](#) model integrating Bayesian layers and attention blocks for uncertainty quantification and superior decoder performance leveraging the *where to look capability* with multi-task capabilities to estimate bot real-valued(regression) and categorical(classification) based values. The Decoder is used to obtain real-valued segmentation maps
2. [Deep Reinforcement Learning](#) using deep deterministic policy gradient (DDPG) and a custom made multi physics manufacturing environment to build agents to correct manufacturing systems
3. [Closed Loop Sampling](#) for faster model training and convergence using epistemic uncertainty of the Bayesian CNN models
4. [Matlab Python Integration](#) to enable low latency connection between multi-physics manufacturing environments (Matlab) and TensorFlow based DDPG agents
5. [Multi-Physics Manufacturing System Simulations](#) to generate custom datasets for various fault scenarios using Variation Response Method (VRM) kernel
6. [Uncertainty guided continual learning](#) to enable life long/incremental training for multiple case studies
7. [Automated Model Selection](#) using Keras Tuner that enables hyperparameter optimization and benchmarking for various deep learning architectures
8. [Exploratory notebooks](#) for various case studies
9. [3D Gradient-weighted Class Activation Maps](#) for interpretability of deep learning models
10. [Datasets for Industrial multi-station case studies](#) for training and benchmarking deep learning models

Installation

The library can be cloned using:

```
git clone https://github.com/sumitsinha/Deep_Learning_for_Manufacturing
```

Dataset Download

The datasets can be downloaded by running the `download_data.py` file within the `downloads` file. The specifics of the download can be specified in the `download/config.py` file.

The library consists of the following two key datasets:

1. [3D Cloud of Point data with node deviations and process parameters for Single Part Car Halo Reinforcement](#) – Obtained due to variations in the Measurement Station locators and Stamping Process
2. [3D Cloud of Point data with node deviations and process parameters for Two part assembly for Car Door Inner and Hinge Reinforcement](#) – Obtained due to variations in the Assembly System locators and joining tools.

Bayesian 3D CNN Model Architecture

Motivated by the recent development of Bayesian Deep Neural Networks Bayesian models considering parameters to be distributions have been build using TensorFlow Probability. The Aleatoric uncertainty have been modelled using Multi-variate normal distributions as outputs while the epistemic distributions have been modelled using distributions on model parameters by using Flip-out layers.

The Bayesian 3D CNN model for single station system has the following layers:

```
> negloglik = lambda y, rv_y: -rv_y.log_prob(y)
> model = tf.keras.Sequential([
tf.keras.layers.InputLayer(input_shape=(voxel_dim,voxel_dim,voxel_dim,deviation_channels)), tfp.layers.Convolution3DFlipout(32, kern
tfp.layers.Convolution3DFlipout(32, kernel_size=(4,4,4),strides=(2,2,2),activation=tf.nn.relu),
tfp.layers.Convolution3DFlipout(32, kernel_size=(3,3,3),strides=(1,1,1),activation=tf.nn.relu),
tf.keras.layers.MaxPooling3D(pool_size=[2, 2, 2]),
tf.keras.layers.Flatten(),
tfp.layers.DenseFlipout(128,activation=tf.nn.relu),
tfp.layers.DenseFlipout(64,activation=tf.nn.relu),
tfp.layers.DenseFlipout(self.output_dimension),
tfp.layers.DistributionLambda(lambda t: tfd.MultivariateNormalDiag(loc=t[..., :self.output_dimension], scale_diag=aleatoric_tensor))
model.compile(optimizer=tf.keras.optimizers.Adam(),loss=negloglik,metrics=[tf.keras.metrics.MeanAbsoluteError()])
```

Bayesian 3D U-Net Model Architecture

For scaling to multi-station systems consisting of both categorical and continuous process parameters and prediction of point-clouds (object shape error) in previous stations a 3D - Net Attention based architecture is leveraged.

```
def attention_block(x, g, inter_channel):

    theta_x = Conv(inter_channel, [1,1,1], strides=[1,1,1])(x)
    phi_g = Conv(inter_channel, [1,1,1], strides=[1,1,1])(g)

    f = Activation('relu')(add([theta_x, phi_g]))
    psi_f = Conv(1, [1,1,1], strides=[1,1,1])(f)

    rate = Activation('sigmoid')(psi_f)

    att_x = multiply([x, rate])
    return att_x

input_size=(voxel_dim,voxel_dim,voxel_dim,deviation_channels)
inputs = Input(input_size)
x = inputs

# Down sampling
for i in range(depth):
    out_channel = 2**i * filter_root

    # Residual/Skip connection
    res = tfp.layers.Convolution3DFlipout(out_channel, kernel_size=1, kernel_divergence_fn=k1_divergence_function,padding='s

    # First Conv Block with Conv, BN and activation
    conv1 = tfp.layers.Convolution3DFlipout(out_channel, kernel_size=3, kernel_divergence_fn=k1_divergence_function,padding=
    #if batch_norm:
        #conv1 = BatchNormalization(name="BN{}_1".format(i))(conv1)
```

```

act1 = Activation(activation, name="Act{}_1".format(i))(conv1)

# Second Conv block with Conv and BN only
conv2 = tfp.layers.Convolution3DFlipout(out_channel, kernel_size=3, padding='same', kernel_divergence_fn=k1_divergence_f
# if batch_norm:
    conv2 = BatchNormalization(name="BN{}_2".format(i))(conv2)

resconnection = Add(name="Add{}_1".format(i))([res, conv2])

act2 = Activation(activation, name="Act{}_2".format(i))(resconnection)

# Max pooling
if i < depth - 1:
    long_connection_store[str(i)] = act2
    x = MaxPooling(padding='same', name="MaxPooling{}_1".format(i))(act2)
else:
    x = act2

feature_vector_reg=Conv(reg_kccs, 1, padding='same', activation=final_activation, name='Process_Parameter_Reg_output')(x)
process_parameter_reg=GlobalAveragePooling3D()(feature_vector_reg)

feature_vector_cla=Conv(categorical_kccs, 1, padding='same', activation=final_activation, name='Process_Parameter_Cla_output
process_parameter_cla=GlobalAveragePooling3D()(feature_vector_cla)

#feature_categorical=Flatten()(feature_vector)
#reg_output=tfp.layers.DenseFlipout(output_dimension, kernel_divergence_fn=k1_divergence_function)(process_parameter)

#Process Parameter Outputs
reg_distribution=tfp.layers.DistributionLambda(lambda t:tf.d.MultivariateNormalDiag(loc=t[...], :reg_kccs], scale_diag=aleatori
cla_distribution=Activation('sigmoid', name="classification_outputs")(process_parameter_cla)
#cla_distribution=tfp.layers.DenseFlipout(categorical_kccs, kernel_divergence_fn=k1_divergence_function, activation=tf.nn.sig

# Upsampling
for i in range(depth - 2, -1, -1):
    out_channel = 2**(i) * filter_root

    # long connection from down sampling path.
    long_connection = long_connection_store[str(i)]

    up1 = UpSampling(name="UpSampling{}_1".format(i))(x)
    up_conv1 = Conv(out_channel, 2, activation='relu', padding='same', name="upConvSam{}_1".format(i))(up1)

    attention_layer = attention_block(x=long_connection, g=up_conv1, inter_channel=out_channel // 4)
    # Concatenate.

    #up_conc = Concatenate(axis=-1, name="upConcatenate{}_1".format(i))([up_conv1, long_connection])
    up_conc = Concatenate(axis=-1, name="upConcatenate{}_1".format(i))([up_conv1, attention_layer])

    # Convolutions
    up_conv2 = Conv(out_channel, 3, padding='same', name="upConv{}_1".format(i))(up_conc)

    up_act1 = Activation(activation, name="upAct{}_1".format(i))(up_conv2)

    up_conv2 = Conv(out_channel, 3, padding='same', name="upConv{}_2".format(i))(up_act1)

    # Residual/Skip connection
    res = Conv(out_channel, kernel_size=1, padding='same', use_bias=False, name="upIdentity{}_1".format(i))(up_conc)

    resconnection = Add(name="upAdd{}_1".format(i))([res, up_conv2])

    x = Activation(activation, name="upAct{}_2".format(i))(resconnection)

```

Verification and Validation

Details of verification and validation of the model on an actual system can be found here: [Real System Implementation](#)

Benchmarking

Benchmarking of the model is done against various deep learning and machine learning approaches to highlight superiority.

Decoder Outputs

The segmentations outputs for the Bayesian 3D U-Net model enables estimation of dimensional quality of products in between stages and stations of the process.

Model Interpretability

3D Grad-weighted Class Activation Maps (3D Grad-CAMS) for each level of the encoder provides insights into the working of the model and integrate a measure of trust in all estimates.

Please cite work as:

Sinha, S., Glorieux, E., Franciosa, P., & Ceglarek, D. (2019, June). 3D convolutional neural networks to estimate assembly process p



@inproceedings(Sinha2019, author = {Sinha, Sumit and Glorieux, Emile and Franciosa, Pasquale and Ceglarek, Dariusz}, booktitle = {Multimodal Sensing: Technologies and Applications}, doi = {10.1117/12.2526062}, month = {jun}, pages = {10}, publisher = {SPIE}, title = {{3D convolutional neural networks to estimate assembly process parameters using 3D point-clouds}}, year = {2019} }



Data generation has been done using a Multi-fidelity CAE simulation software known as VRM :

Franciosa, P., Palit, A., Gerbino, S., & Ceglarek, D. (2019). A novel hybrid shell element formulation (QUAD+ and TRIA+): A benchmar



Collaboration:

Please contact [Sumit Sinha](#), [Dr Pasquale Franciosa](#) or [Prof Darek Ceglarek](#) in case of any clarifications or collaborative work with the [Digital Lifecycle Management](#) group at [WMG, University of Warwick](#)

Appendix C: Published Journal Paper 1

S. Sinha, P. Franciosa, and D. Ceglarek, "Object Shape Error Response using Bayesian 3D Convolutional Neural Networks for Assembly Systems with Compliant Parts," vol. 17, no. 10, pp. 6676-6686, IEEE Transactions on Industrial Informatics, 2021, DOI: 10.1109/TII.2020.3043226.

Object Shape Error Response Using Bayesian 3-D Convolutional Neural Networks for Assembly Systems With Compliant Parts

Sumit Sinha , Pasquale Franciosa , and Dariusz Ceglarek 

Abstract—This article proposes a novel object shape error response (OSER) approach to estimate the dimensional and geometric variation of assembled products and then, relate these to process parameters, which can be interpreted as root causes (RC) of the object shape defects. The OSER approach leverages Bayesian 3-D convolutional neural networks integrated with computer-aided engineering simulations for RC isolation. Compared with the existing methods, the proposed approach: 1) addresses a novel problem of applying deep learning for object shape error identification instead of object detection; 2) overcomes fundamental performance limitations of current linear approaches for RC analysis (RCA) of assembly systems that cannot be used on point cloud data; and 3) provides capabilities for unsolved challenges such as ill-conditioning, fault-multiplicity, RC prediction with uncertainty quantification, and learning at design phase when no measurement data are available. Comprehensive benchmarking with existing machine learning models demonstrates superior performance with $R^2 = 0.98$ and $MAE = 0.05$ mm, thus improving RCA capabilities by 29%.

Index Terms—Assembly, Bayesian deep learning, manufacturing, 3-D convolutional neural networks (CNNs).

I. INTRODUCTION

OBJECT shape error modeling and diagnosis are important enablers of Industry 4.0 and provide a transformative framework integrating facilitators such as big data, in-line 3-D scanners, robotics, and AI algorithms toward achieving near-zero-defect manufacturing. In this article, the proposed 3-D object shape error response (OSER) approach translates into estimating and discriminating between shape error patterns and linking them to manufacturing process parameters.

Manuscript received July 7, 2020; revised October 28, 2020; accepted November 24, 2020. Date of publication December 8, 2020; date of current version June 30, 2021. This work was supported in part by the U.K. EPSRC under Project EP/K019368/1: “Self-Resilient Reconfigurable Assembly Systems With In-Process Quality Improvement,” in part by the UKRI Open Access Block Grant, and in part by the WMG-IIT Scholarship. Paper no. TII-20-3286. (Corresponding author: Sumit Sinha.)

The authors are with the Digital Lifecycle Management Group, Warwick Manufacturing Group, University of Warwick, CV4 7AL Coventry, U.K. (e-mail: sumit.sinha.1@warwick.ac.uk; p.franciosa@warwick.ac.uk; d.j.ceglarek@warwick.ac.uk).

This article has supplementary material provided by the authors and color versions of one or more figures available at <https://doi.org/10.1109/TII.2020.3043226>.

Digital Object Identifier 10.1109/TII.2020.3043226

Estimating at first and then reducing or eliminating these error patterns ensures dimensional product quality (as defined by GD&T), which is a major challenge for industries such as automotive, aerospace, and shipbuilding. Two-thirds of the quality issues in the automotive and aerospace sectors are caused by dimensional variations [1]. The key goal is developing a root cause (RC) analysis (RCA) model that can identify the relationship between shape errors and manufacturing process parameters.

Past methods used to diagnose manufacturing dimensional quality faults are based on: 1) statistical estimation and 2) pattern matching-based approaches. These approaches have been shown to have limitations in their applicability to complex, high-dimensional, and nonlinear systems [2] as these used linear models between process parameters and measurements of product dimensional quality for both systems with rigid [3] and compliant parts [4]. Ceglarek *et al.* [5] used computer-aided design (CAD)-based variation patterns and a fault matching technique which combined principal component analysis (PCA) and pattern similarity for fault diagnosis. This work was later extended to include the effect of measurement noise [6] and then generalized for multistage assembly process using state-space model, stream-of-variation [7]. Jin *et al.* [8] used a Bayesian network approach for estimating fixture faults using all measured points. Bastani *et al.* [9] used a spatially correlated Bayesian learning algorithm for an underdetermined system by exploiting the spatial correlation of dimensional variation from various error sources. In summary, the aforementioned approaches are linear and are designed to work for relatively small number of measurement points on each manufactured part. This significantly limits the application of the methods for 3-D object shape error modeling and diagnosis in manufacturing. The 3-D shape error modeling and diagnosis used in manufacturing must have the capability to satisfy a number of requirements with respect to the following.

- 1) *High data dimensionality* of a batch of 3-D objects [10], which are defined by CAD (ideal parts) and point-clouds (nonideal parts) with millions of points for each part or subassembly.
- 2) *Nonlinearity* due to compliant parts being constrained by assembly fixtures and part-to-part interactions [11].
- 3) *Collinearities* due to many manufacturing systems being ill-conditioned [12] with error patterns of key process

This work is licensed under a Creative Commons Attribution 4.0 License. For more information, see <http://creativecommons.org/licenses/by/4.0/>

parameters being near parallel, thus, yielding widely discrepant results.

- 4) *High faults multiplicity* [13], [14] due to current near-zero-defects strategies requiring to take into consideration 6-sigma defects that lead to redefining defects from binary {0, 1}, i.e., fault/no-fault, to continuous $<0,1>$, i.e., the fault being measured as a level of variation with dynamically changing threshold of acceptance, which significantly increases fault multiplicity.
- 5) *Uncertainty quantification* in the RCA output, as the identified RC frequently leads to costly corrective actions [15]; it is crucial therefore to enhance the RCA model by an uncertainty estimation of the predictions.
- 6) *Dual data generation capability* by using metrology gages and multiphysics-simulator needed for RCA model training. As the RCA model needs to be trained on a very large number of fault scenarios, which cannot be generated via real systems, and the training needs to be done before the real assembly systems are ready for production, there is a strong need to generate data via high-fidelity multiphysics simulator for training RCA model. Then, the RCA model will use point-cloud data of real free-form surfaces obtained via robotic 3-D scanner when implemented in a real system.

This article will address the above requirements as follows.

- 1) Requirements 1)–4) by developing a 3-D deep learning approach. As markets get competitive in terms of product quality, production volume, and costs, manufacturers aim to leverage developments in the field of artificial intelligence. Deep neural networks have revolutionized data-intensive tasks that involve generating insights from high-dimensional input data [16], [17]. 2-D/3-D convolutional neural networks (CNNs) are known to perform well when spatial data such as depth images, point-clouds, mesh, and medical scans have to be analyzed for tasks such as control systems, object detection, video analysis, and cancer detection. Manufacturing is one of the major domains that has benefited from this development [18]. This article proposes a 3-D CNN architecture that enables the extraction of spatial features from point clouds and hence, models nonlinear relationships between features and process parameters. This approach has high performance for nonlinear and ill-conditioned systems having high fault multiplicity.
- 2) Requirement 5) by leveraging a Bayesian 3-D CNN-based approach. Recent developments in artificial intelligence cautions making real-life decisions based on point-estimates. As compared to traditional CNNs with deterministic weights, Bayesian CNNs leverage probability distributions over model weights and model outputs and enable quantification of predictive uncertainty, prevent overfitting, and require comparatively lesser data to train [19]. Successful applications of the above have been done in healthcare [20] and load forecasting [21]. Using such models enables segregation of the uncertainties into aleatoric and epistemic, the former quantifying the uncertainty due to uncontrollable factors such as system noise,

while the latter quantifies uncertainties due to model structure and insufficient training data [19]. The proposed approach estimates each model parameter as a distribution (epistemic uncertainty) while also modeling the output estimates as parameters of a multivariate distribution (aleatoric uncertainty). Such estimates involving different types of uncertainties in model predictions are crucial as these quantify when the model is “randomly guessing” as compared to making a confident prediction. Particularly within manufacturing environments, these uncertainty estimates integrate a degree of confidence within the estimates and hence, support the decision-maker in making cost-effective selection of corrective action(s) which can be quite costly.

- 3) Requirement 6) by making the developed Bayesian 3-D CNN from (2) compatible with point cloud data obtained via either multiphysics-simulator or 3-D scanners and leveraging the epistemic uncertainty estimates to perform intelligent *closed-loop training* and enable model convergence using a lesser number of training samples. In turn, this reduces total data generation and model training time. Since multiphysics-simulator is computationally expensive and generating each sample for assembly applications can be time intensive for high-fidelity simulations, it is crucial to reduce overall simulation time. The reduction in simulation time provided by leveraging the epistemic uncertainty estimates of Bayesian 3-D CNNs is significantly higher than the increased training time for Bayesian deep learning approaches. The approach developed in this article will utilize high-fidelity multiphysics-simulator of the assembly process, called variation response method (VRM) [22]. The VRM has the capability for, first, modeling and simulating the assembly process with compliant parts constrained by assembly fixtures and part-to-part interactions, and then, it enables high-fidelity point-cloud data generation of 3-D assembled products/subassemblies with error patterns as obtained under different sets of process parameters. The VRM model accuracy was verified and validated for various assembly processes [22]. Additionally, the approach is equipped to utilize data obtained from measurements using 3-D optical scanners. 3-D optical scanners enable real-time high-dimensional point cloud data extraction from manufacturing systems within short cycle times. This point cloud data can be postprocessed using alignment techniques to extract deviations for points, thus enabling dual data generation and integration.

In summary, the article develops a novel 3-D OSER approach in an effort to enable RCA within manufacturing systems using point cloud data. The proposed methodology integrates deep learning [which addresses requirements 1)–4)], Bayesian training enabled by Bayes-by-Backprop [23] and Flipout [24] [requirement 5)], and multiphysics-simulator to address requirement 6).

The key contributions of the article are as follows.

- 1) Propose a *3-D OSER methodology* based on a novel Bayesian 3-D CNN architecture: It builds on current

work done in the area of 3-D object detection [16] by expanding it to manufacturing systems where the key goal is not to detect the object but to estimate various shape error patterns present on the final object/product and relate these variation patterns to manufacturing process parameter variations within the system. To the best of our knowledge, this is the first article to propose an uncertainty enabled 3-D CNN-based deep learning model for RCA of assembly systems.

- 2) Propose a *closed-loop framework for training and deployment* of the Bayesian 3-D CNN model that leverages a computer-aided engineering (CAE) simulator known as VRM [22] to emulate the multistage assembly system. The VRM performs sampling, which leverages the epistemic uncertainty estimates of the Bayesian 3-D CNN thereby, reducing overall simulation and training time. Given that data availability within manufacturing systems is costly, scarce, and the data can be highly skewed, the VRM functions as a physics-based digital twin for generating augmented data that are close to the real system and can therefore, be used to train the proposed model. The trained model can then be leveraged for applications such as RCA of assembly systems using point cloud scans obtained from 3-D scanners.
- 3) Verify and validate the methodology on an *industrial automotive door assembly process* made of compliant parts.
- 4) *Benchmark 3-D OSER methodology* against three categories of methods that can be leveraged to estimate the dimensional and geometric variation of assembled products namely: a) current linear state-of-the-art RCA models; b) machine learning models in a multi-output regression setting; and c) deep learning models such as various types of CNNs and fully connected networks to highlight the performance and the ability to fulfill the aforementioned six requirements.

The rest of the article is organized as follows. Section II formulates the object shape error estimation problem, presents the proposed Bayesian 3-D CNN architecture, the steps involved in architecture optimization, and the overall steps required to train and deploy the model. Section III presents the industrial case study. Finally, Section IV concludes this article.

II. METHODOLOGY

A. Object Shape Error Estimation in Manufacturing

Multistage assembly systems can be mathematically expressed as a state-space model where different states correspond to different stages of the manufacturing system [3]. The input is an object (set of parts to be assembled) entering the assembly process. Within the process, object shape errors can be introduced in any of the stages due to one or multiple variations in the process parameters and are further propagated through the stages (see Fig. 1). Any object o at its design nominal shape is characterized by a set of nominal points $P_o = \{p_{ok}\}$, $k = 1, \dots, n_o$, where p_{ok} is a vector consisting of the x -, y -, and z -coordinates of the k th input point and n_o represents the total

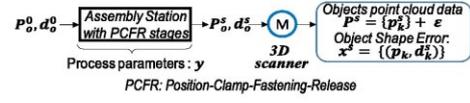


Fig. 1. Object shape error propagation in assembly systems.

number of points on object o . The object here represents a single subassembly which is assembled in a single station, which can be understood as a collective reference to all parts used in this assembly station. In practice, the points correspond to mesh nodes in the computer-aided design model of the object when considering CAE simulations and to actual points within the point cloud when considering the 3-D scan of the object. $d_o = \{d_{ok}\}$ denotes the deviation of each point k after the nominal object o has gone through different stages of the process, d_{ok} is a vector comprised of deviations of each point in x -, y -, and z -axes on object o . An assumption made in this article is that the assembly process has a single station which includes multiple stages $s = 1, \dots, 4$ involving objects/parts—positioning (P), clamping (C), fastening (F), and release (R). Stage $s = 0$ is used to represent the incoming parts that includes deviations from the previous processes, such as part fabrication. As the object o goes through multiple stages, the set of points are represented as P_o^s , while d_o^s represents the deviations.

As the main goal of this article is object shape error estimation, the article extends the problem formulation in object detection, which only considers the set of points $\{p_{ok}\}$ [17], by including deviations for each point $\{d_{ok}\}$ as additional features. This adds the required discriminative ability in the data, hence, enabling object shape error estimation. Thus, the object shape error for object o after stage s can be represented as

$$x_o^s = (P_o^s, d_o^s). \quad (1)$$

On the other hand, the set of all process parameters across all stages are denoted by y where $y = \{y_1, \dots, y_h\}$, h denotes the total number of process parameters. The deviation of points at each stage s for object o can be expressed as the sum of all deviations accumulated in all stages from stage 0 up to stage s

$$d_o^s = \sum_{j=0}^s d_o^j \quad (2)$$

where d_o^0 represents the shape error of incoming object o caused by upstream manufacturing processes. After each stage s , the actual points of the object o with error can be written as

$$P_o^s = P_o + d_o^s. \quad (3)$$

At the end of the final assembly stage $s = 4$, the object shape error data for the assembly $P^{s=4}$ is collected and decomposed into the nominal points P and their deviations $d^{s=4}$ by using alignment techniques [10], where $P^{s=4}$, $d^{s=4}$ are now a collective reference to the set of all incoming objects that have been assembled. The measurement system error ϵ is considered to be negligible ($\epsilon \approx 0$). The object with errors is represented as a point cloud of nonideal parts

$$x^s = \{(p_k, d_k^s)\} = (P, d^s) \quad (4)$$

where d_k^s can be considered as features at each point p_k .

The aim of the Bayesian 3-D CNN model training is to learn assembly process transfer function $f(\cdot)$ (equivalent to state transition matrix in [7]). The function $f(\cdot)$ is parametrized by weights and biases of a CNN that can accurately estimate the process parameters y given the point cloud data of nonideal x^s parts collected from the system

$$y = f(x^s). \quad (5)$$

The high accuracy of the 3-D CNN in estimating all assembly process parameters y provides the underlined capability of the OSER approach for high RC isolability. Essentially, within assembly systems, RCs are estimated as a subset of the estimated process parameters

$$RC \subseteq y. \quad (6)$$

Based on the requirements and the production phase of the assembly system, the exact definition of an RC may differ but the key requirement to conduct RCA under any definition is to accurately estimate all process parameters y . Hence, the proposed OSER approach aims to do the aforementioned by estimating $f(\cdot)$ as specified in (5).

B. Three-Dimensional Object Shape Error Voxelization

In the presented OSER approach, the simulation output represented as mesh or point cloud data $\{(p_k, d_k^s)\}$ (4) is transformed to voxel grids $\{V_{u,v,w}\}$ with discrete voxel coordinates (u, v, w) in the following way—for all points $p_k = (x_k, y_k, z_k)$ that fall within a voxel grid $\{V_{u,v,w}\}$, the maximum value of $d_k = (\bar{x}_k, \bar{y}_k, \bar{z}_k)$ characterizes the features of the corresponding voxel grid and is represented as $\{V_{u,v,w,d}\}$. The voxelization techniques used in object detection [17] are applied to construct the initial voxel structure of the object and for each unique object, the voxel features are characterized by the shape error d_k . The key difference is that in object detection, voxel grids are characterized by either binary voxels or voxels containing RGB values for each point instead of real values of shape error d_k , as in the OSER approach. Although binary voxels, traditionally used in object detection retain the spatial structure, the granularity of voxelization required to discriminate between minor differences in the shape error will make the problem computationally infeasible and hence, limit performance. In the proposed approach, the nominal object is voxelized and each voxel is characterized by real values of the shape error d_k . This is critical in representing the geometric variations with the required granularity for effective RCA. This efficiently retains all information about the spatial structure of the object as well as the components of object shape errors. Given the alignment ensures a fixed orientation, there is no need for data augmentation to achieve rotation invariance.

C. Uncertainty Estimation

Given the uncertainties of the system and the availability of only a limited dataset, a deterministic estimate of function $f(\cdot)$, as shown in (5), is not feasible. Hence, by leveraging Bayesian inference, a prior distribution can be allocated over the space of

possible functions $p(f)$, which represents a prior belief of the possible functions $f(\cdot)$. Given a dataset, a likelihood $p(y|f, x^s)$ is defined to model the function from which the observation is generated, and hence, given a dataset (x^s, y) , the posterior distribution over the functions $p(f|x^s, y)$ can be inferred. The function is characterized by model parameters ω represented by f^ω (weights and biases for neural networks) and the posterior over the function can be inferred by estimating the posterior over the parameters ω . In Bayesian neural networks, this is achieved through Bayes-by-Backprop [23] and Flipout [24]. Given a dataset the posterior can be written as

$$p(w|x_s) = p(y|x_s, w), p(w)/p(y|x_s) \quad (7)$$

For complex models such as deep neural networks, it is not analytically possible to infer the true posterior for all model parameters $p(w|x_s, y)$; hence, an approximating variational distribution $q_\theta(w)$ parameterized by θ , such as normal distribution, is used to approximate the posterior. This approach is known as variational inference [25]. The approximating distribution should be as close as possible to the true posterior, which is achieved by minimizing the Kullback–Leibler (KL) divergence with respect to θ

$$KL(q_\theta(w)||p(w|x_s)) = \int q_\theta(w) \log q_\theta(w)/p(w|x_s) d\omega. \quad (8)$$

Using the estimated variational distribution $q_\theta^*(\omega)$, the process parameter distribution quantifying the uncertainties for a new data point x^{**} can be obtained using

$$p(y^*|x^{**}, x^s, y) \approx \int p(y^*|x^{**}, \omega) p(w|x^s, y) d\omega =: q_\theta^*(y^*|x^{**}). \quad (9)$$

D. Bayesian 3-D CNN Model Architecture

Building on the work done on voxel-based approaches for 3-D object detection such as VoxNet [17], the research proposes a Bayesian 3-D CNN architecture to enable object shape error estimation. The 3-D convolutions aggregate features from the input, which are then utilized by the fully connected layers and mapped to process parameters. The model consists of three 3-D convolutional Flipout layers, a 3-D max-pooling layer, followed by three fully connected Flipout layers; the final layer estimates parameters of the predictive distribution for all process parameters. The convolution can be represented as

$$v_{ab}^{xyz} = ReLU \left(\beta_{ab} + \sum_m \sum_{p=0}^{P_a-L_a} \sum_{q=0}^{Q_a-M_a} \sum_{r=0}^{R_a-N_a} w_{ab}^{pqr} v_{(i-1)m}^{(x+p)(y+q)(z+r)} \right) \quad (10)$$

where v_{ab}^{xyz} represents the layer output value at position (x, y, z) in the a th layer and b th feature map. ReLU is the rectified linear unit activation function [26]. β_{ab} represents the bias; m represents the number of filters from the previous layer; (P_a, Q_a, R_a) and (L_a, M_a, N_a) represent the kernel dimensions and stride lengths in the three directions, respectively; w_{ab}^{pqr} represents the weights of the connections. The convolution operation as in (10) is done consecutively for the three convolutional

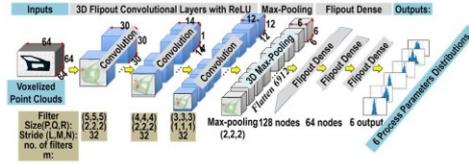


Fig. 2. Bayesian 3-D CNN model architecture of the OSER method.

layers. In 3-D max-pooling operation, the resolution of the feature map is reduced by taking the maximum value of the local neighborhood of the layer outputs. Given the Bayesian framework, each parameter of the Bayesian 3-D CNN model follows a distribution. In the case of neural networks, it is not feasible to assign informative priors; hence, noninformative prior distributions are placed over the model parameters. Each parameter is approximated using variational inference approach assuming that the posterior follows a normal distribution. The overall model has 1 997 286 trainable parameters. Output nodes have linear activation units. Fig. 2 shows the proposed Bayesian 3-D CNN model architecture with annotated hyperparameters.

E. Architecture Selection and Optimization

Hyperparameter optimization for Bayesian 3-D CNNs is done to maximize performance and eliminate architectures that are more likely to overfit. As this is computationally intensive, in order to perform optimization in a computationally feasible manner, the following steps were involved.

Step 1 – Set baseline: VoxNet [16], which is a 3-D CNN architecture used for object detection consisting of two 3-D convolutional layers, one max-pooling layer, and two fully connected, is set as the baseline. A dataset consisting of 1500 samples is generated to conduct k -fold cross-validation ($k = 6$). The hyperparameters are split into two categories: Category one consists of the number of convolutional layers $N_C = \{2, 3, 4\}$ and number of dense layers $N_D = \{1, 2, 3\}$ and category two consisted of the number of filters in each 3-D convolutional layer, filter size for each 3-D convolutional layer, and number of hidden units in each dense layer.

Step 2 – Grid search for category one hyperparameters: In this step, grid search for category one hyperparameters are conducted and each selection is evaluated using k -fold cross-validation (see Fig. 3). For computational feasibility, category two hyperparameters are kept constant and equal to the VoxNet architecture values. $N_C = 3$ and $N_D = 3$ were obtained as the optimal hyperparameters having the minimum cross-validation mean absolute error (MAE) average of 0.08 mm.

Step 3 – Hyperband for category two hyperparameters: The optimal values for category one hyperparameters are fixed and further hyperband [27] is leveraged to obtain the optimal values for category two hyperparameters given its ability to speed up the random search process through adaptive resource allocation and early stopping.

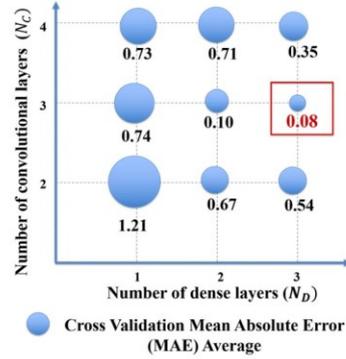


Fig. 3. Grid search for category one hyperparameters.

TABLE I
OBJECT DETECTION AND OSER COMPARISON

Object Detection (VoxNet)	Object Shape Error Detection (OSER)	Rationale
Single-channel binary occupancy input voxel of dimension $32 \times 32 \times 32$	Multi-channel real-valued object shape error input voxel of dimension $64 \times 64 \times 64 \times 3$	High data dimensionality of shape error (P, d^3 [requirement 1])
Two 3D Convolutional layers, one dense layer and categorical output layer	Three 3D Convolutional layers, three dense layers and real-valued output layer	Increased model capabilities to handle non-linearity, collinearity and high fault multiplicity [requirements 2–4])
Deterministic Layer	Bayesian Flipout Layer	Uncertainty quantification for corrective actions and closed-loop sampling for faster convergence [requirements 5 and 6])

Step 4 – Deterministic to Bayesian model: The final step includes replacing the deterministic layer with Bayesian Flipout layers and then training using Bayes-by-Backprop. Various learning rates and prior distributions for the model weights were tested. Standard normal distribution provided the best balance between weight initialization and weight exploration, which was inferred by conducting an uncertainty versus error calibration study. The training hyperparameters that provided the best uncertainty calibration and ensured that the model performance was greater than or equal to the deterministic counterpart were selected as the final Bayesian 3-D CNN architecture training hyperparameters. The key changes from the baseline architecture of object detection that enable the fulfillment of the aforementioned six requirements are summarized in Table I.

F. Model Training and Deployment

Training of the model is done in a closed-loop framework using data generated by VRM. The key motivation behind using a closed-loop framework as opposed to an open-loop framework

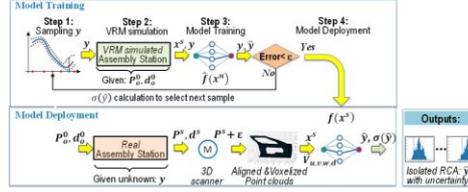


Fig. 4. Model training and deployment framework.

is to minimize the bottle-neck computation, i.e., multiphysics simulation using the VRM model. Although this increases the number of training iterations, nonetheless, the overall time of VRM simulation and training is significantly reduced, as fewer samples need to be generated. The key steps of the proposed framework are summarized as follows (see Fig. 4).

Step 1 – Sampling: Process parameters y are sampled from the allowable ranges. Latin Hypercube Sampling [28] is used to generate initial process parameter sample values given it distributes samples optimally across the h -dimensional process parameter space by stratifying the possible ranges. The consecutive sets of samples are generated using Monte Carlo (MC) sampling based on the uncertainty $\sigma(\hat{y})$ of the model.

Step 2 – VRM simulation: The samples are used as input to the VRM to simulate the assembly process and generate the output mesh from which the point cloud and deviations of each point are extracted after the desired stage s of the assembly system, as in (4), $x^s = \{(p_k, d_k^s)\}$.

Step 3 – Model training: The point cloud and deviation data of object shape errors along with the respective process parameters ($x^s = \{(p_k, d_k^s)\}, y$) are used for model training. Note that x^s is voxelized $\{V_{u,v,w,d}\}$ before it is used for training. Bayes-by-Backprop and Flipout are applied for model training. The loss function optimized while training comprises of the sum of KL divergence for each layer (9) and the negative log-likelihood (11) of the predictive distribution

$$-\ln L = 1/2[\ln(|\Sigma|) + (y - \mu)^T \Sigma^{-1} (y - \mu) + h \ln(2\pi)]. \quad (11)$$

The KL divergence term quantifies the divergence between the standard normal prior and the learnt posterior and hence, prevents overfitting by penalizing weights for diverging from the prior. Group normalization [29] with four groups is used after each convolutional layer. This also prevents overfitting and accounts for the small minibatch size due to GPU memory size constraints and aids in stabilizing the training process. Weights of the network are initialized using normal initializer [30]. The Adam method for stochastic optimization was used to optimize the loss function while training [31]. The initial learning rate is fine-tuned to $\alpha = 0.0005$ and monotonic KL annealing was leveraged to ensure the model initially learns the object shape error and process parameter relations before applying the KL

penalty for uncertainty quantification. The learning rate fine-tuning, monotonic KL annealing, and ReLU activations prevent gradient vanishing. The predictive distribution is modeled as a multivariate normal \mathcal{N}_h with h components (the same number of components as the number of process parameters h), $y \sim \mathcal{N}_h(\mu, \Sigma)$ where each component corresponds to a process parameter; hence, the mean across all components of the multivariate distribution corresponds to the set of process parameters y . The distribution is assumed to have a diagonal covariance matrix Σ . The scale parameters in the diagonal are assumed to be fixed since the noise has been assumed to be negligible.

After each iteration of training, the model is evaluated on the validation set. For evaluation, MC sampling from the model is done and the sample means \bar{y} and standard deviations (SDS) $\sigma(\hat{y})$ are estimated for each process parameter. $\sigma(\hat{y})$ represents the epistemic uncertainty while the fixed scale parameters of the predictive distribution represent the known aleatoric uncertainty [19]. Given the assumption of negligible measurement noise, aleatoric uncertainty is considered to be negligible and hence, the overall uncertainty in the prediction can be assumed to be equal to epistemic uncertainty $\sigma(\hat{y})$. This uncertainty is used for sampling in the next iteration.

MAE between the model estimates $\hat{y} = \bar{y}$ and actual value y across all process parameters h (12) are used as the metrics for model performance evaluation given the ease of interpretation and given that the model outputs are continuous and real valued. Training is stopped when MAE is below the required threshold ϵ . The threshold value for this metric is determined based on the quality requirements for a specific product as required by design tolerances and the accuracy of the measurement system. The model is trained within the measurement system accuracy. For example, automotive body assembly process tolerances are within $[-1 \text{ mm}, 1 \text{ mm}]$, and the 3-D optical scanner used has a repeatability of 0.05 mm and accuracy within 0.15 mm

$$\text{MAE} = \sum_h |y - \hat{y}|/h. \quad (12)$$

Step 4 – Model deployment: After training, the model can be deployed within an actual system. The data collected from the 3-D scanner P^s is aligned to obtain point cloud and deviations $x^s = \{(p_k, d_k^s)\}$ and then, voxelized $V_{u,v,w,d}$ before it can be given to the trained model for conducting RCA inference. Inferring estimates the process parameters for a given x^s (5) using MC sampling from the trained model. Using these samples, process parameters (distribution mean) \bar{y} and their uncertainty (distribution SD) $\sigma(\hat{y})$ can be estimated. The sample mean \bar{y} is considered as the model estimate \hat{y} , while $\sigma(\hat{y})$ quantifies the uncertainty. Further, the RCs can be inferred as a subset of \hat{y} (6). The work has been implemented using Python 3.7 and TensorFlow - GPU 2.0 [32] and TensorFlow-Probability 0.8. A python library, Bayesian Deep Learning for Manufacturing [33] has been developed to validate and replicate the results of the methodology. For this article, both, the data generation and evaluation of the OSER methodology have been done using

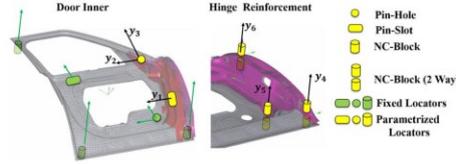


Fig. 5. Assembly process parameters.

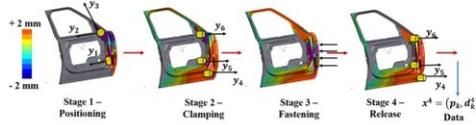


Fig. 6. PCFR stages of the assembly process.

VRM. Two Nvidia Tesla V100 32 GB GPUs are used for model training and deployment.

III. CASE STUDY

A. Assembly Setup

For verification and validation of the proposed OSER approach, an automotive assembly of two components, namely, the door inner and hinge reinforcement are selected. The assembly setup and parameters are shown in Fig. 5. The assembly process is controlled by the six ($h = 6$) parametrized process parameters y_1, y_2, \dots, y_6 (depicted using yellow symbols in Fig. 4). Assembly parameters such as pin-hole, pin-slot, and NC blocks for the door inner are considered constant (depicted using green symbols in Fig. 5) and are not parameterized. Data are collected after stage $s = 4$. The point cloud is characterized by $n = 10841$ points, which are preprocessed and voxelized to $(u, v, w) = (64, 64, 64)$ voxel grids. The deviation features d include deviations in all directions for all points $(\tilde{x}_k, \tilde{y}_k, \tilde{z}_k)$. The assembly consists of four stages (see Fig. 6): Stage 1 involves positioning (i) the door inner on the pin-hole, pin-slot, and the three NC blocks (not parameterized; marked in green in Fig. 1) and (ii) hinge reinforcement using the pin-slot (y_1), pin-hole (y_2, y_3); Stage 2 comprises of clamping two parts together using three NC blocks with clamps (y_4, y_5 and y_6); Stage 3 involves fastening/joining the two parts using self-piercing riveting (SPR); and finally, Stage 4 involves releasing the clamps (y_4, y_5 and y_6) after the fastening is completed. The training range for all process parameters is $[-1 \text{ mm}, 1 \text{ mm}]$ while the validation range is $[-2 \text{ mm}, 2 \text{ mm}]$. Point cloud and deviation data $\{(p_k, d_k^s)\}$ are collected after release, i.e., Stage 4 (see Fig. 5). The data are voxelized $\{V_{64,64,64,3}\}$ and used as model input and the process parameters y_1, y_2, \dots, y_6 are used as model outputs.

After starting with 200 initial samples for model training, 200 samples are adaptively added during each iteration of the closed-loop training based on the uncertainty estimates, and the

model is trained on the combined dataset including all previous samplings to ensure that there is no catastrophic forgetting (using 200 samples provided an optimal trade-off between VRM simulation time and model training time). These samples and outputs are used for training the Bayesian 3-D CNN model. The diagonal scale parameters for all process parameters in the covariance matrix are fixed at 0.001. A validation set of 300 samples is generated within the validation range, and after each iteration, the trained model is evaluated on the validation set. During the evaluation for each of the 300 samples, 1000 MC samples are drawn from the trained model. The sample means are considered as the estimate for the process parameters while the sample SDs quantify the uncertainty for each process parameter for the given sample. RCs can be inferred from the process parameter estimates. The closed-loop training is stopped when average MAE across all process parameters for the validation set is below the threshold which is selected to be 0.05 mm for automotive assembly applications as the impact of variations less than 0.05 mm is not detectable by the 3-D scanner. After this, the model is ready for deployment with measurement data collected from 3-D optical scanners, followed by alignment and voxelization. For each measurement, MC samples from the trained Bayesian 3-D CNN model can be drawn to estimate process parameter mean and SDs (uncertainty). Measurement data collection is done using WLS400A mounted on an ABB robot.

In summary, the industrial assembly process selected for case study consists of the following.

- 1) High dimensionality point cloud (10841 points).
- 2) Nonlinearity, as induced by fixturing ($N-2-1$, where $N = 6$), two compliant parts, and part-to-part interactions (door inner to hinge reinforcement).
- 3) Collinearity induced by fixturing, as locators y_4, y_5 , and y_6 are within 5° of collinearity (-3° to 2° deviation from y -axis).
- 4) High fault multiplicity, as we take into consideration 6-sigma defects at the level of variation within 3-D scanner accuracy ($<0.05 \text{ mm}$) that significantly increases fault multiplicity from zero to six process parameters manifesting errors (100% fault multiplicity).

The door assembly requirements are: 1) *product*: Design tolerances of door assembly: $<-1.0, 1.0> [\text{mm}]$; 2) *process*: Fixturing calibration and commissioning is achieved within $<-0.1; 0.1> [\text{mm}]$; and 3) *shape error detection*: Using the 3-D optical scanner for measurement.

Key performance indicators (KPIs) used for assessment of the results are as follows: 1) MAE $<0.05 \text{ mm}$ and 2) $R^2 > 0.95$ for the model to have the capability to explain more than 95% variance in the process parameters under the assembly system requirements 2)–4).

B. Results

The KPIs of model performance are summarized for all y_1, \dots, y_6 in Fig. 7. The model convergence is shown in Fig. 8.

The model converges with average MAE across all process parameters equal to 0.05 (below the required threshold) and

TABLE II
BENCHMARKING RESULTS

	Models	Requirement 1)	Requirement 2)-4)				Requirement 5)	Requirement 6)			
			Accuracy (MAE)		Goodness-of-fit (R^2)			Sampling	CAE Simulations Time (minutes)	Model Training Time (minutes)	Total Training Time (minutes)
			Mean	SD	Mean	SD					
Proposed	OSER (Bayesian 3D CNN)	3D Shape Error Voxelization	0.05	0.03	0.98	0.01	Aleatoric & Epistemic Uncertainty	Epistemic uncertainty sampling (2000 samples)	4400	424	4824
	OSER (3D CNN)	3D Shape Error Voxelization	0.05	0.01	0.98	0.009	No	Random Sampling (4000 samples)	8800	268	9068
Deep Learning	Multi-View 2D CNNs (MV-CNNs)	6 face projection and 2D gridding	0.08	0.02	0.94	0.01	No	Random Sampling (4000 samples)	8800	321	9121
	Depth Based 2D CNNs	1 face projection and 2D gridding	0.12	0.04	0.93	0.02	No	Random Sampling (3000 samples)	6600	248	6848
	Deep Dense Neural Networks	Flattening	0.28	0.09	0.91	0.07	No	Random Sampling (5000 samples)	11000	358	11358
Machine Learning	Gradient Boosted Trees	PCA	0.26	0.08	0.93	0.08	No	Random Sampling (3000 samples)	6600	120	6720
	Random Forests	PCA	0.29	0.09	0.92	0.08	No	Random Sampling (3000 samples)	6600	136	6736
	Support Vector Regression	PCA	0.38	0.09	0.85	0.1	No	Random Sampling (2500 samples)	5500	180	5680
Currently used linear approaches	Regularized Linear Regression	PCA	0.41	0.01	0.76	0.01	No	Random Sampling (1600 samples)	3300	10	3310

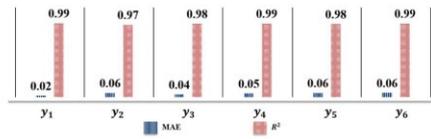
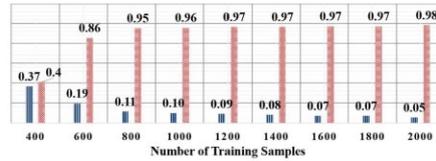
Fig. 7. MAE and R^2 across all process parameters.

Fig. 8. Bayesian 3-D CNN OSER convergence.

average R^2 equal to 0.98 after ten iterations of closed-loop training, which included a total of 2000 samples being generated adaptively. For validation purposes, this study trained both Bayesian 3-D CNN and a deterministic version of the model, i.e., 3-D CNN with the same architecture as in Fig. 2.

C. Benchmarking and Discussion

The benchmarking analysis is conducted by using the six requirements as listed in Section I. The case study and results along with analysis of collinearity, multiplicity, and uncertainty are used to demonstrate the capabilities of the proposed approach to fulfill the aforementioned requirements.

The benchmarking analysis of the proposed 3-D OSER approach is discussed on two levels.

1) *OSER Versus Currently Used Approaches at Production Phase When Point Cloud Data are Available*: The benchmarking is conducted for two scenarios: 1) RCA and 2) RCA with uncertainty quantification.

a) *RCA*: As discussed in Section I, the state-of-the-art models used for assembly process RCA, such as [9] and [34], are linear and can be classified as regularized linear regression approaches (see Table II). Hence, their upper limit performance can be estimated by using regularized linear regression on all point deviations d within the point cloud. They also use a limited number of sampled points from the point cloud on a single part (less than 100 out of >10000), which additionally limit their performance for assembly processes. The OSER methodology validation against the six requirements as presented in Section I is as follows. Requirement 1) is fulfilled by the proposed voxelization approach which ensures that irrespective of the dimensionality of the point cloud, it is transformed into a sparse tensor of dimensions (64, 64, 64, 3) which preserves information in terms of the object spatial structure and point deviation features. This also enables the application of the OSER-based models that require a regular data structure as input. Second, the model performance of the state-of-the-art regularized linear regression approaches is at MAE = 0.41 mm and $R^2 = 0.76$ (see Table II), which is unsatisfactory as compared to the required MAE < 0.05, $R^2 > 0.95$. This is because the regularized linear regression model can explain only the linear variance in the system. By comparison, the proposed OSER model demonstrates good performance at MAE = 0.05, $R^2 = 0.98$, hence fulfilling requirements 2)–4). Fig. 9 compares the performance of regularized linear regression (i.e., upper limit for state-of-the-art approaches) with the proposed OSER approach under different levels of fault multiplicity and collinearity. For example, in scenarios 1–3 (fault multiplicity up to 50%), both approaches

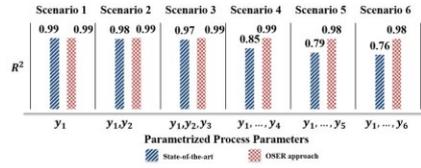


Fig. 9. Performance under different levels of fault collinearity and multiplicity (from single fault y_1 ; and two-fault y_1, y_2 ; ... up to all parameters being simultaneously at fault y_1, y_2, \dots, y_6).

have similar performance. However, in scenarios 4–6, as the fault multiplicity increases to 4, 5, and 6 parameters being simultaneously at fault, i.e., 100% of parameters, and also induced by the designed collinear relation between process parameters and input, the performance of the linear model decreases, while the OSER approach exhibits performance above the required threshold ($R^2 > 0.95$).

The benchmarking also comprehensively assesses the OSER against existing deep learning and machine learning techniques [35] in ways that are not currently used for RCA of assembly processes (see Table II). This article implemented these techniques and applied them for the aforementioned case study. CNN-based deep learning methods were selected as they retain spatial information while learning which is essential for object shape error estimation. Each model is compared in its ability to fulfill the aforementioned six requirements. Table II shows the implemented benchmark approaches and the results. The multiview 2-D CNN (MV-CNN) [36] approach considers six 2-D projections of the object shape error. Gridding is performed on each projection. Each projection has an input dimension of $64 \times 64 \times 3$. Each of the six projections is given as input to the MV-CNN consisting of six heads. These are pooled before the fully connected layers. Depth-based CNN [37] considers a single projection along the y -axis with dimension $64 \times 64 \times 4$. The first three channels consist of the shape error (d) while the fourth channel consists of the y -coordinate of the nominal point-cloud P_o . Both 2-D CNN-based approaches have same hyperparameters as the OSER (but only considering 2-D convolutions and 2-D max-pooling). The deep dense neural network is given the flattened vector of shape error $d = \text{Flatten}\{(\tilde{x}_k, \tilde{y}_k, \tilde{z}_k)\}$ as input. It consists of two hidden layers (1024, 512 nodes, respectively) with ReLU activations and a linear output layer with six nodes. All machine learning models take a transformed input of $d = \text{Flatten}\{(\tilde{x}_k, \tilde{y}_k, \tilde{z}_k)\}$ as input. PCA is used for the transformation and reduced features explaining 99% of the variance are retained. Comparison for requirement 1) is done on the basis of transformation used for input. Shape error voxelization retains information pertaining to the 3-D structure and shape error features, while projection retains only 2-D spatial structure. Flattening eliminates all information related to the spatial structure while PCA also eliminates information explaining 1% of the variance. Comparison for requirements 2)–4) are done on performance attributes, namely, accuracy (MAE) and goodness-of-fit (R^2). Comparison for requirement 5) is done

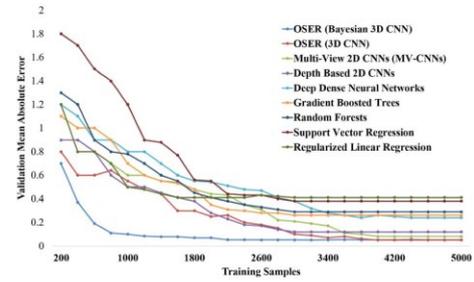


Fig. 10. Convergence comparison for all benchmarking models.

on the ability to quantify and segregate uncertainties. Finally, comparison for requirement 6) is done on overall training time, which is inclusive of the CAE simulation time and model training time. Although the proposed model has higher model training time, the overall training time is significantly lesser due to the ability to leverage the epistemic uncertainty to generate informative samples leading to faster convergence with only ~ 2000 samples. All other models are trained using random sampling until convergence. Fig. 10 summarizes the convergence of the entire set of benchmarking models. The hyperparameters of the machine learning models were optimized using grid search. For statistical quantification of accuracy and goodness-of-fit, 20 runs of training and testing are conducted using a set of 4000 randomly sampled data points for training and 300 for validation within the validation range. The mean and SD for each model, averaged across six process parameters, have been reported. The model performance of the proposed OSER model is significantly better in terms of accuracy and goodness-of-fit. Result from ANOVA followed by *post hoc* Tukey-HSD test at 95% significance level considering two sources of variations (model type and process parameter) showed the differences to be statistically significant. This comes at the expense of increased model complexity.

b) RCA with uncertainty quantification: As discussed in Section I, the identified RCA frequently leads to costly corrective actions conducted in the manufacturing environment; therefore, it is crucial, especially for 6-sigma faults to have decision-driven RCA directed toward informing choices by uncertainty quantification of solving problems. The OSER methodology provides the SD of the predicted process parameter distributions $\sigma(\hat{y})$ that quantifies this uncertainty hence, fulfilling requirement 5). Although the performance of the OSER with 3-D CNN and OSER with Bayesian 3-D CNN models are similar, the latter can quantify and segregate the aleatoric and epistemic uncertainty while estimating the process parameters. To demonstrate the capability of the model in quantifying the uncertainty on unseen samples, evaluation is done on 500 samples within the training range [-1 mm, 1 mm] and 500 samples outside of the training range [-2 mm, 2 mm]. The SD across all observations has been averaged and compared for each process

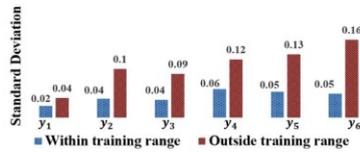


Fig. 11. Process parameter distribution SDs.

parameter y_1, \dots, y_6 . Results are shown in Fig. 11. Additionally, the epistemic uncertainty estimates enable closed-loop training, reducing overall training time.

2) *OSER Versus Approaches at Design Phase When No Point Cloud Data are Available*: In manufacturing environments, the availability of a comprehensive dataset inclusive of all fault scenarios is not feasible; hence, augmenting the dataset with high-fidelity multiphysics simulation enables training and deployment of deep learning approaches during the design phase of a new product/production system introduction. Given that the proposed OSER approach transforms the simulation mesh nodes output and scanned point cloud output to the same voxelized shape error that is compatible with 3-D CNN, it enables this integration, hence fulfilling requirement 6). This provides the capability for modeling and simulation of the assembly process and conducting system diagnosability and resilience analysis. Currently, none of the approaches provide this capability for object shaper error RCA at the design phase.

IV. CONCLUSION

This article presented an OSER approach which is relevant to manufacturing industries where dimensional and geometric variations can be quantified as object shape errors. This is also relevant to areas such as robotics, computer-aided detection, stamping, machining, and additive manufacturing where RCA of dimensional variations translates to estimating object shape error patterns and relating them to process parameters. Transfer learning can be leveraged for application in these domains with exponentially lesser training samples [18], a focus for future work. The proposed approach leveraged a Bayesian 3-D CNN model trained within a closed-loop framework using a multiphysics simulation (VRM) model, to estimate shape errors and relate them to process parameters while quantifying uncertainty. This could then be deployed on real data collected from 3-D surface scanners and thereby, enabling more effective and efficient decision-making for control and correction of manufacturing systems. The approach was benchmarked against state-of-the-art assembly RCA models and other machine learning models to highlight statistically significant better model performance while fulfilling the manufacturing system design requirements. Leveraging such automated RCA models ensured early estimation and elimination of process variations before they become defects, which could improve the quality and productivity of the system by reducing scrap and machine downtime. This also eliminated the need for trial-and-error approaches for RCA, which is often ineffective and inefficient. Future work aims

to explore scaling up the work to multistation assembly systems. Various encoder-decoder-based CNN architectures, such as U-Net [38] and Pointnet [16], that enable process parameter estimation for a heterogeneous set of process parameters, i.e., continuous and categorical as well as enable object shape error estimation in-between stages/stations could be explored to comprehensively perform RCA on multistation systems. Approaches for uncertainty guided continual learning will also be explored that enable transfer learning to different manufacturing systems while simultaneously retaining knowledge of previous assembly systems. The future work also aims to develop a life-long continual learning approach leveraging Bayesian 3-D CNNs, which is crucial for continuously changing manufacturing environments.

REFERENCES

- [1] D. Ceglarek and J. Shi, "Dimensional variation reduction for automotive body assembly," *Manuf. Rev.*, vol. 8, pp. 139–154, 1995.
- [2] S. Sinha *et al.*, "3D convolutional neural networks to estimate assembly process parameters using 3D point-clouds," *Proc. SPIE*, vol. 11059, 2019, Art. no. 110590B.
- [3] W. Huang *et al.*, "Stream-of-variation (SOVA) modeling II: A generic 3D variation model for rigid body assembly in multi station assembly processes," *J. Manuf. Sci. Eng.*, vol. 129, no. 4, pp. 323–333, 2006.
- [4] H. Wang and D. Ceglarek, "Variation propagation modeling and analysis at preliminary design phase of multi-station assembly system," *Assem. Autom.*, vol. 29, no. 2, pp. 154–166, 2009.
- [5] D. Ceglarek and J. Shi, "Fixture failure diagnosis for autobody assembly using pattern recognition," *J. Eng. Ind.*, vol. 118, pp. 55–66, 1996.
- [6] D. Ceglarek and J. Shi, "Fixture failure diagnosis for sheet metal assembly with consideration of measurement noise," *J. Manuf. Sci. Eng.*, vol. 121, no. 4, pp. 771–777, Nov. 1999.
- [7] Y. Ding, D. Ceglarek, and J. Shi, "Fault diagnosis of multistage manufacturing processes by using state space approach," *J. Manuf. Sci. Eng.*, vol. 124, no. 2, pp. 313–322, 2002.
- [8] S. Jin, Y. Liu, and Z. Lin, "A Bayesian network approach for fixture fault diagnosis in launch of the assembly process," *Int. J. Prod. Res.*, vol. 50, no. 23, pp. 6655–6666, 2012.
- [9] K. Bastani, B. Barazandeh, and Z. Kong, "Fault diagnosis in multistation assembly systems using spatially correlated Bayesian learning algorithm," *J. Manuf. Sci. Eng.*, vol. 140, no. 3, 2017, Art. no. 031003.
- [10] G. Arvanitis, A. Lalos, and K. Moustakas, "Robust and fast 3D saliency mapping for industrial modeling applications," *IEEE Trans. Ind. Informat.*, vol. 17, no. 2, pp. 1307–1317, Feb. 2021.
- [11] L. D. Xu, C. Wang, Z. M. Bi, and J. Yu, "AutoAssem: An automated assembly planning system for complex products," *IEEE Trans. Ind. Informat.*, vol. 8, no. 3, pp. 669–678, Aug. 2012.
- [12] Q. Rong, J. Shi, and D. Ceglarek, "Adjusted least squares approach for diagnosis of ill-conditioned compliant assemblies," *J. Manuf. Sci. Eng.*, vol. 123, no. 3, pp. 453–461, 2001.
- [13] F. Adly *et al.*, "Simplified subspace regression network for identification of defect patterns in semiconductor wafer maps," *IEEE Trans. Ind. Informat.*, vol. 11, no. 6, pp. 1267–1276, Dec. 2015.
- [14] Z. Kong, D. Ceglarek, and W. Huang, "Multiple fault diagnosis method in multistation assembly processes using orthogonal diagonalization analysis," *J. Manuf. Sci. Eng.*, vol. 130, no. 1, 2008, Art. no. 011014.
- [15] G. A. Susto, A. Schirru, S. Pampuri, S. McLoone, and A. Beghi, "Machine learning for predictive maintenance: A multiple classifier approach," *IEEE Trans. Ind. Informat.*, vol. 11, no. 3, pp. 812–820, Jun. 2015.
- [16] C. R. Qi *et al.*, "PointNet: Deep learning on point sets for 3D classification and segmentation," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, 2017, pp. 77–85.
- [17] D. Maturana and S. Scherer, "VoxNet: A 3D convolutional neural network for real-time object recognition," in *Proc. IEEE Int. Conf. Intell. Robots Syst.*, 2015, pp. 922–928.
- [18] S. Shao, S. McAleer, R. Yan, and P. Baldi, "Highly accurate machine fault diagnosis using deep transfer learning," *IEEE Trans. Ind. Informat.*, vol. 15, no. 4, pp. 2446–2455, Apr. 2019.
- [19] A. Kendall and Y. Gal, "What uncertainties do we need in Bayesian deep learning for computer vision?" in *Proc. Neural Inf. Process. Syst. Conf.*, 2017, pp. 5574–5584.

- [20] C. Leibig *et al.*, "Leveraging uncertainty information from deep neural networks for disease detection," *Sci. Rep.*, vol. 7, no. 1, 2017, Art. no. 17816.
- [21] Y. Yang, W. Li, T. A. Gulliver, and S. Li, "Bayesian deep learning-based probabilistic load forecasting in smart grids," *IEEE Trans. Ind. Informat.*, vol. 16, no. 7, pp. 4703–4713, Jul. 2019.
- [22] P. Franciosa *et al.*, "A novel hybrid shell element formulation (QUAD+ and TRIA+): A benchmarking and comparative study," *Finite Elem. Anal. Des.*, vol. 166, 2019, Art. no. 103319.
- [23] C. Blundell *et al.*, "Weight uncertainty in neural networks," in *Proc. 32nd Int. Conf. Mach. Learn.*, vol. 2, 2015, pp. 1613–1622.
- [24] Y. Wen *et al.*, "Flipout: Efficient pseudo-independent weight perturbations on mini-batches," in *Proc. 6th Int. Conf. Learn. Representations*, 2018.
- [25] D. M. Blei *et al.*, "Variational inference: A review for statisticians," *J. Amer. Statist. Assoc.*, vol. 112, no. 518, pp. 859–877, 2016.
- [26] A. Krizhevsky, I. Sutskever, and G. E. Hinton, "ImageNet classification with deep convolutional neural networks," in *Proc. Neural Inf. Process. Syst. Conf.*, 2012, pp. 1097–1105.
- [27] L. Li *et al.*, "Hyperband: A novel bandit-based approach to hyperparameter optimization," *J. Mach. Learn. Res.*, vol. 18, no. 1, pp. 6765–6816, 2017.
- [28] M. Stein, "Large sample properties of simulations using Latin hypercube sampling," *Technometrics*, vol. 29, no. 2, pp. 143–151, 1987.
- [29] Y. Wu and K. He, "Group normalization," in *Proc. Eur. Conf. Comput. Vis.*, 2018, pp. 3–19.
- [30] X. Glorot and Y. Bengio, "Understanding the difficulty of training deep feedforward neural networks," in *Proc. 13th Int. Conf. Artif. Intell. Statist.*, 2010, pp. 249–256.
- [31] D. P. Kingma and J. L. Ba, "Adam: A method for stochastic optimization," in *Proc. Int. Conf. Learn. Representations*, 2015.
- [32] M. Abadi *et al.*, "TensorFlow: A system for large-scale machine learning," in *Proc. 12th USENIX Conf. Oper. Syst. Des. Implementation*, 2016, pp. 265–283.
- [33] S. Sinha, P. Franciosa, and D. Ceglarek, "Bayesian deep learning for manufacturing," 2020. [Online]: Available. https://github.com/sumitsinha/Deep_Learning_for_Manufacturing
- [34] D. Ceglarek and P. Prakash, "Enhanced piecewise least squares approach for diagnosis of ill-conditioned multistation assembly with compliant parts," *Proc. Inst. Mech. Eng. B J. Eng. Manuf.*, vol. 226, no. 3, pp. 485–502, 2012.
- [35] F. Pedregosa *et al.*, "Scikit-learn: Machine learning in python," *J. Mach. Learn. Res.*, vol. 12, pp. 2825–2830, 2011.
- [36] H. Su *et al.*, "Multi-view convolutional neural networks for 3D shape recognition," in *Proc. IEEE Int. Conf. Comput. Vis.*, 2015, pp. 945–953.
- [37] J. Han, H. Chen, N. Liu, C. Yan, and X. Li, "CNNs-based RGB-D saliency detection via cross-view transfer and multiview fusion," *IEEE Trans. Cybern.*, vol. 48, no. 11, pp. 3171–3183, Nov. 2018.
- [38] X. Li, Y. Jiang, M. Li, and S. Yin, "Lightweight attention convolutional neural network for retinal vessel segmentation," *IEEE Trans. Ind. Informat.*, vol. 17, no. 3, pp. 1958–1967, Mar. 2021.



Sumit Sinha received the B.Tech. degree in industrial and systems engineering from the Indian Institute of Technology (IIT), Kharagpur, Kharagpur, India, in 2017. He is currently working toward the Ph.D. degree in deep learning for manufacturing with Warwick Manufacturing Group, University of Warwick, Coventry, U.K.

From 2017 to 2018, he was a Data Scientist with ZS Associates, Evanston, IL, USA, in the application of data science in pharmaceutical sales and marketing. He was also briefly with the Hong Kong University (HKU), Hong Kong, and Chongqing University, China, working on data science projects. His research interests include Bayesian deep learning and digital twins of manufacturing assembly systems.



Pasquale Franciosa received the Ph.D. degree in mechanical engineering from the University of Naples Federico II, Italy, in 2010.

He is an Assistant Professor with the University of Warwick, Coventry, U.K., and the Head of the Laser Welding Applications Laboratory, Warwick Manufacturing Group, University of Warwick. He has been Principal Investigator (PI) and Co-Investigator (Co-I) on several funded projects with a total income to the University of Warwick of circa £2.8M since 2015.

He has authored or coauthored more than 80 papers. His research interests include smart manufacturing, process monitoring, closed-loop control, applications of machine learning/artificial intelligence, and multidisciplinary optimization, with specific attention on assembly systems and laser processes.

Dr. Franciosa was the recipient of four Best Paper Awards. He is a Member of the Editorial Board of the *ASTM Smart and Sustainable Manufacturing Systems* journal.



Dariusz Ceglarek received the Ph.D. degree in mechanical engineering (ME) from the University of Michigan-Ann Arbor, Ann Arbor, MI, USA, in 1994.

He is an EPSRC Star Recruit Research Chair with the University of Warwick, Coventry, U.K. He was a Professor with IS&E, University of Wisconsin-Madison, Madison, WI, USA. He has been mechanical engineering (ME)/Co-Investigator (Co-I) on research grants of over £30M: NSF/NIST/EPSCRC/InnovateUK/APC/EU-FP7/Curie and industry. He has authored or coauthored more than 180 papers and is listed by Stanford University among top 2% of the world's leading scientists. His research interests include smart manufacturing and data mining/AI for root cause analysis across design, manufacturing, and service.

Prof. Ceglarek is a Fellow of Collège International pour la Recherche en Productique (CIRP). He was the recipient of several Best Paper Awards, the 2018 JLR "Innovista" Award for the most innovative "piloted technology," the EPSRC Star Award, and the NSF CAREER Award. He was the Associate Editor for *ASTM Smart and Sustainable Manufacturing Systems*, the *IEEE TRANSACTIONS ON AUTOMATION SCIENCE AND ENGINEERING*, and the *ASME Journal of Manufacturing Science and Engineering*.

Appendix D: Published Journal Paper 2

S. Sinha, P. Franciosa, and D. Ceglarek, "Building a Scalable and Interpretable Bayesian Deep Learning Framework for Quality Control of Free Form Surfaces," IEEE Access, vol. 9, 2021, DOI: 10.1109/ACCESS.2021.3068867.

Building a Scalable and Interpretable Bayesian Deep Learning Framework for Quality Control of Free Form Surfaces

SUMIT SINHA¹, PASQUALE FRANCIOSA¹, AND DARIUSZ CEGLAREK¹, (Member, IEEE)

Warwick Manufacturing Group (WMG), University of Warwick, Coventry CV4 7AL, U.K.
 Corresponding author: Sumit Sinha (sumit.sinha.1@warwick.ac.uk)

This work was supported by the UK Engineering and Physical Sciences Research Council (EPSRC) project EP/K019368/1: "Self-Resilient Reconfigurable Assembly Systems with In-process Quality Improvement", WMG-IIT scholarship and WMG Centre HVM Catapult.

ABSTRACT Deep learning has demonstrated high accuracy for 3D object shape error modeling necessary to estimate dimensional and geometric quality defects in multi-station assembly systems (MAS). Increasingly, deep learning-driven Root Cause Analysis (RCA) is used for decision-making when planning corrective action of quality defects. However, given the current absence of scalability enabling models, training deep learning models for each individual MAS is exceedingly time-consuming as it requires large amounts of labelled data and multiple computational cycles. Additionally, understanding and interpreting how deep learning produces final predictions while quantifying various uncertainties also remains a fundamental challenge. In an effort to address these gaps, a novel closed-loop in-process (CLIP) diagnostic framework underpinned algorithm portfolio is proposed which simultaneously enhances scalability and interpretability of the current Bayesian deep learning approach, Object Shape Error Response (OSER), to isolate root cause(s) of quality defects in MAS. The OSER-MAS leverages a Bayesian 3D U-Net architecture integrated with Computer-Aided Engineering simulations to estimate root causes. The CLIP diagnostic framework shortens OSER-MAS model training time by developing: (i) closed-loop training to enable faster convergence for a single MAS by leveraging uncertainty estimates of the Bayesian 3D U-net model; and, (ii) transfer/continual learning-based scalability model to transmit meta-knowledge from the trained model to a new MAS resulting in convergence using comparatively less training samples. Additionally, CLIP increases the transparency for quality-related root cause predictions by developing interpretability model which is based on 3D Gradient-based Class Activation Maps (3D Grad-CAMs) and entails: (a) linking elements of MAS model with functional elements of the U-Net architecture; and, (b) relating features extracted by the architecture with elements of the MAS model and further with the object shape error patterns for root cause(s) that occur in MAS. Benchmarking studies are conducted using six automotive-MAS with varying complexities. Results highlight a reduction in training samples of up to 56% with a loss in performance of up to 2.1%.

INDEX TERMS Bayesian deep learning, continual learning, transfer learning, multi-station assembly, U-Net, 3D convolutional neural networks (CNN), manufacturing.

I. INTRODUCTION

A. PROBLEM DESCRIPTION

The manufacturing industry is currently undergoing a data revolution [1], [2] which requires scalable and interpretable approaches that enable digitalization and integration of

in-process monitoring data (i.e. 3D scanners), Artificial Intelligence (AI) algorithms [3], and Computer-Aided Engineering (CAE) simulations to support current Industry 4.0 strategies such as Zero-Defect-Manufacturing and Right-First-Time. These strategies collectively drive towards scenarios where all quality requirements of manufactured products are satisfied starting from Job 1 (right-first-time strategy) and continue during the lifecycle of the product

The associate editor coordinating the review of this manuscript and approving it for publication was Her-Terng Yau¹.

(near zero-defect manufacturing strategy). This is a very challenging task due to increased product variety and smaller batch sizes with ever-decreasing time-to-market. Currently, numerous modern manufacturing systems applications use multiple robotic assembly stations placed in a production line to improve productivity and product quality. For example, a typical automotive body assembly (Body-in-White (BIW)) consists of hundreds of stamping parts and components processed along 60-85 assembly stations with a production rate of 30-65 vehicles per hour with 3-5 varieties of vehicles being produced simultaneously on a single assembly line. Automatic robotic assembly lines can significantly increase productivity and the variety of products produced on a single line. Nonetheless, the embedded complexity of an automatic production line can lead to failures involving robot, end effector with joining head, and/or fixtures, resulting in diminished product quality, scrap, rework, or production downtimes. As a result, there is an urgent need for diagnostics of modern automatic assembly lines. The diagnostics approaches should have the ability to work with current requirements of modern assembly lines, i.e., have capability for (i) multi-variety products with non-rigid parts and shape variances which are rapidly scaled-up from low to high volume production (this translates to the need for rapid *scalability* of diagnostics models between products and production volumes); and (ii) automatic recovery from disruptions at minimum cost (translating to need for *interpretability* of the diagnostic results, which will provide transparency and the causes of their prediction, as they are often used to plan costly corrective action of quality defects).

Therefore, it is crucial to develop diagnostic approaches that are scalable and interpretable to ensure that multi-station assembly systems (MAS) can continue to produce high-quality products in the presence of uncertainties induced by non-ideal parts and operational errors [1], [4]. Recently Object Shape Error Response (OSER) approaches have been proposed for single-station [5] and multi-station assembly systems (OSER-MAS) [6] that have integrated Bayesian deep learning with CAE simulations, hence enabling effective root cause analysis (RCA). The OSER-MAS leverages a Bayesian 3D U-Net encoder-decoder based architecture with multiple output heads. Multiple output heads enable simultaneous estimation of a heterogeneous set of process parameters that can be real-valued or categorical while quantifying uncertainties. The decoder is leveraged to estimate object shape errors for upstream stations. The OSER-MAS gives superior performance for tasks such as process parameter estimation, object shape error estimation and uncertainty quantification but lack capabilities for scalability and interpretability.

This paper proposes a novel *closed-loop in-process (CLIP)* diagnostic framework which simultaneously enhances scalability and interpretability of the current OSER-MAS based approach by integrating an algorithm portfolio inclusive of techniques such as closed-loop training, transfer learning [7], continual learning [8] for scalability and 3D gradient-based

class activation maps (3D Grad-CAMs) [9] for interpretability within the OSER framework. The CLIP diagnostic framework shortens OSER-MAS model training time by developing: (i) closed-loop training to enable faster convergence for a single MAS by leveraging uncertainty estimates of the OSER-MAS model; and, (ii) scalability model to transfer meta-knowledge from the trained model to a new MAS and thus, each new MAS requires comparatively less training samples. Additionally, CLIP increases the OSER-MAS transparency for their quality root cause predictions by developing an interpretability model which entails: (a) linking elements of MAS model with functional elements of the Bayesian deep learning architecture; and, (b) relating features extracted by the Bayesian deep learning architecture with elements of the MAS model and further with the object shape error patterns for root causes(s) that occur in MAS. Although the CLIP framework is developed and validated considering the previously proposed OSER based approaches, the framework can be leveraged to simultaneously enhance the scalability and interpretability of similar approaches in related domains such as stamping and machining where deep learning and CAE simulations are leveraged to relate object shape errors to root causes.

B. RELATED WORK

1) SCALABILITY

Scalability within manufacturing systems has been stated as a set of capabilities to provide transfer of knowledge and ideas from other engineering and management areas [10]. Scalability for algorithms to perform RCA of MASs translates into effectively leveraging the learning for one type of assembly system and then transferring this learning in form of features and relationships which can be relevant for another similar assembly system, and hence can enable learning for the latter using an exponentially lesser amount of data and computation capabilities. Applications of using transfer learning techniques have been done for fault diagnosis [11]. Digital Twins [12] have also been proposed as a way to enable scalability. Successful applications of transfer learning across multiple domains [7], [13]–[16] have enabled scalability in a sustainable manner that does not require exhaustive training data and computation capabilities. Recent works have also proposed that scalability should be life-long or continual and should not come at an expense of forgetting previous learning when new features or relationships are learnt for new systems [8], [17]–[20]. Hence, to enable scalability for MAS frameworks and methodologies that integrate transfer/continual learning with existing deep learning approaches for RCA, it is essential to ensure that training data and computation times do not become barriers for the application of such models within industrial setups.

2) INTERPRETABILITY

Interpretability has been another major concern for the application of deep learning-based RCA models within MAS as they do not provide the required context, trust and confidence

within root cause estimates. This lack of transparency when coupled with costly actions driven by them result in such models not being adopted at scale. Various methodologies such as Gradient-based class activation maps that can integrate deep learning estimates with the required transparency have been proposed [21]–[23]. Bayesian deep learning [5] has been proposed to integrate confidence and uncertainty measures with root cause estimates but there is a need for frameworks within MAS that can provide interpretability while accounting for context and confidence. Such frameworks enable trust in black-box deep learning models and provide different levels of interpretability.

3) ROOT CAUSE ANALYSIS OF ASSEMBLY SYSTEMS

Dimensional and geometric variations are some of the biggest challenges faced by the manufacturing industry. Indeed, two-thirds of quality issues in the automotive and aerospace sectors are caused by dimensional variations [24]. Driven by the development of in-line measurement systems such as coordinate measuring machines (CMM) and 3D scanners, models for RCA of assembly systems have seen a lot of development in both, industrial applications, and academic research. These models [25] can be grouped into two categories: (a) knowledge-based models; and (b) estimation-based data-driven models leveraging statistical and machine learning techniques [26]. For single station systems, Apley and Shi [27] established the deviation transfer model based on process information such as fixture positioning and used least-squares to diagnose fault sources. Chang *et al.* [28] leveraged a linear model between shape error sources and measurement features followed by parameter estimation and statistical tests to diagnose shape error sources. Yu *et al.* [29] leveraged influence coefficients based on finite element modeling to establish shape errors between the sources of variation and measurements of flexible sheet metal parts. Further least-squares estimation was used to estimate errors in fixture positioning.

For MASs, Agrawal *et al.* [30] used regression models of sensor data. Zou *et al.* [31] proposed integrating BIC with LASSO variable selection. Shang *et al.* [32] proposed a Binary State Space Model (BSSM) for MASs to perform binary diagnosis. Jin *et al.* [33] proposed state-space and stream-of-variation (SoV) for multi-station shape error propagation of automotive assemblies. Ding *et al.* [34] extended the SoV method of assembly shape error of rigid parts using state space considering different fixture locating scheme. Using the above approaches as a base, various RCA models for MASs have been proposed. Ding *et al.* [35], [36] compared different variance estimation techniques and concluded a basis for method selection under different scenarios. Ceglarek and Prakash [37] proposed shape error diagnosis based on enhanced piecewise least squares (EPLS) to detect and isolate collinear dimensional faults caused by fixture variation. Ceglarek and Shi [38], [39] employed pattern matching for diagnosis of fixtures based on principal component analysis. Liu and Hu [40] used designated component

analysis for shape error diagnosis of flexible sheet metal parts. Various enhancements have been proposed using the knowledge of MASs [41], [42].

Given the ever-increasing complexity of MASs, increased computation capabilities and developments in machine learning, recently, RCA approaches [26] using machine learning have been proposed to overcome limitations of the above-stated methods such as linear approximations of the MAS. Du *et al.* [43] utilized artificial neural networks to monitor and identify process variability. Beruvides *et al.* [44] applied reinforcement learning to perform RCA. Bayesian Networks [45], [46] are seen as an alternative to solve small dataset problems and integrate process data and engineering knowledge. Recently, Sinha *et al.* developed Object Shape Error Response (OSER) for single-station [47], [5] and Object Shape Error Response for Multi-Station Assembly Systems (OSER-MAS) [6] that aim to integrate Bayesian deep learning elements such as Bayesian 3D Convolutional Neural Networks and Computer-Aided Engineering (CAE) simulations thereby, blending (a) engineering knowledge-techniques with (b) estimation-based data-driven approaches. This satisfies various model capability requirements for RCA of MASs such as (i) high data dimensionality [48]; (ii) non-linearity [49]; (iii) collinearities [50]; (iv) high faults multiplicity [51]; (v) uncertainty quantification [52]; (vi) dual data generation capabilities [12]; (vii) high dimensionality and heterogeneity of process parameters [53]; and, (viii) fault localization [54]. In addition to the aforementioned model capability requirements, RCA techniques must be further developed and enhanced to fulfil two additional key requirements [55] in order to enable implementation and large-scale adoption across different manufacturing environments:

(ix) *Scalability* as automotive multi-station assembly processes include hundreds of stamping parts and components, multiple stations with multiple stages in each station [4] namely, place-clamp-fasten-release (PCFR) to finish the final assembly product. The multiple variation sources in the MAS interact and accumulate in a non-additive manner. The final product accuracy and performance depend upon the accumulated performance of individual stations in the system. In MASs, the quality of the components is influenced by (i) incoming non-ideal and deformable parts; (ii) PCFR-to-part interactions as parts move through PCFR stages (shape error induced by fixturing and joining operations); (iii) part-to-part interactions within each station and further magnified between stations; and, (iv) station-to-station interactions due to re-orientation errors between stations (change of fixture locating layout between stations).

(x) *Interpretability* as the estimations of root cause(s) will require insights into why such estimates were made by the deep learning model. Such interpretability insights are essential for contextualizing the root cause(s) estimated by the deep learning model. Additionally, root cause (RC) estimates drive costly corrective actions hence, model interpretability integrated with measures of uncertainty [5] are crucial

requirements for effective and efficient corrective and control actions.

The paper will address the aforementioned requirements as follows:

Requirement (ix) by developing a closed-loop training framework that leverages the epistemic uncertainty [5] estimates of the Bayesian 3D U-Net based OSER-MAS model to intelligently sample from the process parameter hyperspace for faster convergence and hence, reduce the computation time bottleneck of the CAE simulations. The presented CLIP diagnostic framework in this paper will utilize high fidelity CAE simulator of the assembly process called Variation Response Method (VRM) [56]. Further, to exponentially enhance the scalability for high dimensional MAS and reduce CAE simulation time, uncertainty guided continual learning [8] and transfer learning [7] features are integrated with the CLIP diagnostic framework underpinned algorithm portfolio. This enables the transfer of meta-knowledge from the trained model to a new MAS and thus, each new MAS requires comparatively less training samples. Theoretically, given that multi-physics processes of the assembly system are similar within each station hence, the features extracted by spatial convolutional operations are transferable across different assemblies; thus, making transferability within the model essential for scalability. Given that models for different MAS would be trained sequentially, leveraging continual approaches reduces catastrophic forgetting. A model with continual learning capabilities can also account for the dynamic nature of the assembly system and hence, achieve life-long learning. This is accomplished using uncertainty guided continual learning [8] that leverages the Bayesian neural network parameter uncertainty to assign importance for each task i.e., a particular assembly case study thereby, enabling continual learning by updating less important parameters at a faster rate.

Requirement (x) by developing an interpretability model which is based on 3D Gradient-weighted Class Activation Maps (3D Grad-CAMs) and entails: (a) linking elements of MAS model with functional elements of the 3D Bayesian U-Net model; and, (b) leveraging 3D Grad-CAMs [9] that provide insights into the regions within the input that the model focuses on to estimate process parameters i.e. root cause(s). These collectively provide the required interpretability for RCA. Additionally, Bayesian Deep Learning enables uncertainty quantification and segregation into aleatoric and epistemic uncertainties, thus, providing a measure of the required confidence while conducting RCA.

The key contribution of the paper is the development of the CLIP diagnostic framework underpinned algorithm portfolio that includes the following models:

- (1) A closed-loop training model to enable faster convergence for a single MAS by leveraging uncertainty estimates of the Bayesian 3D U-net OSER-MAS model.
- (2) Uncertainty guided transfer/continual learning-based scalability model to transfer meta-knowledge from the

trained model to a new MAS and thus, each new MAS requires comparatively less training samples.

- (3) A 3D Grad-CAMs based interpretability model which links functional elements and features extracted by the 3D U-Net architecture with elements of the MAS model and further with the shape error patterns for root causes(s) that occur in MAS.
- (4) Verify and validate the scalability and interpretability capabilities of the CLIP diagnostic framework underpinned algorithm portfolio using six different industrial automotive assemblies of varying complexities.

The rest of the paper is organized as follows; Section II formulates the object shape error estimation and RCA problem for MASs, Sections III and IV discuss the methods for scalability and interpretability, respectively, Section V presents the industrial case studies and finally, conclusions and future work are summarized in Section VI.

II. PROBLEM FORMULATION

A. MULTI-STATION ASSEMBLY SYSTEMS

As illustrated in Fig. 1 [6], MAS can be represented as a process tree where different nodes correspond to stages within a single assembly station (Fig. 1a) or as stations within the assembly system (Fig. 1b). A station consists of multiple stages namely, positioning, clamping, fastening and release (PCFR). The input to each station is a set of incoming parts (objects) that need to be assembled. Within the process, object shape errors can be induced in any station by one or multiple variations of the process parameter(s). These errors are further propagated and accumulate in a non-additive manner [34]. Any variation in the process parameter(s) is a source of shape error and thus must be first quantified and then estimated as a root cause(s) of the shape errors. In MAS, these process parameters are classified into three categories: (a) Real-valued parameters of incoming parts (objects) variation as caused by upstream fabrication processes such as stamping, extrusions, etc.; (b) Real-valued process parameters related to PCFR stages of each assembly station. They represent any deviation from nominal in fixturing/tooling or joining operations; and, (c) Binary joining-based process parameters in the fastening stage that indicate the success of the joint. The value is {1} when joint is successfully completed or {0} for an unsuccessful joint due to the excessive gap between objects to be joined or current failure in the tool. In this paper, Self-Piercing Riveting (SPR) is the considered fastening /joining process.

Stations in the MAS are represented by $s: s = 1, \dots, N_s$, where N_s represents the total number of stations within the system and $s': s' = 1, 2, 3, 4$ represents the four stages within each station. Any object $o: o = 1, \dots, n$ at its design nominal shape is characterized by a set of nominal points $P_o = \{p_{ok}\}$, $k = 1, \dots, n_o$, where p_{ok} is a vector consisting of the x , y and z coordinates of the k th input point and n_o represents the total number of points on object o . $d_o = \{d_{ok}\}$ denotes the deviation of each point k after the nominal object o has gone through different stages of the process, d_{ok} is a vector

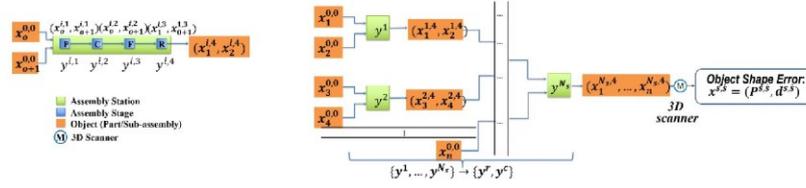


FIGURE 1. Multi-station assembly system with PCFR assembly stages. (a) PCFR stages of an assembly station (b) Multi-station assembly system with N_c stations

comprised of deviations of each point in x, y and z axes on object o . As the object o goes through multiple stations and stages within the stations, the set of points are represented as $P_o^{s,\delta}$ while $d_o^{s,\delta}$ represents the deviations. Object shape error for object o after station s and stage δ can be represented as:

$$x_o^{s,\delta} = (P_o^{s,\delta}, d_o^{s,\delta}) \quad (1)$$

where $x_o^{0,0} = (P_o^{0,0}, d_o^{0,0})$ represents the incoming non-ideal part inclusive of the part variations and $(x_1^{s,\delta}, \dots, x_n^{s,\delta})$ represents a set of objects (sub-assembly) after stage δ of station s .

On the other hand, the aforementioned process parameters which can be real-valued or binary are represented by $y^{s,\delta}$ for station s stage δ . The process parameters for station s , inclusive of all stages, is represented by y^s while y represents the total set h of process parameters across the entire assembly system. The process parameter vector y consists of c binary parameters and r real-valued parameters denoted by y^c and y^r , respectively.

The aim of the proposed 3D U-Net CNN model training is to learn a function $f(\cdot)$ that takes as input the combined object shape error at the end of the system $x^{N_s,4}$, i.e., after the final stage of the last station (N_c), and estimates the process parameters across the entire system and the object shape error for all objects at the end of the previous stations:

$$[y^r, y^c, x^{1,4}, \dots, x^{N_s-1,4}] = f(x^{N_s,4}) \quad (2)$$

B. ROOT CAUSE ANALYSIS

For comprehensive RCA for MASs, the paper proposes three key steps namely: (i) fault identification; (ii) fault localization; and, (iii) fault isolation. Using the estimates obtained within (2), RCA can be done using the following steps to isolate single or multiple faults that occur in a MASs.

1) FAULT IDENTIFICATION

Fault identification involves identifying which process parameters are potentially at fault. Faults can be identified by comparing the values of process parameters with given standards. For all binary process parameters y^c that are $\{0\}$ are identified as faults as they represent a failure in the joining process. Real-valued process parameters can be identified as potential faults based on the fault identification strategy.

If a threshold approach is leveraged, each $y^{s,\delta}$ that is greater than the threshold is identified as a fault on the other hand if six-sigma fault identification strategies are used a sample of products s_p are observed and mean $\mu_p^{s,\delta}$ and standard deviation $\sigma_p^{s,\delta}$ of each process parameter is calculated. Based on the significance level used, a mean shift or a change in variance (heteroskedasticity) in process parameters can be identified as a fault. The subset of process parameters identified as fault via a threshold or a six-sigma approach can be denoted as:

$$y_F \subseteq y \quad (3)$$

2) FAULT LOCALIZATION

Fault localization involves identifying particular stations within which the object(s) (sub-assemblies) shape error becomes significant. The shape error estimates $[x_o^{1,4}, \dots, x_o^{N_s,4}]$ for all objects $o : o = 1, \dots, n$, are compared with the design nominal. If the shape error is beyond the threshold (assembly tolerances) at the end of a station and within the threshold, for the previous station the fault is localized to that particular station for the corresponding object. This is done for each object and thus, multiple faults for different objects can also be localized. The subset of stations localized is denoted as:

$$s_F \subseteq s, \quad o_F \subseteq o \quad (4)$$

3) FAULT ISOLATION

Fault isolation involves integrating the information from fault identification and fault localization to isolate which process parameters within the potentially identified faults y_F and localization stations s_F are ‘malignant’ and have a significant impact on the shape error of the final object (product). For all potentially identified process parameters, the process parameters that lie within the localized stations s_F (and corresponding objects o_F) are isolated as faults and estimated as RCs. Given that manufacturing systems are stochastic, all process parameters always have an inherent level of variation, hence, using such a three-step approach enables differentiating faults that are benign (have no significant impact on the product shape error) from faults that are malignant (cause product shape error to go beyond assembly thresholds). The RCs can

TABLE 1. Closed loop sampling & training framework.

Algorithm 1: Closed-loop sampling and training	
Procedure:	
Generate and evaluate Testing (E) and Validation (V) set	
Generate Initial Training Set (T^0)	
Repeat $l = 1$ to n^m :	
Train model on training set (T^l)	
Perform inference on validation (V) set	
Estimate GMM parameters (ϕ, μ, Σ) using (8), (9) and (10)	
Sample from the GMM and evaluate T^s	
Add samples to training set $T^{l+1} \leftarrow T^l \cup T^s$	
Stopping Checks:	
Error on test (E) set, below the threshold $e^o < \varepsilon$	
Maximum number of iterations reached $l = n^m$	

as input a set of process parameters and outputs the object shape errors after the desired stage/station. Initially, Latin Hypercube Sampling (LHS) [57] of the process parameters within the allowable ranges is done for input to the CAE simulation model to generate the test (E) and validation (V) set.

$$E = (X^E, Y^E), \quad V = (X^V, Y^V) \quad (6)$$

While each element in X is characterized by the object shape error $x^{s,s}$ (1) and each row in Y consists of a vector of process parameters: $y = \{y^r, y^c\}$. The initial training set T^0 is also generated using LHS. This is used to train the proposed architecture $f(\cdot)$. After training inference is performed on the validation set to obtain the predictions and uncertainty on the validation set (V)

$$[\hat{Y}^V, \sigma^V] = f(X^V) \quad (7)$$

For each sample the absolute error is calculated and summed up across all the process parameters:

$$E^v = |\hat{Y}^V - Y^V| \quad (8)$$

The error is summed up across all h process parameters for each sample e^v which is a column vector consisting of combined error for each sample.

$$e^v = \sum_h E^v \quad (9)$$

The normalized error (\bar{e}^v) and uncertainties ($\bar{\sigma}^V$) are weighted to obtain the sampling importance metric τ for each sample:

$$\tau = w \cdot \bar{e}^v + (1 - w) \cdot \bar{\sigma}^V \quad (10)$$

Samples are sorted based on the sample metric. This is done considering that the samples having the highest importance, i.e., having the maximum sum of error and uncertainty would have a more significant contribution to model convergence than other samples. Based on sampling metrics and actual process parameters Y^V , the parameters of the sampling distribution are estimated. Gaussian Mixture Model (GMM)

with a pre-specified number of mixtures is considered as the sampling distribution. The sorted set of sampling metric and process parameter is represented as $(\check{\tau}^v, \check{Y}^V)$. Given the GMM has K mixtures, the sorted set is subdivided into K blocks each block i^b having samples $\tau_{i^b} = \dim(V)/K$ where $\dim(V)$ represents the total number of samples within the validation set. Each block i^b is used to estimate distribution parameters for the i^b th mixture. Each mixture is characterized by a multi-variate normal distribution while the component within the multi-variate normal corresponds to the process parameter. The distribution parameters for the i^b th component consisting of s samples is estimated as in (11), where $Y_{i^b}^V$ represents the subset of samples in the block i^b

$$\phi_{i^b} = \sum_s \check{\tau}^v, \quad \mu_{i^b} = \text{Mean}_s(Y_{i^b}^V), \quad \Sigma_{i^b} = \text{Covar}_s(Y_{i^b}^V) \quad (11)$$

where ϕ vector is normalized, to sum up to 1 and hence represent mixture component weights ϕ_{i^b} , and where μ_{i^b} , Σ_{i^b} represent the mean vector and covariance matrix respectively for the i^b th mixture. After estimating the distribution, T^l samples are drawn and evaluated and then added to the training set to further train the model. This ensures that samples are drawn considering the error and uncertainty while accounting for the fact that the model should not over fit on the validation set. It should be noted that although the validation set is used for sampling, the model is never trained on the validation samples, the randomness in the sampling ensures that samples are drawn from regions where the error and uncertainty are high. After each training is done the model is tested on the test set (E) to determine the model performance. The training and sampling are terminated either after i) the performance on the test set (E) reaches the required threshold ε or ii) the maximum number of training iterations n^m is reached. The performance threshold can be decided based on the case study and application, and the maximum number of iterations is decided based on the CAE simulation budget.

C. UNCERTAINTY GUIDED CONTINUAL LEARNING

The closed-loop sampling approach enables faster convergence within one MAS but does not enable the trained $f(\cdot)$ to be used across different MASs. Leveraging the trained function $f(\cdot)$ across different MASs to enable transferring of relevant knowledge (features) and hence enable convergence in comparatively lesser samples is crucial in enabling scalability. Continual learning methods (also known as sequential/lifelong learning) aim to incrementally learn new tasks without forgetting previous tasks for which they have been trained. In the context of MASs, the scale-up starts from tasks or cases as simple as a coupon or Top-hat assembly to full-scale MASs such as automotive car door or cross member assemblies and can cumulatively consist of up to 100 assembly stations each consisting of multiple stages. Each assembly case is treated as a task T_i , and continual learning is performed for a total of T_n tasks. Continual learning enables the transfer of the process parameter estimation

capabilities of previous assembly cases to more complex assemblies while retaining the essential capabilities required for process parameter estimation of previous assemblies. The key to achieving continual learning requires assignment of importance to each neural network weight ω and further updating only non-important weights such that the model learns the new task without forgetting the previous task [8]. The approach leverages uncertainty guided continual learning because the weight uncertainty σ_ω of the Bayesian 3D U-Net model serves as an implicit measure of importance. Additionally, the ease of interpretation, strong mathematical foundation and good results on various datasets [8] motivate the use of such a learning algorithm.

Given the use of Bayesian neural networks and a normal distribution parametrized by $\theta_\omega = (\mu_\omega, \sigma_\omega)$ for each weight ω within the network, the standard deviation of each weight distribution is leveraged as the metric for importance. To enable continual learning the learning rate α for each parameter is updated by the corresponding importance Ω .

$$\alpha_\mu \leftarrow \alpha_\mu / \Omega_\mu, \alpha_\sigma \leftarrow \alpha_\sigma / \Omega_\sigma \quad (12)$$

The importance of the parameters is set to be inversely proportional to the standard deviation, which mathematically translates that weights with higher standard deviation are less important and hence can be updated at a higher rate to learn new tasks, while weights with lower standard deviation are more important and hence should be updated at a lower rate to prevent catastrophic forgetting (performance loss) for the old tasks.

$$\Omega \propto 1 / \sigma_\omega \quad (13)$$

Based on various empirical studies done by Ebrahimi et al. [8], the learning rate adaptations were determined as:

$$\Omega_\mu \leftarrow 1 / \sigma_\omega, \quad \Omega_\sigma \leftarrow 1 \quad (14)$$

The overall algorithm consisting of closed-loop sampling and continual learning, to train the model on multiple tasks T_1, \dots, T_n (different assembly case studies) is shown in Table 2. After each task, the learning rates are updated as shown in (14). The number of output nodes within the model is kept equal to sum all process parameters across all cases studies. For each assembly case, the specific process parameters nodes output the values while other nodes corresponding to process parameters for other assembly cases are set to output a nominal fixed value (generally set to zero). Overall continual learning aims to learn process parameter estimation capabilities of each assembly case study incrementally while minimizing the forgetting for previous assembly case studies.

D. TRANSFER LEARNING

Transfer learning is an effective method for transferring learning (process parameter estimation for MASs) from one task to related tasks (between different assembly case studies) using exponentially lesser training samples and hence acts as a key

TABLE 2. Continual learning framework.

Algorithm 2: Uncertainty based continual learning with closed-loop sampling	
Procedure:	
Setup all tasks (Assembly Case Studies)	
Set initial learning rate $\alpha_\mu = \alpha_\sigma = \alpha_0$	
Repeat for all tasks T_i:	
Perform closed loop training for T_i^{th} task (Algorithm 1)	
Update Learning Rates:	
$\Omega_\mu \leftarrow 1 / \sigma_\omega$	
$\Omega_\sigma \leftarrow 1$	
$\alpha_\mu \leftarrow \alpha_\mu / \Omega_\mu$	
$\alpha_\sigma \leftarrow \alpha_\sigma / \Omega_\sigma$	
end	

enabler for scalability. The paper leverages transfer learning and continual learning as a combined algorithm portfolio enabling scalability. The choice between them can be made based on training results and deployment performance.

Transfer learning is mathematically formalized [14], [13] as a domain D which consists of features X and a distribution over the feature space $P(X)$. In this case the domain entails process parameter estimation $f(\cdot)$ on a particular assembly case T_i with shape error features X and distribution over the feature space as $P(X)$. Within Domain D task T is performed that constitutes learning a conditional distribution $P(Y|X)$ to estimate process parameters Y

$$D = \{X, P(X)\} \quad (15)$$

The paper aims to ‘transfer learn’ from the source domain (D_s) corresponding to a particular assembly case to a target domain (D_T), i.e. a similar assembly case to perform the same task of estimating $f(\cdot)$ (2) while accounting for differences between cases such that at least one of the elements between the domain and target are not the same:

$$(X_s \neq X_T), (P(X_s) \neq P(X_T)), (Y_s \neq Y_T), \\ P(Y_s|X_s) \neq P(Y_T|X_T) \quad (16)$$

Considering the prior knowledge on the similarity of assembly cases studies it can be estimated that: (i) $(X_s \approx X_T)$ —given that similar features [58] need to be extracted from the object shape error data that include bends, twists, rotations, translations etc.; (ii) $(P(X_s) \approx P(X_T))$ —similarly, the distribution around the input features is approximately the same; (iii) $Y_s \neq Y_T$ —the outputs for each study are different given a different number of process parameters are involved; and, (iv) $(P(Y_s|X_s) \neq P(Y_T|X_T))$ —the conditional distribution is also significantly changed given the change in the assembly system and the output. To account for (iii) the final layer of the network is replaced with nodes corresponding to the new set of process parameters; to account for (iv) the approach leverages standard protocols established in transfer learning to achieve transferability.

Based on past work on successful applications of transfer learning that involved using ImageNet data to aid

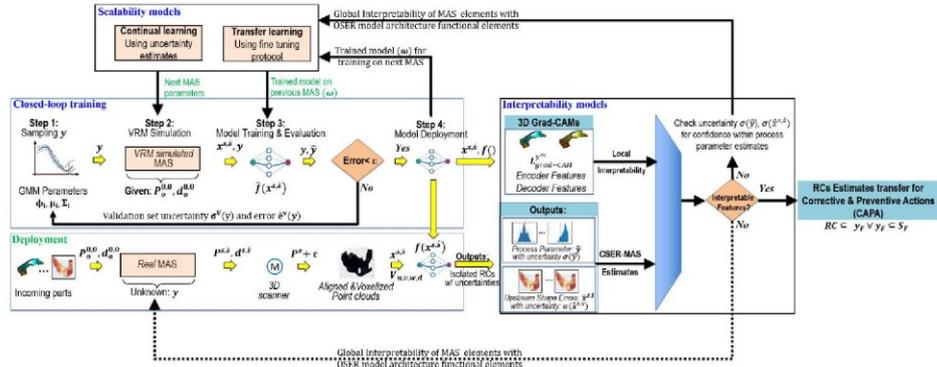


FIGURE 3. Closed-loop in-process (CLIP) diagnostic framework underpinned algorithm portfolio.

Computer-Aided Detection [15], the fine-tuning transfer learning protocol is leveraged. The network weights are initialized using the weights of a network trained on the previous assembly case study. The whole network is then fine-tuned while keeping the learning rate of the convolutional layers α_c in the first two encoder levels ten times less than the rest of the network α_f . The regression and classification heads are replaced and reinitialized. The nodes in each of the heads are determined by the number of process parameters for the case study. Overall transfer learning (Table 3) aims to learn process parameter estimation capabilities of each assembly case study while ‘transferring’ knowledge from previous case studies. Although while doing this the model can ‘forget’ estimation capabilities for previous case studies. Fig. 3 summarizes the overall framework for closed-loop sampling and training integrated with continual and transfer learning approaches.

IV. METHODS FOR INTERPRETABILITY

A. 3D U-NET ARCHITECTURE INTERPRETABILITY

The implementation of deep learning models within industrial environments requires opening the black box and providing interpretability and causation on why the deep learning models can give superior performance as compared to traditional linear or piece-wise linear approaches traditionally used for RCA of MASs. The paper proposes to do that on two levels:

Providing a link between MASs requirements and functional elements of the Bayesian 3D U-Net architecture of the OSER-MAS approach. This also aims to provide a link between the engineering challenges faced in RCA of MASs and the developments done within the OSER-MAS model to overcome these challenges.

Leveraging 3D Grad-CAMs to interpret the features that are extracted by the architecture and then propagated through various encoder and decoder layers to be interpreted as root causes. To integrate high measures of confidence within the

TABLE 3. Transfer learning framework.

Algorithm 3: Transfer learning with closed-loop sampling

Procedure:
Setup all tasks (Assembly Case Studies)
Set initial learning rate α_0 for all layers
Perform closed-loop training on 1 st task
Set learning rate lower convolutional layers $\alpha_c \leftarrow \alpha_0/10$
Set learning rate for all other layers $\alpha_f \leftarrow \alpha_0$
Repeat for all tasks (2 to T_n):
Replace output layer of neurons equal to the number of process parameters for the T_i^{th} task
Perform closed loop training for T_i^{th} task (Algorithm 1)
Initialize network with trained weights
end

deep learning model estimates, it must be established that the input context $x^{N_s,4}$ (object shape error) on which the model focuses should be directly related to the estimated output y (process parameter or root cause), e.g. if the model estimates ‘part variation’ as a root cause, it should focus on the ‘part variation’ rather than other possible root causes such as ‘clamping’ or ‘positioning’. Clearly extracted semantics in convolutional layers integrates a much-needed measure of ‘trust’ within the root cause estimates.

B. 3D GRADIENT-WEIGHTED CLASS ACTIVATION MAP

3D Gradient-weighted class activation maps (3D Grad-CAMs) aim to visualize the input features that led to a particular output. In the context of MASs, this aims to localize key regions within the input shape error $x^{N_s,4}$ that led to the estimation of a process parameter y . This is estimated by taking a discriminative gradient of a particular process parameter y^m output with respect to the feature map of a selected convolutional layer within the 3D U-Net architecture. The map for a particular output process parameter y^m is

represented as $L_{grad-CAM}^{y^m}$ and can be calculated as a weighted sum of the features maps:

$$L_{grad-CAM}^{y^m} = ReLU \sum_{f=1}^F \alpha_f^{y^m} A^f \in \mathbb{R}^{u \times v \times w} \quad (17)$$

where A^f represents f feature maps ($f = 1, 2, \dots, F$) for the selected convolutional layer and a, b, c represent the dimensions of the 3D feature maps. ReLU represents the activation function which is rectified linear unit. The weights are calculated by summing the gradients for each element within the feature map:

$$\alpha_f^{y^m} = \frac{1}{abc} \sum_{i=1}^a \sum_{j=1}^b \sum_{k=1}^c \frac{dy^m}{dA_{i,j,k}^f} \quad (18)$$

The $L_{grad-CAM}^{y^m}$ is interpolated to match the dimensions of the input voxelized object shape error. The overlay between the voxelized object shape error and interpolated 3D Grad-CAM provides interpretability information on what features/spatial regions within the shape error did the model focus on to estimate the selected process parameter y^m of interest. The interpolated 3D Grad-CAM is then transformed to a point-based shape error and smoothing is done using a median filter for consistency across the mesh. These can be visualized to obtain regions within the shape error input that the neural network model is focusing on to estimate the process parameter.

Fig. 3 summarizes the algorithm portfolio including the integration of closed-loop training with continual/transfer learning-based scalability model and 3D Grad-CAMs based interpretability model. The effective integration of these models enhances the diagnostic capabilities of the OSER-MAS model and enables scalable and interpretable RCA.

V. CASE STUDIES

A. EXPERIMENTAL SETUP

Verification and validation is done using $T_n = 6$ tasks or assembly systems (Fig. 4, Table 4) with varying complexities ranging from a single part coupon level assembly to automotive industrial multi-station assemblies. Each assembly case is considered as a unique task. Continual or transfer learning is done sequentially for all case studies as in the order mentioned below. The case studies include:

(1) *Flat Plate (Coupon) Assembly*: consists of $n = 1$ ideal compliant part with a flat 2D geometry. It involves $N_s = 1$ station and four stages (PCFR) and is controlled by $h = 7$ real-valued y^r fixturing and joining based process parameters.

(2) *Top-Hat Assembly*: consists of a $n = 2$ ideal compliant parts with a simple 3D geometry. It involves $N_s = 1$ station and 4 stages (PCFR) and is controlled by $h = 17$ real-valued y^r fixturing and joining based process parameters.

(3) *Door Halo Reinforcement Panel Assembly*: consists of $n = 1$ ideal compliant part with complex 3D geometry. It involves $N_s = 1$ station and 2 stages (PC) and is controlled by $h = 3$ real-valued y^r fixturing based process parameters.

(4) *Door Inner and Hinge Reinforcement Assembly*: consists of $n = 2$ ideal compliant parts with a complex

TABLE 4. Assembly cases.

Case	Geometry	Input Parts	Stages	Stations	Process Parameters
(1) Flat Plate	2D	1 ideal part	4	1	7
(2) Top-Hat	3D	2 ideal parts	4	1	17
(3) Door Halo	3D	1 ideal parts	2	1	3
(4) Door Inner and Hinge	3D	2 ideal parts	4	1	6
(5) Cross Member	3D	4 non-ideal parts	12	3	12
(6) Cross Member	3D	4 non-ideal parts	12	3	148

3D geometry. It involves $N_s = 1$ station and four stages (PCFR) and is controlled by $h = 6$ real-valued y^r fixturing based process parameters.

(5) and (6) *Cross Member Assembly*: consists of $n = 4$ non-ideal compliant parts with complex 3D geometry. It involves $N_s = 3$ stations each with four stages (PCFR). Two sub-cases within this are considered: case (5) consisting of $h = 12$ real-valued y^r part variation and fixturing based process parameters; and, case (6) consisting of a heterogeneous (real-valued and binary) set of $h = 158$ process parameters including 123 real-valued y^r part variation, fixturing and joining based process parameters and 25 binary process parameters y^c indicating the success of joining (Fig. 5).

For comparison and benchmarking of scalability, all five cases are analyzed under $T_s = 4$ training scenarios are considered:

(i) *Random Sampling*: Involves randomly sampling from the CAE simulator within the allowable ranges for each process-parameter. Each case study is trained on a re-initialized network with random weights. This also serves the baseline performance expectations.

(ii) *Closed-loop Sampling*: Involves training the five case studies using Algorithm 1 as shown in Table 1.

(iii) *Transfer Learning with Closed-loop sampling*: Involves training the five case studies sequentially using Algorithm 3 as shown in Table 3.

(iv) *Continual Learning with Closed-loop Sampling*: Involves training the five case studies sequentially using Algorithm 2 as shown in Table 2.

Interpretability is verified by considering the cross member assembly (5) to provide links between MASs requirements and architecture functional elements and to obtain 3D Grad-CAMs for key process parameter variations. While obtaining 3D Grad-CAMs the weights and biases of the network are fixed at the mean values ($\omega = \mu_\omega$).

Before training all shape errors are pre-processed and voxelized to $(u, v, w, d) = (64, 64, 64, 3)$ voxel grids $V_{64,64,64,3}$. The deviation features d include deviations in all directions for all points $(\bar{x}_k, \bar{y}_k, \bar{z}_k)$. The shape error after the final station ($x^{N_s,4}$) is used as input while the process parameters y and *upstream* stations shape errors

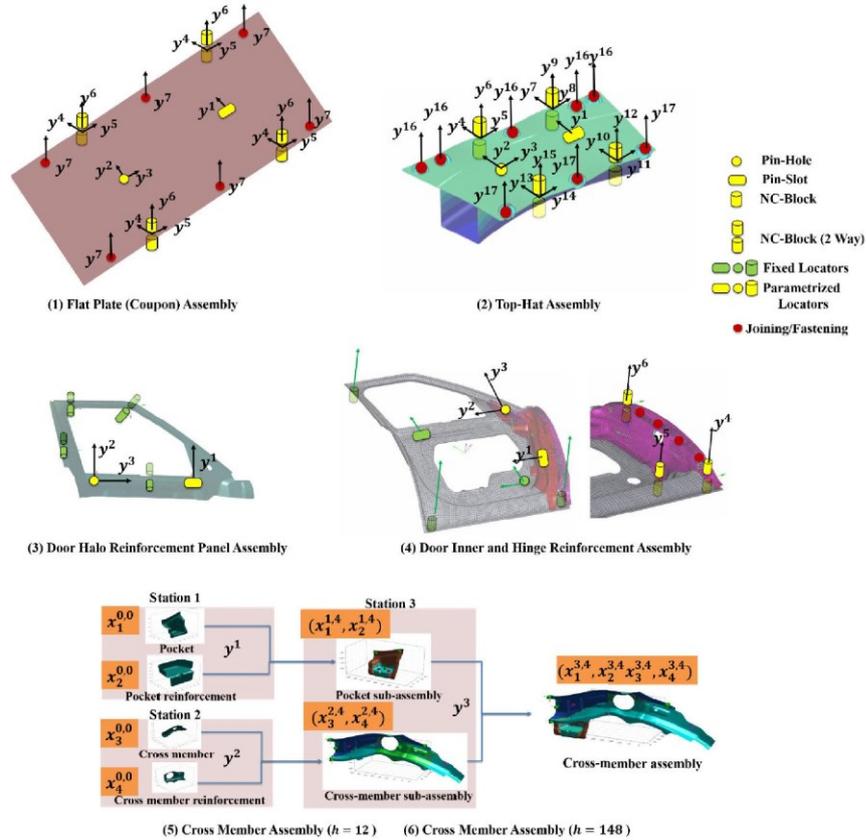


FIGURE 4. Assembly cases.

$x^{1,4}, \dots, x^{N_s-1,4}$ are used as output. The model architecture hyperparameters were selected as proposed in the OSER-MAS approach. Training hyperparameters were optimized for all scenarios. Adam optimizer [59] is used for training in scenarios (i), (ii) and (iii). Initial learning rates $\alpha_0 = [0.1, 0.01, 0.001, 0.0001]$ were compared for scenario (i) and (ii), $\alpha_0 = 0.001$ gave optimal performance in terms of error and convergence, $\alpha_0 = [0.1, 0.01]$ gave inferior performance as compared to $\alpha_0 = [0.001, 0.0001]$, $\alpha_0 = 0.001$ was finally selected as the learning rate gives faster convergence between the two values. The same combinations were tested for scenario (iii) while under the constraint that $\alpha_C = \alpha_F / 10$ given the fine-tuning protocol to ensure later layers learn at a faster rate as compared to initial layers. $\alpha_C = 0.0001$ and $\alpha_F = 0.001$ gave the most optimal performance.

Scenario (iv) tested initial learning rate for stochastic gradient descent (SGD), $\alpha_0 = 0.001$ gave optimal performance. The learning rates were multiplied in each case by the weight uncertainty as described in Table 2. Minibatch sizes of 8, 16 and 32 were tested. Larger batch sizes could not be used given the high GPU memory requirements of 3D CNNs. Minibatch size of 32 gave the best performance. Smaller sizes such as 8 and 16 caused the training process to be unstable. The model is trained for 300 epochs. Group normalization [60] with four groups is used after each convolutional layer. This prevents overfitting and accounts for small minibatch size due to GPU memory size constraints and aids in stabilizing the training process. Optimization for training hyperparameters was done for case (1) (Flat Plate Assembly) to ensure computation feasibility. Given case

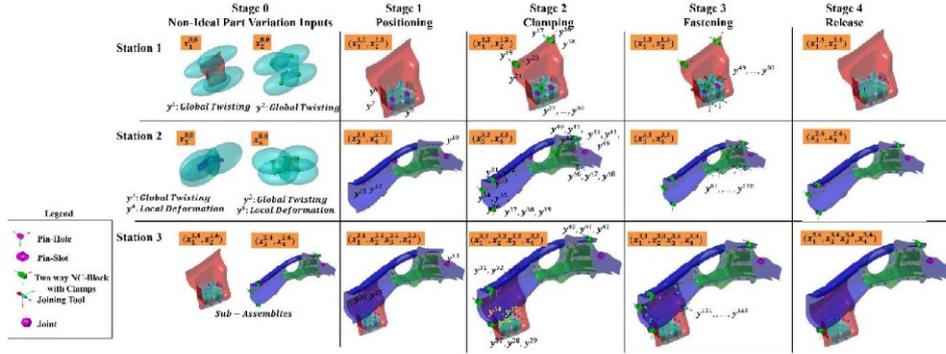


FIGURE 5. Process parameters for cross member assembly for case (5) and (6).

studies (1) to (4) consist of a single station; the decoder of the 3D U-Net model is not used as upstream shape error for previous station stations does not need to be estimated. Case study (5) leverages the decoder to estimate shape error after upstream stations. The work has been implemented using Python 3.7 and TensorFlow - GPU 2.1 [61] and TensorFlow Probability 0.9. A python library named DLMFG [62] has been developed to validate and replicate the results of the methodology. For this paper, both, the data generation and evaluation of the approaches have been done using VRM. Two Nvidia Tesla V100 32 GB GPUs are used for model training and deployment.

B. DISCUSSION: SCALABILITY

The results for training assembly cases sequentially in the aforementioned scenario are summarized in Fig. 6. Training in all scenarios is done until convergence. Model is considered converged when the performance metrics on the test set $E = (X^E, Y^E)$ are better than the threshold. R-Squared (R^2) ≥ 0.90 is considered as the convergence criteria for y^r and Receiving Operating Characteristics - Area Under Curve (ROC-AUC) ≥ 0.90 is considered as a convergence for y^c . The initial training set size T^0 is set to be 500 samples and $T^i = 100$ samples are added in each closed-loop iteration based on the estimated GMM parameters. Based on empirical tests the pre-specified number of mixtures in the GMM is fixed at $K = 5$.

The results show that for low complexity cases such as Flat Plate (1) and Top-Hat (2) the effects of using closed-loop sampling with continual or transfer learning gives only minor reductions in training samples for converging. As the complexity of the assembly cases increases and the effect of pre-trained weights of continual and transfer learning become significant reduction up to 50% in the number of required training samples for case (6) can be seen. This validates the need for scalable approaches required for training

high-dimensional assembly cases while leveraging the pre-trained models on low-dimensional assembly cases.

The performance measure of using continual learning for all T_n tasks are done by comparing R^2 on the T_i th task when learning is performed only till T_i th task ($R^2_{T_i, T_i}$) and when learning is performed till the T_n th task ($R^2_{T_i, T_n}$). Catastrophic forgetting for each task (CF_{T_i}) is quantified as the difference between performance in the aforementioned situations:

$$CF_{T_i} = R^2_{T_i, T_n} - R^2_{T_i, T_i} \quad (19)$$

Negative value of CF_{T_i} means catastrophic forgetting of previous cases while positive values mean that learning new tasks has improved the performance of previous tasks. Table 5 summarizes the results. Average Generalized Performance $\frac{1}{T_n} \sum_{T_i} R^2$ and Average Catastrophic Forgetting $\frac{1}{T_n} \sum_{T_i} CF_{T_i}$ are summarized and highlight generalized performance (95%) greater than the required threshold while with average forgetting of only up to 2.1%.

TABLE 5. CLIP framework performance.

Assembly Case	$R^2_{T_i, T_n}$		$R^2_{T_i, T_i}$		CF_{T_i}	$\Delta T_{s=(tr)}$
	Mean	SD	Mean	SD		
(1) Flat Plate	0.93	0.01	0.96	0.02	-0.03	0.33
(2) Top-Hat	0.91	0.02	0.95	0.01	-0.04	0.57
(3) Door Halo Reinforcement	0.98	0.01	0.98	0.01	0.00	0.62
(4) Door Inner and Hinge	0.97	0.01	0.98	0.02	-0.01	0.68
(5) Cross Member	0.95	0.02	0.98	0.01	-0.03	0.59
(6) Cross Member	0.96	0.03	0.96	0.03	0.00	0.57
Average Generalized Performance = 0.95						
Average Catastrophic Forgetting = -0.02						
Average Convergence Improvement = 0.56						

The convergence is measured by the number of samples (S^{T_s}) for training for the given training scenario T_s .

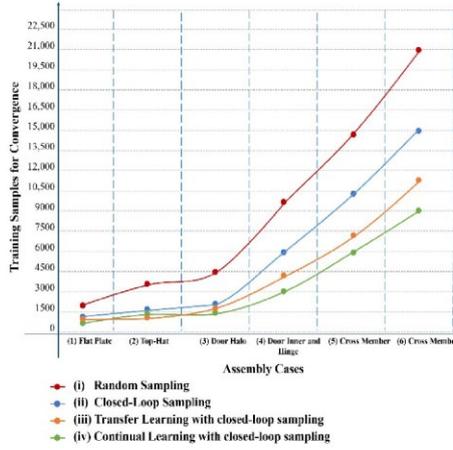


FIGURE 6. Convergence comparison of all assembly cases in all training scenarios.

Improvement in convergence ΔT_s for a training scenario T_s is measured as the percentage difference in training samples required for convergence as compared to the baseline training scenario which for this case is (i) random sampling

$$\Delta T_s = \frac{S^{T_s} - S^{T_{s=(i)}}}{S^{T_{s=(i)}}} \quad (20)$$

Table 5 also summarizes the improvement in convergence for the CLIP framework i.e. improvement in convergence when using training scenario (iv) Continual Learning with Closed-Loop Sampling $\Delta T_{s=(iv)}$. Fig. 7 aims to compare loss in performance (CF_{T_s}) with improvement in convergence ($\Delta T_{s=(iv)}$). Overall, across the six cases, the proposed CLIP framework provides 56% improvement scalability as quantified by improvement in convergence with a loss in performance of only 2.1% as quantified by Catastrophic forgetting.

C. DISCUSSION: INTERPRETABILITY

The interpretability of the Bayesian 3D U-Net is done on two levels:

(1) By linking requirements of the MASs with functional elements of the Bayesian 3D U-Net architecture. These include:

(1a) *Object Shape Error Voxelization*: Shape error voxelization provides an intermediate 3D data structure linking mesh obtained from CAE simulation and point clouds obtained from 3D optical scanners. Voxelization ensures that both these data structures are converted to voxels and are hence, compatible with 3D convolution operations fulfilling: Requirement (i) high data dimensionality; and, Requirement (vi) dual data generation capabilities. The voxels are multi-channels with each channel corresponding to one component of shape error. The resolution of voxels depends on the

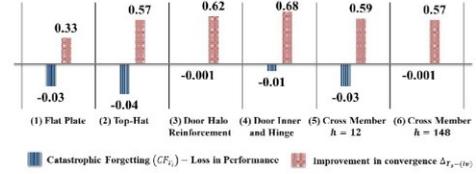


FIGURE 7. Loss in performance vs improvement in convergence comparison for CLIP.

required performance. Fig. 8 shows voxelization for one component of shape error at different resolutions. Low-resolution voxels capture global shape error patterns, as the resolution is increased, local shape error patterns are effectively captured. This also increases the discriminative capability required to differentiate between collinear shape error patterns, although this comes at a higher computational cost. Empirical studies have shown that there is no significant increase in performance above $(64 \times 64 \times 64)$ for RCA of MAS. Case (6) Cross Member Assembly ($h = 148$) is used for a sensitivity study. Object Shape Error Reconstruction Error and performance is compared against voxel granularity. The reconstruction error is less than 1% given $(64 \times 64 \times 64)$ or more granular voxels. The model performance does not increase over $R^2 = 0.96$ even with voxels as granular as $(96 \times 96 \times 96)$. Additionally, with increased voxel sizes above $(64 \times 64 \times 64)$ the minibatch size used during training has to be further reduced to 16 given GPU memory constraints which result in unstable model training and hence negatively impacts performance. Table 6 summarises the results of the sensitivity study.

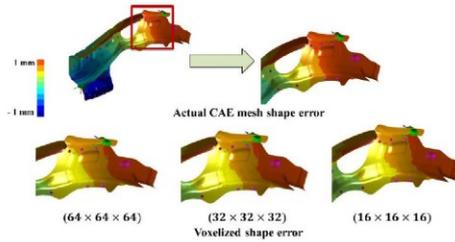


FIGURE 8. Object shape error voxelization.

(1b) *Encoder with Down Sampling Kernels*: As described earlier the Bayesian 3D U-Net architecture consists of four levels of the encoder and decoder models (Fig. 2). Each level of encoding consists of the down-sampling kernel (see Down-sampling kernel in Fig. 2). The kernel consists of 3D Max pooling, which is duplicated to a residual connection and encoding connections. The residual connection consists of a 3D convolution with a filter size of one and a stride length of one in all three dimensions. The encoding connection consists of two 3D convolutions of filter size three and stride length one with ReLU activation in between.

TABLE 6. Voxel resolution sensitivity study.

Voxel Resolution	Reconstruction Error	Performance (R^2)		Minibatch Size
		Mean	SD	
$(16 \times 16 \times 16)$	9.2%	0.85	0.01	64
$(32 \times 32 \times 32)$	3.8%	0.91	0.01	32
$(64 \times 64 \times 64)$	1.4%	0.96	0.02	32
$(80 \times 80 \times 80)$	1.2%	0.96	0.04	16
$(96 \times 96 \times 96)$	0.9%	0.95	0.05	16

Then, the residual connection and the encoding connection are merged using element-wise addition. Finally, ReLU is applied before duplicating the output into the decoder input and next level encoder input. Overall, the down-sampling kernels in the encoder with consecutive 3D convolutions and pooling are essential for spatial correlation filtering, feature extraction, and non-linear transformations. Consecutive levels of the decoder extract more discriminating features from the high-resolution voxelized shape error input. The discriminative ability of the features increases at each consecutive encoder level thus, enabling accurate estimate of process parameters hence high root cause isolability. Each level of the encoder is also linked to the corresponding decoder, this enables transfer of features related to the part geometry and hence enables accurate estimation of upstream part shape error at the end of the decoder. Fig. 9 highlights the features extraction capabilities of different levels of the encoder, while lower levels focus on the entire part higher levels focus on the regions that contain the shape error. This enables fulfilment of requirements: (ii) non-linearity; (iii) collinearities; and, (iv) high faults multiplicity. The 3D Grad-CAMs for encoder levels provides interpretability by visualizing the extracted shape error features. The transparency provided in the extracted shape error features enables interpretability on why a particular root cause was isolated.

(1c) *Decoder with Up-sampling Kernels*: Each level of the decoder consists of the up-sampling kernel (see Up-sampling kernel in Fig. 2) and provides real-valued segmentation maps that estimate object shape error, i.e., the three components of deviation for each subassembly at the end of all upstream stations. Each level of the decoder consists of two input sources; the encoder input from the corresponding level encoder and the decoder input from the previous decoder level. The following operations are then performed: (i) up-sampling of the decoder input, which is duplicated and sent to the attention gate and feature concatenation layer; (ii) the attention gate [63] distills information from the encoder and then generates relevant features that are concatenated with the up-sampling output from (i); and, (iii) this concatenated feature set is duplicated to the residual connection and the decoder connection and similar operations as in the encoder layer are performed. The number of channels of the decoder output equal to the number of components of shape error multiplied by the number of upstream stations, while the granularity of the output is the same as the input voxel size. Various levels of the decoder aggregate

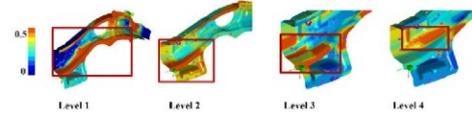


FIGURE 9. 3D Grad-CAMs for various encoder levels.

features from the corresponding encoder and previous decoder. This integrates part geometry features (as provided by the encoder) with the shape error features (as provided by the previous decoder) enabling accurate estimation of upstream part shape error. Different levels of decoder reconstruct shape error within different regions of the part. This enables fulfilment of requirement (viii) Fault Localization. Fig. 10(a) highlights the up-sampling capabilities of different levels of the decoder that enable estimation of object shape error of upstream assemblies. Level 2 reconstructs features of the pocket reinforcement subassembly while levels 3 and 4 reconstruct features of the cross-member reinforcement assembly. The 3D Grad-CAMs for decoder levels provides interpretability into the stations within which the fault is localized. Fig. 10(b) compares the actual upstream assembly shape errors as estimated using CAE simulation with those estimated by the decoder output.

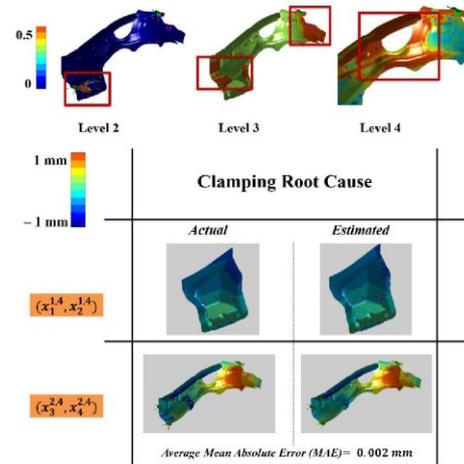


FIGURE 10. (a) 3D Grad-CAMs for various levels of the decoder. (b) Comparison of actual vs estimated upstream assemblies.

(1d) *Multiple Output Heads*: The model consists of two output heads one head estimates real-valued process parameters y^r as done in a regression setting while the second head estimates categorical/binary process parameters y^c as done in a multi-label classification setting. This is essential for MASS as they have a large number of process parameters inclusive of

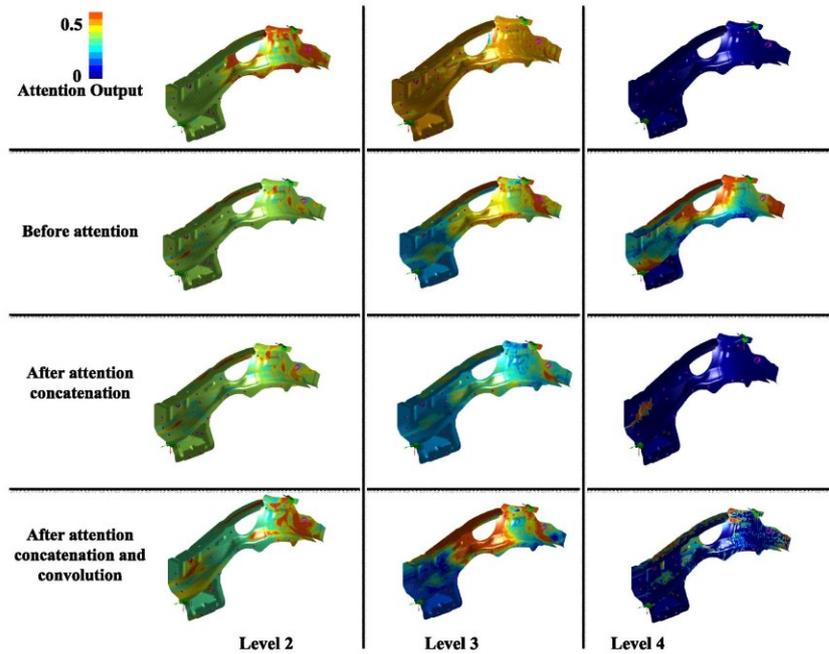


FIGURE 11. 3D Grad-CAMs for attention at various levels of the decoder.

(a) real-valued parameters for non-ideal parts and fixturing/tooling (within Positioning (P) and Clamping (C) stages of the PCFR assembly cycle); and, (b) binary parameters for joining operations (within the

Fastening (F) stage of the PCFR assembly cycle). The number of regression output nodes is equal to the number of real-valued parameters and the number of classification output nodes is equal to the number of binary process parameters. This enables fulfilment of requirement (vii) High Dimensionality and heterogeneity of process parameters.

(1e) *Attention Gate*: The soft-attention mechanism as proposed by Oktay et al. [63] is used between corresponding levels of the encoder and decoder. The attention approach allows the model to be specific to local regions. In the context of shape error estimation of assemblies, this helps the model focus on particular parts/subassemblies in each station. Adding of the attention gate increases in accuracy of upstream stations shape error estimation as the model learns where to look within the final assembly to estimate upstream sub-assemblies $[x^{1,4}, \dots, x^{N_s-1,4}]$. This decoder inclusive of attention gates improves performance for requirement (viii) Fault Localization. Fig. 11 shows 3D Grad-CAMs for areas of focus at different functions within the up-sampling kernel.

Attention 3D Grad-CAMs enables interpretability by providing insights into the regions focused by the decoder in estimating upstream shape errors.

(1f) *Bayesian Flipout Layers*: Given the uncertainties in the system and the availability of only a limited dataset, a deterministic estimate of function $f(\cdot)$ as shown in (2) is not feasible. The Flipout [64] layers leveraged in the encoder enable uncertainty quantification. These estimates of uncertainty integrate measures of confidence within isolated RC(s) and hence drive costly corrective actions [6]. This is realized by using Bayes-by-Backprop [65] which integrates backpropagation with variational inference [66] to estimate a posterior distribution $q_\theta(\omega)$ which is parameterized by θ over the neural network weights based on the pre-specified prior $p(\omega)$. This enables fulfillment of requirement (v) uncertainty quantification. The uncertainties are key elements of interpretability insights as they integrate a measure of confidence within the root cause estimates.

(1g) *Residual Connections*: Given the deep architecture of the model, vanishing gradients can be a major issue, hence residual [67] or skip connections are added within each down-sampling and up-sampling kernel that ensure effective propagation of gradients by providing a skip route.

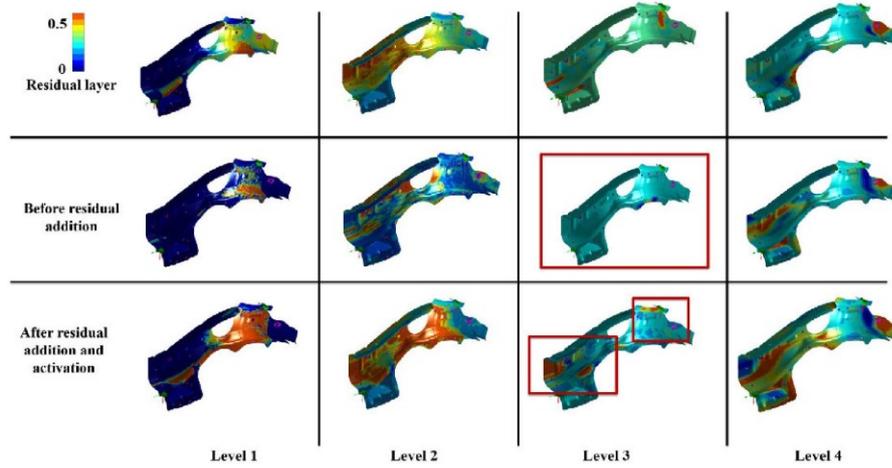


FIGURE 12. 3D Grad-CAMs for residual connection at various levels of the encoder.

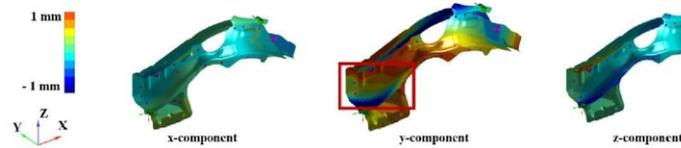


FIGURE 13(a). Part Variation Root Cause: Input Shape Error Components

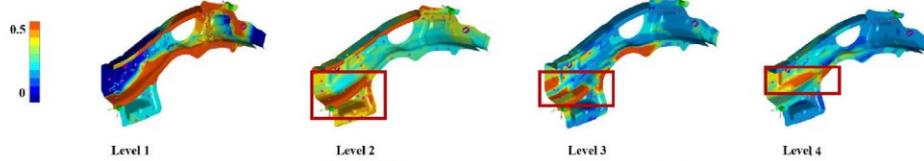


FIGURE 13(b). Part Variation Root Cause: Encoder 3D Grad CAMs

FIGURE 13. Part variation root cause.

Fig. 12 highlights the 3D Grad-CAMs for various stages of the residual connection. As seen (highlighted in red rectangle) in level three the layer before the residual connection has negligible activations due to skipping of the layer while the gradients become significant after the addition of the residual. The residual connections improve performance for requirements: (ii) non-linearity, (iii) collinearities, and (iv) high faults multiplicity.

(2) Using 3D Grad-CAMs to interpret the working of the architecture for different process parameter variations or root cause(s). The above 3D Grad-CAMs provide a *global*

level of interpretability by linking functions elements of the architecture with requirements of the MAS. The next local level of interpretability aims to provide transparency into the 3D Grad-CAMs for various levels of the encoder for key root cause scenarios. This links the shape error features extracted by each level of the encoder to estimate that particular root cause. Fundamentally, to interpret that the architecture is isolating a root cause correctly the features extracted by various levels of the encoder should correspond to the shape error patterns caused by that root cause. To validate this the cross member assembly (case (5)) is considered and the working

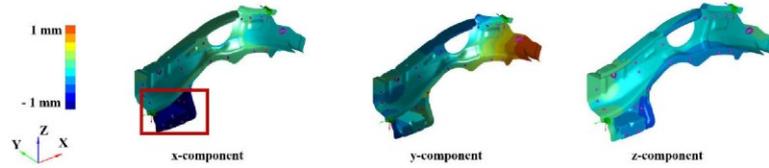


FIGURE 14(a). Pin Slot Displacement Root Cause: Input Shape Error Components

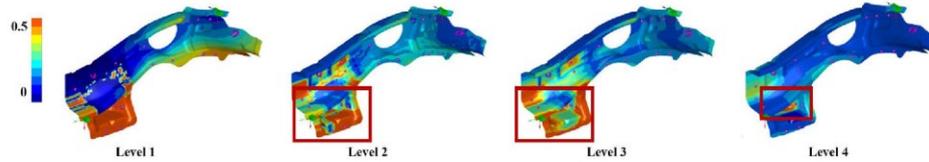


FIGURE 14(b). Pin Slot Displacement Root Cause: Encoder 3D Grad CAMs

FIGURE 14. Positioning root cause.

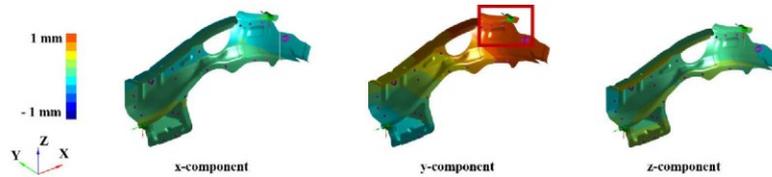


FIGURE 15(a). Clamping Root Cause: Input Shape Error Components

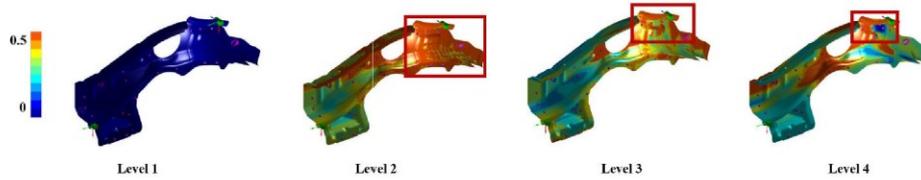


FIGURE 15(b). Clamping Root Cause: Encoder 3D Grad CAMs

FIGURE 15. Clamping root cause.

of the architecture in analyzed for five key root cause(s) scenarios:

(2a) *Part Variation Root Cause*: This is caused due to variation in upstream fabrication processes is estimated as variation in $y^m = y^1 = 2 \text{ mm}$. Fig. 13 represents the output of the assembly given the incoming part (cross-member) $n = 3$ has part variation. The region marked in red depicts a bend in the part that is unique to a part variation root cause [58]. The 3D Grad-CAMs as shown in Fig. 13 highlights, that the first encoder focuses around the entire part, the second encoder level can identify the edges near the bend

and the final encoder levels (three and four) can identify the region where the bend has occurred and hence accurately estimate y^1 as a part variation root cause.

(2b) *Positioning Root Cause*: This is caused by tooling installation and calibration error, or tooling deterioration due to gradual wearing out of fixture locators and is estimated as variation in $y^m = y^5 = 1 \text{ mm}$. They affect the part placement including orientation/reorientation and stability. The 3D Grad-CAMs as shown in Fig. 14 highlights that the encoder focuses around the entire part that has an error in orientation and estimates y^5 as the magnitude of the error.

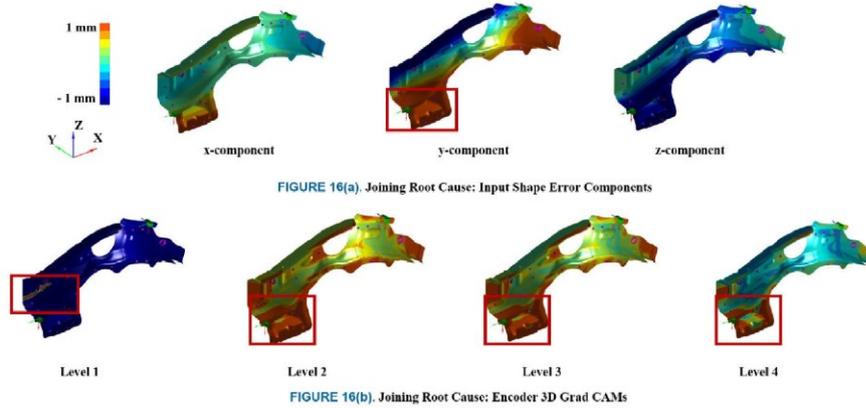


FIGURE 16. Joining root cause.

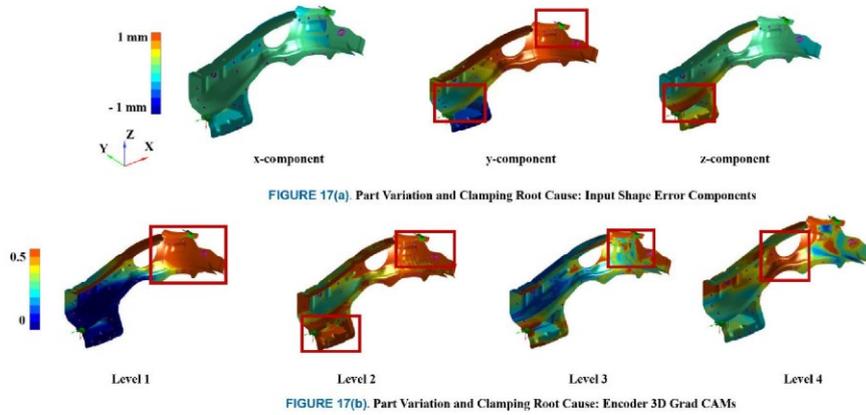


FIGURE 17. Part variation and clamping root causes.

(2c) *Clamping Root Cause*: This is caused by misalignment of the clamp in the y-direction and estimated as $y^m = y^{11} = 2 \text{ mm}$. They cause part bending of compliant parts. The 3D grad-CAMs as shown in Fig. 15 highlights that the encoder can focus on the local bend pattern at the location of the clamp and estimate y^{11} as the magnitude of the clamp misalignment in the y-direction.

(2d) *Joining Root Cause*: This is caused by misalignment of the joining tool (SPR) in the y-direction and estimated as $y^m = y^{12} = 2 \text{ mm}$. They lead to a defective joint between the two assemblies. The 3D grad-CAMs as shown in Fig. 16 highlights that the first level of the encoder can focus on the region of defective joint and the later levels focus on the

subassembly affected due to the defective joint and hence estimates y^{12} as the magnitude of joining tool misalignment in the y-direction.

(2e) *Part Variation and Clamping Root Causes*: This is caused when there is an upstream part variation ($y^m = y^1 = 2 \text{ mm}$) and misalignment of the clamp in the y-direction ($y^m = y^{11} = 2 \text{ mm}$). These lead to multiple simultaneous bends across the assembly. The 3D Grad-CAMs as shown in Fig. 17 highlights that various levels of decoder focus on all effected regions within the sub-assembly to simultaneously estimate multiple root causes. This capability is crucial in ensuring that deep learning models have high RCA capabilities even in scenarios when all process parameters

have variation and potentially are at fault (100% fault multiplicity). Such cases of high fault multiplicity cause various shape errors that are collinear (highly similar). The ability of architecture to simultaneously focus on multiple areas within the multi-channel voxelized input and localize various bends, twists and other shape error patterns which are potentially overlapping (interacting) and then relate them to the process parameter(s) causing it, makes it the ideal approach to do RCA of high-dimensional MASs with high fault multiplicity using granular 3D data structures such as mesh (CAE) or point clouds from 3D scanners.

VI. CONCLUSION & FUTURE WORK

The paper proposed a novel *closed-loop in-process (CLIP)* diagnostic framework underpinned algorithm portfolio to address the current limitations of scalability and interpretability. Scalability is enabled by leveraging closed-loop training integrated with uncertainty guided continual learning or transfer learning. The approach enables effective transfer of knowledge through invariant features between MASs thereby, achieving quicker convergence with 56% lesser training samples. The overall loss in performance was limited to only 2.1 % as quantified by average catastrophic forgetting (Table 5). Interpretability is enabled by leveraging 3D Grad-CAMs that provide insights into the functioning of key elements within the architecture and also relate features extracted by the architecture to shape error features within MAS. The visual interpretability explanations and uncertainty estimates integrate confidence hence, enabling *trust in black-box* deep learning models.

Scalability and interpretability are key challenges that must be solved to enable widespread adoption of deep learning methodologies in industrial environments. Key industrial application entails RCA of assembly processes of discrete components made of sheet metal parts used in automotive, aerospace or consumer products industries. These applications will leverage directly the CLIP diagnostic framework with the OSER approaches to enable scalable and interpretable root cause analysis and will be especially beneficial for processes with larger number of parts and/or larger number of assembly stations. The framework can also be leveraged for transfer of learning to different type of manufacturing processes such as stamping, machining and additive manufacturing that can be formulated using the proposed formulation of object shape error estimation for root cause analysis, this will lead to leveraging transfer and continual learning to other manufacturing processes that can be linked to assembly processes. Interpretability has been a major barrier preventing the adoption and deployment of deep learning models in the industry. The interpretability elements proposed by the work aim to eliminate the barrier and integrate context and confidence to the estimates hence, enabling wider adoption. Leveraging such automated and interpretable RCA models provides a transformative framework by ensuring early estimation and elimination of process variations before

they become defects thereby, helping to achieve Zero-Defect-Manufacturing and Right-First-Time.

Future work involves addressing the current limitations of the approach such as estimating dynamic changes in manufacturing systems and quantifying that as concept drifts or covariate shifts. Such changes when detected would then result in the model being fine-tuned such that the model accounts for the dynamic changes in the manufacturing system. This would enable lifelong learning for dynamic manufacturing environments. Further work also involves quantitative modeling of invariant features between different MASs. These invariant features can be linked to first principle models of the MASs and hence, further enhance scalability and interpretability.

REFERENCES

- [1] P. Franciosa, M. Sokolov, S. Sinha, T. Sun, and D. Ceglarek, "Deep learning enhanced digital twin for remote laser welding of aluminium structures," *CIRP Ann.*, vol. 69, no. 1, pp. 369–372, 2020.
- [2] Q. Qi and F. Tao, "Digital twin and big data towards smart manufacturing and industry 4.0: 360 degree comparison," *IEEE Access*, vol. 6, pp. 3585–3593, 2018.
- [3] W. G. Hatcher and W. Yu, "A survey of deep learning: Platforms, applications and emerging research trends," *IEEE Access*, vol. 6, pp. 24411–24432, 2018.
- [4] V. J. Shahi, A. Masoumi, P. Franciosa, and D. Ceglarek, "A quality-driven assembly sequence planning and line configuration selection for non-ideal compliant structures assemblies," *Int. J. Adv. Manuf. Technol.*, vol. 106, nos. 1–2, pp. 15–30, Jan. 2020.
- [5] S. Sinha, P. Franciosa, and D. Ceglarek, "Object shape error response using Bayesian 3D convolutional neural networks for assembly systems with compliant parts," *IEEE Trans. Ind. Informat.*, early access, Dec. 8, 2020, doi: 10.1109/TII.2020.3043226.
- [6] S. Sinha, P. Franciosa, and D. Ceglarek, "Object shape error response using Bayesian 3D U-net for multi-station assembly systems with non-ideal compliant parts," *IEEE Trans. Ind. Informat.*, 2020.
- [7] S. Shao, S. McAleer, R. Yan, and P. Baldi, "Highly accurate machine fault diagnosis using deep transfer learning," *IEEE Trans. Ind. Informat.*, vol. 15, no. 4, pp. 2446–2455, Apr. 2019.
- [8] S. Ebrahimi, M. Elhoseiny, T. Darrell, and M. Rohrbach, "Uncertainty-guided continual learning with Bayesian neural networks," 2019, *arXiv:1906.02425*. [Online]. Available: <http://arxiv.org/abs/1906.02425>
- [9] R. R. Selvaraju, M. Cogswell, A. Das, R. Vedantam, D. Parikh, and D. Batra, "Grad-CAM: Visual explanations from deep networks via gradient-based localization," in *Proc. IEEE Int. Conf. Comput. Vis. (ICCV)*, Oct. 2017, pp. 618–626.
- [10] G. Putnik, A. Sluga, H. ElMaraghy, R. Teti, Y. Koren, T. Tolio, and B. Hon, "Scalability in manufacturing systems design and operation: State-of-the-art and future developments roadmap," *CIRP Ann.*, vol. 62, no. 2, pp. 751–774, 2013.
- [11] P. Wang and R. X. Gao, "Transfer learning for enhanced machine fault diagnosis in manufacturing," *CIRP Ann.*, vol. 69, no. 1, pp. 413–416, Jan. 2020.
- [12] Y. Xu, Y. Sun, X. Liu, and Y. Zheng, "A digital-twin-assisted fault diagnosis using deep transfer learning," *IEEE Access*, vol. 7, pp. 19990–19999, 2019.
- [13] J. Yosinski, J. Clune, Y. Bengio, and H. Lipson, "How transferable are features in deep neural networks?" in *Proc. Adv. Neural Inf. Process. Syst.*, 2014, pp. 3320–3328.
- [14] Y. Bengio, "Deep learning of representations for unsupervised and transfer learning," in *Proc. ICML Workshop Unsupervised Transf. Learn.*, 2012, pp. 17–36.
- [15] H.-C. Shin, H. R. Roth, M. Gao, L. Lu, Z. Xu, I. Nogues, J. Yao, D. Mollura, and R. M. Summers, "Deep convolutional neural networks for computer-aided detection: CNN architectures, dataset characteristics and transfer learning," *IEEE Trans. Med. Imag.*, vol. 35, no. 5, pp. 1285–1298, May 2016.

- [16] R. Zhang, H. Tao, L. Wu, and Y. Guan, "Transfer learning with neural networks for bearing fault diagnosis in changing working conditions," *IEEE Access*, vol. 5, pp. 14347–14357, 2017.
- [17] J. Wang and C. Zhao, "Mode-cloud data analytics based transfer learning for soft sensor of manufacturing industry with incremental learning ability," *Control Eng. Pract.*, vol. 98, May 2020, Art. no. 104392.
- [18] F. Feng, R. H. M. Chan, X. Shi, Y. Zhang, and Q. She, "Challenges in task incremental learning for assistive robotics," *IEEE Access*, vol. 8, pp. 3434–3441, 2020.
- [19] G. A. Tahir and C. K. Loo, "An open-ended continual learning for food recognition using class incremental extreme learning machines," *IEEE Access*, vol. 8, pp. 82328–82346, 2020.
- [20] H.-J. Song and S.-B. Park, "Enriching translation-based knowledge graph embeddings through continual learning," *IEEE Access*, vol. 6, pp. 60489–60497, 2018.
- [21] A. Bielski and T. Trzcinski, "Understanding multimodal popularity prediction of social media videos with self-attention," *IEEE Access*, vol. 6, pp. 74277–74287, 2018.
- [22] H.-Y. Chen and C.-H. Lee, "Vibration signals analysis by explainable artificial intelligence (XAI) approach: Application on bearing faults diagnosis," *IEEE Access*, vol. 8, pp. 134246–134256, 2020.
- [23] L. Zhao, Y. Zeng, P. Liu, and G. He, "Band selection via explanations from convolutional neural networks," *IEEE Access*, vol. 8, pp. 56000–56014, 2020.
- [24] D. Ceglarek and J. Shi, "Dimensional variation reduction for automotive body assembly," *Manuf. Rev.*, vol. 8, pp. 139–154, Jun. 1995.
- [25] J. Shi and S. Zhou, "Quality control and improvement for multistage systems: A survey," *IIE Trans.*, vol. 41, no. 9, pp. 744–753, Jul. 2009.
- [26] Y. Liu, R. Sun, and S. Jin, "A survey on data-driven process monitoring and diagnostic methods for variation reduction in multi-station assembly systems," *Assem. Autom.*, vol. 39, no. 4, pp. 727–739, Sep. 2019.
- [27] D. W. Apley and J. Shi, "Diagnosis of multiple fixture faults in panel assembly," *J. Manuf. Sci. Eng.*, vol. 120, no. 4, p. 793, 1998.
- [28] M. Chang and D. C. Gossard, "Computational method for diagnosis of variation-related assembly problems," *Int. J. Prod. Res.*, vol. 36, no. 11, pp. 2985–2995, Nov. 1998.
- [29] K.-G. Yu, S. Jin, and X.-M. Lai, "Fixture variation diagnosis of compliant assembly using sensitivity matrix," *J. Shanghai Jiaotong Univ. Sci.*, vol. 14, no. 6, pp. 707–712, Dec. 2009.
- [30] R. Agrawal, J. F. Lawless, and R. J. Mackay, "Analysis of variation transmission in manufacturing processes—Part II," *J. Qual. Technol.*, vol. 31, no. 2, pp. 143–154, Apr. 1999.
- [31] C. Zou, W. Jiang, and F. Tsung, "A LASSO-based diagnostic framework for multivariate statistical process control," *Technometrics*, vol. 53, no. 3, pp. 297–309, Aug. 2011.
- [32] Y. Shang, F. Tsung, and C. Zou, "Statistical process control for multistage processes with binary outputs," *IIE Trans.*, vol. 45, no. 9, pp. 1008–1023, Sep. 2013.
- [33] J. Jin and J. Shi, "State space modeling of sheet metal assembly for dimensional control," *J. Manuf. Sci. Eng.*, vol. 121, no. 4, pp. 756–762, Nov. 1999.
- [34] Y. Ding, D. Ceglarek, and J. Shi, "Fault diagnosis of multistage manufacturing processes by using state space approach," *J. Manuf. Sci. Eng.*, vol. 124, no. 2, pp. 313–322, May 2002.
- [35] Y. Ding, J. Shi, and D. Ceglarek, "Diagnosability analysis of multi-station manufacturing processes," *J. Dyn. Syst., Meas., Control*, vol. 124, no. 1, pp. 1–13, Mar. 2002.
- [36] Y. Ding, S. Zhou, and Y. Chen, "A comparison of process variation estimators for in-process dimensional measurements and control," *J. Dyn. Syst., Meas., Control*, vol. 127, no. 1, pp. 69–79, Mar. 2005.
- [37] D. Ceglarek and P. Prakash, "Enhanced piecewise least squares approach for diagnosis of ill-conditioned multistation assembly with compliant parts," *Proc. Inst. Mech. Eng. B, J. Eng. Manuf.*, vol. 226, no. 3, pp. 485–502, Mar. 2012.
- [38] D. Ceglarek and J. Shi, "Fixture failure diagnosis for autobody assembly using pattern recognition," *J. Eng. Ind.*, vol. 118, no. 1, pp. 55–66, Feb. 1996.
- [39] D. Ceglarek and J. Shi, "Fixture failure diagnosis for sheet metal assembly with consideration of measurement noise," *J. Manuf. Sci. Eng.*, vol. 121, no. 4, pp. 771–777, Nov. 1999.
- [40] Y. G. Liu and S. J. Hu, "Assembly fixture fault diagnosis using designated component analysis," *J. Manuf. Sci. Eng.*, vol. 127, no. 2, pp. 358–368, May 2005.
- [41] D. Ceglarek, J. Shi, and S. M. Wu, "A knowledge-based diagnostic approach for the launch of the auto-body assembly process," *J. Eng. Ind.*, vol. 116, no. 4, pp. 491–499, Nov. 1994.
- [42] S. Du and L. Xi, "Fault diagnosis in assembly processes based on engineering-driven rules and PSOSAEN algorithm," *Comput. Ind. Eng.*, vol. 60, no. 1, pp. 77–88, Feb. 2011.
- [43] S. Du, J. Lv, and L. Xi, "An integrated system for on-line intelligent monitoring and identifying process variability and its application," *Int. J. Comput. Integr. Manuf.*, vol. 23, no. 6, pp. 529–542, Jun. 2010.
- [44] G. Beruvides, A. Villalonga, P. Franciosa, D. Ceglarek, and R. E. Haber, "Fault pattern identification in multi-stage assembly processes with non-ideal sheet-metal parts based on reinforcement learning architecture," *Procedia CIRP*, vol. 67, pp. 601–606, Jan. 2018.
- [45] S. Dey and J. A. Stori, "A Bayesian network approach to root cause diagnosis of process variations," *Int. J. Mach. Tools Manuf.*, vol. 45, no. 1, pp. 75–91, Jan. 2005.
- [46] Y. Liu and S. Jin, "Application of Bayesian networks for diagnostics in the assembly process by considering small measurement data sets," *Int. J. Adv. Manuf. Technol.*, vol. 65, nos. 9–12, pp. 1229–1237, Apr. 2013.
- [47] S. Sinha, P. Franciosa, and D. Ceglarek, "Object shape error response using Bayesian 3D convolutional neural networks for assembly system with compliant parts," in *Proc. 18th IEEE Int. Conf. Ind. Inf.*, Dec. 2020, pp. 104–109.
- [48] G. Arvanitis, A. S. Lalos, and K. Moustakas, "Robust and fast 3-D saliency mapping for industrial modeling applications," *IEEE Trans. Ind. Informat.*, vol. 17, no. 2, pp. 1307–1317, Feb. 2021.
- [49] L. D. Xu, C. Wang, Z. Bi, and J. Yu, "AutoAssem: An automated assembly planning system for complex products," *IEEE Trans. Ind. Informat.*, vol. 8, no. 3, pp. 669–678, Aug. 2012.
- [50] Q. Rong, J. Shi, and D. Ceglarek, "Adjusted least squares approach for diagnosis of ill-conditioned compliant assemblies," *J. Manuf. Sci. Eng.*, vol. 123, no. 3, pp. 453–461, Aug. 2001.
- [51] F. Adly, O. Alhoussein, P. D. Yoo, Y. Al-Hammadi, K. Taha, S. Muhaidat, Y.-S. Jeong, U. Lee, and M. Ismail, "Simplified subspace regression network for identification of defect patterns in semiconductor wafer maps," *IEEE Trans. Ind. Informat.*, vol. 11, no. 6, pp. 1267–1276, Dec. 2015.
- [52] A. Kendall and Y. Gal, "What uncertainties do we need in Bayesian deep learning for computer vision?" in *Proc. Adv. Neural Inf. Process. Syst.*, 2017, pp. 580–590.
- [53] X. Ding, Y. Chen, Z. Tang, and Y. Huang, "Camera identification based on domain knowledge-driven deep multi-task learning," *IEEE Access*, vol. 7, pp. 25878–25890, 2019.
- [54] K. Mannar, D. Ceglarek, F. Niu, and B. Abifaraj, "Fault region localization: Product and process improvement based on field performance and manufacturing measurements," *IEEE Trans. Autom. Sci. Eng.*, vol. 3, no. 4, pp. 423–439, Oct. 2006.
- [55] X.-W. Chen and X. Lin, "Big data deep learning: Challenges and perspectives," *IEEE Access*, vol. 2, pp. 514–525, 2014.
- [56] P. Franciosa, A. Palit, S. Gerbino, and D. Ceglarek, "A novel hybrid shell element formulation (QUAD+ and TRIA+): A benchmarking and comparative study," *Finite Elements Anal. Des.*, vol. 166, Nov. 2019, Art. no. 103319.
- [57] M. Stein, "Large sample properties of simulations using Latin hypercube sampling," *Technometrics*, vol. 29, no. 2, pp. 143–151, May 1987.
- [58] W. Huang and D. Ceglarek, "Mode-based decomposition of part form error by discrete-cosine-transform with implementation to assembly and stamping system with compliant parts," *CIRP Ann.*, vol. 51, no. 1, pp. 21–26, 2002.
- [59] D. P. Kingma and J. Ba, "Adam: A method for stochastic optimization," in *Proc. ICLR*, 2015, pp. 1–15.
- [60] Y. Wu and K. He, "Group normalization," in *Proc. Eur. Conf. Comput. Vis. (ECCV)*, 2018, pp. 3–19.
- [61] M. Abadi, A. Agarwal, P. Barham, E. Brevdo, Z. Chen, C. Citro, G. S. Corrado, A. Davis, J. Dean, M. Devin, and S. Ghemawat, "TensorFlow: Large-scale machine learning on heterogeneous systems," in *Proc. 12th USENIX Conf. Operating Syst. Design Implement.*, 2015, pp. 265–283.
- [62] S. Sinha, P. Franciosa, and D. Ceglarek. (2020). *Bayesian Deep Learning for Manufacturing*. [Online]. Available: https://github.com/sumitsinha/Deep_Learning_for_Manufacturing

- [63] O. Oktay, J. Schlemper, L. Le Folgoc, M. Lee, M. Heinrich, K. Misawa, K. Mori, S. McDonagh, N. Y. Hammerla, B. Kainz, B. Glocker, and D. Rueckert, "Attention U-net: Learning where to look for the pancreas," 2018, *arXiv:1804.03999*. [Online]. Available: <http://arxiv.org/abs/1804.03999>
- [64] Y. Wen, P. Vicol, J. Ba, D. Tran, and R. Grosse, "Flipout: Efficient pseudo-independent weight perturbations on mini-batches," in *Proc. 6th Int. Conf. Learn. Represent. (ICLR)*, 2018, pp. 1–16.
- [65] C. Blundell, J. Cornebise, K. Kavukcuoglu, and D. Wierstra, "Weight uncertainty in neural networks," in *Proc. 32nd Int. Conf. Mach. Learn. (ICML)*, vol. 2, 2015, pp. 1613–1622.
- [66] D. M. Blei, A. Kucukelbir, and J. D. McAuliffe, "Variational inference: A review for statisticians," *J. Amer. Stat. Assoc.*, vol. 112, no. 518, pp. 859–877, Apr. 2017.
- [67] K. He, X. Zhang, S. Ren, and J. Sun, "Deep residual learning for image recognition," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2016, pp. 770–778.



SUMIT SINHA received the B.Tech. degree in industrial and systems engineering from the Indian Institute of Technology (IIT) Kharagpur, in 2017. He is currently pursuing the Ph.D. degree in deep learning for manufacturing with the WMG, University of Warwick, Coventry, U.K. From 2017 to 2018, he worked as a Data Scientist with ZS Associates in the application of data science in pharmaceutical sales and marketing. He also worked briefly on data science projects with The University of Hong Kong (HKU) and Chongqing University, China. His research interests include data science, Bayesian deep learning, and digital twins of manufacturing assembly systems.



PASQUALE FRANCIOSA received the Ph.D. degree in mechanical engineering from the University of Naples Federico II, Italy, in 2010. He is currently an Assistant Professor with the University of Warwick, Coventry, U.K., and the Head of the Laser Welding Applications Laboratory, Warwick Manufacturing Group, the University of Warwick. He has been a Principal Investigator (PI) and a Co-Investigator (Co-I) on several funded projects with a total income to the University of Warwick of circa £2.8M (\$3.9), since 2015. He has authored or coauthored more than 80 articles. His research interests include smart manufacturing, process monitoring, closed-loop control, applications of machine learning/artificial intelligence, and multidisciplinary optimization, with specific attention on assembly systems and laser processes. He was a recipient of four best paper awards. He is also a member of the Editorial Board of the *ASTM Smart and Sustainable Manufacturing Systems* journal.



DARIUSZ CEGLAREK (Member, IEEE) received the Ph.D. degree in mechanical engineering (ME) from the University of Michigan–Ann Arbor, Ann Arbor, MI, USA, in 1994. He was a Professor with the IS&E, University of Wisconsin–Madison, Madison, WI, USA. He is currently an EPSRC Star Recruit Research Chair with the University of Warwick, Coventry, U.K. He has been a Principal Investigator (PI)/Co-Investigator (Co-I) on research grants of over £30M (\$41): NSF/NIST/EPSRC/InnovateUK/APC/EU-FP7/Curie and industry. He has authored or coauthored more than 200 articles and is listed by Stanford University among top 2% of the world's leading scientists. His research interests include smart manufacturing and data mining/AI for root cause analysis across design, manufacturing, and service. He is also a Fellow of the Collège International pour la Recherche en Productique (CIRP). He was a recipient of several best paper awards, the 2018 JLR "Innovista" Award for the most innovative "piloted technology," the EPSRC Star Award, and the NSF CAREER Award. He was an Associate Editor of *ASTM Smart and Sustainable Manufacturing Systems*, the *IEEE TRANSACTIONS ON AUTOMATION SCIENCE AND ENGINEERING*, and the *ASME Journal of Manufacturing Science and Engineering*.

Appendix E: Under Review

Journal Paper 3

S. Sinha, P. Franciosa, and D. Ceglarek, " Root Cause Analysis of Multi-Station Assembly Systems with Non-Ideal Compliant Parts using Bayesian 3D U-Nets," in review, IEEE Transactions on Automation Science and Engineering, 2021

Root Cause Analysis of Multi-Station Assembly Systems with Non-Ideal Compliant Parts using Bayesian 3D U-Nets

Sumit Sinha, Pasquale Franciosa, and Dariusz Ceglarek, *Member, IEEE*

Abstract—This paper proposes a novel approach, Object Shape Error Response for multi-station assembly system (OSER-MAS), to isolate root cause(s) (RC) of dimensional and geometric product shape errors. The proposed OSER-MAS approach leverages Bayesian 3D U-Net based Convolutional Neural Network (CNN) architecture, and is integrated with Computer-Aided Engineering (CAE) simulations. This enables (i) fault localization, (ii) isolation of RCs with uncertainty quantification and (iii) learning during the design phase of assembly system when no measurement data is available. The approach relates the estimated shape errors to three categories of process parameters: (i) incoming parts variation; (ii) place-clamp-fasten-release (PCFR) assembly cycle; and, (iii) part-to-part contact chain. The approach overcomes fundamental limitations of current approaches by (i) addressing root cause analysis of dimensional and geometric defects for MAS with non-ideal parts using deep learning on point cloud data; and, (ii) providing capabilities to estimate upstream shape errors and process parameters. Benchmarking is done using an industrial, automotive cross-member MAS.

Note to Practitioners—Performing root cause analysis (RCA) of quality defects induced by dimensional and geometric variations in MAS requires first, localizing the assembly station and then, isolating faulty process parameter(s) within the localized assembly station using data collected at the end of the MAS. Current industrial practice for RCA is centered on experience-based methods supported by statistical analysis of inspection data. However, this can lead to errors and deteriorated product quality, excessive machine downtime and scrap. This paper developed a deep learning approach that first estimates variations propagation within the MAS and next, identify process parameters in the stations which induced the unnecessary variation, thereby, enabling timely correction of RCs. The approach leverages point cloud data of parts shape errors collected at the end of the assembly by using CMMs or 3D scanners. It also leverages CAE simulation for model training that accounts for data scarcity within manufacturing environments.

Index Terms—Bayesian Deep Learning, U-Net, 3D CNN, Multi-Station Assembly

I. INTRODUCTION

THE manufacturing industry requires a transformative framework [1][2] that can enable scalable digitalization and integration of 3D scanners data, Artificial Intelligence (AI) algorithms, and Computer-Aided Engineering (CAE) simulations in order to migrate from the current sustainable

Hybrid Strategies (Industry 3.5) [3] that include semi-automated but cost-effective approaches to manage the potential socio-economic impacts of infrastructure disruptions, and support fully automated Industry 4.0 strategies to achieve *Zero-Defect-Manufacturing* and *Right-First-Time*. Integrating such strategies with *resilience* [4] can ensure that manufacturing systems are simultaneously efficient and resilient. Zero-Defect-Manufacturing and Right-First-Time aim to prevent quality defects while strategies for Resilient Manufacturing Systems (RMS) integrate capabilities within the system to recover quickly from such defects within a stipulated amount of time.

Currently, various manufacturing applications require multiple assembly stations to be placed in the production line [5], known as multi-station assembly systems (MAS). Each station comprises a robot, end effector with a joining head, and fixtures that can fail, resulting in diminished product quality. Additionally, many products are made of deformable non-ideal parts for which dimensional and geometric error/variation is one of the leading quality problems. Hence, it is crucial to develop approaches that ensure MAS can continue to produce high-quality products in the presence of non-ideal parts. A typical subsection of a Body-in-White (BiW) MAS is given in Fig. 1. Overall, the final assembly is constituted by multiple layers of assembly of incoming non-ideal parts using robotic fixtures and joining operations.

Dimensional and geometric variations are some of the biggest challenges faced by the manufacturing industry. Two-thirds of all quality issues in the automotive and aerospace sectors are caused by dimensional variations [6]. Past methods for Root Cause Analysis (RCA) of MAS have mostly relied on statistical, and machine learning approaches. Ding et al. [7][8] modelled multi-station assembly systems using state space approach and stream-of-variation analysis. Jin et al. [9] employed a Bayesian network for estimating fixture faults using measurement points. Bastani et al. [10][11] applied a spatially correlated Bayesian Learning algorithm for an underdetermined system by exploiting the spatial correlation of dimensional variation from various error sources. These approaches have been shown to have limitations in their applicability to non-linear and high dimensional manufacturing

This study was supported by the UK EPSRC project EP/K019368/1: “Self-Resilient Reconfigurable Assembly Systems with In-process Quality Improvement”, the UKRI open access block grant and the WMG-IIT scholarship. (Corresponding author: Sumit Sinha)

Sumit Sinha, Pasquale Franciosa and Dariusz Ceglarek are with the Digital Lifecycle Management (DLM) group at WMG, University of Warwick, Coventry, CV4 7AL, United Kingdom (U.K.). (e-mail: Sumit.Sinha.1@warwick.ac.uk; P.Franciosa@warwick.ac.uk; D.J.Ceglarek@warwick.ac.uk)

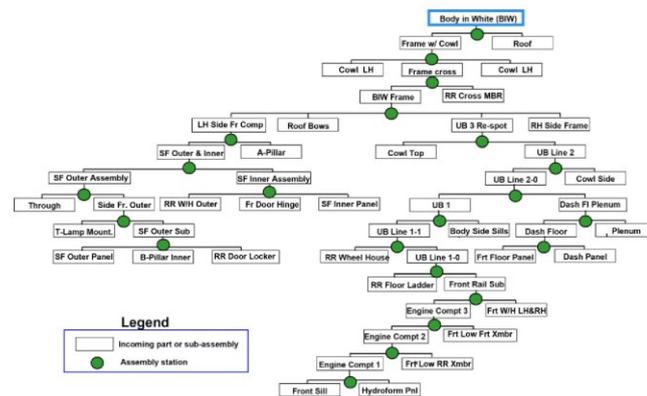


Fig. 1. Body-in-white multi-station assembly

systems [12], as these relied on linear models between process parameters and measurements of dimensional product quality [13]. Recently, Sinha et al. [14] proposed Bayesian 3D Convolutional Neural Network (CNN) architecture to relate shape errors to PCFR assembly process parameters within a *single* assembly station consisting of *ideal* compliant parts with only *fixturing related process parameters*. However, such an architecture cannot be used for MAS as these consist of a heterogeneous set of three categories of process parameters inclusive of *part variations, fixturing and joining* involving *non-ideal* compliant objects. The above process parameters can be classified within manufacturing/enterprise systems into [4]: (a) substance which refers to goods made of materials; and, (b) infrastructure, referring to production machines. The process parameters can be both real-valued or categorical. Additionally, single station assembly systems do not require fault localization as there is only one station. However, for MAS, the faults need to be localized to a single or a small number of assembly stations. The architecture proposed in [14] cannot estimate shape errors for upstream assembly stations and, therefore, cannot be used for fault localization and, eventually, root cause analysis for MAS. This limits the application of the previously proposed CNN methodology for RCA of MAS with non-ideal compliant parts. The methodology for RCA of object shape defects must satisfy a number of requirements [14] such as (i) *high data dimensionality* [15]; (ii) *non-linearity* [16]; (iii) *collinearities* [17]; (iv) *high faults multiplicity* [18][19]; (v) *uncertainty quantification* [20]; and, (vi) *dual data generation capabilities*. These have been previously addressed as requirements for assembly systems consisting of a single station [14]. However, the previous works cannot be used in MAS as the above challenges are significantly amplified due to station-to-station interactions, which affect shape variation propagation on multiple levels: (i) PCFR-to-part interactions as parts move through PCFR stages (shape error induced by fixturing and joining operations); (ii) part-to-part interactions within each station and further magnified between stations; and, (iii)

station-to-station interactions due to re-orientation errors between stations (change of fixture locating layout between stations). These interactions can cause object shape errors to accumulate in a non-linear manner. Different part-to-part interactions frequently lead to similar or collinear shape error patterns, thereby causing the system to become ill-conditioned. Additionally, MAS presents the following two requirements that cannot be fundamentally met by the previously proposed Bayesian 3D CNN architecture:

(vii) *High Dimensionality and heterogeneity* of process parameters [21], as MASs have a large number of process parameters inclusive of (a) real-valued parameters for non-ideal parts and fixturing/tooling (Positioning (P) and Clamping (C) stages of the assembly cycle); and, (b) binary parameters for joining operations (Fastening (F) stage of the assembly cycle). (viii) *Fault Localization for RCA* [22] as MAS have multiple stations; hence as compared to single-station assembly systems that required only isolating faulty process parameters within a single station [14], MAS require first localization of the assembly stations at fault followed by isolation of the faulty process parameter within the localized stations. Furthermore, faults originating in one station are propagated through all downstream stations. Hence for RCA within MAS, it is critical to localize the originating station of shape errors. The localized station can be further analyzed to isolate a specific process parameter at fault. Methods for RCA of single-station assembly system cannot be adapted to perform RCA of MAS. Even if the methods assume all stations as a single station, the RC isolation will result in multiple process parameters being indicated as faulty rather than the specific process parameter at fault in the originating station. Hence RCA of MAS requires first localization of the faulty assembly station followed by isolating faulty process parameters within the localized station. *This requires estimation of shape error of sub-assemblies at the end of each upstream station of MAS as well as estimation of all process parameters within all stations of the MAS.*

Overall, the above eight challenges need to be overcome on

two levels of modelling: (i) analysis model for the variation propagation by using high fidelity CAE simulations (estimate shape error given process parameters); and, (ii) synthesis models inherently needed for RCA to relate shape errors backwards chaining to abnormal process parameters under the scenarios described by the above requirements.

This paper proposes a novel approach, Object Shape Error Response for MAS (OSER-MAS), to isolate root cause(s) (RCs) of dimensional and geometric product shape errors (*synthesis*). The approach relates the estimated product shape errors to three categories of MAS process parameters as related to errors of (i) incoming non-ideal and deformable parts; (ii) assembly cycle in each assembly station consisting of place-clamp-fasten-release (PCFR); and, (iii) part-to-part contact chains. The paper addresses the requirements by proposing a Bayesian 3D-Net architecture and an RCA methodology that has superior performance for requirements (i) to (v) necessary for single-station assembly and provides capabilities to fulfil requirements (vii) and (viii) for MASs in the following way:

(1) Requirements (i)-(vi) by proposing a 3D fully convolutional 3D U-Net [23] based encoder-decoder network architecture. The encoder consists of probabilistic Bayesian Flipout layers [24], while the decoder consists of attention-based [25] layers. Overall, the model has three output heads, two at the end of the encoder and one at the decoder's end. The model is trained using dual data generation capabilities by utilizing high fidelity CAE simulator of the assembly process called Variation Response Method (VRM) [26] and 3D Optical scanners. This integration of probabilistic network and CAE simulation provides a step towards understanding causality and having a safer, self-aware and interpretable solution, essential for RCA solutions in industrial setups.

(2) Requirement (vii) by leveraging two output heads at the end of the encoder, one head estimates real-valued continuous process parameters as done in a regression setting. In contrast, the second head estimates categorical/binary process parameters as done in a multi-label classification setting.

(3) Requirement (viii) by leveraging the decoder of the proposed 3D U-Net based architecture to provide segmentation maps corresponding to the object shape error for previous stations. Volumetric segmentation has seen immense development in the past few years. The initially proposed 2D U-Net architecture has been extended for 3D applications such as MRI images and kidney scans [27]. In U-Nets, 3D volume is mapped to a latent space via successive convolutional and pooling layers; this latent representation is then upsampled and convolved until it reaches the size of the original volume and generates a per-voxel segmentation. *The proposed network extends this capability to estimate object shape errors at the end of each upstream station of the MAS.* Further, a methodology to perform RCA is proposed that leverages the upstream object shape error estimates for fault localization and the process parameters within the localized station for fault isolation.

The key contributions of the paper are as follows:

(1) A novel *Bayesian 3D U-Net Architecture* with a *probabilistic encoder*, an *attention-based decoder* and multiple output heads that enable simultaneous estimation of (i) object shape error components of upstream shape errors, (ii) real-

valued process parameters and (iii) binary process parameters. This serves as the critical kernel to enable RCA of MASs.

(2) A novel methodology for RCA of MAS using the outputs of the Bayesian 3D U-net to sequentially perform (i) *fault isolation*, (ii) *fault localization* and (iii) *fault identification*.

(3) *Verify and validate* the methodology by implementing it on an industrial, automotive cross member assembly consisting of 3 stations, 25 binary and 123 real-valued process parameters and four non-ideal parts using a *closed-loop training approach that leverages a CAE simulator known as VRM* [26].

(4) *Benchmark* the proposed methodology against (i) current state-of-the-art RCA models; (ii) traditional machine learning models in a multi-output regression or classification settings; and, (iii) state-of-the-art deep learning models used for image segmentation on three levels, namely, (i) *Process Parameter Estimation*; (ii) *Uncertainty Quantification*; and, (iii) *Upstream Object Shape Error Estimation* to highlight the ability to fulfil the aforementioned eight requirements.

The rest of the paper is organized as follows; Section II formulates the problem for MAS; Section III discusses the methods; Section V presents the industrial case study, and finally, conclusions and future work are summarized in Section VI.

II. PROBLEM FORMULATION

A. Multi-Station Assembly Systems

Multistage manufacturing systems (Fig. 2a) with a single assembly station have been previously represented as state-space models. Each stage corresponds to operations, namely *positioning*, *clamping*, *fastening* and *release* (PCFR as denoted in Fig. 2a). On the other hand, multi-station assembly systems can be considered a process tree consisting of multiple interconnected stations (Fig. 2b). The process tree consists of the final station after which data is collected and a set of *upstream* stations. The input to each station is a set of incoming parts (objects) that need to be assembled. Shape errors within the objects can be induced due to variation in any process parameter(s) within the upstream stations [7] and are propagated through the process tree to the final station.

Previously the formulation of object shape error propagation in single station (multistage) assembly has been proposed [14] that represents shape error \mathbf{x}_o^s for object o after stage s as:

$$\mathbf{x}_o^s = (\mathbf{P}_o^s, \mathbf{d}_o^s) \quad (1)$$

Where \mathbf{P}_o^s represents set of nominal points and \mathbf{d}_o^s represents the three components (x,y and z) of shape error for each nominal point.

As the goal of this paper is RCA for MAS, the paper extends the problem formulation for multistage systems by considering multiple stations. Stations are represented by s : $s=1, \dots, N_s$, where N_s represents the total number of stations within the system, and the previously represented stages are represented as \acute{s} : $\acute{s}=1,2,3,4$ as each station has four stages. Hence for MAS object shape error for object o after station s and stage \acute{s} can be represented as:

$$\mathbf{x}_o^{s,\acute{s}} = (\mathbf{P}_o^{s,\acute{s}}, \mathbf{d}_o^{s,\acute{s}}) \quad (2)$$

$\mathbf{x}_o^{0,0}$ correspond to shape errors from upstream part variations

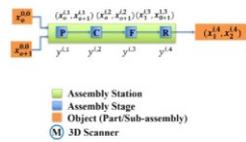


Fig. 2a. An assembly station with PCFR stages

which are also addressed as non-ideal part variations[5]. A set of objects (sub-assembly) after stage \acute{s} of station s is represented as $(\mathbf{x}_1^{s,\acute{s}}, \dots, \mathbf{x}_h^{s,\acute{s}})$.

Process parameters within a station and stage is represented as $\mathbf{y}^{s,\acute{s}}$, process parameters within a station is represented as \mathbf{y}^s , while the entire set of h process parameters can be represented as a vector \mathbf{y} . Potentially variation in each process parameter is a source of shape error and must be isolated as a root cause. In MAS, these process parameters are classified into three categories [28]: (a) Real-valued parameters of incoming parts (objects) variation as caused by upstream fabrication processes such as stamping, extrusions, etc.; (b) Real-valued process parameters related to PCFR stages of each assembly station. They represent any deviation from nominal in fixturing/tooling or joining operations; and (c) Binary joining-based process parameters in the fastening stage indicate the joint's success. The value is $\{1\}$ when joint is successfully completed or $\{0\}$ for an unsuccessful joint due to the excessive gap between objects to be joined or current failure in the tool. The overall vector of h process parameters \mathbf{y} can be split into a vector of r real-valued process parameters \mathbf{y}^r (category (a) and (b)) and c binary process parameters \mathbf{y}^c (category (c)).

For RCA of single station assembly systems performing RCA is equivalent to estimating process parameters of the station based on the shape error at the end of the station. Sumit et al. [14] proposed using a Voxnet [29] based 3D CNN approach to estimate a function $f(\cdot)$ that could estimate only real-valued process parameters as a function of shape error at the end of all stages:

$$\mathbf{y}^r = f(\mathbf{x}^{s=4}) \quad (3)$$

In MAS, shape error is propagated through the upstream assembly stations and shape error data is collected only at the end of the final station $\mathbf{x}^{s=N_s, \acute{s}=4}$. To perform RCA in such scenarios, firstly, the RC(s) must be localized to the originating station, followed by which RC(s) can be isolated within the localized station. Hence to comprehensively perform RCA shape errors for upstream stations $[\mathbf{x}^{1, \acute{s}}, \dots, \mathbf{x}^{N_s-1, \acute{s}}]$ need to be estimated, which enable localization of the station and further binary and real-valued process parameters must be estimated to isolate RC(s) within the station. Therefore the paper proposes a 3D U-Net CNN model which can be trained to learn a function $f(\cdot)$ that takes as input the combined object shape error at the end of the system $\mathbf{x}^{N_s, \acute{s}=4}$, i.e., after the final stage of the last station ($s = N_s, \acute{s} = 4$), and estimates the process parameters across the entire system and the object shape error for all objects

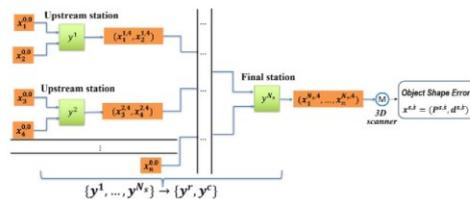


Fig. 2b. Process tree for multi-station assembly systems

at the end of the upstream stations:

$$[\mathbf{y}^r, \mathbf{y}^c, \mathbf{x}^{1, \acute{s}}, \dots, \mathbf{x}^{N_s-1, \acute{s}}] = f(\mathbf{x}^{N_s, \acute{s}}) \quad (4)$$

Further, an RCA methodology is proposed that leverages the outputs of $f(\cdot)$ to perform RCA.

III. METHODS

A. 3D Object Shape Error Voxelization

For the application of CNN based models to estimate $f(\cdot)$ as in (4), the shape error needs to be transformed to a regular structure hence, the output shape error $\mathbf{x}_o^{s,\acute{s}} = (\mathbf{p}_o^{s,\acute{s}}, \mathbf{d}_o^{s,\acute{s}})$ is voxelized to voxel grids $\{\mathbf{V}_{u,v,w}\}$ with discrete voxel coordinates (u,v,w) in the following way: for all points $\mathbf{p}_k = (x_k, y_k, z_k)$ that fall within a voxel grid $\{\mathbf{V}_{u,v,w}\}$ the maximum value of $\mathbf{d}_k = (\bar{x}_k, \bar{y}_k, \bar{z}_k)$ characterizes the features of the corresponding voxel grid and is represented as $\{\mathbf{V}_{u,v,w,d}\}$ [29]. Similar processing is done for previous station shape errors $\mathbf{x}^{1, \acute{s}}, \dots, \mathbf{x}^{N_s-1, \acute{s}}$ that need to be estimated by the U-Net model. As opposed to traditional binary image 2D segmentation maps that are generated by U-Net models, the proposed model estimates real-valued multi-channel 3D shape error maps for all previous stations and all components of object shape error.

B. Bayesian Neural Networks

A deterministic estimate of function $f(\cdot)$ as shown in (4) is not feasible as a limited dataset will be leveraged for mode training. Further data collection done using 3D scanners will inherently be noisy. Therefore, for uncertainty quantification, Bayesian learning enabled by variational inference is used to learn-weight distributions instead of only point estimates. This is realized by using Bayes-by-Backprop [30], which integrates backpropagation with variational inference [31] to estimate a posterior distribution $q_\theta(\omega)$ which is parametrized by θ over the neural network weights based on the pre-specified prior $p(\omega)$. The posterior weight distribution parameters θ are obtained by minimizing the variational free energy cost function known as the expected lower bound (ELBO). ELBO includes sum of (i) Kullback-Leibler (KL) divergence (6) which measures the distance between the prior and posterior and (ii) negative log-likelihood (NLL) of the dataset which measures the goodness-of-fit of the model.

$KL(q_\theta(\omega)||p(\omega)|\mathbf{x}^s, \mathbf{y}) = \int q_\theta(\omega) \log q_\theta(\omega)/p(\omega)|\mathbf{x}^s, \mathbf{y} d\omega$ (5) While training in a stochastic manner using a mini-batch of examples, the Flipout [24] estimator is used to solve the challenge associated with similarly sampled weights within a mini-batch. Flipout achieves ideal variance reduction by sampling weights pseudo-independently for each example.

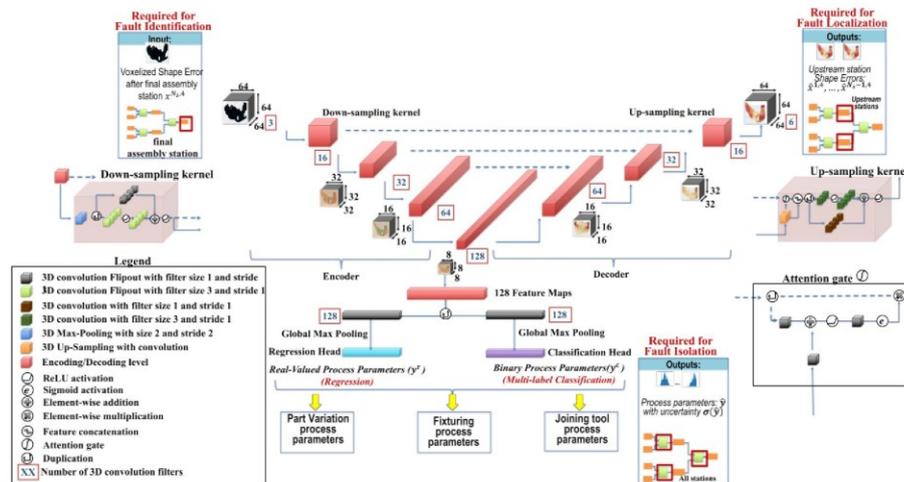


Fig. 3. Bayesian 3D U-Net Architecture, left: down-sampling kernel, right: up-sampling kernel, right-down: attention gate

C. Bayesian 3D U-Net Convolutional Architecture

The paper extends encoder-decoder u-net architectures to design an architecture that can accurately estimate the outputs for $f(\cdot)$ as shown in (4) (Fig. 3). The architecture consists of an encoder, a decoder and two output heads at the encoder's end.

The encoder consists of four levels (Fig. 3). Each encoder level consists of down-sampling kernels (Fig. 3 left). The first level takes as input the voxelized shape error ($x^{N_s, A}$) while subsequent levels take as input the output of the previous level. A 3D Max pooling is performed within the kernel, which is duplicated to a residual and encoding connection. A 3D convolution operation with a filter size of one and a stride length of one is performed within the residual connection. Two 3D convolutions (with ReLU activations) of filter size three and stride length one are performed within the encoding connection. These are further merged using element-wise addition followed by ReLU activation. The merged features are then duplicated and given as input into the corresponding level decoder and next level encoder. Overall, the encoder enables spatial correlation filtering, feature extraction, and non-linear transformations. Consecutive levels of the encoder extract more discriminating features from the high-resolution voxelized shape error input. The discriminative ability of the features increases at each consecutive encoder level. The link to the corresponding decoder enables transfer of features related to the part geometry, thus, enabling accurate estimation of upstream part shape error at the decoder's end. *From a MAS engineering perspective, the encoder can be considered as a feature extractor that enables the synthesis of the MAS, i.e. it extracts the necessary shape error features from the output of the MAS that can be leveraged to estimate process parameters and upstream shape errors.*

The decoder levels consist of up-sampling kernels (Fig. 3, right). Each level of the decoder takes two inputs, the encoder input from the corresponding level encoder and the decoder input from the previous decoder level. The decoder input is then up-sampled using upsampling and 3D convolution operations. The operations collectively upsample and add necessary detail into the decoder input. The upsampled features are duplicated and sent to the attention gate and feature concatenation layer. The attention gate [25] distills features from the corresponding encoder and then generates relevant features that are concatenated with the upsampled features. The concatenated feature set is duplicated to the residual connection, and the decoder connection and similar operations as in the encoder layer are performed. The decoder output dimension is equal to the number of components of shape error multiplied by the number of upstream stations, and the voxel granularity of the output is equal to the input size. Each level of the decoder consolidates part geometry features (corresponding encoder input) with the shape error features (previous decoder input), enabling accurate estimation of upstream shape errors. In the context of object segmentation, the decoder outputs real-valued segmentation maps that estimate three components of shape errors for all upstream stations. *From a MAS engineering perspective, the decoder can be considered as a feature consolidator that enables the synthesis of real-valued segmentation maps that estimate upstream shape errors ($x^{1, A}, \dots, x^{N_s-1, A}$).*

The architecture consists of two output heads at the end of the encoder; one head estimates real-valued process parameters y^r as done in a regression setting, while the second head estimates binary process parameters y^c as done in a classification setting. The feature maps at the end of the decoder are given as input to

both heads. 3D convolution and global-max pooling operations are performed within each head, followed by a Dense Flipout layer operation with 64 nodes and ReLU activation. The number of regression output nodes in the regression head is equal to the number of real-valued parameters. The number of classification output nodes is equal to the number of binary process parameters. *From a MAS engineering perspective, the output-heads can be considered enablers for RCA across multiple domains of process parameters, including (a) real-valued parameters for non-ideal parts and fixturing/tooling; and (b) binary parameters for joining operations.*

The attention gate (Fig. 3 right-down) at each level of the decoder enhances the architecture's ability to perform synthesis of MAS. As proposed by Oktay et al. [24], the soft-attention mechanism is used between corresponding levels of the encoder and decoder. Attention enables the model to be specific to local regions. The attention mechanism allows the model to reweight the input features based on a given set of features. The mechanism leverages a convolutional layer with the output having the same dimensions as the number of incoming features. The outputs of the network go through a sigmoid activation to generate values that lie between 0 and 1 and are known as attention weights. These attention weights are multiplied by the input features, which reweights the features and helps the model be specific to certain input features. In the proposed 3D U-net, the attention weights are generated based on the input from the previous decoder level. These are then used to generate attention weights which are used to reweight input features coming from the corresponding encoder. This helps the model focus on particular parts/subassemblies as the incoming features about the geometry of the part/subassembly is reweighted based on the previous level decoder features. The attention gate increases model performance as it learns on which part/subassembly to focus on to estimate upstream shape errors accurately. *From a MAS engineering perspective, attention gates improve performance for upstream synthesis and provide enhanced and calibrated estimates of upstream shape errors.* Fig. 4 aims to summarize the link between the functionalities of different elements of the 3D U-net architecture and the MAS engineering challenges that they fulfil. The residual connections also enhance synthesis performance by eliminating the vanishing gradients problem using residual [32] or skip connections within down-sampling and upsampling kernels, ensuring gradient propagation. Finally, the architecture leverages Bayesian layers within the architecture. Flipout [24] layers within the encoder and Bayes-by-Backprop [30] which combines backpropagation with variational inference, are leveraged for uncertainty quantification. *From a MAS engineering perspective, uncertainty estimates integrate confidence measures within isolated RC(s) and drive costly corrective actions* [14]. The previously proposed architecture [14] for RCA of assembly systems only consisted of an encoder with a regression head limiting its application scope to single-station assembly with only real-valued process parameters. The proposed architecture includes multiple output heads and a decoder that broadens the OSER-MAS approach's application scope to *assembly systems with multiple stations with both real-valued and binary process parameters.*

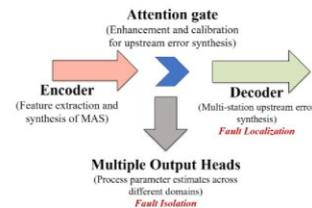


Fig. 4. Linking Bayesian 3D U-Net with MAS engineering

D. Architecture Design Decisions

The proposed architecture was designed by making key changes to the base architecture proposed for 2D biological image segmentation [23], which was later extended for 3D images [27] and further incorporated residual connections [33]. The design changes were done based on object shape error estimation requirements while accounting for standard guidelines of 3D U-Net architecture optimizations. Architecture hyperparameter optimization using grid search exhaustively searches for the best set of hyperparameters and has high computational costs [34]; hence, to perform this optimization in a computationally feasible manner key decisions are taken sequentially. The first key decision involved the total depth (Number of levels) N_L and the number of filters in each level $l = 1, 2, \dots, N_L$. The U-Net standards require the number of filters in each level to be constraint at values 2^{F+l} for the l^{th} level. Grid search was performed for $N_L = \{3, 4, 5\}$ and $F = \{2, 3, 4\}$ to determine the best configuration. Fig. 5 visualizes the results of the grid search. $N_L = 4, F = 3$ gave the best performance (boxed in red) in terms of the minimum value of the weighted loss function. The weighted loss function includes the losses all the outputs of the network including real-valued (regression) and binary process parameters (classification) and upstream shape errors (real-valued segmentation) is discussed later in model training. The mathematical formulation of the loss function is discussed later section (9). The next decision step involved experimenting with the filter size $F_s = \{3, 5, 7\}$. For computation feasibility, the filter size was kept universal across the network. $F_s = 3$ gave the best performance given the higher resolution and more granular feature extraction due to small filter size. Residual connections were added in each level of the encoder and decoder, this increased performance as these help in efficient propagation of gradients in deep networks while model training. To further increase performance experimentation was done by adding the soft-attention mechanism as proposed by Oktay et al. [25] between corresponding levels on the encoder and decoder. The attention approach allows the model to be specific to local regions. In the context of shape error estimation of assemblies, this helps the model focus on particular parts/subassemblies in each station thereby, enabling fault localization. Adding of the attention gate provides a significant increase in accuracy of upstream stations shape error estimation $[x^{1,4}, \dots, x^{N_s-1,4}]$. The final step involved replacing standard layers in the encoder layers with Flipout layers for uncertainty quantification. As opposed to [35] which leverages Flipout layers in the decoder our architecture uses Flipout layers in the encoder as this enables uncertainty

quantification for the process parameters that are estimated at the end of the encoder. For the regression and classification heads, experiments were done using global average pooling and global max pooling. Global Max pooling gave superior performance given the ability to extract the most significant feature. After this, the KL penalty for uncertainty quantification was added in the weighted loss function as shown in (4), and the model was trained. Experiments are done with three priors in the Flipout layers, namely, standard normal, scale mixture and spike and slab. An uncertainty calibration study that involved a comparison of model error and model uncertainty was conducted to ensure that the model predicts high uncertainty for samples with high error and vice-versa. Standard Normal prior gave the most calibrated uncertainty measure compared to other priors and was selected as the prior distribution for all weights in the encoder.

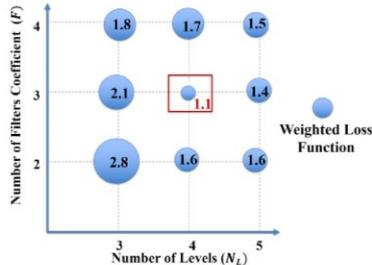


Fig. 5. Grid search results for U-Net architecture optimization; $N_L = 4$, $F = 3$ (boxed in red) is selected as the best model architecture

E. Closed-Loop Sampling, Training and Deployment

The architecture $f(\cdot)$ is trained within an iterative closed-loop framework proposed by Sinha et al. [28] using data generated by VRM, a CAE simulator. The framework enables efficient sampling of training data from VRM using uncertainty estimates hence reducing the overall data generation time. The closed-loop sampling dynamically samples data using a gaussian mixture model-based sampling distribution. The parameters of the sampling distribution are determined based on the error and uncertainty of previous sampling iterations. The approach has been shown to converge with 56% lesser training samples with a loss in performance of only 2.1%, hence reducing the key time-intensive step [28], i.e., VRM simulation. The key steps of the closed-loop framework are summarized below and shown in Fig. 6:

Step 1 – Sampling: The sampling is initiated using Latin hypercube sampling (LHS). Further sampling is done on the basis of the uncertainty and error of the previous iterations while ensuring that the sample generation has a degree of randomness to prevent overfitting [28].

Step 2 – VRM Simulation: The set of process parameters generated during sampling are used as input to the VRM to simulate the assembly process and generate the output mesh from which the shape errors are extracted after each desired station of the MAS.

Step 3 – Model Training and Evaluation: The model is trained using a weighted sum of four loss functions to estimate

process parameters, shape errors and uncertainty. They include:

- i) *Negative Log-likelihood* for the real-valued process parameters \mathbf{y}^r that are modelled using a multivariate normal with a diagonal matrix to quantify aleatoric uncertainty. Aleatoric uncertainty (known-unknowns) quantifies uncertainty due to factors such as noise, while epistemic uncertainty (unknown-unknowns) quantifies uncertainty due to model structure and limited data availability.

$$L^1 = 1/2 [\ln(|\Sigma|) + (\mathbf{y}^r - \boldsymbol{\mu})^T \Sigma^{-1} (\mathbf{y}^r - \boldsymbol{\mu}) + h \ln(2\pi)] \quad (6)$$

- ii) *Binary cross-entropy* for the binary process parameters \mathbf{y}^c

$$L^2 = \sum \mathbf{y}^c \cdot \ln \hat{\mathbf{y}}^c + (1 - \mathbf{y}^c) \cdot \ln (1 - \hat{\mathbf{y}}^c) \quad (7)$$

- iii) *Mean squared error* for the decoded object shape error
$$L^3 = \sum (\mathbf{x}^{s,s} - \hat{\mathbf{x}}^{s,s})^2 \quad (8)$$

- iv) *KL divergence* (L^4) as shown in (5) for all network weights and biases distribution parameters to quantify epistemic uncertainty. The final loss function (L) used while training is a weighted sum of all the above-specified loss functions:

$$L = \sum_{l=1}^4 w^l \cdot L^l \quad (9)$$

In usual practice, the values for these weights for the loss function are determined by empirical tuning, considering them as hyperparameters for optimization, but this can result in the requirement of exhaustive computational budget and time. The paper leverages the homoscedastic uncertainty for multi-task learning approach [36]. The approach estimates the weights during the training process without the need for any manual tuning. The approach aims to quantify the task-based uncertainty depending on the data and leverages the uncertainty estimate as the basis for weighting losses in a multi-task learning problem. The approach has been shown to provide superior performance in multi-task settings involving regression, classification and semantic segmentation as compared to performing each task individually. Leveraging this approach ensures that extensive fine-tuning is not required for training the model on different MAS. The loss function weights $\{w^1, w^2, w^3\}$ are generated using this approach. Monotonic KL annealing is used for w^4 to ensure that the model first learns the necessary estimation capabilities for the process parameters and upstream shape errors before the KL penalty is applied for epistemic uncertainty quantification. This also stabilizes the training process. Adam optimizer [37] is used for training, and exponential decay of the learning rate is applied. A Minibatch size of 32 is used, and the model is trained for 300 epochs. Group normalization [38] with four groups is used after each convolutional layer. This prevents overfitting and accounts for small minibatch size due to GPU memory size constraints and aids in stabilizing the training process.

For evaluation, Monte Carlo (MC) sampling from the model is done, and the sample means $\bar{\mathbf{y}}$ and standard deviations $\sigma(\bar{\mathbf{y}})$ are estimated for all real-valued and binary process parameters. $\sigma(\bar{\mathbf{y}})$ represents the uncertainty, while the fixed scale parameters of the predictive distribution for real-valued process parameters represent the known aleatoric uncertainty [39]. Similarly, means $\bar{\mathbf{x}}^{s,s}$ and standard deviations $\sigma(\bar{\mathbf{x}}^{s,s})$ of

all object shape errors at upstream stations are calculated. The model is trained until the error is below the set threshold. The thresholds are set based on assembly requirements. For example, in the automotive body assembly process, the threshold is set considering tolerances are within [-1mm,1mm], and the 3D optical scanner used has a repeatability of 0.05 mm and accuracy within 0.15 mm. During training, the uncertainty of the object shape errors is estimated but is not considered for sampling as the VRM requires process parameters as input.

Final evaluation and comparison for all model outputs are made using the known true values and the estimated means. Real-valued process parameters are evaluated on Mean Absolute Error (MAE) and R-squared (R^2). Binary process parameters are evaluated on Accuracy and Receiving Operating Characteristics – Area Under Curve (ROC-AUC). Given object shape errors are real-valued, they are also evaluated on MAE, R^2 and Root Mean Squared Error (RMSE).

Step 4 – Model Deployment: After training, the model can be deployed for an actual system. The product shape data is collected from the 3D scanner after the final station [40], and then the data is aligned to obtain the final shape error $\mathbf{x}_o^{s,s} = (\mathbf{p}_o^{s,s}, \mathbf{d}_o^{s,s})$ and finally, voxelized $V_{u,v,w,d}$ before it can be given to the trained model for estimating process parameters and shape error as coming from all upstream stations. Inferencing estimates the process parameters and shape errors for a given $\mathbf{x}_o^{s,s}$ using MC sampling from the trained model. Using these samples, process parameters (distribution mean) $\bar{\mathbf{y}}$ and their uncertainty (distribution standard deviation) $\sigma(\bar{\mathbf{y}})$ can be estimated. The sample mean $\bar{\mathbf{y}}$ is considered as the model estimate $\hat{\mathbf{y}}$, while $\sigma(\bar{\mathbf{y}})$ quantifies the uncertainty. Similarly, $\hat{\mathbf{x}}^{s,s}$ estimates the shape error and $\sigma(\hat{\mathbf{x}}^{s,s})$ the uncertainty in the estimated shape error. Further, fault localization and RCA can be done using these estimates. The work has been implemented using Python 3.7 and TensorFlow - GPU 2.1 [41] and TensorFlow Probability 0.9. In addition, a python library [42] has been developed to validate and replicate the results of the methodology. Two Nvidia Tesla V100 32 GB GPUs are used for model training and deployment.

For this paper, both the data generation and evaluation of the OSER-MAS methodology have been done using VRM. Although the VRM has been previously verified and validated for the simulation of assembly systems, constraints in the computational budget might inversely hamper the data quality due to modelling assumptions such as mesh granularity and boundary conditions. In such situations, the data generated might not be representative of a physical assembly system. *Uncertainty quantification (reg. (v)) capability* is crucial for the recognition of such situations as the model will indicate high uncertainty when the distribution of the physical data is different from the VRM data used while training. In such situations, the model should be fine-tuned using data collected from the physical MAS. The Bayesian nature of the proposed architecture enables effective fine-tuning using uncertainty-guided continual learning and ensures that the model can be continually adapted to changes in the MAS while minimizing catastrophic forgetting [28].

F. Root Cause Analysis Framework for MAS

After model deployment, the measured point cloud data

$[\mathbf{x}^{N_s-1}]$ and the estimates of the model $[\mathbf{y}^r, \mathbf{y}^c, \mathbf{x}^{1,4}, \dots, \mathbf{x}^{N_s-1,4}]$ are leveraged to perform RCA for MASs. Building on previous RCA models that leveraged point-based measurement systems [43], the paper proposes three key steps, namely: (i) fault identification; (ii) fault localization; and, (iii) fault isolation. The proposed approach (Fig. 6) can be leveraged to estimate single or multiple faults within the MAS.

(i) *Fault Identification:* The first is identifying if the MAS is at fault. In practice, all systems are at fault due to inherent variations and tolerances. Significant faults are identified by analysing the output $[\mathbf{x}^{N_s-1}]$ of the MAS. Two approaches can be leveraged, namely:

(a) *Threshold approach:* In this approach, a fault is identified if the measured point cloud is beyond a particular a set threshold. The thresholds are determined based on the tolerance limits of the assembled products. This approach is prone to false alarms as a single outlier in the output may indicate faults even when the no process parameter within the MAS is at fault.

(b) *Six-sigma approach:* In this approach, a fault is identified by analyzing the statistics of a sample of assembled products. The mean and variance of the output (\mathbf{x}^{N_s-1}) for a sample of products is calculated, a mean shift or a change in variance (heteroskedasticity) can be identified as a fault based on the significance level used. This approach is robust to outliers and can be leveraged for continuous improvement of the system.

(ii) *Fault Localization:* If the MAS has been identified at fault, the estimates of the upstream shape errors $[\mathbf{x}^{1,4}, \dots, \mathbf{x}^{N_s-1,4}]$ as obtained from the deployed model can be leveraged to localize stations within the MAS at fault. This entails identifying particular stations within the MAS process tree within which the object(s) (sub-assemblies) shape error becomes significant. The shape error estimates $[\mathbf{x}^{1,4}, \dots, \mathbf{x}^{N_s-1,4}]$ for all objects $o: o = 1, \dots, n$, are compared with the design nominal. If the shape error for the objects within the station exceeds the threshold (sub-assembly tolerances) at the end of a station but within the threshold for the previous upstream station, then the fault is localized to that particular station for the corresponding station object. This is done for all stations. The set of localized stations at fault is denoted as \mathbf{s}_F . Table I summarizes the steps used to localize assembly stations at fault. Using the proposed bottom-up approach ensures that the fault is localized to the originating station. This is crucial as MAS involve re-orientation/re-positioning of objects between stations [7] [43]. Localizing using a bottom-up approach ensures that the station within which the fault originated is localized before the object was re-oriented in later stations.

TABLE I. FAULT LOCALIZATION

Procedure:
List of localization stations: $\mathbf{s}_F = \{\}$
Repeat for all stations $i=0$ to N_s:
Compare $\mathbf{x}^{i,4}$ to nominal design
Compare $\mathbf{x}^{i+1,4}$ to nominal design
If: $\mathbf{x}^{i,4}$ within tolerance and $\mathbf{x}^{i+1,4}$ is beyond tolerance
Localize station $s = i + 1$
$\mathbf{s}_F \leftarrow \mathbf{s}_F \cup \{s = i + 1\}$

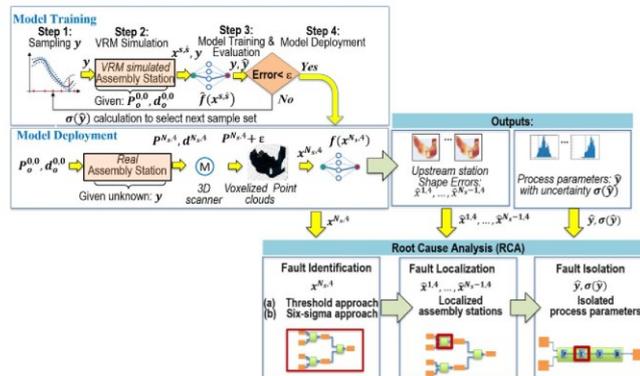


Fig. 6. Closed loop sampling, training and deployment with Root Cause Analysis

(ii) *Fault Isolation*: Fault isolation involves isolating which process parameters within localization stations \mathcal{S}_F are *root causes* and have a significant impact on the shape error of the final object (product). The set of process parameters (denoted a $\mathbf{y}^{s=\mathcal{S}_F}$) that lie within the localized stations \mathcal{S}_F are compared with the assembly process tolerances and isolated as RCs if they fulfil the below criteria (κ):

(a) All real-valued process parameters \mathbf{y}^r within the set of localized process parameters $\mathbf{y}^{s=\mathcal{S}_F}$ that are beyond the assembly process tolerances are isolated as RCs.

(b) All binary process parameters \mathbf{y}^c within the localized stations that are estimated as failed (e.g. for joining tool failure $\mathbf{y}^c = 0$) are isolated as RCs. Overall RCs can be denoted as a subset of process parameters within localized stations that :

$$RC \subseteq \mathbf{y}^{s=\mathcal{S}_F} \vee \mathbf{y}^{s=\mathcal{S}_F} \equiv \kappa \quad (5)$$

Within fault isolation, it is also crucial to consider the uncertainty of the process parameters $\sigma(\hat{\mathbf{y}})$ isolated as RCs drive costly corrective actions. High values of uncertainty indicate the trained model is not confident about the process parameter estimates, which may be due to noise in the measurement systems (*aleatoric uncertainty*) or lack of similar instances within the training set (*epistemic uncertainty*) [14]. In such scenarios, it is crucial to consider expert system knowledge to get conclusive insights about the isolated RCs

The OSER-MAS methods, requirements, MAS application and assumptions are summarized in Table II. The overall framework of OSER-MAS inclusive of closed-loop sampling, training and deployment and the RCA framework that leverages the outputs of the trained model to perform fault identification, fault localization and fault isolation is summarized in Fig. 6. In real-life applications, all process parameters within MAS have an inherent level of variation. The proposed approach for RCA using the estimates of the proposed architecture is crucial in differentiating benign faults (have no significant impact on the product shape error) from malignant faults (cause product shape error to go beyond assembly thresholds).

Another key consideration for RCA is the existence of a unique RC. Given the high dimensional process parameter space,

multiple shape error patterns may be caused by different combinations of RCs (known as collinear RCs [14]). Such scenarios will be very limited in the proposed approach. The input voxelized object shape error data is highly granular; hence, scenarios of different RCs giving the same voxelized object shape error are unlikely to happen. 3D CNN based approaches that leverage granular voxelized point cloud data have been shown to have high discriminative ability for various collinear RCs given the 3D deep learning capabilities [14]. This contrasts with previously proposed approaches that used simplified representations such as a vector of surface points or sensor readings (low granularity), leading to frequent scenarios when different RCs resulted in the same vector output. Additionally, uncertainty quantification enables the identification of such scenarios as a single shape error caused by multiple RCs would lead to the estimation of RC distributions that are multi-modal or have high variance. During such scenarios, engineering expertise can be applied to isolate RCs accurately.

IV. CASE STUDY: CROSS MEMBER ASSEMBLY

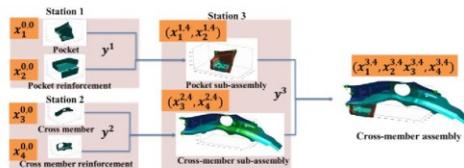
A. Multi-Station Assembly Setup

The selected cross member assembly consists of $N_s = 3$ stations and $n = 4$ non-ideal compliant parts namely, pocket, pocket reinforcement, cross member and cross member reinforcement (Fig. 7). The assembly is controlled by $h = 148$ process parameters (Fig. 8) \mathbf{y} including 123 real-valued parameters \mathbf{y}^r and 25 binary parameters \mathbf{y}^c . Table III summarizes the process parameter and their physical interpretation corresponding to RCs of the MAS. The point cloud of the final assembly is characterized by 11875 points. Shape error data is collected at the end of all three stations $\mathbf{x}^{1,4}, \mathbf{x}^{2,4}, \mathbf{x}^{3,4}$. The shape error after the final station ($\mathbf{x}^{3,4}$) is used as input while the process parameters \mathbf{y} and upstream stations shape errors $\mathbf{x}^{1,4}, \mathbf{x}^{2,4}$ are used as output. Before training all shape errors are preprocessed and voxelized to $(u, v, w, d) = (64, 64, 64, 3)$ voxel grids $V_{64,64,64,3}$. The

TABLE II. OSER-MAS METHODS, APPLICATION AND ASSUMPTIONS

OSER-MAS method	Requirement	MAS Application	Assumption
3D object shape error voxelization	(i) High data dimensionality	Enables processing of 3D object shape error	Availability of point cloud data
Encoder	(ii) non-linearity, (iii) collinearities, (iv) high faults multiplicity	Incorporates capabilities for handling the non-linearity and interactions due to multiple stages and stations within MAS	Sufficient discriminative information within the incoming point cloud data and sufficient modelling capabilities within the architecture
Bayesian model	(v) Uncertainty quantification	Enables uncertainty qualification for isolated RC for costly corrective action and enabling closed-loop sampling	Ability of bayes-by-backprop and Flipout to model uncertainty using a normal distribution as prior for each model parameter
Closed-loop sampling	(vi) Dual data generation capabilities	Enables training of the proposed model using a closed-loop framework that generates samples based on uncertainty	Ability of VRM (CAE simulator) to simulate data that is close to a physical MAS
Multiple output heads	(vii) High dimensionality and heterogeneity	Enables estimation of real-valued and categorical RCs across part variations, fixturing and joining	Process parameters are either real-valued or binary
Decoder with attention gate	(viii) Fault localization for RCA	Enables estimation of upstream shape error required to localize faulty stations to enable RCA	Sufficient modelling capabilities within the decoder architecture and the soft attention mechanism
RCA framework	Fault identification, fault localization and fault isolation	Leverages the process parameter and upstream shape error estimates to perform RCA	Six sigma indicators sufficient for fault identification

deviation features \mathbf{d} include deviations in all directions for all points $(\tilde{x}_k, \tilde{y}_k, \tilde{z}_k)$. For this particular case, the model has 123 output nodes in the regression head, 25 output nodes in the classification head, and the decoder outputs shape errors with dimension (64,64,64,6), corresponding to the shape errors at the end of two upstream stations. The training range for all real-valued process parameters is [-1 mm, 1 mm] while the validation and testing range is [-2 mm, 2 mm].



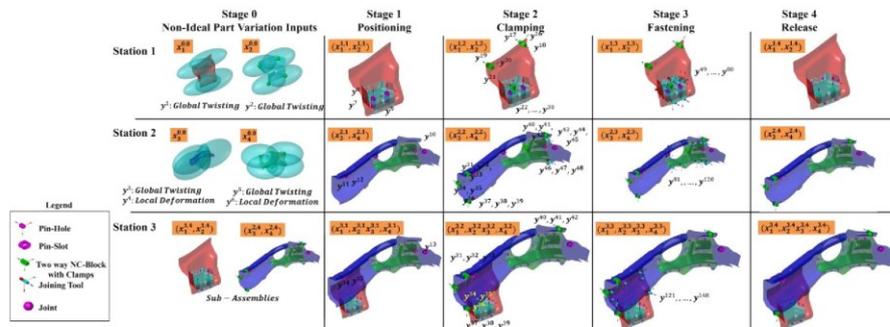


Fig. 8. Multi-station cross member assembly process parameters

closed-loop training based on the uncertainty estimates. The model is trained on the combined dataset, including all previous samplings, to ensure no catastrophic forgetting. The diagonal scale parameters for all process parameters in the covariance matrix are fixed at 0.005, assuming a fixed level of noise hence, constant aleatoric uncertainty. A validation set of 300 samples is generated. After each iteration, the trained model is evaluated on the validation set. During evaluation for each of 300 samples, 100 MC samples are drawn from the trained model. The sample means are considered the estimate for the process parameters, while the sample Inter-Quartile Range (IQR) quantifies the epistemic uncertainty. The closed-loop training is stopped when the average MAE across all real-valued process parameters for the validation set is below the threshold, which is selected to be 0.3 mm and the accuracy of all binary process parameters is above 90%. Overall, ten trials are done for comparison of model performance across all the requirements. The results and convergence of the closed-loop training are shown in Fig. 9.

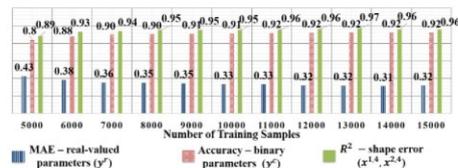


Fig. 9. Model results and convergence

B. Interactions

The above-mentioned requirements and the proposed solutions have various levels of interactions and hence can be aggregated to understand the tradeoff between the proposed solutions further and conduct effective benchmarking for one or multiple requirements collectively. Requirement (i) drives to build an input data structure that can effectively retain all information regarding the object geometry and components of shape error. 3D Shape error voxelization retains all such information and also enables convolution operations by

building a regular 3D structure. Using simpler data structures, dimensionality reduction or downsampling points will negatively affect performance for requirements (ii), (iii), (iv) and (vii) as the global geometry information and local shape error granularity will be reduced. Requirements (ii), (iii), (iv) and (vii) can be collectively considered as an indicator of the complexity of the MAS. Complex MASs with more stations and process parameters will have higher the non-linearity, collinearity and possible fault multiplicity within the system. Hence, benchmarking for all these are done collectively as *process parameter estimation* capabilities against various machine/deep learning approaches used for regression and classification. Requirement (v) enables quantifying the uncertainties and is fulfilled using probabilistic weights in the network. The quality of the uncertainty estimates (calibration) is directly affected by MAS complexity. An increase in the number of process parameters increases uncertainty within various estimation ranges. Benchmarking for requirement (v) is done as *Uncertainty Quantification* for different prior distributions assuming the total number of process parameters are fixed. Benchmarking for requirement (vi) is done by comparing the number of samples required for convergence for each methodology. Finally, benchmarking for requirement (vii) that aims to estimate upstream shape errors is done as *Upstream Object Shape Error Estimation* against deep learning methods used for segmentation.

C. Benchmarking: Process Parameter Estimation

Process parameter estimation capabilities are benchmarked considering requirements (i) to (vii) against three categories of methods specifically, (1) CNN based Deep Learning approaches; (2) Traditional Machine Learning approaches in a multi-output regression or classification setting; and, (3) currently used linear approaches. The results of the benchmarking are summarized in Table IV. Comparison for requirement (i) is made based on the data preprocessing technique used to handle the point cloud data dimensionality. Shape error voxelization retains the 3D structure and shape error features, while projection retains only 2D spatial structure. Flattening eliminates all information related to the spatial structure, while PCA also eliminates information explaining

TABLE IV. BENCHMARKING: PROCESS PARAMETER ESTIMATION

	Models	Requirement (i)	Requirements (ii),(iii),(iv) and (vii)								Requirement (vi)				
			MAE y^t		R^2 y^t		Accuracy y^c		AUC-ROC y^c		Uncertainty Estimates	Samples	CAE Simulation Time (minutes)	Training Time (minutes)	Total Time (minutes)
			Mean	SD	Mean	SD	Mean	SD	Mean	SD					
Proposed	OSER-MAS (Bayesian 3D U-Net)	3D Shape Error Voxelization	0.32	0.04	0.94	0.03	0.92	0.02	0.88	0.01	Standard Normal prior (corr. = 0.99)	Closed-loop (15000)	54000	1865	55865
	OSER-MAS (3D U-Net)	3D Shape Error Voxelization	0.32	0.01	0.94	0.02	0.91	0.01	0.87	0.01	No	30000	108000	1782	109782
Bayesian Deep Learning	OSER (Bayesian 3D CNN)	3D Shape Error Voxelization	0.35	0.05	0.91	0.03	0.88	0.01	0.84	0.02	Standard Normal prior (corr. = 0.91)	Closed-loop (16000)	57600	1802	59402
Deep Learning	PV CNN [50]	3D Shape Error Voxelization	0.36	0.02	0.90	0.03	0.86	0.01	0.81	0.03	No	30000	108000	1389	109389
	Pointnet++[51]	Uniformly downsampled set of points	0.37	0.01	0.91	0.04	0.84	0.02	0.80	0.02	No	30000	108000	1389	109389
	Multi-View 2D [52] CNNs (MV-CNNs)	6 face projection and 2D gridding	0.41	0.03	0.88	0.02	0.81	0.01	0.78	0.01	No	30000	108000	1389	109389
	Depth Based 2D CNNs [53]	1 face projection and 2D gridding	0.45	0.03	0.85	0.03	0.80	0.02	0.77	0.02	No	30000	108000	1321	109321
	Sparse Deep Neural Networks [54]	Flattening	0.62	0.02	0.78	0.01	0.78	0.03	0.65	0.03	No	30000	108000	1445	109445
Machine Learning	Boosted Trees	PCA	0.57	0.02	0.79	0.01	0.75	0.02	0.67	0.02	No	20000	72000	488	72488
	Random Forests	PCA	0.63	0.03	0.65	0.02	0.71	0.02	0.61	0.01	No	20000	72000	468	72468
	SVM/SVR	PCA	0.78	0.04	0.59	0.02	0.68	0.03	0.59	0.04	No	20000	72000	425	72425
Linear Approaches	Reg. Linear/Logistic Regression	PCA	0.81	0.01	0.58	0.01	0.65	0.02	0.62	0.01	No	20000	72000	58	72058

1% of the variance. Comparison for requirements (ii), (iii), (iv) and (vii) are made on regression performance attributes, namely, accuracy (MAE) and goodness-of-fit (R^2) for y^t ; and, classification performance metrics namely, accuracy and AUC-ROC. The OSER-MAS is able to perform better compared to all models. The OSER model also provides good results for these requirements as they overlap for single- and multi-station assembly systems. A two-sample t-test at 95% significance between the OSER-MAS and OSER model for metrics, namely R^2 ($p = 0.03$) and AUC-ROC ($p = 0.01$) demonstrates that the performance of the proposed OSER-MAS model is statistically significant

D. Benchmarking: Uncertainty Quantification

Benchmarking for quality of uncertainty estimates, i.e. requirement (v), is done using an error vs uncertainty calibration curve. For calibrated uncertainty quantification, high error should be associated with high uncertainty and vice-versa. RMSE is used as the error metric, and Inter-Quartile-Range (IQR) is used as the uncertainty metric given the non-normal nature of the estimated distributions. The RMSE for all observations between two consecutive thresholds is averaged. The correlation between the IQR and RMSE is also compared. Four priors for the weights and biases are considered: Standard Normal, Two Mixture GMM, Slab-Spike and Bernoulli. The Bernoulli prior method for uncertainty quantification is also known Monte Carlo (MC) dropout [44] and leverages dropout for approximate Bayesian inference.

As shown in Fig. 10, Standard Normal gives the most calibrated estimates for uncertainty with a correlation of 0.99. Both the OSER-MAS and OSER use standard normal priors, but given the Voxnet based 3D CNN architecture of the OSER approach, the correlation between IQR and RMSE is limited to

0.91, hence demonstrating the capability of the OSER-MAS model to provide more calibrated uncertainty estimates.

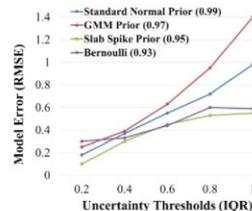


Fig. 10. Error (RMSE) vs uncertainty (IQR) comparison

E. Benchmarking: Upstream Object Shape Error Estimation

Benchmarking for requirement (viii), i.e., object shape error estimation for previous assembly stations, is done against other variants of 3D U-Nets and state-of-the-art methods used for semantic segmentation such as fully convolutional networks (FCN) [45], U-Net++, SegNet and DeepLab. Methods other than U-Net based approaches use a 2D projection and a VGG [46] based feature extractor. Given that shape errors are real-valued, regression-based metrics RMSE, MAE and R^2 are used to compare performance and have been summarized in Table V. The proposed model performs better in shape error estimation than others given the attention enabled link between corresponding levels of the encoder and decoder, which allows the model to be particular to parts/subassemblies within upstream shape errors. Figs. 11a, 11b and 11c compare the actual and estimated upstream shape error of the proposed approach for various types of RCs (part variation RC: $y^1 = 0.5$ mm, fixturing RC: $y^{11} = 0 - 0.5$ mm, joining RC: $y^{124} =$

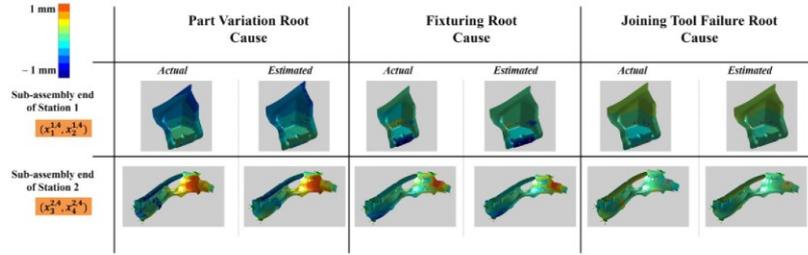


Fig. 11a. Comparison for x-component of object shape error

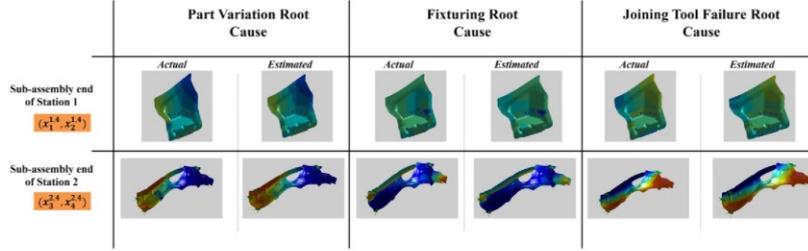


Fig. 11b. Comparison for y-component of object shape error

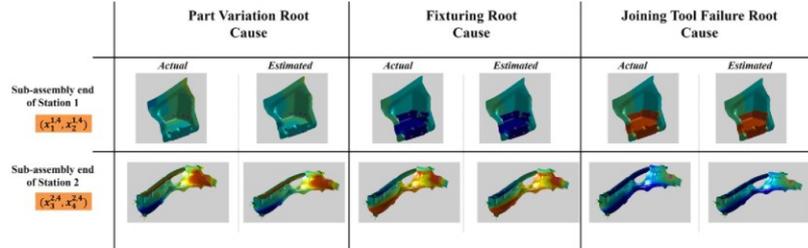


Fig. 11c. Comparison for z-component of object shape error

0) across all three components (x,y and z) of shape error. The proposed approach is able to accurately estimate all components of upstream shape errors inspite of all RCs being located in station 1. These shape errors enable the localisation of faulty assembly stations; the process parameters within these stations can be further analyzed to isolate the particular process parameter (RC). A comparison between the OSER approach for single station assembly systems and the OSER-MAS approach for MAS is shown in Table VI.

TABLE V: BENCHMARKING: UPSTREAM OBJECT SHAPE ERROR ESTIMATION

Models	Requirement (viii) $x^{1,4}, x^{2,4}$						Parameters
	RMSE (mm)		MAE (mm)		R^2		
	Mean	SD	Mean	SD	Mean	SD	
OSER-MAS (3D Attention U-Net)	0.014	0.005	0.002	0.001	0.96	0.1	2.4 M
U-Net ++ [47]	0.21	0.02	0.09	0.03	0.93	0.1	1.8 M

3D U-Net without attention [33]	0.25	0.01	0.15	0.02	0.91	0.2	1.5 M
SegNet [48]	0.31	0.01	0.21	0.02	0.88	0.1	1.5M
DeepLab [49]	0.38	0.02	0.25	0.03	0.86	0.1	20.5M
Fully Convolutional Networks (FCN) [45]	0.44	0.02	0.33	0.03	0.84	0.3	1.6M

V. CONCLUSIONS

The paper presented a 3D U-Net based methodology, Object Shape Error Response for Multi-station Assembly System (OSER-MAS), to isolate root cause(s) (RC) of dimensional and geometric product shape errors. It leverages a Bayesian U-Net 3D CNN architecture and closed-loop training to enable RCA with uncertainty quantification and learning during the design phase of the assembly system in the absence of measurement data. Additionally, the approach enables estimation of a heterogeneous set of process parameters, which include (i)

TABLE VI: COMPARISON OF THE OSER & OSER-MAS PROPOSED IN THIS PAPER

Requirements	Methodology		Performance	
	OSER	OSER-MAS	OSER	OSER-MAS
(i) <i>high data dimensionality</i>	3D Shape Error Voxelization	3D Shape Error Voxelization	(64*64*64*3) – voxels with shape error components	(64*64*64*3) – voxels with shape error components
(ii) <i>non-linearity</i>	Voxnet based CNN architecture	3D U-Net based CNN architecture	$MAE = 0.32 \text{ mm}$	$MAE = 0.34 \text{ mm}$
(iii) <i>collinearities</i>			$R^2 = 0.94$	$R^2 = 0.91$
(iv) <i>high faults multiplicity</i>			83 % fault multiplicity	100 % fault multiplicity
(v) <i>uncertainty quantification</i>	Bayes-by-backprop and Flipout	Bayes-by-backprop and Flipout	Calibration correlation = 0.91	Calibration correlation = 0.99
(vi) <i>dual data generation capabilities</i>	Uncertainty based closed-loop sampling from CAE Simulation	Uncertainty based closed-loop sampling from CAE Simulation	15000 samples for convergence	16000 samples for convergence
(vii) <i>High Dimensionality and heterogeneity of process parameters</i>	Single output head to estimate real-valued parameters	Multiple output heads to estimate real-valued and binary process parameters	N/A	$Accuracy = 0.92$ $AUC - ROC = 0.88$
(viii) <i>Fault Localization for RCA (Required for RCA of MAS)</i>	No	3D U-Net architecture with attention-based decoder	N/A	$RMSE = 0.014 \text{ mm}$ $MAE = 0.002 \text{ mm}$ $R^2 = 0.96$

incoming non-ideal and deformable parts; (ii) place-clamp-fasten-release (PCFR) parameters in assembly cell; and, (iii) part-to-part contact chains. The approach also enables fault localization by estimation of shape errors for upstream assembly stations. The approach is benchmarked against several categories of deep learning and statistical models. Results demonstrated better performance, quicker convergence and unique uncertainty quantification. Future work will entail leveraging continual learning approaches to comprehensively transfer the learning and perform RCA of shape defects in similar domains such as stamping, machining and additive manufacturing. Future work also involves building an integrated system where the OSER-MAS approach is not only used for RCA but can be leveraged to provide feedback to CAE simulators for inaccurate generation of simulation data. Leveraging such RCA models can provide an automated transformative framework to migrate from Industry 3.5 based Hybrid strategies to Industry 4.0 strategies to help achieve Zero-Defect-Manufacturing and Right-First-Time.

REFERENCES

- [1] P. Franciosa, et al., "Deep learning enhanced digital twin for Closed-Loop In-Process quality improvement," *CIRP Ann.*, vol. 69, no. 1, pp. 369–372, 2020.
- [2] D. De Silva, et al. "Toward Intelligent Industrial Informatics: A Review of Current Developments and Future Directions of Artificial Intelligence in Industrial Applications," *IEEE Industrial Electronics Magazine*, vol. 14, no. 2, pp. 57–72, 2020.
- [3] C. F. Chien, et al., "A Conceptual Framework for 'Industry 3.5' to Empower Intelligent Manufacturing and Case Studies," *Procedia Manuf.*, vol. 11, pp. 2009–2017, 2017.
- [4] W. J. Zhang and C. A. Van Luttervelt, "Toward a resilient manufacturing system," *CIRP Ann.*, vol. 60, no. 1, pp. 469–472, 2011.
- [5] V. J. Shahi, et al., "A quality-driven assembly sequence planning and line configuration selection for non-ideal compliant structures assemblies," *Int. J. Adv. Manuf. Technol.*, vol. 106, no. 1–2, pp. 15–30, 2020.
- [6] D. Ceglarek and J. Shi, "Dimensional variation reduction for automotive body assembly," *Manuf. Rev.*, vol. 8, pp. 139–154, 1995.
- [7] Y. Ding, D. Ceglarek, and J. Shi, "Fault Diagnosis of Multistage Manufacturing Processes by Using State Space Approach," *J. Manuf. Sci. Eng.*, vol. 124, no. 2, p. 313, 2002.
- [8] J. Liu, J. Jin, and J. Shi, "State space modeling for 3-D variation propagation in rigid-body multistage assembly processes," *IEEE Trans. Autom. Sci. Eng.*, vol. 7, no. 2, pp. 274–290, 2010.
- [9] S. Jin, Y. Liu, and Z. Lin, "A Bayesian network approach for fixture fault diagnosis in launch of the assembly process," *Int. J. Prod. Res.*, vol. 50, no. 23, pp. 6655–6666, 2012.
- [10] K. Bastani, B. Barazandeh, and Z. Kong, "Fault Diagnosis in Multistation Assembly Systems Using Spatially Correlated Bayesian Learning Algorithm," *J. Manuf. Sci. Eng.*, vol. 140, no. 3, p. 031003, 2017.
- [11] K. Bastani, et al., "Fault diagnosis using an enhanced relevance vector machine (RVM) for partially diagnosable multi-station assembly processes," *IEEE Trans. Autom. Sci. Eng.*, 2013.
- [12] S. Sinha, et al., "3D convolutional Neural networks to estimate assembly process parameters using 3D point-clouds," in *Proceedings of SPIE*, vol. 11059, 2019.
- [13] W. Huang, et al., "Stream-of-Variation (SOVA) Modeling—Part II: A Generic 3D Variation Model for Rigid Body Assembly in Multistation Assembly Processes," *ASME Trans. Journal of Manufacturing Science and Engineering*, vol. 129, no. 4, pp. 832–842, 2007.
- [14] S. Sinha, P. Franciosa and D. Ceglarek, "Object Shape Error Response Using Bayesian 3-D Convolutional Neural Networks for Assembly Systems With Compliant Parts," *IEEE Transactions on Industrial Informatics*, vol. 17, no. 10, pp. 6676–6686, 2021.
- [15] B. M. Haddad, et al., "Multifeature, Sparse-Based Approach for Defects Detection and Classification in Semiconductor Units," *IEEE Trans. Autom. Sci. Eng.*, vol. 15, no. 1, pp. 145–159, 2016.
- [16] E. Sales-Setien, et al., "Estimation of Nonstationary Process Variance in Multistage Manufacturing Processes Using a Model-Based Observer," *IEEE Trans. Autom. Sci. Eng.*, vol. 16, no. 2, pp. 741–754, 2019.
- [17] D. W. Apley and Y. Ding, "A characterization of diagnosability conditions for variance components analysis in assembly operations," *IEEE Trans. Autom. Sci. Eng.*, vol. 2, no. 2, pp. 101–110, 2005.
- [18] F. Adly et al., "Simplified subspace regression network for identification of defect patterns in semiconductor wafer maps," *IEEE Trans. Ind. Informatics*, vol. 11, no. 6, pp. 1267–1276, 2015.
- [19] R. Jin and K. Liu, "Multimode variation modeling and process monitoring for serial-parallel multistage manufacturing processes," *IIE Trans. Institute Ind. Eng.*, vol. 45, no. 6, pp. 617–629, 2013.
- [20] J. Zhong, J. Liu, and J. Shi, "Predictive control considering model uncertainty for variation reduction in multistage assembly processes," *IEEE Trans. Autom. Sci. Eng.*, vol. 7, no. 4, pp. 724–735, 2010.
- [21] X. Wang, L. T. Yang, L. Song, H. Wang, L. Ren, and J. Deen, "A Tensor-based Multi-Attributes Visual Feature Recognition Method for Industrial Intelligence," *IEEE Trans. Ind. Informatics*, 2020.
- [22] K. Mannar, et al., "Fault region localization: Product and process improvement based on field performance and manufacturing measurements," *IEEE Trans. Autom. Sci. Eng.*, vol. 3, no. 4, pp. 423–439, 2006.
- [23] O. Ronneberger, P. Fischer, and T. Brox, "U-net: Convolutional networks for biomedical image segmentation," in *Lecture Notes in Computer Science*, vol. 9351, pp. 234–241, 2015.
- [24] Y. Wen, et al., "Flipout: Efficient pseudo-independent weight perturbations on mini-batches," in 6th, ICLR, 2018.
- [25] O. Oktay et al., "Attention u-net: Learning where to look for the pancreas," *arXiv1804.03999*, 2018.
- [26] P. Franciosa, et al., "A novel hybrid shell element formulation (QUAD+ and TRIA+): A benchmarking and comparative study," *Finite Elem. Anal. Des.*, vol. 166, p. 103319, 2019.

- [27] Ö. Çiçek et al., "3D U-net: Learning dense volumetric segmentation from sparse annotation," in *Lecture Notes in Computer Science*, vol. 9901 LNCS, pp. 424–432, 2016.
- [28] S. Sinha, P. Franciosa, and D. Ceglarek, "Building a Scalable and Interpretable Bayesian Deep Learning Framework for Quality Control of Free Form Surfaces," *IEEE Access*, vol. 9, pp. 50188–50208, 2021.
- [29] D. Maturana and S. Scherer, "VoxNet: A 3D Convolutional Neural Network for real-time object recognition," in *IEEE International Conference on Intelligent Robots and Systems*, pp. 922–928, 2015.
- [30] C. Blundell, et al., "Weight Uncertainty in Neural Networks," 32nd Int. Conf. Mach. Learn. ICML 2015, vol. 2, pp. 1613–1622, 2015.
- [31] D. M. Blei, et al., "Variational Inference: A Review for Statisticians," *J. Am. Stat. Assoc.*, vol. 112, no. 518, pp. 859–877, 2016.
- [32] K. He, et al., "Deep residual learning for image recognition," in *Proceedings of the IEEE CVPR*, pp. 770–778, 2016.
- [33] F. Milletari, et al., "V-Net: Fully convolutional neural networks for volumetric medical image segmentation," 4th International Conference on 3D Vision, 3DV, pp. 565–571, 2016.
- [34] J. Bergstra, J. B. Ca, and Y. B. Ca, "Random Search for Hyper-Parameter Optimization Yoshua Bengio," 2012.
- [35] T. LaBonte, et al., "We Know Where We Don't Know: 3D Bayesian CNNs for Credible Geometric Uncertainty," arXiv1910.10793, 2019.
- [36] R. Cipolla, Y. Gal, and A. Kendall, "Multi-task Learning Using Uncertainty to Weigh Losses for Scene Geometry and Semantics," *Proc. IEEE CVPR.*, pp. 7482–7491, 2018.
- [37] D. P. Kingma et al., "Adam: A Method for Stochastic Optimization" arXiv1412.6980, 2014.
- [38] Y. Wu and K. He, "Group normalization," in *Proceedings ECCV*, pp. 3–19, 2018.
- [39] A. Kendall and Y. Gal, "What uncertainties do we need in bayesian deep learning for computer vision?," in *Advances in neural information processing systems*, pp. 5574–5584, 2017.
- [40] Y. M. Hsieh, et al., "Convolutional Neural Networks for Automatic Virtual Metrology," *IEEE Robot. Autom. Lett.*, 2021.
- [41] Google Research, "TensorFlow: Large-scale machine learning on heterogeneous systems," *Google Res.*, 2015.
- [42] S. Sinha, P. Franciosa, and D. Ceglarek, "Bayesian Deep Learning for Manufacturing," 2020. [Online]. Available: https://github.com/sumit-sinha/Deep_Learning_for_Manufacturing.
- [43] Z. Kong, D. Ceglarek, and W. Huang, "Multiple Fault Diagnosis Method in Multistation Assembly Processes Using Orthogonal Diagonalization Analysis," *J. Manuf. Sci. Eng.*, vol. 130, no. 1, p. 011014, 2008.
- [44] Y. Gal and Z. Ghahramani, "Bayesian convolutional neural networks with Bernoulli approximate variational inference," arXiv1506.02158, 2015.
- [45] J. Long, et al., "Fully convolutional networks for semantic segmentation," in *Proceedings of the IEEE CVPR*, 2015, pp. 3431–3440.
- [46] K. Simonyan and A. Zisserman, "Very deep convolutional networks for large-scale image recognition," arXiv Prepr. arXiv1409.1556, 2014.
- [47] Z. Zhou et al., "U-net++: A nested u-net architecture for medical image segmentation," in *Lecture Notes in CS*, vol. 11045 LNCS, pp. 3–11, 2018.
- [48] V. Badrinarayanan et al., "SegNet: A Deep Convolutional Encoder-Decoder Architecture for Image Segmentation," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 39, no. 12, pp. 2481–2495, 2017.
- [49] L. C. Chen, et al., "DeepLab: Semantic Image Segmentation with Deep Convolutional Nets, Atrous Convolution, and Fully Connected CRFs," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 40, no. 4, pp. 834–848, 2018.
- [50] Z. Liu, H. Tang, Y. Lin, and S. Han, "Point-voxel CNN for efficient 3d deep learning," arXiv1907.03739, 2019.
- [51] C. R. Qi, et al., "PointNet++: Deep Hierarchical Feature Learning on Point Sets in a Metric Space," arXiv 1706.02413, 2017.
- [52] H. Su, et al., "Multi-view Convolutional Neural Networks for 3D Shape Recognition," in *ICCV*, 2015.
- [53] J. Han, et al., "CNNs-Based RGB-D saliency detection via cross-view transfer and multiview fusion," *IEEE Trans. Cybern.*, vol. 48, no. 11, pp. 3171–3183, 2018.
- [54] F. W. Qin, et al., "Research on intelligent fault diagnosis of mechanical equipment based on sparse deep neural networks," *J. Vibroengineering*, vol. 19, no. 4, pp. 2439–2455, 2017.



SUMIT SINHA received his Bachelor of Technology (B.Tech.) degree in Industrial & Systems Engineering from the Indian Institute of Technology (IIT), Kharagpur, in 2017. He is currently pursuing his Ph.D. degree in deep learning for manufacturing at WMG, University of Warwick, Coventry, UK.

From 2017 to 2018, he worked as a data scientist at ZS Associates in the application of data science in pharmaceutical sales and marketing. He also worked briefly on data science projects at the Hong Kong University (HKU) and Chongqing University, China. His research interests include data science, Bayesian deep learning and digital twins of manufacturing assembly systems.



PASQUALE FRANCIOSA received the Ph.D. degree in mechanical engineering from the University of Naples Federico II, Italy, in 2010. He is an Assistant Professor with the University of Warwick, Coventry, U.K., and the Head of the Laser Welding Applications Laboratory, Warwick Manufacturing Group, University of Warwick. He has been Principal Investigator (PI) and Co-Investigator (Co-I) on several funded projects with a total income to the University of Warwick of circa £2.8M

(\$3.9M) since 2015. He has authored or coauthored more than 80 papers. His research interests include smart manufacturing, process monitoring, closed-loop control, applications of machine learning/artificial intelligence, and multidisciplinary optimization, with specific attention on assembly systems and laser processes.

Dr. Franciosa was the recipient of four Best Paper Awards. He is a Member of the Editorial Board of the *ASTM Smart and Sustainable Manufacturing Systems journal*



DARIUSZ CEGLAREK received the Ph.D. degree in mechanical engineering (ME) from the University of Michigan-Ann Arbor, Ann Arbor, MI, USA, in 1994. He is an EPSRC Star Recruit Research Chair with the University of Warwick, Coventry, U.K. He was a Professor with IS&E, University of Wisconsin-Madison, Madison, WI, USA. He has been Principal Investigator (PI) Co-Investigator (Co-I) on research grants of over £30M (\$41M): NSF/NIST/EPSC/ InnovateUK/APC/EU-FP7/Curie and industry. He has authored or coauthored more than

180 papers and is listed by Stanford University among top 2% of the world's leading scientists. His research interests include smart manufacturing and data mining/AI for root cause analysis across design, manufacturing, and service.

Prof. Ceglarek is a Fellow of College International pour la Recherche en Productique (CIRP). He was the recipient of several Best Paper Awards, the 2018 JLR "Innovista" Award for the most innovative "piloted technology," the EPSRC Star Award, and the NSF CAREER Award. He was the Associate Editor for *ASTM Smart and Sustainable Manufacturing Systems*, the *IEEE Transactions on Automation Science and Engineering*, and the *ASME Journal of Manufacturing Science and Engineering*.

Appendix F: Under Review

Journal Paper 4

S. Sinha, P. Franciosa, and D. Ceglarek, "Building a Quality Correction Framework for Assembly Systems using Deep Reinforcement Learning " in review, Journal of Manufacturing Systems, 2021.

Building a Quality Correction Framework for Multi-Station Assembly Systems using Deep Reinforcement Learning

Sumit Sinha¹
Digital Lifecycle Management
WMG, University of Warwick
Coventry, CV4 7AL, U.K.
Sumit.Sinha.1@warwick.ac.uk

Pasquale Franciosa
Digital Lifecycle Management
WMG, University of Warwick
Coventry, CV4 7AL, U.K.
P.Franciosa@warwick.ac.uk

Dariusz Ceglarek
Digital Lifecycle Management
WMG, University of Warwick
Coventry, CV4 7AL, U.K.
D.J.Ceglarek@warwick.ac.uk

Abstract—The paper proposes a novel deep reinforcement learning-based approach, Object Shape Error Correction (OSEC), to eliminate root cause(s) (RCs) of dimensional and geometric product shape errors by determining optimal corrective action(s) (CA(s)). It leverages the Deep Deterministic Policy Gradient (DDPG) algorithm to learn optimal process parameter correction policies based on RC state estimates of multi-station assembly systems (MAS). These policies can be interpreted in engineering terms as sequential corrective adjustments of process parameters necessary to mitigate RCs of product shape errors. The approach has the capability to estimate adjustments of process parameters related to fixturing and joining while simultaneously accounting for (i) RC estimation uncertainty, (ii) Key Performance Indicator (KPI) improvement, (iii) MAS design architecture; and, (iv) MAS inherent stochasticity. The OSEC methodology leverages a novel reward function parametrised by user-defined new product introduction (NPI)-coefficients that balance the trade-off between requirements, benefits and costs and enable application of the proposed approach throughout the NPI and production lifecycle. Benchmarking the methodology using an industrial, automotive cross-member assembly system demonstrated a 40% increase in corrective actions' effectiveness compared to current approaches.

Keywords—Deep Reinforcement Learning, Assembly Correction, Manufacturing, Deep Deterministic Policy Gradient

1. INTRODUCTION

THE manufacturing industry requires a *transformative framework* [1][2] that can map high-dimensional manufacturing data (e.g. point clouds from 3D scanners [3]) to actionable insights such as root cause(s) (RC) and corrective action(s) (CAs). Currently, CAs are performed using two separate steps (i) isolation of the RC using process and quality data and supported by statistical or deep learning approaches; and, (ii) determination and implementation of CA(s) to mitigate RCs which is based on experience and expertise to account for system constraints and stochasticity. The two steps are not integrated in an end-to-end manner. As a result, the implementation of CAs can lead to high mean-time-to-resolution (MTTR) of quality defects that increases machine downtime, scrap and/or need for rework or repair, thereby increasing the cost of quality and reducing productivity. Even the most experienced staff is not able to take into consideration stochasticity of manufacturing systems which directly affect mapping between RCs and CAs. At present there are no systematic data-driven analytical approaches which can link both steps into a single *end-to-end* approach, i.e., an approach which spans from data to root cause analysis (RCA) and to CAs via a single forward pass *without the presence of human expert in the loop*. This is primarily because currently, there are no CA approaches that mathematically formulate the steps necessary to determine and implement CAs, thereby, resulting in CAs that are dependent on extensive manual expertise. Frequently, manually determined CAs cannot be implemented given the constraints of the system, such as the inability to implement CAs due to ongoing production schedules; this further motivates the need for end-to-end approaches that can transform manufacturing data to RCs and effective and feasible CAs that can be implemented within reasonable time and costs.

The key advantage of an end-to-end approach is that it integrates RC isolation and CA determination within a single framework by simultaneously accounting for RCs, system constraints and stochasticity, and maps manufacturing data to RCs and to CAs that are feasible to implement and effective when implemented. Learning such end-to-end transformation functions can enable the realisation of various Industry 4.0 milestones such as Right-First-Time and Zero-Defect-Manufacturing, which collectively drive towards scenarios where all quality requirements are met from Job 1.

Currently, numerous manufacturing applications require multiple assembly stations, many of which are fully automatic robotic stations, to be placed in the production line [4]. These are known as multi-station assembly systems (MAS). Each robotic assembly station within MAS comprises of a robot with end effector holding joining head (i.e., laser welding, resistance spot welding (RSW), self-piercing riveting (SPR), friction stir welding (FSW), among others), and fixtures locating parts to be assembled. All elements

¹ Corresponding Author

of a robotic station are affected by inherent variation caused by initial calibration, deterioration, and routine maintenance, which might lead to diminished product quality. Additionally, many assembled products are made of deformable non-ideal parts [4], for which dimensional and geometric error/variation (*object shape errors*) is one of the leading quality problems. Hence, it is essential to develop approaches for CAs to ensure that MAS can continue to produce high-quality products in the presence of non-ideal parts.

To date, three groups of approaches have been proposed to perform CAs of MAS. The approaches include: (1) mapping manufacturing data to RC using mathematical models ($f^{RC}(\cdot)$) and then mapping RC to CAs based on manual experience. Process parameters y are identified as faults and isolated as RCs and CAs and generated as the quantitative negative of the RC ($CA = -RC = \Delta y_{RC=f^{RC}(y)}$); (2) mapping manufacturing data directly to CAs using engineering feedback control models such as statistical process control (SPC)/ automated process control (APC) that aim to eliminate process abnormalities (*special causes*) rather than to explicitly eliminate quality defects ($CA = \Delta y_{feedback}$); and, (3) mapping manufacturing data directly to CAs by manually monitoring key product characteristics (KPC) and implementing CAs that change process parameters affecting the faulty product characteristics ($CA = \Delta y_{KPC}$).

The first group of approaches ($CA = -RC = \Delta y_{RC=f^{RC}(y)}$) estimate RCs using statistical [5] and deep learning models [6] [7], based on quality data collected from the MAS. The outputs of these models are then used by human experts to determine the CAs. The effectiveness of CAs for such approaches depends considerably on the accuracy and fidelity of the RC isolation model. Accuracy is measured as the ability of the model to discriminate between different combinations of RCs and fidelity is measured as the ability of the model to precisely estimate the magnitude of the isolated RC. Previously, RC models leveraged a state space methodology to develop a stream-of-variation model [8] of the MAS. However, the RCA models have had limited-applicability to complex, high dimensional and nonlinear systems [9] due to the use of linear models between process parameters and measurements of dimensional product quality [10]. Recently, Object Shape Error Response (OSER) was proposed for single- [7] and multi-station assembly systems (OSER-MAS) [11] by integrating Bayesian deep learning with Computer-Aided Engineering (CAE) simulations as a closed-loop framework [12]. It isolated one or multiple RCs simultaneously as *probability distributions*. Given the inherent uncertainties, collinearities and fault multiplicity of MAS [7], a single deterministic estimate of an RC is inadequate hence, the OSER approach trains a Bayesian 3D convolutional neural network model to estimate RCs as a multivariate normal distribution. Currently, the CAs corresponding to the RCs are generated by human experts by integrating the estimated RCs with the MAS's constraints and stochasticity [13][14]. Mathematically, these CAs can be quantified as the negative of the mean value of RC distribution ($CA = -RC$). In certain cases, such CAs might not be *feasible* for implementation due to maintenance related time constraints. Moreover, even in cases where implementation is feasible, nonetheless the cost consideration can be exorbitant particularly for MAS with limited reconfigurability. On the whole the implementation of the manually generated CAs might be *ineffective* due to the inherent level of stochasticity in the MAS caused by non-ideal part variations generated from upstream processes such as stamping. This inherent level of stochasticity in the MAS is further magnified in case of certain parts fabrication processes such as stamping, as the magnitude and pattern of incoming part variations can change between subsequent batches of parts (batch-to-batch variation in stamping). Hence, a manually determined CA, derived based on the experience of a previous batch, can fail if applied to subsequent batches with different variation patterns thereby, resulting in increased scrap and diminished productivity.

The second group of approaches ($CA = \Delta y_{feedback}$) determine CAs of quality defects involve integrating SPC/APC approaches collectively known as engineering feedback control [15–19]. The SPC/APC approaches enable automatic CA generation, however, they assume that throughout the new product introduction (NPI) and then production lifecycle, the *design nominals* of process parameters would continue to provide the best product quality and process performance in terms of costs and productivity. Hence, they generate CAs by attempting to revert any process parameter with special causes to the design nominal. The CAs are then given as feedback to process controllers, or as a maintenance task involving faulty equipment [20][21]. Thus, the SPC/APC methods aim to control special causes by identifying process parameters that have temporal patterns such as mean-shift or heteroskedasticity and/or are beyond the pre-specified control limits. As a result, these methods aim to correct the MAS by reverting process parameters with special causes rather than accounting for dynamically changing stochasticity, constraints and KPIs of the MAS. Various studies have shown that as MAS progresses through the NPI and production lifecycle, the design nominals and specified tolerances no longer provide optimal key performance indicators (KPIs) values [13] [14]. Moreover, Mannar et al. [13] observed that a number of product warranty faults occur when all the key process parameters are within the design tolerance windows. Such faults are often classified as no-fault-found/no-defect-found phenomenon, i.e., a product/subassembly is reported as faulty, however, the external tests do not detect any fault.

The no-fault-found phenomenon might be caused by issues in the design stage such as inaccurate design nominals and/or imprecise tolerance windows of key parameters (assembly process design). They may also be caused to issues within the production stage such as imprecise deterioration of the production equipment (maintenance plan) or/and inaccurate and/or imprecise incoming parts (fabrication process). Mannar et al. [13] developed Fault Region Localization (FRL) methodology to address no-fault-found issues by identifying and localising quality faults even if they occur inside the tolerance window caused by imprecise design tolerance boundaries. The FRL methodology combines warranty and production data to identify and localise in-tolerance faulty

regions, i.e., low-yield regions within the tolerance. Additionally, Mannar et al. [14] proposed re-evaluating the manufacturing acceptance criteria or re-evaluating product tolerances to prevent delivering to customers defective products that fall into the in-tolerance faulty regions. Overall, the inaccurate design nominals and imprecise tolerance windows of key parameters can be the result of unmodelled interactions during product and process design given the limitations of simulation-based design methods [22].

Hence, the second group ($CA = -\Delta y_{feedback}$) and third group ($CA = -\Delta y_{y \rightarrow KPC}$) of approaches directly map data to CAs ($CA = -\Delta y$) assuming a fixed *design nominal* without systematic RCA. For such approaches, CAs are estimated as the quantitative negative of the difference between process parameter deviation estimates and fixed design nominals. In scenarios when the design nominal is not giving optimal KPI performance, the CAs determined by both category approaches may be ineffective in practice and a *functional nominal* based on the current stochasticity and interactions needs to be estimated to determine effective CAs. Functional nominals can be interpreted as the values of process parameters that ensure the lowest process fallout rate [13] or/and maximize KPIs under the current MAS constraints and stochasticity.

As the aforesaid discussion underscores that CAs determined by the three groups of approaches may be inadequate in practice. Therefore, determining CAs within MAS should account for various requirements which can be grouped as either RC or MAS related.

RC related requirements include:

(i) *RC uncertainty estimation* as justified in current RCA approaches (OSER [7], OSER-MAS [11]). These RCA approaches account for various key requirements such as high data dimensionality, non-linearity, collinearities and high faults multiplicity while estimating RCs [7]. Under such requirements, RCs are estimated as multivariate normal distributions which need to be taken into consideration when determining CA. The CA cannot be determined by leveraging only the mean of the RC distribution as it might lead to biased and be *ineffective* in practice for assembly processes with non-ideal parts and in scenarios involving multiple collinear defects. For example, part-to-part gap variation affects the quality of welded joints during remote laser welding assembly and can be caused either due to fixture-induced errors or/and incoming part sudden variation change. The part-to-part gap variation can be corrected for small gap variation by laser welding power adjustment as CA. However, it requires addressing sudden variation change of incoming parts (as CA) if part-to-part variation is relatively large (>50% of part thickness).

(ii) *KPI improvement* [23] as CAs should eliminate RCs and is subject to cost-benefit assessment, i.e., the CA are evaluated by their impact on the final KPIs improvement (quality improvement) in relation to the cost of their implementation and loss of quality in case the CA is not implemented. The cost of CA implementation varies between MASs as some may have built-in automation and reconfigurability, allowing for quick CA implementation without any major impact on the production schedule, while some other MAS may require manual implementation of CA that might result in increased downtime and decreased productivity. On the other hand, the impact of eliminating RC on the KPIs provide measurable benefits of increased quality. Therefore, the final choice between multiple CAs that can be implemented is subject to cost-benefit analysis with consideration to the quality KPIs

MAS related requirements include:

(iii) *MAS design architecture* [13] which can result in certain CAs being infeasible to implement in a given MAS because of process design and MAS reconfigurability limitations that constrain certain degrees of freedom for process parameters adjustments. Additionally, high costs or restricted time and frequency of CA implementation also limit the set of feasible CAs. For instance, certain CAs requires a long duration for implementation, but only a limited time window may be available for regular maintenance and CA implementation. These limitations need to be considered in the CA methodology as they might create scenarios wherein the implementation of a generated CA to eliminate a given RC may be *unjustified* in terms of cost-benefits analysis as in (ii).

(iv) *MAS inherent stochasticity* [4] as caused by incoming part variations (i.e., non-ideal parts due to inherent variation of upstream processes such as stamping, extrusions, etc.) and tooling variation (i.e., tooling calibration error, repeatability and reproducibility) might lead to certain CAs being *ineffective* in practice.

Hence, it is essential to develop an approach that can determine *effective* CAs while accounting for the trade-offs between the aforementioned RC and MAS related requirements, which can lead to *multiple* mappings between RCs and CAs, instead of 1-to-1 mapping, where $CA = -RC$. Therefore, the approach should have the capability to learn CA policies that account for RC uncertainty, KPI improvement, MAS design architecture constraints and inherent stochasticity. A CA policy can be interpreted as a sequence (referred to as episode in the rest of the paper) of corrective adjustments of process parameters (CAs) that eventually lead to RC elimination and KPI maximization. These CA policies enable the correction of object shape error quality defects by providing optimal CAs that can be further implemented in process controllers or executed as maintenance actions [24].

This article presents the CA methodology, OSEC, which addresses the requirements (i)-(iv). The OSEC approach takes as input RC estimates from currently used RCA models (such as in [7] [11]) and estimates CAs that are feasible to implement and effective in eliminating the RCs in the MAS. This is achieved by:

1. Formulating MAS correction as a Markov Decision Process (MDP) and proposing a *deep reinforcement learning (DRL) approach that can estimate CAs (actions) based on the RCs (state) provided as inputs while accounting for the four requirements (reward)* (i) RC uncertainty estimation, (ii) KPI improvement, (iii) MAS design architecture and (iv) MAS inherent stochasticity. This approach is developed as the end-to-end framework, i.e., combines both estimation of: (i) RCs; and (ii) CAs. For this, first, high dimensional point cloud data which are collected at the end of the MAS (end-of-line sensing) are used as input to current RCA models (such as OSER [7] [11]). Next, the isolated RCs are used as input to CA methodology

- to estimate optimal CAs as the final outputs.
2. Leveraging deep reinforcement learning as in (1) above using *Deep Deterministic Policy Gradients (DDPG) technique*. Given that the output space for CAs consists of multi-dimensional real-valued process parameters adjustments, traditionally used action-value based methods cannot be used as they are limited to work on discrete action spaces [25]. Therefore, the proposed approach leverages DDPG as this can map states (RCs) to multi-dimensional real-valued actions (CAs) by approximating both the CA policy (i.e., sequence of CAs) and CA value function (i.e., overall reward of taking a CA) using deep neural networks. In the proposed CA approach, OSER is leveraged as the RCA model to estimate the state of the MAS, which is given as input to the DDPG agent. DDPG agent is constituted as the combination of the CA policy and CA value function which takes as input the RCs and estimates the CA with maximum value.
 3. Training DDPG agent as in (2) above with a *novel reward function* that mathematically models all (i)-(iv) requirements that need to be accounted for the design and implementation of CAs in MAS. The reward generates positive values (*benefits*) when RCs are eliminated and KPIs are improved. Simultaneously, it penalizes the DDPG agent (*costs*) for generating CAs that are infeasible due to cost and time needed for implementation. No reward is allocated when CAs do not improve KPIs due to the stochasticity present in the MAS or in scenarios when the RC uncertainty estimates are high. The proposed OSEC approach does not limit CAs to adjust process parameters to the fixed design nominals; but rather, it selects the CAs to adjust process parameters to the *functional nominals in order to maximize the KPIs*. This is in contrast to current SPC/SPC approaches which assume a fixed design nominal to generate CAs that adjust values of a subset of process parameters identified as abnormal due to temporal patterns or variations beyond the specified control limits.
 4. Parametrizing the reward function in (3) with *NPI-coefficients* to enable the application of the proposed method throughout all NPI stages namely design, pre-production, production launch and full production. This is especially important as the importance and priorities of the aforementioned four requirements changes during each NPI stages. At design and pre-production stages of the NPI, there are fewer constraints on cost, number of changes and no constraints related to production/maintenance schedules, however, the improvement of KPIs is crucial, and thus, the flexibility of having functional nominals instead of fixed design nominals provide necessary flexibility and thus, frequent CAs can be implemented to improve the KPIs. On the other hand, during production stage, the MAS becomes more '*rigid*' due to increased costs of change and limited time permitted for changes (maintenance scheduled breaks). Therefore, to account for the changing trade-offs and priorities, the method proposes to parametrize the reward function with *NPI-coefficients*. The NPI-coefficients include estimates for KPI importance, costs, system rigidity and sample size. The user can select the NPI-coefficients for a specific NPI stage which are then used to parametrize the reward function. Overall the proposed reward function simultaneously maximises KPIs by eliminating RCs while accounting for the MAS stochasticity and ensuring that the CAs are within the allowable MAS reconfigurability for executing corrections [26].

Overall, the article develops an End-to-End Object Shape Error Correction (OSEC) approach with the goal of building a framework that can generate feasible and effective CAs. Within the approach high-dimensional manufacturing data collected at the end of the MAS is first transformed to RCs, and then RCs are transformed into CAs. The proposed approach enables correction of object shape error quality defects for MAS with non-ideal parts. It leverages DRL, specifically DDPG agent, to optimize a novel reward function resulting in CA policies that generate sequential process parameter adjustments (CAs). The approach can be leveraged to generate CAs when the CAs estimated by current techniques cannot be directly transformed to CAs ($CA \neq -RC$, $CA \neq -\Delta y$). This is because the RCs map to CAs, which might be infeasible to implement due to KPI improvement and MAS constraints or rendered ineffective after implementation due to RC uncertainty or MAS stochasticity. The approach also provides capabilities (Fig. 1) to estimate alternative CAs for those classes of CAs that are infeasible or potentially ineffective due to NPI lifecycle requirements or assumptions such as a fixed design nominal. Where such class of CAs does not exist, the CAs estimated by the planned approach correspond to the CAs estimated by current approaches ($CA = -RC$, $CA = -\Delta y$) which assume that all CAs are implementable, and the design nominal always maximizes KPI performance.

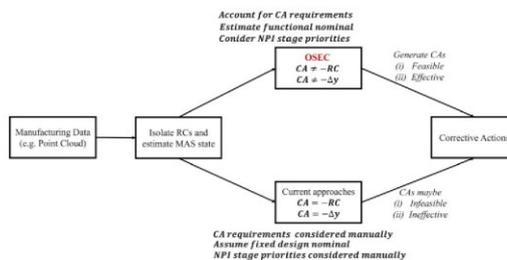


Fig. 1. OSEC vs current approaches

The key contributions of the paper are as follows:

- (1) Propose a novel deep reinforcement learning based *OSEC approach* that leverages a DDPG agent to eliminate object shape errors by learning CA policies that can map isolated RCs and estimated state of MAS to sequential process parameter adjustments (CA). The CA policy generates a functional nominal that simultaneously optimises for requirements such as RC uncertainty, KPI improvement, MAS design constraints and MAS inherent stochasticity.
- (2) Propose a quantitative formulation for MAS correction that mathematically accounts for the (a) benefits of CAs provided due to RC elimination and KPI improvement, (b) the costs incurred due to MAS reconfigurability and adjustments constraints, (c) MAS stochasticity due to incoming part and tooling variations and (d) the NPI stages of the MAS. The formulation proposes a novel correction reward function parametrised by user-defined NPI-coefficients, enabling the fulfilment of all RC and MAS requirements and providing capabilities for mitigating object shape error across all stages of the NPI and production lifecycle.
- (3) Verify and validate the methodology by implementing it on an industrial, automotive cross member assembly consisting of three stations, four non-ideal parts and twelve process parameters using a CAE simulator known as Variation Response Method (VRM) [27].
- (4) Conduct comprehensive benchmarking against the three groups of frameworks used for correction of MAS, that includes (a) first group of approaches that map data to RCs and to CAs based on manual expertise ($CA = -RC$), and (b) second and third group of approaches that directly map data to CAs ($CA = \Delta y$) such as SPC/APC assuming a fixed design nominal and do not consider the NPI stage. The benchmarking highlights superior performance in terms of higher quality KPIs (RC elimination and reduction of object shape error), lower costs (penalty due to MAS reconfigurability and adjustment constraints) and quicker convergence to optimal CAs under varying levels of stochasticity (part and tooling variations).

The rest of the paper is organised as follows: Section 2 formulates the correction problem for MAS using MDP; Section 3 discusses the methodology; Section 4 presents the industrial case study; and, finally, conclusions and future work are summarised in Section 5.

2. PROBLEM FORMULATION

The application of OSEC requires the formulation of the correction problem as an MDP so that DDPG agent can be trained to perform CA. The MAS (Fig. 2) can be represented as a process tree where different nodes correspond to stages within a single assembly station (Fig. 2a) or as stations within the assembly system (Fig. 2b). A *station* consists of *four stages: positioning, clamping, fastening, and release* (PCFR). Object shape errors can be induced in any station by variations in one or multiple process parameter(s). These errors are propagated and accumulate in a non-additive manner [9][8]. At any given state, the variation in the process parameter(s) from the nominal is a source of shape error and is known as the RC(s) of the shape errors. CAs involve adjustments of controllable/reconfigurable process parameters to mitigate RCs of object shape errors.

The object shape error response and RCA problem has been previously formulated by Sinha et al. [11]. Within the formulation, stations in the MAS are represented by $s: s = 1, \dots, N_s$, where N_s represents the total number of stations within the system and $\acute{s}: \acute{s} = 1, 2, 3, 4$ represents the four stages within each station. $s = 4$ represents the final stage of the assembly station. The MAS consists of n incoming parts also known as objects that need to be assembled. Any object $o: o = 1, \dots, n$ at its design nominal shape is characterised by a set of nominal points $\mathbf{P}_o = \{\mathbf{p}_{ok}\}$, $k = 1, \dots, n_o$, where \mathbf{p}_{ok} is a

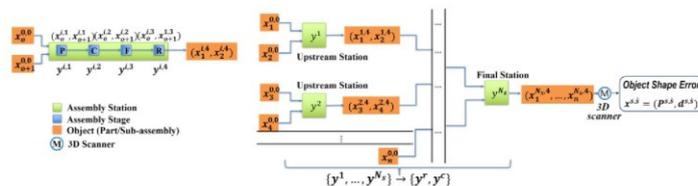


Fig. 2a. PCFR stages of an assembly station

Fig. 2b. Multi-station assembly system with N_s stations

vector consisting of the coordinates of the k th input point and n_o represents the total number of points on object o . $\mathbf{d}_o = \{\mathbf{d}_{ok}\}$ denotes the deviation of each point k after the nominal object o has gone through different stages of the process, \mathbf{d}_{ok} is a vector comprised of deviations of each point in x , y and z axes on object o . As the object o goes through multiple stations and stages within the stations, the set of points are represented as $\mathbf{P}_o^{s,\acute{s}}$ while $\mathbf{d}_o^{s,\acute{s}}$ represents the corresponding shape error. Object shape error for object o after station s and stage \acute{s} can be represented as:

$$\mathbf{x}_o^{s,\acute{s}} = (\mathbf{p}_o^{s,\acute{s}}, \mathbf{d}_o^{s,\acute{s}}) \quad (1)$$

where $\mathbf{x}_o^{0,0} = (\mathbf{p}_o^{0,0}, \mathbf{d}_o^{0,0})$ represents the incoming non-ideal part inclusive of the part variations. A subassembly of incoming parts

after the stage \hat{s} of station s is represented as the set of objects $(x_1^{\hat{s}}, \dots, x_n^{\hat{s}})$. The set of process parameters within the MAS are real-valued and represented by a vector \mathbf{y} for the total set \mathbf{h} of process parameters across the entire MAS. An assumption made in this article is that point cloud data is only collected at the end of MAS $\mathbf{x}^{N_s,4}$ not at the end of intermediate stations $\{\mathbf{x}^{1,4}, \dots, \mathbf{x}^{N_s-1,4}\}$. This is known as end-of-line sensing [3]. This is generally the case with 3D scanners as they are expensive and have a high measurement time. Installing scanners at end-of-line ensures products can be measured without any interruption in the production line.

OSER [11] RCA model learns a function using Bayesian 3D U-Net based Convolutional Neural Networks (CNN) and Computer-Aided Engineering (CAE) simulations that takes as input the combined object shape error at the end of the MAS $\mathbf{x}^{N_s,4}$, i.e., after the last station of the MAS ($s = N_s$) and final stage ($\hat{s} = 4$) of the station, and estimates the process parameters across the entire system \mathbf{y} with uncertainties $\sigma(\mathbf{y})$ and the object shape error for all objects at the last stage ($\hat{s} = 4$) of all upstream stations ($s = 1, 2, \dots, N_s - 1$) collectively represented as $\{\mathbf{x}^{1,4}, \dots, \mathbf{x}^{N_s-1,4}\}$. Overall, the inputs and outputs of the RCA function $f(\cdot)$ can be represented as:

$$\{\mathbf{y}, \sigma(\mathbf{y}), \mathbf{x}^{1,4}, \dots, \mathbf{x}^{N_s-1,4}\} = OSER(\mathbf{x}^{N_s,4}) \quad (2)$$

The RCs are estimated as a subset of process parameters $\mathbf{RC} \subseteq \mathbf{y}$ and hence, are directly linked to the point cloud data by the OSER model. The model has capabilities to estimate one or multiple RCs. The article assumes that the MAS is diagnosable (i.e. one or multiple RCs can be estimated based on input point cloud data collected and the OSER model, i.e., the measurement data and model capability is sufficient such that all possible RCs (single RC or multiple RCs) can be discriminated. However, this may not always be the case as the data is collected only at the end of MAS ($\mathbf{x}^{N_s,4}$) (end-of-line sensing) and distributed sensing approaches that collect data at multiple stages within the MAS may need to be used to ensure all RC combinations can be discriminated. For notation consistency, the dimension of the \mathbf{RC} vector is the same as the process parameter vector \mathbf{y} , i.e., the total of h process parameters and all process parameters that are not RCs are assumed at their nominal (considered zero for simplicity). This also translates into the process parameter uncertainty being equal to the RC estimation uncertainty $\sigma(\mathbf{y}) = \sigma(\mathbf{RC})$.

The article extends the object shape error response formulation to leverage the outputs of the OSER RCA model as shown in (2) to define state, action and reward and formally formulate the correction problem as an MDP. Firstly, the state representation of the MAS \mathbf{s}_t^{MAS} for time step (t) is constructed for which the upstream shape errors $\{\mathbf{x}^{1,4}, \dots, \mathbf{x}^{N_s-1,4}\}$ are consolidated into a fixed-length vector $\mathbf{x}_t^{upstream}$ using a dimensionality reduction function $g(\cdot)$. The 3D CNN model [7] as proposed in the OSER methodology is leveraged as $g(\cdot)$ as this enables the extraction of local and global features from the upstream shape errors. The output after the last hidden layer with 64 nodes is leveraged as the consolidated shape error vector with dimension 64 for each set of shape errors. This is concatenated across all $N_s - 1$ shape errors after all upstream stations and has a length of $d = 64 \times (N_s - 1)$

$$\mathbf{x}_t^{upstream} = g(\mathbf{x}^{1,4}, \dots, \mathbf{x}^{N_s-1,4}) \quad (3)$$

In practice, a time step (t) corresponds to a single product manufactured by the MAS, i.e., after all objects that need to be assembled have gone through the respective stations and stages. At the end of this time step, the state (\mathbf{s}_t^{MAS}) of the MAS is described by a consolidated set of estimated \mathbf{RC}_t , their estimation uncertainties $\sigma(\mathbf{RC}_t)$ and the consolidated set of all upstream shape errors $\mathbf{x}_t^{upstream}$:

$$\mathbf{s}_t^{MAS} = \{\mathbf{RC}_t, \sigma(\mathbf{RC}_t), \mathbf{x}_t^{upstream}\} \quad (4)$$

Where \mathbf{RC}_t is vector of dimension h corresponding to RCs within the MAS, $\sigma(\mathbf{RC}_t)$ is a vector of dimension h corresponding to the uncertainty in the respective RCs and $\mathbf{x}_t^{upstream}$ is a consolidated vector of all upstream shape errors as shown in (3). The \mathbf{RC}_t vector can also be augmented with other data collected from other sensors distributed across the MAS. This set is the *markovian* representation of the MAS's current state that contains all information required (i.e., access to earlier states of the MAS is not required) for the agent to take a CA.

Apart from the current state \mathbf{s}_t^{MAS} , the DDPG agent needs a *reward signal* to maximise while learning a CA policy (behaviour function that generates sequential process parameter adjustments) to estimate CA-value and determine optimal CAs (process parameter adjustments). A sequence of these CAs is known as an *episode*. The episode eventually terminates into states with negligible shape error. The reward should primarily account for the aforementioned four RC and MAS requirements and should be useable across all stages of the NPI lifecycle. The article formulates such as reward function by leveraging a set of NPI-coefficients δ .

The first part of the reward aims to quantify the *benefits* gained due to a CA taken at timestep t . This part also ensures that RC requirements (r_δ^{RC}) are met and can be formulated as:

$$r_\delta^{RC} = -(1 - \sigma^{RC_t}) \frac{1}{k^n} \sum_{i=1}^{k^n} K^i (KPI_0^i - KPI_t^i)^2 \quad (5)$$

r_δ^{RC} quantifies the loss of k^n KPIs deviating from their nominal KPI_0 across the system leveraging a Taguchi loss function [23].

Each KPI has a relevance coefficient K^i that accounts for the KPI importance and can be used as a measure for the sensitivity of the KPI to the overall cost or quality (cost of quality). The reward is discounted by a normalised uncertainty [0,1] factor σ^{RC_t} which is a summation of uncertainty in all process parameters $\sigma(RC_t)$. σ^{RC_t} is 0 when uncertainty is below a minimum threshold and 1 when uncertainty is above the maximum threshold. This effectively translates into the reward being near zero for CAs taken when the estimated RCs are highly uncertain. Collectively this mathematically formulates req. (i) *RC estimation uncertainty and req. (ii) KPI improvement*.

The next part of the reward aims to quantify the *costs* incurred due to a CA being taken at timestep t . This part also ensures that MAS requirements (R_δ^{CA}) are met and can be formulated as:

$$r_t^{CA} = -(1 - \sigma^{RC_t}) \lambda \frac{1}{h} \sum_{j=1}^h C^j (y_{t+s^p}^j - RC_t^j)^2 \quad (6)$$

r_t^{CA} quantifies the cost for violating req. (iii) *MAS design architecture* by including a process parameter change penalty that penalises the agent from frequently changing h process parameters. The negative of this term ($-r_t^{CA}$) can also be interpreted as a *MAS requirement penalty* as it penalises the agent from taking frequent infeasible CAs that violate MAS reconfigurability constraints. Permitted reconfigurability of a MAS measured by the degree of freedom for adjustments of various tools, the allowable ranges of adjustments and cost of one or multiple adjustments. For consistency, each degree of freedom (DOF) for a given tool (e.g. pin-hole, pin-slot, or NC-block) is considered a unique process parameter. One or multiple of these process parameters can be adjusted if selected as a CA. Each process parameter y^j can only vary between allowable limits $[y_{min}^j, y_{max}^j]$. Any adjustment to a process parameter is associated with a cost coefficient (C^j) which accounts for the costs of changing specific process parameters to correct RCs. Correcting RCs requires a sequence of CAs (episode). Each time a CA is performed within an episode, a cost proportional to the cost coefficient C^j is incurred. High values of C^j can be leveraged to constraint the agent from changing certain degrees of freedom for various process parameters. The overall impact of penalty for changing process parameters is controlled by a MAS permitted reconfigurability coefficient (λ), which can be used to limit the frequency of CAs. This accounts for MAS constraints such as permitted maintenance duration and frequency. The coefficient also enables trade-off between the benefits (RC related requirements) and costs (CA related requirements) of the reward function based on the current MAS reconfigurability. To account for RC estimation uncertainty r_t^{CA} is also multiplied by the uncertainty factor.

The reward further accounts for req. (iv) *MAS system inherent stochasticity* due to incoming non-ideal part variations and tooling variations by allowing the agent to take CAs only after observing a maximum sample of products s_p products rather than a single product. This is crucial there are varying levels of stochasticity due to batch-to-batch variation in incoming parts. CAs determined based on previous batch stochasticity may be ineffective if the stochasticity variation pattern and magnitude has changed in subsequent batches. The sample size can be determined based on the required confidence interval [4]. High values of s_p translate into increased confidence scores for MAS stochasticity but also into higher costs as more product samples would need to be measured. Overall, the reward function that the agent should maximise to determine CAs while accounting for the aforementioned requirements can be written as a combination of r_δ^{RC} and r_δ^{CA} as:

$$r_t = -(1 - \sigma^{RC_t}) \left(\frac{1}{s^p \cdot k^n} \sum_{p=1}^{s^p} \sum_{i=1}^{k^n} K^i (KPI_0^i - KPI_t^{i,p})^2 + \lambda \frac{1}{h} \sum_{j=1}^h C^j (y_{t+s^p}^j - RC_t^j)^2 \right) \quad (7)$$

Lastly the set of NPI-coefficients δ is leveraged to account for the stage of the NPI lifecycle. The set δ consists of:

$$\delta = \{K, C, \lambda, s_p\} \quad (8)$$

This includes set of relevant KPIs importance (K), adjustment costs of reconfigurable process parameters (C), MAS permitted reconfigurability (λ), maximum available data sample (s_p). The NPI-coefficients enable continual tuning of the reward based on the stage of NPI lifecycle. As the MAS goes through the lifecycle the priorities of different requirements related to benefits and costs can change. Using a set of user-parametrizable NPI-coefficients ensures that the CA methodology can be leveraged throughout the lifecycle by fine-tuning the NPI-coefficients based on requirements of the current stage. The coefficients are leveraged to adapt the reward function for agents to learn robust CA policies across different stages of the NPI lifecycle. Such examples for adaption are discussed in later sections. Intuitively in the design stage NPI coefficients are selected to focus primarily on improving quality KPIs while during production stages importance is given to both cost and quality KPIs.

Correction involves learning a deterministic CA policy $\pi(\cdot)$ which is a function that maps the current MAS state s_t^{MAS} into a CA. The CA physically translates into a new set of process parameters (or functional nominal) y_{t+s^p} , that can maximise reward r_t and eventually, eliminate the shape errors $x^{N_s, A}$.

$$CA_{t+s^p} = y_{t+s^p} = \pi(s_t^{MAS}) \quad (9)$$

A CA_t taken by the DDPG agent corrects one or multiple process parameters within the allowable ranges that aim to maximize the overall reward.

$$\mathbf{CA}_t = \mathbf{y}_{t+s_p} \quad (10)$$

In summary, the correction of MASs can be formalised as a Markov Decision process where an environment E , i.e., MAS, can exist in a set of possible states $s \in \mathcal{S} = \mathbf{s}_t^{MAS}$. CAs $a \in A = \mathbf{CA}_t$ are taken to correct the MAS to observe a real-valued reward $r_t(\mathbf{s}_t^{MAS}, \mathbf{a}_t)$ that ensures that the aforementioned requirements are met. The Markovian property, i.e., *the effects of an action taken in a state, depends only on that state and not on historical states*, is applicable for the correction process. Within an episode of sequential CAs only the previous MAS state is considered while determining the CA. The overall reward $R^{s_t^{MAS}}$ for a state \mathbf{s}_t^{MAS} is defined as the cumulative discounted sum over the next T time steps where γ is the discount factor that discounts rewards in the future and gives higher weights to current rewards. It should be noted that for correction of MAS, each timestep is a sample of \mathbf{s}_p products.

$$R^{s_t^{MAS}} = \sum_{i=t}^T \gamma^{(i-t)} r(\mathbf{s}_i^{MAS}, \mathbf{a}_i) \quad (11)$$

In practice, the MAS state and CAs are deterministic – each state and CA lead to a new state. The states are observable – the new state after a CA is known. The goal is to learn a CA policy π using DDPG that generates optimal CAs \mathbf{CA}_t based on the current MAS state \mathbf{s}_t^{MAS} that maximises the expected reward $R^{s_t^{MAS}}$.

3. METHODOLOGY: OBJECT SHAPE ERROR CORRECTION

The OSEC aims to enable shape error correction in an end-to-end manner by first leveraging RCA models to use manufacturing data (point clouds) to isolate RCs and estimate MAS state and then map the isolate RC and MAS state into effective and feasible CAs using DDPG within a MDP setup. The overall approach includes steps to:

- (i) *Isolate RCs and estimate MAS state*: Involves point cloud data $\mathbf{x}^{s=N_s, \delta=4}$ collection using 3D scanners (1). Further RCs, RC uncertainties $\sigma(\mathbf{RC}_t)$ and upstream shape errors $\mathbf{x}_t^{upstream}$ are estimated using the OSER model (2). These collectively represent the MAS state \mathbf{s}_t^{MAS} (4).
- (ii) *Account for CA requirements (RC and MAS related)*: Involves defining the reward function r_t (7) subject to RC uncertainty, MAS correction constraints (reconfigurability) and inherent stochasticity (part and process variation). Further allocation of NPI-coefficients $\delta = \{\mathbf{K}, \mathbf{C}, \lambda, s_p\}$ (8) is done based on the NPI and production lifecycle stage to optimally trade-off the aforementioned requirements.
- (iii) *Generate feasible and effective CAs*: The correction problem is formulated as an MDP as described in the previous section to bound the scope of possible CA policies $\pi(\mathbf{s}_t^{MAS})$ (9) that can be learnt and enable the application of DDPG. DDPG agent is leveraged as it can take as input real-valued multi-dimensional state estimates and output real-valued multi-dimensional CAs \mathbf{CA}_{t+s_p} (10). The DDPG architecture is optimized to enable the mapping of RCs to CAs based on the engineered reward function. This involves selecting the hyperparameters of the DDPG agent network architecture, matching the output layer nodes with the CA output dimension (i.e., adjustable process parameters) and selecting an exploration policy to comprehensively explore the performance of various CA policies. The DDPG agent is trained based on the engineered reward function to learn CA policies π (9) that can generate feasible and effective CAs. Finally, the trained DDPG agent is deployed to obtain CAs for elimination of RCs within MAS. The steps are summarized in Fig. 3.

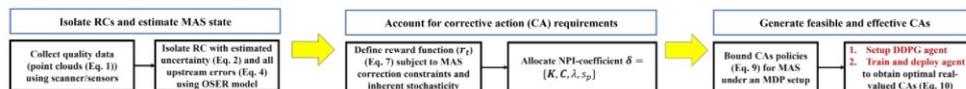


Fig. 3. OSEC approach

3.1 NPI-Coefficient Selection

The first step before training DDPG agent is engineering the reward function (see (5), (6) and (7)) to account for MAS correction requirements. This involves selecting the values for the user-defined NPI-coefficients $\delta = \{\mathbf{K}, \mathbf{C}, \lambda, s_p\}$ based on MAS KPIs, reconfigurability constraints and stochasticity that are dependent on the stage of NPI and production lifecycle [1]. From a MAS engineering perspective, different stages of the lifecycle entail different levels of reconfigurability and stochasticity. E.g. during design and pre-production stages, the costs of correcting process parameters (CAs) are much lower than when this is done in production stages [1]. The paper proposes guidelines for selecting NPI-coefficient δ values to optimally trade-off the importance of the requirements, costs and benefits through the design and production stages of the NPI lifecycle.

(i) *Design Stage* – During design, the MAS permitted reconfigurability is set to zero ($\lambda = 0$) as the design is done virtually within CAE simulation models resulting in near-zero restrictions on process parameter adjustments. This sets the $r_t^{CA} = 0$ and effectively changes the reward function to a weighted sum of improvement in KPIs multiplied by the RC uncertainty thus only focusing on the benefits and not the costs. The CA policies learnt with this reward aim to simultaneously maximise KPIs and minimise sensitivity to RCs (RC uncertainty) within the allowable system reconfigurability (process parameter ranges) hence

maximising product quality, process *robustness* and process *resilience*. Previously approaches [4] focused mainly on maximising KPIs under a predefined level of system stochasticity due to incoming part variations and tooling variations hence only maximising product quality and process robustness. Small values of $s^p \approx [3,10]$ are leveraged to ensure that all the factors that induce stochastic variations in the MAS are considered. Large allowable ranges $[y_{min}^j, y_{max}^j]$ are selected for all process parameters to allow for effective exploration of the design space. The adjustment costs of reconfigurable process parameters (C) are not required given the MAS permitted reconfigurability is set to zero. The agent is then trained over various stochastic sources such as part variations, tooling variations and system noise to learn a CA policy that yields optimal process parameter adjustments across all stochastic variations of the MAS.

(ii) *Production Stage* – To account for the substantial costs for adjusting process parameters during production stages [1], the MAS permitted reconfigurability can be fine-tuned within ranges $\lambda = [0.1, 1]$ to allow for changes in process parameters only when the impact in KPI improvement is significant. Relatively larger values of $s^p \approx [30,40]$ are leveraged to account for inherent MAS stochasticity. The relevant KPIs importance (K) and adjustment costs of reconfigurable process parameters (C) can be selected based on the MAS design architecture and constraints. In this scenario, CAs for only those process parameters that have a low-cost coefficient C^j would be estimated. Process parameters that cannot be changed during production have high values of C^j . The allowable ranges $[y_{min}^j, y_{max}^j]$ for adjustable process parameters is limited based on the permissible design tolerance, which is relatively smaller than the ranges during design given the process constraints. The NPI-coefficients (δ) can be further fine-tuned such that CA policies achieve a good trade-off between RC requirements (benefits) (r_t^{RC}) and MAS requirements (costs) (r_t^{CA}). Table I summarizes the guidelines to select values of the NPI-coefficients based on the NPI stage. After selecting NPI-coefficients CA policies bound by the MDP assumption are learnt using DDPG and the engineered reward function.

TABLE I: NPI-COEFFICIENT SELECTION

NPI-Coefficient	NPI Stage	
	Design	Production
Relevant KPIs importance (K)	High importance to quality	Importance to both cost and quality
Adjustment costs of reconfigurable process parameters (C)	No adjustment costs	High adjustment costs based on MAS constraints
MAS permitted reconfigurability (λ)	0 (high reconfigurability)	[0.1,1] (limited reconfigurability)
Maximum available data sample (s_p)	[3,5] / [5,10] (small sample sizes) Prototype batches available at design stage between early and more mature design	[50,100] (large sample sizes) As a rule of thumb production volume between breaks; often equals to 2 hours of production

3.2 Deep Deterministic Policy Gradient (DDPG)

DDPG is a model-free, off-policy actor-critic algorithm [28] that leverages neural networks to learn approximate CA policies π and CA-value (Q-value) functions Q^π in multi-dimensional real-valued CA spaces. The algorithm leverages the deterministic policy gradient (DPG) framework and approximates the CA policy function using a neural network known as the actor and CA action-value function using a neural network known as the critic. The actor estimates a CA while the critic generates the corresponding Q-value (expected reward) for the CA. Key insights from Deep Q-Networks (DQN) [25], such as samples from a replay buffer to train the network off-policy minimising correlations and using target networks, are leveraged to give consistent targets during temporal difference updates. The DPG algorithm leverages an actor (μ) parametrised by weights and biases of a neural network (θ^μ) which generates the current CA policy by a learning a deterministic mapping from the current state (RCs) to the CAs $\mu(s_t^{MAS} | \theta^\mu)$. The critic network takes as input the CA and the state $Q(s_t^{MAS}, CA_t | \theta^Q)$ to generate the value of the current CA policy.

The CA action-value (Q-value) function is used to describe the expected return after taking an CA CA_t in state s_t^{MAS} and following the given CA policy π :

$$Q^\pi(s_t^{MAS}, CA_t) = E_{r_t \geq t, s_t \geq t \sim E, CA_t \geq t \sim \pi} [R_t | s_t^{MAS}, CA_t] \quad (12)$$

The Q-value can also be represented recursively using the Bellman's equation. Given a target deterministic policy Bellman's equation can be expressed as:

$$Q^\mu(s_t^{MAS}, CA_t) = E_{r_t, s_{t+s_p}^{MAS} \sim E} [r(s_t^{MAS}, CA_t) + \gamma Q^\mu(s_{t+s_p}^{MAS}, \mu(s_{t+s_p}^{MAS}))] \quad (13)$$

Given Q^μ is only dependent on the stochasticity of the environment; it is learnt off-policy using transitions generated from a different stochastic behaviour policy β . The loss function minimised to optimise the parameters θ^Q of the critic is expressed as:

$$L(\theta^Q) = E_{s_t \sim \beta, CA_t \sim \beta, r_t \sim E} [(Q(s_t^{MAS}, CA_t | \theta^Q) - p_t)^2] \quad (14)$$

The term is known as temporal differencing (TD) error, where

$$p_t = r(s_t^{MAS}, CA_t) + \gamma Q(s_{t+s_p}^{MAS}, \mu(s_{t+s_p}^{MAS})) - Q(s_t^{MAS}, CA_t | \theta^Q) \quad (15)$$

The loss function (14) is minimized iteratively to train the critic. Simultaneously after updating the critic parameters, the actor parameters θ^μ are updated to obtain policy π by applying the policy gradient theorem to the training objective J [28] and obtaining

the gradient for actor parameters θ^μ as:

$$\nabla_{\theta^\mu} J = \mathbb{E}_{s_t \sim \beta} \left[\nabla_{\theta^\mu} Q(s_t^{MAS}, CA_t | \theta^Q) \Big|_{s_t \sim s_t^{MAS}, a_t \sim \mu(s_t^{MAS} | \theta^\mu)} \right] \quad (16)$$

The parameter update while training the actor and critic is done by gradient descent based models such as the Adam [29], which assume that observations are independently and identically distributed (IID). This is not the case when MAS state, CA and reward samples are sequentially generated and temporally correlated. Various techniques are leveraged to stabilise the learning process. (i) *A replay buffer* [25] is leveraged, which consists of a finite-sized cache and stores sequential transitions from the MAS environment. A transition is quantified as a tuple of the MAS state, CA, reward and resulting state $(s_t^{MAS}, CA_t, r_t, s_{t+s_p}^{MAS})$. While training, a random *minibatch* is uniformly sampled from the buffer. This minibatch consists of temporally decorrelated samples and hence aids to stabilize training. As the replay buffer is full, the oldest transitions are discarded in a First In First Out (FIFO) setting. (ii) *Target critic and actor networks* [25] are also used to account for the divergence issue caused as the critic network $Q(s, a | \theta^Q)$ being updated is also used in calculating the target value. The weights of the target networks are constrained to update slowly compared to the learned networks:

$$\theta' \leftarrow \tau \theta + (1 - \tau) \theta' \quad (17)$$

where $\tau < 1$. Although this slows the learning process in terms of the propagations of the value estimations, there is a significant increase in learning stability and a decrease in the sensitivity to actor-critic network architecture hyperparameters.

Learning in multi-dimensional real-valued action spaces also requires an effective exploration of different CA policies that can generate feasible and effective CAs under the aforementioned requirements. Given the off-policy nature of DDPG, an exploration policy μ' independent of the learning algorithm is chosen:

$$\mu'(s_t^{MAS}) = \mu(s_t^{MAS} | \theta^\mu) + \mathcal{N} \quad (18)$$

Where \mathcal{N} is generated using an Ornstein-Uhlenbeck process [30] that generates temporally correlated noise, the generated noise is added to corrective action estimated by the actor-network, which is then leveraged for transition and generation of the next state. This serves as an effective exploration strategy β for CA policies. The exploration ensures that the agent can search and test various CAs, observe the generated rewards that account for MAS benefits, costs, stochasticity and NPI stage and finally learn CA policies that generate CAs that are effective and feasible given the current state of the MAS.

3.3 DDPG Actor-Critic Network Optimization & Training

The performance of DDPG to generate feasible and effective CAs is heavily dependent on careful hyperparameter optimization of the actor and critic networks (Fig. 4). The paper uses the original architecture as a baseline [28] and optimises it enable CA estimation. The actor network μ that estimates the CA policy function is optimized to contain two dense hidden layers with 128 and 64 hidden nodes with ReLU activations. The output layer of the actor is modified to have neurons equal to the CA_t dimension, i.e., the number of controllable/adjustable process parameters \mathbf{y} . The output layer leverages Tanh activation to bound the CA within the process parameter allowable limits.

The optimised critic network Q (action-value network) consists of two heads with 128 and 64 nodes that input the state s_t^{MAS} and action CA_t . These are concatenated and given as input to the next layer with 64 nodes with ReLU activation. The output layer consists of a linear activation and estimates the CA value for the MAS state and CA $Q^\pi(s_t^{MAS}, CA_t)$ (13).

Further optimizations are done to enable robust and stable learning. Batch normalization is leveraged as this scales all features to have similar ranges and minimises covariance shift in the deep networks as each layer within the network receives a whitened input [28]. A replay buffer of size 10^5 is leveraged with a minibatch size of 64. This converts sequential samples from consecutive states into near independent and identically distributed samples, reducing divergence for actor-critic network parameters during training. Initializations of final layer weights were done within small ranges $[-3 \times 10^{-3}, 3 \times 10^{-3}]$ to ensure that initial CAs and values of CAs were near zero to enable quicker learning, especially during initial training stages. The MAS state vector s_t^{MAS} is scaled between $[-1, 1]$, and the reward r_t is scaled between $[0, 1]$. Such scaling is crucial for the network to train comparatively faster and reduce the risk of divergence. The Adam [29] with learning rate $\alpha = 0.001$ is used for training. The discount factor is set to $\gamma = 0.5$ (15) to give higher importance to current CAs (myopic agent). $\tau = 0.01$ (17) is used for soft target updates. The action exploration temporally correlated noise is generated using the Ornstein-Uhlenbeck process with $\theta = 0.15$ and $\sigma = 0.2$ (18). Each episode is run until termination. *Episodes are terminated when the difference in KPI compared to the nominal is below a pre-determined threshold*. A maximum of 500 training runs are done for each episode to account for the computer-aided engineering (CAE) simulation [27] time and computational budget constraints. Testing is done after training for 200 episodes at an interval of 20 episodes. VRM [27] is used as the CAE environment for training. The work has been implemented using Python 3.7 and TensorFlow - GPU 2.1. A python library [31] has been developed to validate and replicate the results of the methodology. Two Nvidia Tesla V100 32 GB GPUs are used for model training and deployment.

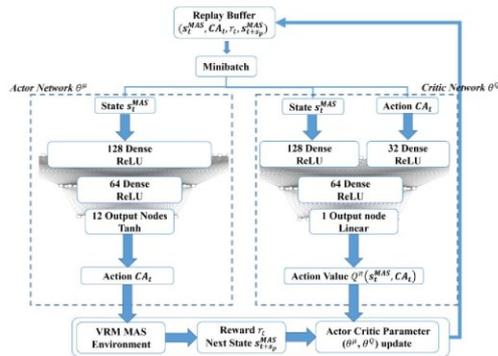


Fig. 4. DDPG agent

3.4 DDPG Deployment

After training, the actor of the DDPG agent is deployed within the MAS (Fig. 5). The overall steps within the deployment framework include:

- (i) *Product assembly*: the physical MAS assembles s_p products,
- (ii) *Collect point cloud data*: point cloud data for the products are collected at the end of the MAS using a 3D scanner and processed into object shape error data $\mathbf{x}^{s=N_s, \delta=4}$ (1),
- (iii) *Isolate RC and estimate MAS state*: based on the object shape error data, the RCs are isolated and the MAS state s_t^{MAS} is estimated using previously proposed OSER models (see (2), (3) and (4)),
- (iv) *Correct MAS using trained DDPG model*: the actor $\pi(s_t^{MAS})$ (CA policy function) estimates feasible and effective CAs CA_t (process parameter adjustment) (see (9) and (10)). The process parameter adjustment values are given as input to process controllers or maintenance is scheduled.

The CA policies generate new process parameter values known as the *functional nominal*, which can be interpreted as the value of process parameters that maximizes MAS KPIs given the current level of stochasticity while accounting for the constraints and costs. This may be different from the *design nominal* as this is learnt dynamically given the MAS's current state, while the design nominal is pre-determined based on limited modelling of all possible physical interactions. Current SPC/APC assume that the design nominal is permanently fixed and gives the best performance. CAs using these approaches only minimise the difference between current process parameters values and the design nominal. However, such CAs are frequently ineffective and infeasible to implement.

It should be noted that the OSEC DDPG agent *does not perform control actions*; instead, it behaves as an enabler by generating the process parameter adjustment (functional nominal) for the process controller/maintenance. This is also known as the process parameter setpoint and is generally assumed to be the design nominal in SPC/APC literature. Conventional controllers perform the task of going from the current process parameter to the new process parameter setpoint under process disturbances while minimising delay and overshoot. Using the controller also ensures an additional layer of safety and stability while correcting process parameters. Before deploying the agent in physical MAS, extensive testing should be carried out in the simulation environment (VRM) to ensure that CA policies have converged. During the initial stages, a human within-the correction loop can also be leveraged to ensure that the CA policy is safe and stable. The paper leverages VRM as the simulation environment for both training and deployment. The overall framework for training and deployment of the DDPG based OSEC approach is summarized in Fig. 5.

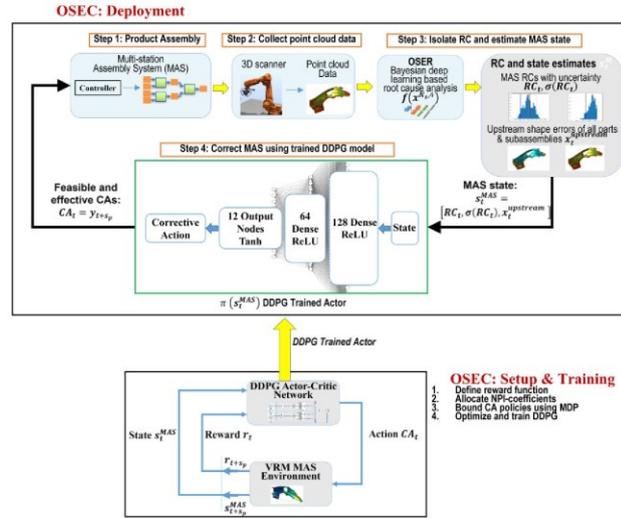


Fig. 5. OSEC training and deployment framework

3.5 Engineering Interpretation of CA Policies

From a MAS engineering perspective, CAs are one or multiple process parameter adjustments generated sequentially by a CA policy that mitigates one or multiple RCs, leading to quality defects elimination and KPI improvement. Given a MAS with h process parameters \mathbf{y} and no RCs, all values of \mathbf{y} are at their design nominal $\tilde{\mathbf{y}}$ (considered 0 for simplicity). Generally, it is assumed that after designing the MAS, the MAS is a deterministic function $\phi: \mathbb{R}^h \rightarrow \mathbb{R}^{k^n}$ that maps h process parameters to k^n KPIs (KPIs = $\phi(\mathbf{y})$). The function ϕ is assumed to have a single optimum, i.e., nominal KPI performance when $\mathbf{y} = \tilde{\mathbf{y}}$. In the presence of RCs ($\mathbf{RC}_t = \mathbf{y}_t$) the value of \mathbf{y} is estimated by using various RCA models [11] or process sensors. A CA, i.e., a process parameter adjustment, is determined to eliminate the RC. Hence in a general case (i.e., a single optimum for the ϕ exist, ϕ is a deterministic assembly function with no constraints on \mathbf{y}), there exists a *one-to-one mapping between CAs and RCs* such that RCs can be corrected and optimal KPI performance can be achieved by leveraging

$$CA = \tilde{\mathbf{y}} - \mathbf{RC}_t = -\mathbf{RC}_t \quad (19)$$

This correction policy is known as *Mean RC Estimates policy* π^0 which generates CAs, which are equivalent to the difference between the mean value of the RC \mathbf{RC}_t and design nominal $\tilde{\mathbf{y}}$. This policy is currently used in manual approaches and also in SPC/APC methods $CA = \tilde{\mathbf{y}} - \mathbf{y}_t$ that aim to revert the process parameters to the design nominal.

However, as outlined, CA and RC requirements pose various challenges in determining optimal CAs using π^0 policy mapping. Firstly, given various RCA models use an estimator function to estimate \mathbf{RC}_t , this can result in a biased and high variance estimate of the actual value of RCs. This requirement is known as (i) *RC estimation uncertainty*. OSER [11] leverages Bayesian deep learning to estimate a distribution on RC: $[\mathbf{RC}_t, \sigma(\mathbf{RC}_t)]$. Generating CAs now requires mapping a distribution to process parameter adjustments (CAs). This means multiple CAs can now be mapped by drawing multiple samples from the RC distribution. Using only the mean of this distribution will give lower convergence and will require a higher number of sequential process adjustments to converge to the optimal CA that reverts the system to $\tilde{\mathbf{y}}$. Alternatively, the proposed OSEC approach learns a policy π^{OSEC} aims to account for this by factoring in an uncertainty factor which results in the reward $r_t^{RC} = -(1 - \sigma^{RC_t}) \frac{1}{k^n} \sum_{i=1}^{k^n} K^i (\phi(\mathbf{RC}_t)^i - \phi(\tilde{\mathbf{y}})^i)^2$. This ensures the agent is only rewarded for CAs when RC uncertainty is low; if the RC uncertainty is high, the agent is not rewarded irrespective of its CA. This introduces an ability to *wait* and observe more MAS states before the RC uncertainty increases and the respective CAs are assigned higher rewards.

The other three requirements state that the assumed design nominal $\tilde{\mathbf{y}}$ may not be the value for optimal performance of ϕ and hence a *functional nominal* $\tilde{\mathbf{y}}$ should be estimated, and then the CA should be determined as:

$$CA = \tilde{\mathbf{y}} - \mathbf{RC}_t \quad (20)$$

The goal of the OSEC CA policy π^{OSEC} is to also account for the remaining three requirements while estimating a functional nominal and then generate CAs. The requirements are accounted for as follows:

(ii) *KPI improvement*: In practice, MAS systems are ill-conditioned, there may not exist a unique value $\tilde{\mathbf{y}}$ that generates nominal performance for all KPIs. Multiple process parameter configurations can exist that would lead to local optimum for a subset of KPIs. OSEC aims to handle this by leveraging optimal relevant KPIs importance \mathbf{K} for all KPIs. This constructs a weighted sum of KPIs, which can be maximized by estimating the functional nominal $\tilde{\mathbf{y}}$ after which the CA can be determined using (20).

(iii) *MAS design architecture* entails that not all process parameter adjustments can be implemented due to constraints in MAS reconfigurability, production schedules and adjustment costs that limit the feasible space of CAs. In such scenarios, the CA that reverts the MAS to the design nominal $\tilde{\mathbf{y}}$ may be infeasible to achieve in practice; hence a functional nominal $\tilde{\mathbf{y}}$ that achieves similar KPI performance must be determined to estimate CAs (20). OSEC aims to leverage NPI-coefficients namely adjustment costs of reconfigurable process parameters \mathbf{C} and MAS permitted reconfigurability λ to penalize the agent from learning CA policies that output infeasible CAs.

(iv) *MAS inherent stochasticity* due to factors such as non-ideal part variations entails that a deterministic estimate of the MAS $\phi: \mathbb{R}^h \rightarrow \mathbb{R}^{k^n}$ will not exist. Hence a unique value for $\tilde{\mathbf{y}}$ cannot be determined. As ϕ varies due to change in batch-to-batch stochasticity, the optimal $\tilde{\mathbf{y}}$ will also vary, which will result in CAs being ineffective under different stochastic variations. OSEC aims to learn CAs policies that dynamically account for other system information apart from RCs such as upstream shape errors by augmenting the state information with upstream shape errors ($\mathbf{x}^{upstream}$) such that the state is represented as $\mathbf{s}_t^{MAS} = [\mathbf{RC}_t, \boldsymbol{\sigma}(\mathbf{RC}_t), \mathbf{x}^{upstream}]$. It also aims first to observe a sample of \mathbf{s}_p products which provide estimates for current stochasticity and a confidence level before determining the functional nominal and the corresponding CA.

Overall policies π^0 and π^{OSEC} fundamentally differ on how CAs are generated. CAs generated by π^0 aims to transition the MAS from the current process parameters (RCs) to the design nominal, while CAs generated by the OSEC policy π^{OSEC} aims to transition the system to a functional nominal that is determined by maximizing a combination of KPI improvement and MAS requirement penalty function:

$$Max \left(-(1 - \sigma^{yt}) \left(\frac{1}{s^p * k^n} \sum_{p=1}^{s^p} \sum_{i=1}^{k^n} K^i (\phi(\mathbf{RC}_t)^i - \phi(\tilde{\mathbf{y}})^{i,p})^2 + \lambda \frac{1}{h} \sum_{j=1}^h C^j (y_{t+s^p}^j - \mathbf{RC}_t^j)^2 \right) \right) \quad (21)$$

If the effect of the aforementioned requirements is insignificant and the MAS requirement change penalty is zero ($\lambda=0$). Theoretically, the functional nominal will be the same as the design nominal ($\tilde{\mathbf{y}} = \tilde{\mathbf{y}}$) and the policies will converge towards π^0 (20), i.e., restoring the MAS to the design nominal. Under conditions of RC uncertainty and MAS stochasticity the π^0 CA policy might perform poorly but after convergence π^{OSEC} will always perform as good as π^0 . In summary CA policies must account for all these requirements related to benefits, costs, stochasticity and NPI stage to ensure effective and feasible CAs are implemented that simultaneously improve KPIs, reduce machine downtime and eliminate scrap in a cost-efficient manner.

4 CASE STUDY

4.1 Experimental Setup

For verification and validation of the OSEC approach, an industrial, automotive cross member assembly is selected. It consists of $n = 4$ non-ideal compliant parts, namely, pocket, pocket reinforcement, cross member and cross member reinforcement (Fig. 6) and $N_s = 3$ stations, with each station consisting of four stages (Fig. 7). The assembly is controlled by $h = 12$ process parameters y^1, y^2, \dots, y^{12} (Table II). The upstream shape errors are consolidated into a fixed-length vector $\mathbf{x}^{upstream}$ with dimensionality $d = 64 \times (N_s - 1) = 128$ (3). The overall state vector \mathbf{s}_t^{MAS} (4) consists of $\mathbf{x}^{upstream}$ and $\mathbf{RC}: \{y^1, y^2, \dots, y^{12}\}$ and has a dimensionality of $h + d = 140$. The CA space $\mathbf{CA}_t: \{y^5, y^6, \dots, y^{12}\}$ (10) has a dimensionality equal to the number of adjustable process parameters (8 process parameters). The CA policy learnt for this case study will account for all the aforementioned requirements: (i) *RC estimation uncertainty* as the state estimation OSER [11] model will estimate uncertainties with the RCs (2); (ii) *KPIs improvement (benefits)* related to shape error of the final assembly; (iii) *MAS system design architecture (costs)* as frequent changes are not permissible depending on the stage of the NPI cycle. Additionally, process parameters have different costs and range of allowed readjustments; (iii) *MAS system inherent stochasticity* given the incoming part variations within non-ideal parts as induced by uncontrollable (*stochastic*) process parameters y^1, y^2, y^3, y^4 .

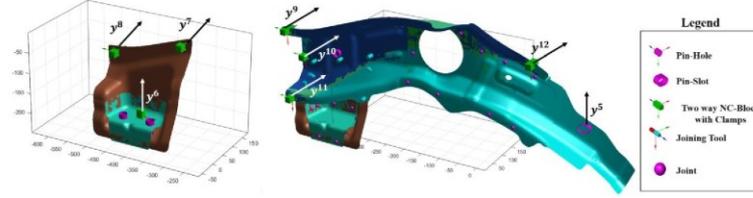


Fig. 6. Cross member assembly

Each training step consists of s^p samples being generated. Each episode is run until the maximum number of training steps are reached, or the terminal state is reached. Terminal state within each episode is defined when the mean absolute shape error of the final assembly across s^p samples are below the set threshold set as 0.05 mm, as variations more minor than this are not detectable by the 3D scanner. The goal within scenarios is to learn CA policies that can adjust one or multiple process parameters sequentially and take the MAS from any state (i.e., state with one or multiple RCs) that has shape error to a terminal state (i.e., state with no RCs) where the shape error is below the threshold. An episode is terminated if the terminal state is not reached after 500 training steps, i.e., a total of $500 * s^p$ product samples. This termination criterion is introduced to limit the VRM CAE simulation time. As each computation can take up to five minutes for computation, allowing the training to run without a hard stopping criterion will require exponentially more time and computation resources.

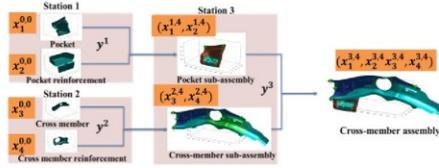


Fig. 7. Multistation cross member assembly topology

TABLE II: PROCESS PARAMETERS DESCRIPTION

Process Parameter	Description	Type
y^1, y^2, y^3, y^4	Part variation parameters	Uncontrollable (stochastic)
y^5	Pin-slot displacement in x	Adjustable (CA)
y^6	Clamp displacement in x	Adjustable (CA)
y^7	Clamp displacement in y	Adjustable (CA)
y^8	Clamp displacement in z	Adjustable (CA)
$y^9, y^{10}, y^{11}, y^{12}$	Clamp displacements in y	Adjustable (CA)

4.2 Application Scenarios

As described in Section 3.4, different values of the NPI-coefficients are leveraged to engineer the reward function of the proposed OSEC DDPG framework for the correction of MAS. The paper applies the framework for three scenarios:

(i) *Scenario 1 (Design Stage)* – To leverage OSEC during design stages, the MAS permitted reconfigurability $\lambda = 0$ is set to zero as the design is done within CAE simulations. The absolute value of shape errors after the final station is considered as the KPIs. A maximum sample size of $s^p = 10$ is leveraged. The reward function for this scenario can be written as:

$$r_t = -(1 - \sigma^{RC_t}) \left(\frac{1}{s^p * k^n} \sum_{p=1}^{s^p} \sum_{i=1}^{k^n} (|x_i^{3,4}|)^2 \right) \quad (22)$$

The relevant KPI importance for shape error across all points is assumed constant and equal to 1. k^n consists of all points across the assembly, $k^n = n_1 + n_2 + n_3 + n_4 = 10841$. The process parameter ranges for CAs are set as $[-4 \text{ mm}, 4 \text{ mm}]$ for all process parameters for efficient exploration of design space.

(ii) *Scenario 2 (Production Stage)* – To leverage OSEC during production stages, the MAS permitted reconfigurability of $\lambda = 0.5$ and maximum sample size of $s^p = 30$ is leveraged. The reward function is:

$$r_t = -(1 - \sigma^{RC_t}) \left(\frac{1}{s^p * k^n} \sum_{p=1}^{s^p} \sum_{i=1}^{k^n} (|x_i^{3,4}|)^2 \right) + \lambda \frac{1}{h} \sum_{j=1}^h C^j (y_{t-s^p}^j - RC_t^j)^2 \quad (23)$$

The process parameter ranges for CAs is set as $[-1 \text{ mm}, 1 \text{ mm}]$ for all process parameters as the flexibility to change process parameters is limited during production stages. The adjustment costs for y^5 is set to $C^1 = 0.7$ while all others it is set to $C^j = 0.9$ $j = \{2, 3, \dots, 8\}$ as costs related to changing clamps are higher than adjustments for pin-slot. All other NPI-coefficients are the same as for Scenario 1.

(iii) *Scenario 3 (Production Stage)* – Further to leverage OSEC during production stages for MAS that cannot be reconfigured significantly, the MAS permitted reconfigurability is set to $\lambda = 1$ so that process parameters are only adjusted only when the effect on KPI is significant. A maximum sample size of $s^p = 40$ is leveraged to ensure changes are only done after observing a significantly large number of products. The process parameter ranges for CAs are set as $[-0.5 \text{ mm}, 0.5 \text{ mm}]$ for all process parameters as these are permissible tolerance limits during production stages. The reward function and remaining NPI-coefficients are the same as for Scenario 2.

4.3 Results

Ten training trials are done for the actor-critic DDPG network, each with 500 episodes. To reduce sensitivity to initial parameters, all trials use the same seed for initialization. Testing is done for 200 episodes. The exploration policy is not used while testing. The agent can successfully learn optimal CA policies for all scenarios. The overall average metrics of the ten trials for all three scenarios are summarised in Table III. Fig. 8 summarises the normalised averaged reward for all ten trials while training in all three scenarios. Fig. 9 summarises the average training steps per episode while training for all ten trials. The failed episodes quantify the percentage of test episodes that did not go below the required threshold after 500 steps.

The results demonstrate how CA policies differ across different stages of the NPI lifecycle. CA policies during design (Scenario 1) converse faster with lesser number of CAs per episode and have lower object shape error as compared to CA policy during production (Scenario 2). This is because the MAS requirements are significant during production stages resulting in the requirement of higher number of CAs to reduce shape error below the required threshold. Further comparison between scenario 2 and 3 demonstrates the difference in CA policies under different MAS permitted reconfigurability. The final reward for Scenario 3 is lower given the high penalty for MAS related requirements as the MAS permitted reconfigurability is minimum in Scenario 3. This also results in Scenario 3 having the highest failure rate of 5.1% due to the high MAS penalty (low reconfigurability) which prohibits the agent from frequently changing process parameters even during situations when the shape error is above the minimum threshold.

TABLE III: OSEC RESULTS

	Design		Production			
	Scenario 1		Scenario 2		Scenario 3	
	Mean	SD	Mean	SD	Mean	SD
Total training time	87.6 hrs	5 hrs	255.3 hrs	8 hrs	369.14 hrs	16 hrs
Average training steps per episode	57	8	61	12	74	20
% Failed episodes	2.1%	0.3%	4.3%	0.6%	5.1%	1.1%
Normalized test reward	0.97	0.05	0.79	0.08	0.68	0.12
Test mean absolute shape error	0.2 mm	0.05 mm	0.17 mm	0.04 mm	0.11 mm	0.03 mm

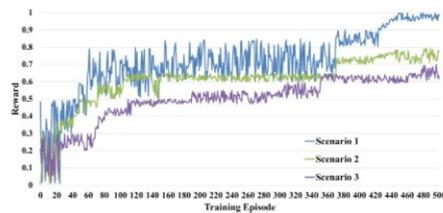


Fig. 8. Training reward per episode

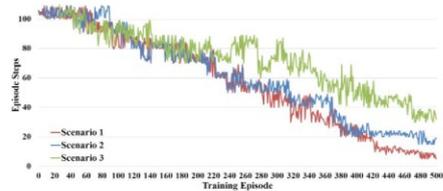


Fig. 9. Training steps per episode

The estimated CA policies provide the required CAs (process parameter adjustments) to significantly reduce shape errors. After training, the process parameter adjustments for CA policies are analysed across all three scenarios. The policies learnt in Scenario 1 converge to terminal states much faster due to zero MAS requirement penalty, while Scenarios 2 and 3 take more steps to converge to the terminal states given CAs have to be sequentially generated while ensuring that the MAS requirements costs do not dominate the KPI improvement and RC elimination benefits. Fig. 10a, 10b and 10c summarises the behaviour of agent when incoming stochasticity i.e. y^1, y^2, y^3, y^4 follow a normal distribution with $\mu = 0 \text{ mm}, \sigma = 0.1 \text{ mm}$. CA taken by the agents ensure that all process parameters y^5, \dots, y^{12} can converge at the functional nominal where the shape error is minimum, although with different number of steps and within different ranges. The object shape error outputs after different steps for the policies learnt for all scenarios are shown in Figs. 11a, 11b and 11c.

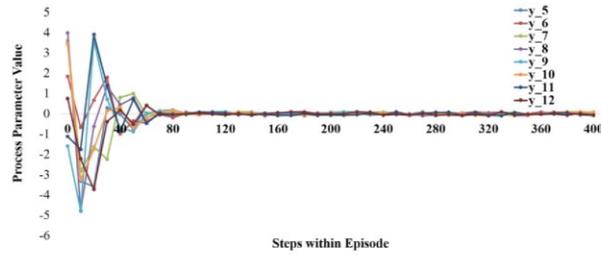


Fig. 10a. Policies for Scenario 1

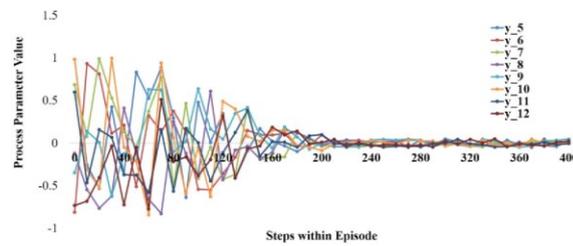


Fig. 10b. Policies for Scenario 2

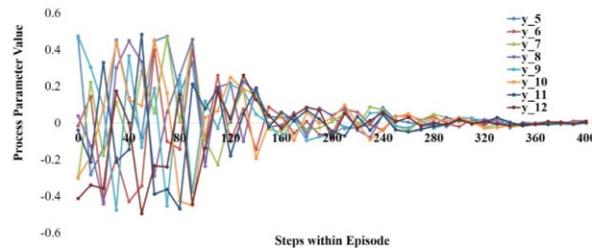


Fig. 10c. Policies for Scenario 3

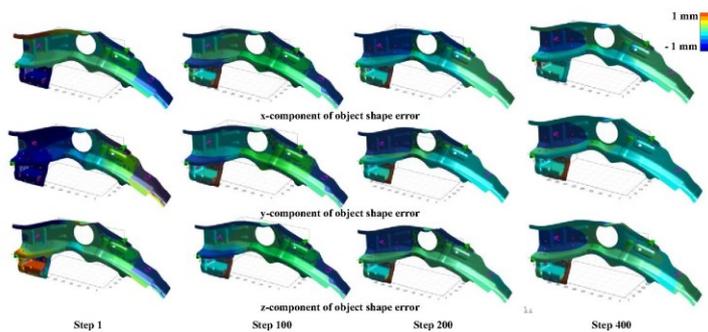


Fig. 11a. Object shape error outputs for scenario 1

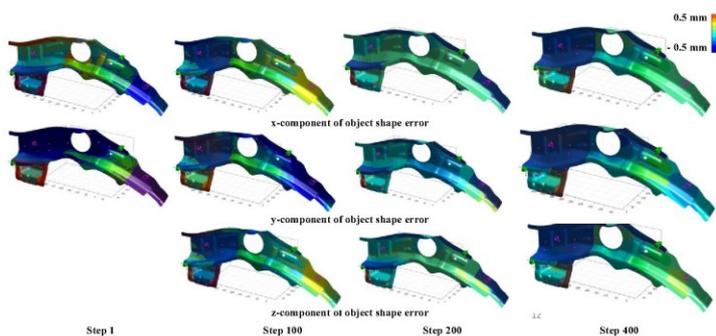


Fig. 11b. Object shape error outputs for scenario 2

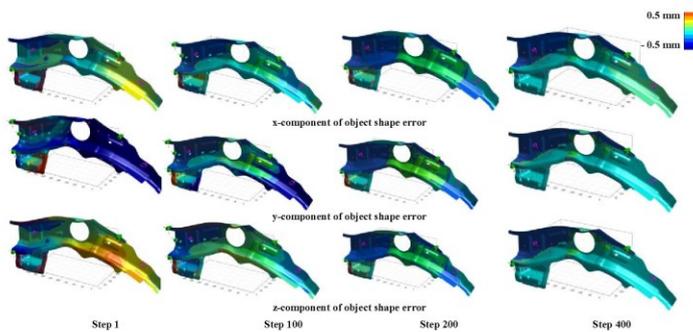


Fig. 11c. Object shape error outputs for scenario

4.4 Benchmarking

Scenario 3 is selected as benchmarking scenario, given that the NPI-coefficients for the reward function selected for this scenario ensure all requirements are taken into account. This scenario consists of cases in which $RC \neq CAs$ given less MAS permitted reconfigurability ($\lambda = 1$) which translated into CAs not being feasible to implement. Additionally, varying levels of incoming part variations are considered which result in CAs that are generated using a fixed design nominal being ineffective. The Benchmarking of the OSEC CA policy is done against three CA policies, namely:

(i) *Mean RC estimates π^0* – in this, the process parameters are changed based on the estimated mean value of the process parameter \mathbf{y} (RC) by the OSER model (2). The difference between the design nominal and mean RC is estimated as the CA. This is case for first group of approaches that map data to RCs and to CAs based on manual expertise ($CA = -RC$)

(ii) *SPC/APC based approaches π^1 (DDPG with basic KPI improvement reward)* – in this, DDPG with the same architecture as proposed in the paper is leveraged, but the reward function is limited to KPI improvement $r_t = -(\frac{1}{s^p+k} \sum_{p=1}^s \sum_{i=1}^k (|\mathbf{x}_i^{3,4}|)^2)$ (i.e., requirement (ii)). Other requirements such as RC uncertainty, MAS constraints and stochasticity are not considered. This case is the *upper bound* for second and third group of approaches that directly map data to CAs ($CA = \Delta y$) such as SPC/APC assuming a fixed design nominal and do not consider the NPI stage.

(iii) *OSEC π^{OSEC} (DDPG with reward engineering)* – in this, the proposed DDPG agent and the engineered reward function consisting of the NPI-coefficients (7) accounting for all requirements is leveraged to learn the policy. The difference between the functional nominal as estimated by the CA policy and the current value of the process parameter (RC) is estimated as the CA.

For the testing of all three CA policies, 300 episodes are run until the terminal criteria of shape error (KPI) being below the threshold or the maximum number of steps (computational budget) is reached. For each episode, the stochastic non-ideal part variation parameters y^1, y^2, y^3, y^4 follow a normal distribution with a randomly sampled mean between [-1 mm, 1 mm] for each episode while the standard deviation is always fixed at 0.1 mm. The initial values for all process parameters are randomly sampled between the allowable ranges [-0.5 mm, 0.5 mm]. Comparison is made for key benchmarking metrics, namely: (i) mean absolute object shape error (quality KPI) across all episodes, this quantifies the benefits of the CA policies; (ii) mean MAS requirement penalty across all episodes, this quantifies the costs (in terms of process parameter adjustments) of the CA policies to get the KPI below the required threshold; (iii) mean steps per episode this quantifies the converge as the number of sequential adjustments (CA) to get the KPI below the required threshold; and, (iv) percentage of failed episodes that did not converge, i.e. shape error was still beyond the threshold after the maximum number of steps. The results are summarised in Table IV.

The OSEC policies perform better across all benchmarking metrics. The overall mean absolute shape error (quality KPI) is limited to 0.14 mm as compared to 1.2 mm for mean RC estimates change policy (first group of approaches) and 0.35 mm for SPC/APC based approaches (second and third group of approaches). The cost in terms of MAS requirement penalty is also limited to 0.25 mm for the OSEC CA policy as compared to 0.64 mm for SPC/APC approaches, highlighting the ability of the policy to estimate functional nominals that maximize KPI performance under the current level of stochasticity and are within the MAS reconfigurability constraints. The mean steps per episode is lower for SPC/APC CA policy as compared to OSEC CA policy given it assumes a fixed design nominal, but it has a high failure rate of 21.5%, which is caused by CAs that are ineffective when applied given the varying levels of stochasticity. The OSEC CA policy estimates a functional nominal under the given level of stochasticity hence resulting in a failure rate of only 6.2%. This ensures that the effective convergence of the CA policy under varying levels of stochasticity is better than other groups of approaches. Overall, the OSEC policies give a higher and robust performance by giving higher quality KPI improvement (shape error reduction) and lower MAS penalty costs while converging faster under varying levels of stochasticity.

TABLE IV: BENCHMARKING

Benchmarking Metric Policy	Mean absolute shape error (Quality KPI: Object Shape Error)		Mean MAS penalty (Costs)		Mean steps per episode (Convergence)	% Failed episodes
	Mean (mm)	SD (mm)	Mean (mm)	SD (mm)		
Mean RC estimates (π^0) ($CA = -RC$)	1.2	0.47	0.79	0.33	380	64%
SPC/APC based approaches (π^1) (DDPG agent with basic KPI improvement reward) ($CA = \Delta y$)	0.35	0.11	0.64	0.24	330	21.5%
OSEC (π^{OSEC}) (DDPG agent with engineered reward) ($CA = \pi(s_i^{MAS})$)	0.14	0.06	0.25	0.12	270	6.2%

5 CONCLUSIONS

The paper presented a deep reinforcement learning-based methodology, Object Shape Error Correction (OSEC), for multi-station assembly systems, to mitigate root cause(s) of dimensional and geometric product shape errors. It leverages Deep Deterministic Policy Gradient and formulates a reward function parametrized by NPI-coefficients that enable mitigation of shape errors throughout the NPI and production lifecycle. It enables estimating real-valued corrective actions while accounting for root cause estimation uncertainty, KPI improvement, MAS design architecture and inherent stochasticity. The approach is compared under different scenarios. Results and benchmarking demonstrated better performance in terms of quality and cost KPIs and quicker

convergence under stochasticity when compared to current approaches for CAs. The OSEC is able to learn CA policies that adjust process parameters at a required frequency to optimally balance the benefits and cost priorities depending on the NPI stage. Under cases of low reconfigurability the CA policies take a greater number of steps to converge but ensure that reconfigurability constraints are not violated. The ability of CA policies to generate functional nominal ensures that when the stochasticity varies, or the RC uncertainty is high the corresponding effective CAs are generated. The CA approach is crucial for enhancement of productivity by reducing machine downtime and scrap. The approach also considers six-sigma quality and process indicators ensuring that the CAs improve quality KPIs and minimize the costs incurred due to CA implementation. Future work will expand the framework to include preventive actions (PA) aimed at changing downstream process parameters to compensate for upstream shape variations. Leveraging the OSEC approach provides a transformative framework by ensuring that feasible and effective CAs that eliminate RCs can be generated with minimal human expertise.

ACKNOWLEDGEMENT

This study was supported by the UK EPSRC project EP/V062158/1: “Made Smarter Innovation - Research Centre for Smart, Collaborative Industrial Robotics” and the WMG-IIT scholarship.

REFERENCES

- [1] Franciosa P, Sokolov M, Sinha S, Sun T, Ceglarek D. Deep learning enhanced digital twin for Closed-Loop In-Process quality improvement. *CIRP Ann* 2020;69:369–72. <https://doi.org/10.1016/j.cirp.2020.04.110>.
- [2] De Silva D, Sierla S, Alahakoon D, Osipov E, Yu X, Vyatkin V. Toward Intelligent Industrial Informatics: A Review of Current Developments and Future Directions of Artificial Intelligence in Industrial Applications. *IEEE Ind Electron Mag* 2020;14:57–72. <https://doi.org/10.1109/MIE.2019.2952165>.
- [3] Babu M, Franciosa P, Ceglarek D. Spatio-Temporal Adaptive Sampling for effective coverage measurement planning during quality inspection of free form surfaces using robotic 3D optical scanner. *J Manuf Syst* 2019;53:93–108. <https://doi.org/10.1016/j.jmsy.2019.08.003>.
- [4] Shahi VJ, Masoumi A, Franciosa P, Ceglarek D. A quality-driven assembly sequence planning and line configuration selection for non-ideal compliant structures assemblies. *Int J Adv Manuf Technol* 2020;106:15–30. <https://doi.org/10.1007/s00170-019-04294-w>.
- [5] Shi J, Zhou S. Quality control and improvement for multistage systems: A survey. *IIE Trans* 2009;41:744–53. <https://doi.org/10.1080/07408170902966344>.
- [6] Liu Y, Sun R, Jin S. A survey on data-driven process monitoring and diagnostic methods for variation reduction in multi-station assembly systems. *Assem Autom* 2019;39:727–39. <https://doi.org/10.1108/AA-10-2018-0174>.
- [7] Sinha S, Franciosa P, Ceglarek D. Object Shape Error Response using Bayesian 3D Convolutional Neural Networks for Assembly Systems with Compliant Parts. *IEEE Trans Ind Informatics* 2021;17:6676–86. <https://doi.org/10.1109/TII.2020.3043226>.
- [8] Ding Y, Ceglarek D, Shi J. Fault Diagnosis of Multistage Manufacturing Processes by Using State Space Approach. *J Manuf Sci Eng* 2002;124:313. <https://doi.org/10.1115/1.1445155>.
- [9] Sinha S, Glorieux E, Franciosa P, Ceglarek D. 3D convolutional Neural networks to estimate assembly process parameters using 3D point-clouds. *Proc. SPIE*, vol. 11059, 2019. <https://doi.org/10.1117/12.2526062>.
- [10] Huang W, Lin J, Kong Z, Ceglarek D. Stream-of-variation (SOVA) modeling - Part II: A generic 3D variation model for rigid body assembly in multi-station assembly processes. *J Manuf Sci Eng Trans ASME* 2007;129:832–42. <https://doi.org/10.1115/1.2738953>.
- [11] Sinha S, Franciosa P, Ceglarek D. Root Cause Analysis of Multi-Station Assembly Systems with Non-Ideal Compliant Parts using Bayesian 3D U-Nets. *IEEE Trans Autom Sci Eng* 2021.
- [12] Sinha S, Franciosa P, Ceglarek D. Building a Scalable and Interpretable Bayesian Deep Learning Framework for Quality Control of Free Form Surfaces. *IEEE Access* 2021;9:50188–208. <https://doi.org/10.1109/ACCESS.2021.3068867>.
- [13] Mannar K, Ceglarek D, Niu F, Abifaraj B. Fault region localization: Product and process improvement based on field performance and manufacturing measurements. *IEEE Trans Autom Sci Eng* 2006;3:423–39. <https://doi.org/10.1109/TASE.2006.880526>.
- [14] Mannar K, Ceglarek D. Functional capability space and optimum process adjustments for manufacturing processes with in-specs failure. *IIE Trans (Institute Ind Eng)* 2010;42:95–106. <https://doi.org/10.1080/07408170902789027>.
- [15] Box GEP, Luceño A, Paniagua-Quñones M del C. Statistical control by monitoring and adjustment 2009:333.
- [16] Tsung F, Shi J, Wu CFJ. Joint Monitoring of PID-Controlled Processes. <https://doi.org/10.1080/00224065.1999.11979926>.
- [17] Krüger J, Wang L, Verl A, Bauermhansl T, Carpanzano E, Makris S, et al. Innovative control of assembly systems and lines. *CIRP Ann* 2017;66:707–30. <https://doi.org/10.1016/j.cirp.2017.05.010>.
- [18] Metzger M, Polaków G. A survey on applications of agent technology in industrial process control. *IEEE Trans Ind Informatics* 2011;7:570–81. <https://doi.org/10.1109/TII.2011.2166781>.
- [19] Wells LJ, Camelio JA. A bio-inspired approach for self-correcting compliant assembly systems. *J Manuf Syst* 2013;32:464–72. <https://doi.org/10.1016/j.jmsy.2013.03.002>.
- [20] van den Berg R, Lefeber E, Rooda K. Modeling and control of a manufacturing flow line using partial differential equations. *IEEE Trans Control Syst Technol* 2008;16:130–6. <https://doi.org/10.1109/TCST.2007.903085>.
- [21] Mourtzis D, Vlachou E. A cloud-based cyber-physical system for adaptive shop-floor scheduling and condition-based maintenance. *J Manuf Syst* 2018;47:179–98. <https://doi.org/10.1016/j.jmsy.2018.05.008>.
- [22] Wang Y, Yue X, Tuo R, Hunt JH, Shi J. Effective Model Calibration via Sensible Variable Identification and Adjustment, with Application to Composite Fuselage Simulation. *Ann Appl Stat* 2019;14:1759–76.
- [23] Antony J. Simultaneous optimisation of multiple quality characteristics in manufacturing processes using Taguchi’s quality loss function. *Int J Adv Manuf Technol* 2001;17:134–8. <https://doi.org/10.1007/s001700170201>.
- [24] Susto GA, Schirru A, Pampuri S, McLoone S, Beghi A. Machine learning for predictive maintenance: A multiple classifier approach. *IEEE Trans Ind Informatics* 2015;11:812–20. <https://doi.org/10.1109/TII.2014.2349359>.
- [25] Muih V, Kavukcuoglu K, Silver D, Rusu AA, Veness J, Bellemare MG, et al. Human-level control through deep reinforcement learning. *Nature* 2015;518:529–33. <https://doi.org/10.1038/nature14236>.
- [26] Bortolini M, Galizia FG, Mora C. Reconfigurable manufacturing systems: Literature review and research trend. *J Manuf Syst* 2018;49:93–106. <https://doi.org/10.1016/j.jmsy.2018.09.005>.
- [27] Franciosa P, Palit A, Gerbino S, Ceglarek D. A novel hybrid shell element formulation (QUAD+ and TRIA+): A benchmarking and comparative study. *Finite Elem Anal Des* 2019;166:103319. <https://doi.org/10.1016/j.finel.2019.103319>.

- [28] Lillicrap TP, Hunt JJ, Pritzel A, Heess N, Erez T, Tassa Y, et al. Continuous control with deep reinforcement learning. 4th ICLR, International Conference on Learning Representations, ICLR; 2016.
- [29] Kingma DP, Ba JL. Adam: A method for stochastic optimization. 3rd Int. Conf. Learn. Represent. ICLR 2015 - Conf. Track Proc., International Conference on Learning Representations, ICLR; 2015.
- [30] Uhlenbeck GE, Ornstein LS. On the theory of the Brownian motion. Phys Rev 1930;36:823-41. <https://doi.org/10.1103/PhysRev.36.823>.
- [31] Sinha S, Franciosa P, Ceglarek D. Bayesian Deep Learning for Manufacturing 2020:[Online]. https://github.com/sumitsinha/Deep_Learning_for_Manufacturing.

Appendix G: Industrial Demonstrator

Industrial Demonstrator – In-Process Quality Improvement (IPQI) EPSRC Project

S. Sinha, E. Glorieux, M. Babu, P. Franciosa, and D. Ceglarek, "Demonstrator: Inline Quality Monitoring with Root Cause Diagnosis," 2020. [Online]: Available. https://sumitsinha.github.io/Deep_Learning_for_Manufacturing/html/real_system_implementation.html

"In-Process Quality Improvement (IPQI) Project" 2020. [Online]: Available. https://warwick.ac.uk/fac/sci/wmg/research/materials/dlm/projects/ipqi_new/

DEMONSTRATOR: Inline Quality Monitoring with Root-Cause Diagnosis

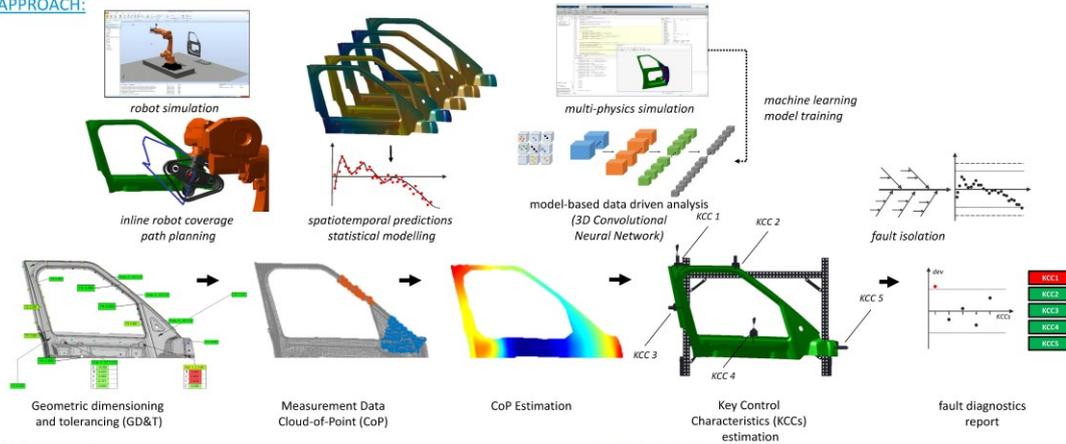
INTRODUCTION:

Achieving resilient performance in terms of quality improvement within production systems is nowadays a key goal for manufacturing industry to be able to predict behaviour of process machinery, to optimise existing equipment for increased quality or to introduce new equipment into an existing assembly line with minimum disruption. Within this context, there is a high need for inline quality monitoring capabilities that allow to rapidly diagnose faults. On top of that, data alone is often insufficient to reveal underlying interdependent relationships between system configuration, process faults and quality defects. It is therefore necessary to enhance data-analysis using multi-disciplinary simulation.

OBJECTIVES:

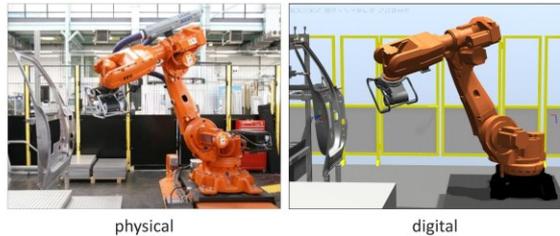
The objectives for this demonstrator are to showcase inline quality monitoring with root cause diagnosis for a case study of a sheet metal assembly. The incorporated digital methodologies use multi-physics simulation combined with data analytics, statistical modelling, optimisation and machine learning to minimise inspection cycle-time while maintaining fault diagnosis capabilities. This capability is a crucial element for systematic quality improvement within manufacturing systems found for example in automotive or aerospace industry.

APPROACH:



DIGITAL TWIN:

This demonstrator, utilising Industry 4.0 technologies, showcases a precise **digital twin** for inline quality monitoring with root cause diagnosis to achieve **resilience performance** in manufacturing systems. It incorporates novel digital methods combining **multi-physics simulation** with **data analytics** are deployed to systematically monitor and improve product quality.



physical

digital

METHODOLOGY:

The overall approach for fault diagnosis includes four different steps, therefore the following four sets of methodologies have been developed:

- 1. Inline robot coverage path planning** – self-programming robot for positioning metrology gauge to optimised viewpoints for inspecting targeted areas on workpiece
- 2. Spatiotemporal predictions** – statistical modelling to predict the entire component deviation pattern from nominal from partial measurement data in real-time
- 3. Model-based data driven analysis** – 3D Convolutional Neural Networks analysing cloud-of-point data for process parameter estimation trained on multi-physics simulation data
- 4. Fault isolation** – identifying the key control characteristic (KCC) that lies at the root cause of the detected fault

These are integrated following a specialised architecture in order to work in tandem as well as to be connected. The system hardware is equipped with state-of-the-art software for cloud-based data-storage, machine learning, metrology configuration and robot simulation.

OUTCOME:

(1) adaptive robotic inline quality inspection, (2) partial measurements with spatiotemporal predictions, (3) extract functional information from high volume measurement data, (4) real-time isolation of process faults

IMPACT:

utilising dimensional measurement systems inline for dimensional quality to optimally control quality in sheet metal assembly processes. This demonstrator show how these technologies can help to eliminate, reduce and correct defects rapidly. This will lead to increased productivity and product quality.

Digital Lifecycle Management (DLM), WMG, University of Warwick

Contact

Professor Darek Ceglarek
D.J.Ceglarek@warwick.ac.uk