3D Computer Vision: Passive Depth from Defocus and Manifold Learning-based Human Activity Recognition

by

Ang Li

Thesis

Submitted to the University of Warwick

for the degree of

Doctor of Philosophy

School of Engineering

August 2014

THE UNIVERSITY OF WARWICK

Contents

List of Tables

List of Figures

Acknowledgments

I would like to thank Dr Richard Staunton for supervising me on depth from defocus. During his supervision, I have read, learnt and implemented a number of algorithms, from which I was able to obtain ideas on what to do and how in order to fulfil my PhD requirements. Also, thank you Dr Staunton for treating me to a meal in your house in Christmas 2011, which is the first time I celebrated Christmas with English people. Most importantly, thank you for helping with my bursary application, so that I no longer needed to work extra hour to obtain my living expenses.

I would also like to thank Dr Tardi Tjahjadi for continuing supervising me to complete a large amount of work after Dr Staunton retired. Dr Tjahjadi is a very studious and hard-working person, who arrives in the School at 7.30 am every morning and leaves at 6.00 pm. He often works extra hours in the weekends to revise my manuscripts. His conscientious personality also helped me correct numerous errors in my articles and encourage me to be more careful. Thank you very much for helping me to submit three journal articles, two of which has been published.

I would like to thank my lab-mates Liang, Minghe, Sruti, Ting, and Xijian at Image Processing and Expert Systems Laboratory, for providing various kinds of help. I would like to thank my beloved girlfriend, Yi Zou, for giving me constant strength for four years. I want to thank my parents, other family relatives and friends as well. Finally, I would like to express my gratitude to the School of Engineering of Warwick University for providing the financial support throughout my PhD study.

Declarations

I hereby declare that I have produced this thesis without the prohibited assistance of third parties and without making use of aids other than those specified; notions taken over directly or indirectly from other sources have been identified as such. This thesis has not previously been presented in identical or similar form to any other publication unless otherwise specified. The thesis work was conducted from 2011 to 2014 under the supervision of Assoc. Prof. Dr. Richard Staunton and Reader Dr. Tardi Tjahjadi at Image Processing Laboratory, School of Engineering, University of Warwick.

Abstract

The rational operator-based approach to depth from defocus (DfD) using pill-box point spread function (PSF) enables texture-invariant 3-dimensional (3D) surface reconstructions. However, pill-box PSF produces errors when the amount of lens diffraction and aberrations varies. This thesis proposes two DfD methods, one using the Gaussian PSF that addresses the situation when diffraction and aberrations are dominant, and the second based on the generalised Gaussian PSF that deals with any levels of the problem. The accuracy of DfD can be severely reduced by elliptical lens distortion. This thesis also presents two correction methods, correction by distortion cancellation and correction by least squares fit. Each method is followed by a smoothing algorithm to address the low-texture problem of DfD.

Most existing human activity recognition systems pay little attention to an effective way to obtain training silhouettes. This thesis presents an algorithm to obtain silhouettes from any view using 3D data produced by Vicon Nexus. Existing background subtraction algorithms produce moving shadow that has a significant impact on silhouette-based recognition system. Shadow removal methods based on colour and texture fail when the surrounding background has similar colour or texture. This thesis proposes an algorithm based on known position of the sun to remove shadow in outdoor environment, which is able to remove essential part of the shadow to suffice recognition purpose. Unlike most recognition systems that are either speed-variant, temporal-order-variant, inefficient or computational expensive, this thesis presents a near real-time system based on embedded silhouettes. Silhouettes are first embedded with isometric feature mapping, and the transformation is learned by radial basis function. Complex human activities are then learnt with spatial objects created from the patterns of embedded silhouettes.

Publications

1. A. Li, R. C. Staunton and T. Tjahjadi. Rational-operator-based depth-from-defocus approach to scene reconstruction. *JOSA A*, 30(9):1787-1795, 2013.

2. A. Li, R. C. Staunton and T. Tjahjadi. Adaptive deformation correction of depth from defocus for object reconstruction. *JOSA A*, 31(12):2694-2702, 2014.

3. A. Li and T. Tjahjadi, Video-based human activity recognition using embedded silhouettes, Submitted to *Pattern Recognition*, 2014, awaiting review.

Abbreviations

- 1D: 1-dimensional
- 2D: 2-dimensional
- 3D: 3-dimensional
- ANN: artificial neural network
- CCD: charge-coupled device
- CDC: correction by distortion cancellation
- CLSF: correction by least squares fit
- CMM: coordinate measuring machine
- CRF: conditional random field
- CNN: convolutional neural network
- DfD: depth from defocus
- DfF: depth from focus
- DMC: double micrometers controlled system
- DTW: dynamic time warping
- EPL: embedded pattern learning
- FFT: fast Fourier transform
- FR: frequency response

- GGRO: generalised Gaussian rational operator
- GRO: Gaussian rational operator
- GUI: graphical user interface
- HMM: hidden Markov model
- IMU: inertial measurement unit
- LED: light emitting diode
- LiDAR: light detection and ranging
- LLE: locally linear embedding
- LOG: Log of Gaussian
- MAP: maximum a posterior
- MEI: motion energy image
- MFR: magnitude frequency response
- MHI: motion history image
- MRF: Markov random field
- NIR: normalised image ratio
- OTF: optical transfer function
- PCA: principal component analysis
- PMF: post median filtering
- PSF: point spread function
- RBF: radial basis function
- RMSE: root mean square error

- RO: rational operator
- RO-DfD: rational operators-based depth from defocus
- SD: standard deviation
- SMC: single micrometer controlled system
- SVD: singular value decomposition
- SVM: support vector machine

Chapter 1

Introduction

This thesis presents the investigation into two topics that are important in 3-dimensional (3D) computer vision: the 3D surface reconstruction approach of depth from defocus (DfD), and a manifold learning-based method for human activity recognition that uses a 3D technique to obtain its training data. This chapter starts with Section 1.1 by giving an overview of existing 3D surface reconstruction approaches. Section 1.2 introduces DfD, Section 1.3 introduces human activity recognition system, and finally Section 1.4 presents the thesis organisation.

1.1 Overview of Surface Reconstruction Approaches

As shown in Fig. 1.1, surface reconstruction techniques can be roughly divided into contact-based and non-contact-based [1]. Non-contact techniques are further categorised into active and passive techniques.

1.1.1 Contact-based techniques

Contact-based techniques require physical contact with the target by a solid object, such as a touch probe. An example of such a system is the coordinate measuring machine (CMM), where a touch probe is used to find and calculate the distance of each point of a target object. Fig. 1.2 illustrates a CMM. During measurement, component 7 is first moved along the y direction. Component 6 (the touch probe) is also moved in both x and z directions. Its location in terms of x, y and z are recorded as it is making contact with the object numbered as 12.

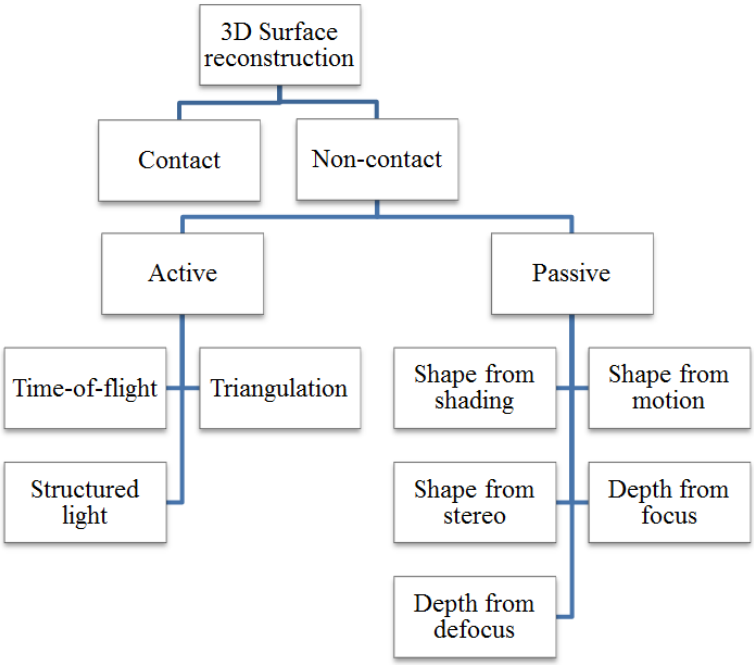


Figure 1.1: 3D surface reconstruction techniques.

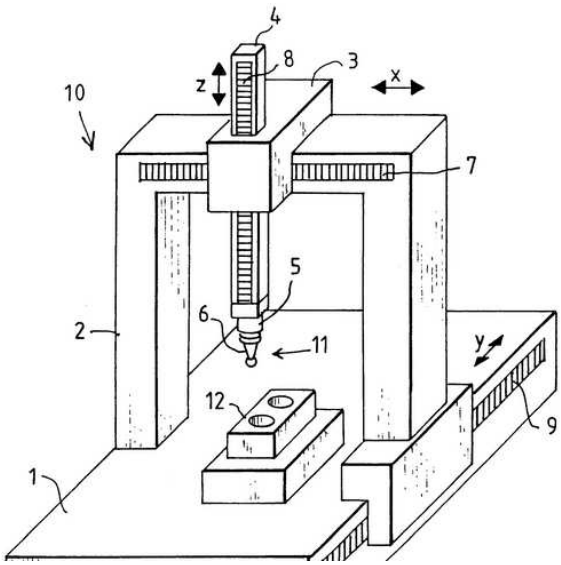
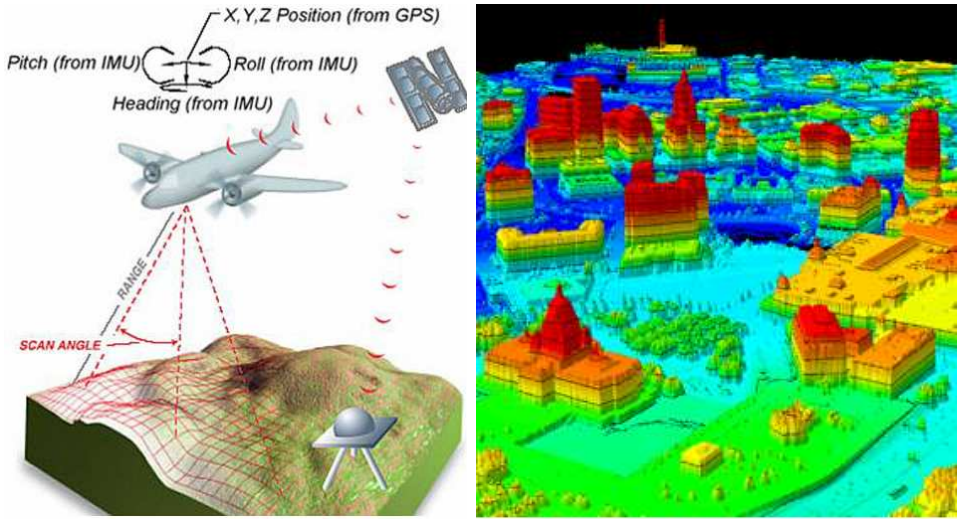


Figure 1.2: CMM adapted from [2].

Non-contact techniques use electromagnetic waves or acoustic waves instead of a solid object. Electromagnetic waves include Gamma-ray, X-ray, visible light and infrared. Acoustic waves include ultrasound. Gamma-ray, X-ray and ultrasound have short wave-length and can easily penetrate object, they are thus primarily used for inspecting interior structure of an object [3]. Visible light and infrared have median wavelength and cannot normally penetrate object, they are thus used to reconstruct the outer surface.



(a) (b)

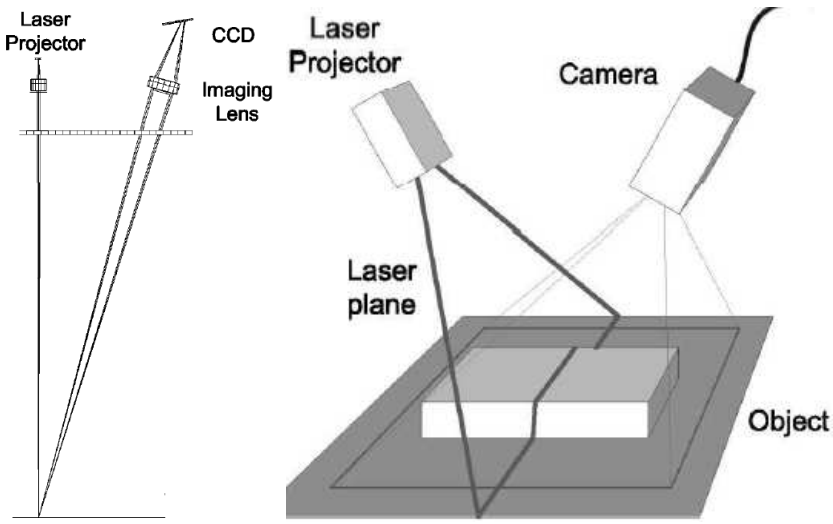
Figure 1.3: LiDAR-based surface reconstruction: (a) Schematic diagram (adapted from [4]) and (b) reconstruction of Rhode Island (adapted from [5]).

1.1.2 Active techniques

Active techniques require the operator or the instrument to provide a source of waves that is used for distant measurement. Popular examples include time-of-flight, triangulation and structured light. In time-of-flight-based techniques, a pulse of laser is emitted onto the target point, reflected by the target and received by a sensor. The time interval for the process is used to calculate the distance. Time-of-flight is capable of operating over very long range up to 800 metres with 1 centimetre accuracy [6]. They are thus mainly used for scanning buildings and geographic features [4]. Light detection and ranging (LiDAR) is a typical example of these techniques. Fig. 1.3(a) illustrates the working principle of an airborne LiDAR carried by an aircraft. The LiDAR constantly scans left and right downwards as the aircraft proceeds. The system obtains the x,y,z position with GPS, and the motion information from the inertial measurement unit (IMU) to obtain the 3D terrain surface recon-

struction. Fig. 1.3(b) shows a pseudo-coloured surface reconstruction of the Rhode Island in spring 2011.

In triangulation-based techniques, the light source, the camera and the object are placed in three different locations, forming a triangle. When the light source is fixed, the position of the reflected light in the image indicates how far the object is from the camera. Triangulation-based techniques provide very high accuracy up to 2.25 micrometres [6]. Applications include inspecting surface defects and precision measurement. Fig. 1.4 illustrates two common types of triangulation techniques, i.e., the single point laser scanner and the slit laser scanner. The single point scanner shown in (a) works by projecting a single ray of laser onto the object with known orientation. A camera with known configuration and orientation captures the image of the reflected ray as a sharp point. Finally the depth is computed using the location of the sharp point in the image. This process is repeated many times at different locations of the target object so that a depth map is obtained. Instead of projecting a single ray, the slit scanner in (b) projects a slit of rays onto the object, which considerably increase the reconstruction speed.



(a) (b)

Figure 1.4: An illustration of the triangulation laser scanner (adapted from [6]): (a) Single point and (b) slit laser scanner.

In structured-light approach, a pattern of light is projected onto the object, and the reflected pattern indicates the distance [7]. Fig. 1.5 illustrates an example of surface reconstruction using a structured-light technique. Patterns of parallel lines are projected onto the object as shown on the left. Regions with lines having larger separation correspond to greater depth and vice versa. The reconstruction result

Figure 1.5: Structured-light-based 3D surface reconstruction (adapted from [10]).

is shown on the right by 3D surface rendering. Structured light approach provides lower accuracy than triangulation, but it is capable of producing smooth surface reconstruction at video rate [6]. Hence, it is ideal for human body measurements where absolute accuracy is not important. Real-time system has been developed, e.g., Microsoft Kinect sensor which uses infrared as the light source [8]. Stereoscopic systems with structured-light are also commercially available. Two cameras are used in such a system, the locations of the projected pattern in their field of view are used to calculate distance [9].

1.1.3 Passive techniques

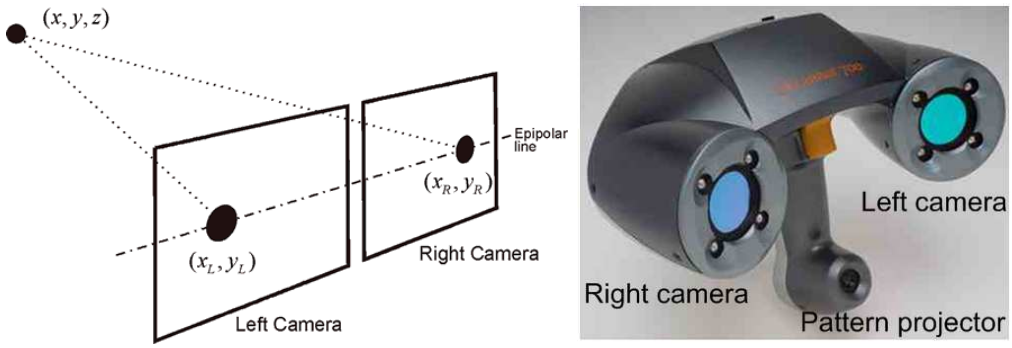
Passive surface reconstruction techniques do not require any active light projection. Instead, the natural radiance, colour and textures of the target surface are used for distance measurement. Shape from shading and shape from motion are famous examples. Depth from stereo, depth from focus (DfF) and DfD are also passive if no active projection is performed. Shape from shading is based on the fact that the surface shape affects the reflectance property [11] as illustrated in Fig. 1.6. The reflection distribution, viewing direction and the light source are used as input. High-quality lighting is required for accurate result. Shape from shading will not work for a random scene where no information is obtained for the light source.

Figure 1.6: Artist's drawings of 3D shapes (adapted from [12]).

In shape from motion, either the camera or the target is moved while images are captured. Every set of corresponding points in these images are found. Their locations in the images and the movement of the camera/target are then used to

estimate the 3D information. Fig. 1.7 illustrates an implementation of shape from motion on a smart phone, where the orientation and displacement of the camera are obtained with the internal sensors of the camera. Depth from stereo is a similar technique, where two images are captured by two cameras at different views. Fig. 1.8(a) illustrates the working principle of a stereoscopic system. Two cameras with known orientation and baseline capture images of an object. The images of the object points from the two cameras are in different locations. The difference in locations, known as disparity, is larger if the object is closer to the camera. Thus, disparity is computed to infer depth.

Figure 1.7: An implementation of shape from motion on a smart phone (adapted from [13]).



(a) (b)

Figure 1.8: Depth from stereo: (a) working principle (adapted from [14]) and (b) a commercially available active stereoscopic system (adapted from [15]).

The biggest problem with shape from motion or depth from stereo is finding point correspondences, i.e. which two pixels correspond to the same object point, and this requires large computational cost and is not always possible. Correspondence cannot be found if a point only appears in one of the images but is occluded in the other. If correspondence cannot be found, the depth cannot be recovered.

Fig. 1.8(b) shows an example of an active stereoscopic system, where a projector projects a pattern of light rays onto the object to aid the correspondence search.

Passive DfF computes distance by analysing camera parameters after the image is in focus. When the image is in focus, the focal length, and sensor to lens distance are used to calculate depth. The challenge of DfF is in deciding when the object is in focus. Auto focus has become commercially available, but its operational speed is limited to up to few frames per second. Thus, a dense depth map using DfF is very unlikely to be achieved in real-time.

1.2 Depth from Defocus

In this thesis, depth is the distance from a target object to the viewer, and defocus is the artefact of image blurring. Thus, DfD is a 3D surface reconstruction approach based on image blurring effect. DfD can take either a single image, two images, or more than two images as its input. Techniques using a single image analyses the frequency content near sharp edges. However, they have an inherent difficulty in distinguishing whether low frequency regions correspond to blurred sharp edges or focused smooth surfaces. Thus they are primarily used when only one input image is available. Most existing DfD methods use two images as input that effectively addresses the above-mentioned ambiguity problem. More images can be used which impose over-constraints, thus improving the reconstruction accuracy.

This thesis is concerned with DfD using two images. Fig. 1.9 illustrates the workflow of the DfD approach. First, two images of a scene are captured by a digital camera with different but known focus settings, where focus settings or optical settings refer to aperture size, sensor to lens distance and focal length. In the top image, the background sandpaper is the furthest object in the scene, and it is in focus. Thus, this image is called the far-focused image. In the bottom image, the nearest bottom wooden chunk is in focus, this image is thus called the near-focused image.

Far/near-focused image pair is often used as the input to DfD methods, but it is not the only option. Other means of obtaining the image pair include changing aperture size and focal length. The single-channel (greyscale) image pair obtained by averaging the multi-channel (colour) image pair is used in this thesis as the cases in [16; 17]. A DfD method estimates the depth, pixel by pixel, and generates a result that can be presented as a depth map or a mesh plot. In Fig. 1.9 a radiance coded mesh-plot is produced as seen on the right. Please refer to Appendix B for the difference between depth map, mesh plot, 3D surface and volumetric rendering.

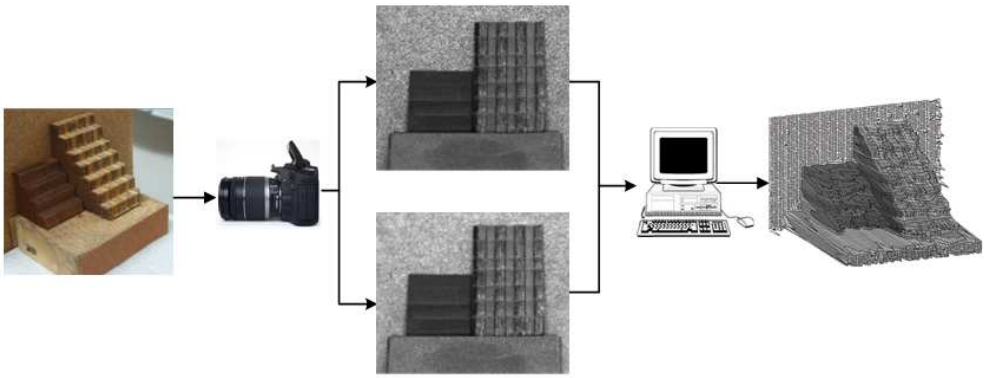


Figure 1.9: Overview of DfD using 2 images.

1.2.1 Mathematical background

Fig. 1.10 shows an image of flowers. It is clear that the flowers which are closer to the camera are sharper than those further away, or the closer flowers are less blurred than the further ones. This is the basis for DfD, where the amount of blur infers distance. The blurring effect is modelled mathematically as the convolution of the point spread function (PSF) with a focused image, i.e.

$$\mathbf{I} = \mathbf{H} * \mathbf{M} \ , \tag{1.1}$$

where $\mathbf{M}$ is the focused image, $\mathbf{H}$ is the PSF and $\mathbf{I}$ is the blurred image. The PSF is associated with a parameter indicating the amount of blurring, which is the defocus parameter. For example, the Gaussian PSF is given by [18]

$$\mathbf{H}(x, y) = \frac{1}{2\pi\sigma^2} e^{-\frac{x^2+y^2}{2\sigma^2}} \ , \tag{1.2}$$

where σ is the standard deviation (SD) or the defocus parameter of the PSF, x and y are the horizontal and vertical indices respectively. The larger the SD, the more blurred the resulting image becomes. Note also that circular symmetry applies for all PSFs assuming the lens has a non-distorted circular symmetrical shape.

Figure 1.10: Blur and depth [19].

Please also note the mathematical notations used in the thesis: v (lower-case italic letter) denotes a scalar variable, V (upper-case letter) denotes a scalar constant, $\vec{v}$ (arrow overhead) denotes a 1-dimensional (1D) array of elements, $\mathbf{V}$ (bold upper-case) denotes a 2-dimensional (2D) array of elements, matrix or digital image, and $\check{\mathbf{V}}$ (check overhead) denotes the magnitude frequency response (MFR) of $\mathbf{V}$.

Fig. 1.11 shows an example of modelling blur with the Gaussian PSF. The first row shows an all-focused image where every pixel in the image is in focus. The second row shows Gaussian PSFs with SD of 1, 1.3 and 2.5, respectively from left to right. The third row shows the resulting blurred images. As the figure shows, the blurring effect is higher for large value of SD. Hence, when Gaussian PSF is used, the SD is estimated to obtain depth.

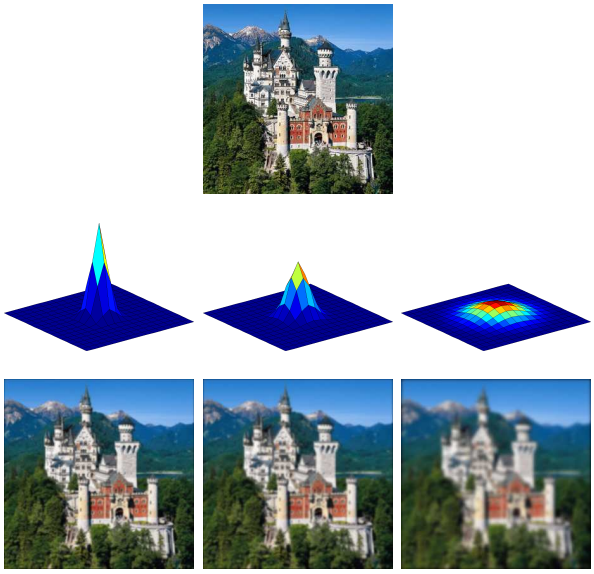


Figure 1.11: An example of blur modelling with Gaussian PSF. Row1: the original image (adapted from [20]). Row 2: surface plot of the PSF with different SD. Row 3: the blurred images.

There are numerous DfD methods. The generic DfD method, i.e., the Subbarao's method [18] is based on estimating depth from the amount of defocus, which is represented by the defocus parameter of PSF. The defocus parameter is computed by solving a system of equations involving input pixel values and optical parameters such as focal length, aperture diameter and sensor-to-lens distance. An outline description of the generic DfD method, i.e., the Subbarao's method, is presented in Appendix C. Although the generic DfD offers a general clue for estimating depth with PSF modelling, the result is very noisy and inaccurate due to lack of filtering

and failure to consider the frequency dependency problem, which states that different frequency components have different depth responses. In [16], a number of spatial filters, called rational filters or rational operators (ROs) were designed to address this problem.

Fig. 1.12 illustrates the DfD image acquisition system with telecentric optics[1]. The light rays from an object point pass through the telecentric aperture of radius a. The focused image point is tagged at a position between the far-focused position l_1 and the near-focused image position l_2. The normalised depth α is -1 at l_1 and 1 at l_2. The distance between the focused point and l_1 is $(1+\alpha)e$ and that between the focused point and l_2 is $(1-\alpha)e$. u is the distance between an object point and the lens and F is the focal length. First, the f-number of the lens

$$F_e = \frac{F}{2a} \ . \tag{1.3}$$

The optical transfer function (OTF) (please see Appendix C for its definition) of the frequency-dependant pill-box

$$\check{\mathbf{H}}(f_r, \alpha) = \frac{2F_e}{\pi(1+\alpha)ef_r} J_1\left(\frac{\pi(1+\alpha)e}{F_e} f_r\right) \ , \tag{1.4}$$

where, J_1 is the first-order Bessel function of the first kind, and f_r denotes the radial frequency.

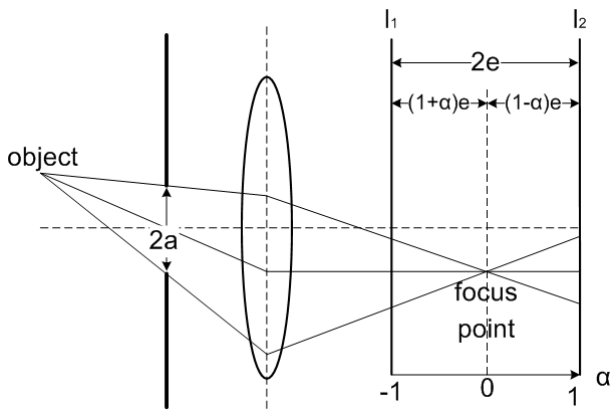


Figure 1.12: The telecentric DfD system.

[1]Telecentric optics involves placing an external aperture on a lens in order to eliminate image magnification when obtaining the DfD input images. See Appendix D for a more detailed description.

The normalised image ratio (NIR) or the M/P ratio is given by:

$$\frac{\check{\mathbf{M}}}{\check{\mathbf{P}}}(f_r, \alpha) = \frac{\check{\mathbf{H}}_1(f_r, \alpha) - \check{\mathbf{H}}_2(f_r, \alpha)}{\check{\mathbf{H}}_1(f_r, \alpha) + \check{\mathbf{H}}_2(f_r, \alpha)} \ , \tag{1.5}$$

where $\check{\mathbf{H}}_1$ and $\check{\mathbf{H}}_2$ are the OTFs of the far-focused and near-focused image, respectively. Eqn. (1.5) was modelled as a third order polynomial of the depth α[16], i.e.,

$$\frac{\check{\mathbf{M}}}{\check{\mathbf{P}}}(f_r, \alpha) = \frac{\check{\mathbf{G}}_{p1}(f_r)}{\check{\mathbf{G}}_{m1}(f_r)}\alpha + \frac{\check{\mathbf{G}}_{p2}(f_r)}{\check{\mathbf{G}}_{m1}(f_r)}\alpha^3 \ , \tag{1.6}$$

where the coefficients are expressed as rational forms; $\check{\mathbf{G}}_{m1}$, $\check{\mathbf{G}}_{p1}$ and $\check{\mathbf{G}}_{p2}$ are the Fourier transform of the ROs.

For offline preparation, Eqn. (1.4) and (1.5) were first used to obtain the NIR. least squares fit was then used to find the first and third order coefficient in Eqn. (1.6). After $\check{\mathbf{G}}_{m1}$ was initialised to be a band-pass filter, the other two filters $\check{\mathbf{G}}_{p1}$ and $\check{\mathbf{G}}_{p2}$ were computed with the coefficients. The corresponding spatial filters are denoted by $\mathbf{G}_{m1}$, $\mathbf{G}_{p1}$ and $\mathbf{G}_{p2}$, respectively, which are the rational filters or ROs.

During run-time, the difference image $\mathbf{M}$ and sum image $\mathbf{P}$ were computed first using

$$\mathbf{M} = (\mathbf{I}_2 - \mathbf{I}_1) * \mathbf{Q} \tag{1.7}$$

$$\mathbf{P} = (\mathbf{I}_2 + \mathbf{I}_1) * \mathbf{Q} \ , \tag{1.8}$$

where $\mathbf{Q}$ is a pre-filter that removes the frequency components that degrade surface reconstruction. Denoting $\mathbf{A}$ as the depth map and using Eqn. (1.6),

$$\mathbf{G}_{m1} * \mathbf{M} = (\mathbf{G}_{p1} * \mathbf{P}) \cdot \mathbf{A} + (\mathbf{G}_{p2} * \mathbf{P}) \cdot \mathbf{A}^3 \ , \tag{1.9}$$

where $\cdot$ denotes the dot product. Solving Eqn. (1.9) gives the depth map.

This DfD method performs detailed analysis on every frequency component from the input images by obtaining the frequency-variant expression of the OTF and the NIR using Eqn. (1.4) - (1.6). The run-time computation is linear with only 5 convolutions as shown using Eqn. (1.7) - (1.9). Thus, this method is both accurate and of low computational cost, and it is ideal for real-time 3D surface reconstruction for human activity analysis. In [16], the ROs were designed with a non-linear optimisation technique. A simplified algorithm to design these operators was presented in [17].

1.2.2 Motivation

The interest in DfD is motivated by its advantages over other 3D reconstruction techniques and its potential applications. Contact-based techniques provide micrometre precision. However, they are very slow with speed of up to few hundred object points per second. They are also not suitable for delicate object which might be damaged or modified. Their applications include measurement of the surface of a flat object, and objects with simple curvature in manufacturing industry. Active techniques provide high accuracy and higher speed than contact-based ones. DfD with structured-light is studied by researchers, where the amount of blurring of the projected pattern is used to calculate depth. However, active techniques require dedicated hardware, professional calibration and are usually very expensive.

Passive techniques provide considerably lower accuracy than contact-based active techniques, and thus cannot be used for high precision measurement. However, their implementations are much faster and cheaper since active illumination is not required. Amongst these techniques, DfD is capable of producing dense depth map with high speed, does not have the correspondence problem associated with depth from stereo since only one view is used, and it can adapt to various lighting conditions by changing sensor exposure time or aperture diameters. Recently DfD was applied in Panasonic Lumix GH4 digital camera for rapid auto-focus [21], which was significantly faster than traditional DfF-based technique. In contrast with depth from stereo that requires two lenses, DfD requires only one lens to capture input images, thus it is ideal for applications where the input device needs to be miniaturised, such as 3D endoscopy. Other potential applications include 3D-modelling, human-computer interaction and human activity analysis where active illumination hardware is not available or practical.

1.2.3 Challenges and contributions

As discussed in Section 1.2.1, the DfD method in [16] is accurate and fast. However, it suffers the following serious drawbacks that must be addressed. First, it assumes no aberrations or diffraction, and thus uses a pill-box PSF. However, this is not true for most of the off-the-shelf lenses. Second, the designed pre-filter fails to remove significant amount of frequency components which lead to suboptimal reconstruction. Third, spurious assumptions are made in the unnecessarily complex design procedure for the ROs. Finally, an elliptical depth distortion resulting from optical lens is not addressed which leads to a distorted depth map.

To address these drawbacks, we propose a DfD system with Gaussian ROs and generalised Gaussian ROs. The Gaussian ROs are designed with Gaussian

PSF for the imaging environment when aberrations and diffraction are significant, and the generalised Gaussian ROs with generalised Gaussian PSF[2] is applied to the environment with any amount of aberrations and diffraction. Frequency components containing low gradient along with the zero gradient are removed with the proposed pre-filter. Only one cost function is formed during filter optimisation without the need of any assumption. The elliptical depth distortion is addressed by the proposed two correction methods: correction by distortion cancellation (CDC) that works by cancelling the distortion with a known similar distortion, and correction by least squares fit (CLSF) where a mapping from the distorted value to the corrected value at a given location and distance is learnt efficiently by least squares fit.

1.3 Human Activity Recognition

Thus far, a number of common 3D surface reconstruction techniques has been briefly introduced in Section 1.1, and an introduction on the first topic of our thesis, i.e. DfD, has been given in Section 1.2 along with its advantages over the alternative techniques and its major problems. In this section, an introduction of human activity recognition system that is the second topic of the thesis is provided. The proposed system makes uses of a 3D reconstruction technique to obtain the training data.

Video-based human activity recognition is one of the current most important topic of computer vision research. In recent years, it has attracted the interest of many researchers from academia, industry, consumer agencies and security agencies. A recognition system aims at the automatic analysis of an activity performed by a person or multiple persons captured in a video or image sequence. In the simplest case where a video sequence is segmented to contain only a person performing one activity, the system is expected to classify this activity as one of the learnt categories.

There is no consensus terminology for an activity and an action [22]. We thus define an action primitive as a sequence of individual postures of a single body part such as rising arm and kicking. We define an action as a periodic movement comprising a number of primitives, such as jumping and hand-clapping. We define an activity as a sequence of individual actions that serves a goal, such as walking while using a mobile phone, and a sequence of digging towards different direction. In this thesis we limit our research to full body of a single human subject using silhouettes. Therefore, recognition of hand gestures and group activities are beyond the scope of this thesis.

Research in video-based activity recognition is preceded by research in ob-

[2]Refer to Appendix E for a description on generalised Gaussian PSF.

ject recognition and speech recognition before digital video hardware became widely available. Thus, it is significantly inspired by both preceding recognition systems. Systems that are inspired by the former consider a video as a spatio-temporal volume, which is a 3D volume created by combining the frames from consecutive time instances, and 2D object feature analysis is extended into the 3D case. Examples include a system that incorporates 3D interest points extraction [23] and a system that uses 3D convolutional neural network (CNN) [24]. Systems inspired by the latter is based on sequential analysis of features extracted from every frame of the video, and they often use techniques that have been successfully implemented in the speech recognition systems such as the hidden Markov model (HMM) [25] and dynamic time warping (DTW) [26].

The spatio-temporal approach to activity recognition is illustrated in Fig. 1.13. This approach involves a training process and a learning process. The training includes: generating a spatio-temporal volume from each of a number of videos containing different classes of activities; and generating an activity model by extracting features from the volume. During learning, the similarity between different activity models is defined using measures such as Euclidean distance, Mahalanobis distance and subspace angles. When a video containing an unknown activity is to be recognised, similar processes as in training are performed. A classification algorithm, e.g., nearest-neighbour and Support Vector Machine (SVM), is then performed to label the video as one of the learnt categories based on the defined distance measure.

The sequential approach is illustrated in Fig. 1.14. Raw data such as silhouettes and body joints are extracted from the video frame by frame. A feature vector is created by extracting features from each frame of raw data, and an activity model is generated from the feature vectors. Finally the learning and classification procedures are similar to those in the spatio-temporal approach.

Human activities contain complex information and are of high dimensionality. Thus, there is a need for dimensionality reduction techniques to retain only the discriminating features of human activities. Examples of popular dimensionality reduction techniques include principal component analysis (PCA) [28], multidimensional scaling [29], locally linear embedding (LLE) [30] and isometric feature mapping (Isomap) [31].

Fig. 1.15(a) illustrates LLE for feature extraction, where high dimensional images of human faces are embedded into a 2D space. The horizontal dimension is related to the amount of relaxation of the facial expression and the vertical dimension is related to the direction of the face. Fig. 1.15(b) illustrates Isomap where a different

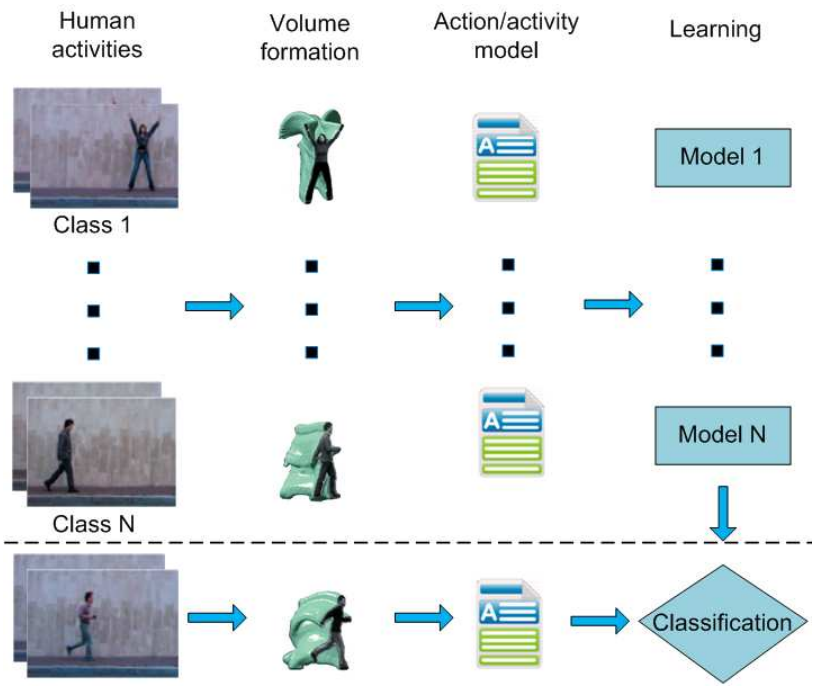


Figure 1.13: Spatio-temporal approach to human activity recognition (adapted from [27]).

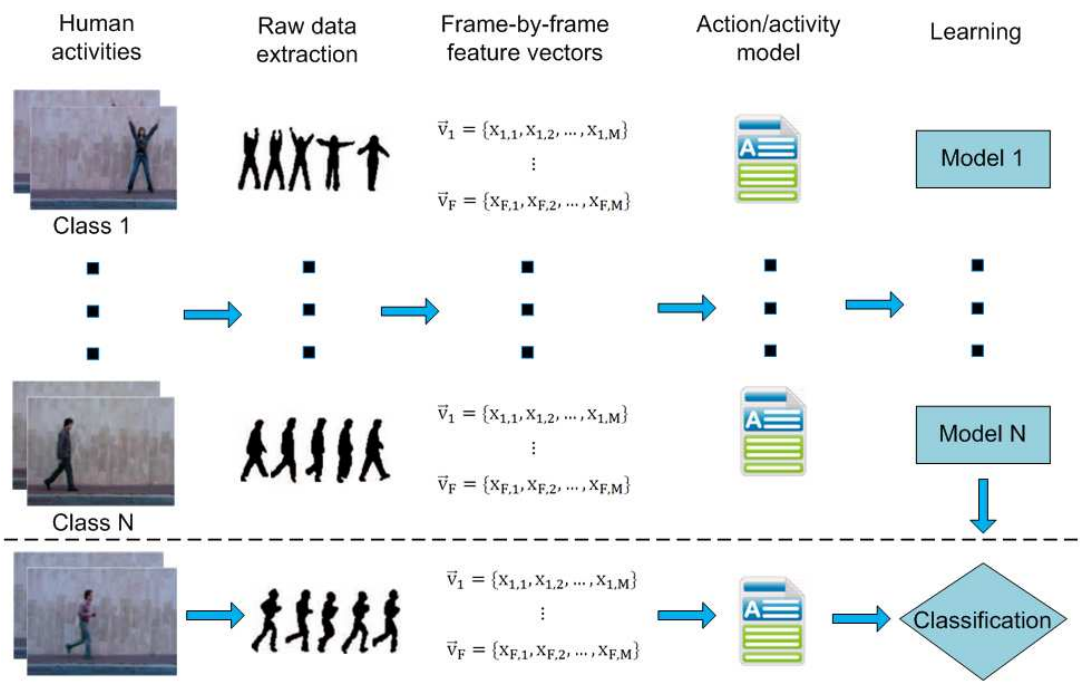
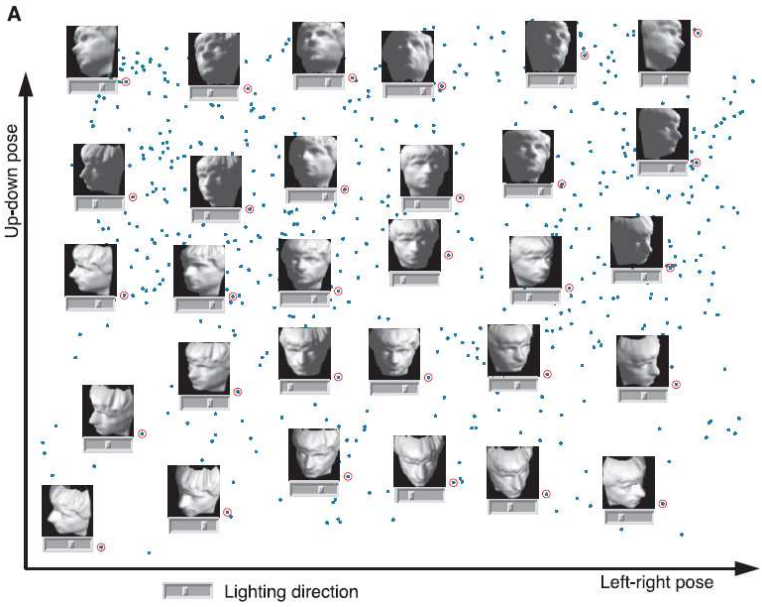


Figure 1.14: Sequential approach to human activity recognition (adapted from [27]).

(a)



(b)

Figure 1.15: An illustration of dimensionality reduction techniques for feature extraction: (a) LLE (adapted from [30]; and (b) Isomap (adapted from [31]).

set of face images are embedded into a 2D space. The horizontal axis represents left or right pose of the head, and the vertical axis represents up and down pose of the head.

During training, a number of feature vectors are embedded and stored. During testing, a new feature vector needs to be embedded as well in order so as to match it with the stored vectors. Manifold learning involves learning the mapping from original space to the embedded space. Previous recognition systems made use of linear methods such as PCA [32], where new input vectors could be embedded by direct projection onto the principal axioms generated by the Singular Value Decomposition (SVD) applied to the training data. However, human activity is highly complex and non-linear. A significant improvement was thus made using non-linear dimensionality reduction methods including Isomap and LLE [33; 34; 35].

1.3.1 Motivation

The interest in human activity recognition is motivated by several important applications, including content-based video analysis, human-machine interaction, patient monitoring, safety monitoring, and security and surveillance [36]. Content-based video analysis aims to categorise a video according to its contents containing humans performing activities. This has become a very important application for video sharing websites, which may have the need to automatically assort or evaluate their videos. This is also very important for sport videos [37], such as one containing a football match, where the match statistics require accurate identifications of passing ball, shooting, and etc.

Human-machine interaction is the mutual action between human and machine, which involves both input (human to machine) and output (machine to human) communication. Traditional input methods include switching switches, turning knobs, pressing buttons and typing with keyboards, and output methods include acoustic and optical ones. However, these methods are often not intuitive and difficult to learn. Recent success in object recognition and speech recognition has made it easier for human users to use machine. For example, fingerprint devices have been widely available which save users from typing password, and speech recognition systems allows users to control machines by simply talking to them. Similarly, activity recognition would encourage users to use machines by performing an action or an activity. However, the required technique for human-machine interaction is not sufficiently mature and thus it is still an active research area.

Hospitals may require constant monitoring of patients in the absence of medical staff, or when patients are recuperating at home. This is particularly useful for

patients with chronic disease who need to take medicine regularly for a long period. This is also important for patients in critical conditions who may not be able to contact medical staff. It is inevitable that continuous monitoring of patients leads to monitoring staff's physical or mental fatigue, thus an automatic system is highly beneficial for monitoring or surveillance application. Highly hazardous environment can cause injury or death if a dangerous activity is performed, e.g., smoking in a petrol station and entering a biomedical laboratory without wearing protective clothing. An automatic system that identifies dangerous human activities can potentially reduce such fatal incidents. Traditional security and surveillance systems require a number of video cameras monitored by a human operator. Due to fatigue caused by repetitive nature of the work, abnormal behaviours often go unnoticed. In addition, with the decrease in the cost of high quality video cameras and the increase in the cost of employing human operators, an automatic system that is both accurate and cost effective has gained a lot of interest from security agencies and other researchers. Another similar application is to automatically identify a target activity from a large video database.

1.3.2 Challenges and contributions

Hitherto, there has been little interest in developing an efficient and accurate means of obtaining 3D training data for activity recognition [38]. For most recognition systems based on silhouettes, the training data is generated with a camera, or multiple cameras if more views are required. This is impractical if a large number of views are required, and errors occur in manual camera placement. DfD is potentially useful for this application, where a 3D human body model can be obtained with only two cameras with one facing another and the subject in between. However, our current DfD system is neither portable nor incorporate real-time automatic input video acquisition. Therefore, we propose a method for generating silhouettes from any view using 3D body coordinates (extracted using Vicon Nexus [39], of each marker placed on an actor while performing an activity) and a triangulation technique for 3D surface reconstruction of the activity. Shadows are inevitably present in real scenes and are difficult to remove reliably from a silhouette [40]. To address this issue we propose a shadow removal method for outdoor scenes based on an estimate of the current position of the sun and 3D analysis of shadow formation.

Various problems are also encountered by an activity recognition system. When the temporal order of the actions comprising an activity is changed, new training data is required for that activity to be recognised, e.g., as in sequential methods that model activities with the HMM [25; 33]. When the speed in which

an activity is performed is changed, the template used for its recognition has to be changed as in the space-time volume approach [41; 42]. Real-time operation with high accuracy and robustness are often necessary. However, methods based on space-time trajectory [43; 44] are very slow because they require accurate 3D modelling of a large number of body parts. They also have problem dealing with occlusion of joints. We address these problems by proposing an embedded pattern learning (EPL) algorithm which uses a spatial object created from the coordinates of embedded silhouettes to denote an activity. The spatial object takes into account the speed variation of the actions and is invariant to their temporal order. The algorithm is fast due to its linear nature. Our main contributions on human activity recognition are: (a) a method for generating training silhouettes from a 3D human model; (b) manifold learning using Isomap [31], where the radial basis function (RBF) learning process is significantly simplified compared to the work in [34]; (c) a reliable shadow removal method; (d) and EPL for recognising activities.

1.4 Thesis organisation

Chapter 2 and Chapter 3 respectively provide literature reviews on a number of most important and well-known passive DfD techniques and human activity recognition methods. These chapters aim to provide the basic working principles and methodologies of different and state of the art methods.

Chapter 4 presents the experimental procedures for acquiring DfD input images, and this includes a description of the designed hardware and software environment. Chapter 5 proposes a RO-based DfD (RO-DfD) algorithm using Gaussian PSF and generalised Gaussian PSF in order to cope with different amount of lens aberrations and diffraction. This work has been published in [45]. Chapter 6 presents two DfD correction methods which address the elliptical distortion problem in DfD, where experiments are performed on seven objects with four existing DfD methods to demonstrate their potential in adapting all other DfD methods. This work has been submitted to a journal [46].

Chapter 7 proposes a silhouette generation technique with 3D data obtained by Vicon Nexus system, which enables efficient acquisition of training silhouettes for any view. Chapter 8 presents a reliable shadow removal technique for outdoor environment using known current position of the sun. As more information is obtained from time, location and camera orientation that are used to estimate the length and angle of the shadow, assumptions on colour and texture are not required, and this increases the robustness of the shadow removal technique. Chapter 9 presents a manifold learning-based algorithm using embedded silhouettes for fast human activ-

ity recognition, which is speed-invariant, temporal-order-variant and efficient. The works in Chapter 7 to Chapter 9 have been submitted as a single article to a journal [47]. Chapter 10 concludes the thesis and discusses possible future work.

Chapter 2

Literature review on Depth from defocus

2.1 Introduction

This chapter is split into two sections. The first section is the review on passive DfD techniques using a single image, which generally compute depth by analysing the frequency contents of the image. The second section reviews the techniques that use two images, which estimate depth maps with more sophisticated approaches resulting in more accurate results. Individual reviews are arranged in time order with respect to the researcher(s)'s first publication date.

2.2 Depth from Defocus using a Single Image

The first idea of DfD was proposed in [48]. It stated that DfD had two major advantages: it provided similar accuracy to stereo vision while not requiring any correspondence search; unlike DfF, the best focus point location was not required. In [49], two DfD methods were presented. One of them used Gaussian PSF and the sharp discontinuities (edges) in a single defocused image to recover depth. First the edges were analysed within every local region by a Laplacian filter. As a result, the SD σ of the PSF was obtained. The final estimate of the depth was given by

$$u = \frac{Fs}{s - F - \sigma F_e} \ , \tag{2.1}$$

where F is the focal length, s is the sensor to lens distance and F_e is the f-number of the lens. To produce a dense depth map with high resolution, each of the input images was divided equally into small local regions with a size of at least 2×2. The

depth value within one region was assumed to be the same and was then computed. This process is repeated for all regions to obtain the depth map. In this thesis, we use the term "local image region", "patch", "window" and "neighbourhood" interchangeably. Experiments with a real image showed that the depths of the region near the sharp edges were recovered. The edges could be categorised into high, medium and low in terms of depth. However, a complete depth map could not be computed with this method.

In [50], a simplified version of the method in [49] was presented. A term called "spread parameter" σ_l was introduced, which indicated the defocus degree of an edge. Hence it is a 1D analogue to the SD σ of the Gaussian PSF. The depth was found with an equation which related σ_l and camera parameters to depth. Experiments were performed with a cardboard paper, the surface of which was drawn with black and white strips. Higher accuracy was achieved for objects closer to the camera than further ones. The working distance was reported to be about 8 feet.

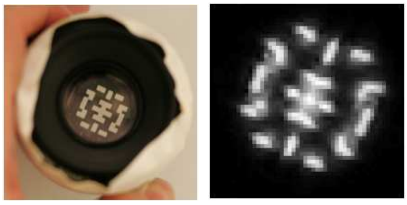
In [51], the method using a single image in [49] was generalised. The edge orientation that was important for [49] was not required. In addition, a new method was developed to estimate the SD of the PSF so that the approach was less sensitive to noise disturbance. An average error rate of 5% was reported using real images. This approach could be applied to both step edges and ramp edges. However, the computational cost was high for ramp edges.

Also based on the work in [49], a DfD method was proposed in [52] using moment preservation and proportion of edge pixels in local regions. The image sensor was pre-adjusted to be behind the nearest focused points, thus only one image was required. A gradient image was then obtained using a Sobel filter. The diameter of the blur circle used to determine depth was found proportional to the edge pixels in a corresponding local region in this gradient image, which was used to calculate depth. The first three moments of the local region were then computed in terms of the proportion of the diameter and the local region. Therefore the proportion could be determined which in turn determines the depth. Furthermore, an artificial neural network (ANN) was used to deal with optical errors. However, the method required a circular local region with a radius of 35 pixels, which would only provide a limited depth resolution.

A DfD method using endoscopic video was proposed in [53]. Traditional DfD methods captured one or two images of the target scene at the same view. In contrast, this method used multiple images from different views with frames from a video. First, the depth map for sharp edges in every frame was estimated using

Laplacian filtering and Sobel filtering. By assuming the location of the camera was known during recording, a complete 3D model was then reconstructed from all the edge depth maps using triangulation.

In [54], a conventional camera was modified by placing a coded aperture in front of the lens (see Fig. 2.1) to achieve better performance. Fig. 2.2 illustrates the working principle using a PSF at 3 different scales and the corresponding frequency domain representation.



(a) (b)

Figure 2.1: Coded aperture illustration adapted from [54]. Left: a coded aperture placed in front of the lens; right: the resulting blur pattern.

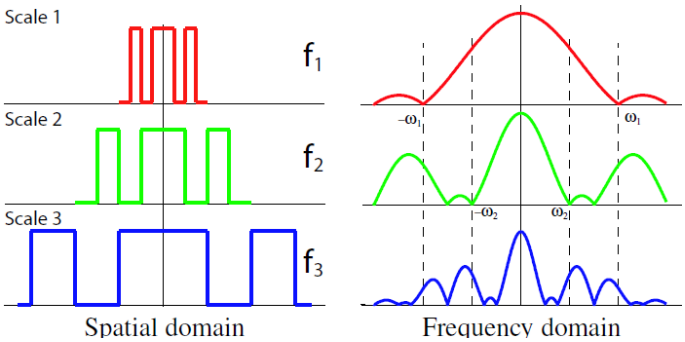
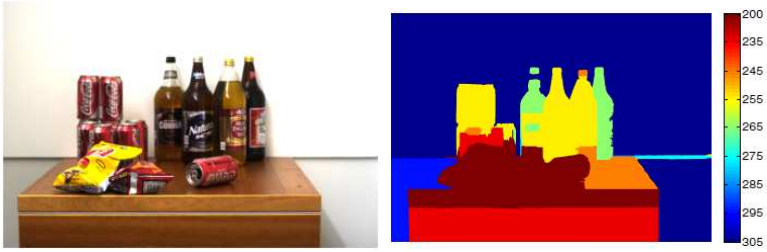


Figure 2.2: PSFs at different scales and their frequency response adapted from [54].

The idea behind the method is to consider the zero-crossing of the frequency. For example, f_1 is less blurred compared to f_2, therefore the zero-crossing ω_1 is larger than ω_2. In other words, the scale in the frequency domain infers depth. The probability distributions of the focused image and blurred image at different scales were formulated with the zero-crossing idea. For aperture design, the Kullback-Leibler divergence was maximised for the distributions of the PSF at different scales thus providing higher distinguishability. The focused image was found first by maximising a probability distribution model. The depth was then estimated in closed form. As a result, a layered depth map was formed as shown in Fig. 2.3.

Instead of modelling the blur analytically, the DfD method in [55] was based on supervised machine learning. The training dataset contained a number of images



(a) (b)

Figure 2.3: An illustration of the layered depth map from [54]. Left: original image; right: layered depth map.

of various scenes and the corresponding depth maps obtained from a laser scanner. It argued that local feature alone was not sufficient to estimate depth, and the global context should have also been considered. Thus, a hierarchical Markov random field (MRF) was used to find the depth relationship between different points in the image. The MRF [56] is an undirected graph (e.g., an image or a depth map) of random variables having the Markov property, i.e., in the depth estimation case the blur operation in a local region depends on its adjacent regions. An average root mean square error (RMSE) of 0.09 on log scale was obtained. Although the performance of this approach was lower than earlier DfD methods such as [50] and [51], the target scenes were much more complex and the input images were obtained from uncontrolled environment. The computing speed was not given, but the complex nature of the approach implies it is unlikely to be a fast algorithm.

In [57], the input image was re-blurred using a Gaussian kernel. The amount of blur near the sharp edges was estimated from the ratio between original image and the re-blurred image. As a result, a defocus map for all sharp edges was acquired. By interpolating the edge defocus map in 3D space, a defocus map was obtained. A full depth map was estimated from the defocus map with camera parameters. The results demonstrated that high definition defocus maps could be obtained with low computational cost.

2.3 Depth from Defocus using Two Images

We group these approaches into five categories. First, Fourier domain approach extracts the defocus parameter with Fourier transform of the input images. Second, spatial-filtering approach computes depth by convolving the input image with digital filters in spatial domain. Third, probabilistic approach attempts to estimate the focused image and the depth map with statistical model of the radiance or depth.

The general procedure comprises three stages: a statistical model is first applied to the focused image and the depth map; an estimator/cost function is then formulated with its property, with the arguments being the focused image and the defocus parameter or depth; finally the result is obtained by optimising the estimator/cost function iteratively. Fourth, machine learning approach stores a number of patterns and their corresponding depth values. When an input patch is given, it matches with the stored library to determine depth. Fifth, other approaches that do not fall into any of the above four categories.

2.3.1 Fourier domain approach

One of the methods in [49] estimated depth using two images at the same view, one of which was captured with a pin-hole camera. Gaussian PSF was used and its SD was computed first. The image captured from an ideal pin-hole camera was all focused. The SD was thus zero throughout the image. Fourier transform was performed for a small local region in both images. As a result, the SD of the second image was obtained. This method could be implemented efficiently with fast Fourier transform (FFT). Experiments showed that the accuracy was comparable to that achieved by depth from stereo or depth from motion.

The DfD method in [18] generalised the method in [49] which required one image to be captured with a pin-hole camera. The two images could be captured with any different set of camera parameters, including aperture size, sensor-to-lens distance and focal length. Gaussian PSF was used and its SD was used to calculate depth. The SD was expressed in terms of the camera parameters and depth. Every local region in the image pairs were converted to the frequency domain by Fourier transform. The depth was estimated within the frequency domain. This method was considerably more accurate than that in [49] and [58] when sensor-to-lens distance was varied to obtain the input images, which avoided the noise associated with using a pin-hole camera set-up.

The method in [59] was based on 1D Fourier transform instead of 2D Fourier transform as in [18]. 1D Fourier transform is faster to compute than the 2D Fourier transform, and it is also robust in the presence of zero-mean noise. First the two input image-patches were summed row-wise to obtain two 1D sequences, which were then normalised with respect to brightness. The first 6 discrete Fourier coefficients were then used to create a computed table that corresponds to a specific depth. This table was created in a lookup table fashion which was searched during run-time depth estimation. Experiments showed a 6% error.

The method in [60] argued that the method in [49] required low-order re-

gression fit in the frequency domain of every local region in order to calculate the defocus parameter, and thus was not well suited to the optical system. Instead, the entropy concept was applied to overcome this problem. As entropy is a measure of information content, a blurred image has less entropy than its focused version. It was used to derive the defocus parameter as a function of the input images without regression. The experiments showed that the method outperformed the method in [49]. However, one of the input images was still obtained by a pin-hole camera.

The method in [61; 62] argued that previous DfD methods assumed the depth to be constant over fairly large local region, and considered the blurring to be shift-invariant over those local regions, which leaded to errors when the neighbourhood regions were not considered. This problem occurred since the blurred image could not be simply modelled as the convolution between a shift-invariant PSF and the focused image due to the blurring from the neighbourhood pixels. Two methods were proposed to address this problem. The first method modelled the DfD system as block shift-variant, where the PSF incorporated the interaction of the blur from the surrounding regions. The second method was based on the space-frequency representation of the local regions. The space-frequency representation is the extension of time-frequency representation [63] in image domain. The Short-time Fourier transform and the Wigner distribution are famous examples of time-frequency representation. The second method applied the space-frequency representation of the local region instead of the Fourier transform to allow the blur operation to be shift-variant. An experiment on simulated image compared these two methods with the one in [18]. It showed that the RMSE of these two methods and [18] are 47%, 14% and 6%, respectively. Experiments were also performed on real images with the object placed between 90 and 120 cm away from the camera, which showed that the RMSEs of these two methods were 7.43 cm and 6.18 cm, respectively.

The windowing effect is an artefact produced during Fourier transform of the local image regions, or windows. This leads to spurious high-frequency components at the region boundary. Xiong and Shafer [64] proposed moment filters to address the windowing effect. Gaussian PSF was used but the method was claimed to be applicable to any model. The moment filters were expressed as a function of exponential function whose coefficient was the generalised Laguerre's polynomial. The nth moment could be thought of nth derivative of the Gaussian PSF, hence the filters were used to replace the single first order Gaussian PSF. By a complex analysis in frequency domain, an equation was derived for estimating the defocus parameter. They reported that the finite window problem and the shift variance problem were effectively solved. Experiments showed a RMSE of 0.0003 was achieved compared

to 0.07 by Subbarao's method [18]. Speed information was not provided. However, this method required five times as many convolutions was needed for a typical filter bank method, which made it computationally expensive.

2.3.2 Spatial-filtering approach

The method in [65; 66] was based on the "sharpness map" computed with Laplacian and Gaussian pyramid. The sharpness map indicated the sharpness of an input image. Interestingly, PSF was not used to model the depth. Instead, the focus point was estimated by comparing sharpness maps of images generated with different sensor-to-lens distances. The depth was then recovered using the lens law which was what DfF is based on. However, more than two images were required for higher depth resolution. Experiments showed that the sharpness map could be generated in every 1/8 second. The method took 10 seconds to compute a 64×64 depth maps with 10 input images.

In [58], the Fourier transformation used in [49] was avoided by using Parsevel's theorem [67], which stated that the sum of squared terms over the spatial domain is equal to the sum of the squared terms in the Fourier domain. In the implementation, the image was initially passed through a band pass filter (e.g., a Laplacian filter) to remove frequencies that degrade the depth estimation. The terms were squared and their average was obtained using a Gaussian filter. The average was matched against a pre-computed lookup table of depth values to estimate the depth. A standard error of 2.5% was reported. Nevertheless, the pin-hole camera image was still required which leaded to high diffraction.

The work in [68; 69] claimed that inverse filtering (calculating defocus parameters in frequency domain) used by the methods in [49] and [58] leaded to inaccuracies in finding the frequency domain representation, windowing effects, and border effects. A term called convolutional ratio was thus proposed which relates to depth in spatial domain. The convolutional ratio was expressed as a matrix and was a ratio between one DfD input image to another. Neither of these two images had to be focused. Before depth computation, a lookup table was created which contained the correspondence of convolutional ratio and depth. At run-time, the convolutional ratio was first computed with the two images, where regularisation in spatial domain improved the shape of the ratio. A RMSE of 1.3% was achieved which outperformed the method in [49]. The limitation of the method was its high computational cost.

The DfD method in [70; 71; 72] was based on a spatial-domain convolution/deconvolution transform, or the S-transform. First, a local image region was

smoothed by a differentiation filter so that it became a third order polynomial. The S-transform was derived which was used to recover the focused image from defocused image, and to estimate depth. This method inherited the advantage of the previous one, i.e., it could use two images with any different set of camera parameters. The use of pre-filtering and fast spatial domain computation made this method less sensitive to noise and faster than the previous one. An average error rate of 2.3% was achieved.

In Xiong's first paper on DfD [73], a method based on the Maximal Resemblance Estimation was used to address the windowing problem. First, one image was blurred iteratively to resemble the other. Curve fitting was then used to find the defocus parameter that maximises the resemblance. The experimental results showed that a depth relative error of 1/200 was achieved when the target was 100 inches away.

A real-time DfD method using rational filters was proposed in [16], where a pill-box PSF was used but it claimed that other models could also be used. Different frequency components were analysed in detail so that the ambiguity between inherent smoothness and optical blur was better dealt with. The implementation was done in spatial domain by convolution with a pre-filter and three rational filters. The method was thus both accurate and efficient. Experiments on real images showed that the RMSE was 0.4% $\sim$ 0.8% for a close object and 0.8% $\sim$ 1.2% for an object further than 880 mm from the camera. The entire method could be executed in 0.16 sec to obtain a 512x480 depth map. This approach was insensitive to scene textures. However, according to [17], an iterative minimisation technique was required to compute the rational filters which was unnecessarily complicated. Thus, a simple procedure was proposed in [17] to determine the coefficients of the rational filters. This involved separating the M/P ratio into a linear and a cubic error correction model. First, a straight line was fit for the ratio as a function of the depth. The error between the line and the ratio was accommodated by an error correction scheme. Experiments showed that this method produced better M/P ratio with an RMSE of 1.18% compared to 1.54% in [16].

2.3.3 Probabilistic approach

A DfD method based on MRF was proposed in [74; 75] to address the shift-invariant problem. Both the focused image and the depth map were produced. A maximum a posteriori (MAP) estimate was formulated with the property of MRF. The results were obtained with respect to the estimate by evaluating iteratively with simulated annealing (a global optimisation technique).

An iterative method to recover both the focused image and the depth map was presented in [76]. This method used the near/far-focused image pair, and PSF was used to model the blur operation with a defocus parameter. Both focused image and depth map were estimated by alternately minimising a cost function associated with the information divergence between the true blur image and the estimated focused image convolved with the PSF. During the alternation, the estimated focused image was first initialised to be one of the input images. The cost function was then minimised with respect to the defocus parameter. In the next iteration, it was minimised with respect to the focused image. This process was repeated, and five iterations were reported to produce the best result.

The equifocal assumption stated that the target surface was locally parallel to the lens. The work in [77; 78] reported that this assumption which was used by most previous methods produced problematic results when the target surface was not perfectly parallel to the lens. This problem arised even locally since a patch was not flat no matter how small it was. Subsequently, the PSF was shift-varying, or varying from point to point. In addition, the focused point appeared slightly blurred due to its blurred neighbour which leaded to problem when recovering depth. To address this problem, they forwent the equifocal assumption and approximated the target surface by tangent planes. Both focused image and the depth map were recovered by minimising a cost function with the alternating minimisation algorithm in [76]. In addition, the depth was recovered by minimising the cost function with gradient descent flow incorporating a partial differential equation [79]. Only pictorial results were shown and no speed information was given.

While traditional DfD methods such as [16; 18; 49; 58; 64; 73; 75] used PSF and convolution to model blur, the method in [80; 81] modelled it as an analogue to heat diffusion. Heat diffusion for a single point is the diffusion of the inner energy to its surroundings which resembles the blurring of a focused object point. The most characteristic of this work was that only the depth was recovered without the focused image. This owed to the relative blur which was used to describe the blur difference between the input images. The relative blur enabled the depth estimation without knowing the focused image that was effectively cancelled out during calculation. The heat equation [82] instead of PSF was used to model the blur, where its time variable and the diffusion coefficient reflected the amount of blur which were analogue to the defocus parameter. In addition, to address the equifocal problem, the inhomogeneous diffusion equation was generated by modifying the isotropic heat equation, with a space-varying diffusion coefficient instead of the original diffusion coefficient. By doing so, the diffusion was allowed to be shift-

varying thus solving the equifocal problem. The depth estimation was based on iterative minimisation of a cost function with gradient flow. Experiments showed that the results were very similar to that obtained in [16] which however could not easily incorporate regularisation. Speed information was not given.

DfD using two images requires them to be taken at the same view without magnification. However, this requirement is violated when the camera location, zoom and aperture change during image capture. The work in [83] addressed this problem by formulating an MAP equation. When the camera location, zoom and aperture changes were given, a MAP estimate was solved iteratively to estimate depth using belief propagation. A smoothness constraint was imposed onto such a process making adjacent pixels have similar depth values. At sharp depth discontinuities, colour-image segmentation and plane-fitting were employed to produce acceptable results. Only pictorial results were shown without quantitative evaluation.

2.3.4 Machine learning-based approach

A supervised machine learning-based DfD method was presented in [84; 85; 86]. The local image region and the PSF are analysed in Hilbert Space, a special case of inner product space[1]. This is because the convolution between an image patch and the PSF is effectively the inner product operation. They claimed that by exploiting the Hilbert space the blur analysis became simple and intuitive. An orthogonal projector matrix[2] was computed for each patch with SVD. Thus, each projector matrix corresponded to a unique depth value. Fig. 2.4 illustrates the algorithm. During training, every set of patches of different patterns but corresponding to the same depth is concatenated and proceeded to SVD to calculate an orthogonal projector matrix. During testing, an input patch multiplies with every projector matrix. The depth estimation is depth_n when the product with projector_n is exactly zero. In fact, zeros product is only produced when the input patch is identical to one of the training patches. Thus the estimated depth is obtained when the product is minimised. The most important feature of this method was its robustness, not only could one learn the blurring with one camera for depth recovery, but one could learn the blurring with synthetic images and then used it to estimate depth of real images. The average absolute reconstruction error was slightly lower than 0.02 with two input images. The accuracy became higher when more than two images

[1]The inner product space is a vector space with an additional structure called inner product, which means the inner product of two vectors in this space produces a single scalar [87].

[2]The orthogonal projector of an image patch is a matrix which produces zero when multiplied with the patch. It spans the null-space of the patch due to the product of zero [85].

were used. The depth estimation at each pixel required 510KFlops. However, there was a potential for real-time operation since estimation at different pixels could be done independently.

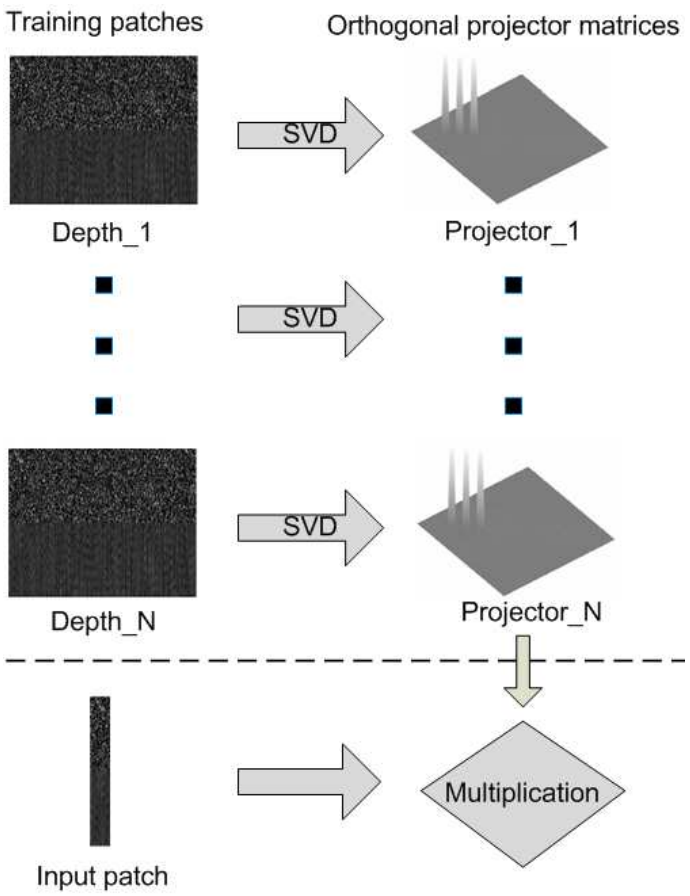


Figure 2.4: Asymmetric aperture pair from [85].

An ANN-based DfD method was proposed in [88]. First, a multi-resolution scheme used for edge detection in [89; 90] was applied to segment the image into foreground and background. This method operated as follows. A multi-resolution pyramid was used to reduce the resolution of any one of the input image, reducing the depth uncertainty and the burden of the ANN. An unsupervised fuzzy clustering technique was then applied to isolate the foreground from the background. The feature vectors that were related to depth were extracted from every image region and used by a pre-trained ANN to determine depth. Experiments showed that the segmentation error for an image whose resolution was reduced to 64×64 is 0.637% and the time taken was 253.8 s using a PC with 600 MHz processor. Accuracies were also given for some objects placed in front of a flat background.

2.3.5 Other approach

A DfD method based on the Hermite polynomial was reported in [91; 92]. The Hermite polynomial was computed using the more blurred image as a function of the partial derivatives of the other image and the blur difference. The blur difference was thus computed by resolving a number of equations. Finally, the blur difference was used to estimate depth with camera parameters. The performance of the algorithm was studied with flat surfaces, step edges, line edges and junctions. Experiments showed that the method was able to avoid uncertain depth results such as those occurring at low frequency regions (or regions that are lack of textures). It also showed that the method outperformed the one in [72].

In [93], the wavelet power was used to measure the defocus difference between the two input image. First, the wavelet transform of a local region was computed. The wavelet power was then calculated using the Parseval's theorem [67]. Finally, the power was used to estimate depth. Experiments on a slanted planar object with the 200×200 input images showed that the method outperformed the Fourier domain [59], spatial domain [72], and Laplacian filter-based methods [16]. In addition, it was many time faster than Fourier domain and spatial domain methods. A similar work is reported in [94] which measured the defocus difference with a ratio obtained from the wavelet coefficients.

The work in [95; 96] stated that circular apertures used in classical DfD methods considerably restricted the accuracy. Instead, a pair of two coded apertures was used to capture the two input images. A criterion was proposed which was optimised to find the best coded pattern, i.e., the asymmetric coded aperture pair (shown in Fig. 2.5). Both focused image and the depth map were recovered. Experiments showed that the method significantly outperformed methods based on circular aperture while only taking 15 seconds for an image pair of size 1024×768.

Figure 2.5: Asymmetric aperture pair from [95].

In [97], a DfD method based on unscented Kalman filter was proposed. The Kalman filter is an algorithm that produces a more accurate (filtering) state based on a sequence of measurements and a mathematical model [98]. It can also be used

for state prediction. For example, the state can be the current location of a car moving with constant speed; the measurements are the previous car locations; the mathematical model is $displacement = speed \times time$. The estimates that based on both measurements and model are theoretically more accurate than using only one of them. However, The Kalman filter can only be applied to linear model, such as $displacement = speed \times time$. It cannot apply to non-linear model, e.g., blurred image intensity as a function of the defocus parameter. A variant of the Kalman filter, the unscented Kalman filter addressed non-linear models [99]. The defocus parameter was measured by gradient descent, and the mathematical model was the convolution between focused image and the PSF. The Kalman filter requires the probability distribution of the state, which was derived with the property of the discontinuity adaptive MRF [100]. Both motion blur and optical blur were considered in this study. These two blurring operations were decoupled by modelling them as convolution with the PSF due to optical blur, and then with the PSF due to motion blur. Experiments performed on simulated images showed a 4% error. No numerical results were shown for real images. Speed information was not provided.

2.4 Summary

Only limited information can be obtained with DfD methods using a single image, which is insufficient to obtain accurate result. In order to estimate a full depth map, assumptions need to be made on the global feature. The most significant problem of these methods is the ambiguity that the cause of low frequencies can be due to both optical blur or lack of texture [68]. For example, a faraway sharp object and a close smooth object produce similar images. They thus often assume the objects have similar sharpness.

For DfD methods using two images, few achieve frequency independence without a complex statistical model or training/testing-based algorithm, i.e., the estimated depth is only related to the blur size rather than the pattern of the blurred object. A solution is to incorporate a frequency parameter into the PSF as suggested in [16]. For most existing methods, the depth is estimated from the ratio of two images with different degree of blur at a particular frequency. In contrast, RO-DfD computes depth using the NIR, or the M/P ratio which is a function of both depth and frequency. The NIR is the ratio between the difference in the magnitude of two images at all frequencies and the sum of the magnitude of them. Due to the complex and iterative optimisation procedure used for the RO design in [16], the method proposed in [17] simplifies the design and improves the depth estimation.

Chapter 3

Literature Review on Human Activity Recognition

3.1 Introduction

A video-based single-person activity recognition system consists of the following four major stages [36]:

- Pre-processing the input video, such as frame-by-frame image segmentation;
- low-level feature extraction from the pre-processed input, such as obtaining interest points;
- mid-level action description with a temporal sequence of low-level features;
- high-level activity description with action descriptions, which copes with complex activities comprising a number of different actions.

We group activity recognition systems into two major categories according to the second stage: the spatio-temporal approaches and the sequential approaches. The former analyses activities directly or indirectly with a XYT volume or spatio-temporal volume created by concatenating all frames of a video, which are further divided into body volume-based, interest points-based and optical flow-based approaches. The latter approaches extract frame-by-frame features and analyse the action/activity as a sequence of features, which are further divided into state model-based approach and exemplar one.

3.2 Spatio-Temporal Approaches

3.2.1 Body volume-based approach

A real-time action recognition system proposed in [101] described actions by motion history image (MHI) and motion energy image (MEI). First, a simple background subtraction operation was used to obtain foreground silhouettes of the moving person or his/her body parts. MHI and MEI were used to describe an action as illustrated in Fig. 3.1, where each row is an example action. MEI is a binary image, where the white pixels indicate where the action takes place. MHI is a grey-scaled image, with low-grey level representing events occurred earlier and vice versa. Thus, MHI is effectively a body volume created by concatenating multiple frames of body silhouettes. During training, the MEIs and MHIs were obtained from training videos of different actions. Seven Hu moments [102] were used to generate statistical descriptions of MEIs and MHIs. When the input moment was obtained, its Mahalanobis distance[1] from every set of the training actions was computed for classification. To address variation in speed, the MHI was normalised to the range [0,1]. An application called the KIDSROOM was suggested, where a child was asked to perform actions that have been taught earlier.

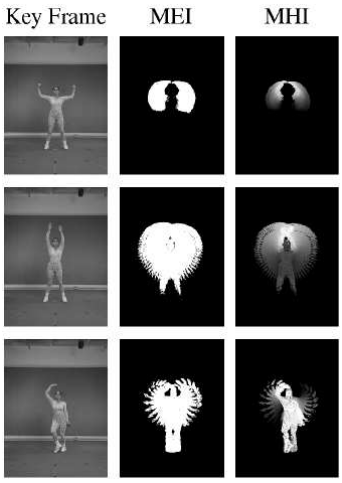


Figure 3.1: An illustration of MEI and MHI from [101]. Row 1-3: example actions.

A spatio-temporal shape-based action descriptor was proposed in [27]. Silhouettes were first extracted from an input video of a person performing an action, which were then used to construct a volumetric spatio-temporal action shape as illustrated in Fig. 3.2. The work in [104] for analysing 2D shapes was generalised

[1] A good explanation with examples can be found in [103].

Figure 3.2: An illustration of the volumetric spatio-temporal action shape from [27]: left: jumping; middle: walking; right: running

to deal with the volumetric shapes. Both spatio-temporal local features and global features were extracted using the properties of Poisson equation, which included local space-time saliency, action dynamics, shape structure and orientation. A simple nearest neighbour classifier was used for action classification. Experiment demonstrates that the method was fast, and was able to deal with partial occlusions, non-rigid deformation, significant changes in scale and viewpoint, high irregularities in action performance and low video quality. The work also introduced the publicly available Weizmann dataset which had been used extensively by researchers in actions recognition for performance comparison.

A recognition system based on over-segmented spatio-temporal volumes was proposed in [41]. A spatio-temporal volume was constructed by concatenating all frames from a video sequence, the three dimensions of which were horizontal, vertical and time. Once the volume was constructed, mean shift [105] was used to cluster the spatio-temporal volume into regions. Since it was impossible to segment the volume correctly without high level semantic knowledge, pyramid regions were segmented from the volume by changing kernel size of the mean shift filter[2]. The shape rather than the value of the segmented regions was analysed. Fig. 3.3 illustrates the volumetric matching algorithm. The input video volume V was first segmented into a number of sub-volume V_i, where $i = 1, 2, 3, ..., 11$. The set of sub-volumes was selected to match the template denoted in bold, which is V_4, V_5, V_7, V_8, with minimal distance represented by the shaded region. In order to find the optimal kernel size, which produced small distance and without too many sub-volumes, the distance was minimised with a penalty for the number of sub-volumes.

Since shape-based matching algorithms performed well for low-texture region and bad for high-texture region, and conversely for flow-based ones, the matching algorithm operated in combination with a flow-based correlation algorithm in [106]. During recognition, a number of action templates were manually constructed, each

[2]First, the volume is segmented using the largest kernel size into segments with largest size; then the volume is segmented using smaller kernel size into segments with smaller size. This process repeats by reducing kernel size until the optimal kernel size is found.

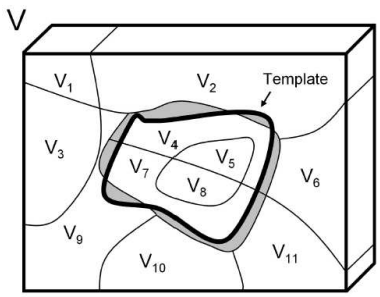


Figure 3.3: An illustration of volumetric matching and kernel size selection from [41].

of which contained one cycle of the periodic action. The correlation score was computed between every frame of the input video and each action template, and the frame was classified as one of the action accordingly. When all frames had been classified, the recognised action was the one with the majority votes.

A 3D CNN-based recognition system was presented in [24]. It stated that most existing methods were problematic and impractical, since they were based on complex hand-crafted features computed from the raw input rather than directly from the raw input itself. The problem was addressed by using CNN that had been successfully applied to object recognition [107]. In the first layer of a CNN, a number of feature maps, such as horizontal-edge maps and vertical-edge maps, were computed by discrete convolutions with relevant kernels (or filters). In order to enable the CNN for spatio-temporal recognition, the 2D convolution was replaced by 3D convolution. During testing, a human detector was used to obtain the raw input. A number of 3D kernels were applied to the input frames to generate a number of features. The output of the 3D CNN was a feature vector containing the motion information in the input frames. The one-against-all linear SVM was learned and used for classification.

3.2.2 Interest points-based approach

2D interest points provide compact representation of an image. The work in [23; 108] extended the concept to the 3D spatio-temporal domain and demonstrated its usefulness in representing actions in video data. Specifically, the Harris and Förstner interest point detection algorithm [109; 110] was extended to spatio-temporal domain. A 2D interest point is a local image region with large variation in image intensity. Similarly, a space-time interest point is found by requiring the image values in space-time to have significant variation in both the spatial and temporal dimensions. Fig. 3.4(a-b) illustrate extraction of interest points from a video of a

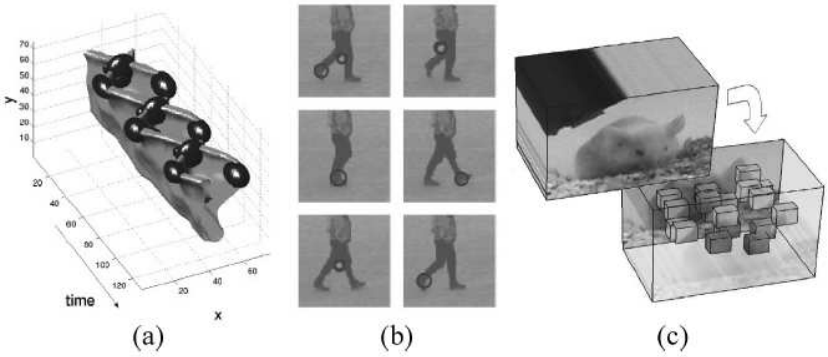


Figure 3.4: An illustration of the extraction of space-time interest points from [108]: (a) interest points detected from the volume of a pair of walking leg (upside down); (b) interest points detected from sample frames; and (c) cuboid features extracted by the method in [111].

pair of walking legs. When the interest points were extracted in terms of their coordinates and neighbourhood, they were used to classify similar events in a video. Using the periodically repeating property of walking, an algorithm in [23; 108] was able to detect a walking person in scenes with occlusions and dynamic background.

The method in [112] extended the work in [23] to recognise complex actions. In order to capture the information of spatio-temporal neighbourhoods of every interest point, a term called spatio-temporal jets was computed which is essentially the Gaussian derivative of each interest point. To enable invariance to camera motion and scale changes, the neighbourhood was transformed with respect to estimated velocity values before computing the spatio-temporal jets. In the training set, all the jets were clustered with K-means clustering, which were then used to build a feature histogram as an action descriptor for recognition. Finally SVM [113] was used to classify actions described by the feature histograms. The work also introduced the publicly available KTH dataset which had been widely used by researchers in action recognition for performance comparison.

A new space-time interest point detector was proposed in [111] for the recognition of human and animal actions. The detector was designed especially for local periodic actions, and it generated a sparse collection of interest points from a video. Each interest point was associated by a small neighbouring 3D volume called cuboid as shown in Fig. 3.4(c). A termed called flattened vector of brightness gradients generated from the interest points was used as the feature vector. A large number of training cuboid features were clustered with K-means, and each action was described as a histogram of cuboid types, i.e., bag-of-words. The system was able to recognise facial expressions, mouse behaviours, and body actions.

A system based on unsupervised learning of human actions was presented in [114]. The space-time interest points were first extracted from a video by the detector in [111]. They were then clustered into a number of spatio-temporal words (or codewords) by K-means algorithm with Euclidean distance as the clustering metric. An action was thus represented by a set of codewords from the codebook. A single codebook was created for all actions, which was a collection of all the codewords obtained by all interest points extracted from all action videos. A simple illustration of a codebook is shown in Fig. 3.5 which contains 3 codewords and 2 actions. The probability distribution of actions was learnt using one of probabilistic Latent Semantic Analysis [115] and Latent Dirichlet Allocation [116], which were previously used in the field of text mining. Thus, spatio-temporal words, action categories and videos were analogue to text words, text topics and text documents respectively. To recognise an action from a video, a posterior probability was maximised taking all the interest points as input. The system was able to deal with noisy feature points resulted from moving cameras and dynamic background.

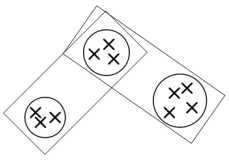


Figure 3.5: A illustration of action codebook in [114]. Key: cross - a interest point; circle - a codeword; and rectangular box - an action.

3.2.3 Optical flow-based approach

Optical flow was the pattern of motion of objects in a visual scene due to the relative motion between the observer, such as an eye or a camera, and the scene [117]. Fig. 3.6 illustrates the optical flow (right) created from the original scene (left), which are denoted by arrows. These arrows reveal the velocity information of different body parts including both direction (denoted by the direction of an arrow) and speed (denoted by the length of the arrow).

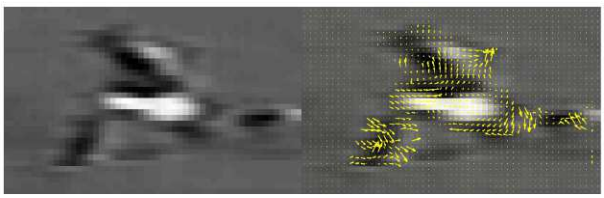


Figure 3.6: Illustration of optical flow with a running footballer from [118].

One of the earliest optical flow-based recognition system was proposed in [118]. A simple normalised-correlation-based tracker was used to track a human subject in a video. A figure-centric spatio-temporal volume centred at the subject is computed. The optical flow with x and y components was computed using the method in [119]. A motion descriptor comprising four blurred channels was created from the optical flow and wavelet transform. The process was done frame by frame to obtain a sequence of motion descriptors. To find the similarity between an input sequence and a stored sequence, the similarity matrix was computed based on frame-by-frame correlation. The system was able to recognise movements of a ballet dancer, a tennis player and football players.

A recognition method proposed in [120] utilised kinematic features obtained from the optical flow of an image sequence. It was claimed the kinematic features allowed more discriminative representation of complex human actions. A set of kinematic features includes divergence, vorticity, symmetric and antisymmetric flow fields, and etc., each of which captures a different aspect of optical flow. A kinematic mode representing an action was computed from the spatio-temporal kinematic features volumes using PCA [121]. For activity classification, the multiple instance learning algorithm was used to address the problem when a complex action required more than one kinematic modes to represent its dynamics.

The work in [43; 44] was based on dense trajectories. It stated that dense interest point samplings resulted in higher accuracy than sparse sampling. First, a dense set of feature points were sampled from each frame. Their trajectories were tracked with a dense optical flow algorithm. The influence from camera motion was reduced by a descriptor based on motion boundaries computed by a derivative operation on the optical flow field. Action classification was achieved using a bag-of-features representation and a SVM classifier.

3.3 Sequential Approaches

3.3.1 State model-based approach

In action recognition, state models are statistical graphical models used to represent an action or an activity, where each distinct observation is associated with a state. The most popular state model for recognition is the HMM. A HMM is a statistical system comprising a number of hidden states and their outputs [122; 123; 124; 125; 126]. The hidden states are not directly visible, but their output can be observed. A HMM consists of three key parameters, start probability (or initial state probability) matrix, transition probability matrix and emission probability (or symbol output

probability) matrix. The start probability matrix contains the probabilities from the initial state to all hidden states. The transition probability matrix contains the probabilities from every hidden state to all other hidden states. The emission probability matrix contains the probabilities from every hidden state to the outputs. During training, the output observations are ordered in time order. The HMM parameters are then estimated by Baun-Welch algorithm with the observations as input. During testing, the forward algorithm [127] is used to compute the similarity between a test activity and the gallery activities, in terms of the probability of a set of parameters given the test output observations.

The action recognition system that first employed a HMM was presented in [25]. In that system, each video frame was transformed into a feature vector with mesh-grid, which was then clustered, with each centre being a codeword. Each codeword corresponded to an output symbol in HMM, and every feature vector was assigned to its nearest symbol. For a test video, the forward-algorithm was used to estimate the similarity between the symbol sequence and the gallery activities.

The method in [128] analysed multi-person activity with a HMM. First, the background subtraction algorithm in [129] based on the Gaussian mixture model was used to extract the moving human subjects. The mean-shift tracker [105] was applied to track individual human subject. An image moment-based shape descriptor was introduced to convert the silhouettes into feature vectors. Posture was defined as a sequence of silhouettes, and postures were recognised by HMM. The interval algebra in [130] was applied to analyse the interaction of different subjects performing a sequence of identified postures.

The method in [131] was also based on HMM. Silhouettes were first extracted from video frames, which were converted into feature vectors by R-transform [132], i.e., a 1D version of the radon transform [133]. PCA was then applied to all the gallery feature vectors to further reduce their dimensionality. A HMM was applied to these dimensionality-reduced feature vectors. A similar work was reported in [33], where PCA was replaced by kernel discriminant analysis [134] that allows non-linear mapping to be learnt.

The paradigm of Layered HMM (LHMM) was first presented in [135], where two layers were used. The work illustrated the classification of human activities in office environment of using microphones, a video camera, keyboard and mouse. These activities included phone conversation, user's presence, engagement in some other activities, distant conversation and presence of nobody. The bottom layer classifies atomic actions with signals obtained with the input devices. The output classification of the bottom layer becomes the input of the top layer that in turn

classifies an activity. The major advantage of LHMM is that only the bottom layer requires retraining when the office environment is changed.

Another similar work using LHMM was reported in [136], which was called hierarchical HMM, where two layers were used. It aimed to classify indoor activities that included taking a small meal, having snack and taking a normal meal. Each one was described as a trajectory of the human subject in a video, obtained by Rao-Blackwellised particle filter [137]. The recognition system in [138] constructed a multi-layered HMM to recognise group activities. The system is also composed of two layers of HMMs. The bottom layer recognises atomic actions such as speaking, writing and idling. With the bottom layer's output as input, the top layer recognises group activities such as monologue, discussion and presentation.

Since HMM-based methods assume Markov property which means each observation is only dependant on its immediate previous one. As a result, a sequence of observations is intra-independent, and this is not the case for activities consisting dependent observations. Conditional random field (CRF)-based methods handle this problem by modelling a conditional distribution of action labels given the observations. One of the major disadvantages of CRF is that they require considerably more training data to reliably estimate all parameters. In [139], CRF was studied where a set of statistical model was created for the conditional dependencies amongst the observations. Learning was implemented using convex optimisation and recognition was archived by dynamic programming. As a result, CRF was found outperforming HMM that failed to consider the contextual dependences between observations. A similar work was found in [140] where the input silhouettes were represented more compactly by kernel PCA. Their experiments showed that the system was robust and was able to handle noise, partial occlusion and irregularities in motion styles. Another CRF-based method in [141] aimed to cope with large variation in orientation and scale of the subject. Both shape (i.e. edges) and optical flows were used as the low level features, and they claimed that this worked better than either of them along. A more recent CRF-based system was proposed in [142] that indented to provide a guaranteed global optimisation during the learning process.

3.3.2 Exemplar-based approach

Exemplar-based approach to activity recognition obtains a feature vector from every frame, and then creates a motion template with multiple of such vectors to describe an action or an activity. Classification techniques such as K-nearest-neighbour are used to recognise the action or activity using the templates.

A hand gesture recognition system based on view-based representation of

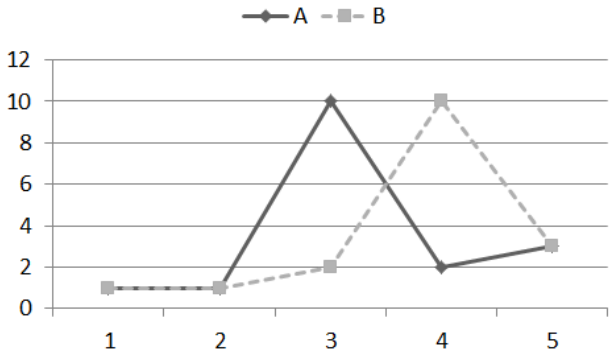


Figure 3.7: An illustration of DTW.

object and DTW was presented in [26]. First, a set of view-models was constructed for a given object, each of which corresponded to a specific transformation, such as a view angle and a scaling coefficient. The correlation scores were a sequence of correlation values, obtained by computing the correlation between every view-model and a training video with a match filter. The mean and variance of these scores were calculated as a function of time, which was the template for that video. In order to achieve speed invariance, DTW [143] was used to compute the similarity between an input video and the video library. DTW is an algorithm for measuring similarity between two time series or sequences which may vary in speed. It was successfully used in speech recognition where the input signal varies significantly in speed [144]. As an illustration, consider two time sequences $A = \{1, 1, 1, 10, 2, 3\}$ and $B = \{1, 1, 1, 2, 10, 3\}$ in Fig. 3.7. They have high similarity with only a slight difference in speed. If one to one correspondence is used to compute their difference, $A(4) = 10$ corresponds to $B(4) = 2$, and $A(5) = 2$ corresponds to $B(5) = 10$, and this leads to very large distance or low similarity. DTW solves the problem by warping each signal in the temporal dimension such that the highest similarity is obtained. Experiments show that hello and good-bye gestures were successfully recognised. With specific hardware, a recognition rate of 10 Hz was achieved.

A system with a combination of Isomap and DTW was proposed in [145]. First, silhouettes of a moving human subject were extracted with a simple background subtraction operation. A bounding box was then generated that encompasses every silhouette. The silhouettes were then aligned with respect to their centre. To cope with different subject size and clothing variance, each silhouette was smoothed by being converted to a grey-scale image with the distance transformation in [34]. Isomap [31] was applied to embed the smoothed silhouettes library into the manifold space. The RBF in [34] was used to learn the mapping from the input silhouettes to the embedded silhouettes. During recognition, the smoothed sil-

houettes obtained from the input video were embedded. The sequence of embedded coordinates was matched with the library using DTW and the nearest neighbour classification scheme.

One of the earlies methods using body joints trajectory was presented in [146], which demonstrated the ability to recognise 9 action primitives in a ballet. Fig. 3.8 illustrates one such primitive with six key postures. Recognition of a primitive was successful if its key postures were identified without those from other primitives. Fourteen markers were placed on the dancer's joints, and parameters such as torso height and joint angles were used to describe a pose, which spanned a phase space. Every possible pair of parameters was found for each primitive, which formed a number of point plotted in 2D space as illustrated in Fig. 3.9, where the two axioms represent the two parameters. The trajectory of an example primitive was denoted by $\times$, where a cubic polynomial was fitted taking one of the parameters as input and another as output. During recognition of a pose, the cubic function took one parameter as input and computes the other. If the output was not far from the output parameter used for the polynomial fit, it was determined as a part of the primitive.

Figure 3.8: An illustration of an action primitive recognised (from [146]), with 14 markers on the actor's clothing.

A recognition system based on PCA and manifold learning was presented in [32]. First, body parts were tracked by using the algorithm in [147]. Their motion parameters computed from every frame were concatenated to form a single column vector that represents one activity. Column vectors corresponding to all examples of the activities were then concatenated and input to PCA using SVD. All video sequences were initially assumed to be of the same length and were temporally aligned. When the input activity video was converted to the column vector, it was projected onto the principal axioms with the orthogonal matrix obtained by SVD. The system then used a set of affine transformations to cope with length and speed variations. Finally, the nearest neighbour classification algorithm was used for activity recognition.

A system for 3D pose inferring from 2D silhouettes based on manifold learning with LLE was reported in [34]. A silhouette was first extracted with a back-

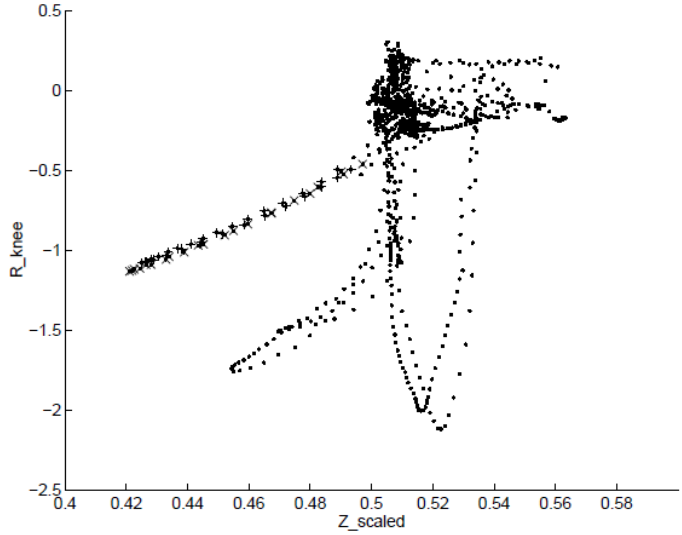


Figure 3.9: An illustration of the trajectories of parameter pairs of 9 primitives (from [146]).

ground subtraction operation. It was then converted to a grey-scale image with respect to the distance between every pixel and the silhouette contour. During training, the grey-scale image was embedded to the manifold space with LLE [30]. Since any input image could not be directly mapped to the manifold space, the mapping was learnt by RBF [148]. The 3D pose was represented by a set of joint angles. The mapping from the manifold to the pose space was also learnt by RBF. During recognition, the input grey-scale image was first embedded by RBF, which was in turn mapped to the pose space by another RBF mapping.

The recognition system in [149] modelled human actions in 4-dimensional (4D) space, XYZT, where X and Y are the horizontal dimensions, Z is the vertical dimension, or height, and T is the temporal dimension, as illustrated in Fig. 3.10. An affine transformation was used to project the 4D points onto XYT space. A matrix was created by concatenating all body points at all the time instances during an action. During testing, the subspace angle between the input matrix and every stored matrix was computed. The difference between the angle and 180 degree was minimised to obtain the action category. Experiments showed that the system was able to deal with variation in viewpoint, execution rate and anthropometry of actors. A similar method was proposed in [150] which recognised action videos acquired from moving cameras.

The work in [151] stated that most current recognition systems were based on local spatio-temporal features which limit the ability to recognise long and com-

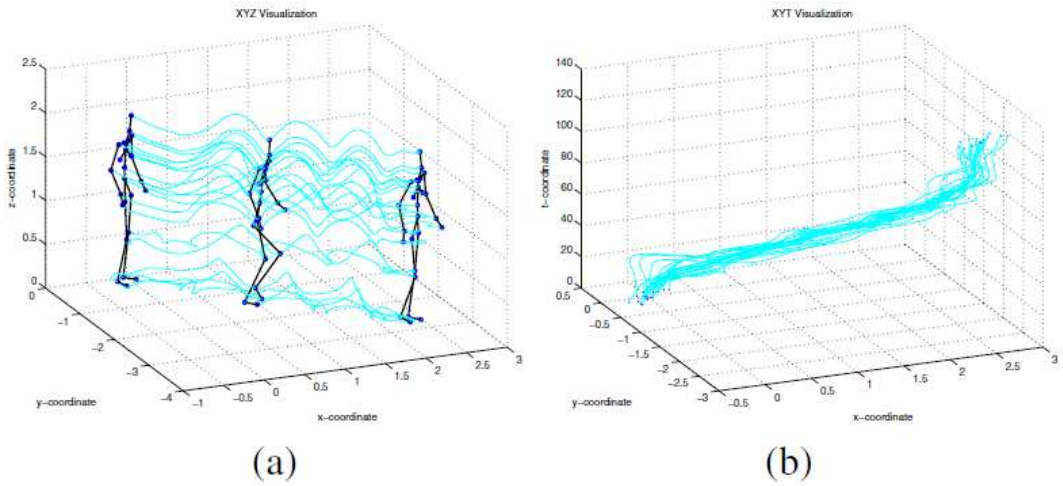


Figure 3.10: An illustration of 4D trajectories of parameter pairs from [149]: (a) XYZ plot of an action, where vertical axis represents height; and (b) XYT plot of the action, where the vertical axis represents time.

plex actions. The problem was handled by describing an activity by the dynamic subspace angles computed from an activity video. First, feature sequences such as interest points or silhouettes were extracted from the video in temporal order. A Hankel matrix was generated for the sequence. The dynamic subspace was then computed from columns of the matrix. To find the similarity between two activities, their dynamic subspace angle was computed by the canonical correlation between the subspaces. If the activities were identical, this angle was 0 since the outputs of the same activity lay in the same subspace. Finally, a multi-class SVM was constructed for classification.

3.4 Summary

In this chapter, a number of different action/activity recognition systems are reviewed. Spatio-temporal approaches can effectively recognise simple and periodic actions such as those in the KTH dataset [112] and the Weizmann dataset [27]. Body volume-based methods provide straightforward solutions but they cannot normally deal with motion speed variations. Interest points-based methods are very robust against noise and illumination changes without background subtraction or body-part modelling. Their major limitation is that they cannot recognise complex actions or non-periodic actions. View-invariance is another unsolved problem which they need to address. Optical flow-based methods are computational efficient and are able to handle camera motion. Nonetheless, they have inherent difficulty in dealing with

variation in action speed.

Sequential approaches generally analyse temporal relations between frame-by-frame features, and this enables them to recognise non-periodic activities and more complex activities than most spatio-temporal methods. State-based methods use probabilistic analysis of activities. They compute a posterior probability of the activity category given an input video, thus enable easy incorporation of other algorithms. These systems can also be trained with new data without the original data [152]. The major limitation of the state-based methods is that they normally require a large number of training data. Exemplar-based methods are more flexible in that the frame-by-frame feature vectors can be arranged in any manner to suit a particular application. In addition, methods based on the DTW algorithm can effectively cope with speed variations. Moreover, exemplar-based methods require less training data than state-based methods.

Chapter 4

Experimental Procedures for Acquiring DfD Input Images

4.1 Introduction

The far-focused and the near-focused image pair is the input of many DfD methods such as [16; 77; 85; 97; 153]. They are generated by focusing a camera at the furthest and the nearest measurable distances of the scene, respectively. Focusing is achieved by fixing the lens and shifting the sensor along the optical axis. Fig. 4.1 illustrates a procedure for capturing DfD image pairs. First, the furthest measurable distance, or the far-focused object distance u_1 is determined. The lens law [154] states that

$$\frac{1}{F} = \frac{1}{u} + \frac{1}{w} \ , \tag{4.1}$$

where F is the focal length (a constant provided by the lens manufacturer), u is the object distance and w is the distance between the lens and the focused image, or the focused image distance. Thus the far-focused image distance

$$w_1 = \frac{1}{\frac{1}{F} - \frac{1}{u_1}} \ . \tag{4.2}$$

Similarly, the near-focused object distance u_2 is defined, and the near-focused image distance w_2 is computed using the lens law. Finally, to obtain the far-focused and near-focused images for any object, the sensor is respectively moved to w_1 and w_2 away from the lens.

Although this procedure is theoretically correct, in practice w_1 and w_2 cannot normally be directly and accurately measured for an off-the-shelf camera. Thus, high-precision tools are also needed to move the sensor. To obtain images for DfD,

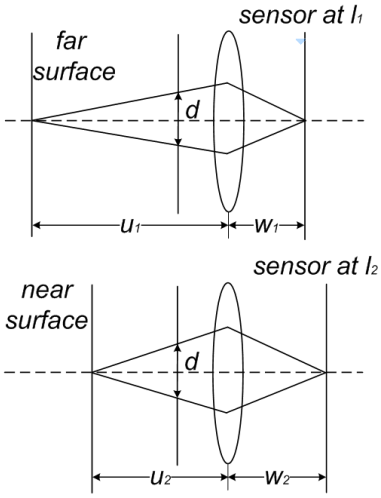


Figure 4.1: Far-focused (top) and near-focused (bottom) images capturing. Dash lines denote the optical axis.

Figure 4.2: Raj's camera system from [155]: a 35 mm lens (left) and a 50 mm lens (right).

a focus ring control was used in [17; 155] as shown in Fig. 4.2. The focus ring as illustrated in Fig. 4.3 was used to focus the target at different distances. When the focus ring is turned, the sensor-lens distance is changed. This is equivalent to moving the lens in the thin lens model in Fig. 4.1 along the optical axis. To obtain a far-focused image, the focus ring is adjusted such that the image for the far object is focused. Similarly, the focus ring is adjusted to focus on the near object to obtain the near-focused image.

The focus ring control "remembers" the two positions so that more DfD image pairs can be generated easily. This memory function is achieved by its two plastic rings. One of them (Ring 1) is firmly attached on the focus ring and the other (Ring 2) on a fixed position on the lens. Ring 1 is turned with the focus ring. Ring 2 has two protruding screws on its edge which mark the two focus ring positions. They stop Ring 1 from turning at the two positions where the near/far-focused images are respectively captured. However, this system has a serious drawback. When the focus ring is turned while the sensor is fixed, not only the sensor-to lens position w

Figure 4.3: The 50 mm lens used in [155; 17].

is changed, the object distance u is also modified. This means the measured depth is ambiguous, since the image pair no longer corresponds to the same object scene.

To address this problem, two practical camera systems are proposed in this thesis: the single micrometer controlled (SMC) system, and the double micrometers controlled (DMC) system. In both systems, the lens is fixed and only the sensor is moved to generate the images. In SMC, a funnel holder is used to fix the lens, and a micrometer of 10 μm precision allows fine adjustment of the sensor-to-lens distance. In DMC, the funnel holder in the first design is replaced by one controlled by a micrometer with other components for finer adjustment. We have also implemented a graphical user interface (GUI) programme *DfDtool* using MATLAB 2008 for efficient image acquisition.

This chapter is organised as follows. Section 4.2 presents the design of SMC and Section 4.3 describes the DMC. The experimental procedure of capturing DfD image pairs is presented in Section 4.4. Section 4.5 describes *DfDtool* and its benefits. Finally Section 4.6 summarises this chapter.

4.2 The Single Micrometer Controlled System

Fig. 4.4 shows the SMC where some important components are tagged. Fig. 4.5(a) is another view of the SMC where the lens holder grabber is highlighted. The camera system consists of a Nikkor 50 mm manual-focus lens [156], a AVT Guppy FireWire charge-coupled device (CCD) sensor [157] connected to a standard desktop PC running on Pentium 4 processor at 3.20GHz and 1 GHz memory, by a FireWire or IEEE 1394a cable [158].

Fig. 4.5(b) is a close view of the grabber showing three of its most important components. Screw A allows the vertical position of the grabber to be changed and fixed; screw B enables the horizontal position to be changed and fixed. Both screws are adjusted by Allen key (or hex key). The lens can be firmly fixed by the tap. When the lens is positioned at the designated location, it is not allowed to move

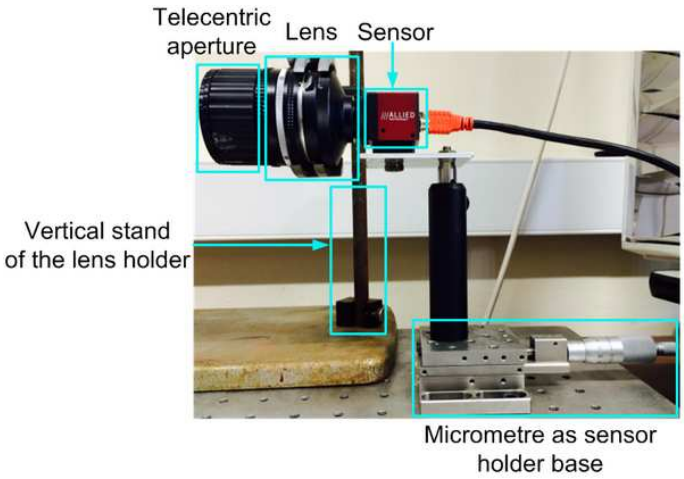
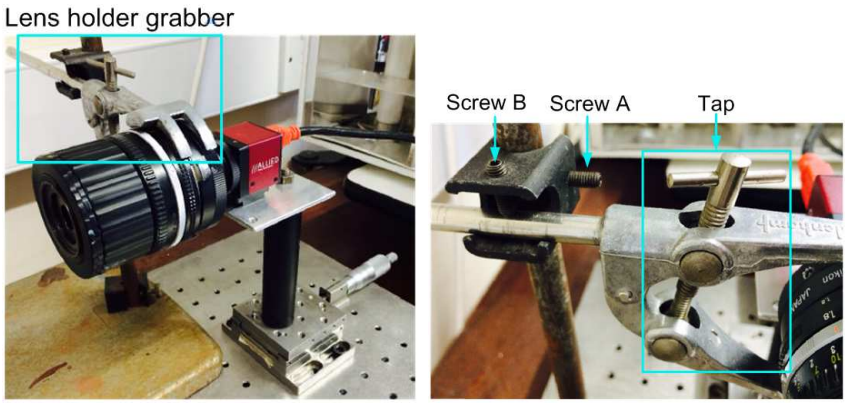


Figure 4.4: Side view of the SMC.



(a) (b)

Figure 4.5: The lens holder grabber of SMC: (a) isometric view; with (b) components highlighted.

during image capture. Fig. 4.6 illustrates how the vertical position of the sensor is adjusted. The screw is loosened to allow the metal cylinder to move vertically with the sensor. The screw is then tightened to fix the position.

The sensor is fixed onto a base controlled by the micrometer as shown in Fig. 4.4. A closer view of the micrometer is shown in Fig. 4.7. When the micrometer roller is turned forward, the lens holder is pushed to the left; when it is turned backward, the sensor holder is pushed back by an internal spring. The micrometer produces a reading which varies when the micrometer is turned. This reading indicates the location of the sensor on the optical axis.

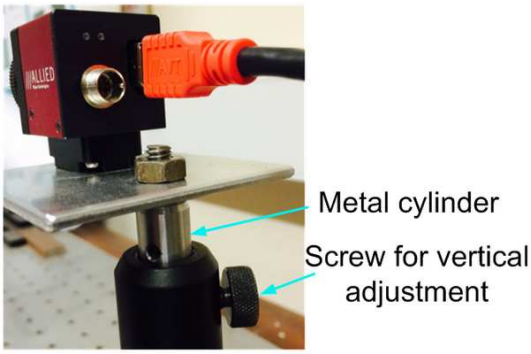


Figure 4.6: Close view of vertical position control of the sensor.

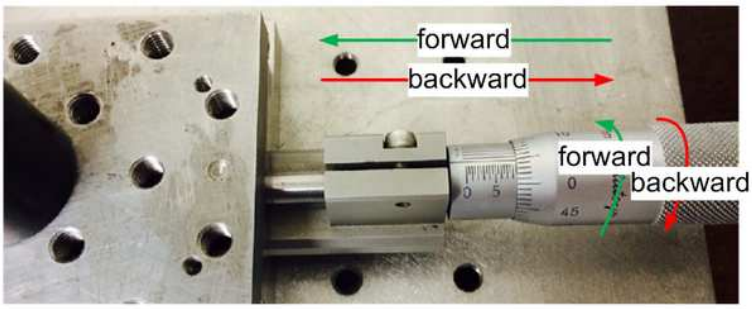


Figure 4.7: Close view of the micrometer.

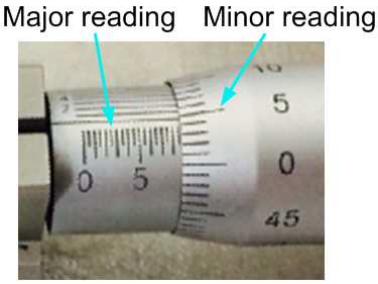


Figure 4.8: Major and minor reading of the micrometer.

The micrometer has the major reading and minor reading as shown in Fig. 4.8. The major reading consists of higher digits and lower digits. Each higher digit is 1 mm equally divided into two lower digits. Thus, each lower digit is 0.5 mm. A rotation of the minor reading by 360 degree leads to one lower digit change in the major reading. The minor reading also consists of higher digits and lower digits. Every higher digit is 50 micrometres equally divided into five lower digits. Thus, each lower digit in the minor reading is 10 micrometres.

4.3 The Double Micrometers Controlled System

Fig. 4.9 shows several views of the lens holder design, where the lens (in sky-blue) is attached to an external telecentric aperture (in purple). The vertical lens holder stand is in cyan and the sensor in violet. The lens holder base (in yellow) is on a tilt control platform (in brown). The platform is on another micrometer (in green), which is fixed on the optical bench (in light-green). Other components of the SMC are shown in grey. Fig. 4.10 shows the DMC with the fabricated lens holder. Since this design has two micrometers, we call the micrometer under the sensor Micrometer A and the one under the lens holder Micrometer B.

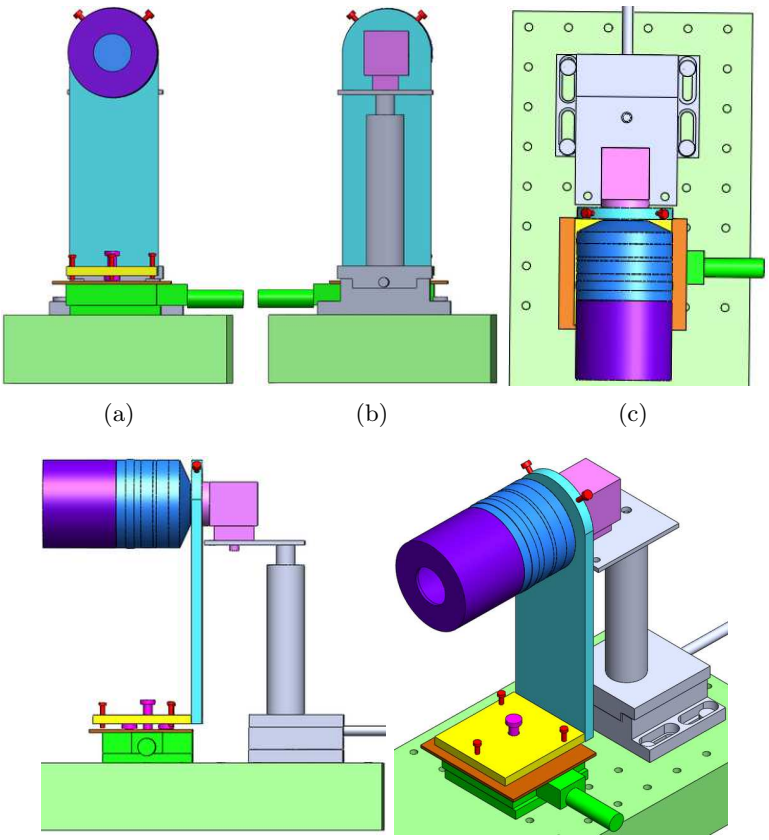


Figure 4.9: 3D model of the camera system designed with SolidWorks 2012 [159]: (a) front view ; (b) back view ; (c) top view ; (d) side view; and (e) the isometric view.

This design has several advantages. First, Micrometer B is introduced which allows precise horizontal adjustment of the lens. Second, the central screw in the holder base allows rotational movement of the lens in the horizontal plane. Third,

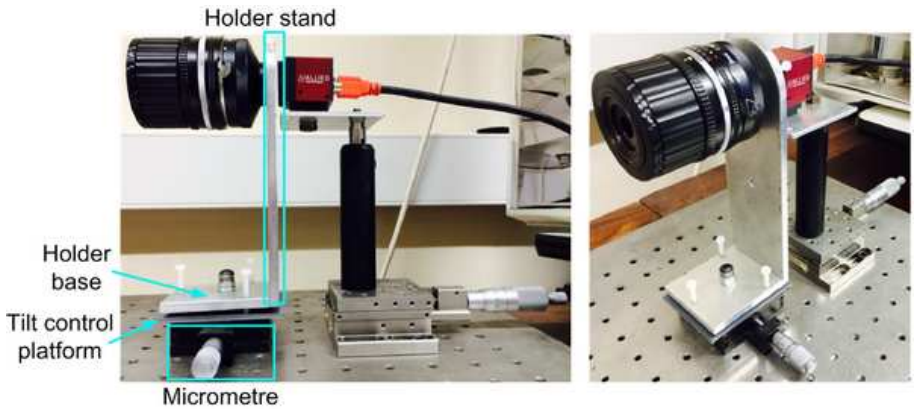


Figure 4.10: The camera system with the fabricated lens holder: side view (left) and isometric view (right).

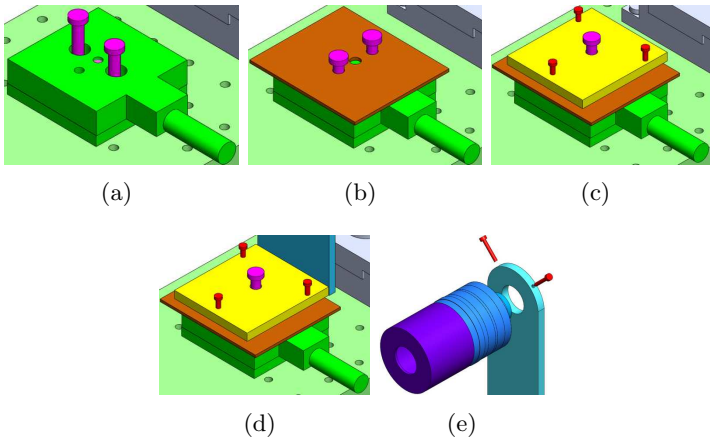


Figure 4.11: Assembly procedure for the lens holder.

the three small screws (in red) on the holder base and the control platform (in brown) enable tilt control of the lens.

Fig. 4.11 shows the assembly of the DMC system. First, Micrometer B is secured on the optical bench with two screws as shown in Fig. 4.11 (a). Second, the tilt control platform is fixed on Micrometer B with another two screws as in (b). Third, the lens holder base is attached on top of the platform with the central magenta screw as in (c), where a spring is inserted between the screw hat and the base (not shown) for double securing. Fourth, the lens holder base is attached to the lens holder stand as in (d). Finally, the lens is inserted to the lens holder hole and secured with the two screws (in red) as in (e).

4.4 Image Capture Procedure

The procedure for image capture using either SMC or DMC requires the user to provide the maximum blur circle radius, which determines the working range along with other optical parameters. The furthest measurable distance is determined first, and the nearest measurable distance is computed accordingly. During image acquisition, the micrometer is adjusted to focus on these two distances to capture the far- and near-focused images, respectively. The following four steps are involved:

1. Define the furthest measurable distance u_1, or the far-focused object distance. Place a flat surface at that distance away from the lens.

2. Move the sensor along the optical axis by turning Micrometer A. When the image is focused, record the reading of the micrometer as m_1.

3. Compute $w_2 - w_1$, which is how far the sensor should be moved to generate the near-focused image (refer to Fig. 4.1), by

$$(w_2 - w_1) = \frac{2FR_{max}}{d} \ , \tag{4.3}$$

   where R_{max} is the predefined maximum blur circle radius that is normally 10 micrometres [16; 17], and d is the aperture diameter. The near-focused reading

$$m_2 = m_1 + w_2 - w_1. \tag{4.4}$$

4. To capture the DfD image pair, turn Micrometer A to m_1 and m_2 for the far-focused and near-focused images, respectively.

Note that the procedure is similar when the nearest measurable distance is determined first.

4.5 *DfDtool*

DfDtool is a self-contained MATLAB-based GUI program to facilitate efficient DfD experiments. It has four major benefits. First, it allows precise search of the most focused image which is required in the far/near-focused position determination. Second, it allows frame grabbing and the frame can be saved as an image file to be used later. Third, it enables calculation and display of the depth map when the DfD image pair is captured. Finally, it contains a parameter calculation module which estimates the experimental settings for generating far- and near-focused images.

4.5.1 Graphical interface

DfDtool contains seven modules as shown in Fig. 4.12. The configuration module enables the user to configure the output directory of the saved image, turn the display of the FFT value on and off, and set the depth map patch size and post median filtering (PMF) kernel size.

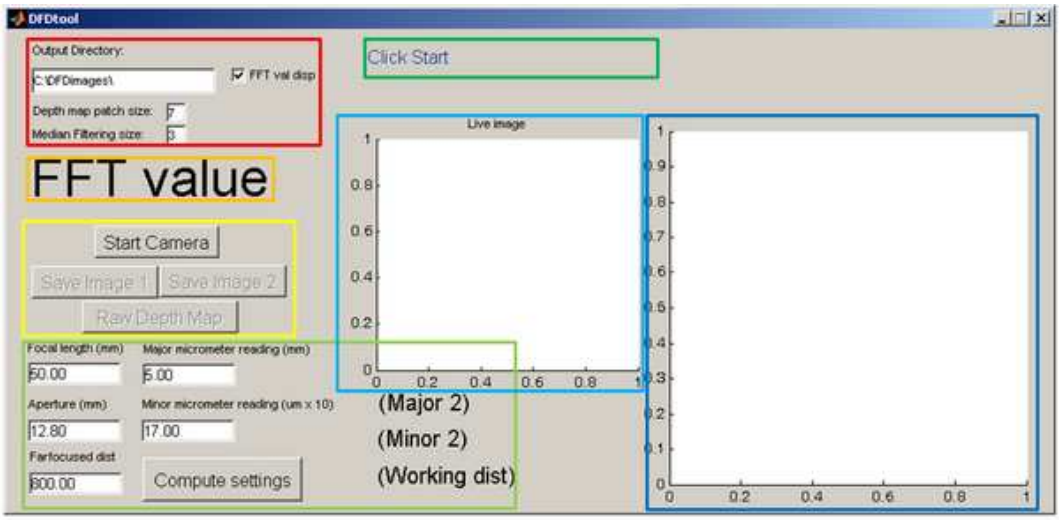


Figure 4.12: The *DfDtool* GUI: configuration module (red box), FFT value (orange box), control module (yellow box), parameter calculation module (light green box), status module (green box), live image display (light blue box), and depth-map display (dark blue box).

The FFT display module shows the current FFT value

$$v_{fft} = \sum_{y}^{0.25Y} \sum_{x}^{0.25X} \check{\mathbf{I}}(x, y) \ , \tag{4.5}$$

where $\check{\mathbf{I}}$ is the absolute values matrix of 2D FFT of the input image $\mathbf{I}$. x and y are the row and column indices, respectively. X and Y are the row number and column number respectively. v_{fft} is effectively the sum of high-frequency components of the image and is maximised when the image is most focused. Our experiments show that this process is faster than computing variance as in [72] while as effective. It takes less than 0.1 seconds per 640×480 image for a standard desktop computer with a Pentium 4 processor and 1 GB memory.

The control module consists of four buttons. When the "Start Camera" button is pressed, the camera starts to record (i.e., image capture) and the "Save Image 1" button is enabled. At the same time, the name "Start Camera" is changed

into "Stop Camera" which when pressed will stop the recording immediately.

When the "Save Image 1" button is pressed, the far-focused image is saved into both memory (as a variable) and the disk (as a file), uniquely named according to the current date and time. The "Save Image 2" button is also enabled as a result, which when pressed will save the near-focused image. The "Raw Depth Map" button is enabled after this action, which when pressed will display the raw depth map using Watanabe's method [16].

The parameter calculation module computes the near-focused Micrometer A setting. When a user has entered the focal length, aperture diameter, far-focused object distance, Micrometer A major reading, minor reading and then pressed the "Compute settings" button, the major and minor readings required for near-focused setting are displayed. The working range in millimetre is also displayed. The status module shows the current status of the program. Finally, the live image display module shows the live recording of the camera.

4.5.2 DfD calibration and image acquisition with *DfDtool*

In this thesis, calibration means finding the far/near-focused optical settings including far/near-focused object distances and Micrometer A readings. The first step of the procedure is illustrated in Fig. 4.13, where the far-focused object distance is determined as 800 mm. A flat surface covered by sandpaper is used as the object for calibration and placed at this distance from the lens. After the "Start Camera" button is pressed, the live image and the corresponding FFT value are displayed by *DfDtool*. The user then needs to adjust Micrometer A to find the far-focused reading m_1 by maximising the FFT value.

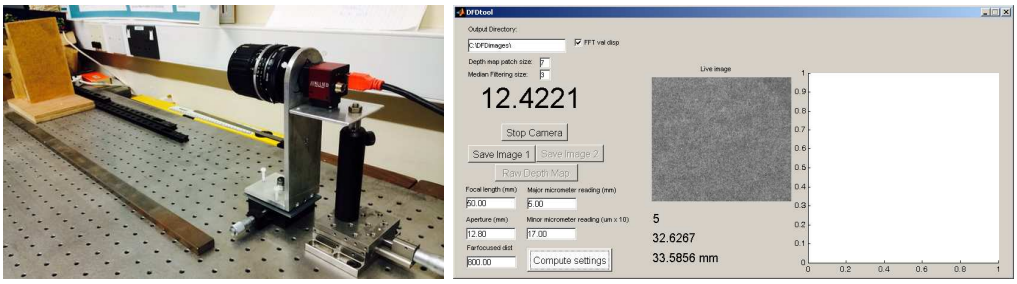
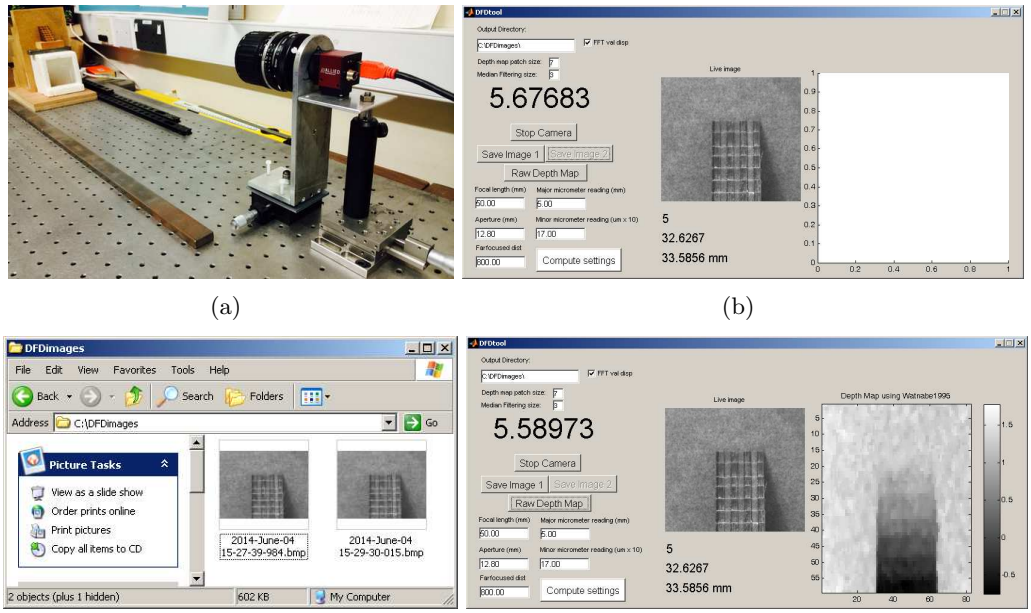


Figure 4.13: DfD calibration using *DfDtool*: left: object used for calibration; right: the *DfDtool* interface.

For a predefined maximum blur circle radius (2.703 pixels in this case), the focal length (50 mm), aperture diameter (12.8 mm), far-focused object distance (800 mm), major Micrometer A reading (5mm) and minor micrometer reading (0.17 mm)



(a) (b)

(c) (d)

Figure 4.14: DfD image acquisition and raw depth map computation with *DfDtool*: (a) test object with camera, (b) the *DfDtool* interface, (c) the saved image files, and (d) the interface with the computed depth map.

are entered in the parameter calculation module. The near-focused Micrometer A reading m_2 is then computed according to Eqn. (4.3) and displayed on the *DfDtool* interface.

In Fig. 4.14(a), a test object (a wooden staircase) is placed in the camera view. Micrometer A is turned to m_1 and "Save Image 1" button is pressed in (b). Micrometer A is then turned to m_2 and "Save Image 2" button is pressed. The image pair is saved on disk shown in (c). When the "Raw Depth Map" button is pressed, a depth map is computed and shown on the right of the interface in (d), where brighter pixels represent larger depth and vice versa.

4.6 Summary

This chapter presents the processes involved in obtaining DfD image pairs. Two systems are proposed. The SMS provides precise control of the sensor-to-lens distance with a micrometer. The DMS allows finer lens position adjustment for a dedicated lens holder. In addition, *DfDtool* enables convenient DfD calibration and image capture. The current limitations of the system include that the lens and the sensor cannot be aligned perfectly with the optical axis. Finer vertical adjustment is also

required for higher-quality image acquisition. Two DfD correction algorithms presented in Chapter 6 are used to address this problem and to eliminate an elliptical distortion problem.

Chapter 5

Depth from Defocus based on Rational Operators

5.1 Introduction

DfD and DfF are methods for recovering 3D shape of a scene. DfF (e.g., [160; 161]) is based on the lens law [154], i.e.,

$$\frac{1}{F} = \frac{1}{u} + \frac{1}{w} \ , \tag{5.1}$$

where F is the focal length, w is the distance between the lens and the image plane when the image is in focus, and u is the distance between an object point P and the lens as shown in Fig. 5.1. Thus, if w is known then u, i.e., the depth of the object point can be recovered. However, for an object with continuous change in depth, at least ten images are required to estimate the object depth map [162]. The challenge of DfF is deciding when the object is in focus. Recent methods to address this challenge include those in [160; 161]. DfD requires only two images captured with different focus setting, hence it is more suitable than DfF for real-time applications. In Pentland's DfD scheme [49], the first image was captured with a large depth-of-field so that its pixels had minimal defocus, while there was considerable blur in the second image. The depth of each pixel and thus its coordinates in 3D space were recovered by measuring the difference in blur. A more general DfD method in [18] uses two images that do not have to be captured with large depth-of-field.

Image blur can be modelled as a 2D convolution of a focused image with a PSF. By modelling the PSF as a downturn quadratic function, and representing it as a look-up-table of the convolution ratio, the corresponding image depth can be found. The method in [69] computes the convolutional ratio of sub-images using

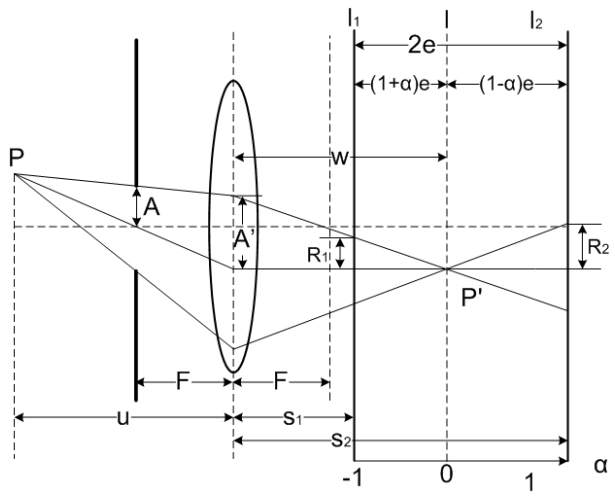


Figure 5.1: The telecentric DfD system.

regularisation, and searches the table iteratively to determine the depth for each sub-image. In [64], the PSFs are modelled as the derivatives of a Gaussian using moment and hyper-geometric filters. In [75] the intensity and depth values of every image pixel were modelled as an MRF and a MAP was maximised using simultaneous annealing to obtain the optimal depth estimation.

An orthogonal projector that spans the null-space of an image of a certain depth was used in [85]. For each depth, a number of different images were captured by a camera and the corresponding orthogonal projector was created, not requiring any PSF. Each projector was multiplied with each sub-image, and the depth corresponding to the projector with the minimal product gave the optimal depth estimation. Image segmentation was used to separate the object from the background together with a three-layered neural network in [88] to achieve high accuracy. However, the use of segmentation implies that the algorithm is only suitable for objects on a flat surface.

Recent DfD methods also include those based on coded aperture (e.g., [54; 96]) which customise the PSF by modifying the aperture shape, and most use a complex statistical model which is computationally expensive. In [163] the image blurring effect was described by oriented heat-flows diffusion, whose direction was determined from the local coherence geometry, and the diffusion strength corresponded to the amount of blur. Using this method, ridge-like artefacts on sharp edges were eliminated. In [164], an improvement on DfD was achieved by manipulating exposure time and guided filtering. In [165], improvement was achieved by minimising, via geometric optics regularisation, the information divergence between

the estimated and actual blurred images.

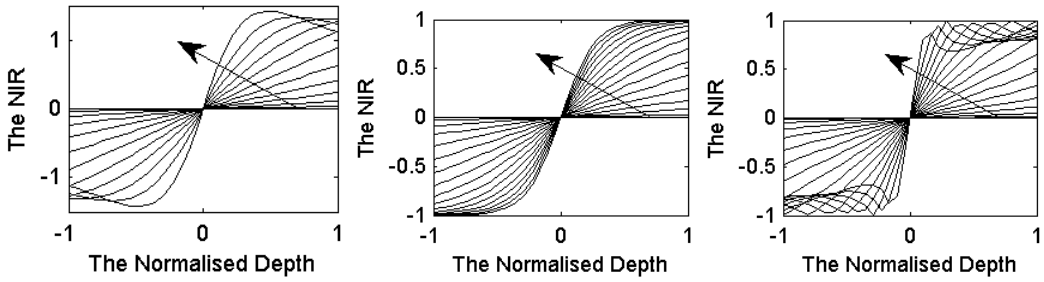
None of the above-mentioned methods achieves frequency independence without a complex statistical model or training/testing-based algorithm, i.e., the estimated depth is only related to the blur size rather than the pattern of the blurred object. A solution is to incorporate a frequency parameter into the PSF as suggested in [16]. For most existing methods, the depth is estimated from the ratio of two images with different degrees of blur at a particular frequency. In contrast, RO-DfD computes depth using the NIR, or the M/P ratio which is a function of both depth and frequency. The NIR is the ratio between the difference in the magnitude of two images at all frequencies (M for minus) and the sum of the magnitude of them (P for plus). Due to the complex and iterative optimisation procedure used for the RO design in [16], a simpler procedure has been proposed in [17] which also improves the depth estimation.

For convolved DfD, the telecentric optics described in [166] prevent an image magnification effect that limits DfD. The telecentric RO-DfD system is illustrated in Fig. 5.1. The light rays from an object point P pass through the telecentric aperture of radius A, and then through a circular part of radius A' on the lens. The focused image of P is at l, a position between the far-focused position l_1 and the near-focused image position l_2. The normalised depth α is -1 at l_1 and 1 at l_2. The distance between l and l_1 is $(1+\alpha)e$ and that between l and l_2 is $(1-\alpha)e$. A blur circle of radius R_1 and another of radius R_2 are formed at l_1 and l_2, respectively. The other parameters are denoted as follows: u is the distance between an object point and the lens; F is the focal length; s_1 is the distance between the lens and l_1; and s_2 is the distance between the lens and l_2.

Two images are captured: image $\mathbf{I}_1$ at l_1, and image $\mathbf{I}_2$ at l_2. When P is at far-focused position, its focused image P' is at l_1. Similarly, when P is at near-focused position, its focused image is at l_2. Hence the working range of the RO-DfD system is from the far-focused object position to the near-focused object position. The NIR or the M/P ratio is defined as [166]

$$\frac{\check{\mathbf{M}}}{\check{\mathbf{P}}}(f_r,\alpha) = \frac{\check{\mathbf{H}}_1(f_r,\alpha) - \check{\mathbf{H}}_2(f_r,\alpha)}{\check{\mathbf{H}}_1(f_r,\alpha) + \check{\mathbf{H}}_2(f_r,\alpha)}\ , \tag{5.2}$$

where $\check{\mathbf{H}}_1$ and $\check{\mathbf{H}}_2$ are respectively the OTFs of $\mathbf{I}_1$ and $\mathbf{I}_2$, α is the normalised depth which is -1 when the focused image is on l_1 and changes linearly from l_1 to l_2 so that it becomes 1 when it reaches l_2, and f_r is the radial frequency parameter in Hz. Using Eqn. (5.2) and the Pillbox PSFs, curves representing the NIR changing with α are shown in Fig. 5.2(a), each of which corresponds to a different discrete



(a) (b) (c)

Figure 5.2: (a) The Pillbox NIR varies with the normalised depth. (b) Gaussian NIR with k=0.4578. (c) Generalised Gaussian NIR with p=4 and k=0.5091. For each plot, the radial frequency of each curve increases in the direction of the arrow. All the frequencies are shown as their ranges are within [-1 1].

frequency. Each curve is modelled as a third order polynomial of NIR with respect to the depth, as described by [166]

$$\frac{\check{\mathbf{M}}}{\check{\mathbf{P}}}(f_r, \alpha) = \frac{\check{\mathbf{G}}_{p1}(f_r)}{\check{\mathbf{G}}_{m1}(f_r)}\alpha + \frac{\check{\mathbf{G}}_{p2}(f_r)}{\check{\mathbf{G}}_{m1}(f_r)}\alpha^3, \tag{5.3}$$

where the first order and third order coefficients are expressed as $\check{\mathbf{G}}_{p1}/\check{\mathbf{G}}_{m1}$ and $\check{\mathbf{G}}_{p2}/\check{\mathbf{G}}_{m1}$, respectively. The corresponding spatial filters (the ROs) are then computed. During run-time, these ROs are convolved with $(\mathbf{I}_1 - \mathbf{I}_2)(\mathbf{I}_1 + \mathbf{I}_2)$ in a specific order as in [16] followed by a coefficient smoothing procedure and a 7x7 PMF.

The advantages of RO-based DfD in [16] include: (a) ability to produce a dense depth map in real time with parallel hardware implementation; (b) the 3D reconstruction is invariant to textures; and (c) the depth error can be as low as 1.18%. Therefore a RO based method is feasible for real-time applications such as robotics and endoscopy. Its drawbacks are: (a) using Pillbox PSF is only valid when the lens induced aberrations and diffraction are small compared to the radius of the blur circle [167]; (b) a problematic filter design procedure used in the Levenberg-Marquardt algorithm (see Section 5.2.2); (c) lack of careful consideration on the NIR leads to the presence of some adverse frequency components. To address these drawbacks, we propose two RO-based methods: the Gaussian rational operator (GRO) based on the Gaussian PSF and the generalised Gaussian rational operator (GGRO) based on the generalised Gaussian PSF.

The novelties of the proposed methods are: (1) the GROs address the situation when the lens aberrations and diffraction are significant, producing smaller RMSE; (2) a practical calibration method finds the linear relationship between the

radius of the blur circle and the SD of the Gaussian or the generalised Gaussian PSF; (3) GGROs can be automatically configured to deal with the any levels of diffraction and aberrations; (4) the ROs are designed with a new and simpler method; (5) the pre-filter is redesigned to achieve better stability; and (6) an accurate and efficient DfD correction method is presented in Section 5.3 to reduce the severe circular distortion encountered in any DfD algorithm.

This chapter is organised as follows. Section 5.2 presents the proposed GRO and GGRO including the method for their calibration, a method for configuring GGRO, and the pre-filter. Section 5.3 presents the DfD correction procedure. The experiments on real images and discussion are presented in Section 5.4, and Section 5.5 concludes the chapter.

5.2 The Proposed Rational Operators

The design of our proposed ROs involves three steps. The first step determines either the Gaussian NIR or the Generalised Gaussian NIR. The second formulates the kernels of the ROs from the corresponding NIR. The third formulates the pre-filter for both types of ROs.

5.2.1 The Gaussian NIR

When the aberrations and diffraction of the camera lens used in DfD are significant when compared to the radius of the blur circle, the Gaussian PSF is a better model than the Pillbox PSF for modelling the image blur [167]. While the depth-related parameter of the Pillbox model is the radius of the blur circle, that of the Gaussian model is the SD. The SD is related to the radius of the blur circle R by $\sigma = kR$ [72], where k is measured for the camera system used. Hence, unlike the Pillbox ROs that are generic for every camera, the Gaussian ROs requires calibration for a specific camera system.

The 2D Gaussian PSF is [18]

$$\mathbf{H}(x, y) = \frac{1}{2\pi\sigma^2} \exp\left[-\frac{(x-\overline{x})^2 + (y-\overline{y})^2}{2\sigma^2}\right], \tag{5.4}$$

where $x - \overline{x}$ and $y - \overline{y}$ are the distances between a point at (x, y) and the reference point $(\overline{x}, \overline{y})$, which is the centre of the PSF. The corresponding OTF is the Fourier transform of the PSF, i.e.,

$$\check{\mathbf{H}}(u, v) = \int\int \frac{1}{2\pi\sigma^2} \exp[C_1] \exp[-jux - jvy]\, dx\, dy, \tag{5.5}$$

where $C_1 = -\frac{(x-\overline{x})^2+(y-\overline{y})^2}{2\sigma^2}$. Rewriting the inner integral of Eqn. (5.5) as a quadratic function of x, and using the quadratic exponential integration formula gives

$$\check{\mathbf{H}}(u,v) = \int C_2 \exp[C_3 + C_4] dy, \tag{5.6}$$

where $C_2 = \frac{1}{\sqrt{2\pi\sigma^2}}$, $C_3 = \frac{1}{2}\left[\frac{1}{\sigma}\overline{x} - ju\sigma\right]^2$, and $C_4 = -\frac{1}{2\sigma^2}\overline{x}^2 - \frac{1}{2\sigma^2}(y-\overline{y})^2 - jvy$. Similarly, rewriting the integral as a quadratic function of y and using the quadratic exponential integration formula gives

$$\begin{aligned} \check{\mathbf{H}}(u,v) &= \exp\left[-\frac{1}{2}\sigma^2(u^2+v^2) - j(\overline{x}u + \overline{y}v)\right] \\ &= \exp\left[(-\frac{1}{2}\sigma^2(u^2+v^2)\right] \exp[-j(\overline{x}u + \overline{y}v)]. \end{aligned} \tag{5.7}$$

In polar coordinates, the OTF is

$$\check{\mathbf{H}}(r,\theta) = r\exp[-j\theta], \tag{5.8}$$

where the magnitude $r = \exp[-\frac{1}{2}\sigma^2(u^2+v^2)]$, and the angle $\theta = \overline{x}u + \overline{y}v$. Assuming the OTF to be circular symmetric, it does not depend on angles and thus

$$\check{\mathbf{H}}(u,v,\sigma) = \exp\left[-\frac{1}{2}\sigma^2(u^2+v^2)\right]. \tag{5.9}$$

σ is related to the blur circle radius R by [72]

$$\sigma = kR, \tag{5.10}$$

where k is a camera constant obtained by measurement. In this chapter k is obtained as follows:

1. Place a flat test pattern, i.e. a flat surface fully covered by a piece of sandpaper, in front of the camera and perpendicular to the optical axis.

2. Focus the camera by computing its Fourier transform, where the image with the largest high-frequency magnitude is chosen as the sharpest (i.e., most focused). Denote the position of the camera sensor as l_1 (i.e., the far-focused sensor position) and the image as $\mathbf{I}_1$ (i.e., the far-focused image).

3. Move the sensor along the optical axis and away from l_1 to the position of l_2, such that $l_2 - l_1 = 2e$, where e is a constant as shown in Fig. 5.1. Capture the image as $\mathbf{I}_2$ (i.e., the near-focused image) at l_2 (i.e., the near-focused position).

Note that $\mathbf{I}_1$ is in focus and $\mathbf{I}_2$ is blurred by a radius $R = (1+\alpha)e/(2F_e)$.

4. Convolve image $\mathbf{I}_1$ with a Gaussian PSF of SD σ_t to generate image $\mathbf{I}'_{2t}$, i.e., $\mathbf{I}'_{2t} = \mathbf{I}_1 * G(\sigma_t)$. The mean square error is $\vec{\epsilon}(t) = \text{mean}(\mathbf{I}_2 - \mathbf{I}'_{2t})^2$.

5. Repeat step (4) with a number of σ_t values. The estimated SD is given by $\sigma_1 = \arg\min_{\sigma_t} \vec{\epsilon}(t)$.

6. Repeat steps 1-5 by making $\mathbf{I}_2$ in focus and $\mathbf{I}_1$ blurred to get another approximation of the SD σ_2. The final estimated SD is $\sigma = (\sigma_1 + \sigma_2)/2$, and k is σ/R. This is because when the image is in focus on $\mathbf{I}_1$, the SD of the blur circle on $\mathbf{I}_2$ is the same as that on $\mathbf{I}_1$ when the image is in focus on $\mathbf{I}_2$.

Using trigonometric similarity in Fig. 5.1 gives

$$\frac{2R_1}{(1+\alpha)e} = \frac{2A'}{w},$$

where the effective f-number F_e is $w/(2A')$, so that

$$\begin{aligned} \frac{2R_1}{(1+\alpha)e} &= \frac{1}{F_e} \\ \Rightarrow R_1 &= \frac{(1+\alpha)e}{2F_e}. \end{aligned} \tag{5.11}$$

Substituting Eqn. (5.11) in Eqn. (5.10) and then in Eqn. (5.9) gives the far-focused OTF:

$$\check{\mathbf{H}}_1(u,v,\alpha) = \exp\left[-\frac{1}{2}\left(\frac{k(1+\alpha)e}{F_e}\right)^2 (u^2+v^2)\right]. \tag{5.12}$$

The near-focused OTF is similarly obtained as

$$\check{\mathbf{H}}_2(u,v,\alpha) = \exp\left[-\frac{1}{2}\left(\frac{k(1-\alpha)e}{F_e}\right)^2 (u^2+v^2)\right]. \tag{5.13}$$

Note that the maximum radius of the blur circle is chosen to be 2.703 pixels, or 20 micrometres since the sensor size is 7.4×7.4 micrometres per pixel. The sensor is the same as the one used in [17], i.e. an AVT Guppy FireWire CCD sensor. Thus according to Eqn. (5.11), the f-number (focal length divided by aperture diameter) controls the sensor separation $2e$ which in turn controls the working range, where by definition $\alpha = 1$ when the radius is maximised on l_1. This is explained as follows. It is noted from Eqn. (5.1) that a larger $2e$ results in a larger difference in image position w between the far-focused and near-focused conditions. This leads to a larger difference between the far and near object positions, i.e. the working

range. This means a larger f-number results in a larger working range and vice versa. However, a larger f-number will generate a higher noise level in the data due to higher level of diffractions [18]. Notably, only the choices of maximum radius and the f-number will change the experimental set-up, while the value e is determined by the radius and the f-number selected according to Eqn. (5.12).

An example Gaussian NIR graph is shown in Fig. 5.2(b), where $k = 0.4578$ as measured for the camera system used to obtain the subsequently reported results. Unlike the Pillbox NIR shown in Fig. 5.2(a), even the high-frequency curves of the Gaussian NIR increase monotonically with depth. However, this does not mean they will not generate any adverse effects. A discussion of this is presented in Section 5.2.4.

5.2.2 The Generalised Gaussian NIR

A 1D generalised Gaussian PSF, generated using a combination of the Pill-box and Gaussian models with an adjustable parameter p is proposed in [168]. When $p = 2$, it is equivalent to a Gaussian PSF, and when $p \to \infty$ it is equivalent to the Pillbox PSF. The 1-D generalised Gaussian PSF is [168]

$$\mathbf{H}(x) = \frac{p^{1-\frac{1}{p}}}{2\sigma\Gamma\left(\frac{1}{p}\right)} \exp\left[-\frac{1}{p}\frac{|x-\overline{x}|^p}{\sigma^p}\right], \tag{5.14}$$

where $\Gamma()$ is the Gamma function, σ is the SD such that $\sigma = kR$, x is the spatial index and $\overline{x}$ is the centre of the PSF. The method for determining k in Section 5.2.1 is effectively a 1-D search, and the calculated value of p, i.e. p_{est}, is obtained here by a 2-D search, with each loop being an 1-D search of k for a test value of p. This process is described as follows:

1. Beginning with a very small value of p (e.g. p=1), find the mean square error $\vec{\epsilon}(t)$ for every attempted $\sigma(t)$. Store the minimum of ϵ as $\vec{\epsilon}_m(p)$, and the corresponding k value as $\vec{k}_m(p)$.

2. Repeat the previous step with a slightly larger p value (e.g., 0.1 larger), until the minimum of $\vec{\epsilon}_m(p)$ is identified. The calculated p value is given by $p_{est} = \arg\min_p \vec{\epsilon}_m(p)$, and the corresponding k value is given by $k_{est} = \vec{k}_m(p_{est})$

   Denoting $C_5 = 2\sigma\Gamma\left(\frac{1}{p}\right)p^{\frac{1}{p}-1}$, Eqn. (5.14) becomes

$$\mathbf{H}(x) = \frac{1}{C_5} \exp\left[-\frac{1}{p}\frac{|x-\overline{x}|^p}{\sigma^p}\right], \tag{5.15}$$

The equivalent 2D PSF is

$$\mathbf{H}(x,y) = \frac{p^{2-\frac{2}{p}}}{4\sigma^2\Gamma^2\left(\frac{1}{p}\right)} \exp\left[-\frac{1}{p}\frac{|(x-\overline{x})^2+(y-\overline{y})^2|^{\frac{p}{2}}}{\sigma^p}\right], \tag{5.16}$$

The OTF can be derived by the 2D Fourier transform of Eqn. (5.16), but no closed formed OTF can be obtained. To address this problem, numerical methods such as Adaptive Simpson Quadrature [169] can be used to calculate the OTF. FFT is a possible alternative but it should be used with care. This is because when the SD is small, it fails to produce accurate OTF values, whereas the Fourier transform by numerical integration almost always give accurate results but is much slower to compute. Fig. 5.2(c) shows an example of the NIR graph generated using p=4 and k=0.5091. As a result of the Pillbox-Gaussian combination, each curve in the NIR graph has a smaller range than the Pillbox NIR while the high-frequency curves do not increase monotonically with depth.

5.2.3 Design of the Rational Operators Kernels

Sections 5.2.1-5.2.2 present the methods for finding the NIRs as given by Eqn. (5.3). The first and third order coefficients can be found by performing a least squares fit of the NIR. The procedure used in [16] first sets $\check{\mathbf{G}}_{p1}$ as the MFR of a Log of Gaussian (LOG) band-pass filter and then derives the other terms accordingly. The proposed method adapts this procedure by using a different method to estimate the corresponding spatial filters, i.e., the ROs.

In [16] and [17], the frequencies within [-0.5 0.5] Nyquist are divided into 32 discrete portions to allow the polynomial coefficients to be found. Since the ROs operate in the spatial domain, their MFRs must be converted to the corresponding spatial filters. Since depth estimation is based on spatial filtering, i.e. several convolutions between the input images and the 2D filters (or effectively matrices), no filter is allowed to contain imaginary values. Therefore, inverse Fourier transform producing imaginary values is not applicable. In [16], the two operators $\mathbf{G}_{m1}$ and $\mathbf{G}_{p2}$ are acquired by the Levenberg-Marquardt algorithm, where only one cost function is used to find both $\mathbf{G}_{m1}$ and $\mathbf{G}_{p2}$. However, $\check{\mathbf{P}}(u,v;\alpha)$ in [16] is assumed to be the MFR of a fractal image, which corresponds to Brownian motion. This may not be a good approximation since $\check{\mathbf{P}}(u,v;\alpha)$ is the MFR of the sum of the two input images, and thus cannot be a constant. Moreover, our experiments show that the weighting factor used in their method significantly affects the optimisation result.

We solve this problem with a different cost function, which is the difference

between the left hand side and right hand side of Eqn. (5.2), i.e., to find the ROs, the MFRs of which best fit the NIR curves (this is also the final target of the ROs' design in [16; 17]). Denoting

$$\frac{\check{\mathbf{M}}'}{\check{\mathbf{P}}'}(u,v,\alpha) = \frac{\check{\mathbf{G}}'_{p1}(u,v)}{\check{\mathbf{G}}'_{m1}(u,v)}\alpha + \frac{\check{\mathbf{G}}'_{p2}(u,v)}{\check{\mathbf{G}}'_{m1}(u,v)}\alpha^3, \tag{5.17}$$

where (u,v) is the frequency index such that $f_r = \sqrt{u^2+v^2}$, $\check{\mathbf{G}}'_{p1}$, $\check{\mathbf{G}}'_{m1}$ and $\check{\mathbf{G}}'_{p2}$ are the MFRs of the ROs $\mathbf{G}_{p1}$, $\mathbf{G}_{m1}$ and $\mathbf{G}_{p2}$ respectively, the cost function is given by:

$$\epsilon^2 = \sum_{u,v,\alpha}\left(\left|\frac{\check{\mathbf{M}}}{\check{\mathbf{P}}}(u,v,\alpha)\right| \cdot |\check{\mathbf{Q}}(u,v)| - \left|\frac{\check{\mathbf{M}}'}{\check{\mathbf{P}}'}(u,v,\alpha)\right|\right)^2, \tag{5.18}$$

where $\frac{\check{\mathbf{M}}}{\check{\mathbf{P}}}$ is the NIR calculated using Eqn. (5.2) and $\check{\mathbf{Q}}$ is MFR of the pre-filter used to filter the input images before DfD computation (see Section 5.2.4). Thus the ROs are estimated as

$$[\mathbf{G}_{p1}, \mathbf{G}_{m1}, \mathbf{G}_{p2}] = \arg\min_{\mathbf{G}_{p1},\mathbf{G}_{m1},\mathbf{G}_{p2}} \epsilon^2. \tag{5.19}$$

Therefore, all three ROs are estimated simultaneously, without approximating $\check{\mathbf{P}}(u,v;\alpha)$. In addition, the weight is set to be the same for every frequency index without the need to compute it as a specific matrix as in [16]. This is because every frequency component has even contribution to minimise the overall cost function of Eqn. (5.18). This results in the right hand side of Eqn. (5.18) not having a denominator.

Eqn. (5.18) and (5.19) can be implemented with any non-linear optimisation algorithm such as the Gauss-Newton algorithm, gradient descent algorithm or Levenberg-Marquardt algorithm. A crucial step is the initialisation of ROs $\mathbf{G}_{m1}$ and $\mathbf{G}_{p2}$. As mentioned earlier, $\mathbf{G}_{p1}$ is initialised as a LOG filter, then its MFR $\check{\mathbf{G}}_{p1}$ is calculated. $\check{\mathbf{G}}_{m1}$ and $\check{\mathbf{G}}_{p2}$ are then computed with the estimated first and third order coefficients. The Parks-McClellan FIR filter design algorithm and McClellan transformation are then used to find the initial guess for $\mathbf{G}_{m1}$ and $\mathbf{G}_{p2}$ [170; 171].

The procedure to initialise $\mathbf{G}_{m1}$ and $\mathbf{G}_{p2}$ is as follows: 1) Generate a 1D vector $\vec{f}_{tar}$ by sampling 6 values of the RO's MFR ($\check{\mathbf{G}}_{m1}$ or $\check{\mathbf{G}}_{p2}$) across 6 equally spaced frequency indices from 0 to 0.5 Hz.; 2) Use $\vec{f}_{tar}$ and the corresponding frequency indices vector as input to the Parks-McClellan algorithm to find the 1D spatial filter $\vec{f}_{1d}$; 3) Use $\vec{f}_{1d}$ as input to the McClellan Transformation algorithm to get the corresponding 2D filter, which is the initial guess for the RO ($\mathbf{G}_{m1}$ or $\mathbf{G}_{p2}$).

5.2.4 The pre-filter

The purpose of the pre-filter is to remove the frequency components which adversely affect the accuracy and stability of the 3D reconstruction. To design the pre-filter appropriately, these adverse frequency components are identified using a NIR graph, e.g., as is shown in Fig. 5.2(b), as follows. For each frequency, the variable represented by the vertical axis in Fig. 5.2(b) (i.e., the NIR) is used as the input argument, and the output is the variable represented by the horizontal axis.

The pill-box PSF was used in [16; 17] that had a depth response as shown in Fig. 5.2(a). Due to non-monotonicity of the high frequency components, one input NIR value corresponded to more than one output depth values. This created an ambiguity during depth measurement. Thus, their pre-filters aimed to remove the high-frequency components. However, we have discovered that low-gradient components also introduced significant errors.

Note that if the input varies even slightly, the resulting output depth will be highly unstable. For example, for the second lowest frequency of 0.1963 Hz, represented by the curve just next to the horizontal line, if the input varies within ± 0.1, the output will vary across the entire depth range. Similarly, some high-frequency components also suffer from this problem since some parts of them are almost horizontal. Thus, a lack of consideration on this problem can result in significant depth variation, requiring larger coefficient smoothing and larger median filtering kernels (our experiments show that a 3x3 median filtering kernel is typically enough to remove most high-frequency noise, but it can be larger if the noise power increases significantly) which slow down the processing.

The frequency components containing a large proportion of low-gradient or negative gradient (corresponding to non-monotonicity) should thus be removed. This is achieved by minimising the cost function

$$\epsilon^2 = \sum_{u,v} \left(\frac{\check{\mathbf{Q}}(u,v) - \check{\mathbf{Q}}'(u,v)}{\mathbf{Z}(u,v)} \right)^2 , \tag{5.20}$$

where $\check{\mathbf{Q}}$ is the target MFR (to be determined later), $\check{\mathbf{Q}}'$ is the MFR of the estimated pre-filter and $\mathbf{Z}$ is the weight matrix (to be determined later). In order to obtain a pre-filter that effectively remove the components that introduce significant instability, which corresponds to the NIR curves that contain parts with small or negative gradient, the corresponding elements of $\check{\mathbf{Q}}$ need to be set small enough while maintaining $\check{\mathbf{Q}}$ to be smooth. In addition, the weight matrix needs also be set in a way that the optimisation is focused on obtaining small values for the adverse

frequency components.

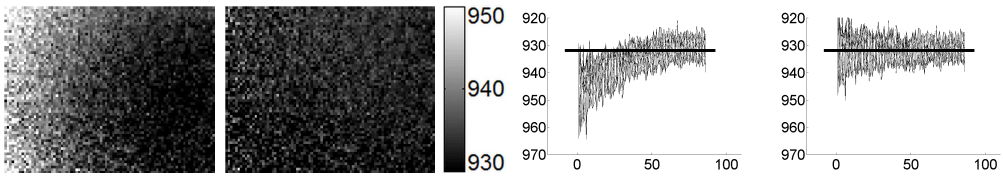
The optimal $\check{\mathbf{Q}}$ is obtained as follows:

1. Compute the gradients along each curve of the NIRs at different depths;
2. Find the smallest gradient for each curve which corresponds to a unique radial frequency f_r;
3. Each element of $\check{\mathbf{Q}}$ is assigned by the smallest gradient value (0 if negative) of corresponding curve with the same radial frequency $f_r = \sqrt{u^2 + v^2}$, resulting in small values for the adverse components while enforcing smoothness of the filter;
4. $\check{\mathbf{Q}}$ is then divided by its maximum value to give a unity gain filter;
5. $\mathbf{Z}(u, v)$ is set to be $\check{\mathbf{Q}}$ incremented by a small value (e.g., 0.05) so that it does not contain zero, resulting in the optimisation focused on the adverse components. The optimisation can be implemented by one of the non-linear optimisation methods, initialised with the same method as the ROs as presented in Section 5.2.3.

5.3 DfD Correction Method

During tens of experiments using three different lenses (a professional 50 mm lens, a 35 mm lens, and a widely available 1 mm Webcam lens), a common and severe problem was observed using all the DfD algorithms mentioned in Section 5.1. Such a problem is illustrated in Fig. 5.3. Here a hill-like result is generated from a flat plane that is perpendicular to the optical axis. The hill is more or less circularly symmetric which is similar to the surface of a common lens. Also its centre deviates from the centre of the map due to the focus adjustment during the experiment.

An analysis of the experiment reveals the following concern. When an image is defocused by moving the sensor, it cannot be defocused evenly across the image, where some parts are more blurred than the others (the shape of the defocused pattern is similar to the distorted depth map in Fig. 5.3(a), where the varying grey levels denote the uneven surface). In addition, the small f-number used results in a narrow depth of field, making the problem easily observed. In other words, the distorted blur field leads to a distorted depth map. Note that both results are generated with the same scale using Watanabe's [16] DfD method. The expected depth map should be flat with a value of 933 mm.



(a) (b) (c) (d) (e)

Figure 5.3: An example DfD correction problem. Grey-coded depth maps of a flat surface: (a) without correction and (b) after correction. (c) The grey-bar of (a) and (b). Mesh plots: (d) the side view of (a); and (e) the side view of (b), where the horizontal units are in pixel and the vertical units are in mm. The horizontal lines in (d) and (e) represent the expected depth.

Generally, it is possible to correct either the input images or the output depth map. The former is considered not practical due to the complex measurement involved, so the proposed method is based on correcting the output depth maps. An immediate thought is to subtract any depth map by a calibration pattern (i.e., the depth map generated for the far-focused object position) and then minus the expected depth of that pattern (the depth offset). However, our experiments show that the appropriate calibration pattern is depth dependent. Thus, a closed-form solution is needed to find the correction factor for each element of the pattern.

The offset $\mathbf{T}$ at the location defined by the Cartesian coordinates (x, y) is modelled as a third order polynomial of the coordinates x and y and the corresponding raw depth $\mathbf{U}_{raw}(x, y)$, i.e.,

$$\mathbf{T}(x, y) = \vec{c}_1(1) + \sum_{i=2}^{4} \vec{c}_1(i)x^{i-1} + \sum_{i=5}^{7} \vec{c}_1(i)y^{i-4} + \sum_{i=8}^{10} \vec{c}_1(i)(\mathbf{U}_{raw}(x, y))^{i-7}. \quad (5.21)$$

Third order is chosen since our experiments showed that first and second order led to lower accuracy, and fourth order and above did not produce significant improvement while increasing the computational cost.

Samples of $\mathbf{T}$ and $\mathbf{U}_{raw}$ are obtained by capturing a pair of DfD images of a test flat surface at a location within the working range, e.g., $\mathbf{T}$ is -1, when the surface is at the far-focused object point, and $\mathbf{U}_{raw}$ is thus the corresponding computed depth map. The coefficient vector $\vec{c}_1$ is in turn obtained using least squares fit. The correction factor is given by

$$\mathbf{U}_c(x, y) = \vec{c}_2(1) + \sum_{i=2}^{4} \vec{c}_2(i)x^{i-1} + \sum_{i=5}^{7} \vec{c}_2(i)y^{i-4} + \sum_{i=8}^{10} \vec{c}_2(i)(\mathbf{T}(x, y))^{i-7}, \quad (5.22)$$

where $\mathbf{U}_c(x, y)$ is equivalent to $\mathbf{U}_{raw}(x, y)$, and $\mathbf{T}(x, y)$ is found by the previous step

and the coefficient vector $\vec{c}_2$ is similarly computed with a least square fit. Finally, to correct any depth result at a specific location (x, y), the following procedure is required:

1. The location coordinate (x, y) and the uncorrected result $\mathbf{U}_{raw}$ are substituted into Eqn. (5.21) to get the offset $\mathbf{T}(x, y)$.

2. The location coordinate (x, y) and $\mathbf{U}_c(x, y)$ are substituted into Eqn. (5.22) to compute the correction factor $\mathbf{U}_c(x, y)$.

3. The corrected depth is thus estimated by $\mathbf{U}_{corrected}(x, y) = \mathbf{U}_{raw}(x, y) - \mathbf{U}_c(x, y) + \mathbf{T}(x, y)$.

An example of the corrected depth map is shown in Fig. 5.3(b) and (e), where the working range is within [886.8 933] mm away from the lens. To show that the correction method works with any RO-DfD, the depth results are generated using Watanabe's ROs [16]. While the local noise level is not visibly magnified, the general shape of the hill has been restored to be flat. A typical correction of a depth map only accounts for 4-8% of the total RO-DfD computational time, hence it is suitable for real-time applications.

5.4 Experiments

Since the proposed RO-DfD methods are based on different PSFs than those used in [16] and [17], experiments with simulated images are not of any use because they are generated with a specific PSF. Thus we only discuss the depth results using real images. In addition, since the ROs in the Raj's method [17] produce better results than the ROs in [16], the comparison is made between the state-of-the-art Raj's method and the proposed method.

A professional 50 mm lens is used with a telecentric aperture whose diameter is 12.8 mm. The f-number F_e is thus obtained by dividing the focal length $F = 50$ mm by the aperture diameter $2A = 12.8$ mm, which is 3.9063. The side-length of each CCD sensor element is 7.4 μm. The maximum radius of the blur circle is 2.703 pixels. Hence the radius of the blur circle is $2.703 \times 7.4^{-3} = 0.0200$ mm The far-focused object position u_1 is set 933 mm away from the lens.

The working range is computed as follows:

1. When radius of the blur circle on l_1 is 0 the scene is far-focused such that $s_1 = w$, substituting $u = u_1 = 933$ mm and $F = 50$ mm into Eqn. (5.1) gives the far-focused sensor-lens separation $s_1 = w = 52.8313$ mm.

2. When radii of the blur circles on l_1 and l_2 are respectively of 0.02000 (the maximum) mm and 0 (the minimum), substituting $R_1 = 0.0200$ mm, $\alpha = 1$ (this is true when R_1 reaches its maximum) and $F_e = 3.9063$ into Eqn. (5.11) gives the sensor separation $2e = 0.1563$ mm, such that the near focused sensor-lens separation is $s_2 = s_1 + 2e = 52.9876$ mm.

3. When radius of the blur circle on l_2 is 0 as in step 2, substituting $w = s_2 = 52.9876$ mm and $F = 50$ mm into Eqn. (5.1) gives the near-focused object position $u_2 = u = 886.8$ mm. Thus the working range is [886.8 933] mm away from the camera.

The input image resolution is 640×480. The proposed RO-DfD method is implemented on a computer with an Intel Pentium Dual-Core 2.16 GHz processor. A flat surface covered with sandpaper is used as the test pattern for evaluation. It is set perpendicular to the optical axis and shifted from the far-focused object position to the near-focused position, with successive locations separated equally by 46.18/23 mm. 24 pairs of images are thus captured and 24 depth maps computed.

A 7×7 window is used for coefficient smoothing. For comparison with Raj's method, the RMSE is measured between the estimated depth map and the ideal flat depth map for each pair of inputs. A set of 24 measurements is obtained for each one of Raj's method, GRO and GGRO. Here the k value for GRO is estimated as 0.4578 using the method presented in Section 5.2.1; p and k for GGRO are respectively estimated as 1.8 and 0.4802 using the method presented in Section 5.2.2.

Fig. 5.4 shows the comparisons of RMSE between Raj's ROs [17] and the ROs (i.e., the Gaussian-based and the Generalised Gaussian-based) of the proposed method. Note that the proposed DfD correction method only deals with the general shape of the depth map, e.g., it restores a noisy hill to a noisy flat surface, maintaining the local noise level as illustrated in Fig. 5.3. In addition, experiments show that the RMSE before correction is dominated by the general shape and depends on the local noise level after correction. Furthermore, the general shape distortion is worse for the smaller depths than the larger ones, while the local noise level is worse for the larger depths than the smaller ones. Thus, for the uncorrected results shown in Fig. 5.4 the RMSE is higher for the smaller depths than the larger depths, while the RMSE after correction is higher for larger depths than smaller depths.

Despite all these general RO-DfD problems, the results before correction of the proposed ROs are more accurate than using Raj's ROs for depths larger than 890.8 mm while GGROs produce the least overall RMSE. Moreover, for the results after correction, the proposed methods outperform the Raj's method across the entire working range.

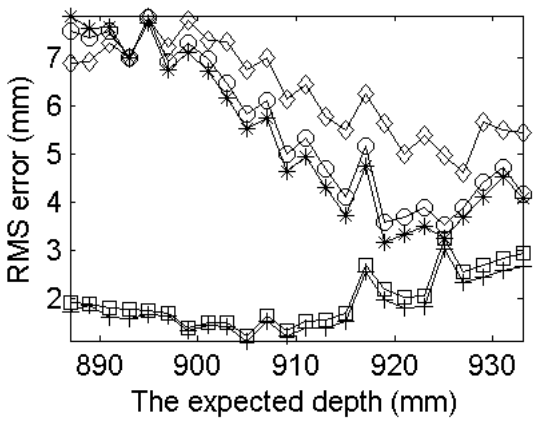


Figure 5.4: Comparison of RMSE of Raj's and the proposed methods. Key: ◊ - Raj; ◯ - GRO uncorrected; ∗ - GGRO uncorrected; □ - GRO corrected; and + - GGRO corrected.

Fig. 5.5 show the four real test scenes comprising different shaped objects for evaluating the performance of the proposed method qualitatively. Note that our experiments show that 3x3 PMF is typically sufficient to remove most noise while preserving the depth discontinuity for scenes that are not reflective and having visible textures. However the results in the third column are generated using 5x5 PMF instead of 3x3 PMF. This is because the associated scene of the fingers is considerably more reflective than the others and does not have sufficient textures to enable DfD to work with blurring. For a scene with no or little texture, there is little different in blur with the two captured images. When the pre-filter fails to remove some adverse frequency components, e.g., low frequencies corresponding to little texture, high noise level is perceived (see Section 5.2.4). For the staircase and cone scene in the first column, there is less noise in the reconstructed surface and the circular distortion is eliminated. The wooden temple results in the second column test the algorithms with complex objects that do not span the full working range, where the expected range of the temple is significantly lower than than the working range. Similar results are obtained. The third column shows the advantage of the smoothing nature of the GRO, which removes much of the low-frequency components (due to the reflective hand surface) while preserving the depth discontinuity. The

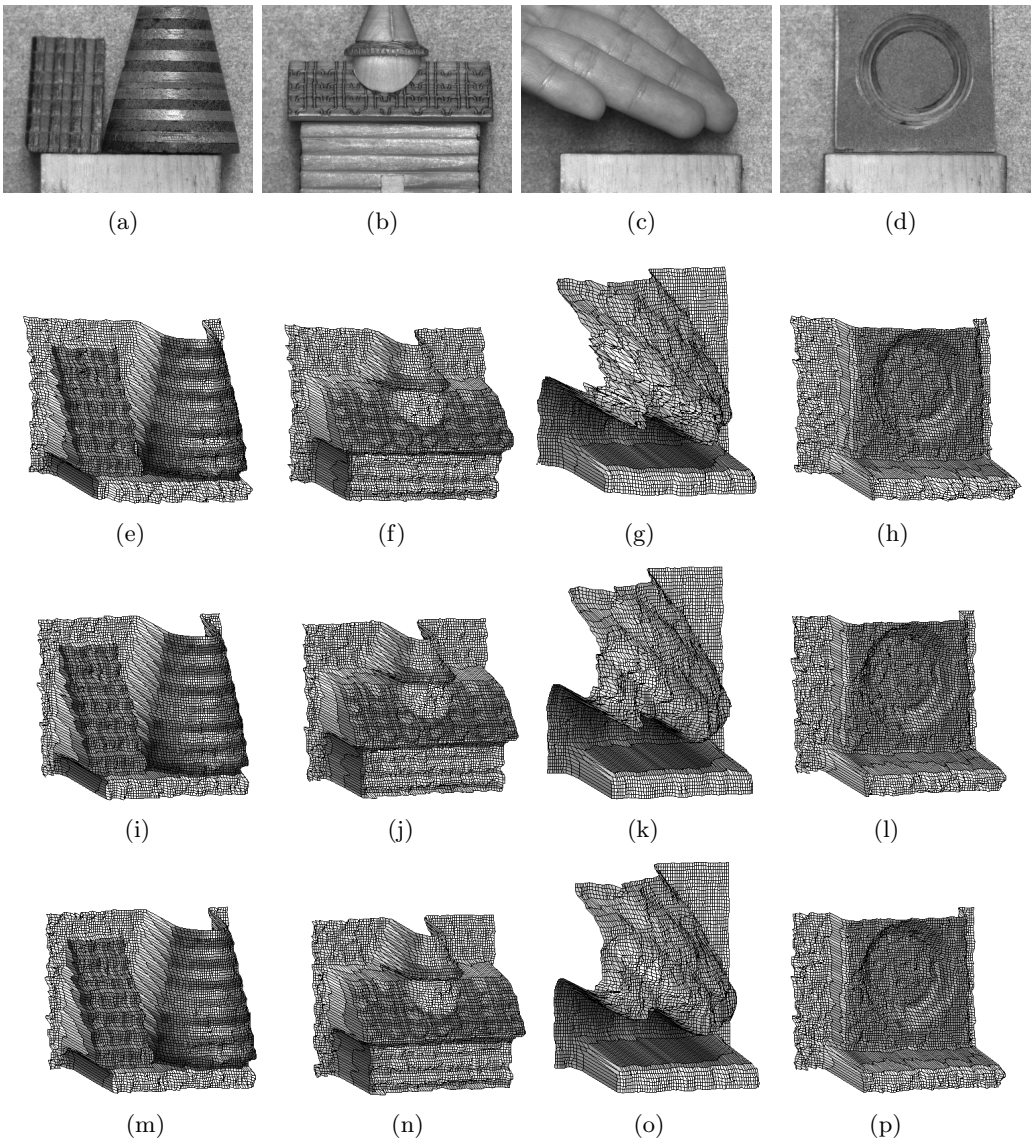


Figure 5.5: Mesh-plots of the results using Raj's and the proposed methods with the test objects. Row 1 - the test scenes. Mesh plots of 3D scene reconstructions using: row 2 - Raj's method; row 3 - GRO; and row 4 - GGRO. Note that the results in column 3 are generated using 5x5 PMF whereas the others are obtained using 3x3 median filtering.

object with a conical depression shown in the last column is used to demonstrate the ability of the DfD correction method to cope with multiple depths. Here not only the background is corrected, the foreground objects are also corrected.

Table 5.1 and 5.2 show the numerical comparisons based on the four sets of results. In these tables, the acronym SC, TP, HD and CD respectively stand for Staircase and Cone, Temple, Hand and Conical depression that are shown in Fig. 5.5(a)-(d), respectively.

Table 5.1: The mean RMSE of all the flat surfaces in the reconstruction results of the test scenes in Fig. 5.5, before and after correction. All units are in mm.

	SC	TP	HD	CD
Raj	6.46	7.23	6.59	8.59
GRO before	4.59	5.43	6.03	8.02
GGRO before	4.17	5.14	6.19	7.88
GRO after	3.90	2.91	4.35	4.20
GGRO after	3.63	2.89	4.11	4.19

Table 5.2: The mean SD of all the flat surfaces in the reconstruction results of the test scenes in Fig. 5.5, before and after correction. All units are in mm.

	SC	TP	HD	CD
Raj	3.14	3.21	11.78	3.20
GRO before	2.49	2.75	6.88	2.45
GGRO before	2.21	2.36	6.32	2.29
GRO after	3.00	3.12	7.68	3.14
GGRO after	2.79	2.96	7.23	2.95

Table 5.1 shows the comparison of the RMSE between Raj's results and the proposed ones, generated from the four sets of results. The RMSE for each test scene is measured as follows. For each flat surface that is perpendicular to the optical axis, the RMSE is calculated between the estimated depth and the actual depth which is measured with a vernier calliper. The average of these RMSEs is used as the RMSE for the scene. The table shows that both GRO and GGRO produce smaller RMSE than Raj's method, while GGRO produces the smallest RMSE. Moreover, the correction method manages to reduce the RMSE significantly.

Table 5.2 compares the noise levels between Raj's results and those of the proposed methods, which are measured by the average SD of the flat surfaces of the four sets of results. For example, for the scene in the first column of Fig. 5.5, the SD is evaluated for each step surface of the staircase, the surface of the wooden chunk at the bottom, and the background. The average of these SDs is used to indicate the noise level of the depth result. The table shows that both GRO and GGRO

produce less noise than Raj's method, while GGRO generates the least noise. In addition, the correction method has little influence on the noise level.

In terms of the computing time, the proposed depth estimation and depth correction only take typically 0.35 second. If a parallel hardware implementation is used, a real-time DfD processing is also achievable. Therefore the proposed DfD method can be very useful for robotic and medical applications.

5.5 Summary

Lens aberrations and diffraction are two undesirable and unavoidable imperfection in image acquisition where 3D object reconstruction techniques including DfD suffers. The former occurs when sub-quality lenses are used and the latter occurs when small aperture is used or the image centre is misaligned with the optical axis. This chapter presents two novel RO-DfD methods, one using Gaussian PSF and another using Generalised Gaussian PSF. The GROs cope well in situations where the lens aberrations and diffraction are significant compared to the blur radius. The GGROs can cope with any levels of aberrations and diffraction. In addition, the pre-filter is designed to take into account of the instability in the measurement of depth as well as its monotonicity. Moreover, the ROs are designed to speed up the filters generation process.

This chapter also presents a practical DfD correction method that addresses the circular lens distortion and misalignment between the image centre and the optical axis. Experiments on real images with both quantitative and qualitative results show that the GROs and the GGROs together with the proposed correction algorithm respectively produce more accurate results than the state-of-the-art Raj's ROs. Furthermore, the proposed methods are fast with the effective and efficient correction stage eliminating the hill-like depth map distortion generated by existing DfD methods.

The proposed DfD method works well because: 1) GROs use Gaussian model that is more suitable than the Pillbox model for defocusing that is dominated by aberrations and diffraction; 2) GGROs use generalised Gaussian model that generalises the RO-DfD method to deal with any levels of aberrations/diffraction; 3) Input frequency components are analysed using the NIR and removed with a pre-filter; 4) The ROs are designed with a simpler method achieving comparable accuracy than [16; 17]; 5) A simple DfD correction procedure is devised to eliminate the strong circular distortion originated from image acquisition.

Although the DfD method works effectively, the general shape of the NIR cannot be reproduced without error and the adverse frequency components cannot

be removed completely due to the small-sized broad-band ROs. Thus a possible future work is the use of coded aperture to produce a PSF suitable for small ROs. Moreover, the current implementation of the proposed method requires the sensor to be moved from one position to another with typical precision of 5 μm. One possible way to increase the precision is to use of a step motor or a batch of piezoelectric-electric materials to control the movement. If the size of the camera system is not an issue, two sensors and a half mirror could be used to remove the need to move the sensors.

Chapter 6

Correction Algorithms for Depth from Defocus

6.1 Introduction

Section 5.1 describes a number of DfD methods. One problem with these methods is that they assume that the blurring effect is uniform across an entire image, so that the same DfD framework can be used for every pixel or sub-image. However, this assumption is violated for most off-the-shelf camera lenses [172; 173], which causes the depth map of a flat surface to be approximately a 3D quadratic surface, i.e., the depth map is severely distorted by the surface peripheral that is of considerably high curvature. Using a 50 mm professional lens, a 35 mm professional lens and a common 1 mm web-cam camera, our experiments show that the quadratic surface is dependent on depth.

A correction method based on two-step least squares fit is incorporated in our DfD method presented in Chapter 5 and published in [45]. First, the depth offset is modelled as a third order polynomial of the x coordinate, y coordinate and the input (distorted) depth value, i.e.

$$\mathbf{D}(x,y) = \vec{c}_1(1) + \sum_{i=2}^{4} \vec{c}_1(i)x^{i-1} + \sum_{i=5}^{7} \vec{c}_1(i)y^{i-4} + \sum_{i=8}^{10} \vec{c}_1(i)(\mathbf{U}_{raw}(x,y))^{i-7},$$

where the set of coefficients c_1 is found by least squares fit, and $\mathbf{U}_{raw}$ is the distorted depth. Second, the corrected depth is modelled as another third order polynomial

of the x coordinate, y coordinate and the depth offset value, i.e.,

$$\mathbf{U}_c(x,y) = c_2(1) + \sum_{i=2}^{4} c_2(i)x^{i-1} + \sum_{i=5}^{7} c_2(i)y^{i-4} + \sum_{i=8}^{10} c_2(i)(\mathbf{D}(x,y))^{i-7}, \qquad (6.1)$$

where the set of coefficients c_2 is found by another least squares fit. The corrected depth is finally computed using Eqn. (6.1). The method as published in [45] has not been shown to be suitable for other DfD methods. Also, we found that the two steps involved are unnecessary and produce accumulative errors. This is because the errors generated from the first fit in Eqn. (6.1) propagates to the second fit in Eqn. (6.1), which generates additional errors.

In this chapter, we propose two correction methods: CDC and CLSF. The first method works by obtaining a number of correction patterns that are used to cancel out the distortion. The second works by finding a map from the distorted depth value and its expected depth value by least squares fit. The other main contribution is that the methods also address the distortion problem which is spatially variant in terms of 3 dimensions: horizontal and vertical dimensions of the depth map and the depth of each pixel. Experiments are performed with four different DfD methods to demonstrate that the correction methods can potentially be adapted to all other DfD approaches.

This chapter is organised as follows. Section 6.2 introduces the distortion problem. Section 6.3 and Section 6.4 present CDC and CLSF, respectively. Section 6.5 proposes a post-processing algorithm which addresses the low-texture problem of the input images. Section 6.6 presents both quantitative and qualitative evaluation of the two correction methods. Finally Section 6.7 summarises the chapter.

6.2 The Depth-Variant Elliptical Distortion Problem

The grey levels of the grey-coded depth map of a flat surface without any distortion should be uniformly distributed. Fig. 6.1 illustrates the distortion problem using Subbarao's method [18]. The working range is set to be within $[933, 887]$ mm away from the lens. The depth map of a flat surface which is 933 mm away from the lens (i.e., furthest away) shown in Fig. 6.1(a) has very strong elliptical distortion, which is centred at the bottom right corner due to inaccurate optical alignment mentioned in Section 4.6. The distortion may be eliminated by subtracting the depth map from the depth map of a flat surface at the specific depth (i.e., the correction pattern), then adding the distance corresponding to that pattern (the offset), thus flattening

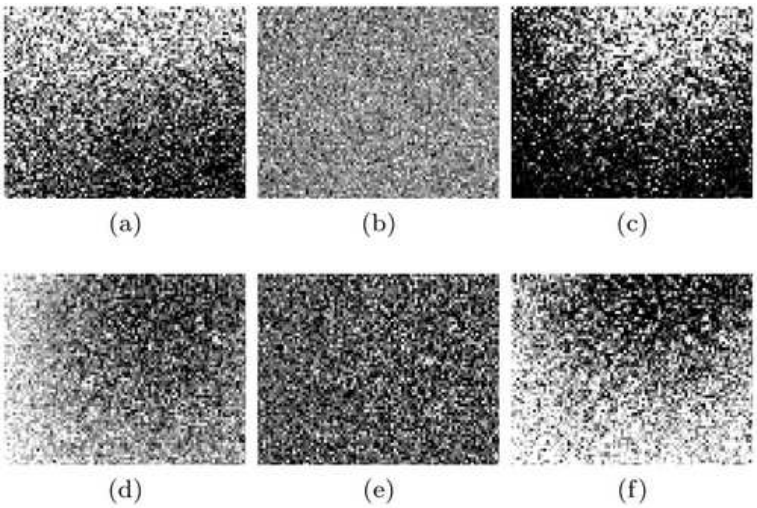


Figure 6.1: Grey-coded depth maps illustrating the distortion problem and the depth independence problem with Subbarao's method [18]: (a) the furthest flat surface with distortion; (b) surface (a) corrected using the furthest correction pattern; (c) surface (a) corrected using the nearest correction pattern; (d) the nearest flat surface with distortion; (e) surface (d) corrected using the nearest pattern; and (f) surface (d) corrected using the furthest pattern.

the general shape while correcting the depth. Note that in Fig. 6.1(a)-(c) the depth maps are plotted with the same scale, where the darkest and the brightest represent the depth of 925 mm 940 mm, respectively. Fig. 6.1(d)-(f) are also plotted with the same scale, where the darkest and the brightest represent 880 mm and 900 mm, respectively.

Fig. 6.1(b) shows that the surface in (a) is effectively corrected by the pattern at the furthest point, as seen by the relatively uniform distribution of the grey levels. It is assumed that the correction patterns within the working range have the same shape. Fig. 6.1(c) shows that this assumption is invalid when the surface in (a) is corrected by the correction pattern at the nearest point, which exacerbates the distortion. Similar results are shown in Fig. 6.1(d-e) where the depth map at the nearest point is distorted, corrected by the nearest correction pattern and the furthest correction pattern, respectively. Similar depth maps are also obtained using Favaro's method [85] and Raj's method [17], i.e., the distortion problem is depth dependent and the above correction method is inadequate.

6.3 Correction by Distortion Cancellation

One means of removing the depth-variant distortion of a DfD method is to cancel the distortion with the stored distortion. We refer this method as CDC. Each correction pattern is a depth map acquired by the DfD method, with a flat surface placed at different distance to the camera. The patterns are obtained from the furthest working limit with equal incrementing distance to the camera until the closest working limit. The patterns are numbered as 1,2,3 ... M from the furthest limit to the closest limit, where M is the total number of patterns. We refer these numbers as correction pattern indices (CPI), and their corresponding depth as the depth offset. The depth value of every location of each pattern is called correction pattern value (CPV).

Correction is achieved by subtracting the distorted depth value at every location in the depth map, i.e., $\mathbf{U}_i$, by the corresponding CPV of a correction pattern $\mathbf{U}_c$ with the most suitable CPI, v, plus the depth offset $\vec{w}(v)$, i.e.,

$$\mathbf{U}_o(x,y) = \mathbf{U}_i(x,y) - \mathbf{U}_c(x,y,v) + \vec{w}(v), \tag{6.2}$$

where (x,y) is the spatial index of the current pixel being corrected, and $\mathbf{U}_o$ is the output corrected depth. The CPI v which corresponds to depth nearest to the distorted depth is given by

$$v = \underset{v\in[1,M]}{\operatorname{argmin}} |\mathbf{U}_o(x,y) - \mathbf{U}_c(x,y,v)|. \tag{6.3}$$

Using Eqn. (6.3) to search for the nearest CPV may give inaccurate results since the single distorted depth may be an unreliable input to Eqn. (6.2). To reduce inaccuracy, the optimal CPV is found within a local region $\mathbf{R}$ of size $(2N+1) \times (2N+1)$ centred at the current location (x,y), i.e.,

$$v = \underset{v\in[1,M]}{\operatorname{argmin}} \sum_{i=-N}^{N} \sum_{j=-N}^{N} |\mathbf{U}_o(x,y) - \mathbf{U}_c(x-i,y-j,v)|. \tag{6.4}$$

In practical applications, $N = 1$ is a good choice. Eqn. (6.4) may not always produce the minimum residual (i.e., its argument of argmin) that is close to zero, i.e., no pixels in the local region are quite similar to the current input pixel. Thus, this problem is addressed by interpolation using the two nearby residuals.

The interpolation process is illustrated in Fig. 6.2(a) for the case where the residual of the left neighbouring CPI is smaller than the right neighbouring CPI, and similarly for the opposite case. First, the minimal residual and the residuals

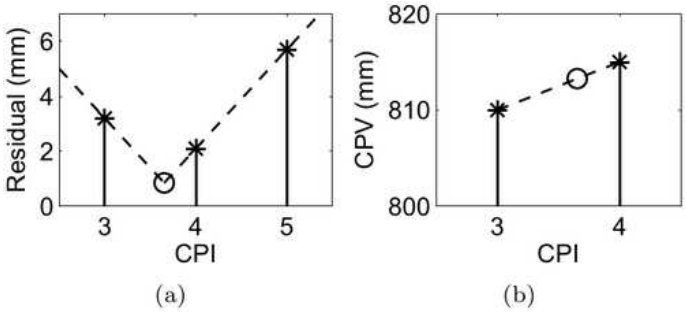


Figure 6.2: (a) Interpolation to find the improved CPI. Key: stem plot with asterisks - the smallest residual value and its two nearest values; dashed lines - the interpolated curve; circle - the minimal of the curve. (b) The second step of interpolation. Key: stem plots with asterisks - the CPV where the estimated index falls in between; dashed line - the straight line connecting both CPV's coordinates; circle - the estimated CPV.

for its two adjacent CPIs are found. In this example, the minimal residual of 2.1 is found at CPI being 4. The two adjacent CPIs are 3 and 4, and their residuals are 3.2 and 5.7, respectively.

Second, the gradient of the line passing through the minimal CPI v_{min} and the CPI with larger residual v_{larg} is given by

$$g = \frac{r_{larg} - r_{min}}{v_{larg} - v_{min}}, \tag{6.5}$$

where

$$r_{larg} = \sum_{i=-N}^{N} \sum_{j=-N}^{N} |\mathbf{U}_o(x, y) - \mathbf{U}_c(x - i, y - j, v_{larg})|, \tag{6.6}$$

$$r_{min} = \sum_{i=-N}^{N} \sum_{j=-N}^{N} |\mathbf{U}_o(x, y) - \mathbf{U}_c(x - i, y - j, v_{min})|. \tag{6.7}$$

In this example, it is the gradient of the line passing CPI=4 and CPI=5 on the right. This gives the equation of the line

$$y = gx + r_{min} - gv_{min}. \tag{6.8}$$

Third, a line with the negative gradient that passes through the other adjacent CPI v_{small} is plotted, the equation of which is

$$y = -gx + r_{small} + gv_{small}, \tag{6.9}$$

where

$$r_{small} = \sum_{i=-N}^{N} \sum_{j=-N}^{N} |\mathbf{U}_o(x, y) - \mathbf{U}_c(x - i, y - j, v_{small})|. \tag{6.10}$$

In this example, it is the line on the left. Fourth, the interpolated CPI is the horizontal coordinates of the intersection of the two lines with Eqn. (6.8) and (6.9).

Finally, since the interpolated CPI is in-between the CPI with smallest residual (4 in this case) and the one with second smallest residual (3 in this case), and the CPVs of them are known (810 for CPI=3 and 816 for CPI=4 as shown in Fig. 6.2(b)), the final estimate of the CPV can be found by a simple linear interpolation as illustrated in the figure. The depth offset $\vec{w}(v)$ is found by another similar interpolation. In cases where the minimal CPI does not have left or right adjacent values, the estimated CPI is the minimal CPI itself.

Note that the system parameters M and N have significant influences on the correction performance. A larger M results in higher accuracy at the cost of lower speed. Thus we suggest the number to be between 20 and 30 for a reasonable trade-off. A larger N provides more neighbouring pixels to search the optimal CPI with. On the other hand, it also means there is a risk that the CPI is determined by pixels that are far away from the current pixel and hence fail to be representative. For a good balance, we suggest N to be between 1/100 and 1/50 of the shorter one of the height and width of the depth map.

6.4 Correction by Least Squares Fit

In [45], we presented a correction method based on least squares fit as part of Li's DfD method. First, the CPI is found by least squares fit and CPV is then found by another least squares fit. Not only are the two fitting procedures unnecessary, accumulative errors are generated that distort the resulting depth map. In this chapter, we propose another correction method by least squares fit, CLSF, which finds the mapping from the distorted depth to the corrected depth directly.

The corrected depth is modelled as a third order polynomial of the spatial indices and the distorted depth $\mathbf{U}_i$, i.e., the corrected depth

$$\mathbf{U}_o(x, y) = \vec{c}_1(1) + \sum_{i=2}^{4} \vec{c}_1(i) x^{i-1} + \sum_{i=5}^{7} \vec{c}_1(i) y^{i-4} + \sum_{i=8}^{10} \vec{c}_1(i) (\mathbf{U}_i(x, y))^{i-7} , \tag{6.11}$$

where $\vec{c}_1$ is the set of coefficients of the fitted polynomial when sufficient samples are collected for $\mathbf{U}_o$ and $\mathbf{U}_i$. In contrast with the correction method in [45], CLSF only requires one least squares fit, and thus avoids any accumulative error. However, the

output depth may not be a perfect third order polynomial of the spatial indices. To address this problem, the entire depth map is divided into a number of equal-sized regions, and appropriate sets of coefficients are used. Large regions (i.e. 2×2) may not be sufficiently valid for all locations, and small regions (e.g. 10×10) contain inadequate number of samples for the fit. In our experiments, the number of regions is set to 3×3, or 9 which yields good results.

6.5 Post-Processing for Low-Texture Region

When the target object surface has little texture, there is insufficient information to retrieve depth. Thus, spurious results are produced such as an extremely low or an extremely high depth value. We assume the low-texture region to have high correlation with its rich-texture neighbourhood and address the problem with the following three steps: (1) A confidence map is produced to identify low-texture regions in the input depth map $\mathbf{D}$; (2) Every individual region is identified and set to no-value; and (3) Each region is filled with 2D least squares fit.

In the first step, the NIR map [16] used to calculate depth map is computed by

$$\mathbf{R} = \frac{\mathbf{I}_1 - \mathbf{I}_2}{\mathbf{I}_1 + \mathbf{I}_2}, \tag{6.12}$$

where $\mathbf{I}_1$ and $\mathbf{I}_2$ are the far-focused and near-focused images respectively. A confidence map is then produced by evaluating the variance of every local region of the ratio map $\mathbf{R}_{loc}$, i.e.,

$$g = \frac{1}{N} \sum_{i=1}^{N} \left(\vec{r}(i) - \mu\right)^2, \tag{6.13}$$

where N is its total number of elements, μ is the mean of $\vec{r}$, and $\vec{r}$ is the 1D vector created by concatenating every row of $\mathbf{R}_{loc}$, i.e.,

$$\vec{r}(N_{col}(i-1) + j) = \mathbf{R}_{loc}(i, j), \tag{6.14}$$

and N_{col} is the number of columns in $\mathbf{R}_{loc}$. The variance g is computed for every local region to form the confidence map $\mathbf{G}$. In this way, low-texture regions corresponding to low variances are readily identified.

In the second step, a mask matrix $\mathbf{B}$ is initialised as a zero matrix of the same size as G by

$$\mathbf{B}(x, y) = \begin{cases} 1, \text{if } \mathbf{G}(x, y) > T \\ 0, \text{otherwise}, \end{cases} \tag{6.15}$$

where T is the variance threshold which is set typically to 0.0144. Since the NIR ranges from 0 to 1, we empirically define the low-texture region to have SD of less than 0.12, which corresponds to a variance of 0.0144. $\mathbf{B}$ indicates the locations of the low-texture pixels. Pixels of the corresponding locations of the input depth map $\mathbf{D}$ are set to be no-value, which are re-estimated later. There may be several unconnected low-texture regions that need to be addressed separately. With $\mathbf{B}$ as its input, Moore-Neighbour tracing [174] is used to find all unconnected low-texture regions defined by their boundary pixels' coordinates.

In the third step, 2D linear regression is used to model the depth values of each region as a third order polynomial of x and y coordinates, i.e.,

$$\hat{\mathbf{D}}(x,y) = \vec{c}_2(1) + \sum_{i=2}^{4} \vec{c}_2(i)x^{i-1} + \sum_{i=5}^{7} \vec{c}_2(i)y^{i-4},$$
$$\forall\ \mathbf{D}(x,y) \neq \text{no-value}, \tag{6.16}$$

where $\hat{\mathbf{D}}$ is the a rectangular region of $\mathbf{B}$ which completely covers the low-texture region, and $\vec{c}_2$ is the set of coefficient obtained by least squares fit.

Figure 6.3: An example of using Moore-Neighbour algorithm to find input samples for least squares fit: left: input binary image with a single low-texture region denoted in white; middle: boundary pixels in grey are identified; right: grey region is identified as the input samples.

Fig. 6.3 illustrates the identification of $\hat{\mathbf{D}}$. In this example, $\mathbf{B}$ is a 7×7 binary image, where the white pixels correspond to the low-texture region, as shown in the left plot. There is only one such region for simplicity in illustration. Moore-Neighbour tracing algorithm uses this image as input and returns the indices of all the boundary pixels, which are drawn in grey in the middle plot. $\hat{\mathbf{D}}$ is then selected as the rectangle that fully covers this region leaving one-pixel margin around its boundaries, as shown in the non-black region in the right plot. We leave this margin because if the region was rectangular, there would be no sampling data for the least squares fit. After $\vec{c}_2$ is estimated, Eqn. (6.16) is used to re-estimate the depth of the no-value pixels. This process is repeated for all remaining low-texture regions.

6.6 Experiments

6.6.1 Experiments condition

A 50 mm professional lens with an aperture of 12.8 mm was used for the DfD reconstruction of objects, which determined the working range to be [887,933] mm away from the lens. The working range was set taking consideration of the length of the optical bench (i.e. 1 metre), the focal length of the lens and the size of the experimental objects (typically 5×5×5 centimetres). A PC with an Intel Core i7 @ 3.40 GHz processor was used for data processing. Four Matlab programs were written for the four DfD methods to evaluate the correction algorithm. The first is the classical generic Subbarao's method [18], the second is learning based Favaro's method [85], the third is the RO based Raj's method [17], and the fourth is the Li's method [45].

Input grey-level images with a resolution of 640 x 480 pixels are used. Each image pair for DfD is divided into a number of contiguous 7×7 sub-image pairs where each iteration of DfD estimation is performed. Thus the resolution of the depth map is 68×91 pixels. A flat surface covered with sandpaper was moved along the optical axis from 933 mm to 887 mm away from the camera, with an increment of 2 mm. Thus, 24 pairs of DfD input images were captured and 24 correction patterns were generated, while the corresponding offsets are from 933 mm to 887 mm incremented by 2 mm.

6.6.2 Quantitative experiments

The correction methods, CDC and CLSF, are applied to the four DfD methods mentioned in Section 6.6.1. Fig. 6.4 shows the results where 9 distorted depth maps are used as the inputs, which were obtained with a flat surface moved from 933 mm to 887 mm away from the camera. The plots in the first row are obtained by averaging the value of each depth map; those in the second row are obtained by calculating the RMSE between the estimated and expected depth for each pixel and taking the average over each depth map; those in the third row are generated by evaluating the variance of each depth map. Thus, the first row shows the average accuracy, the second row indicates the accuracy over the depth map and the third row shows the noise performance.

Note that the experiments are to evaluate the performance of the two correction methods, i.e., not to compare the four DfD methods where the basic difference is whether a PSF model is used or the PSFs used. PSF parameter calibration is involved in [18] and [45], while no calibration is required for the others. Thus, these

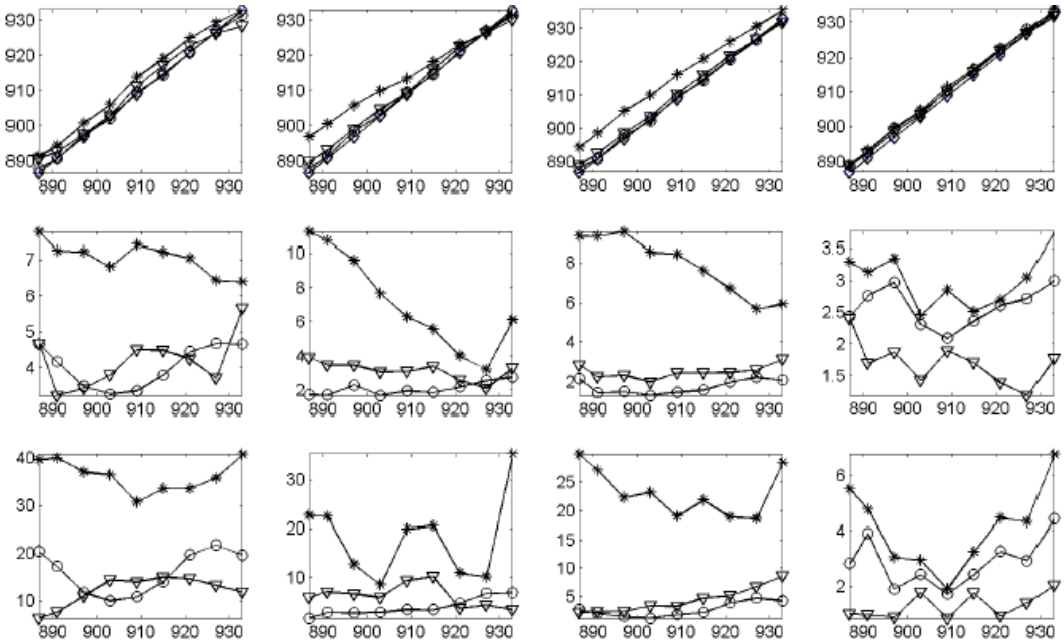


Figure 6.4: Quantitative evaluation of the correction method using Subbarao's method [18] (column 1), Favaro's method [85] (column 2), Raj's method [17] (column 3) and Li's method [45] (column 4). Key: $\diamond$ - expected depth; $*$ - uncorrected result; $\circ$ - corrected with CDC; ∇ - corrected with CLSF. Row 1: estimated depth (vertical axis) against expected depth in mm (horizontal axis). Row 2: RMSE in mm against depth. Row 3: variance in mm^2 against depth.

methods produce the better accuracy. This is verified in the first and fourth column of Fig. 6.4 as the calibration methods bring the estimated depth much closer to the expected depth.

The results also show that the RMSE and variance of the uncorrected results are dominated by its general shape, i.e., an elliptical surface with high curvature, and those of the uncorrected result are dominated by the high-frequency noises. The RMSE and the variance of the uncorrected results generally decrease while those of the corrected ones increase with depth. Despite of this general DfD problem, it can be seen in the second and the third rows of Fig. 6.4 that both correction methods effectively reduce the overall RMSE and the noise level in the DfD reconstruction. Notably, for results using the methods of Subbarao, Favaro and Raj, CDC generally produces smaller RMSE and less noise than the CLSF. However, CLSF produces slightly better reconstruction than CDC for Li's results.

6.6.3 Qualitative experiments

In order to generate 3D surface reconstruction to enable a qualitative visual comparison between the DfD results before and after correction, we have chosen two sets of test scenes: Set A comprises two simple objects and Set B comprises realistic complex objects. Subbarao's, Favaro's, Raj's and Li's methods are used to compare both sets of results. Set A comprises one scene with wooden staircases (Stair) and one scene of a circular depression on a wooden block (Circular) as shown in Fig. 6.5. Stair contains depths across the entire working range.

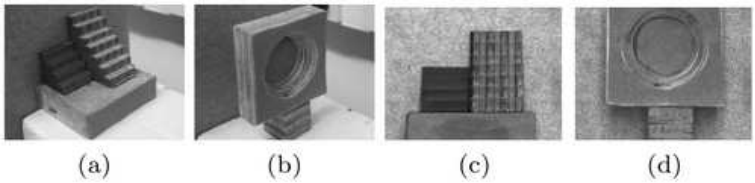


Figure 6.5: Test scenes of Set A: (a), (b) and their near-focused images (c) and (d), respectively.

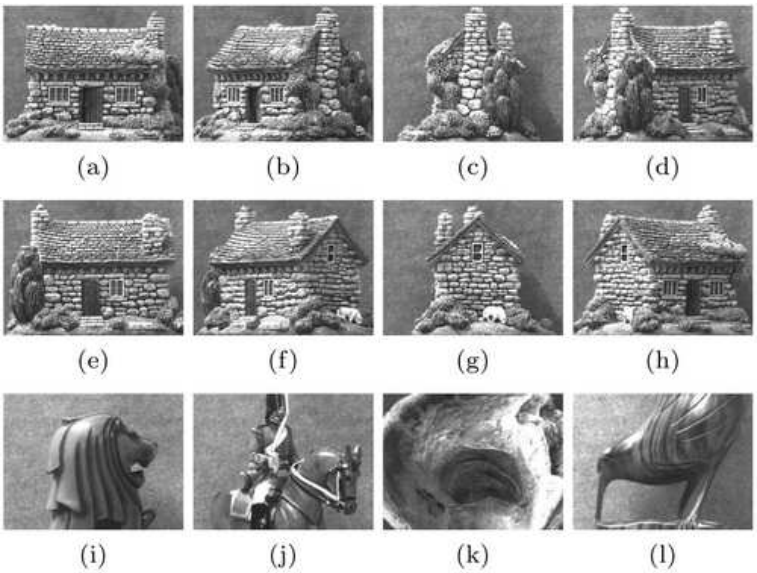


Figure 6.6: Test scenes of Set B: (a)-(h): near-focused images of 8 views of House ; (i)-(l): near-focused image of Lion, Soldier, Shell, and Bird, respectively.

In order to evaluate the correction methods for smaller depths, Circular is moved towards the camera so that its rear is beyond the background. Fig. 6.6 shows the test scenes of Set B: a stone house model (House), a wooden lion stature (Lion), a painted metallic soldier-on-house (Soldier) stature, a mussel shell (Shell) and a wooden bird stature (Bird). The set includes 8 views of House and one view from

the remaining objects.

Fig. 6.7 shows the results of Stair before and after correction for the four DfD methods. The uncorrected result of Subbarao's method in column 1 shows heavy spike-like noises throughout the plot, which are sufficiently severe that the global elliptical distortion centred at the bottom right part of the plot is hardly visible. After correction, the noises are considerable reduced and the distortion is also removed. In the uncorrected result of Favaro's method in column 2, the distortion is more visible in the background, which is removed after correction.

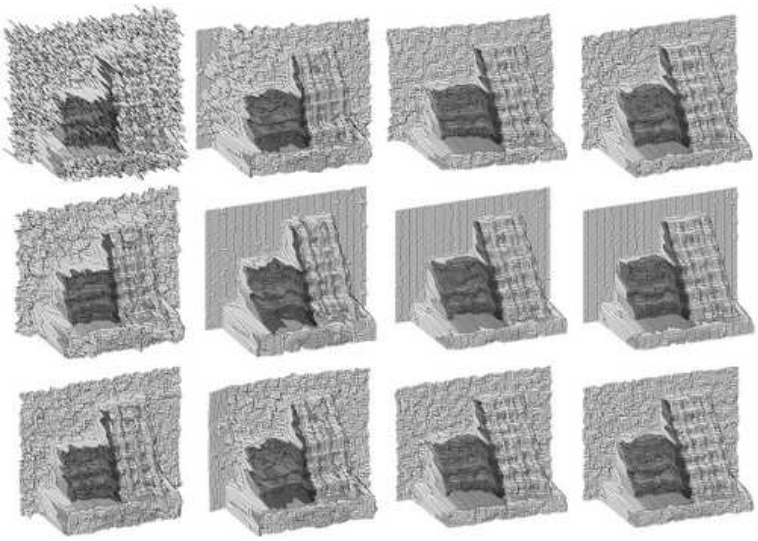


Figure 6.7: Mesh-plots of Stair: column 1-4: Subbarao, Favaro, Raj, and Li. Row 1: Original; row 2: corrected using CDC; row 3: corrected using CLSF.

In the uncorrected result of the Raj's method in column 3, the distortion is visible in both the background and foreground, which is effectively removed after correction. In the uncorrected result of the Li's method in column 4, the distortion is not visible and the correction methods make no significant changes to the reconstruction. Furthermore, for all these methods, CDC produces lower noise levels than CLSF.

Fig. 6.8 shows the results for Circular. In the uncorrected result of the Subbarao's method in column 1, the elliptical distortion makes the top left part further away than it should be. The spike-like noises are also present. After correction, these artefacts are considerably reduced. In the uncorrected result of the Favaro's method in column 2, a distortion makes the left part of the background further than the right part of the background, which is reduced after correction. In the uncorrected result of the Raj's method in column 3, there is a small hill-like distortion with its peak centred at the bottom right part. It is reduced after correction. In

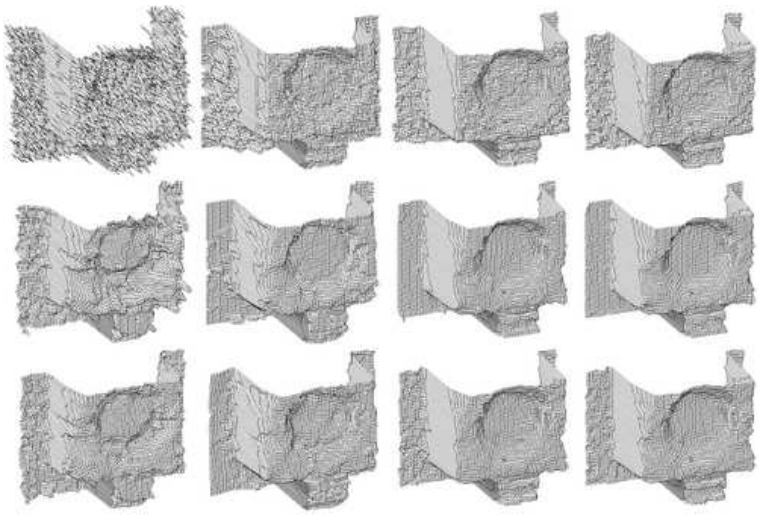


Figure 6.8: Mesh-plots of Circular: column 1-4: Subbarao, Favaro, Raj, and Li. Row 1: Original; row 2: corrected using CDC; row 3: corrected using CLSF.

the uncorrected result of the Li's method in column 4, the distortion is not visible and the correction methods make no significant changes to the reconstruction. In addition, CDC manages to remove noises in the background, and the foreground flat surfaces.

Fig. 6.9 shows the results of the front view of House before and after correction. The spike-like noises throughout the uncorrected result of the Subbarao's method in column 1, are effectively reduced after correction. In the uncorrected results of the Favaro's method in column 2 and Li's method in column 4, there is no visible distortion and the correction methods make no significant changes to the reconstruction. In the uncorrected result of the Raj's method in column 3, a strong elliptical distortion in the background makes the top-left corner further away than it should be. The distortion also appears at the front of the house. The distortion is significantly reduced after correction. In addition, CDC produces lower noise level than CLSF.

Fig. 6.10 shows the results of eight views of House using Li's method corrected by CDC. Although the background is not perfectly flat, and some of the flat surfaces on the wall are bumpy, no spike-like noise and elliptical distortion are visible. Overall, these plots show that CDC produces good quality reconstructions for House.

Fig. 6.11 shows the results of Lion before and after correction. The spike-like noises throughout the plot of the uncorrected result of the Subbarao's method in column 1 are effectively reduced after correction. In the uncorrected result of the

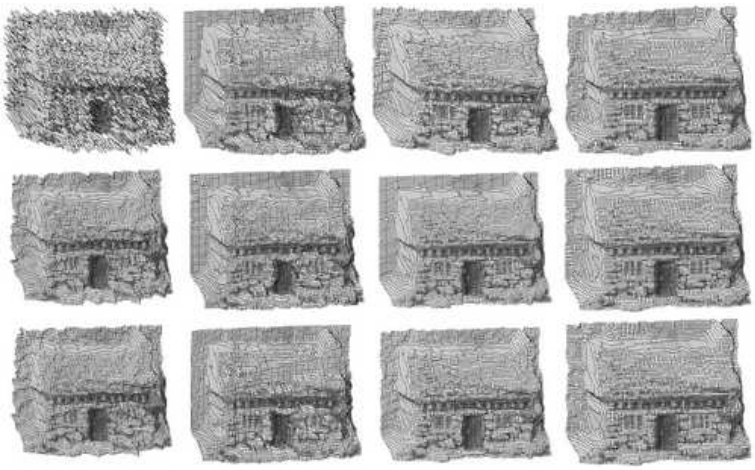


Figure 6.9: Mesh-plots of the front view of House: column 1-4: Subbarao, Favaro, Raj, and Li, respectively. Row 1: Original; row 2: corrected using CDC; row 3: corrected using CLSF.

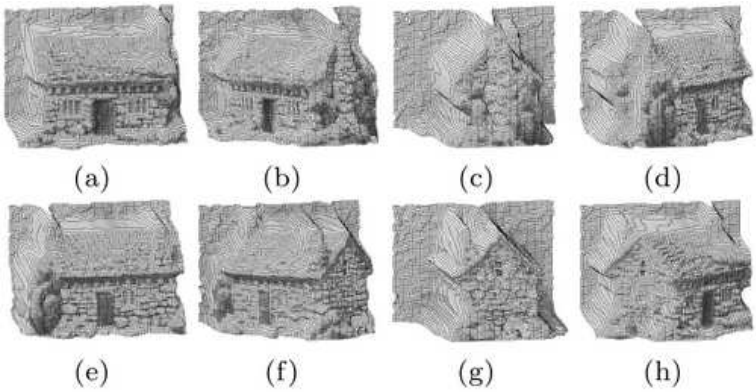


Figure 6.10: Mesh-plots of eight views of the house using Li's method after CDC.

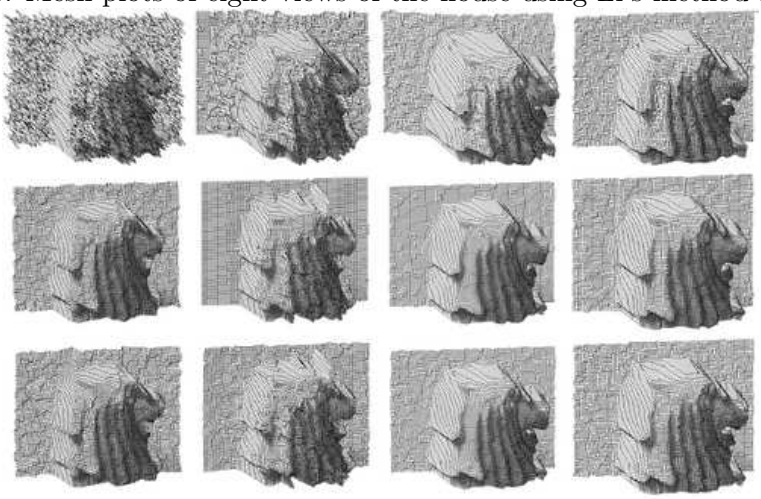


Figure 6.11: Mesh-plots of Lion: column 1-4: Subbarao, Favaro, Raj, and Li, respectively. Row 1: Original; row 2: corrected using CDC; row 3: corrected using CLSF.

Favaro's method in column 2, a small distortion in the background makes the left part further away than the right. The distortion is reduced after correction. Moreover, CDC produces lower noise level than CLSF. In the uncorrected result of the Raj's method in column 3, a strong elliptical distortion in the background makes the top-left corner further away than it should be. The distortion also appears at the front of the house. The distortion is significantly reduced after correction. In addition, CDC produces lower noise level than CLSF. In the uncorrected result of the Li's method in column 4, a small distortion in the background, makes the top right slightly closer than it should be. The distortion is removed after correction.

Fig. 6.12 shows the results of Soldier before and after correction. In the uncorrected result of the Subbarao's method in column 1, the spike-like noise is present throughout the plot. Some parts of the soldier are very smooth causing reflections and lack of textures, leading to reconstruction failure on some parts of the soldier's body. These artefacts are effectively reduced after correction. In the uncorrected result of the Favaro's method in column 2, a strong distortion in the background makes the left part much further away than the right part. This distortion is removed after correction. In addition, CDC produces smoother result than CLSF. In the uncorrected result of the Raj's method in column 3, a strong elliptical distortion throughout the plot makes the top-left part of the background further away than it should be. The house also appears too close to the viewer. These distortions are significantly reduced after correction. In the uncorrected result of the Li's method in column 4, there is no visible distortion. The correction methods make no significant changes to the reconstruction. But CDC improves the smoothness of the surface.

Fig. 6.13 shows the results of Shell before and after correction. In the uncorrected result of the Subbarao's method in column 1, the spike-like noise is present throughout the plot. A very sharp noise is also present at the top right part of the plot due to lack of textures. These artefacts are effectively reduced after correction. In the uncorrected results of the Favaro's method in column 2 and Li's method in column 4, there is no visible distortion and the correction methods make no significant changes to the reconstruction. In the uncorrected result of the Raj's method in column 3, the small elliptical distortion throughout the plot makes the top-left corner further away than it should be, and the bottom right part closer than it should be. These distortions are significantly reduced after correction.

Fig. 6.14 shows the results of Bird before and after correction. In the uncorrected result of the Subbarao's method in column 1, the spike-like noise is present throughout the plot. The smoothness of the object causes lack of textures which leads to sharp noises on the bird's body. These artefacts are effectively reduced after

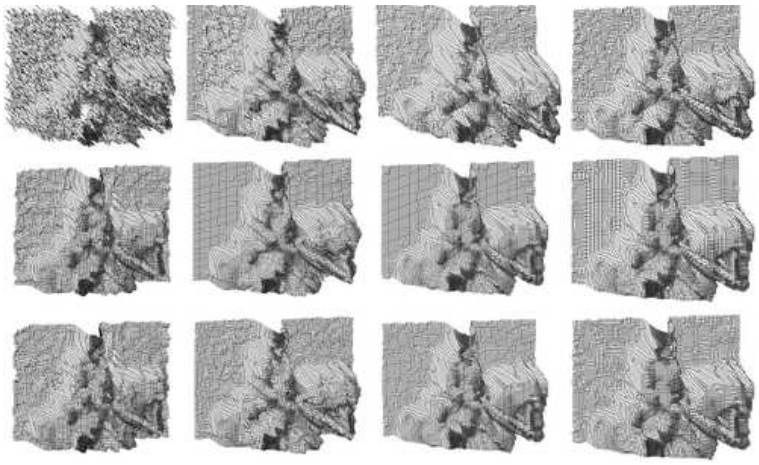


Figure 6.12: Mesh-plots of Soldier: row 1-4: Subbarao, Favaro, Raj, and Li, respectively. Column 1: Original; column 2: corrected using CDC; column 3: corrected using CLSF.

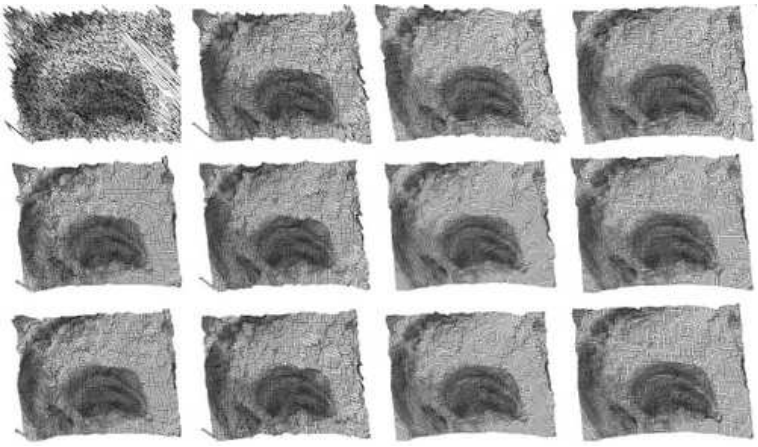


Figure 6.13: Shell: row 1-4: Subbarao, Favaro, Raj, and Li, respectively. Column 1: Original; column 2: corrected using CDC; column 3: corrected using CLSF.

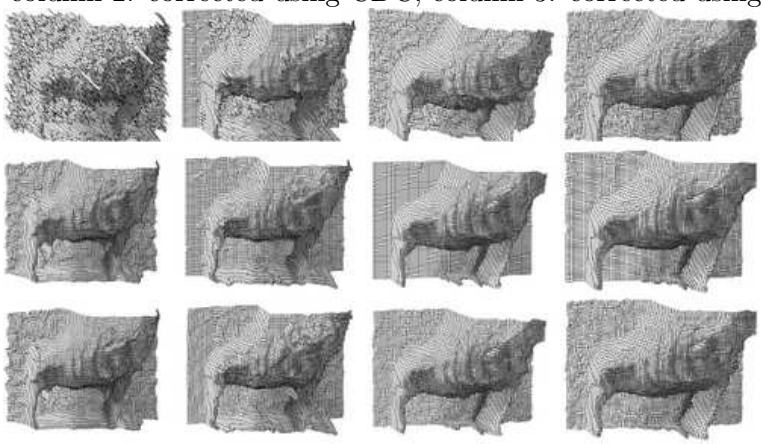


Figure 6.14: Bird: row 1-4: Subbarao, Favaro, Raj, and Li, respectively. Column 1: Original; column 2: corrected using CDC; column 3: corrected using CLSF.

correction. In the uncorrected result of the Favaro's method in column 2, no significant distortion is visible since the right part of the flat background is mostly occluded by the bird. However, the poor reconstruction makes it difficult to determine whether the bottom right corner of the plot is the bird or the background. After correction, it is clear that the bottom right corner is part of the background. The quality of the reconstruction is also considerably improved as a result. In the uncorrected result of the Raj's method in column 3, a strong distortion makes the background not flat and the reconstruction of the bird is poor. After correction, the background is made flat and the reconstruction is improved. CDC also produces smaller noise level than CLSF. In the uncorrected result of the Li's method in column 4, no significant distortion is visible. Thus, no significant changes are made to the reconstruction after correction, except CDC reduces the local noises.

As the experimental results illustrate, both CDC and CLSF are able to eliminate the elliptical distortion effectively. In combination with the post-processing algorithm, both methods significantly mitigate the unstable depth results due to low-texture regions. CDC improves the smoothness of the flat and smooth surfaces. However, it is complex which involves a number of iterative searching procedures. CLSF is very efficient with a closed form equation in Eqn. (6.11). However it produces coarser correction result than CDC.

6.6.4 Computational cost

In terms of processing time taken to correct a 63×86 depth map, CDC requires 994 milliseconds and CLSF requires 5.87 milliseconds without post-processing. The additional 443 milliseconds is typically required for the post-processing. Thus, CLSF is faster than CDC at a cost of lower accuracy and smoothness. The correction method in [45] which does not involve post-processing takes 10.3 milliseconds. Note that each of these results is obtained by averaging five independent measurements.

6.7 Summary

This chapter presents two DfD correction methods to address the depth-variant elliptical distortion that often occurs during DfD computation. The methods correct the depth estimation generated by any DfD algorithm using a number of correction patterns generated by the correction method. CDC finds the nearest CPV to cancel out the distortion at every location and further improves the accuracy with consideration of the local region and interpolation. CLSF finds the mapping from the distorted to the corrected results directly and further improves the accuracy by di-

viding the depth maps into a number of regions sharing separate sets of coefficients. CDC produces better reconstructions than CLSF but at the expense of much lower speed. Both quantitative and qualitative experiments on real images show that the proposed methods effectively remove the distortion and other noise.

One possible future work is to explore the applicability of both CDC and CLSF on DfD using a single image and active DfD. Another is to investigate modelling techniques that are more sophisticated than least squares fit to further improve the reconstruction accuracy. Furthermore, machine learning techniques could be considered for distortion removal.

Chapter 7

Training Silhouettes from any View

7.1 Introduction

Until now, the first topic of this thesis, i.e. passive DfD, has been presented in Chapter 4 - 6. From this chapter forward, the proposed human activity recognition system is described. The system is divided into three parts, which are training data generation, run-time data improvement(shadow removal) and recognition algorithm, and they are respectively reported in Chapter 7 - 9. Training data generation aims to efficiently create a library of silhouettes data which allows the recognition algorithm to embed any input silhouette. Shadow removal intends to improve the run-time input silhouettes used by the recognition algorithm.

One approach to view-invariant recognition is to provide training silhouettes captured from different views. However, there has been little interest in finding an efficient and accurate way of obtaining 3D silhouettes as training data [38]. For most recognition system based on silhouettes, the training data is generated with a camera, or multiple cameras if more views are required. This can be impractical for numerous views, where error in camera placement and manual set-up effort are inevitable.

Real-time 3D object reconstruction methods based on shape from silhouettes, e.g., [175], require complex camera calibration and intensive computation. DfD is potentially useful for this application, where a 3D human body model can be obtained with only two cameras with one facing another and the subject in between. However, our current DfD system is neither portable nor incorporate real-time automatic input video acquisition. Thus, we use 3D coordinates of a human body

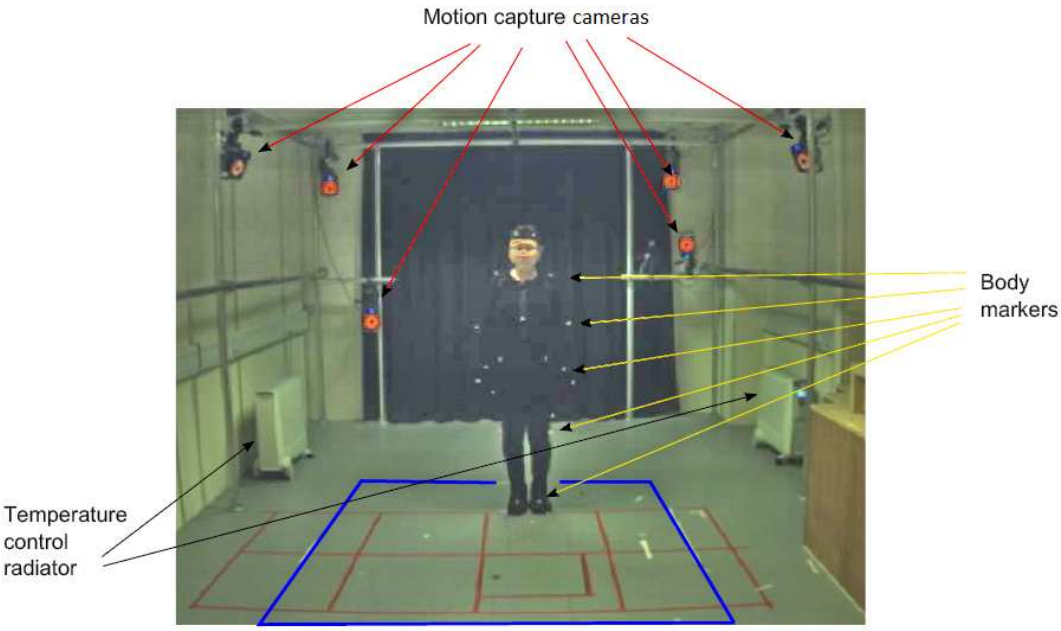


Figure 7.1: Vicon Nexus data-capturing hardware environment.

extracted using Vicon Nexus [39] and markers placed on an actor while performing an activity. A silhouette generation algorithm is introduced which converts the 3D-coordinate data for each frame into a 2D silhouette for a user-specified projection angle (or view). Very similar silhouettes associated with an activity are removed using a selection algorithm and the remaining silhouettes form the gallery (training) silhouettes.

Vicon Nexus uses multiple infrared cameras and is the gold standard in human motion analysis due to its accurate position information [38]. It records several markers placed on a subject during an activity. Its main limitation is its inability to track markers when they are occluded. We address this limitation by appropriate placement of the markers and post manual editing.

Fig. 7.1 shows the hardware environment. Six of the infrared motion cameras are indicated with red lines. Each camera has a number of infrared light emitting diodes (LEDs) which emit infrared, and its sensor picks up any reflections from the body markers. The advantages of infrared cameras include [176]: (1) infrared is not visible, thus has little disturbance on the subjects' performance; (2) infrared cameras have low sensitivity to visible light, thus capture fewer interferences than video cameras; (3) infrared LEDs are low-power sources and thus do not cause harmful radiation to human body. Five of the body markers placed on the subject's clothing are indicated with yellow lines. The area which can be seen by all cameras

are indicated by the blue rectangle. There are two thermal radiators indicated by the black lines which maintain the room temperature to allow natural performance of the subject.

This chapter is organised as follows. Section 7.2 presents the theory and methodology of our method for generating training silhouettes. Section 7.3 demonstrates the usefulness of the method with experiments. Finally Section 7.4 summarises this chapter.

7.2 Theory

7.2.1 Placement of the markers

To obtain the marker correspondence in video sequence, we created a body template for Vicon Nexus using 36 markers placed on a jacket, 32 markers placed on a trouser, 5 markers on the head cover and 2 markers on each hand glove. Fig. 7.2 shows the placement of markers on the jacket, trouser, head cover and gloves. Note that the markers are positioned to allow reconstruction of the body volume hull. Fig. 7.3 shows an actor (whose face is obscured to mask his identity) wearing such clothing. As the figure shows, there are sufficient markers on the clothing to reconstruct the volume hull of the human body. We also create templates for a bag, spade, short gun and long gun which use 9, 10, 4 and 6 markers, respectively as shown in Fig. 7.4.

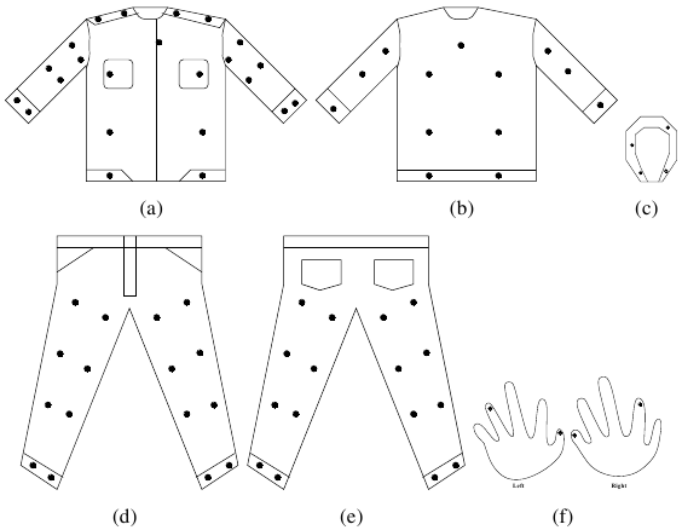


Figure 7.2: Positions of markers on subject's clothing: (a) jacket front; (b) jacket back; (c) head cover(c); (d) trouser front; (e) trouser back; and (f) left and right gloves.

Figure 7.3: Position of markers on an actor's clothing: top left: jacket front; top right: trouser front; bottom left: jacket back; and bottom right: trouser back.

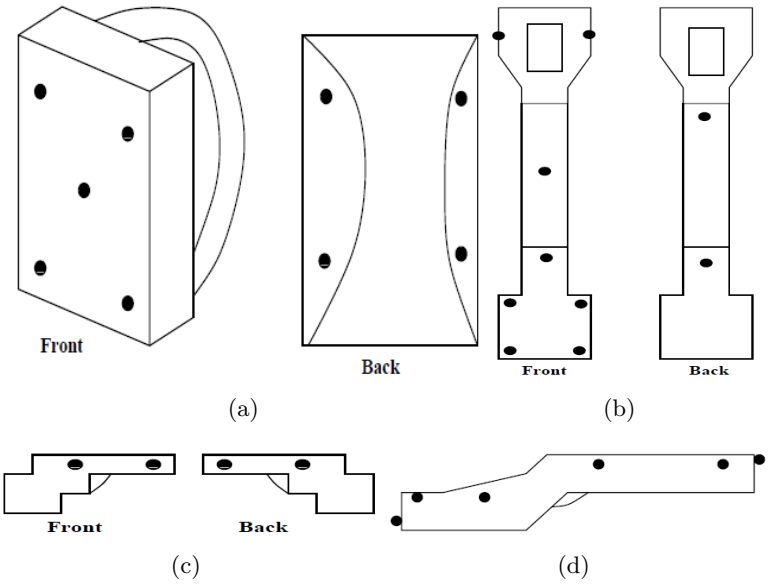


Figure 7.4: Position of markers on (a) bag, (b) spade, (c) short gun and (d) long gun.

7.2.2 Generating silhouettes from any view

The extracted marker points are grouped according to their category, and each category has 3 points which define a triangular surface on the body. These surfaces are projected to a plane according to the view angle specified. Silhouettes are formed if sufficient number of surfaces is created. This silhouette generation algorithm consists of 3 steps.

In step 1, each marker point is classified into its category, where each corresponds to a body part, i.e., forearm with hand, arm, head, upper torso (i.e., chest and shoulder), lower (i.e., remaining) torso, thigh, or leg with feet. The classification is simply achieved by the known positions of the markers on the subject's clothing. Each category is considered rigid and any change in its shape makes little difference to the overall body posture.

In step 2, all possible triplets of points of one category are found by exhaustive search. They are then used for generating triangular surfaces. All possible triplets instead of only those that are necessary to form the volume hull are used because marker points are sometimes occluded which results in missing surfaces, which in-turn leads to broken volume hull and silhouette. By using all triplets, the resulting redundant surfaces are used to fill any gaps. Fig. 7.5 (a) shows a volume hull created where the lower torso and right leg are broken due to occlusion of marker points, which are essential to create the volume hull. Fig. 7.5 (b) shows the volume hull where all possible triangular surfaces are plot within each category, where the broken parts are filled by redundant surfaces.

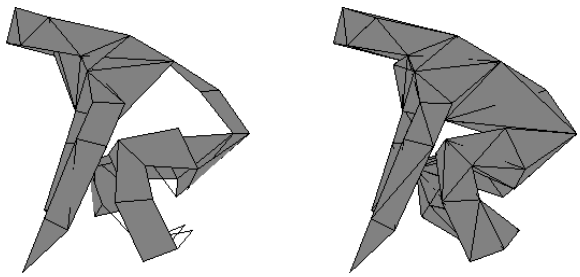


Figure 7.5: The usefulness of using all triplets: (a) a broken volume hull and (b) after redundant triangular surfaces are added.

In step 3, three MATLAB functions are used. *trisurf*() is used to generate the triangular surfaces from the triplet indices and their coordinates. The view angle is determined by *view*(). *getframe*() is used to convert each plot of triangular surfaces to a silhouette image. Fig. 7.6 illustrates the silhouette generation for four views of a walking posture.

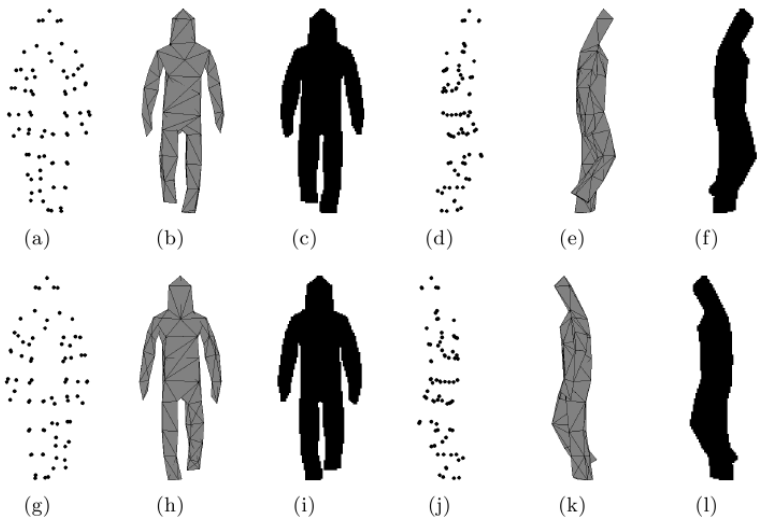


Figure 7.6: Generating training silhouettes (column 3 and 6) from projected volume hull (column 2 and 5) obtained by 3D marker points (column 1 and 4). (a)-(c): Front view: (d)-(f): right view; (g)-(i): back view; and (j)-(l): left view.

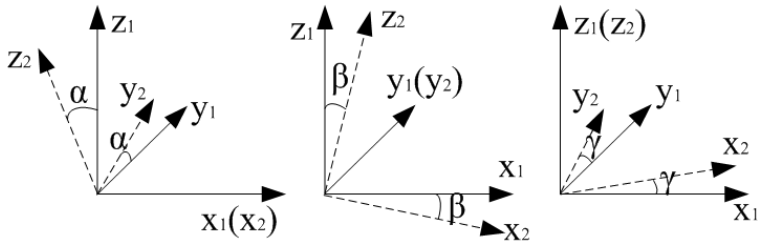


Figure 7.7: Axis rotation about the x-axis (left), y-axis (middle) and z-axis (right). Footer $_1$ and $_2$ denote the original axis and the rotated axis respectively.

If MATLAB is not available, the following algorithm replaces step 3. First, given the view as a set of rotational angles about the x, y and z axis of a marker point, the new coordinates of the marker points are found. Consider Fig. 7.7 where the x-y plane represents the ground, and the z axis is the upright direction. The view position is defined by angles α, β and γ, that correspond to the rotation angle about the x, y and z axis, respectively. According to the Euler's rotation theorem [177], the rotational matrices about the three axes are given by

$$\mathbf{R}_x = \begin{bmatrix} 1 & 0 & 0 \\ 0 & \cos\alpha & -\sin\alpha \\ 0 & \sin\alpha & \cos\alpha \end{bmatrix}, \quad \mathbf{R}_y = \begin{bmatrix} \cos\beta & 0 & \sin\beta \\ 0 & 1 & 0 \\ -\sin\beta & 0 & \cos\alpha \end{bmatrix} \text{ and } \mathbf{R}_z = \begin{bmatrix} \cos\gamma & -\sin\gamma & 0 \\ \sin\gamma & \cos\gamma & 0 \\ 0 & 0 & 1 \end{bmatrix}.$$

Thus, the rotated coordinate of each point (x, y, z) is

$$\begin{bmatrix} \hat{x} \\ \hat{y} \\ \hat{z} \end{bmatrix} = \mathbf{R}_x \times \mathbf{R}_y \times \mathbf{R}_z \times \begin{bmatrix} x \\ y \\ z \end{bmatrix} . \tag{7.1}$$

This process is repeated for every marker points for a given view.

The x-z plane is chosen to be the viewing plane such that the y coordinate is removed (n.b. the y-z plane can also be the viewing plane when the x coordinate is removed, but this corresponds to a different set of rotational angles α, β and γ). Thus, the 2D coordinate of the projected object is $(\hat{x}, \hat{z})$. Each group of three points are set to 1 in a $M \times N$ binary image $\mathbf{C}$ containing zeros, where

$$M = \frac{g_y}{D} , \quad N = \frac{g_x}{D} , \tag{7.2}$$

g_x and g_y are respectively the range of the x coordinate and the y coordinate of a point, and D is the number square millimetres sharing one pixel. A hollow triangle is created for each group of points, where each edge, or pixels that link each pair of points are set to 1 using Bresenham's line-drawing algorithm [178]. The hollow triangles are filled as follows. For each row and scanning from left to right, the first pixel with a value of 1 is detected (considered as a left edge pixel), and all subsequent pixels are set to 1 until the right edge pixel with a value of 1 is detected.

7.2.3 Removal of redundant silhouettes

Since Vicon Nexus produces data at high frame-rate, many of the silhouettes obtained are similar and are thus removed as follows. Denote a $N \times N$ difference matrix

$$\mathbf{M}(x, y) = \sum_{c=1}^{C} \sum_{r=1}^{R} \mathbf{I}_x(c, r) - \mathbf{I}_y(c, r) , \tag{7.3}$$

where $x, y \in [1, N]$ are image indices, I_x and I_y are respectively the xth and yth silhouettes, C and R are respectively number of columns and rows per image, $c \in [1, C]$ and $r \in [1, R]$ are respectively the column and row index. The diagonal terms of $\mathbf{M}$ are set to positive infinity. Instead of computing $\mathbf{M}$ every time a silhouette with minimal difference is removed (which is time consuming), $\mathbf{M}$ is computed once, where the index of minimal value

$$\tilde{x}, \tilde{y} = \underset{x,y}{\operatorname{argmin}} \, \mathbf{M}(x, y) \tag{7.4}$$

is computed iteratively. For odd iteration remove $\mathbf{I}_{\tilde{x}}$ and set all elements of xth row to positive infinity. For even iteration remove $\mathbf{I}_{\tilde{y}}$ and set the yth column to positive infinity. The iteration ends when the desired number of silhouettes is removed.

7.3 Experiments

In order to demonstrate view-invariance, we obtain test data from 9 views by dividing the horizontal plane of the scene equally into 8 portions. Due to difficulty in obtaining data from more than one elevation, data from only one elevation, at zero elevation angle, is obtained. Fig. 7.8 illustrates the 8 views with respect to the walking direction used to capture test data, with the centre where an activity is performed denoted by $\times$. For the training silhouettes we use one actor to perform the ten activities in Table 7.1, which are specified by our sponsor, Defence Science and Technology Laboratory. Note that we allow the actor to move around the centre or change direction during an activity.

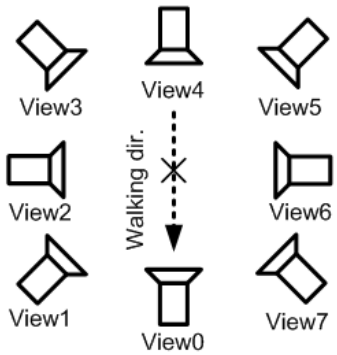


Figure 7.8: Experiment set-up for test data acquisition.

Fig. 7.9 shows the silhouettes generated from one silhouette for each activity. The size of every silhouette is normalised, thus the Crouch silhouette is large in comparison. Fig. 7.10 shows the silhouettes generated for the 10 activities. It can be seen there is not much detail of the reconstructed head. This is because only limited number of markers is placed on the head, and markers are only placed on the actor's hat with none placed on the face. Despite of this minor issue, high quality silhouettes are generated which are visually distinguishable. In addition, it should be noted that our activity recognition system presented in Chapter 9 does not require accurate silhouette matching. Therefore, the quality is adequate for the subsequent processing.

Table 7.1: Activities and their description.

Activity	Description
Walk	Walk and pass centre.
Drop	Walk to centre with a large bag, drop the bag into a large bin and continue walking.
Crouch	Walk to centre, crouch and place or pick-up objects, and continue walking.
Peer	Walk towards a wall near centre, peer over the wall twice and walk back.
Mobile	While talking to a mobile phone, walk to centre, circulate centre, and walk straight on.
Place	Walk to centre, place 6 small indicators one at a time onto different locations near centre, and continue walking.
Bag	Walk with a large bag passing centre.
Shoot	Walk to centre, raise up a small gun, shoot three times and then continue walking.
Gun	Walk to centre with a long gun, circulate centre and then continue walking straight.
Dig	Walk to centre with a spade, dig the ground several times, and continue walking.

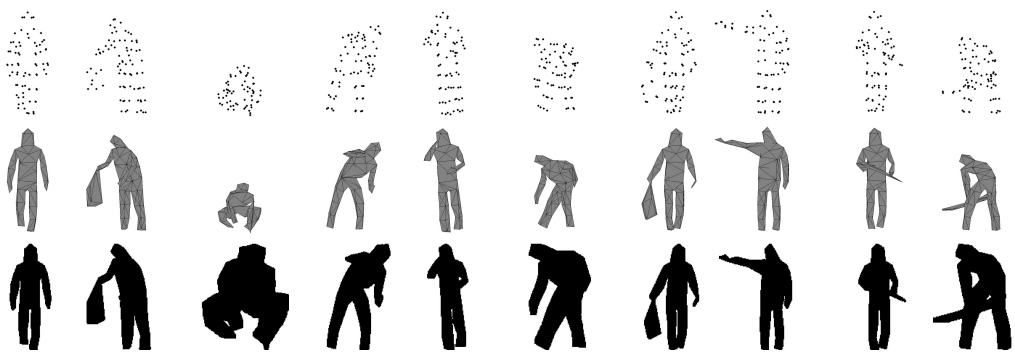


Figure 7.9: Generating training silhouettes for a posture of each activity. Row 1: projected marker points; row2: volume hulls; and row 3: projected volume hulls (i.e., silhouettes).

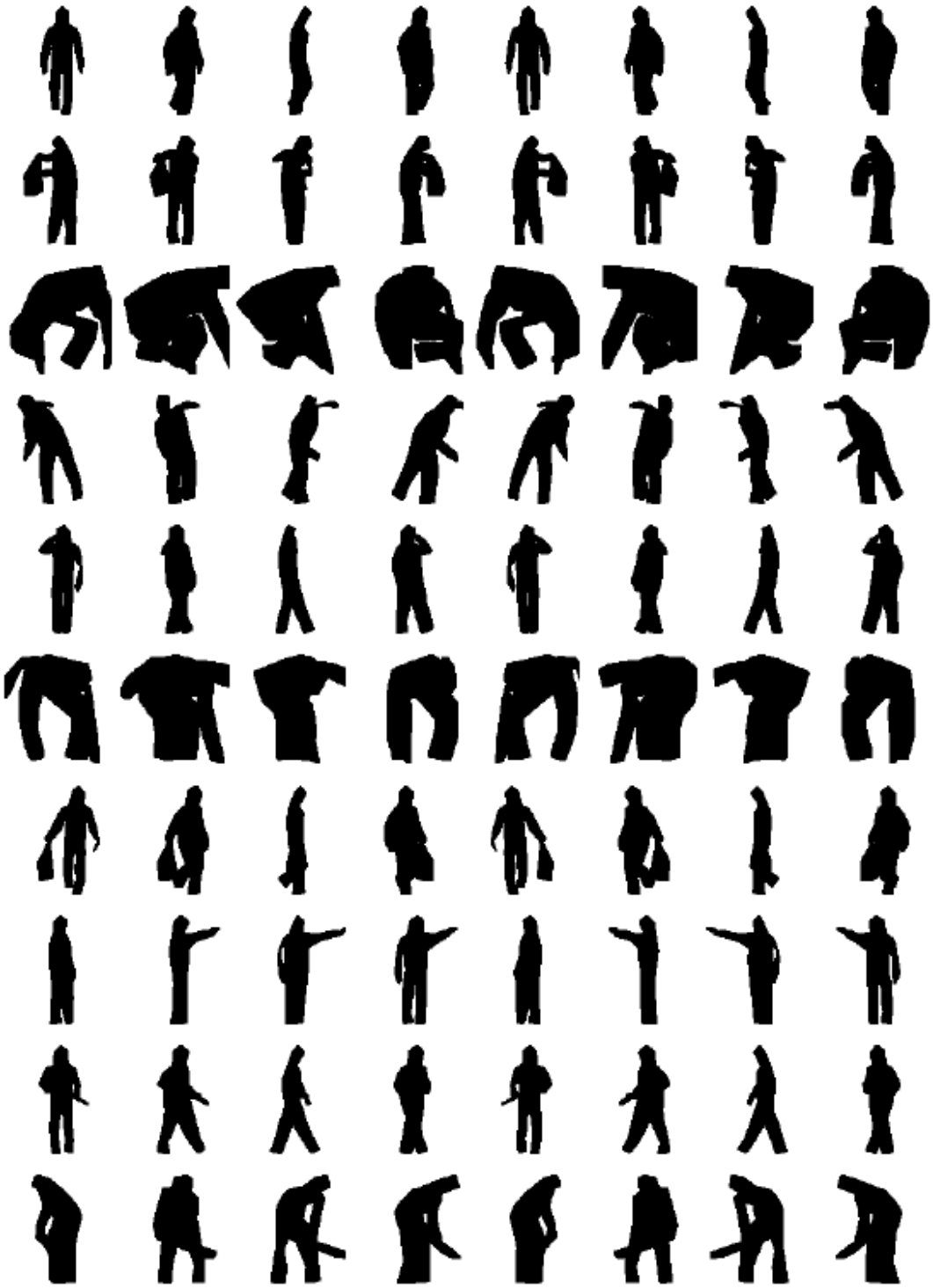


Figure 7.10: 8 views of the 3D silhouettes. Row 1-10: the 10 activities in Table 7.1; column 1-8: view 1-8 respectively.

7.4 Summary

This chapter presents a novel methodology that involves using Vicon Nexus and appropriate placements of marks on an actor's clothing that generates training silhouettes generation from any view. The methodology avoids the effort of capturing training data with many cameras and the need to address the associated calibration errors. It involves firstly the determination of the category of each detected marker point. Second, all possible triplets of points of each category are found. Third, all triangular surfaces are found within each category, which are then plotted and projected according to the view specified to form the silhouette. Finally, a simple and fast algorithm is presented to remove redundant silhouettes.

Chapter 8

Shadow Removal

8.1 Introduction

In the previous section, a training silhouette acquisition algorithm has been presented that enables efficient generation of the silhouette library of the recognition algorithm, and it is executed off-line. The run-time input to the proposed recognition algorithm is a sequence of silhouettes of a person, and every silhouette often contains his/her shadow having a negative influence on the system performance. Thus, a shadow removal algorithm that improves the input data for the recognition algorithm is needed.

Silhouettes can be extracted from video using background subtraction functions in OpenCV [179] based on [180; 181; 182]. Although some of these methods incorporate shadow removal, shadow cannot be completely removed when it has similar colour saturation or textures as the surrounding background. Shadow detection methods can be based on geometry, chromaticity, physical knowledge and texture [40]. Only the large region texture-based method [40] can completely remove a shadow, but it requires large computational effort. Thus, we propose a shadow removal method based on known position of the sun rather than image information. The method comprises four steps: (1) the length of the shadow and its inclination angle to the horizon are computed; (2) projection of the shadow on the camera plane is estimated; (3) the rotational angle and its variance required to remove the shadow is calculated; finally (4) the shadow is removed. This chapter is organised as follows. Section 8.2 presents the theory of the proposed shadow removal algorithm. Experiments and comparison with other methods are presented in Section 8.3. Finally Section 8.4 summarises the chapter.

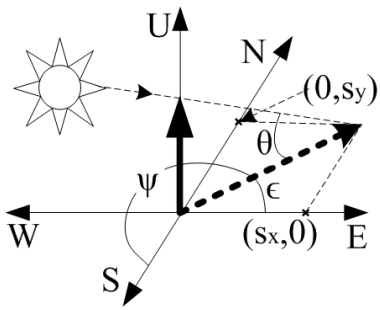


Figure 8.1: Modelling shadow: N - north, S - south, E - East, W - west, and U - upward.

8.2 Theory

8.2.1 Step 1: Determining shadow

Fig. 8.1 shows the isometric view of a shadow at a specific time and location, where the sun is in the south-west and the shadow is towards the north-east. The elevation angle θ is between the sun light and the ground [183]. The angle ψ from the south to the shadow is the solar azimuth angle. The shadow angle ϵ is measured from east to the shadow. The west-east component and north-south component of the shadow are respectively s_x and s_y.

The length of the shadow is determined first, given by

$$s = \frac{H}{\tan(\theta)} \ , \tag{8.1}$$

where H is the average height of the human subject (its variation on the subsequent analysis is discussed in Section 8.2.3),

$$\theta = \arcsin\left(\sin\kappa_s \sin\kappa_l + \cos\kappa_s \cos\kappa_l \cos\xi\right) \ , \tag{8.2}$$

and κ_l is local latitude. The Solar Declination Angle

$$\kappa_s = \arcsin\left[\sin\left(\frac{23.45\pi}{180}\right)\sin\left(\frac{2\pi}{365}(d-81)\right)\right] \ , \tag{8.3}$$

where d is the number of days since the beginning of the year. The Hour Angle and Local Solar Time are respectively

$$\xi = \frac{\pi}{12}(t_s - 12) \tag{8.4}$$

$$t_s = t + \frac{t_c}{60} \ , \tag{8.5}$$

where the Time Correction Factor

$$t_c = 4(\lambda_l - \lambda_s) + t_e \ , \tag{8.6}$$

λ_l is local longitude, the Local Standard Time Meridian

$$\lambda_s = \frac{\pi}{12} \cdot (t - t_g) \ , \tag{8.7}$$

and t and t_g are respectively the current time and Greenwich Mean Time in hour. The Equation of Time

$$t_e = 9.87 \sin(2b_1) - 7.53 \cos(b_1) - 1.5 \sin(b_1) \ , \tag{8.8}$$

where $b_1 = \frac{2\pi}{365} \cdot (d - 81)$. This is followed by computing the shadow angle

$$\epsilon = \frac{3}{2}\pi - \psi \ , \tag{8.9}$$

where

$$\psi = \arccos\left[\frac{\sin\kappa_s \cos\kappa_l - \cos\kappa_s \sin\kappa_l \cos\xi}{\cos\theta}\right], \text{ when } t_s \leq 12$$
$$\psi = 2\pi - \psi, \text{ when } t_s > 12 \ . \tag{8.10}$$

Thus the horizontal x and vertical y component of the shadow are respectively

$$s_x = s\cos(\epsilon) \ , \ s_y = s\sin(\epsilon) \ . \tag{8.11}$$

When the camera is not facing north, but rotated anticlockwise by an azimuth angle α, to estimate the rotated shadow components the new x-axis and y-axis are computed by

$$\vec{a}_{x_r} = \mathbf{R} \times [1, 0]^T, \quad \vec{a}_{y_r} = \mathbf{R} \times [0, 1]^T \ ,$$
$$\text{where } \mathbf{R} = \begin{bmatrix} \cos\alpha & -\sin\alpha \\ \sin\alpha & \cos\alpha \end{bmatrix} \tag{8.12}$$

according to Euler's rotation theorem [177]. The rotated shadow components are the projections onto the new axes

$$r_x = [s_x, s_y] \cdot \vec{a}_{x_r} \ , \quad r_y = [s_x, s_y] \cdot \vec{a}_{y_r} \ . \tag{8.13}$$

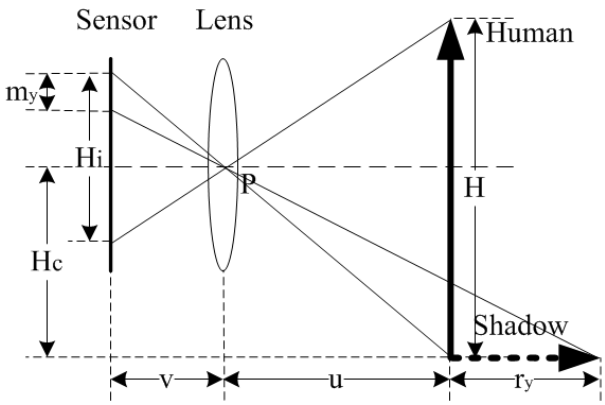


Figure 8.2: Shadow projection on the image plane, looking from side of the camera.

8.2.2 Step 2: Estimating shadow projection

Fig. 8.2 shows shadow projection onto side-view of a scene. If the camera has a large depth of field, all the light ray from one point passing through the lens will be focused at a single point in the image (i.e., the sensor plane). Using properties of similar triangles, the height of the subject image and the vertical component of the length of the shadow image are respectively

$$H_i = H \times \frac{v}{u} \tag{8.14}$$

$$m_y = H_c v \left(\frac{1}{u + r_y} - \frac{1}{u} \right) \quad \text{if } r_y > \frac{2H_c v}{H_s} - u \ , \tag{8.15}$$

where H_c is the height of the camera measured from the ground to the centre of the sensor, v is the separation between the lens and the sensor, u is the distance between the subject and the lens, and H_s is the height of the sensor.

The condition for r_y prevents the problem when the shadow is too close to the camera. Note that m_y is above the horizontal axis when it is negative and vice versa since the vertical axis is reversed in an image/matrix's coordinate system. According to lens law [154]

$$\frac{1}{f} = \frac{1}{u} + \frac{1}{v} \ , \tag{8.16}$$

where f is the focal length of the lens. When u is large compared to f, v is approximately equal to f. This is the case for practical surveillance systems, where u is normally larger than 2 metres and f is approximately less than 5 mm for a wide angle view. Since the input is 2D video without depth information, u is estimated from the width of the silhouette of the person's haed, assuming the top part of the silhouette is the head. This estimation is illustrated in Fig. 8.3, where w is the

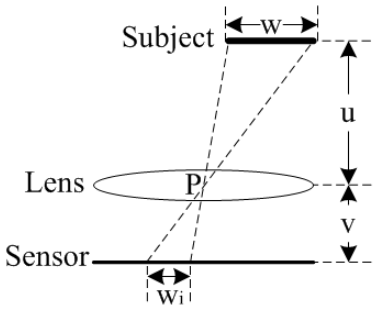


Figure 8.3: Estimating subject's distance from camera lens.

average width of the head, and w_i is the width of the head's image, i.e.,

$$u = \frac{w \cdot v}{w_i} \ , \tag{8.17}$$

where $w_i = w_p \times q$, w_p is the width of the top part of the extracted silhouette in terms of number of pixels and q is the pixel width in metres. This approximation is valid if the top of the shadow is well below the head in the image. The variation of u on the subsequent analysis is discussed in Section 8.2.3.

Referring to Fig. 8.3 and using properties of similar triangles, the projected x-component of the shadow

$$m_x = \frac{r_x \cdot v}{u} \quad \text{if } r_y > \frac{2H_c v}{H_s} - u \ . \tag{8.18}$$

When $r_y \leq 2H_c v/H_s - u$, the vertical component of the shadow on the sensor will always be $2H_c v/H_s - u$. In order to keep the shadow's orientation unchanged, r_x should change accordingly so that r_y and r_x respectively become

$$r'_y = \frac{2H_c v}{H_s} - u \ , \ r'_x = \frac{r_x}{r_y} \times \left(\frac{2H_c v}{H_s} - u \right) \ . \tag{8.19}$$

8.2.3 Step 3: Computing rotational angle and its variance

When the shadow is not pointing vertically downward, the rotation-cutting process is used to remove the shadow. Fig. 8.4 illustrates the process when the shadow appears on the right side of the subject, i.e., $m_x > 0$. The silhouette in Fig. 8.4(a) is first rotated clockwise by an angle β so that the shadow edge has a negative gradient and the body a positive gradient as shown in Fig. 8.4(b). For this to be always the case, y-axis must lies within the angle between the subject and the shadow after rotation as illustrated in Fig. 8.5. We make y-axis bisect this angle so that the

vertical projection is the longest, which is convenient for the gradient analysis of the bottom edge. Thus the shadow angle and rotational angle are respectively

$$\eta = \arctan\left(\frac{m_y}{m_x}\right) \ , \ \beta = \frac{3}{4}\pi - \frac{1}{2}\eta \ . \tag{8.20}$$

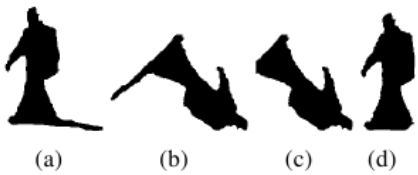


Figure 8.4: Removing shadow: (a) Original silhouette with shadow; (b) original silhouette rotated; (c) rotated silhouette with shadow removed; (d) silhouette rotated back.

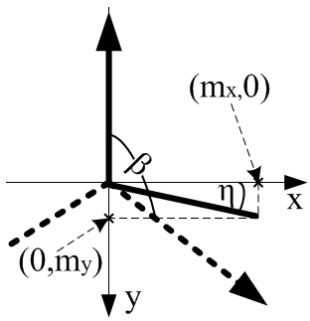


Figure 8.5: Silhouette rotation for shadow removal. Key: bold arrow: standing subject; bold line: shadow; dashed bold arrow: subject after rotation; and dashed bold line: shadow after rotation.

The rotation point of the silhouette is identified and left of that point is considered the shadow, and is removed (Fig. 8.4(c)). The silhouette is then rotated back (Fig. 8.4(d)). The case when $m_x \leq 0$ is dealt with by a flip of the silhouette along the horizontal axis, and followed by another flip along the same axis after shadow removal.

The variation in H, u and the remaining input measurements affect m_x and m_y, which leads to variation in β. This is addressed by analysing β iteratively within the variation range of β. Denote σ_x^2 for the variance of a variable x. When σ_H^2, $\sigma_{H_c}^2$, $\sigma_{H_i}^2$, $\sigma_{H_s}^2$, σ_θ^2, σ_α^2, σ_w^2, $\sigma_{w_i}^2$, σ_v^2, σ_t^2, σ_d^2, $\sigma_{\lambda_l}^2$,$\sigma_{\kappa_l}^2$ are given, $\sigma_{m_x}^2$, $\sigma_{m_y}^2$ and σ_β^2 are derived using the error proportion formula [184]. For linear combination of variables $Y = \sum_i A_i x_i$,

$$\sigma_Y^2 = \sum_i A_i^2 \sigma_{x_i}^2 \ . \tag{8.21}$$

For non-linear combination of variables $Y = f(x_1, x_2, x_3, ...)$,

$$\sigma_Y^2 = \sum_i \left(\frac{\partial Y}{\partial x_i}\right)^2 \sigma_{x_i}^2 \ . \tag{8.22}$$

Using Eqn. (8.2) - (8.10), the variances of κ_s, t_e, ξ, θ, ψ are:

$$\sigma_{\kappa_s}^2 = \left(\frac{b_2 b_3 \cos(b_3 d - b_4)}{\sqrt{1 - (b_2 \sin(b_3 d - b_4))^2}}\right)^2 \sigma_d^2 \ , \tag{8.23}$$

$$\text{where } b_2 = \sin(\frac{23.45\pi}{180}), \ \ b_3 = \frac{2\pi}{365}, \ \ b_4 = \frac{162\pi}{365} \ ;$$

$$\sigma_{t_e}^2 = b_3^2 \sigma_d^2 [384.16 \cos^2(2b_1) + 56.7009 \sin^2(b_1) + 2.25 \cos^2(b_1)] \tag{8.24}$$

$$\sigma_\xi^2 = \left(\frac{\pi}{12}\right)^2 \left[\sigma_t^2 + \frac{1}{3600}(16\sigma_{\lambda_l}^2 + \sigma_{t_e}^2)\right] \tag{8.25}$$

$$\sigma_\theta^2 = \frac{1}{1 - b_5^2}\left[b_5^2 \sigma_{\kappa_s}^2 + b_6^2 \sigma_{\kappa_l}^2 + b_7^2 \sigma_\xi^2\right] \ , \tag{8.26}$$

$$\text{where} \quad \begin{aligned} b_5 &= \sin(\kappa_l)\cos(\kappa_s) - \cos(\kappa_l)\cos(\xi)\sin(\kappa_s), \\ b_6 &= \sin(\kappa_s)\cos(\kappa_l) - \cos(\kappa_s)\cos(\xi)\sin(\kappa_l), \\ b_7 &= \cos(\kappa_s)\cos(\kappa_l)\sin(\xi) \ ; \end{aligned}$$

$$\sigma_\psi^2 = \frac{1}{(1 - b_8^2)\cos^2(\theta)}\left[b_9^2 \sigma_{\kappa_s}^2 + b_{10}^2 \sigma_{\kappa_l}^2 + b_{11}^2 \sigma_\xi^2 + b_{12}^2 \sigma_\theta^2\right] \tag{8.27}$$

$$\text{where } b_8 = \frac{\sin(\kappa_s)\cos(\kappa_l) - \cos(\kappa_s)\sin(\kappa_l)\cos(\xi)}{\cos(\theta)} \ ,$$

$$\begin{aligned} b_9 &= \cos(\kappa_l) cos(\kappa_s) + \sin(\kappa_l)\cos(\xi)\sin(\kappa_s) \ , \\ b_{10} &= \sin(\kappa_s)\sin(\kappa_l) + \cos(\kappa_s)\cos(\xi)\cos(\kappa_l) \ , \\ b_{11} &= \cos(\kappa_s)\sin(\kappa_l)\sin(\xi) \ , \\ b_{12} &= [\sin(\kappa_s)\cos(\kappa_l) - \cos(\kappa_s)\sin(\kappa_l)\cos(\xi)]\tan(\theta) \ . \end{aligned}$$

Since only σ_θ^2 and σ_ψ^2 are used for subsequent calculations, when prior knowledge or appropriate assumption is acquired for them, their computation using Eqn. (8.26) and (8.27) are not necessary. Using Eqn. (8.1) gives

$$\sigma_s^2 = \left(\frac{H \sec^2(\theta)}{\tan^2(\theta)}\right)^2 \sigma_\theta^2 + \left(\frac{1}{\tan(\theta)}\right)^2 \sigma_H^2 \ . \tag{8.28}$$

Using Eqn. (8.11), the variance of s_x and s_y are respectively

$$\sigma_{s_x}^2 = s^2 \sin^2(\epsilon)\sigma_\psi^2 + \cos^2(\epsilon)\sigma_s^2 \tag{8.29}$$

$$\sigma_{s_y}^2 = s^2 \cos^2(\epsilon)\sigma_\psi^2 + \sin^2(\epsilon)\sigma_s^2 \ . \tag{8.30}$$

The variances of r_x, r_y, r'_y and r'_x are then derived as

$$\sigma_{r_x}^2 = \left(\sigma_{s_x}^2 + s_y^2\sigma_\alpha^2\right)\cos^2(\alpha) + \left(\sigma_{s_y}^2 + s_x^2\sigma_\alpha^2\right)\sin^2(\alpha) \tag{8.31}$$

$$\sigma_{r_y}^2 = \left(\sigma_{s_y}^2 + s_x^2\sigma_\alpha^2\right)\cos^2(\alpha) + \left(\sigma_{s_x}^2 + s_y^2\sigma_\alpha^2\right)\sin^2(\alpha) \tag{8.32}$$

$$\sigma_{r'_y}^2 = \left(\frac{2v}{H_s}\right)^2 \sigma_{H_c}^2 + \left(\frac{2H_c}{H_s}\right)^2 \sigma_v^2 + \left(\frac{2H_c v}{H_s^2}\right)^2 \sigma_{H_s}^2 + \sigma_u^2 \tag{8.33}$$

$$\sigma_{r'_x}^2 = (r'_y)^2 \left[\left(\frac{1}{r_y}\right)^2 \sigma_{r_x}^2 + \left(\frac{r_x}{r_y^2}\right)^2 \sigma_{r_y}^2\right] + \left(\frac{r_x}{r_y}\right)^2 \sigma_{r'_y}^2 \ . \tag{8.34}$$

Note that the term $\sigma_{r'_x}^2$ and $\sigma_{r'_y}^2$ respectively replace $\sigma_{r_x}^2$ and $\sigma_{r_y}^2$ if $r_y > 2H_c v/H_s - u - \sqrt{\sigma_{r_y}^2}$. The variances of m_x, m_y and β are in turn obtained as

$$\sigma_{m_x}^2 = \left(\frac{v}{u}\right)^2 \sigma_{r_x}^2 + \left(\frac{r_x}{u}\right)^2 \sigma_v^2 + \left(\frac{r_x v}{u^2}\right)^2 \sigma_u^2 \tag{8.35}$$

$$\sigma_{m_y}^2 = \left(\frac{1}{u+r_y} - \frac{1}{u}\right)^2 \left(v^2\sigma_{H_c}^2 + H_c^2\sigma_v^2\right) + \left(\frac{1}{u^4} + \frac{1}{(u+r_y)^4}\right) H_c^2 v^2 \sigma_u^2 + \frac{H_c^2 v^2 \sigma_{r_y}^2}{(u+r_y)^4} \ . \tag{8.36}$$

$$\sigma_\beta^2 = \frac{m_x^2\sigma_{m_y}^2 + m_y^2\sigma_{m_x}^2}{4(m_x^2+m_y^2)^2} \ . \tag{8.37}$$

Thus, β varies in the range $\left[\beta - \sqrt{\sigma_\beta^2},\ \beta + \sqrt{\sigma_\beta^2}\right]$. Note that Eqn. (8.23) - (8.37) are our derivation.

Eqn. (8.28) shows that the variation in subject height causes variation in shadow length σ_s^2. σ_s^2 causes $\sigma_{s_x}^2$ (Eqn. (8.29)) and $\sigma_{s_y}^2$ (Eqn. (8.30)). These cause $\sigma_{r_x}^2$ (Eqn. (8.31)) and $\sigma_{r_y}^2$ (Eqn. (8.32)), which lead to $\sigma_{m_x}^2$ (Eqn. (8.35)) and $\sigma_{m_y}^2$ (Eqn. (8.36)), and finally lead to σ_β^2 (Eqn. (8.37)). Eqn. (8.35) also shows that the variation in subject distance σ_u^2 causes $\sigma_{m_x}^2$ which affects σ_β^2. Thus, both variation in subject height and distance are addressed.

8.2.4 Step 4: Removing shadow from silhouette

At this stage, the silhouette rotational angle and its variation range are determined. All that remains to do is to determine the optimal rotational angle within its variation range and remove the shadow. The coordinates of every pixel of an silhouette image rotated by an angle β are

$$\begin{bmatrix} x' \\ y' \end{bmatrix} = \begin{bmatrix} \cos\beta & -\sin\beta \\ \sin\beta & \cos\beta \end{bmatrix} \times \begin{bmatrix} x \\ y \end{bmatrix} , \tag{8.38}$$

where (x, y) is the pixel coordinate before rotation. When β is less than π, it is set to be π.

The following process is iterated for the rotational angle $\beta_j = \beta - \sqrt{\sigma_\beta^2}$ to $\beta_j = \beta + \sqrt{\sigma_\beta^2}$, where subscript $_j$ is iteration index and the incrementing value is $10°$:

1. Rotate the silhouette by β_j.
2. Compute gradient vector of bottom edge of the rotated silhouette and smooth it with a unity-gain low pass filter.
3. Find rotation point of the bottom edge, by determining the end of the sequence within the gradient vector with maximum number of successive negative elements, i.e., U_j.

The optimal value of β_j is when U_j is maximised, or when the rotation point is most noticeable.

When η is close to $90°$, or when the shadow is pointing vertically downwards, the rotation point is hardly identifiable. Thus, the identification of this point is difficult. However, with the estimates of H_i and m_y, the shadow are simply removed by removing the bottom m_y/q number of pixels of the extracted silhouette. However, when the subject bends down, the change in H introduces a significant variance in H_i and m_y, thus this simple procedure is not used for all other η values.

8.3 Experiments

A background subtraction function in OpenCV 2.4.8 based on the work in [181] is used to extract the test silhouettes. The proposed shadow removal algorithm is then used to remove the shadows. For comparison, we used the shadow detection program in [40] to generate the results for five different methods based on: chromacity

(Chrm), physical (Phy), geometry (Geo), short-range (SR) texture, long-range (LR) texture. Since our method requires location and time which are not provided by public datasets, we captured the test video in outdoor scene.

Fig. 8.6 shows the comparison results for five scenes. The input parameters are: local longitude = 52.38°; local latitude = −1.56°; number of days since year start in Eqn. (8.3) = 174; time in hour = 11.5; camera angle in Eqn. (8.12) = 140°; and camera height = $0.8m$. Chrm incorrectly identifies large part of the silhouettes as shadow while not completely removes the shadow on ground. Phy removes less incorrect shadow but fails to remove the shadow on ground. Geo manages to remove the majority of the shadow pixels on ground but also removes large parts of the silhouette as shown in the 3rd and 4th rows of Fig. 8.6. SR and LR are not reliable as large part of the silhouettes is removed. In comparison, the proposed algorithm produces silhouettes with the lowest incorrect identification while the majority of the shadow on ground is removed. In addition, it does not leave isolated shadow pixels that are far away from the body (e.g. second scene using LR) which cause problem in silhouette alignment, an important procedure in silhouette embedding.

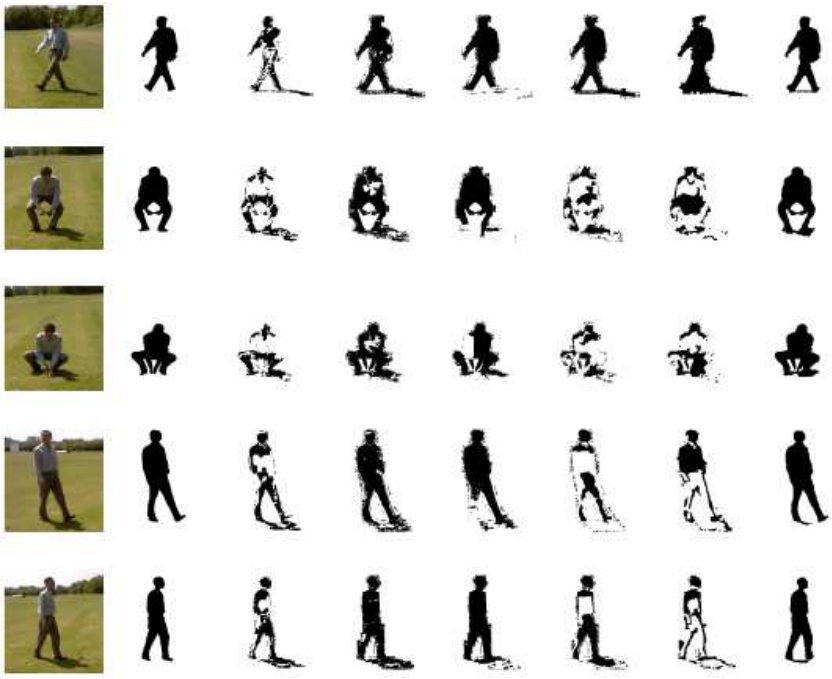


Figure 8.6: Shadow removal: column 1: original frame; column 2: ground truth; column 3-8: results using Chrm, Phy, Geo, SR, LR and the proposed method, respectively.

Table 8.1 shows that on average the proposed method removes the most

Table 8.1: Shadow pixels removed as a percentage of total shadow pixels. S_n are scene numbers.

	S1	S2	S3	S4	S5	Avg.
Chrm	38.5	37.2	38.5	24.7	30.1	33.8
Phy	10.5	28.0	25.8	31.4	24.3	24.0
Geo	82.7	82.7	50.5	56.7	59.6	66.4
SR	1.2	55.9	55.5	96.0	32.6	48.2
LR	0.7	78.5	64.9	79.4	73.2	59.3
Proposed	68.6	63.8	48.1	88.1	85.9	70.9

Table 8.2: Shadow pixels incorrectly removed as a percentage of total silhouette pixels. S_n are scene numbers.

	S1	S2	S3	S4	S5	Avg.
Chrm	41.8	51.4	52.6	41.0	39.2	45.2
Phy	6.3	12.5	14.2	6.4	7.0	9.3
Geo	1.6	4.4	20.4	8.2	0.0	6.9
SR	0.9	53.7	54.6	52.7	34.0	39.2
LR	0.8	47.2	45.5	36.6	51.7	36.4
Proposed	0.7	1.9	3.2	2.7	4.1	2.5

shadow pixels. Table 8.2 shows it also removes the least number of shadow pixels incorrectly. Thus, the proposed algorithm is both effective in removing shadow and reliable in retaining the original silhouette.

8.4 Summary

In this chapter, a shadow removal algorithm based on known position of the sun is presented. First, the shadow is determined with the current time and location information. Second, the shadow is projected onto the image plane. Third, the entire silhouette is then rotated by an angle estimated with the shadow information, and the shadow part of the silhouette is removed. This algorithm uses known position of the sun rather than the texture and colour information used by most alternative algorithms. Hence it can cope with the problem when the shadow has similar texture and colour saturation as the surrounding background. Moreover, it leaves no isolated shadow pixels that are far away from the body. Experiments show that the proposed algorithm removes high portion of the shadow and low potion of the body compared to five existing algorithms.

Chapter 9

Human Activity Recognition using Embedded Silhouettes

9.1 Introduction

Up to now, training silhouette generation for off-line preparation and shadow removal for run-time input improvement have been respectively presented in Chapter 7 and 8. This chapter describes the main component of the recognition system, i.e. the recognition algorithm.

A video-based human activity recognition system aims to categorise activities captured in video sequences [185], and has been an active research area, e.g., for visual surveillance, patient monitoring in hospitals and human-machine interaction. Various problems are encountered by an activity recognition system. When the temporal order of the actions comprising an activity is changed, new training data is required for that activity to be recognised, e.g., as in sequential methods that model activities with the HMM [25; 33]. When the speed in which an activity is performed is changing, the template used for its recognition has to be altered [22] as in the spatio-temporal volume approaches [41; 42]. Real-time operation with high accuracy and robustness are often necessary. However, methods based on space-time trajectory [43; 44] are very slow because they require accurate 3D modelling of a large number of body parts. They also have problem dealing with occlusion of joints.

We address these problems by proposing the EPL algorithm which uses a spatial object created from the coordinates of embedded silhouettes to denote an activity. The spatial object takes into account the speed variation of the actions and is invariant to their temporal order. The algorithm is fast due to its linear

nature. The chapter is organised as follows. Previous related work is presented in Section 9.2. The theory of our algorithm is presented in Section 9.3. Section 9.4 presents the experimental results while Section 9.5 summarises the chapter.

9.2 Related Work

The dimensionality of the information associated with human activity is high. Thus, dimensionality reduction is needed that preserves the discriminating data and removes noise. Previous activity recognition methods have adopted linear dimensionality reduction methods, e.g., PCA [28] and locality preserving projections [186]. These methods generate direct mapping to an embedding space. However, human activity is highly complex and non-linear. Non-linear dimensionality reduction methods including Isomap [31] and Local Linear Embedding [30] provided considerable improvement for recognition in [33; 34; 35].

The silhouettes after dimensionality reduction, i.e., the embedded silhouettes, need to be configured as the activity model. The system in [25] employs a HMM, where each video frame is transformed into a feature vector with mesh-grid, which are then clustered, with each centre being the codeword. Each codeword corresponds to an output symbol in the HMM and every feature vector is assigned to its nearest symbol. For a test video, the forward-algorithm is used to estimate the similarity between the symbol sequence and the gallery activities. Latter HMM based systems include those in [128; 187]. However, many activities are non-deterministic, where elements of an activity can be in any temporal order [188], thus requiring more training data for different orders. The two alternatives to HMM are linear dynamical system and non-linear dynamic system [22]. A linear dynamical system is a more general form of HMMs where the state space can take continuous values. Non-linear dynamic system is a more general form of linear dynamical system which is a combination of several linear dynamical systems. Thus, these alternatives are more complex than HMM. We propose the use of pattern of embedded silhouettes for activity recognition.

9.3 Theory

The major steps of the proposed recognition system are: (1) silhouettes are extracted from a video containing a person performing an activity; (2) shadows are removed from the silhouettes with the algorithm in Chapter 8; (3) shadow-free silhouettes are embedded to a low-dimensional space; (4) the activity is classified with pattern of the embedded silhouettes.

9.3.1 Silhouette embedding

For efficient activity recognition, we transform silhouettes to a lower dimensional space to generate embedded silhouettes, and determine a mapping that maps an unknown silhouette onto this space. Fig. 9.1 shows an overview of the embedding algorithm. First, the gallery (training) silhouettes, obtained by the algorithm in Chapter 7, are processed by the centroid distance function of a shape descriptor to form the silhouette feature-vectors. This set of vectors is further processed by Isomap. RBF [148] is used to learn the mapping from the high-dimensional vector space to the corresponding low-dimensional which produces a set of RBF parameters. During testing (i.e., recognising an activity), a silhouette is extracted from each frame of a test video, and its shaped described and embedded to the low-dimensional space with the RBF mapping.

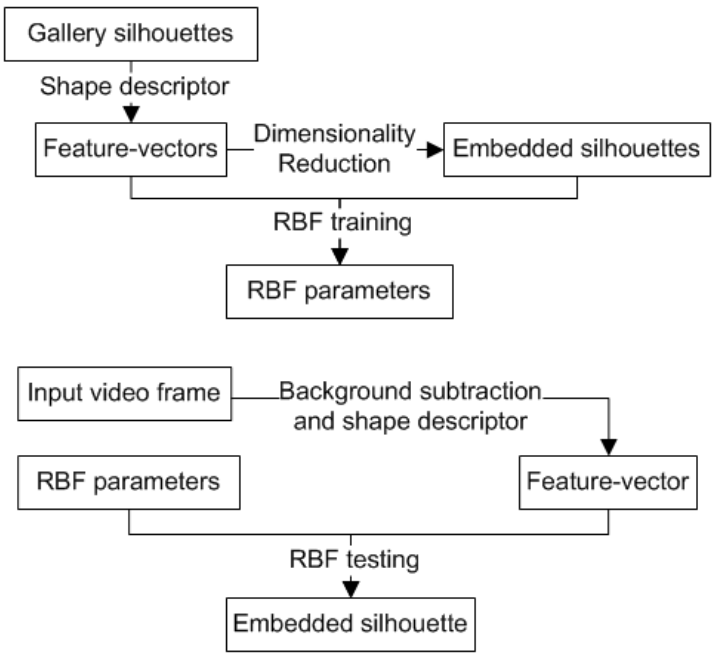


Figure 9.1: Learning for frame-by-frame silhouette embedding. Top: training; and bottom: testing.

Low-dimensional representation of silhouettes

The silhouettes are first scaled to a 100×100 binary image $\mathbf{B}$. The centroid distance function, a function of the distance between the centroid of a shape to its boundary points at all angles, is used as the shape descriptor to convert $\mathbf{B}$ into 1×360 feature vector $\vec{v}$ for real-time performance. The angles are divided equally by 360 degrees,

and the horizontal and vertical indices of boundary pixels are found using Moore-Neighbour tracing [174].

Any broken silhouettes make the tracing algorithm return multiple sets of boundaries. To address this problem, only the outermost boundary point at every angle is used to compute centroid distance function. Fig. 9.2 shows some centroid distance functions and their corresponding silhouette. The horizontal axis of each function represents the angle [1°,360°] and the vertical axis represents the normalised distance [0, 0.7]. All functions are generated using the same scale. They are significantly different for different silhouettes, and their general shape is maintained even if the silhouettes are severely broken.

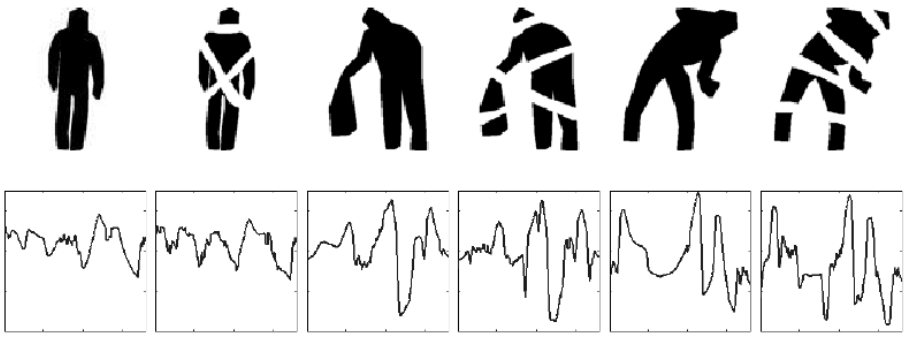


Figure 9.2: Silhouettes (row 1) and their centroid distance function (row 2). Odd columns: original silhouettes; and even columns: broken silhouettes.

N_s training silhouettes are converted into N_s feature vectors and combined into an $N_s \times 360$ matrix $\mathbf{X}$. Isomap is then applied to reduce its dimensionality to D, i.e.,

$$f : R^{N_s \times 360} \mapsto R^{N_s \times D} \ , \tag{9.1}$$

to generate the embedded silhouettes $N_s \times D$ matrix

$$\mathbf{Y} = f(\mathbf{X}) \ . \tag{9.2}$$

Fig. 9.3 shows the embedded data (denoted by dots) generated using 8 views corresponding to 1500 feature vectors. For simplicity, a 2D plot is generated with the first two dimensions. Some silhouettes are also superimposed near their corresponding location to illustrate that Isomap makes similar silhouettes to be close to one another.

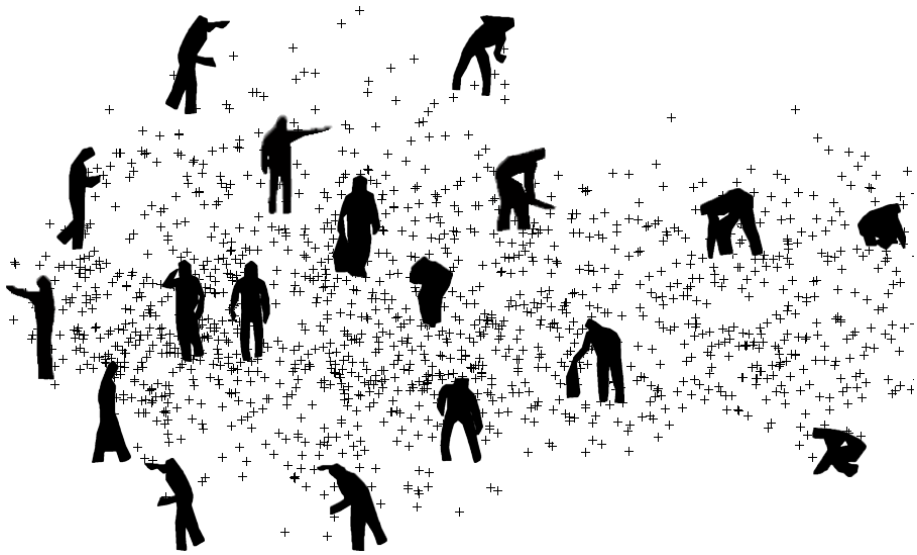


Figure 9.3: The embedded data generated using Isomap.

Learn the embedding mapping with RBF

A test silhouette needs to be transformed onto the embedding space so that it can be compared with the embedded gallery silhouettes to determine its activity. We use RBF to learn the mapping f in Eqn. (9.1) and (9.2), i.e.,

$$\vec{y} = \sum_{m=1}^{M} \vec{w}(m) \cdot \beta(\gamma|\vec{x} - \vec{c}_m|^2) \ , \tag{9.3}$$

where $\vec{y}$ and $\vec{x}$ are respectively an example of $\mathbf{Y}$ and $\mathbf{X}$, $\vec{c}_m$ is mth centre point in the feature-vector space, M is number of centre points, $\vec{w}(m)$ is weight associated with $\vec{c}_m$, $\beta()$ is the basis function and γ is the coefficient of $\beta()$. $|\vec{x}-\vec{c}_m|^2$ is the Euclidean distance between $\vec{x}$ and $\vec{c}_m$. The RBS parameters $\vec{c}_m$, $\vec{w}$ and γ are determined during training using $\mathbf{X}$ and $\mathbf{Y}$. Since $\beta()$ has coefficient γ and $\beta()$ is a function of $\vec{x}$ and $\vec{c}_m$, it is rewritten as $\beta_\gamma(\vec{x}, \vec{c})$ for simplicity. Let $\vec{y}_n$ denote the nth sample of $\mathbf{Y}$, and Eqn. (9.3) is rewritten as

$$\begin{bmatrix} \vec{y}_1 \\ \vec{y}_2 \\ ... \\ \vec{y}_N \end{bmatrix} = \begin{bmatrix} \beta_\gamma(\vec{x}_1, \vec{c}_1) & \beta_\gamma(\vec{x}_1, \vec{c}_2) & ... & \beta_\gamma(\vec{x}_1, \vec{c}_M) \\ \beta_\gamma(\vec{x}_2, \vec{c}_1) & \beta_\gamma(\vec{x}_2, \vec{c}_2) & ... & \beta_\gamma(\vec{x}_2, \vec{c}_M) \\ ... & ... & ... & ... \\ \beta_\gamma(\vec{x}_N, \vec{c}_1) & \beta_\gamma(\vec{x}_N, \vec{c}_2) & ... & \beta_\gamma(\vec{x}_N, \vec{c}_M) \end{bmatrix} \times \begin{bmatrix} w_1 \\ w_2 \\ ... \\ w_M \end{bmatrix} . \tag{9.4}$$

However, Eqn. (9.4) is only suitable if the dimensionality of $\vec{y}$ is 1. To address this, let $\vec{w}_m$ denote the $1 \times D$ weight vector associated with mth centre point to give

$$\begin{bmatrix} \vec{y}_1 \\ \vec{y}_2 \\ ... \\ \vec{y}_N \end{bmatrix} = \begin{bmatrix} \beta_\gamma(\vec{x}_1, \vec{c}_1) & \beta_\gamma(\vec{x}_1, \vec{c}_2) & ... & \beta_\gamma(\vec{x}_1, \vec{c}_M) \\ \beta_\gamma(\vec{x}_2, \vec{c}_1) & \beta_\gamma(\vec{x}_2, \vec{c}_2) & ... & \beta_\gamma(\vec{x}_2, \vec{c}_M) \\ ... & ... & ... & ... \\ \beta_\gamma(\vec{x}_N, \vec{c}_1) & \beta_\gamma(\vec{x}_N, \vec{c}_2) & ... & \beta_\gamma(\vec{x}_N, \vec{c}_M) \end{bmatrix} \times \begin{bmatrix} \vec{w}_1 \\ \vec{w}_2 \\ ... \\ \vec{w}_M \end{bmatrix} \tag{9.5}$$

$$\mathbf{Y} = \mathbf{B} \times \mathbf{W} \ . \tag{9.6}$$

The training process is as follows. The M number of centre points in the feature-vector space are found using k-means clustering to compute $\mathbf{B}$. For higher accuracy, we use all samples of $\mathbf{X}$ as the centre points. Among basis functions, e.g., Gaussian, biharmonic and trihamonic spline, we found Gaussian gives good results, where its coefficient γ is determined using 10-fold cross-validation. Finally the weight matrix

$$\mathbf{W} = \mathbf{B}^{-1} \times \mathbf{Y} \ , \tag{9.7}$$

where $\mathbf{B}^{-1}$ is the inverse of $\mathbf{B}$ if $\mathbf{B}$ is a square matrix or the pseudo-inverse if otherwise. Note that during testing, $\mathbf{B}$ is calculated first, and followed by Eqn. (9.6).

9.3.2 Embedded pattern learning

Fig. 9.4 outlines the EPL. During training, all silhouettes are embedded using RBF. All embedded silhouettes of each activity form a pattern of embedded silhouette (PES). Each PES is converted to a gallery spatial object. During testing, the PES and test spatial object of an unknown activity are similarly obtained from a test video. The test spatial object is then compared with the gallery spatial objects using k-nearest-neighbour classifier to identify its activity.

Patterns of embedded silhouettes

Fig. 9.5 illustrates the PESs of six activities which are shown as black dots. The background embedded silhouettes in each sub-figure shown in grey dots are the same as those in Fig. 9.3 and are drawn to the same scale. Thus refer to Fig. 9.3 for the following description of activities. Walk contains the least variety of silhouettes, which is narrowly distributed near the walking posture region. Drop contains bending, dropping and walking postures. Crou contains crouching and walking postures as well as some bending postures. Bag contains bag carrying and walking postures. Shoot contains shooting postures. Dig contains a variety of postures include walk-

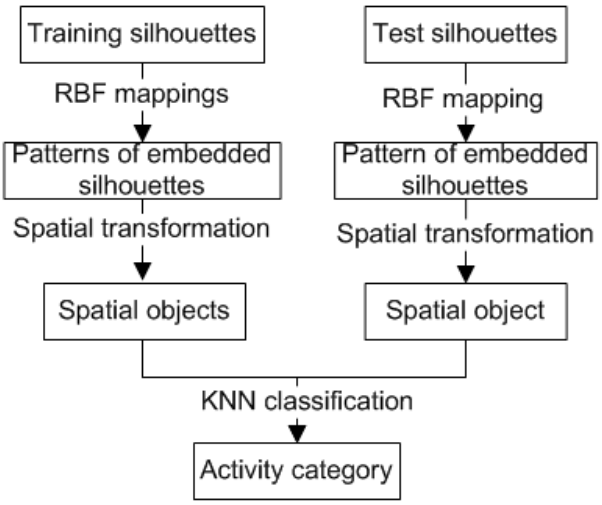


Figure 9.4: Training (left) and testing (right) using EPL.

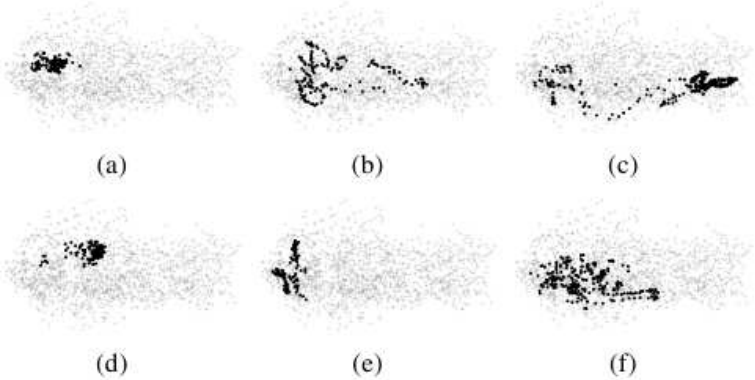


Figure 9.5: PES for (a) Walk, (b) Drop, (c) Crou, (d) Bag, (e) Shoot, and (f) Dig.

ing, bending and leaning down. Note that the PES of one activity is distinguishable from another.

Spatial object formation and activity recognition

Let D denote the dimensionality of each embedded point, and $\vec{e}_f$ denote a $1 \times D$ vector containing the embedded data for frame f and F denote the frame number. The matrix form of the PES for one video sample, i.e., embedded pattern matrix (EPM) is

$$\mathbf{E} = [\vec{e}_1 \ \vec{e}_2 \ ... \ \vec{e}_F]^{\mathrm{T}} \ , \tag{9.8}$$

where T is the transpose operator. Note that F can be different for different video.

Let $\mathbf{E}_v$ denote the vth EPM and V denote the number of EPMs in each

activity category, the dataset for one category

$$\mathbf{G} = [\mathbf{E}_1 \ \mathbf{E}_2 \ ... \ \mathbf{E}_V]^{\mathrm{T}} \ . \tag{9.9}$$

Let $\mathbf{G}_a$ denote the ath category dataset and A denote the number of categories, the overall dataset $\mathbf{H}$ is

$$\mathbf{H} = [\mathbf{G}_1 \ \mathbf{G}_2 \ ... \ \mathbf{G}_A]^{\mathrm{T}} \ . \tag{9.10}$$

Every EPM is first discretised, which makes each value to the index of the closest member in a predefined group of numbers. Each member of the group is

$$p_n = \frac{nR}{N_t} + \min \mathbf{Y}, \forall n = 1, 2, ..., N_t \ , \tag{9.11}$$

where N_t is the number of members, R is the range of the embedded silhouettes $\mathbf{Y}$. Small N_t increases the robustness against noise but at the cost of lower sensitivity, and larger N_t increases the sensitivity but compromises on noise performance. N_t is determined by cross-validation to achieve a reasonable robustness and noise performance.

Each element of $\mathbf{E}_v$ is assigned the index n of the closest p_n. Let the corresponding discretised matrix be $\hat{\mathbf{E}}_v$, where each of its elements is

$$\hat{\mathbf{E}}_v(f, d) = \underset{Pn}{\operatorname{argmin}} \left| \mathbf{E}_v(f, d) - p_n \right| \ , \tag{9.12}$$

and d is the dimension index. As an illustration, an image is used as a spatial object, which is initialised as a $N_t \times N_t$ zero matrix $\mathbf{O}$. For the extracted silhouette of each frame of a video, i.e., each row of $\hat{\mathbf{E}}_v$, its first and the second element are respectively the horizontal and vertical index of a pixel in spatial object $\mathbf{O}$ to be set to 1, i.e.,

$$\mathbf{O}\left(\hat{\mathbf{E}}_v(f, 1), \hat{\mathbf{E}}_v(f, 2)\right) = 1, \forall f = 1, 2, ..., F \ . \tag{9.13}$$

This indicates that the frame number F only affects how many pixels in $\mathbf{O}$ is 1 rather than its size. This process is repeated for all $\mathbf{E}_v$.

Fig. 9.6 illustrates the spatial objects that correspond to the PESs in Fig. 9.5, where black pixels denote 1's and grey pixels denote the background silhouettes. Note that the shape of each spatial object maintains the distinguishable features of an activity. Every $\hat{\mathbf{E}}_v$ is transformed to a $1 \times N_t^2$ vector $\vec{\hat{e}}_v$. All $\vec{\hat{e}}_v$ for the PESs of each activity are then concatenated to give

$$\hat{\mathbf{G}} = \left[\vec{\hat{e}}_1 \ \vec{\hat{e}}_2 \ ... \ \vec{\hat{e}}_V\right]^{\mathrm{T}} \ . \tag{9.14}$$

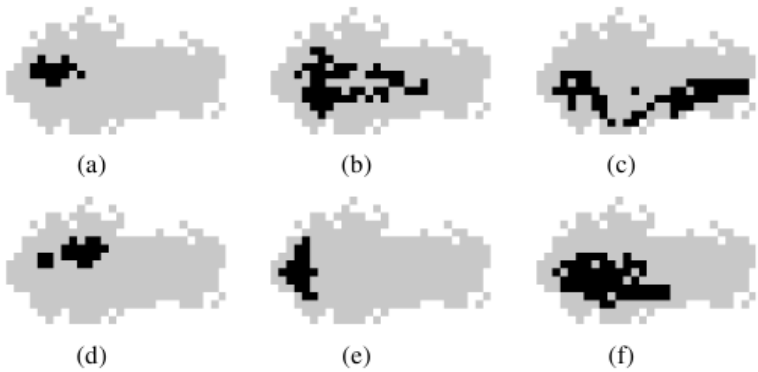


Figure 9.6: Spatial objects (denoted in black) with embedded silhouettes (denoted in grey) for (a) Walk, (b) Drop, (c) Crou, (d) Bag, (e) Shoot, and (f) Dig.

Note that the embedded silhouettes are neither shift-invariant nor rotational-invariant. Thus, any shift or rotation of an embedded data point will make it a different posture which in turn generates a PES which corresponds to a different activity. Thus, shape descriptors based on these two properties cannot be used. Furthermore, for an illustration only two dimensions are plotted while in practice more dimensions are needed to identify numerous activities. In order to generalise the solution for any dimensions and keep it simple, we do not use any shape descriptor before further processing.

Every $\hat{\mathbf{G}}_a$ for different activities, where the subscript a is the activity label, is combined to form the gallery embedded activity matrix

$$\hat{\mathbf{H}} = \left[\hat{\mathbf{G}}_1 \ \hat{\mathbf{G}}_2 \ ... \ \hat{\mathbf{G}}_A\right]^{\mathrm{T}} . \tag{9.15}$$

During testing (i.e. recognising an activity), all frames of a video are embedded with RBF. Every resulting PES is converted into a spatial object $\hat{\mathbf{E}}_t$ which is transformed to $\vec{\hat{e}}_t$. The k-nearest-neighbour classifier [189] is then used to classify $\vec{\hat{e}}_t$ within $\hat{\mathbf{H}}$.

More identical discretised PESs are produced for the same activity performed at low speed than at high speed. However, the same discretised PES is only registered once in the spatial object as shown in Eqn. (9.13). Thus, the PES for the same activity at low and high speed is identical as long as all the associate silhouettes are obtained. Hence EPL is speed-invariant. Since time label is not used in Eqn. (9.13), the PES for the same activity with different temporal order of actions is identical. Thus, EPL is also invariant to the temporal order of actions.

9.4 Experiments and Discussions

The proposed activity recognition system is implemented on a standard desktop computer with Core i7-2600 processor at 3.40 GHz and 4.00 GB memory. An OpenCV function based on the work in [181] is used for background subtraction. The other components of the system are implemented on MATLAB R2013b. Silhouettes of size 100×100 generated by the algorithm in Chapter 7 are used as training data. The centres of the RBFs are chosen as the same as the input vectors. The coefficient γ of the Gaussian basis function for RBF is evaluated with 10-fold cross-validation. The number of neighbours in Isomap is set to be the same as the total number of training vectors. The reduced dimension is set to 3, since our experiments showed that smaller values failed to distinguish all activities well, and any value above 4 inclusive did not provide significant improvement.

For performance evaluation of EPL, 64% [108; 112] of the subjects from a given video database is used to determine the resolution of the spatial object N_t in Eqn. (9.11) using 10-fold cross-validation. The remaining 36% is used for evaluation of the overall accuracy. Let N denotes the number of subjects in evaluation set. EPL is first trained with $N-1$ subjects and then tested on the remaining subjects. As a result, N sets of accuracy results are generated, which are then averaged to give the overall set of accuracy results. In other words, we adopt leave-one-out cross validation.

Our dataset

Publicly available datasets for complex activities are not usable since the background subtraction requires the subject not to appear at the beginning of the video. Thus, we created our own dataset. Our dataset includes 21 actors performing the ten activities in Table 7.1 captured by four cameras at 4 views. Each video is of resolution 640×480. The video length varies between 8-30 seconds. One sample frame from each activity is shown in Fig. 9.7. For RBF training, 1500 silhouettes are selected as the training data. γ is found to be 1×10^{-4}. For EPL, N_t in Eqn. (9.11) are respectively found to be 15, 9 and 9 for the first, second and third dimension. Since this dataset is obtained in-door and no shadow is observed in the test silhouettes, no shadow removal is required.

Table 9.1 shows the confusion matrix which represents the performance of the proposed system. Drop is sometimes mis-classified as Bag because both activities involve carrying a bag, and the subject's arm is occluded during dropping the bag. It is mis-classified as Wall since both activities contain the postures of raising one's

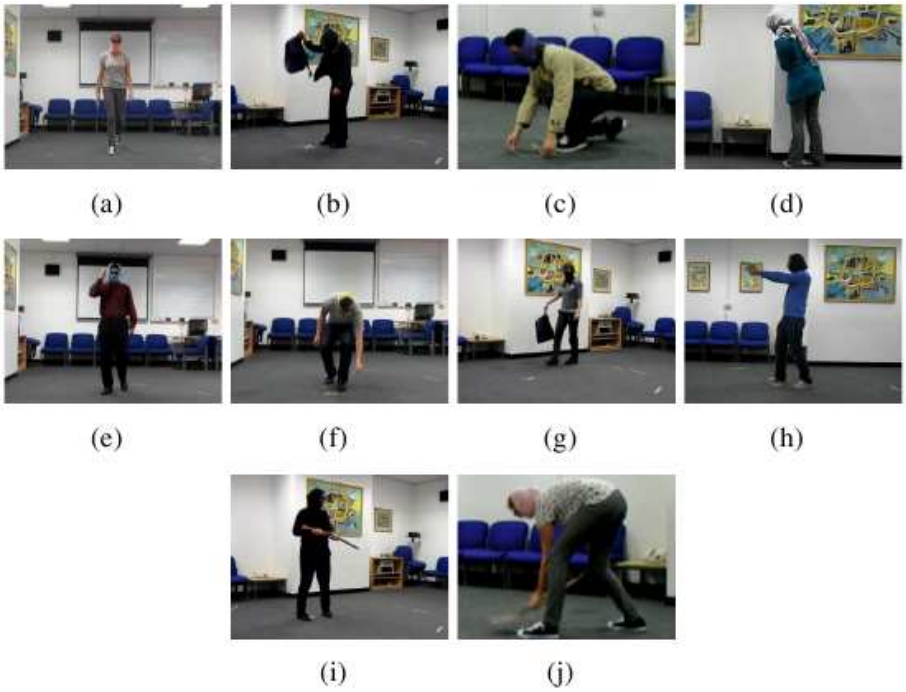


Figure 9.7: Sample frame of activities. (a)-(j): Walk, Drop, Crou, Peer, Mbl, Place, Bag, Shoot, Gun and Dig.

Table 9.1: Confusion matrix using Proposed on our dataset.

	Walk	Drop	Crou	Wall	Mbl	Place	Bag	Shoot	Gun	Dig
Walk	88.8	0.0	0.0	0.0	0.0	3.2	0.0	4.9	3.1	0.0
Drop	0.0	60.2	0.0	5.8	9.8	5.8	15.0	0.0	0.0	3.3
Crou	0.0	0.0	94.8	0.0	0.0	5.2	0.0	0.0	0.0	0.0
Peer	1.7	1.7	0.0	80.4	3.2	0.0	6.5	1.7	0.0	4.9
Mbl	13.9	0.0	0.0	0.0	71.9	0.0	2.0	8.0	4.2	0.0
Place	4.6	4.6	0.0	0.0	4.6	76.9	0.0	4.6	1.5	3.1
Bag	0.0	4.7	0.0	2.5	2.5	0.0	90.3	0.0	0.0	0.0
Shoot	17.2	0.0	0.0	0.0	4.7	0.0	0.0	73.8	4.2	0.0
Gun	8.0	0.0	0.0	0.0	6.9	0.0	0.0	8.0	77.1	0.0
Dig	0.0	3.6	0.0	0.0	7.3	12.9	2.0	0.0	0.0	74.2

arm. Drop is also mis-classified as Mbl possibly due to both activities contain walking postures. Mbl is sometimes mis-classified as Walk because there is only subtle difference between the two activities. Mbl is also mis-classified as Shoot since both activities contain raising one's arm. Shoot is sometimes mis-classified as Walk due to the body shape of different actors. Dig is mis-classified as Place occasionally, because both activities involve bending posture. Despite these problems, the overall recognition accuracy of 78.8% is encouraging considering the complex activities.

KTH and Weizmann datasets

KTH dataset [112] contains 6 actions of Walk, Jog, Run, Box, Hand-wave (Hwv) and Hand-clap (Hcp). These actions are performed with 4 scene variations by 25 subjects. For Walk, Jog and Run the silhouettes are extracted using the algorithm in [190], and for the other actions the silhouettes are extracted by a simple thresholding in colour-space. Since we do not have the Vicon Nexus data for all the actions in KTH, 1500 silhouettes are selected from all extracted silhouettes including those generated using Vicon Nexus to form the training data for silhouette embedding. For RBF training, γ is found to be 1×10^{-5}. For EPL, N_t are respectively found to be 9, 9 and 9 for the first, second and third dimension. For some videos where strong shadow are present in outdoor such as Jog of actor 10, the proposed shadow removal is used where the rotation angle is manually measured from the video.

Table 9.2 shows the confusion matrix. The proposed system performs better for the last three actions. This is because when compared to the first three actions, the silhouettes of these three actions are significantly more distinguishable. For the other three actions: Walk is sometimes misclassified as Jog; Jog is sometimes misclassified as Run; and Run is sometimes misclassified as Jog. These errors are reasonable since even humans sometimes find it difficult to distinguish these actions, and the system is able to produce a mean accuracy of 87.4%.

Table 9.2: Confusion matrix using Proposed on KTH dataset.

	Walk	Jog	Run	Box	Hwv	Hcp
Walk	73.3	18.6	5.8	2.3	0.0	0.0
Jog	0.0	76.4	22.5	0.0	1.1	0.0
Run	0.0	10.6	87.2	0.0	2.1	0.0
Box	0.0	4.7	1.6	92.2	0.0	1.6
Hwv	0.0	0.0	0.0	0.0	98.6	1.4
Hcp	0.0	0.0	0.0	2.2	1.1	96.7

The Weizmann dataset [104] contains 10 actions performed by 9 subjects including Bend, Jump-jack (Jack), Jump-forward-on-two-legs (Jump), Jump-in-place-

Table 9.3: Confusion matrix using Proposed on Weizmann dataset.

	Bend	Jack	Jump	Pjp	Run	Side	Skip	Walk	Wv1	Wv2
Bend	100	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0
Jack	0.0	91.7	0.0	8.3	0.0	0.0	0.0	0.0	0.0	0.0
Jump	11.9	0.0	79.8	8.3	0.0	0.0	0.0	0.0	0.0	0.0
Pjp	0.0	0.0	0.0	100	0.0	0.0	0.0	0.0	0.0	0.0
Run	0.0	0.0	0.0	0.0	85.7	0.0	14.3	0.0	0.0	0.0
Side	0.0	0.0	0.0	0.0	0.0	100	0.0	0.0	0.0	0.0
Skip	0.0	0.0	10.7	0.0	0.0	0.0	89.3	0.0	0.0	0.0
Walk	0.0	0.0	0.0	0.0	0.0	14.3	0.0	85.7	0.0	0.0
Wv1	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	100	0.0
Wv2	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	100

on-two-legs (Pjp), Run, Gallop-sideways (Side), Skip, Walk, Wave-one-hand (Wv1) and Wave-two-hands (Wv2). It provides silhouettes. 1000 silhouettes are selected as training data for silhouette embedding. For RBF training, γ is found to be 1×10^{-8}. For EPL, N_t are respectively found to be 9, 9 and 9 for the first, second and third dimension.

Table 9.3 shows the mean confusion matrix. Jump is occasionally misclassified as Bend since both actions include bending. Run is sometimes misclassified as Skip when their subtle difference is not detected. Skip is sometimes misclassified as Jump, a similar action. Walk is sometimes misclassified as Side due to their similar nature. Despite these errors, a mean accuracy of 93.2% is achieved.

9.4.1 Computational cost

The training on our dataset requires approximately 30 hours. For testing, data preprocessing including background subtraction and silhouette centring requires 38.3 milliseconds per frame. If strong shadow is present in outdoor scenes, the proposed shadow removal method in Chapter 8 requires 283.8 milliseconds per frame for a 20° increment on β in Eqn. (8.20). The time for embedding the silhouette is 51.5 milliseconds per frame. EPL requires 11.1 milliseconds per frame. Therefore the total testing time is 384.7 milliseconds per frame including shadow removal, or 100.9 milliseconds per frame excluding shadow removal.

The algorithm in [120] involves calculating optical flow, kinematic features, kinematic modes and embedding the kinematic modes. All these processes take at least 1.22 seconds per frame. The typical run-time speed of [44] is 1.8 frames per second, which is equivalent to 566.7 millisends per frame. The computational cost of the algorithms in [151] and [24] are not given in [151; 24].

Table 9.4: Perfomance of 5 methods on KTH dataset.

Mtd.	Walk	Jog	Run	Box	Hwv	Hcp	Avg.
[120]	89.1	86.2	91.5	88.5	84.5	86.4	87.7
[151]	97.2	100	83.3	97.2	86.1	94.4	93.1
[44]	-	-	-	-	-	-	95.3
[24]	97	84	79	90	97	94	90.2
Ours	73.3	76.4	87.2	92.2	98.6	96.7	87.4

9.4.2 Comparison with state-of-the-art methods

We compare the proposed system with four methods based on space-time interest points [151], optical flow [120], body point trajectory [44] and CNN [24]. Table 9.4 shows the comparison using KTH dataset. Overall, the performance of the proposed system for Box, Hwv and Hcp are higher or comparable to the other methods, but lower for Walk, Jog and Run. This is because the proposed system does not use speed or temporal information unlike the other methods. The method in [151] extracts and tracks body features such as moving hands. Since the changing speed for these features is lower for Walk than Jog, the resulting Hankel matrix of extracted interest points used for action classification is considerably different. The method in [120] extracts kinematic features of divergence, vorticity, symmetric and antisymmetric flow fields, etc., for classification and is thus capable of distinguishing actions based on speed. The method in [44] uses dense trajectories that are tracked with a dense optical flow algorithm. The 3D CNN in [24] uses feature vectors containing motion information. Nevertheless, the average accuracy of 87.4% achieved by the propose system is comparable to the others.

Of the four methods being compared, only the method in [120] is also evaluated on the Weismann dataset, with an overall accuracies of 94.8%. The accuracy of the proposed system is 93.2%, and is thus comparable. Note however the following. For more complex activities, more interest points are required for the method in [151] which increases their computational cost considerably. The method in [120] can only cope with simple periodic actions. The method in [44] is slow since it requires sampling and tracking of dense interest points. Finally, the method in [24], which only uses 7 or 9 selected frames, cannot recognise more complex activities. In contrast, the proposed system is designed for fast recognition of more complex activities such as those in Table 7.1. It does not require the activity to be performed periodically. It is simple and fast with speed of up to 100.9 milliseconds per frame with serial implementation. Immediate recognition is potentially possible with a parallel implementation.

9.5 Summary

In this chapter, a silhouette-based human activity recognition system is presented. The system includes an improved version of the centroid distance function as the shape descriptor for the training silhouettes, which is robust against broken silhouettes. The use of Isomap captures the global characteristic and nonlinearity of the embedded silhouettes. The required mapping from silhouettes to embedded ones is evaluated with a simplified version of RBF. The embedded patterns are then learned with the proposed EPL, which is invariant to the speed in which an activity is performed and the temporal order of its actions. It is also able to recognise activities more complex than simple periodic actions that are often used to evaluate existing action recognition systems.

The current limitations of the proposed system include the following. Sudden change of illumination and reflections often cause problems in background subtraction. The essential part of the silhouettes such as head and feet are sometimes treated as noise and thus removed. Finally the system is only applicable to single subject. Possible future work thus includes the following investigations: use of a more robust background subtraction based on depth video; extending the activity recognition to multiple subjects; optimising the input parameters of Isomap with a method such as [191] or exploring other dimensionality reduction methods that are more effective than Isomap; and the use of layered EPL to recognise even more complex activities.

Chapter 10

Conclusions and Future Work

10.1 Conclusions

In this thesis, two important areas of 3D computer vision have been studied, namely 3D surface reconstruction based on passive RO-based DfD, and a human activity recognition system based on manifold learning. Chapter 1 has introduced both areas and presents our motivations to undertake the study. The research interest on RO-DfD is motivated by its advantages over other surface reconstruction techniques: (1) it does not require active pattern projections and hence it is cheaper and easier to implement; (2) it is able to produce dense depth map with low computational costs and does not involve problems that occurs in other passive techniques such as illumination changes and correspondence searching. (3) it enables detailed analysis of the input frequency components which makes it produce relatively high accuracy. The research interest on manifold-learning-based human activity recognition system is motivated by its potential applications including content-based video analysis, human-machine interaction, patients monitoring, safety, security and surveillance.

Chapter 2 has provided a thorough literature review on passive DfD techniques. DfD methods using a single image fails to produce high accuracy since it does not have sufficient information to determine the source of blur. DfD methods using two images thus provide significant improvement. Fourier domain approach computes depth by extracting the defocus parameter using Fourier transform. Spatial-filtering approach computes the defocus map in spatial domain with 2D convolutions. Probabilistic approach obtains a depth map by first creating a cost function using a specific statistical model and then computes depths iteratively. Machine learning-based approach creates a library of sample image pairs with known depth values and computes the depth of an input image pair by a form of matching operation. Except RO-DfD, none of these methods address the frequency dependency problem

with low computational costs.

Chapter 3 has presented a literature review on human action/activity recognition approaches classified into two categories, i.e. the spatio-temporal approach and the sequential one. The spatio-temporal approach analyses a video as a 3D volume using body volumes, interest points or optical flows. It achieves acceptable performance for simple and periodic actions. The sequential approach extracts feature vectors from every frame and considers the video as a sequence of features, and effectively handles the non-periodicity of actions/activities. It is further divided into state model-based approach that achieve high accuracy while requiring large amount of training data, and the sequential approach where frame-by-frame features can be flexibly organised to form an action template. Amongst these approaches, manifold-learning based approach is designed for the high dimensionality of human activities and can be implemented with low computational costs.

Chapter 4 has described the proposed experimental procedures involved in obtaining DfD image pairs, i.e., the far-focused image and near-focused images. The image acquisition requires the lens-to-sensor distance to be adjusted with high precision (typically higher than 10 micrometres). However, the distance cannot be precisely measured and controlled with a standard camera system. In addition, we argue that the implementation in [155] is problematic since the object distance is changed during input images capture. To address this problem, we have proposed two optical systems. The SMS allows precise control of the distance by a micrometer attached to the base of the sensor holder, while the lens being decoupled from the sensor is held by a funnel holder. The DMS is a more advanced version where the funnel holder is replaced by another micrometer-controlled lens holder that allows more precise calibrations. In addition, in order to enable convenient DfD calibration and image capture, we have developed a MATLAB GUI software called *DfDtool*. The experimental procedures using *DfDtool*, SMS and DMS have also been presented in this chapter. SMS, DMS and *DfDtool* has been extensively used throughout our DfD experiments, where one pair of input images normally took 5-10 seconds to capture, and focusing the camera at a specific distance took 10-15 seconds. They are especially convenient when large amount of accuracy-testing images and correction patterns need to be acquired.

Chapter 5 has presented a new RO-based passive DfD technique. The pillbox PSF used in [16] and [17] is not valid when lens aberrations and diffraction are significant compared to the diameter of the blurred circle, two varieties of DfD approaches have thus been proposed to address this problem. The GRO using Gaussian PSF was designed for when lens aberrations and diffraction were dominant, and

the GGRO using generalised Gaussian PSF could adapt to any amount of aberrations and diffraction levels. An experimental procedure has also been developed to calculate the value of k, which was a ratio of the defocus parameter to the radius of a blurred circle, by a 1D search. Another produce was presented to calculate the value of p for GGRO by a 2D search, each loop of which was a 1D search of k. Apart from the monotonicity, the low-gradient components of the NIR also have a negative influence due to its high depth variation in the presence of noise. Thus, the pre-filter was redesigned to address this problem. We argued that the ROs design procedure in [16] was problematic and unnecessarily complicated. Hence we formed a new cost function according to the definition of the NIR, and this was able to produce the estimates of all the ROs simultaneously. A depth map distortion has been found which led to the depth map of a flat surface to be a curved surface. This problem refuted the assumption made by most DfD methods that the blurring effect was uniform across the image. To handle this problem, a DfD correction method based one two-step least squares fit was incorporated to GRO and GGRO. Experiments on real images have shown that GGRO achieved higher accuracy than GRO, which was in turn more accurate than Raj's method [17]. They have also shown that the correction algorithm made a considerable improvement.

Two different DfD correction methods have been presented in Chapter 6 to address the depth-variant elliptical distortion that often occurred due to optical distortion. They were designed to correct the depth estimation generated by any DfD algorithm using a number of correction patterns computed by the correction method. CDC searches for the optimal CPV to cancel out the distortion at every pixel location. It further improves the accuracy of reconstruction by considering the neighbourhood region and interpolation of CPI and CPV. CLSF finds the mapping from the distorted to the corrected reconstruction results directly. The accuracy is further improved by dividing the depth maps into a number of equal-sized regions sharing separate sets of coefficients. We suggest the parameter M in Eqn. (6.3) to be set within $[20, 30]$, and N in Eqn. (6.4) to be within [1/100,1/50] of the shorter one of the width and height of the depth map in terms of pixels. In order to deal with the large depth variation in low-texture region, both CDC and CLSF were incorporated with an interpolation algorithm that was based on variance analysis and hole-filling. With experiments using four different DfD methods on seven sets of real scenes, we have demonstrated their potentials of being adapted to all DfD methods. In addition, we have found that CDC generally produced better reconstructions than CLSF, where the reconstructed flat surfaces were flatter, at the cost of much lower speed. Experiments have also revealed that the sharp local noises at low-texture

regions were effectively mitigated.

Chapter 7 has proposed a method to efficiently generate training silhouettes from any view. Although there existed 3D silhouette generation techniques with multiple cameras, errors occur during the placement of cameras and high computational costs were involved. DfD is able to produce the surface reconstruction of one view of any object. Hence a complete 3D model can be built by stitching two views together. Hence DfD is potentially useful for this application where only two cameras are needed with one facing another and the subject in between. Since our current DfD system is not portable and cannot acquire input images in real-time, we have proposed a method to generate silhouettes from any view by using real-time data of a set of 3D coordinates of markers placed on the subject clothing while performing activities. The markers should be placed in a way that a volume hull of the body can be easily reconstructed. The coordinates are extracted by Vicon Nexus. This silhouette generation method first determines all possible triplets of point on every rigid body part. It then draws the corresponding 3D triangular surfaces. This is followed by projecting the 3D surfaces according to the view specified to create a 2D silhouette. The projection can also be obtained directly using the *trisurf()* function on MATLAB taking all possible triples as input. Finally, a simple and fast algorithm was proposed to remove very similar silhouettes. The silhouettes generated by this method has been used as the training data in Chapter 9. Note that we did not aim for accurate silhouette-to-silhouette matching. The training silhouettes were only used for low-level frame-by-frame manifold embedding. The high-level training data for activity recognition was obtained from sequences of embedded silhouettes, which had been extracted in advance with background subtraction that was also used at run-time for input silhouettes extraction.

Chapter 8 have presented a novel shadow removal algorithm based on known position of the sun. Most existing works were based on geometry, chromaticity, physical knowledge and texture [40], where only the large region texture-based method [40] was found to completely remove a shadow, but it required large computational effort. The proposed method for outdoor scenes avoids analysing such image information and determines the shadow using the known position of the sun. Using solar information rather than image information, it is thus able to avert the problem when the shadow has similar texture or colour saturation as the surrounding background. The algorithm contains three steps. First, shadow length and orientation are computed according to current time and location. Second, the shadow is projected onto the image plane. Third, the entire silhouette including shadow is rotated by an angle calculated with the shadow information, and the shadow part

of the silhouette is removed. Since there are significant variation on the subject height, head width and other input parameters, the variance of the angle of rotation is calculated with the error propagation formula. Hence instead of being rotated by an exact angle, the shadow is iteratively rotate within its variation range. Note that this algorithm only applies in outdoor sunny day or indoor environment with a single known light source, and it assumes the top part of the silhouette is the person's head. Experiments have demonstrated that the proposed method removed high portion of the shadow and low portion of the body compared to five existing methods. Moreover, no isolated shadow pixel region that was far away from the body was left. The shadow removal algorithm has been used to remove the shadow in KTH dataset during performance evaluation of the EPL algorithm in Chapter 9. Since input information including date and location of capture was not known, the projection of shadow angle was measured by hand.

Chapter 9 have presented a silhouette-based human activity recognition system. Various problems were encountered by existing activity recognition systems, such as temporal order-variant, speed-variant, cannot handle complex activities or inefficient. The proposed system operates on silhouettes extracted by a fast background subtraction algorithm available from openCV. It then transforms every silhouette into a feature vector using the proposed improved version of the centroid distance function that is robust against broken silhouettes. All training silhouettes obtained by the algorithm in Chapter 7 are embedded into the manifold space by Isomap. The mapping from feature vector space to manifold space is learnt efficiently by the proposed simplified version of RBF, which produces a set of parameters used during run-time silhouette embedding. During activity recognition, all silhouettes from an input video are embedded by RBF. The EPL algorithm then transforms the embedded pattern of the silhouettes into a spatial object, which is the motion template. The spatial object is invariant to the order of the sub-events (or actions) that comprises an activity, and the speed of execution of the activity. The order of the spatial object can be determined by cross-validation, but we found that third order was usually enough for 10 activities. Experiments have shown that the recognition system was able to produce good accuracy for our dataset contained 10 complex activities, most of which were non-periodic. In addition, using KTH and Weizmann dataset, the proposed method produced similar accuracies compared to a number of recent recognition methods. Furthermore, not using any parallel implementation, it has been implemented in near-real-time, i.e. 384.7 milliseconds per frame including shadow removal and 100.9 milliseconds excluding shadow removal.

10.2 Future work

The performance of the proposed DfD method is limited by two problems due to the small-sized ROs. First, the shape of the NIR cannot be reproduced without error. Second, the adverse frequency components cannot be removed completely. A possible solution is to investigate the use of a coded aperture to produce a PSF suitable for small ROs. In addition, the current optical system requires manual adjustment of sensor-to-lens distance with a precision of 5 micrometres. For real-time implementation, this adjustment mechanism can be replaced by a step motor. Another possible solution is to place a half mirror behind the lens which splits light into two halves received by two sensors. The real-time implementation can also be used to generate 3D silhouettes more efficiently than the proposed algorithm in Chapter 7, which has the problem of occasional occlusion of the markers during their coordinates recording.

For the DfD correction methods, one possible future work is to investigate the applicability of both CDC and CLSF on DfD approaches using a single image and active DfD. Another is to explore curve fitting techniques that are more sophisticated than least squares fit to further improve the correction accuracy. Furthermore, machine learning algorithms can also be exploited for distortion removal.

The major limitations of the proposed shadow removal algorithm includes: the significant variance resulting from an estimate of the subject height and the distance between the subject and the camera. These two problems can be addressed by using a range camera which is able to measure the distance and the height.

There are four major limitations of the EPL algorithm. First, sudden change of illumination and reflections often leads to silhouettes of very poor quality. Second, the essential parts of the subject such as head and feet are sometimes treated as noise and thus removed. Third, the background subtraction algorithm cannot deal with moving camera. Fourth, EPL currently only works with single subject performing an activity in one segmented video. Therefore, the future research includes: designing a new background subtraction algorithm that is robust against illumination changes and reflections (possibly with a human tracker), and is able to handle moving camera (by utilising other feature extraction techniques such as those in [118; 44]); investigating other dimensionality reduction techniques other than Isomap or exploit algorithms that provide optimal parameter for Isomap such as [191]; extending the algorithm to multiple subjects possibly with a human detector such as that used in [128]; transform EPL into an action or activity detector so that the algorithm is able to handle un-segmented video; use layered EPL to recognise more complex activities.

Appendix A

Terms Used Interchangeably

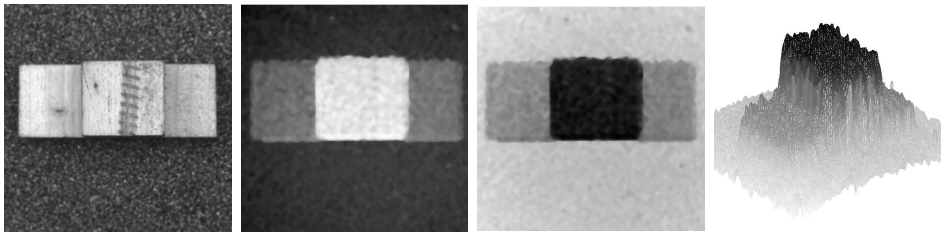
- Shape from shading/motion/stereo/depth/focus,
  depth from shading/motion/stereo/depth/focus
- Shape from stereo, depth from stereo, stereoscopic system
- Camera parameters, lens parameters, optical settings, focus settings
- Sharp discontinuity, edge
- Blur, defocus
- Depth, object distance, distance
- Image distance, sensor-to-lens distance
- Training data(silhouettes), gallery data(silhouettes), data library
- Local image region, patch, sub-image, window
- Normalised Image Ratio (NIR), M/P ratio
- Rational operator, rational filter
- Windowing effect, shift-invariant blurring, equifocal assumption
- Frequency response, Fourier transform, frequency-domain representation
- Least squares fit, linear regression, least squares regression, regression
- Subject, actor, person
- Furthest/nearest measurable distance, far/near-focused object distance

Appendix B

3D Representations

DfD uses images captured from a single view to obtain a depth map. In order to understand the final output of DfD, a number of 3D representation techniques including defocus map, depth map, mesh-plot, and 3D volumetric rendering along with their relationships and differences are discussed.

A digital image is a matrix of radiances of sample points in the camera view, such as shown in Fig. B.1(a), where brighter pixels represent high radiances and vice versa. A defocus map is a matrix of defocus amounts of sample points in the camera view, and a depth map is the corresponding matrix of depths (distances to the camera) calculated from the defocus values. An example grey-coded defocus map obtained from Fig. B.1(a) is shown in Fig. B.1(b), where brighter pixels represent larger defocus amount and vice versa.



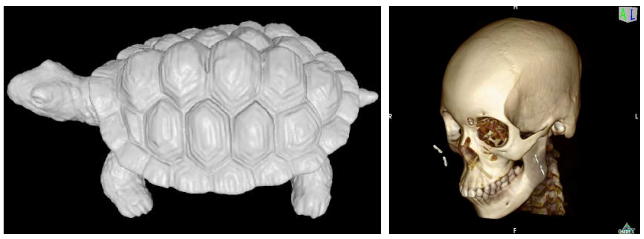
(a) (b) (c) (d)

Figure B.1: Depth map and mesh-plot: (a) A digital image of an object; (b) the defocus map; (c) the depth map; (d) the mesh-plot.

The grey-coded depth map obtained from the defocus map is shown in Fig. B.1(c), where brighter pixels represent larger depths and vice versa. Note that the furthest background is chosen to be focused such that the greater the blur the nearer the object is. The foreground can also be focused but in this case the greater blur means smaller depth.

The shape of the object can be perceived directly from the grey-coded mesh-plot shown in Fig. B.1(d). Since depth map is obtained from one view, the depth map and the mesh-plot cannot reveal any occluded part. Thus, the mesh-plot would be wrong if the depth change from the top of the wooden object to the background is not vertical.

Apart from depth map and mesh-plot, volumetric rendering is another popular 3D representation technique where depth information is available from any views as data is captured from more than one view. Thus, occlusion will not normally present. An example of 3D surface rendering is shown in Fig. B.2 (a), where the source of 3D perception is the shading of different parts of the object. An example of 3D volumetric rendering is shown in Fig. B.2 (b) obtained with X-ray computed tomography, where the interior structure of the skull is also reconstructed.



(a) (b)

Figure B.2: Examples of volumetric rendering. (a) Surface rendering of a turtle adapted from [192]. (b) Volumetric rendering of a skull from [193].

Notably, depth maps from multiple views can be stitched together to obtain a 3D surface rendered graph [1]. In this thesis, we discuss the use of DfD to obtain depth map from a single view only. Thus, graphical results such as those generated as shown in Fig. B.2 are not presented.

Appendix C

Generic Depth from Defocus

Fig. C.1 illustrates the principle of DfD with a point object. The object is u away from the lens, the lens has an aperture of diameter d, and the distance between the lens and the sensor is s. When the focus point is not on the sensor, the image of the object point is a blurred circle on the sensor with diameter a. The Gaussian PSF is used, and its standard deviation (SD) is determined by camera parameters, including focal length F, aperture d and sensor-to-lens distance s, as illustrated in Fig. C.1, and is given by

$$\sigma = kd \times (\frac{1}{F} - \frac{1}{u} - \frac{1}{s}) \ , \tag{C.1}$$

where k is a camera constant obtained by calibration and is equal to the blur circle diameter a divided by σ, and u is the depth to be estimated. Taking the Fourier transformation of both side of Eqn. (1.1) gives

$$\check{\mathbf{I}} = \check{\mathbf{H}} \times \check{\mathbf{M}} \ , \tag{C.2}$$

where $\check{\mathbf{I}}$, $\check{\mathbf{H}}$ and $\check{\mathbf{M}}$ are the Fourier transform of $\mathbf{I}$, $\mathbf{H}$ and $\mathbf{M}$, respectively. $\check{\mathbf{H}}$ is also called the optical transfer function (OTF).

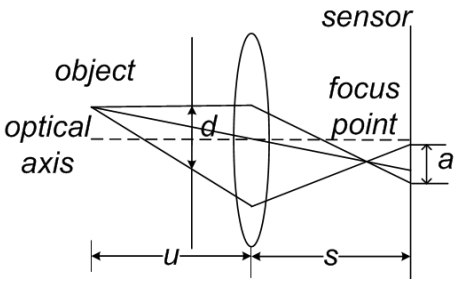


Figure C.1: A simple DfD optical system.

Two images are captured with different parameters, i.e.

$$\check{\mathbf{I}}_1 = \check{\mathbf{H}}_1 \times \check{\mathbf{M}} \ , \tag{C.3}$$

$$\check{\mathbf{I}}_2 = \check{\mathbf{H}}_1 \times \check{\mathbf{M}} \ . \tag{C.4}$$

Dividing Eqn. (C.3) by Eqn. (C.4) gives

$$\frac{\check{\mathbf{I}}_1}{\check{\mathbf{I}}_2} = \frac{\check{\mathbf{H}}_1}{\check{\mathbf{H}}_2} \ . \tag{C.5}$$

If only the sensor separation s is different for the two images, and F and d are known, Eqn. (1.2), (C.1) and (C.5) can be used to compute the depth u. Similarly, the aperture d can also be different for the two images.

To produce a dense depth map with high resolution, each of the input images is divided equally into small local regions with a size of at least 2×2. The depth value within one region is assumed to be the same and is then computed. This process is repeated for all regions to obtain the depth map. In this thesis, we use the term "local image region", "patch", "window" and "neighbourhood" interchangeably.

Appendix D

Telecentric Optics

The size of the image is changed during DfD image capture for near/far-focused images if telecentric optics is not used [194] [166]. As discussed in Section 1.2.2 there should be no correspondence problem for DfD, or any pixel at a location of the first image must correspond to the pixel at the same location of the second image. However, this size-changing problem results in both input images to be scaled differently. Hence the pixel-to-pixel correspondence becomes invalid which leads to inaccurate depth map.

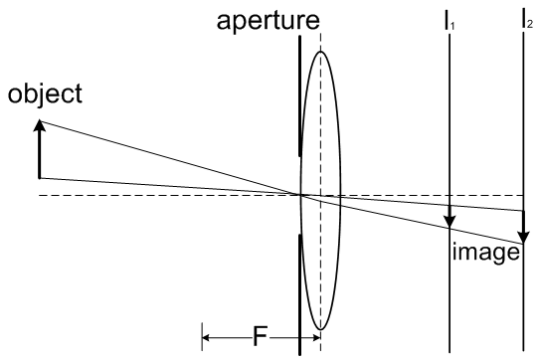


Figure D.1: The problem of image magnification when aperture-to-lens distance is smaller than the focal length.

Fig. D.1 illustrates the problem when the aperture to lens distance is smaller than the focal length F. In order to understand the change in size, the light rays from two points on the object are used for illustration. Both of them are principal rays which pass through the centre of the aperture. Other rays emitted from these points can be neglected because they are only related to the degree of blur rather than size. Because the aperture-to-lens distance is not equal to focal length, the principal rays will not be parallel to the optical axis when they have passed through

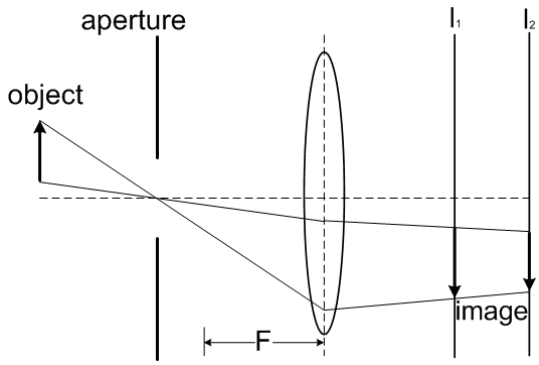
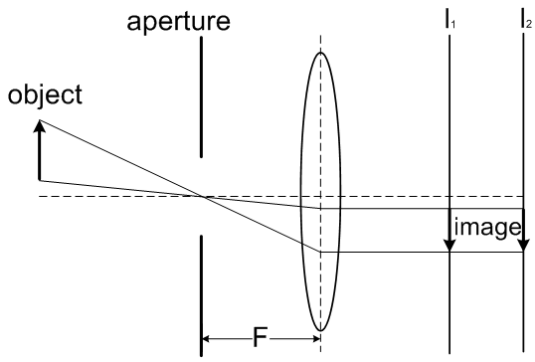


Figure D.2: The problem of image magnification when aperture-to-lens distance is larger than the focal length.



(a)

Figure D.3: The telecentric system.

the lens. As a result, images taken at l_1 and l_2 are of different size.

Fig. D.2 illustrates the problem when the aperture to lens distance is larger than the focal length F. Again, images taken at I_1 and I_2 are of different sizes. Fig. D.3 illustrates the telecentric optics to address the problem. The aperture-to-lens distance is equal to focal length, thus the principal rays become parallel to each other when they pass through the lens. As a result, images taken at I_1 and I_2 are of the same size.

Appendix E

Generalised Gaussian PSF

There are a number of other point spread functions (PSFs) other than the Gaussian PSF, such as the pill-box PSF, the generalised Gaussian PSF, Fermi-Dirac PSF, etc. For a pill-box PSF, its value is 1 within its radius parameter and 0 otherwise.

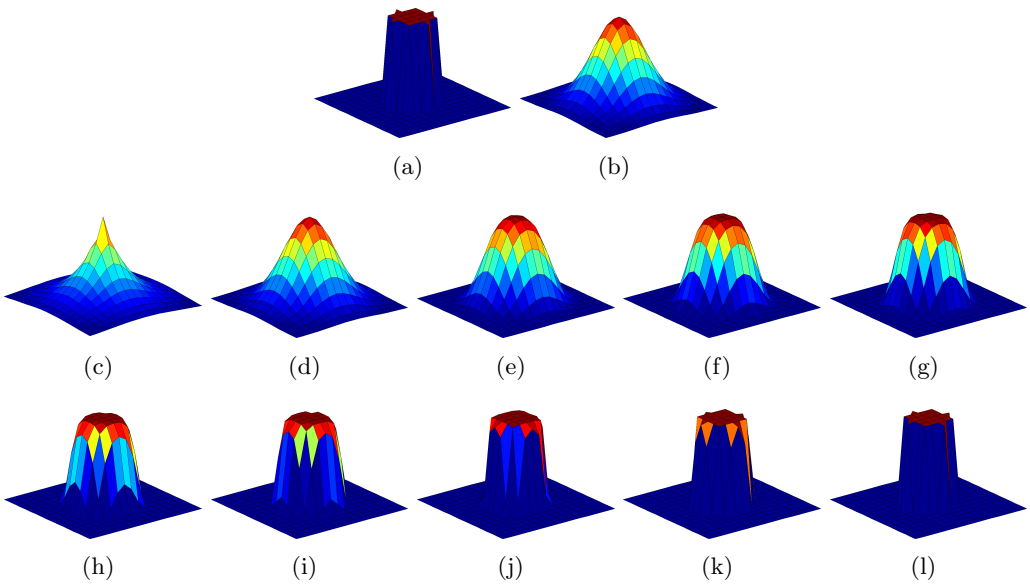


Figure E.1: Gaussian PSF: (a) The pill box PSF with radius equal to 3. (b) The Gaussian PSF when $\sigma = 3$. (c-l): The generalised Gaussian PSF when $p = 1, 2, 3, 4, 5, 7, 10, 20, 50$ and 120 and $\sigma = 3$.

Fig. E.1(a) shows an example of the pill-box PSF. Pill-box PSF is a good model for blur when the blur is uniform. However, due to aberrations, diffraction and other effects the Gaussian PSF is a better model of blur [18]. The generalised Gaussian proposed in [168], which is a combination between those two can adapt situations with any amount of aberrations and diffraction. The generalised Gaussian

PSF is

$$\vec{h}(x) = \frac{p^{1-\frac{1}{p}}}{2\sigma\Gamma\left(\frac{1}{p}\right)} \exp\left[-\frac{1}{p}\frac{|x-\overline{x}|^p}{\sigma^p}\right], \tag{E.1}$$

where $\Gamma()$ is the Gamma function, σ is the SD, x is the spatial index, p is a parameter such that when it is 2 the PSF is a Gaussian and when it is positive infinity the PSF is pill-box. The 2-dimensional version is

$$\mathbf{H}(x,y) = \frac{p^{2-\frac{2}{p}}}{4\sigma^2\Gamma^2\left(\frac{1}{p}\right)} \exp\left[-\frac{1}{p}\frac{|(x-\overline{x})^2+(y-\overline{y})^2|^{\frac{p}{2}}}{\sigma^p}\right]. \tag{E.2}$$

Fig. E.1 illustrates the generalised Gaussian PSF with comparison to the pill-box PSF and the Gaussian PSF. All plots are of the same size of 17×17 and are generated with the same scale. Fig. E.1(a) shows the pill-box PSF with radius equal to 3. Fig. E.1(b) shows the Gaussian PSF with $\sigma = 3$ equal to 3. Fig. E.1(c-l) shows the generalised Gaussian PSF when $\sigma = 3$ and $p = 1$ to $p = 120$. The figure shows that the generalised Gaussian PSF is identical to the Gaussian PSF when $p = 2$. As the value of p increases, the PSF becomes more and more similar to a pill-box PSF. At $p = 120$, the PSF is visually identical to the pill-box one.

Bibliography

[1] Brian Curless. From range scans to 3D models. *ACM SIGGRAPH Computer Graphics*, 33(4):38–41, 1999.

[2] Thomas Guth and Bernd Czepan. Coordinate measurement device and method for controlling same. U.S. Patent: 6587810, issued date 2003.

[3] Carlo G. Someda. *Electromagnetic Waves*. CRC Press, 2006.

[4] LiDAR basics. In Ohio Department of Transportation. Retrieved 11:29, 9/6/2014, from `http://www.dot.state.oh.us/Divisions/Engineering/CaddMapping/RemoteSensingandMapping/Pages/LiDAR-Basics.aspx`.

[5] Spring 2011 Rhode island statewide LiDAR data. In Rhode Island Geographic Information System. Retrieved 11:19, 9/6/2014, from `http://www.edc.uri.edu/rigis/data/download/lidar/2011USGS`.

[6] François Blais. Review of 20 years of range sensor development. *Journal of Electronic Imaging*, 13(1):231–243, 2004.

[7] Song Zhang. Recent progresses on real-time 3D shape measurement using digital fringe projection techniques. *Optics and Lasers in Engineering*, 48(2):149–158, 2010.

[8] Zhengyou Zhang. Microsoft kinect sensor and its effect. *IEEE MultiMedia*, 19(2):4–10, 2012.

[9] David Marr and Tomaso Poggio. Cooperative computation of stereo disparity. *Science*, 194(4262):283–287, 1976.

[10] 3D photography and geometry processing. In Taubin Group. Retrieved 14:57, 9/8/2014, from `http://mesh.brown.edu/3dpgp-2009/homework/hw2/hw2.html`.

[11] Alex Paul Pentland. Local shading analysis. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, PAMI-6(2):170–187, March 1984.

[12] Basic shading. Retrieved 12:22, 9/6/2014, from `http://hippie.nu/~unicorn/tut/xhtml-chunked/ch05.html`.

[13] Transform your smartphone into a mobile 3D scanner. In ETHzurich Department of Computer Science. Retrieved 11:15, 9/6/2014, from `http://www.inf.ethz.ch/news-and-events/spotlights/mobile_3dscanner.html`.

[14] G. Calin and V. O. Roda. Real-time disparity map extraction in a dual head stereo vision system. *Latin American Applied Research*, 37:1, 2007.

[15] Zscanner 700PX. In 3DSystems. Retrieved 11:39, 9/6/2014, from `http://www.zcorp.com/documents/380_ZScanner700_PX_SpecSheet_HiRes.pdf`.

[16] Masahiro Watanabe and Shree K Nayar. Rational filters for passive depth from defocus. *International Journal of Computer Vision*, 27(3):203–225, 1998.

[17] Alex N. J. Raj and Richard C. Staunton. Rational filter design for depth from defocus. *Pattern Recognition*, 45:198–207, 2011.

[18] Murali Subbarao. Parallel depth recovery by changing camera parameters. In *Proceedings of IEEE Conference on Computer Vision and Pattern Recognition*, 1988.

[19] Frangipani trees. In How To Garden. Retrieved 14:15, 25/7/2014, from `http://howto-garden.com.au/selecting-plants/frangipani-trees/`.

[20] Neuschwanstein castle, Germany. In 8ThingsToDo. Retrieved 07:19, 7/6/2014, from `http://www.8thingstodo.com/neuschwanstein-castle-germany`.

[21] What is dfd (depth from defocus) technology? In Panasonic. Retrieved 15:32, 28/7/2014, from `http://eng.faq.panasonic.com/app/answers/detail/a_id/26478/~/what-is-dfd-(depth-from-defocus)-technology%3F---dmc-gh4`.

[22] J. K. Aggarwal and Michael S. Ryoo. Human activity analysis: A review. *ACM Computing Surveys (CSUR)*, 43(3):16, 2011.

[23] I. Laptev and T. Lindeberg. Space-time interest points. In *Proceedings of International Conference on Computer Vision*, pages 432–439, 2003.

[24] Shuiwang Ji, Wei Xu, Ming Yang, and Kai Yu. 3D convolutional neural networks for human action recognition. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 35(1):221–231, 2013.

[25] Junji Yamato, Jun Ohya, and Kenichiro Ishii. Recognizing human action in time-sequential images using hidden Markov model. In *Proceedings of IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, pages 379–385. IEEE, 1992.

[26] Trevor Darrell and Alex Pentland. Space-time gestures. In *Proceedings of IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, pages 335–340. IEEE, 1993.

[27] Moshe Blank, Lena Gorelick, Eli Shechtman, Michal Irani, and Ronen Basri. Actions as space-time shapes. In *Proceedings of International Conference on Computer Vision*, volume 2, pages 1395–1402. IEEE, 2005.

[28] Osama Masoud and Nikos Papanikolopoulos. A method for human action recognition. *Image and Vision Computing*, 21(8):729–743, 2003.

[29] Joseph B. Kruskal. Multidimensional scaling by optimizing goodness of fit to a nonmetric hypothesis. *Psychometrika*, 29(1):1–27, 1964.

[30] Sam T. Roweis and Lawrence K. Saul. Nonlinear dimensionality reduction by locally linear embedding. *Science*, 290(5500):2323–2326, 2000.

[31] Joshua B. Tenenbaum, Vin De Silva, and John C. Langford. A global geometric framework for nonlinear dimensionality reduction. *Science*, 290(5500):2319–2323, 2000.

[32] Yaser Yacoob and Michael J Black. Parameterized modeling and recognition of activities. In *Proceedings of IEEE International Conference on Computer Vision*, pages 120–127. IEEE, 1998.

[33] Z. A. Khan and W. Sohn. Hierarchical human activity recognition system based on R-transform and nonlinear kernel discriminant features. *Electronics Letters*, 48(18):1119–1120, 2012.

[34] Ahmed Elgammal and Chan-Su Lee. Inferring 3D body pose from silhouettes using activity manifold learning. In *Proceedings of IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, volume 2, pages II–681. IEEE, 2004.

[35] Xiaoyu Deng, Xiao Liu, Mingli Song, Jun Cheng, Jiajun Bu, and Chun Chen. LF-EME: Local features with elastic manifold embedding for human action recognition. *Neurocomputing*, 99:144–153, 2013.

[36] Pavan Turaga, Rama Chellappa, Venkatramana S. Subrahmanian, and Octavian Udrea. Machine recognition of human activities: A survey. *IEEE Transactions on Circuits and Systems for Video Technology*, 18(11):1473–1488, 2008.

[37] Shih-Fu Chang. The holy grail of content-based media analysis. *IEEE MultiMedia*, 9(2):6–10, 2002.

[38] Huiyu Zhou and Huosheng Hu. Human motion tracking for rehabilitation – a survey. *Biomedical Signal Processing and Control*, 3(1):1–18, 2008.

[39] Bonita. In 3D Vicon. Retrieved 10:12, 9/5/2014, from `http://www.vicon.com/System/Bonita`.

[40] Andres Sanin, Conrad Sanderson, and Brian C. Lovell. Shadow detection: A survey and comparative evaluation of recent methods. *Pattern Recognition*, 45(4):1684–1695, 2012.

[41] Yan Ke, R. Sukthankar, and M. Hebert. Spatio-temporal shape and flow correlation for action recognition. In *Proceedings of IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, pages 1–8, June 2007.

[42] M. D. Rodriguez, J. Ahmed, and M. Shah. Action MACH a spatio-temporal maximum average correlation height filter for action recognition. In *Proceedings of IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, pages 1–8, June 2008.

[43] Heng Wang, Alexander Klaser, Cordelia Schmid, and Cheng-Lin Liu. Action recognition by dense trajectories. In *Proceedings of IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, pages 3169–3176. IEEE, 2011.

[44] Heng Wang, Alexander Kläser, Cordelia Schmid, and Cheng-Lin Liu. Dense trajectories and motion boundary descriptors for action recognition. *International Journal of Computer Vision*, 103(1):60–79, 2013.

[45] Ang Li, Richard C. Staunton, and Tardi Tjahjadi. Rational-operator-based depth-from-defocus approach to scene reconstruction. *Journal of the Optical Society of America A*, 30:1787–1795, 2013.

[46] Ang Li, Richard C Staunton, and Tardi Tjahjadi. Adaptive deformation correction of depth from defocus for object reconstruction. *Journal of the Optical Society of America A*, 31(12):2694–2702, 2014.

[47] Ang Li and Tardi Tjahjadi. Video-based human activity recognition using embedded silhouettes. Submitted to *Pattern Recognition*, 2014.

[48] Alex Paul Pentland. Depth of scene from depth of field. In *Proceedings of Image Understanding Workshop*, 1982.

[49] Alex Paul Pentland. A new sense for depth of field. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 9(4):523–531, 1987.

[50] Muralidhara Subbarao and Natarajan Gurumoorthy. Depth recovery from blurred edges. In *Proceedings of IEEE Conference on Computer Vision and Pattern Recognition*, pages 498–503, Jun 1988.

[51] Shang-Hong Lai, Chang-Wu Fu, and Shyang Chang. A generalized depth estimation algorithm with a single image. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 14(4):405–411, 1992.

[52] Du-Ming Tsai and Chin-Tun Lin. A moment-preserving approach for depth from defocus. *Pattern Recognition*, 31(5):551 – 560, 1998.

[53] A. Nedzved, V. Bucha, and S. Ablameyko. Augmented 3D endoscopy video. In *Proceedings of 3DTV Conference on The True Vision-Capture, Transmission and Display of 3D Video*, pages 349–352. IEEE, 2008.

[54] Anat Levin, Rob Fergus, Frédo Durand, and William T. Freeman. Image and depth from a conventional camera with a coded aperture. *ACM Transactions on Graphics*, 26(3):70, 2007.

[55] A. Saxena, S. Chung, and A. Ng. 3-d depth reconstruction from a single still image. *International Journal of Computer Vision*, 76(1):53–69, 2008.

[56] Stan Z. Li. *Markov Random Field Modeling in Computer Vision*. Springer-Verlag New York, Incorporation, 1995.

[57] Shaojie Zhuo and Terence Sim. Defocus map estimation from a single image. *Pattern Recognition*, 44(9):1852 – 1858, 2011. Computer Analysis of Images and Patterns.

[58] Alex Paul Pentland, T. Darrell, M. Turk, and W. Huang. A simple, real-time range camera. In *Proceedings of IEEE Conference on Computer Vision and Pattern Recognition*, pages 256–261, Jun 1989.

[59] Murali Subbarao and T-C Wei. Depth from defocus and rapid autofocusing: a practical approach. In *Proceedings of IEEE Conference on Computer Vision and Pattern Recognition*, pages 773–776. IEEE, 1992.

[60] V. Michael Bove Jr. Entropy-based depth from focus. *Journal of the Optical Society of America A*, 10(4):561–566, 1993.

[61] Ambasamudram N. Rajagopalan and Subhasis Chaudhuri. A block shift-variant blur model for recovering depth from defocused images. In *Proceedings of International Conference on Image Processing*, volume 3, pages 636–639. IEEE, 1995.

[62] Ambasamudram N. Rajagopalan and Subhasis Chaudhuri. Space-variant approaches to recovery of depth from defocused images. *Computer Vision and Image Understanding*, 68(3):309–329, 1997.

[63] Franz Hlawatsch and G. Faye Boudreaux-Bartels. Linear and quadratic time-frequency signal representations. *IEEE Signal Processing Magazine*, 9(2):21–67, 1992.

[64] Yalin Xiong and Steven A. Shafer. Moment and hypergeometric filters for high precision computation of focus, stereo and optical flow. *International Journal of Computer Vision*, 22(1):25–59, 1997.

[65] Trevor Darrell and Kwangyoen Wohn. Pyramid based depth from focus. In *Proceedings of IEEE Conference on Computer Vision and Pattern Recognition*, pages 504–509. IEEE, 1988.

[66] Trevor Darrell and Kwangyoen Wohn. Depth from focus using a pyramid architecture. *Pattern Recognition Letters*, 11(12):787–796, 1990.

[67] Mark A. Pinsky. *Partial Differential Equations and Boundary-value Problems with Applications*. American Mathematical Society, 2011.

[68] John Ens and Peter Lawrence. A matrix based method for determining depth from focus. In *Proceedings of IEEE Conference on Computer Vision and Pattern Recognition*, pages 600–606, Jun 1991.

[69] John Ens and Peter Lawrence. An investigation of methods for determining depth from focus. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 15(12):97–108, 1993.

[70] Murali Subbarao. Spatial-domain convolution/deconvolution transform. Technical report, State University of New York at Stony Brook, NY, 1991.

[71] Gopal Surya and Murali Subbarao. Depth from defocus by changing camera aperture: a spatial domain approach. In *Proceedings of IEEE Conference on Computer Vision and Pattern Recognition*, pages 61–67, Jun 1993.

[72] Murali Subbarao and Gopal Surya. Depth from defocus: a spatial domain approach. *International Journal of Computer Vision*, 13(3):271–294, 1994.

[73] Yalin Xiong and Steven A. Shafer. Depth from focusing and defocusing. In *Proceedings of IEEE Conference on Computer Vision and Pattern Recognition*, pages 68–73. IEEE, 1993.

[74] Ambasamudram N. Rajagopalan and Subhasis Chaudhuri. Optimal recovery of depth from defocused images using an MRF model. In *Proceedings of IEEE International Conference on Computer Vision*, pages 1047–1052. IEEE, 1998.

[75] Ambasamudram N. Rajagopalan and Subhasis Chaudhuri. An MRF model-based approach to simultaneous recovery of depth and restoration from defocused images. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 21(7):577–589, 1999.

[76] Paolo Favaro and Stefano Soatto. Shape and radiance estimation from the information divergence of blurred images. In *Computer Vision-ECCV 2000*, pages 755–768. Springer, 2000.

[77] Hailin Jin and Paolo Favaro. A variational approach to shape from defocus. In *Proceedings of European Conference on Computer Vision*, pages 18–30. Springer, 2002.

[78] Hailin Jin, Anthony J Yezzi, Yen-Hsi Tsai, Li-Tien Cheng, and Stefano Soatto. Estimation of 3D surface shape and smooth radiance from 2d images: A level set approach. *Journal of Scientific Computing*, 19(1-3):267–292, 2003.

[79] Olivier Faugeras and Renaud Keriven. *Variational Principles, Surface Evolution, PDE's, Level Set Methods and the Stereo Problem.* IEEE, 2002.

[80] Paolo Favaro, Stanley Osher, Stefano Soatto, and L Vese. 3D shape from anisotropic diffusion. In *Proceedings of IEEE Conference on Computer Vision and Pattern Recognition*, pages 179–186, 2003.

[81] Paolo Favaro, Stefano Soatto, Martin Burger, and Stanley J. Osher. Shape from defocus via diffusion. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 30(3):518–531, 2008.

[82] Jan J. Koenderink. The structure of images. *Biological Cybernetics*, 50(5):363–370, 1984.

[83] Arnav V. Bhavsar and Ambasamudram N. Rajagopalan. Depth estimation with a practical camera. In *BMVC*, pages 1–11, 2009.

[84] Stefano Soatto and Paolo Favaro. A geometric approach to blind deconvolution with application to shape from defocus. In *Proceedings of IEEE Conference on Computer Vision and Pattern Recognition*, volume 2, pages 10–17. IEEE, 2000.

[85] Paolo Favaro and Stefano Soatto. Learning shape from defocus. In *Proceedings of European Conference on Computer Vision*, pages 735–745, 2002.

[86] Paolo Favaro and Stefano Soatto. A geometric approach to shape from defocus. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 27(3):406–417, 2005.

[87] Jin Ho Kwak and Sungpyo Hong. *Linear Algebra.* Springer Science & Business Media, 2004.

[88] Li Ma and Rihcard C. Staunton. Integration of multiresolution image segmentation and neural networks for object depth recovery. *Pattern Recognition*, 38:985–996, 2005.

[89] Deok J. Park, Kwon M. Nam, and Rae-Hong Park. Multiresolution edge detection techniques. *Pattern Recognition*, 28(2):211–229, 1995.

[90] Philippe Schroeter and Josef Bigün. Hierarchical image segmentation by multidimensional clustering and orientation-adaptive boundary refinement. *Pattern Recognition*, 28(5):695–709, 1995.

[91] D Ziou. Passive depth from defocus using a spatial domain approach. In *Proceedings of International Conference on Computer Vision*, pages 799–804. IEEE, 1998.

[92] Djemel Ziou and Francois Deschênes. Depth from defocus estimation in spatial domain. *Computer Vision and Image Understanding*, 81(2):143–165, 2001.

[93] Muhammad Asif, Aamir Saeed Malik, and Tae-Sun Choi. 3D shape recovery from image defocus using wavelet analysis. In *Proceedings of IEEE International Conference on Image Processing*, volume 1, pages I–1025. IEEE, 2005.

[94] Jinlong Lin, Chao Zhang, and Qingyun Shi. Estimating the amount of defocus through a wavelet transform approach. *Pattern Recognition Letters*, 25(4):407–411, 2004.

[95] Changyin Zhou, Stephen Lin, and Shree Nayar. Coded aperture pairs for depth from defocus. In *Proceedings of IEEE International Conference on Computer Vision*, pages 325–332. IEEE, 2009.

[96] Changyin Zhou, Stephen Lin, and Shree K. Nayar. Coded aperture pairs for depth from defocus and defocus deblurring. *International Journal of Computer Vision*, 93(1):53–72, 2011.

[97] Chandramouli Paramanand and Ambasamudram N. Rajagopalan. Depth from motion and optical blur with an unscented Kalman filter. *IEEE Transactions on Image Processing*, 21(5):2798–2811, 2012.

[98] Richard J. Meinhold and Nozer D. Singpurwalla. Understanding the Kalman filter. *The American Statistician*, 37(2):123–127, 1983.

[99] Simon J Julier and Jeffrey K. Uhlmann. A new extension of the Kalman filter to nonlinear systems. In *International Society for Optics and Photonics,Signal Processing, Sensor Fusion, and Target Recognition VI*, volume 3068. Orlando, FL, 1997.

[100] Chandramouli Paramanand and Ambasamudram N. Rajagopalan. Unscented transformation for depth from motion-blur in videos. In *Proceedings of IEEE Conference on Computer Vision and Pattern Recognition Workshops*, pages 38–44. IEEE, 2010.

[101] Aaron F. Bobick and James W. Davis. The recognition of human movement using temporal templates. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 23(3):257–267, 2001.

[102] Ming-Kuei Hu. Visual pattern recognition by moment invariants. *IRE Transactions on Information Theory*, 8(2):179–187, 1962.

[103] Mahalanobis distance description. In Jenness Enterprises. Retrieved 15:15, 20/7/2014, from `http://www.jennessent.com/arcview/mahalanobis_description.htm`.

[104] Lena Gorelick, Moshe Blank, Eli Shechtman, Michal Irani, and Ronen Basri. Actions as space-time shapes. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 29(12):2247–2253, 2007.

[105] Dorin Comaniciu, Visvanathan Ramesh, and Peter Meer. Real-time tracking of non-rigid objects using mean shift. In *Proceedings of IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, volume 2, pages 142–149. IEEE, 2000.

[106] Eli Shechtman and Michal Irani. Space-time behavior based correlation. In *Proceedings of IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, volume 1, pages 405–412. IEEE, 2005.

[107] S. Duffner. *Face Image Analysis with Convolutional Neural Networks*. GRIN Verlag, 2009.

[108] Ivan Laptev. On space-time interest points. *International Journal of Computer Vision*, 64(2-3):107–123, 2005.

[109] Chris Harris and Mike Stephens. A combined corner and edge detector. In *Proceedings of Alvey vision conference*, volume 15, page 50. Manchester, UK, 1988.

[110] Wolfgang Förstner and Eberhard Gülch. A fast operator for detection and precise location of distinct points, corners and centres of circular features. In *Proceedings of ISPRS intercommission conference on fast processing of photogrammetric data*, pages 281–305, 1987.

[111] Piotr Dollár, Vincent Rabaud, Garrison Cottrell, and Serge Belongie. Behavior recognition via sparse spatio-temporal features. In *Joint IEEE International Workshop on Visual Surveillance and Performance Evaluation of Tracking and Surveillance*, pages 65–72. IEEE, 2005.

[112] Christian Schuldt, Ivan Laptev, and Barbara Caputo. Recognizing human actions: a local SVM approach. In *Proceedings of the 17th International Conference on Pattern Recognition*, volume 3, pages 32–36. IEEE, 2004.

[113] Christian Wallraven, Barbara Caputo, and Arnulf Graf. Recognition with local features: the kernel recipe. In *Proceedings of IEEE International Conference on Computer Vision*, pages 257–264. IEEE, 2003.

[114] Juan Carlos Niebles, Hongcheng Wang, and Li Fei-Fei. Unsupervised learning of human action categories using spatial-temporal words. *International journal of computer vision*, 79(3):299–318, 2008.

[115] Thomas Hofmann. Probabilistic latent semantic indexing. In *Proceedings of international ACM SIGIR conference on Research and development in information retrieval*, pages 50–57. ACM, 1999.

[116] David M Blei, Andrew Y Ng, and Michael I Jordan. Latent dirichlet allocation. *Journal of machine Learning research*, 3:993–1022, 2003.

[117] Andrew Burton and John Radford. *Thinking in Perspective: Critical Essays in the Study of Thought Processes*. Methuen, 1978.

[118] Alexei A Efros, Alexander C Berg, Greg Mori, and Jitendra Malik. Recognizing action at a distance. In *Computer Vision, 2003. Proceedings. Ninth IEEE International Conference on*, pages 726–733. IEEE, 2003.

[119] Bruce D Lucas, Takeo Kanade, et al. An iterative image registration technique with an application to stereo vision. In *DARPA Image Understanding Workshop*, volume 81, pages 674–679, 1981.

[120] Saad Ali and Mubarak Shah. Human action recognition in videos using kinematic features and multiple instance learning. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 32(2):288–303, 2010.

[121] Ian Jolliffe. *Principal component analysis*. Wiley Online Library, 2005.

[122] Leonard E. Baum and Ted Petrie. Statistical inference for probabilistic functions of finite state Markov chains. *The Annals of Mathematical Statistics*, 37(5):1554–1563, 1966.

[123] Leonard E. Baum and J. A. Eagon. An inequality with applications to statistical estimation for probabilistic functions of Markov processes and to a model for ecology. *Bulletin of the American Mathematical Society*, 73(3):360–363, 1967.

[124] Leonard E. Baum and George Sell. Growth transformations for functions on manifolds. *Pacific Journal of Mathematics*, 27(2):211–227, 1968.

[125] Leonard E. Baum, Ted Petrie, George Soules, and Norman Weiss. A maximization technique occurring in the statistical analysis of probabilistic functions of Markov chains. *The annals of mathematical statistics*, 41(1):164–171, 1970.

[126] Lawrence Rabiner and Biing-Hwang Juang. An introduction to hidden Markov models. *ASSP Magazine, IEEE*, 3(1):4–16, 1986.

[127] Xuedong D. Huang, Yasuo Ariki, and Mervyn A. Jack. *Hidden Markov models for speech recognition*, volume 2004. Edinburgh university press Edinburgh, 1990.

[128] Weilun Lao, Jungong Han, and Peter H. N. de With. Automatic video-based human motion analyzer for consumer surveillance system. *IEEE Transactions on Consumer Electronics*, 55(2):591–598, 2009.

[129] Zoran Zivkovic and Ferdinand van der Heijden. Efficient adaptive density estimation per image pixel for the task of background subtraction. *Pattern recognition letters*, 27(7):773–780, 2006.

[130] James F. Allen and George Ferguson. Actions and events in interval temporal logic. *Journal of logic and computation*, 4(5):531–579, 1994.

[131] Zafar Ali Khan and Won Sohn. Feature extraction and dimensions reduction using R transform and principal component analysis for abnormal human activity recognition. In *Proceedings of International Conference on Advanced Information Management and Service*, pages 253–258. IEEE, 2010.

[132] Ying Wang, Kaiqi Huang, and Tieniu Tan. Human activity recognition based on r transform. In *Proceedings of IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, pages 1–8. IEEE, 2007.

[133] Gregory Beylkin. Discrete radon transform. *IEEE Transactions on Acoustics, Speech and Signal Processing*, 35(2):162–172, 1987.

[134] Bernhard Scholkopft and Klaus-Robert Mullert. Fisher discriminant analysis with kernels. *Neural networks for signal processing IX*, 1:1, 1999.

[135] Nuria Oliver, Eric Horvitz, and Ashutosh Garg. Layered representations for human activity recognition. In *Proceedings of IEEE International Conference on Multimodal Interfaces*, pages 3–8. IEEE, 2002.

[136] Nam Thanh Nguyen, Dinh Q Phung, Svetha Venkatesh, and Hung Bui. Learning and detecting activities from movement trajectories using the hierarchical

hidden Markov model. In *Proceedings of IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, volume 2, pages 955–960, 2005.

[137] Arnaud Doucet, Nando De Freitas, Kevin Murphy, and Stuart Russell. Rao-Blackwellised particle filtering for dynamic Bayesian networks. In *Proceedings of the Sixteenth conference on Uncertainty in artificial intelligence*, pages 176–183. Morgan Kaufmann Publishers Inc., 2000.

[138] Dong Zhang, Daniel Gatica-Perez, Samy Bengio, and Iain McCowan. Modeling individual and group actions in meetings with layered HMMs. *IEEE Transactions on Multimedia*, 8(3):509–520, 2006.

[139] Cristian Sminchisescu, Atul Kanaujia, and Dimitris Metaxas. Conditional models for contextual human motion recognition. *Computer Vision and Image Understanding*, 104(2):210–220, 2006.

[140] Liang Wang and David Suter. Recognizing human activities from silhouettes: Motion subspace and factorial discriminative graphical model. In *Proceedings of IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, pages 1–8. IEEE, 2007.

[141] Pradeep Natarajan and Ram Nevatia. View and scale invariant action recognition using multiview shape-flow models. In *Proceedings of IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, pages 1–8. IEEE, 2008.

[142] Jianguo Zhang and Shaogang Gong. Action categorization with modified hidden conditional random field. *Pattern Recognition*, 43(1):197–203, 2010.

[143] Art Lew and Holger Mauch. *Dynamic Programming: A Computational Tool*. Springer Science & Business Media, 2006.

[144] Hiroaki Sakoe and Seibi Chiba. Dynamic programming algorithm optimization for spoken word recognition. *IEEE Transactions on Acoustics, Speech and Signal Processing*, 26(1):43–49, 1978.

[145] Jaron Blackburn and Eraldo Ribeiro. Human motion recognition using Isomap and dynamic time warping. In *Human motion–understanding, modeling, capture and animation*, pages 285–298. Springer, 2007.

[146] Lee W Campbell and Aaron F Bobick. Recognition of human body motion using phase space constraints. In *Proceedings of IEEE International Conference on Computer Vision*, pages 624–630, 1995.

[147] Shanon X. Ju, Michael J. Black, and Yaser Yacoob. Cardboard people: A parameterized model of articulated image motion. In *Proceedings of the Second International Conference on Automatic Face and Gesture Recognition*, pages 38–44. IEEE, 1996.

[148] Tomaso Poggio and Federico Girosi. Networks for approximation and learning. *Proceedings of the IEEE*, 78(9):1481–1497, 1990.

[149] Y. Sheikh, M. Sheikh, and M. Shah. Exploring the space of a human action. In *Proceedings of IEEE International Conference on Computer Vision*, volume 1, pages 144–149 Vol. 1, Oct 2005.

[150] A. Yilmaz and M. Shah. Actions sketch: a novel action representation. In *Proceedings of IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, volume 1, pages 984–989 vol. 1, June 2005.

[151] Binlong Li, Mustafa Ayazoglu, Teresa Mao, Octavia I. Camps, and Mario Sznaier. Activity recognition using dynamic subspace angles. In *Proceedings of IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, pages 3193–3200. IEEE, 2011.

[152] Yulia Alexandrovna Hicks, Peter M Hall, and Andrew David Marshall. A method to add hidden Markov models with application to learning articulated motion. In *Proceedings of British Machine Vision Conference*, 2003.

[153] Paolo Favaro, Andrea Mennucci, and Stefano Soatto. Observing shape from defocused images. *International Journal of Computer Vision*, 52(1):25–43, 2003.

[154] Max Born and Emil Wolf. *Principles of Optics*. Pergamon, 1965.

[155] Alex Noel Joseph Raj. *Accurate Depth from Defocus Estimation with Video-Rate Implementation*. PhD thesis, School of Engineering, University of Warwick, 2009.

[156] NIKKOR 50mm f/1.4. In Nikon. Retrieved 14:16, 30/7/2014, from `http://www.nikonusa.com/en/Nikon-Products/Product/Camera-Lenses/NIKKOR-50mm-f%252F1.4.html`.

[157] Guppy. In Allied Vision Technologies. Retrieved 17:42, 28/7/2014, from `http://www.alliedvisiontec.com/us/products/cameras/firewire/guppy.html`.

[158] IEEE 1394 (AKA 'FireWire' & 'iLink'). In Jaycar Electronics. Retrieved 14:29, 30/7/2014, from `http://www.jaycar.com.au/images_uploaded/firewire.pdf`.

[159] 3D CAD SolidWorks Software SolidWorks. In Dassault Systems. Retrieved 14:39, 30/7/2014, from `http://www.solidworks.com/`.

[160] Rashid Minhas, Abdul Adeel Mohammed, and QM Jonathan Wu. Shape from focus using fast discrete curvelet transform. *Pattern Recognition*, 44(4):839–853, 2011.

[161] Ik-Hyun Lee, Muhammad Tariq Mahmood, Seong-O Shim, and Tae-Sun Choi. Optimizing image focus for 3D shape recovery through genetic algorithm. *Multimedia Tools and Applications*, 1:1–16, 2013.

[162] Subhasis Chaudhuri and Ambasamudram N. Rajagopalan. *Depth From Defocus: a Real Aperture Imaging Approach.* Springer, 1998.

[163] Liu Hong, Jia Yu, Cheng Hong, and Wei Sui. Depth estimation from defocus images based on oriented heat-flows. In *Proceedings of IEEE International Conference on Machine Vision*, pages 212–215. IEEE, 2009.

[164] Hao Wang, Fengyun Cao, Shuai Fang, Yang Cao, and Chunlong Fang. Effective improvement for depth estimated based on defocus images. *Journal of Computers*, 8(4):888–895, 2013.

[165] Qiu Feng Wu, Kuan Quan Wang, and Wang Meng Zuo. Depth from defocus using geometric optics regularization. *Advanced Materials Research*, 709:511–514, 2013.

[166] Masahiro Watanabe and Shree K Nayar. Telecentric optics for focus analysis. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 19(12):1360–1365, 1997.

[167] Max Born and Emil Wolf. *Principles of Optics: Electromagnetic Theory of Propagation, Interference and Diffraction of Light.* CUP Archive, 1999.

[168] Christopher D. Claxton and Richard C. Staunton. Measurement of the point-spread function of a noisy imaging system. *Journal of Optical Society of America A*, 25(1):159–170, 2008.

[169] Walter Gander and Walter Gautschi. *Adaptive quadrature–revisited.* BIT Numerical Mathematics, 2000.

[170] IEEE Acoustics Speech and Signal Processing Society. *Programs for Digital Signal Processing*. Digital Signal Processing Committee, 1979.

[171] Jae S. Lim. *Two-Dimensional Signal and Image Processing*. Prentice Hall, 1990.

[172] Sidney F. Ray. *Applied photographic optics: imaging systems for photography, film and video*. Focal Press, 1988.

[173] Hari N. Nair and Charles V. Stewart. Robust focus ranging. In *Proceedings of IEEE Conference on Computer Vision and Pattern Recognition*, pages 309–314, 1992.

[174] Rafael C. Gonzalez, Richard Eugene Woods, and Steven L. Eddins. *Digital Image Processing Using MATLAB*. New Jersey, Pearson Prentice Hall, 2004.

[175] German K. M. Cheung, Takeo Kanade, J-Y Bouguet, and Mark Holler. A real time system for robust 3D voxel reconstruction of human motions. In *Proceedings of IEEE Conference on Computer Vision and Pattern Recognition*, volume 2, pages 714–720. IEEE, 2000.

[176] Ronald G. Driggers, Melvin H. Friedman, and Jonathan Nichols. *Introduction to Infrared and Electro-optical Systems*. Artech House, 2012.

[177] Rick Parent. *Computer Animation: Algorithms and Techniques*. Newnes, 2012.

[178] D. P. Mukherjee and Debashish Jana. *Computer Graphics : Algorithms and Implementations*. PHI Learning Pvt. Ltd, 2010.

[179] Opencv. In OpenCV. Retrieved 08:05, 9/5/2014, from `opencv.org`.

[180] Chris Stauffer and W. Eric L. Grimson. Learning patterns of activity using real-time tracking. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 22(8):747–757, 2000.

[181] Liyuan Li, Weimin Huang, Irene YH Gu, and Qi Tian. Foreground object detection from videos containing complex background. In *Proceedings of the eleventh ACM international conference on Multimedia*, pages 2–10. ACM, 2003.

[182] Liyuan Li, Weimin Huang, IY-H Gu, and Qi Tian. Statistical modeling of complex backgrounds for foreground object detection. *IEEE Transactions on Image Processing*, 13(11):1459–1472, 2004.

[183] Viorel Badescu. *Modeling Solar Radiation at the Earth's Surface: Recent Advances*. Springer Science & Business Media, 2008.

[184] Paolo Fornasini. *The uncertainty in physical measurements: an introduction to data analysis in the physics laboratory*. Springer, 2008.

[185] Jose M. Chaquet, Enrique J Carmona, and Antonio Fernández-Caballero. A survey of video datasets for human action and activity recognition. *Computer Vision and Image Understanding*, 117(6):633–659, 2013.

[186] Xiaofei He and Partha Niyogi. Locality preserving projections. In *Proceedings of Neural Information Processing Systems Conference*, volume 16, pages 234–241, 2003.

[187] Liang Wang and David Suter. Analyzing human movements from silhouettes using manifold learning. In *Proceedings of IEEE International Conference on Video and Signal Based Surveillance*, pages 7–7. IEEE, 2006.

[188] Eunju Kim, Sumi Helal, and Diane Cook. Human activity recognition and pattern discovery. *Pervasive Computing, IEEE*, 9(1):48–53, 2010.

[189] Bhavani Thuraisingham, Latifur Khan, Mamoun Awad, and Lei Wang. *Design and Implementation of Data Mining Tools*. CRC Press, 2010.

[190] Zoran Zivkovic. Improved adaptive Gaussian mixture model for background subtraction. In *Proceedings of International Conference on Pattern Recognition*, volume 2, pages 28–31. IEEE, 2004.

[191] Oksana Samko, A David Marshall, and Paul L Rosin. Selection of the optimal parameter value for the Isomap algorithm. *Pattern Recognition Letters*, 27(9):968–979, 2006.

[192] Szymon Rusinkiewicz, Olaf Hall-Holt, and Marc Levoy. Real-time 3D model acquisition. In *ACM Transactions on Graphics*, volume 21, pages 438–446. ACM, 2002.

[193] Volumetric rendering of a skull. In BenR's Softimage Shaders. Retrieved 10:11, 5/6/2014, from `http://shaders.moederogall.com/`.

[194] Masahiro Watanabe and Shree K Nayar. Telecentric optics for constant magnification imaging. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 19:1360–1365, 1995.