

University of Warwick institutional repository: <http://go.warwick.ac.uk/wrap>

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

<http://go.warwick.ac.uk/wrap/3183>

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

Please scroll down to view the document itself.

Please refer to the repository record for this item for information to help you to cite it. Our policy information is available from the repository home page.

# Autonomous Surveillance in an Urban Environment

John Douglas Oliver

A thesis submitted in partial fulfilment of the requirements  
for the degree of Doctor of Philosophy in Engineering

University of Warwick

School of Engineering

May 2009



# Contents

<b>1</b>	<b>Introduction</b>	<b>1</b>
<b>2</b>	<b>Related Work</b>	<b>7</b>
2.1	The Wider Field . . . . .	12
2.2	Definition of the Environment . . . . .	15
2.3	Tracking Algorithms . . . . .	19
2.3.1	A-CMOMMT . . . . .	19
2.3.2	Shortest Escape Path . . . . .	22
2.4	Searching . . . . .	26
2.4.1	Jung . . . . .	28
<b>3</b>	<b>Simulation Platform</b>	<b>33</b>
3.1	Definition of the Environment . . . . .	34
3.2	Maps . . . . .	36
3.2.1	Map Data Pre-processing . . . . .	40




3.2.2	Physical Layout . . . . .	40
3.2.3	Road Network . . . . .	42
3.3	Simulation Implementation . . . . .	43
3.3.1	Basis . . . . .	43
3.3.2	Collision Detection . . . . .	44
3.3.3	Modifications . . . . .	45
3.3.4	Cluster . . . . .	48
3.4	Target Architecture . . . . .	48
3.5	Robot Architecture . . . . .	49
3.6	Control Architecture . . . . .	55
<b>4</b>	<b>Tracking Algorithms</b>	<b>59</b>
4.1	Combined Urban Tracker . . . . .	60
4.1.1	E-CMOMMT . . . . .	61
4.1.2	E-Jung . . . . .	68
4.2	Short Cut Path Algorithm (SCP) . . . . .	70
4.3	Branch . . . . .	76
4.4	No Movement . . . . .	80
4.5	Contribution . . . . .	81

<b>5 Experiments and Results</b>	<b>83</b>
5.1 Aims . . . . .	83
5.2 Design of experiments . . . . .	84
5.2.1 Test Areas . . . . .	86
5.2.2 Trial Structure . . . . .	95
5.3 Results . . . . .	95
5.3.1 Analysis of Maps . . . . .	95
5.3.2 Analysis by Algorithm . . . . .	120
5.3.3 Comparison by Problem . . . . .	123
<b>6 Critical Assessment and Further Work</b>	<b>127</b>
6.1 Contribution . . . . .	127
6.2 Critical Assessment . . . . .	130
6.2.1 Environment Construction . . . . .	131
6.2.2 Robot construction . . . . .	133
6.2.3 Data Limitations . . . . .	134
6.2.4 Control Problem . . . . .	135
6.3 Further Work . . . . .	135
6.3.1 Sensor Noise and Uncertainty . . . . .	136
6.3.2 Additional Agents . . . . .	136
6.3.3 Additional Robots . . . . .	137

<b>7 Conclusions</b>	<b>138</b>
<b>References</b>	<b>143</b>
<b>A Heat Maps</b>	<b>153</b>
A.1 Map 1 . . . . .	154
A.2 Map 2 . . . . .	158
A.3 Map 3 . . . . .	162
A.4 Map 4 . . . . .	166
<b>B Histograms of Effectiveness</b>	<b>170</b>
<b>C Example of Map Data</b>	<b>179</b>
<b>D List of Symbols</b>	<b>182</b>
D.1 General . . . . .	182
D.2 Symbols Used in E-CMOMMT Algorithm . . . . .	183
D.3 Symbols Used in E-Jung Algorithm . . . . .	183
D.4 Symbols Used in SCP Algorithm . . . . .	184
D.5 Symbols Used in Branch Algorithm . . . . .	185

# List of Figures

2.1	Field breakdown . . . . .	9
2.2	Left: Illustration of the environment proposed in (Parker, 2002). Right: An example of targets being observed and equations produced. . . . .	17
2.3	Example of a vector field containing one attractive and one repulsive object(Generated using (VFDFLA, 2009)). . . . .	19
2.4	A-CMOMMT Forces. . . . .	21
2.5	Local minima restrictions of A-CMOMMT. Ordnance Survey© Crown Copyright. All rights reserved. . . . .	22
2.6	Escape path tree example . . . . .	24
2.7	Escape path tree for Figure 2.6 . . . . .	24
2.8	Example of how the distance from robot to the Occlusion Vertex affects the robots movement . . . . .	26
2.9	Example of an impossible search for a restricted robot due to there being no valid position in which the robot can see the target. Ordnance Survey© Crown Copyright. All rights reserved. . . . .	28
2.10	Example of how the function $D_t$ will act for the scenario given. . . .	30
2.11	An example of a tracking situation and the corresponding values for the Jung equations for each area. . . . .	31
3.1	Left: Illustration of the environment space for the given problem. Right: Space demonstrated on a map. Ordnance Survey© Crown Copyright. All rights reserved. . . . .	37

3.2	Left: Aerial photograph of real environment. Right: Simulated view of environment. Ordnance Survey© Crown Copyright. All rights reserved. . . . .	38
3.3	Node positioning on the road network. Ordnance Survey© Crown Copyright. All rights reserved. . . . .	39
3.4	Illustration of adding width to walls. Original path on the left, polygon formed on the right . . . . .	41
3.5	Left: Robot and target in simulated environment. Red denotes space that cannot be traversed. Centre: Robots permissible space. Right: Targets permissible space. Ordnance Survey© Crown Copyright. All rights reserved. . . . .	46
3.6	Illustration of the target used in the simulation . . . . .	49
3.7	Illustration of the robots sensor configuration. . . . .	53
3.8	Data Flow Architecture . . . . .	55
3.9	Control Program layout . . . . .	58
4.1	Switching behaviour of the Jung/A-CMOMMT algorithms . . . . .	60
4.2	The summation of forces using the A-CMOMMT algorithm . . . . .	62
4.3	Left: Robot to target weighting function $\omega_{lk}$ . Right: robot to robot weighting function $\psi_{li}$ . Taken from (Parker, 1999). . . . .	62
4.4	An example of a local maxima forming due to the restrictions of A-CMOMMT. Ordnance Survey© Crown Copyright. All rights reserved. . . . .	63
4.5	Robot force( $F_o$  ) , Cartesian target force( $F_{tc}$  ) projected along the road network and topological target force( $F_{tt}$  ) . Ordnance Survey© Crown Copyright. All rights reserved. . . . .	67
4.6	An example of possible short cut path. Ordnance Survey© Crown Copyright. All rights reserved. . . . .	71
4.7	Target urgency over time. X,Y axis show Cartesian distance from the centre, where the target was last observed. . . . .	75
4.8	Illustration of the switching between algorithms for the SCP tracker. . . . .	76

4.9	Urgency of nodes using Branch positioning. Larger nodes are more urgent. Ordnance Survey© Crown Copyright. All rights reserved. .	80
5.1	Labelled image of Map 1. Ordnance Survey© Crown Copyright. All rights reserved. . . . .	89
5.2	Labelled image of Map 2. Ordnance Survey© Crown Copyright. All rights reserved. . . . .	91
5.3	Labelled image of Map 3. Ordnance Survey© Crown Copyright. All rights reserved. . . . .	92
5.4	Labelled image of Map 4. Ordnance Survey© Crown Copyright. All rights reserved. . . . .	94
5.5	Effectiveness against number of robots for map 1. Speed relative to the robots labelled above each graph. . . . .	98
5.6	Close up of area A3 on Map 1 (Figure 5.1). Colour map scaled to highlight the differences in performance. . . . .	100
5.7	Areas of highest risk for map 1(Figure 5.1) and various speed ratios (highest risk is denoted by red). Ordnance Survey© Crown Copyright. All rights reserved. . . . .	101
5.8	Effectiveness against number of robots for map 2. Speed relative to the robots labelled above each graph. . . . .	105
5.9	Normalised heat map of map 2 with heavily obscured areas labelled.	106
5.10	Areas of highest risk for map 2(Figure 5.2) and various speed ratios (highest risk is denoted by red). Ordnance Survey© Crown Copyright. All rights reserved. . . . .	107
5.11	Effectiveness against number of robots for map 3. Speed relative to the robots labelled above each graph. . . . .	110
5.12	Areas of highest risk for map 3(Figure 5.3) and various speed ratios (highest risk is denoted by red). Ordnance Survey© Crown Copyright. All rights reserved. . . . .	111
5.13	Normalised heat map of map 3. . . . .	112
5.14	Close up of area A1 on Map 4. Colour maps have been scaled to highlight the differences in performance. . . . .	113
5.15	Normalised heat map of map 4 with heavily obscured areas labelled.	113

5.16	Effectiveness against number of robots for map 4. Speed relative to the robots labelled above each graph. . . . .	115
5.17	Areas of highest risk for map 4(Figure 5.4) and various speed ratios (highest risk is denoted by red). Ordnance Survey© Crown Copyright. All rights reserved. . . . .	116
5.18	Effectiveness of algorithms against speed ratio of robot. Differing numbers of robots are shown on different lines. . . . .	119
5.19	Heat map of Map 2 (Figure 5.2) illustrating Branch positions inability to cope with shortcut paths . . . . .	121
5.20	An area of Map 4 that proved particularly problematic for the Branch algorithm (Figure 5.4). . . . .	121
5.21	An area of Map 3 that proved particularly problematic for the Branch algorithm(Figure 5.3). . . . .	122
5.22	Positions at which the targets became lost and their movements before(Black) and after becoming lost in an area of Map 2. Top: CUT algorithm, Middle: SCP algorithm, Bottom: Branch algorithm. Ordnance Survey© Crown Copyright. All rights reserved. . . . .	124
5.23	All positions at which the target became lost for the area on Map2. Ordnance Survey© Crown Copyright. All rights reserved. . . . .	125
A.1	Heat map of lost positions for the CUT algorithm on map 1. Speed of the target relative to the robots listed above each diagram. Values have been normalised to highlight areas that have a high probability of losing the target as opposed to areas that the target spent a significant amount of time unobserved. Ordnance Survey© Crown Copyright. All rights reserved. . . . .	154
A.2	Heat map of lost positions for the SCP algorithm on map 1. Speed of the target relative to the robots listed above each diagram. Values have been normalised to highlight areas that have a high probability of losing the target as opposed to areas that the target spent a significant amount of time unobserved. Ordnance Survey© Crown Copyright. All rights reserved. . . . .	155
A.3	Heat map of lost positions for the Branch algorithm on map 1. Speed of the target relative to the robots listed above each diagram. Values have been normalised to highlight areas that have a high probability of losing the target as opposed to areas that the target spent a significant amount of time unobserved. Ordnance Survey© Crown Copyright. All rights reserved. . . . .	156



A.4	Heat map of lost positions for the No Movement on map 1. Speed of the target relative to the robots listed above each diagram. Values have been normalised to highlight areas that have a high probability of losing the target as opposed to areas that the target spent a significant amount of time unobserved. Ordnance Survey© Crown Copyright. All rights reserved. . . . .	157
A.5	Heat map of lost positions for the CUT algorithm on map 2. Speed of the target relative to the robots listed above each diagram. Values have been normalised to highlight areas that have a high probability of losing the target as opposed to areas that the target spent a significant amount of time unobserved. Ordnance Survey© Crown Copyright. All rights reserved. . . . .	158
A.6	Heat map of lost positions for the SCP algorithm on map 2. Speed of the target relative to the robots listed above each diagram. Values have been normalised to highlight areas that have a high probability of losing the target as opposed to areas that the target spent a significant amount of time unobserved. Ordnance Survey© Crown Copyright. All rights reserved. . . . .	159
A.7	Heat map of lost positions for the Branch algorithm on map 2. Speed of the target relative to the robots listed above each diagram. Values have been normalised to highlight areas that have a high probability of losing the target as opposed to areas that the target spent a significant amount of time unobserved. Ordnance Survey© Crown Copyright. All rights reserved. . . . .	160
A.8	Heat map of lost positions for the No Movement on map 2. Speed of the target relative to the robots listed above each diagram. Values have been normalised to highlight areas that have a high probability of losing the target as opposed to areas that the target spent a significant amount of time unobserved. Ordnance Survey© Crown Copyright. All rights reserved. . . . .	161
A.9	Heat map of lost positions for the CUT algorithm on map 3. Speed of the target relative to the robots listed above each diagram. Values have been normalised to highlight areas that have a high probability of losing the target as opposed to areas that the target spent a significant amount of time unobserved. Ordnance Survey© Crown Copyright. All rights reserved. . . . .	162

A.10	Heat map of lost positions for the SCP algorithm on map 3. Speed of the target relative to the robots listed above each diagram. Values have been normalised to highlight areas that have a high probability of losing the target as opposed to areas that the target spent a significant amount of time unobserved. Ordnance Survey© Crown Copyright. All rights reserved. . . . .	163
A.11	Heat map of lost positions for the Branch algorithm on map 3. Speed of the target relative to the robots listed above each diagram. Values have been normalised to highlight areas that have a high probability of losing the target as opposed to areas that the target spent a significant amount of time unobserved. Ordnance Survey© Crown Copyright. All rights reserved. . . . .	164
A.12	Heat map of lost positions for the No Movement on map 3. Speed of the target relative to the robots listed above each diagram. Values have been normalised to highlight areas that have a high probability of losing the target as opposed to areas that the target spent a significant amount of time unobserved. Ordnance Survey© Crown Copyright. All rights reserved. . . . .	165
A.13	Heat map of lost positions for the CUT algorithm on map 4. Speed of the target relative to the robots listed above each diagram. Values have been normalised to highlight areas that have a high probability of losing the target as opposed to areas that the target spent a significant amount of time unobserved. Ordnance Survey© Crown Copyright. All rights reserved. . . . .	166
A.14	Heat map of lost positions for the SCP algorithm on map 4. Speed of the target relative to the robots listed above each diagram. Values have been normalised to highlight areas that have a high probability of losing the target as opposed to areas that the target spent a significant amount of time unobserved. Ordnance Survey© Crown Copyright. All rights reserved. . . . .	167
A.15	Heat map of lost positions for the Branch algorithm on map 4. Speed of the target relative to the robots listed above each diagram. Values have been normalised to highlight areas that have a high probability of losing the target as opposed to areas that the target spent a significant amount of time unobserved. Ordnance Survey© Crown Copyright. All rights reserved. . . . .	168

A.16	Heat map of lost positions for the No Movement on map 4. Speed of the target relative to the robots listed above each diagram. Values have been normalised to highlight areas that have a high probability of losing the target as opposed to areas that the target spent a significant amount of time unobserved. Ordnance Survey© Crown Copyright. All rights reserved. . . . .	169
B.1	Map 1 results. Horizontal axis denotes effectiveness, vertical denotes frequency. . . . .	171
B.2	Map 1 results. Horizontal axis denotes effectiveness, vertical denotes frequency. . . . .	172
B.3	Map 2 results. Horizontal axis denotes effectiveness, vertical denotes frequency. . . . .	173
B.4	Map 2 results. Horizontal axis denotes effectiveness, vertical denotes frequency. . . . .	174
B.5	Map 3 results. Horizontal axis denotes effectiveness, vertical denotes frequency. . . . .	175
B.6	Map 3 results. Horizontal axis denotes effectiveness, vertical denotes frequency. . . . .	176
B.7	Map 4 results. Horizontal axis denotes effectiveness, vertical denotes frequency. . . . .	177
B.8	Map 4 results. Horizontal axis denotes effectiveness, vertical denotes frequency. . . . .	178
C.1	Example of OSMasterMap road node data. (Ordnance Survey ©Crown Copyright. All rights reserved) . . . . .	180
C.2	Example of OSMasterMap road dimensions. (Ordnance Survey ©Crown Copyright. All rights reserved) . . . . .	181

# List of Tables

4.1	Algorithm for assigning branch nodes to robots . . . . .	79
5.1	Effectiveness for map 1.	
	$Effectiveness = \frac{\text{Time samples in which the target was observed}}{\text{Total simulation time}}$ . . . . .	97
5.2	Effectiveness for map 2.	
	$Effectiveness = \frac{\text{Time samples in which the target was observed}}{\text{Total simulation time}}$ . . . . .	104
5.3	Effectiveness for map 3.	
	$Effectiveness = \frac{\text{Time samples in which the target was observed}}{\text{Total simulation time}}$ . . . . .	109
5.4	Effectiveness for map 4.	
	$Effectiveness = \frac{\text{Time samples in which the target was observed}}{\text{Total simulation time}}$ . . . . .	114
5.5	Mean effectiveness over over all trials . . . . .	117
5.6	Comparison of algorithms over all maps, comparing the speed ratio of the target relative to the robots. . . . .	118

# Acknowledgements

I would like to thank the many people who helped in the completion of this thesis. In particular Prof Ken Young and Dr Peter Jones for their guidance and help.

I would also like to thank Simon Hammond and the Department of Computer Science for providing the cluster systems that made this work possible. I would also like to thank the numerous contributors to the open source projects used in this work.

Lastly I would thank Alison and my family for their continued support.

# **Declaration**

I declare that all the work described in this report was undertaken by myself (unless otherwise acknowledged in the text) and that none of the work has been previously submitted for any academic degree. All sources of quoted information have been acknowledged by means of references.

# Abstract

A number of algorithms have been developed in the past for the purposes of target tracking, these have generally been for simple polygonal environments. However as the technology for autonomous vehicles develops for use in the real world these tracking algorithms need to be tested in larger more realistic environments.

This work investigates the use of tracking algorithms to control a team of road based robotic platforms, tracking pedestrian targets in urban environments.

The novelty of this work is in the identification of the aspects of the environment that affect target tracking algorithms, and modifying the algorithms to cope with them. Problems such as the frequent stalemates reached as an algorithms movement is limited by the highly restricted movement space or the identification of “short cuts” in which the target can take much shorter routes between positions than the robots. Algorithms are developed that overcome these limitations and they are tested in a simulation that is an accurate representation of a real environment. The algorithms are partly based on existing work and are developed extensively to be suitable for the environment. These algorithms are tested for their ability to maintain visual contact with the target.

The scenario is tested with varying numbers of robots, speeds and locations. Three algorithms were developed and tested, one built as an extension of existing target tracking algorithms (Combined Urban Tracker) and another two algorithms developed specifically for this environment (Short Cut Path, and Branch).

It is concluded that the Combined Urban Tracker and Short Cut Path algorithms performed comparably with a less than 0.3% difference in performance between the two both averaging roughly 54% effectiveness overall, however the Branch algorithm fared significantly worse averaging only 43% overall. The areas within the environment that give significant problems are large open spaces and areas that are significantly occluded from the road network.

This work provides a platform on which further development in this area can be based in order to progress tracking algorithms towards being of practical use.

# Chapter 1

## Introduction

Autonomous surveillance has been of increasing interest in recent years, particularly with respect to producing platforms that can perform surveillance in the physical world. Recent projects such as the Ministry Of Defence Grand Challenge (MOD, 2006) and the European Land Robot Trials (ELRT, 2008b) are both pushing towards developing autonomous platforms on which it is possible to perform automated reconnaissance and surveillance. The motivation for exploring the use of robotic platforms is the increased efficiency and cost savings that can be achieved with automation. Of great importance is also increased safety to personnel, particularly when operating in dangerous environments in which it is necessary to locate people, however it would be dangerous to explore without prior information. The general goal of such programs are to collect as much information about the environment as possible, and most importantly to be able to identify items of interest, generally people, and to then report this information. Then as an extension, track the move-



ments of these people in order to provide up to date information. It is therefore envisaged that the ultimate use for this work will be in military operations in unfamiliar, foreign, hostile urban environments. By providing a more complete picture of the state of the environment to the user, it is hoped that the ultimate benefit of this work is to make operating in such environments safer. With a greater degree of information about the environment, personal will be able to plan more effectively in these problematic urban areas where visibility is extremely limited.

The goal of autonomous surveillance could be achieved with a wide range of configurations, from static sensors such as CCTV cameras ((Collins et al., 2001; Siebel and Maybank, 2002), also various case studies on CCTV use provided in (Lipton et al., 2003)) to mobile robots (Rybski et al., 2000; Grocholsky et al., 2006). Various configurations have been explored and as technology progresses robots are becoming a more practical and affordable solution. Teleoperated robots for the purposes of reconnaissance are already in operation (Voth, 2004), however as the requirements of the Ministry Of Defence Grand Challenge made clear, it is of value for the system to have as great a degree of automation as possible in order to reduce the burden of operation from the personnel. This thesis therefore is attempting to provide a stepping stone towards achieving such a goal.

Broadly the types of mobile systems envisaged are either ground based, aerial or a hybrid (Grocholsky et al., 2006; Sukhatme et al., 2001). This work focuses on the ground based domain, although it is acknowledged that aerial surveillance can be particularly effective due to avoiding limitations such as occlusion and obstacles. However ground based systems

are likely to be able to contain more robust sensors and processing power due to the weight limitations of an aerial vehicle. Additionally ground based systems are likely to be able to get closer to areas of interest to collect more detailed sensor information and possibly directly respond to the situation. It should also be noted that there has been a significant amount of work exploring the aerial surveillance domain.

A number of prerequisite technologies are required in order to perform automated surveillance including the physical platform, sensors, sensor processing and algorithms for the coordination and movement of the robot. Work is being carried out in all these areas and advances are being made that will allow autonomous surveillance to become a reality. The building of physical platforms in particular is a fairly well established area with a number of commercial off the shelf products already available on the market, many from the IRobot Corporation, particularly in the all terrain area such as the packbot (Yamauchi, 2004) and ATRV-2 (as used in (Rybski et al., 2000)). Although many robotic platforms exist that can travel along the roads, little literature can be found on the design of such platforms. The most common method of producing a road based platform appears to be to take a suitable car and modify it by adding sensors and automatic control mechanisms (Kammel et al., 2008; Stone et al., 2007; Leonard et al., 2007). Sensor interpretation is also a highly researched area, particularly with respect to static CCTV style sensors. The detection and tracking of objects and events of interest in CCTV images has been particularly well explored (Collins et al., 2001; Siebel and Maybank, 2002; Fuentes and Velastin, 2006). On mobile platforms there

has also been incredible amounts of work in the field of mapping the environment and location of the robot itself (overview of various methods in (Filliat and Meyer, 2003) and (Meyer and Filliat, 2003)), particularly in the popular combined field of simultaneous localisation and mapping (SLAM, (Durrant-Whyte and Bailey, 2006; Bailey and Durrant-Whyte, 2006)). For the latter of the fields mentioned, algorithms for the coordination and movement of the robots, it has been found that this area is particularly unexplored with respect to achieving the goal of autonomous surveillance. Work has been performed on various similar and generally simplified environments (Jung, 2005; Parker, 1999), however due to the complexity of the environment at hand the existing work is viewed as insufficient to meet the demands of the task. This work therefore focuses upon the latter problem of the coordination and movement of robots in order to attempt to progress the state of the art to a point that is sufficient for the task.

Movement algorithms would be required that achieve various tasks such as searching algorithms for the purposes of finding objects of interest. Also tracking algorithms for the following of targets once found. In extremely adversarial scenarios such as the pursuit evasion problems, capturing of the target is required. The area of tracking targets has been found to be lacking, particularly when exploring the urban environment, where the target is capable of potentially more agile movement as well as traversing terrain that a robot is not. The goal of this study is to therefore explore this problem of tracking a target using a team of autonomous robots, where the target is able to traverse terrain that the robots cannot.

The form of this target is therefore a pedestrian being tracked by road based robots.

The focus of this work is thus to maintain visual contact of a pedestrian target in an urban environment using a team of road based robots.

In the environment at hand, the target has relative freedom of the environment with the ability to traverse all spaces with the exception of obstacles such as buildings, walls and fences. The robots are however confined to the road network. The robots have sonar, laser range finders, GPS and a target detection system, each have limited fields of view and range. The structure of the team is decentralised with each robot having complete autonomy to make its own decisions, no mechanism is provided to allow a robot to directly induce a decision or action in another. However, a communications channel is provided to allow the robots to pass appropriate information between team members in order to make effective decisions. This information will influence the actions of other robots but will not instruct them.

This work is undertaken as an engineering piece, aimed towards developing algorithms that can be utilised with limited hardware and reasonable sensors. One of the goals is therefore to develop decentralised algorithms that scale well with the number of robots and that uses a minimum of communications bandwidth. Due to requiring the test to be an accurate representation of reality, a minimum of simplifications have been made in the construction of the environment.

Relevant work in the area of autonomous target tracking is discussed and

limitations of existing algorithms are identified. Algorithms are produced that are based upon existing work and developed further to be suitable to this environment. Original algorithms are also developed, specifically designed to account for aspects of an urban environment that are not present in previously explored work. A representative simulation is developed in which to test the algorithms. The simulations are based on real environments in the UK using data from Ordnance Survey© and use realistic values for sensor performance. These algorithms are tested and their performance compared. During the experimentation the maps used, speed of the target, number of robots and the starting positions are varied to both eliminate them as a factor that biases the results and explore their effect upon the algorithms performance. A critical assessment of the work is performed, identifying the areas of improvement that could be made and identification of future work to address these issues.

Throughout this work a number of maps are displayed, all maps are property of Ordnance Survey© Crown Copyright. All rights reserved.

## Chapter 2

### Related Work

Before discussing the field it should be noted that throughout the thesis the term sensor platform is used to refer to a static device that merely collects information of the surrounding environment. This is in contrast to a “robot”, which is used to refer to a device that not only senses but actively moves and reacts to changes in the environment. Sensor platforms should also be distinguished from the term sensor, a sensor refers to an element that can detect a property of the environment such as a camera or sonar. A sensor platform can therefore contain one or more sensors.

Robotic surveillance has been explored in a variety of forms. These have varied from simulation exercises to practical implementations in hardware. The problem can be categorised in a number of forms. Figure 2.1 provides a break down of the various aspects that can be varied while exploring the task of automated surveillance. Broadly the work in this area can be classified by: the domain of interest, the dimensionality

of the problem explored, the sensors movement and capabilities and the type of environment that is explored. These are broken down further to show some of the common choices explored for each parameter. Some of these parameters require further explanation: in the domain of interest category sensor interpretation is generally the concept of analysing sensor data to try to identify and extract the location of objects of interest, robot placement is deciding upon the position to locate robots (or static sensor platforms) in order to maximise a metric such as the total area monitored (Poduri and Sukhatme, 2004), tracking and data association involves attempting to determine an objects movement as well as possibly trying to maintain the identity of multiple tracked objects as they travel through an environment, robot movement and coordination is then how to manoeuvre robots in order to best keep track of the targets. Cellular space refers to environments where the targets or robots can take position as relatively coarse positions such as in a grid formation (Chakrabarty et al., 2002). Graph based environments are where the environment is formed as a series of vertices connected by edges, such as in the GRAPH-CLEAR problem (formalised in (Kolling and Carpin, 2007)) where agents attempt to clear a graph by moving a team in such a manner that no intruder could possibly remain undetected. Vertices and edges are generally used to represent rooms and areas and the connections between them.

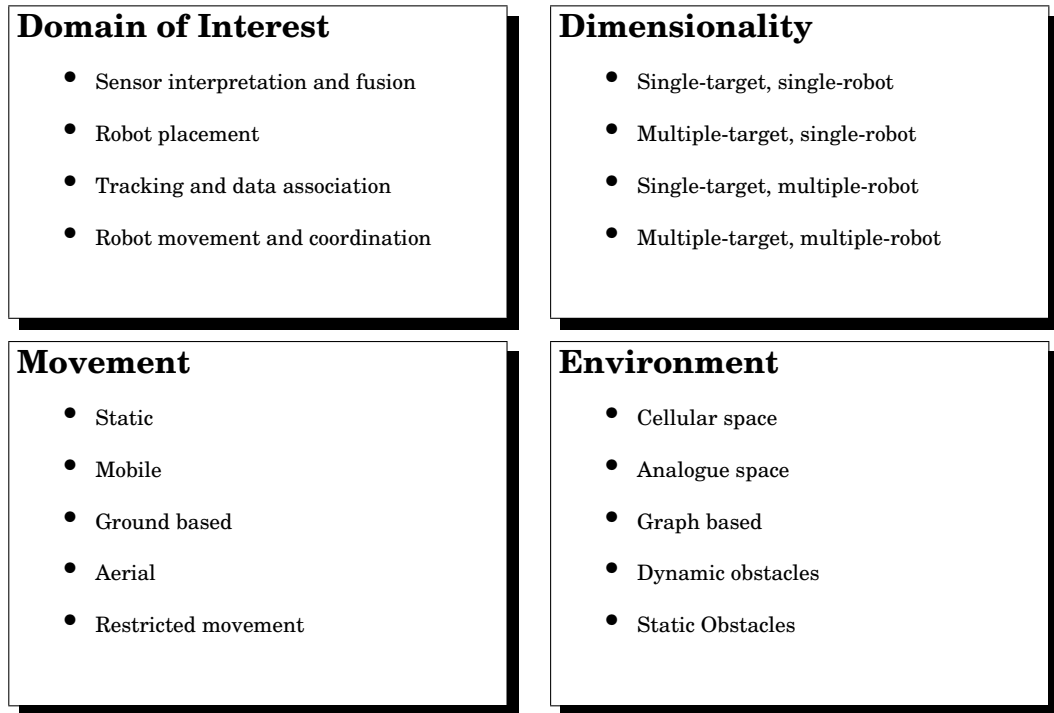


Figure 2.1: Field breakdown

It should also be noted that although ground based and aerial mobility were listed as separate parameters there has also been work on hybrid systems that incorporated both (Grocholsky et al., 2006).

Due to the large number of variables this leads to a great number of possible combinations for research, and a number have been explored in the past and will be briefly discussed in Section 2.1. Then more directly related work is reviewed in Sections 2.3 and 2.4.

The environment that is explored in this work is as follows:

- Robot movement and coordination.
- Multiple mobile robots and a single target (MRST).



- Analogue space.
- Static obstacles (excluding the robots and target).
- Restricted movement of the robot relative to the target.

Little work has been performed specifically upon this domain in the past. This specific environment is important particularly due to the choice of restricted movement of the target relative to the robot, this is a more accurate representation of the challenges that would be encountered were surveillance to be attempted in an urban environment than previous work, as well as closer to the requirements that would be needed to address problems such as the Ministry Of Defence Grand Challenge (MOD, 2006) and the European Land Robot Trials (ELRT, 2008b). The Grand Challenge was a competition set by the Ministry Of Defence and was aimed at producing a system that could locate threats such as snipers, explosives, vehicles with mounted weaponry, groups of enemies as well as civilians. This was to be performed within an urban war zone, with the intention of then relaying their locations to local ground troops. It was also a major requirement that such a platform should be automated and not teleoperated in order to keep the burden of operation to a minimum. In the trial itself, only static objects were required to be detected however it is a natural extension to the requirements of such a system that once a threat has been identified that it be tracked in order to keep an accurate account of its location. The European Land Robot trials is specified as a European equivalent to the DARPA Grand Challenge (Due to entry into the DARPA GC being difficult for non-US teams) and is aimed at assess-

ing the current state of the art in robotics. A number of scenarios are put forward in the ELROB trials, the most relevant of which is the “Camp security” trial, in which a robot is required to “Search pre-designated area, detect, report and monitor moving objects of interest” (ELRT, 2008a) in an urban and semi-urban environment. The objectives of this challenge are detailed as: “Detect and report intruders, Pursue intruders, Acquire position and imagery of intruders and transmit to control station” (ELRT, 2008a).

There are a number of prerequisites in order to explore tracking on such a large scale. If a test using an actual implementation was sought a suitable environment and platforms on which to perform the trials would be required, both of which would be difficult and costly to obtain. To explore in simulation requires firstly the required computing power to execute a simulation on such a large environment. Secondly suitable map data is required that can accurately represent a real environment. It is proposed that primarily the lack of availability of such data in the past is the reason that such work has not been explored, however advances in computing power also make the ability to explore this situation easier.

For the sake of practicality a few compromises have been made and are addressed in Chapter 6. This work however does not attempt to address all the issues required to solve performing surveillance in an urban environment, merely to progress towards this final goal.

## 2.1 The Wider Field

The general goal of Multiple Robot Single Target(MRST) problems is to use the redundant advantage of numbers to increase the accuracy of the localisation estimate (Spletzer and Taylor, 2003) as opposed to using the redundant advantage of numbers to overcome the restrictions that the environment places upon the robot.

Sensor interpretation is one of the most active fields of surveillance; the predominant methods are in the area of image analysis which attempt to identify people (Siebel and Maybank, 2002), crowds or vehicles (Maurin et al., 2005; Reisman et al., 2004) from images in order to track them through an environment. These often use techniques such as optical flow or image subtraction. Sensor platform configurations range from static CCTV style images to mobile sensors (robots). Current state of the art visual systems cannot guarantee identification of a human target, particularly across multiple sensor platforms however research is being carried out in this field (Siebel, 2003; Haritaoglu et al., 2000; Lipton et al., 1998) with growing success and it is believed that such systems will become available. It is therefore viewed as acceptable to assume that such a vision system is provided to the robot. This area is not within the scope of this thesis as it is assumed that a visual sensor system capable of identifying the target is available. It is known that through wall radar (Lubecke et al., 2007) has been developed for urban scenarios and would potentially provide a benefit however for this work only basic sensors are presumed available.

Sensor coverage is a significantly related topic, this focuses on the placement of sensor platforms in order to maximise the overall coverage of the environment (Poduri and Sukhatme, 2004; Meguerdichian et al., 2001). Sensor coverage often uses wireless sensor networks ((Culler et al., 2004) provides a general overview) and generally does not intend to actively follow a target, however by covering the entire space or key areas and sharing information it is able to track targets throughout. This work generally approaches the problem via a large number of sensor platforms, often more than 100 sensor platforms are proposed and used in tests (Wang et al., 2006). The number is justified by the use of small scale devices of limited cost, ability and power consumption (such as the Robomote (Sibley et al., 2002) or the Urban Emergency Response System (Shi et al., 2005)). This is in contrast to the approach taken in this work that proposes a small number of relatively large and inevitably costly devices. Ultimately it is a question of whether a large number of cheap robots can match or out perform fewer more capable devices, and if there is a cost benefit to either option. Small platforms will most likely be unable to navigate rough terrain or even moderately sized obstacles, this will lead to similar issues to the problem that is explored in this work where targets are able to traverse areas that the robot is unable to. It is currently not known if either the highly distributed approach or small team approach has an advantage, as it is very dependant upon the cost and capabilities of a particular implementation. This thesis will concentrate on the latter option. A potentially useful concept from this area is the barrier coverage problem (introduced in (Gage, 1992), overview in (Cardei and Wu, 2006)). This attempts to cover a zone so that a target may not pass the cordon

without being detected. If we were to take the possibility of a target hiding in an unobservable area then this concept is useful to ensure that the target does not leave that zone without being observed, sweep coverage (Gage, 1992) is also introduced which is essentially a moving barrier.

Pursuit evasion (Isler et al., 2005; Guibas et al., 1999; Cheng, 2003) also provides a very similar problem, however due to the additional complexity of our environment the targets will not be evasive, consequently these algorithms will not directly be applicable to this thesis. Many problems in this domain also deal with capturing the target (Oh et al., 2007) whereas the goal in this work is merely to keep the target within the field of view.

Autonomous air vehicles have also been of particular interest in the area of autonomous surveillance (Frew, 2007). Such vehicles overcome many of the inherent problems of ground based surveillance such as environment occlusion, restriction of movement via obstacles and localisation that can be difficult using methods such as GPS in an “urban canyon”. Hegazy (Hegazy, 2004) explores agent placement with specific reference to urban environments, his work points out that although a birds eye view removes a number of the problems of environment occlusion there are still significantly differing levels of occlusion depending upon the type of urban environment even to an aerial robot. This explores two types of environment, suburban type areas with many low buildings such as two story houses, secondly areas with many high rise buildings that provide more significant problems to aerial robots.

## 2.2 Definition of the Environment

One of the most relevant pieces of work is the CMOMMT (Cooperative Multi-robot Observation of Multiple Moving Targets) work (Parker, 2002) due to the fact that it formalises an environment for surveillance that provides a similar environment to this work, albeit in a simpler form. This can be built upon to formalise the environment used in this work. The definition describes a relatively simple environment containing convex objects (shapes with no internal angles on its perimeter). As this caters for the most general of problems (multiple robot multiple target) the algorithm is applicable to the intended work since the multiple robot single target problem is a subset of this. This work establishes the method of comparison that has been adopted for this thesis: *“to minimise the total time in which targets escape observation by some robot team member”* (Parker, 2002).

The problem is formalised as follows:

$S$ :	a two-dimensional, bounded, enclosed spatial region
$V$ :	a team of $m$ robot vehicles, $v_i$ ; $i = 1, 2 \dots m$ , with $360^\circ$ field of view observation sensors that are noisy and of limited range
$O(t)$ :	a set of $n$ targets, $o_j(t)$ , $j = 1, 2, \dots n$ such that target $o_j(t)$ is located within region $S$ at time $t$

$$B(t) = [b_{ij}(t)]_{m \times n} \quad (2.1)$$

$$b_{ij}(t) = \begin{cases} 1 & \text{if robot } v_i \text{ is observing target } o_j(t) \text{ in } S \text{ at time } t \\ 0 & \text{otherwise} \end{cases} \quad (2.2)$$

$$g(B(t), j) = \begin{cases} 1 & \text{if there exists an } i \text{ such that } b_{ij}(t) = 1 \\ 0 & \text{Otherwise} \end{cases} \quad (2.3)$$

$$A = \sum_{t=1}^T \sum_{j=1}^n \frac{g(B(t), j)}{T} \quad (2.4)$$

*Equations and descriptions taken verbatim from (Parker, 2002).*

This is then illustrated in Figure 2.2. As shown on the left of Figure 2.2, the environment is two dimensional and entirely enclosed ( $S$ ), contained within the environment is a set of targets ( $O$ ) and a set of robots ( $V$ ). Those that lie within the collective field of view of the robots are regarded as under observation ( $g$ ). Equation 2.1 defines a matrix over all targets and robots such that  $B(t)_{x,y}$  is 1 if robot  $x$  is observing target  $y$  (Illustrated on the right of Figure 2.2). The matrix  $B(t)$  therefore describes which

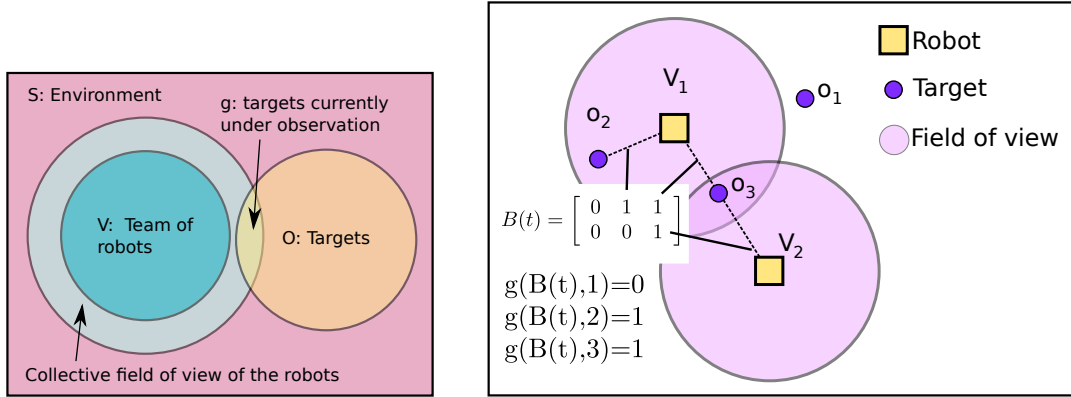


Figure 2.2: Left: Illustration of the environment proposed in (Parker, 2002). Right: An example of targets being observed and equations produced.

robots are observing which targets. This is then used by equation 2.3, where the function  $g(B(t), j)$  defines whether or not an individual target is being observed by any of the robots, therefore identifying those that are within the collective field of view of the team. This is shown on the right of Figure 2.2, where two robots are tracking three targets. The targets  $o_2$  and  $o_3$  are under observation, producing ones in the appropriate rows and columns of  $B(t)$  and the first column remaining zero to indicate that  $o_1$  is not under observation. The columns are then combined in the function  $g(B(t), j)$ , showing a zero overall for the first target and ones for the other two indicating that they are under observation.

The equations then culminate in equation 2.4, which calculates the percentage of time in which each target remains under observation, then sums this for each target to produce a metric that defines how successfully the team tracked the targets. This metric would range from 0 to  $n$ ,  $n$  indicating at no point in time did any of the targets escape observation, 0 indicating at no time were the targets observed.



The problem is then defined as maximising  $A$  in equation 2.4, that is to maximise the number of targets that are being observed throughout the trial.

The  $A$  value is also used as the metric for success and comparison of various strategies. For this purpose  $A$  is normalised since it is relative to the number of targets to  $\frac{A}{n}$ , which is the average percentage of targets that are observed over the trial. This metric will also be adopted for this work. For the case of a single target,  $j$  will always equal 1; this causes the first summation of equation 2.4. over the the  $A$  metric simplifies to equation 2.5, which is essentially the percentage of time intervals that the target is observed.

$$A = \sum_{t=1}^T \frac{g(B(t))}{T} \quad (2.5)$$

The work of Parker (Parker, 2002) also illustrates a number of important factors. One of which is the need for a simple and easily distributed algorithm to reduce reliance on centralised processing and communications as well as preventing the complexity of the solution rapidly increasing with the number of targets and robots.

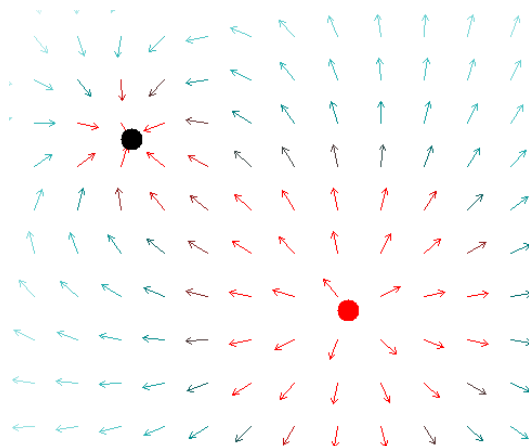


Figure 2.3: Example of a vector field containing one attractive and one repulsive object(Generated using (VFDFLA, 2009)).

## 2.3 Tracking Algorithms

### 2.3.1 A-CMOMMT

As previously stated, this work attempts to produce a solution where the computational complexity scales well with the number of targets and robots, to this end a local vector field approach is taken. A vector field is shown in Figure 2.3, that contains a single repulsive force and a single attractive force, the arrows then show the resultant force that acts upon a robot and thus the direction in which it will travel.

The vector field in this work is formed as a sum of a number of sub-goals, as described by (Reif and Wang, 1999). This technique of summing sub goals and behaviours produces very scalable and robust solutions. The scalability of vector fields is due to the fact that each object of interest (i.e. target or robot) produces only a single force each of which is then combined with the simple operation of summation. Calculating this force

for each robot then scales linearly with the number of objects of interest in the environment due to each new object simply producing a single additional force to the calculation.

Firstly the robot is attracted to local targets to attempt to keep them within range of the robot, secondly the robots are repelled from each other in order to prevent the robots clustering together and making redundant observations. The equation used for this behaviour is shown in equation 2.6.  $\Delta$  is the the direction of movement produced as a result of the calculations, this is a 2 dimensional vector denoting the  $x$  and  $y$  velocities relative to the environment.  $f_{lk}$  is the force applied by the target  $o_k$ ,  $g_{li}$  is the force applied by robot  $v_i$  and  $\omega_{lk}$  is a weighting that reduces the influence of the target  $o_k$ . Where there are  $n$  targets and  $M$  robots. Figure 2.4 shows an example in which two robots are providing the repelling forces ( $g$ ) and two targets providing the attractive force ( $f$ ), that are then combined to produce the resultant force (and therefore movement,  $\Delta$ ).

$$\Delta = \sum_{k=1}^n \omega_{lk} f_{lk} + \sum_{i=1, i \neq l}^M g_{li} \quad (2.6)$$

This heavily influences the work of this thesis by directly inspiring one of the algorithms trialed. Also the general philosophy of producing computationally efficient algorithms through purely reactive strategies are used throughout this work. Despite directly inspiring one of the algorithms tested in this work, there are a number of modifications required. Firstly the simple model that any target within range is observed is removed as it is one of the assumptions that through this simplification the

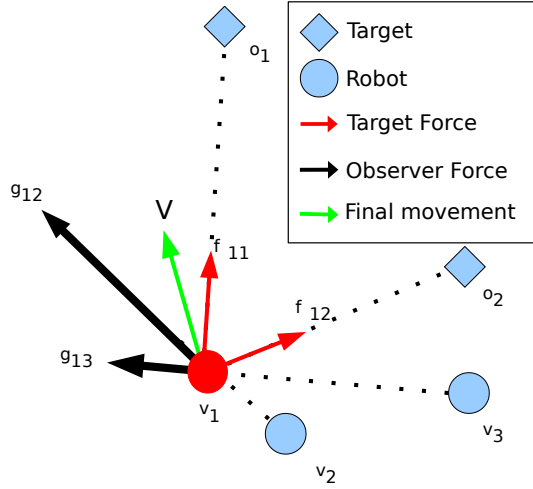


Figure 2.4: A-CMOMMT Forces.

algorithm may not be suitable for environments with complex structures of obstacles. Secondly the simplicity of a pure vector field approach using only the robot and target forces is inadequate due to the fact that the robot must remain constrained to the road network. A road network will have many local minima that are likely to drastically reduce the robots performance while using a simple vector field approach, as shown in Figure 2.5, the final trajectory produced will pull the robot towards a dead end as well as off of the road network. The local minima is formed by restrictions of the dead end, this will then trap the robot in the location causing it to be of little use. Consequently, a prior knowledge of the road network would need to be incorporated, so that a route can be planned towards specific objects and avoid such minima.

Further details of the solution produced using A-CMOMMT is provided in section 4.1.1 where the algorithm is extended and modified for use in this work.

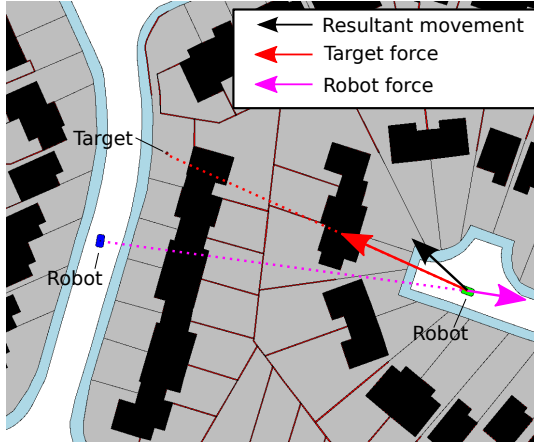


Figure 2.5: Local minima restrictions of A-CMOMMT. Ordnance Survey© Crown Copyright. All rights reserved.

### 2.3.2 Shortest Escape Path

The A-CMOMMT method although simple takes no account of occlusion. An alternative method that is also very relevant is the approach of (Lee et al., 2002; Gonzalez-Banos, 2002). This addresses the problem of moving the robot in order to maintain visibility of a target taking into account the current field of view. This obviously increases the complexity of the algorithm over the simple vector field approach. This was designed for the single robot single target scenario however the algorithm is applicable to the MRST environment. It is based upon a concept called the escape path tree.

The algorithm uses a range scan similar to that produced by a laser scanner (such as the SICK LMS range or the Hokuyo URG-04LX Laser Sensor). It identifies the points at which the robot can escape the field of view and then finds the shortest path the target can take to reach each escape point. An example of a target and the escape paths generated

from the robots' laser scan is shown in Figure 2.6. As shown the shortest paths consist of a series of straight lines between either pivot nodes or escape points, escape points are points at which the target can leave the current field of view. Free edges labelled on the diagram are edges in the field of view through which a target can escape. Pivot nodes are centred on the corners of obstacles that the target must circumvent in order to reach escape points the other side of the object. These escape points, pivot nodes and paths form the Escape Path Tree (EPT) that defines all the possibilities for escape. The tree is formed with the target as the root node, each node of the tree is a pivot and each branch terminates at a leaf node that is the escape point, the vertices's connecting these nodes of the tree then form the escape path. The EPT is illustrated in Figure 2.7 that shows the tree formed from the situation shown in Figure 2.6. This tree can be formed efficiently by iterating over each ray of the laser scan once (Gonzalez-Banos, 2002), generating the tree is therefore very efficient and scales only relative to the resolution of the laser scanning.

To each escape path that originates from the target there is an associated risk that defines the danger of the target escaping via that path, longer paths are naturally lower risk since the robot has longer to modify its field of view to compensate. A great number of distant escape points could however overshadow the importance of a single more immediate threat. To prevent this from happening the threat for a given pivot node or escape point is calculated by taking the average risk of its child nodes. The risk for a given escape point( $\varphi_e$ ) is detailed in equation 2.7.

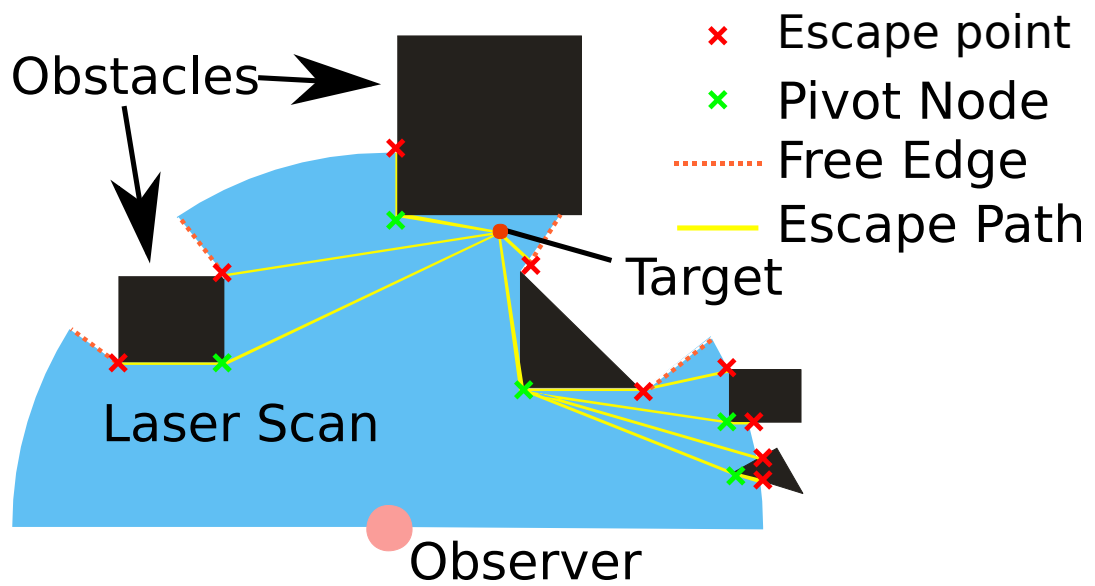


Figure 2.6: Escape path tree example

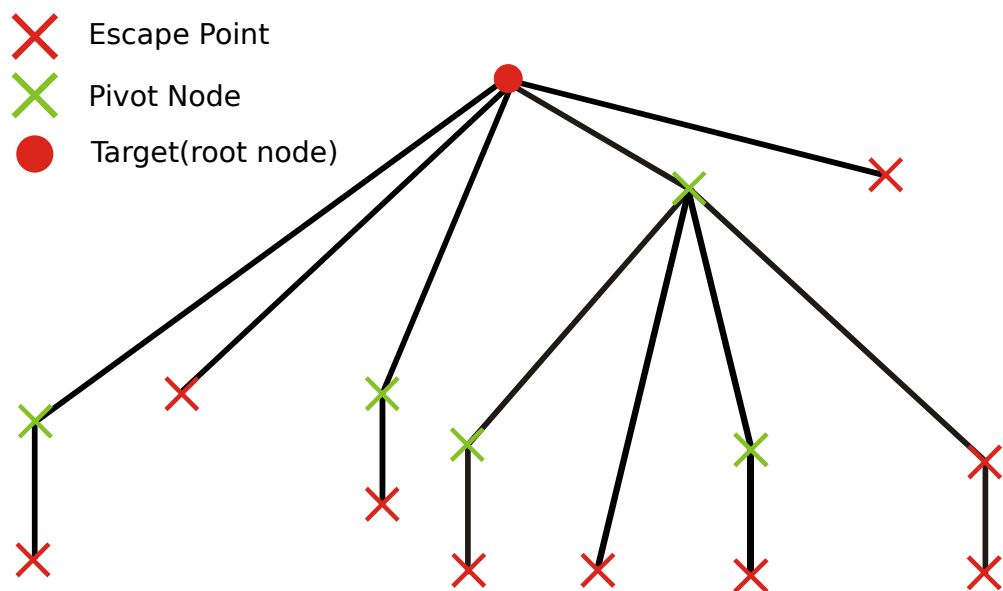


Figure 2.7: Escape path tree for Figure 2.6

$$\varphi_e = cr^2 \left( \frac{1}{h} \right)^{m+2} \quad (2.7)$$

$c$	Scaling factor
$r$	Distance from escape point to robot
$h$	Length of escape path
$m$	Look-ahead component

*Equation based upon work in (Gonzalez-Banos, 2002)*

From the function 2.7 as the length of the escape path increases the overall risk decreases, however as the distance from the robot to the escape point increases the risk increases. The reason for the latter is shown in Figure 2.8. This features two possible examples of a robot observing a target, in the left example the occlusion vertex is closer to the target, and in the right the occlusion vertex is closer to the robot. The left example in which the vertex is closer to the target requires a much greater movement in order to keep the target in view compared to the example on the right hence the risk increases with the distance to the occlusion vertex.  $m$  is a component that can be tuned to define how reactive or proactive the movement is, its minimisation causing the robot to be highly proactive.

The robots reaction to a given escape point (derived in (Gonzalez-Banos, 2002)) is determined by the following vector:

$$-\Delta\phi^e = \begin{bmatrix} 2cr \left( \frac{1}{h} \right)^2 \Delta r \\ -2c\delta r \left( \frac{1}{h} \right)^3 \Delta h \end{bmatrix}$$



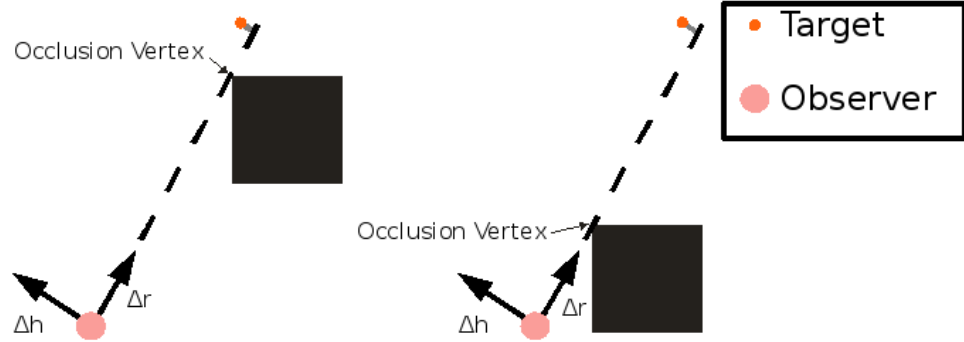


Figure 2.8: Example of how the distance from robot to the Occlusion Vertex affects the robots movement

$\delta$                       The radius of the target.

This is based upon a coordinate system  $(\Delta r, \Delta h)$  which is centred on the robot with the  $\Delta r$  axis in the direction of the given escape point and  $\Delta h$  perpendicular to it. At times of low risk  $h$  is large,  $\Delta r$  will dominate in this situation due to being inversely proportional to  $h^2$  compared to  $h^3$  for the  $\Delta h$  component. The robot will therefore approach the target positioning itself better for the future. Alternatively at times of high risk the  $\Delta h$  component will dominate and the robot will move perpendicular to the direction of the occlusion, hence making an emergency move merely to keep the target in view for the immediate future.

## 2.4 Searching

The algorithms that have been detailed so far are concerned with tracking a target. However the behaviour while a target is not being observed will also impact upon the system's performance. Although the goal of

this work is to develop a system that maximises the time in which a target is observed, it is inevitable that a target will escape observation and a strategy will be required to reacquire it. A number of possibilities arise for dealing with this. Through the simulation environment it would be possible to locate a target even when it is out of view, this would allow focus upon the tracking algorithms and not the search algorithm. This is however an unrealistic assumption that a target's precise location can be obtained despite being out of view.

The optimal solution would be to perform a complete sweep, covering the environment in a systematic manner in order to ensure detection such as the pursuit-evasion problem and the museum problem (Gerkey et al., 2006; Guibas et al., 1999). However these algorithms rely upon the ability to cleanse areas and prevent re-contamination by moving in a manner which would prevent the target from slipping past and re-entering. This relies upon the ability of the robot to traverse the same space as the target. As can be seen in Figure 2.9, a robot with restricted movement can easily be prevented from entirely cleansing an area due to being unable to view the entire space. Additionally, as stated in (Gerkey et al., 2006), a solution taking into account multiple robots increases the complexity exponentially and thus makes such a prospect impractical. Consequently an approximate function must be found.

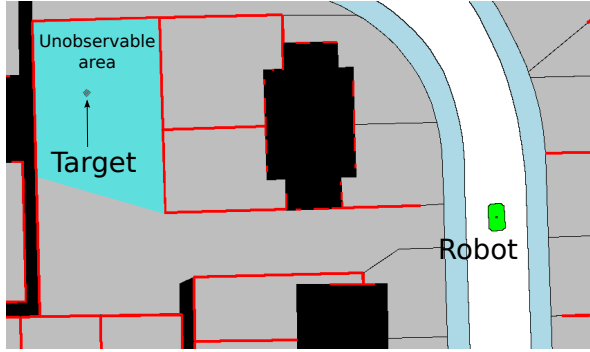


Figure 2.9: Example of an impossible search for a restricted robot due to there being no valid position in which the robot can see the target. Ordnance Survey© Crown Copyright. All rights reserved.

### 2.4.1 Jung

Jung provides effective methods for the tracking of multiple targets in environments that involve environment occlusions (Jung, 2002, 2005). This presents a framework for distributing robots across an environment for the collaborative sensing of multiple targets. Two algorithms are provided, one for use in a continuous, analogue environment and the second for a topological map (i.e. a collection of interconnected nodes). The analogue version works by convolving probability distributions over the environment using the positions of the robots and targets. This produces an urgency map indicating the areas that are of highest urgency and thus in need of inspection. The topological version works in a similar manner calculating urgencies at a given node based upon the time since an observation was made and the size of the region. The topological algorithms will be used in this work due to the road map being a topological structure. This will be the space traversed by the robots. The equations used in the topological algorithm are shown in equations 2.8-2.12.

$D_r(R)$	a simple estimate of the density of robots in region $R$
$D_t(R)$	an estimate of the target density in region $R$
$\theta_d$	a tunable parameter that details how willing a robot is to travel long distances
$\alpha$	This roughly estimates the intrinsic importance of an area, valuing places with a large area as more valuable
$d$	Distance from robot to region
$d_{avg}$	Average distance between regions

$$D_r(R) = \frac{\text{the number of robots in region } R}{\frac{\text{area of region } R}{\text{unit coverage}}} \quad (2.8)$$

$$\alpha = \frac{\text{area of region } R}{\text{unit coverage}} \times \theta_d \quad (2.9)$$

$$D_t(R) = \begin{cases} -1 & D_t(R) = 0 \& D_r(R) \neq 0 \\ D_t(R) + \alpha & D_t(R) < 0 \& D_r(R) = 0 \\ \frac{\text{the number of targets in region } R}{\frac{\text{area of region } R}{\text{unit coverage}}} & otherwise \end{cases} \quad (2.10)$$

$$u(D_r, D_t) = \begin{cases} \frac{D_t}{D_r} & D_r \neq 0 \\ D_t \times \alpha & D_r = 0 \& D_t \neq 0 \\ 1.0 & D_r = D_t = 0 \end{cases} \quad (2.11)$$

$$U(R) = u(D_t(R), D_r(R)) \times \frac{d_{avg}}{d} \quad (2.12)$$

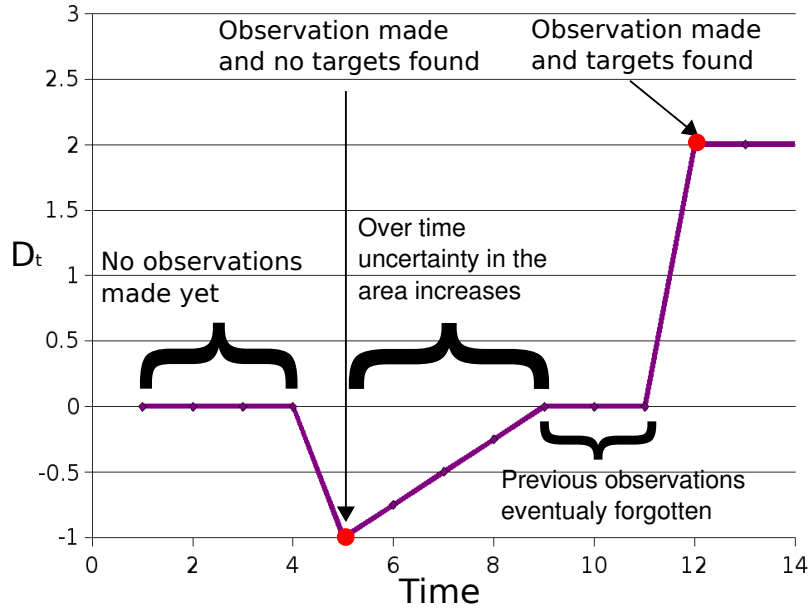


Figure 2.10: Example of how the function  $D_t$  will act for the scenario given.

$D_t(R)$  is designed upon the concept that if a robot is not present it gradually increases the target density to indicate the level of uncertainty about how many targets are in that region, as shown in the centre room in Figure 2.11. If an observation has been made and no targets observed then it marks the area clear by reducing its value to a low value of  $-1$  (as shown in the right most room in Figure 2.11). Otherwise targets are being observed in this area therefore set it to a similar density estimate as used for  $D_r$  (left of Figure 2.11).

Figure 2.10 shows how  $D_t$  will react in a given situation. At the start the value stays at  $0$  indicating no observation has been made, an observation is then made in the area and no targets are found and is therefore reset to  $-1$ . The robot then leaves the area and gradually over time the uncertainty increases until the fact that no targets are observed is entirely

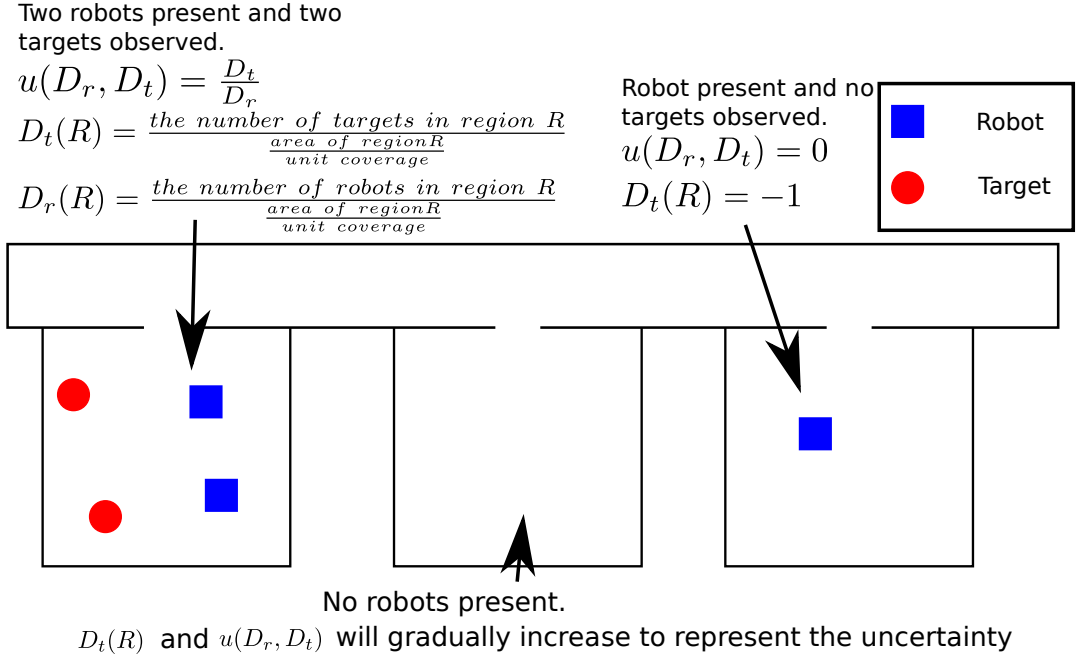


Figure 2.11: An example of a tracking situation and the corresponding values for the Jung equations for each area.

forgotten. Then a robot returns to make an observation, this time targets are found and the value is set to an appropriate value.

The core of the urgency calculation is then performed by  $u$ . This uses the concept that if a robot is present making an observation then the urgency is proportional to the density of targets and inversely proportional to the density of robots. As shown in Figure 2.11, the left and right rooms have an urgency relative to the number of robots, for the room with two robots it is set to  $\frac{D_r}{D_t}$  and 0 for the room with no targets (due to  $D_r$  being 0). If however no robots are present to make an observation and the target density is unknown (either no observation has been made or it has been sufficiently long since an observation has been made) then the urgency is set to a high value of 1.0 to attract robots to make an obser-

vation. Otherwise there are no robots present to make an observation, however a previous target density is known, the urgency is therefore set proportional to the target density. Due to the target density gradually increasing the urgency will also naturally increase.

In  $U(R)$  the final urgency is then made inversely proportional to the distance a given robot will need to travel in order to make an observation (i.e. encourage a robot to check near by urgent regions) and then normalised using the average distance between nodes  $d_{avg}$ .

This produces a searching behaviour as robots move to areas of high uncertainty in order to make an observation, once an observation has been made the uncertainty and by association the overall urgency are reduced by the target density being reset to -1 (assuming no target is observed), thus making other areas of the map more urgent and attracting robots to make observations. Over time, cleared areas urgency gradually increase indicating that a target may have moved into that area and thus encouraging a robot to check the given area again.

Although this algorithm would work as it is with few modifications, it could be made more efficient by providing the searching behaviour with additional information such as initial estimates of target density based upon the last known position of the target as well as incorporating the Cartesian distance between topological nodes. Since targets are not restricted to using the road network the Cartesian distance is also an indicator of target density.

# Chapter 3

## Simulation Platform

Due to limitations in the current level of technology and funding with respect to building platforms that can reliably operate in an urban environment safely, it is not possible to test in a real environment. Consequently this work uses simulation as a basis for comparing the performance of the algorithms developed. A number of simplifications have been made in the construction of the simulation due to limited computational resources and data, however these have been kept to a minimum. These simplifications will be detailed in the subsequent chapters as well as in the Further Work section (section 6.3).

In this chapter details are provided of the simulation platform developed in order to test an algorithms performance in an urban environment. Firstly a definition of the environment is given. This describes the environment that is to be implemented in the simulation platform.

Secondly the map data that was used in the trails is discussed and the



modifications required in order to make it suitable for use in the simulation.

This is followed by a description of the implementation of the simulation platform. The implementations of the robots and target are then discussed, including the types of sensors and physical dimensions. Finally the control architecture used to control the robots is shown.

### **3.1 Definition of the Environment**

The simpler version of the problem was formally defined in section 2.2 and deals with the case of an environment that is relatively simple and with only convex obstacles within the environment.

The goal of the exercise is to maximise the metric  $A$ , which is the ratio of frames in which the target was observed. In the environment we have three types of spaces. Firstly a space that both the target and robot can traverse ( $J$ ), those that the target alone can traverse ( $I$ ) and one that neither can traverse ( $H$ ). These are all enclosed within the overall environment space ( $S$ ).

Then a team of robots ( $V$ ) is placed within the environment and a single target ( $o$ ).

As before the equations 3.1-3.3 define whether or not the target is under observation.

The environment provided in section 2.2 is therefore reformulated as follows to describe the problem at hand:

$t$	Time.
$T$	Time period for a given simulation run.
$A$	The ratio of frames in which the target was observed.
$H$	A two dimensional space that can not be traversed by any agent and occludes all sensors, entirely enclosed within $S$ .
$I$	A two dimensional space that cannot be easily traversed by robots but can be traversed by the target, entirely enclosed within $S$ .
$J$	Free space traversable by all agents, entirely enclosed within $S$ .
$S$	A two-dimensional, bounded spatial region containing spaces $H$ , $I$ and $J$ .
$V$	A team of $m$ robot vehicles, $v_i, i = 1, 2 \dots m$ , with $360^\circ$ field of view observation sensors that are of limited range and restricted to traversing space $J$ freely.
$o(t)$	A target that is located within region $S$ , but not within $H$ at time $t$ .

$$B(t) = [b_i(t)]_m \quad (3.1)$$

$$b_i(t) = \begin{cases} 1 & \text{if robot } v_i \text{ is observing target } o(t) \text{ in } S \text{ at time } t \\ 0 & \text{otherwise} \end{cases} \quad (3.2)$$

$$g(B(t)) = \begin{cases} 1 & \text{if there exists an } i \text{ such that } b_i(t) = 1 \\ 0 & \text{Otherwise} \end{cases} \quad (3.3)$$

$$A = \sum_{t=1}^T \frac{g(B(t))}{T} \quad (3.4)$$

Extending the diagram of the simpler environment we produce the space as shown on the left of Figure 3.1. As shown the entire environment is split into three spaces ( $H$ ,  $I$  and  $J$ ) that represent the various spaces that the target and robots can traverse. Generally the space  $H$  (buildings and walls) in real environments is found enclosed by  $I$  (land and paths) and in turn the space  $I$  could be seen to be encompassed by  $J$ , as shown on the right of Figure 3.1. There is however no requirement of this, merely a common occurrence and therefore the spaces are said to merely reside within the enclosed region  $S$  and not within each other.

## 3.2 Maps

The contents of the environment have been chosen to emulate real urban environments in order to make the test close to a realistic scenario. The map data is provided by the OSMasterMap® data set from Ordnance

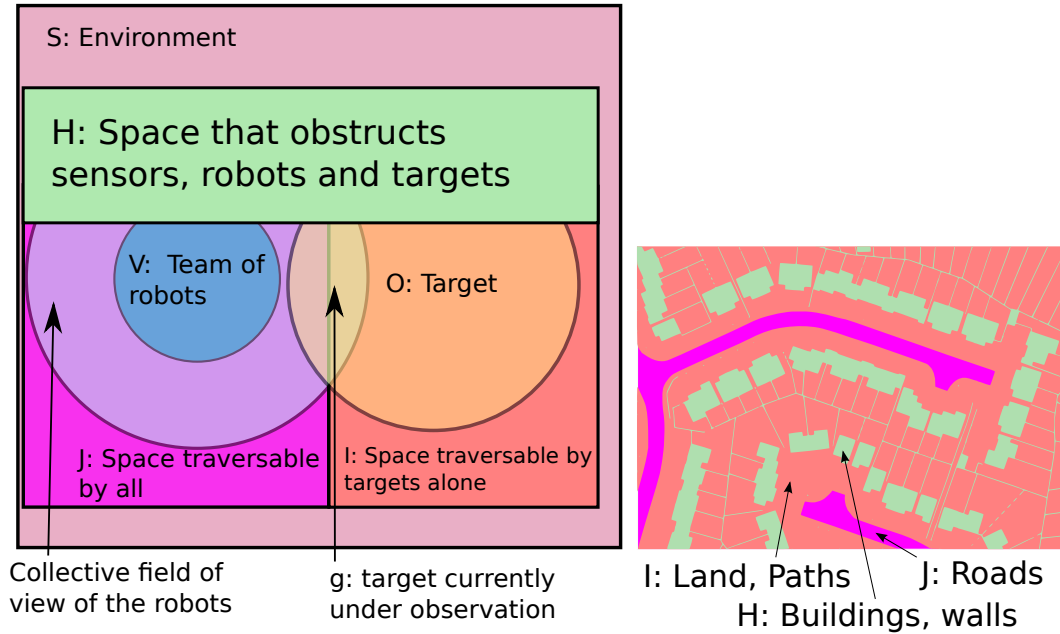


Figure 3.1: Left: Illustration of the environment space for the given problem. Right: Space demonstrated on a map. Ordnance Survey© Crown Copyright. All rights reserved.

Survey© (Figure 3.2). The data is typically accurate to one metre and contains features larger than a few metres. A notable limitation however is that it does not contain information about visually transparent obstacles such as chain link fencing or iron railings. Due to this lack of information, these obstacles are presumed to be opaque to all sensors.

Shapes within the environment are defined as a series of  $x$  and  $y$  coordinates that produce a polygon defining the perimeter of the shape. In the case of obstacles with minimal width such as walls and fences there is merely a line that defines the path of the object and not a closed polygon.

A prior road map is made available to the robots in the form of a series of road nodes as shown in Figure 3.3, each node being displayed as a red square. These nodes are spaced differently depending on the configura-

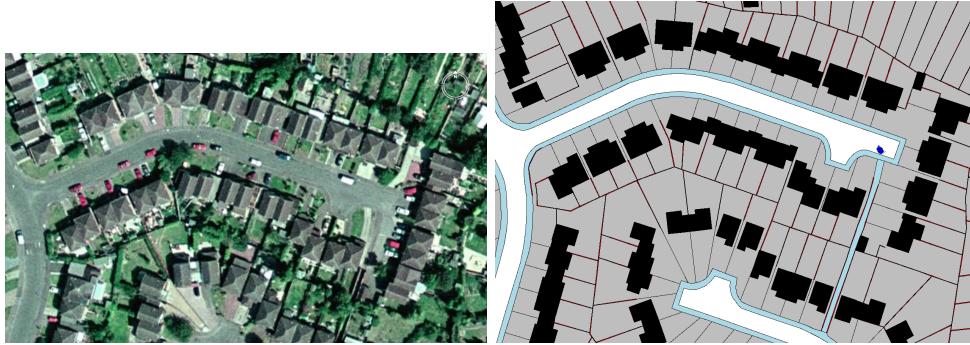


Figure 3.2: Left: Aerial photograph of real environment. Right: Simulated view of environment. Ordnance Survey© Crown Copyright. All rights reserved.

tion of the road at that point. Obviously more nodes are used to represent a curve than a straight road. Looking at one of the maps used in the trials (will later be referred to as Map 2) these are spaced with an average of  $17m$  between nodes. The smallest being  $0.15m$  and the largest roughly  $200m$ . Each node comprises of:

- A unique reference number.
- The location of the node in a global coordinate frame.
- The enclosing area that defines the geometry of the road around the node which allows the robot to detect when it is in danger of leaving the road network.
- Which other road nodes are immediately adjacent, provided for path planning purposes.

An example of the data is shown in Appendix C.

Other than the geometry of the roads, no prior information is made available to the robots as to the geographical layout or obstacles.



Figure 3.3: Node positioning on the road network. Ordnance Survey© Crown Copyright. All rights reserved.

Two types of map obstacles are present within the environment: hard and soft. Hard obstacles (denoted by  $H$  in the problem definition) are features such as walls, fences and buildings; no part of either the robot or the target can intersect with these obstacles. Hard obstacles also block all sensors and thus restrict the field of view of all agents as well as providing sensory information about the local environment. Soft obstacles (denoted by  $I$ ) are areas such as paths and grass. These offer no resistance to the target or the sensors provided to the target for the purposes of obstacle avoidance, this allows the target to freely traverse these areas. For the robot however these areas will block the sensors used for navigation and obstacle avoidance, thus restricting the robot to the road network. Sensors that are used for target detection however will be allowed to pass, allowing the robot to detect a target that is within a soft obstacle. Collision by a robot with a soft obstacle will not result in an absolute stop in movement however by activating the collision sensors present on the robot will force the robot entirely on to the road.

### **3.2.1 Map Data Pre-processing**

In order to import the maps into the simulations they have to be converted into an appropriate format. This was done by pre-processing the files, then supplying these to the simulation at run time. It would be possible to process the files at run time. However the map files are extremely large and almost all of the machines performing the simulations are unable to process the raw maps due to their limited memory.

The objectives of the pre-processing are to:

- Convert the files into an easily importable and compact format.
- Translate the map to be centred at  $[0, 0]$  within the environment.
- Remove elements not within the test area.
- Remove problematic structures.

The translation step is performed on all items that are parsed, there is however no scaling of the map performed. Removing the elements not within the test environment was also performed on all types of physical obstacle. An obstacle is defined as being within the environment if a single point of its perimeter falls within the test area.

### **3.2.2 Physical Layout**

The pre-processing involves a number of steps. Firstly the buildings, walls and fences are extracted. Buildings are identified within the raw

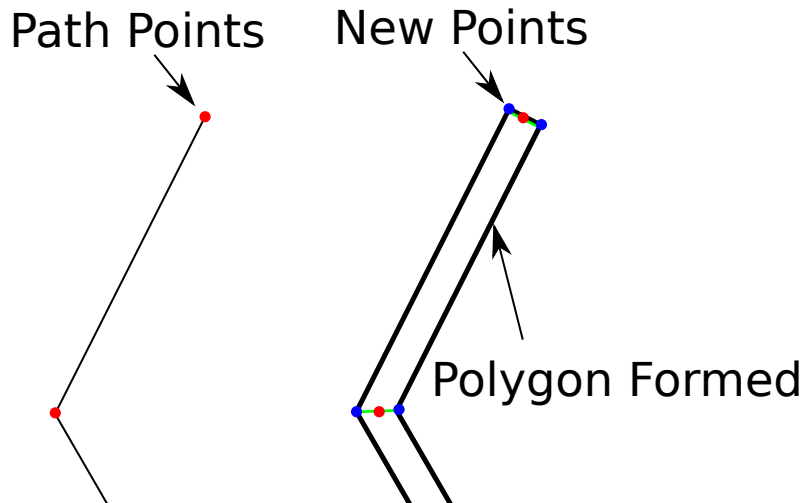


Figure 3.4: Illustration of adding width to walls. Original path on the left, polygon formed on the right

map data as those containing the theme “Buildings” or “Structures”. Other obstructing features (generally walls and fences) are identified as those with the theme “Land” and physicalPresence “Obstructing”. Ordnance Survey define obstructing as “*indicates that the feature prevents pedestrian access*” (MasterMap, 2008).

Walls and fences are represented as a single line. The simulation environment however requires an object with some degree of width therefore these had to be widened to give some form of substance. This is done by creating 2 new points to replace each existing one, these are placed by extending out  $10cm$  in each direction perpendicular to the angle of the turn. This forms a polygon around the line as shown in Figure 3.4. As shown, a polygon is formed by the perimeter of these points and thus an object with a width of  $20cm$ .

Land and paths are also extracted, identified as being within the theme



“Roads Tracks And Paths” and descriptive group “Roadside”, or being within the theme “Land”.

As previously stated, problematic structures were removed. Due to the simulation being 2 dimensional, certain structures cause problems, such as foot bridges which cross the road. Were they not removed they would appear as obstacles to the robots and prevent the robots from passing them. These were identified by detecting structures whose body would cross between two road nodes, and thus block the road.

Finally an enclosing box was placed around the perimeter of the environment to prevent either the robots or target from leaving the test area.

### **3.2.3 Road Network**

The road nodes are parsed to form a graph like structure that can then be loaded into each simulation. Nodes are identified by elements called RoadLink within the raw map data, these define roads as a sequence of  $x, y$  coordinates, each of these coordinates is defined as a node. The graph is then formed as these nodes are linked to adjacent nodes on the road network. The information extracted from the nodes is detailed in section 3.2 on page 38.

In addition to extracting this data from the raw files, the nodes coordinates were translated by the same offset as used for the physical objects. Nodes outside the test area were also removed from the graph.

## 3.3 Simulation Implementation

### 3.3.1 Basis

The simulation environment was developed using the simulation environment Player/Stage (Player, 2008). This is an open source simulation platform that can simulate a wide range of 2 dimensional environments. The project is split into two components Player and Stage. Player is a generic robot control platform that can take instructions from a control system and feed the sensor data back. The purpose of this is to provide an abstraction layer which allows you to program algorithms relatively independently of the physical hardware used, i.e. you can use various brands of laser or GPS system interchangeably without modifying the higher level code. Player acts as an interface between the high level control and the hardware. This can also be used irrespective of whether a physical robot or a simulation is used.

Stage provides a simulator that interfaces with Player to provide the full simulation platform. This implements many common sensors such as laser range sensors, sonars and cameras. Parameters can be set for each sensor such as its range and accuracy. The sensors used in this work are laser sensors, rangers (a sonar style range finder), position (emulating a GPS system), bump sensors and fiducial (emulating a target identification system). These are discussed in Sections 3.4 and 3.5 in which the target and robots sensors configurations are shown.

Stage works by executing the simulation at a fixed rate irrespective of the

speed of any control loops. This is an accurate representation of reality in that real robotic systems or environment will not pause to allow the control loop to finish its execution. This requires that the control algorithms execute in real time at a sufficient rate to maintain control of the robot.

In this work the Player and Stage versions used were 2.1 and 2.0.2 respectively. They are both developed on C++ and runs on POSIX operating systems, in the case of this work it was run on the Linux operating system.

Communication between Player and the control algorithms are performed over TCP/IP. The algorithms can therefore be executed remotely to the simulation provided there is a network link between the control platform and simulation platform. It also means that any language can be used for programming the control algorithms. Java was used for the development of all of the control algorithms. This utilises the JavaClient libraries (Javaclient, 2008) that provide a simple interface, to the TCP/IP protocol for communication with Player.

### **3.3.2 Collision Detection**

The stage simulator uses a occupancy grid for its underlying representation of the obstacles within the environment. Ray tracing is then used to detect collisions between dynamic objects and objects within the map. This is a very efficient solution in terms of computational speed however is not particularly efficient in terms if memory usage. A trade off exists

between the computational complexity and memory usage verses the accuracy of the collision detection, a finer grid will use more memory and computational power. Ultimately a resolution of the underlying model has to be chosen and was fixed at 60cm. This was chosen as the highest resolution possible with the memory available to the computers that were running the simulations. This was decided upon due to memory being experimentally found to be the limiting factor on the machines performing the simulations. This however provided sufficient accuracy for the robots to operate in a realistic manner and receive information through these sensors that are an accurate representation of the original data.

### **3.3.3 Modifications**

A number of modifications were made to make the simulator suitable to be used in this work.

Firstly some of the logging features of the environment were not present in the simulation in its original form, although logging of the positions and orientations of each robot existed, logging abilities for the bumper and fiducial sensors had to be added. The fiducial sensors logging was of particular importance. Due to being the sensor used for target identification, it was essential to log this information so that post simulation processing could calculate at what times the target was under observation.

Additionally a layered collision model was used in which all elements of the environment are assigned integer “heights”. Objects can then collide

with obstacles at a higher height. This was used to enforce the environment's spaces as described in Section 3.1. The buildings and walls (space  $H$ ) is the highest, followed by the target, robots then land (space  $I$ ). The free space ( $J$ ) is then simply left empty. As previously stated collision between land and a robot does not result in a collision (i.e. induce a total stop in movement). However by activating the robot's collision detection sensors, cause the robots control system to move the robot back on to the road. The sensors are therefore set so that the bumpers and sonars are activated despite the fact that land is at a lower level. The laser sensors and fiducial sensors are however set so that they are not blocked by land. In the case of the laser sensor this was achieved by setting the existing configuration property `laser_return` to zero. In the case of the fiducial sensor its ray tracing method was modified to ignore land and paths. Thus robots, targets and buildings will block the field of view of lasers and fiducial sensors. A target and robot and their permissible spaces in the simulated environment are depicted in Figure 3.5.

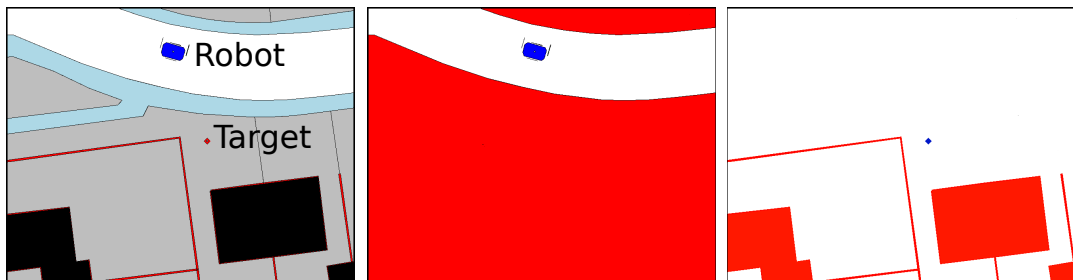


Figure 3.5: Left: Robot and target in simulated environment. Red denotes space that cannot be traversed. Centre: Robots permissible space. Right: Targets permissible space. Ordnance Survey© Crown Copyright. All rights reserved.

A communications system was also developed as an extension to the

JavaClient libraries. This was an extension of the opaque interface that already exists within the JavaClient. The opaque interface takes data transmitted to it and forwards it on to all others subscribed to the interface. This emulates a WiFi network between the robots through which data can be passed. This is mainly used to pass the current location and destination node of the robots to the rest of the team.

The map data provided by OSMasterMap® is in a vector format (Appendix C) where shape geometries are provided as a series of points that form the shape of the object. The general method of inputting maps into Stage is using a bit-mapped image with black and white pixels denoting filled and empty space. With such a large environment this would require two (one for each type of obstacle) particularly large bitmaps to be supplied with the simulation. An ability to directly import polygons from a file however was produced. This allows the maps to be stored in a relatively compact format (a series of points defining the outlines of each shape) and with little pre-processing from its MasterMap form. Once the polygons have been imported the occupancy grid is formed. This also meant that the resolution of the underlying occupancy grid could be altered without adjusting the map. In an environment using a bitmap however, the map's accuracy can only be as high as the resolution of the bitmap provided.

### 3.3.4 Cluster

The experiments were executed on a cluster of roughly 100 nodes, each node possessing a single core of a quad-core Q6600 processor and 512 megabytes of memory and running a Linux operating system. The Condor clustering system was used to manage the cluster and to submit jobs (Condor, 2009). The Condor system allows the execution of batch jobs submitted to the cluster, once a job is submitted Condor waits for a computer to become available (i.e. not in physical use by an operator, and not already executing a job) that fits the specified requirements. The requirements generally specify the operating system, processor and memory requirements. Once a machine is found the job is then uploaded, executed and when done any files left over are copied back to the machine that submitted the job. In this work the simulation logs were the only files returned from the job.

Due to the execution nodes of the cluster not having a windowing system it was also required to remove Player/Stages reliance on having a visual interface. This also reduced processing power as time was not wasted drawing the visual feedback.

## 3.4 Target Architecture

The target is an agent with omni-directional movement who has a  $0.7 \times 0.7m$  footprint. For the purposes of obstacle avoidance, the target has a laser range finder as shown in Figure 3.6. Its maximum speed is varied

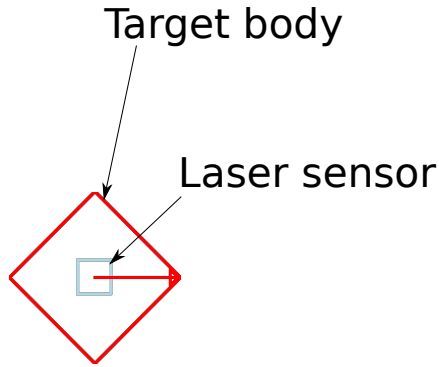


Figure 3.6: Illustration of the target used in the simulation

relative to the top speed of the robots. The target's absolute speed ranged from  $1.25m/s$  to  $4.0m/s$ . The target moves using a vector field pushing away from the previous locations of the target, forcing it to move to new areas of the map. Obstacle avoidance is also incorporated into the vector field in order to prevent the target crashing into hard obstacles.

### 3.5 Robot Architecture

The robot is a  $2.5 \times 1.5m$  differential drive robot that is provided with:

- $360^\circ$  Laser range finder that can detect the distance to “hard” obstacles up to  $100m$  away.
- Sonar array that can detect both hard and soft obstacles up to  $5m$  away, primarily used for obstacle avoidance.
- Bumper sensor for the purposes of detecting collisions.
- Target identification system. This is approximate to a suitable camera system that would be able to identify a human (Siebel, 2003).



This can identify the location of the target if it is within the field of view. A “fiducial sensor” within player/stage is used to approximate this system.

- Communications channel with which it can communicate to all other robots.
- Global positioning system.

These sensors were chosen firstly as they provide the required sensing capabilities to achieve the task at hand, but also as sensors commonly available and used in mobile robotics (with the exception of the target identification system).

Lasers are common in mobile robotics. Their functionality is generally to provide obstacle avoidance and mapping of the environment (Hahnel et al., 2003). Their advantage over sensors such as sonars are firstly the range at which they can operate. The commonly used LMS-200 operates at up to  $80m$  (SICK AG, 2006), however, relatively long range sonars only reach about  $10m$  (MaxBotix, 2005; SensComp, 2003). Additionally, the narrow focus of an individual measurement allows sensors to produce relatively high resolution cross sections of the near environment. Sonars by contrast have a relatively wide sensing area, only allowing for a coarse representation of the environment. The narrow focus of the sensing also means that cross talk between measurements and different sensors is less likely.

Sonars, despite in many respects inferior to lasers, are suitable to their task of obstacle avoidance as they need only a very low resolution and

relatively short range ability. They are also preferable over lasers due to being significantly less costly. These are commonly used in the ring formation to provide a low resolution cross section of the area, and in conjunction with algorithms such as VFH (Borenstein and Koren, 1991), used for obstacle avoidance.

GPS systems are also extremely common. Their advantage is providing an absolute position reference without any prior knowledge of the environment. Despite this advantage their performance suffers significantly from occlusion of the sky, and are generally unable to operate reliably indoors. Additionally they require time to warm up and can have relatively high noise and errors, typically a DGPS system can manage only 4 to 6 metre accuracy (Borenstein et al., 1997). More advanced systems such as rtk-gps (Meguro et al., 2005) or incorporating an IMU (Sukkarieh et al., 1999) can address these problems, however at a significantly higher price.

The target identification system is not defined as a specific sensor, however it is envisaged that it is likely to be a visual system due to vision being a common method of identifying humans at a distance in robotics. A number systems exists that identify people using image sensors (Siebel, 2003; Foresti et al., 2005; Haritaoglu et al., 2000), these tend to be focused towards static CCTV style systems however research into tracking from mobile platforms has also been investigated with some success (Wilhelm et al., 2004; Feyrer and Zell, 1999).

The layout of these sensors are shown in Figure 3.7. The bumpers are merely placed at each of the four sides to provide complete coverage and detection of collisions. The configuration of the sonars is then in a typical

sonar ring formation. This allows the robot to detect obstacles approaching from the front then to steer way and slowdown as needed. The laser, fiducial sensor and GPS system are then placed at the centre of the robot. The communications however have no physical presence, therefore have no physical position on the robots body. In the context of the player/stage simulation fiducial sensors “detect coded fiducials (markers) placed in the environment”(Gerkey et al., 2004), this has been used to approximate a vision system by adding a marker to the target that the fiducial sensor detects and locates. Traditionally fiducial markers refer to a mark placed within an image systems field of view as a point of reference, a fiducial sensor would then be a sensor that can identify this mark.

In the simulation perfect sensors are used, no artificial noise or false sensor readings are added. This is a simplification and would not be possible in a real environment, however building a system that is robust to imperfect sensors is not within the scope of this work. It is however a natural extension to include this in future work.

A number of constants have been chosen for the characteristics of the various sensors. Varying these values may well affect the performance of the robots. However the large number of parameters that would need to be varied would make testing impractical. These have therefore been fixed at reasonable values.

The laser range finder’s maximum range has been set at  $100m$ . The ability for the technology to work at such a range is easily feasible. High range systems such as surveying laser equipment are already capable of operating at  $2km$  (Zhang et al., 2008), and systems such as the 3DLS-K

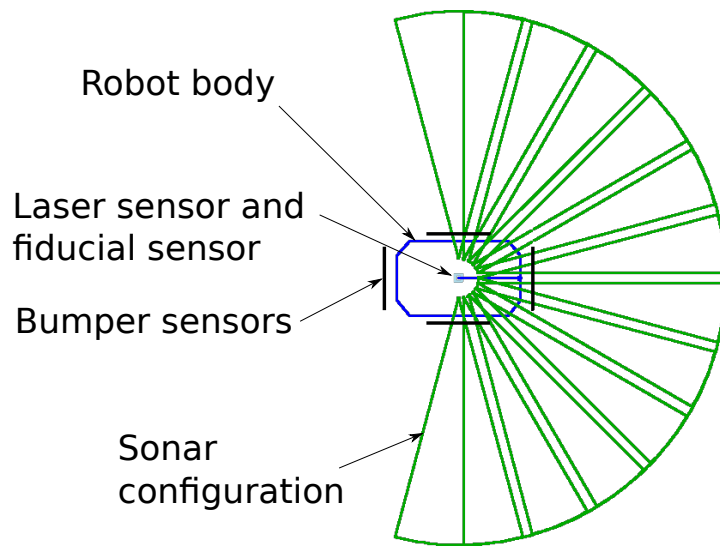


Figure 3.7: Illustration of the robots sensor configuration.

that is already in use on robotic platforms has a  $80m$  range (IAIS, 2008). It is therefore likely that  $100m$  range will be available in the near future. It is in fact probable that longer ranges will be available. However when scanning from a central point, as the distance of the scan increases the resolution will decrease. For instance, if the distance between scans is  $1^\circ$ , at a  $100m$  range the distance between scans will be  $1.75m$ . Thus the data gained at that range is likely to be fairly minimal and not of use to the algorithms. The radius of this target identification system was similarly set at  $100m$ . This was again viewed as an acceptable range that a camera could be expected to work at. In this situation, a larger radius could be of benefit. It is likely to be of fairly minor benefit due to the fact that in an urban environment line of sight visibility is rarely such a long distance.

The sonars were set at  $5m$ . This is a reasonable value to expect sonars to be capable of operating at due to products already existing that can

achieve this (MaxBotix, 2005). This sonar reads at up to  $20Hz$  and at a  $42KHz$  frequency. As these are being used purely for the purpose of obstacle avoidance they do not need to be of any greater range. This would also not affect the performance of the algorithms and therefore can remain fixed.

The bumper sensors are kept at a zero range, thus only activated upon contact with an object. GPS is assumed to be available throughout the environment. It is accepted that GPS is not always available, particularly in an “urban canyon”. However introducing such complexity of intermittent sensors is beyond the scope of this work. Similarly the communications channel is not affected by distance. Many communications technologies have limited range and this would again affect the performance of the system. However is not within the scope of this work.

The robot is a differential drive system allowing it to turn on the spot, its turning rate is limited to  $\pi rad/s$ , and its maximum speed is limited to  $4.0m/s$ . Infinite acceleration and deceleration are allowed by the simulation. This is also a simplifying assumption for the purposes of simplifying the robots control mechanism however is not a realistic representation of a robots true dynamics.

In this work only the robots speed is varied, another natural extension in future work would be to incorporate a more sophisticated model. For instance using a car like drive where the robot has a minimum turning radius and very limited turning rate. Additionally, to test how varying the various dynamics effects the robots ability to track the target.

## 3.6 Control Architecture

The flow of data through the system is shown in Figure 3.8. The algorithms are implemented in Java, utilising the JavaClient libraries (Javaclient, 2008). JavaClient sends the instructions generated by the algorithms over TCP/IP to the “Player Robot Interface” (Player, 2008) with a well defined message structure. These signals are then passed on to the Stage simulator who applies them to the current simulation. Stage then continually feeds back sensor information to the algorithms using the same path. The log of the sensor data is then extracted from the Player interface and stored for later analysis.

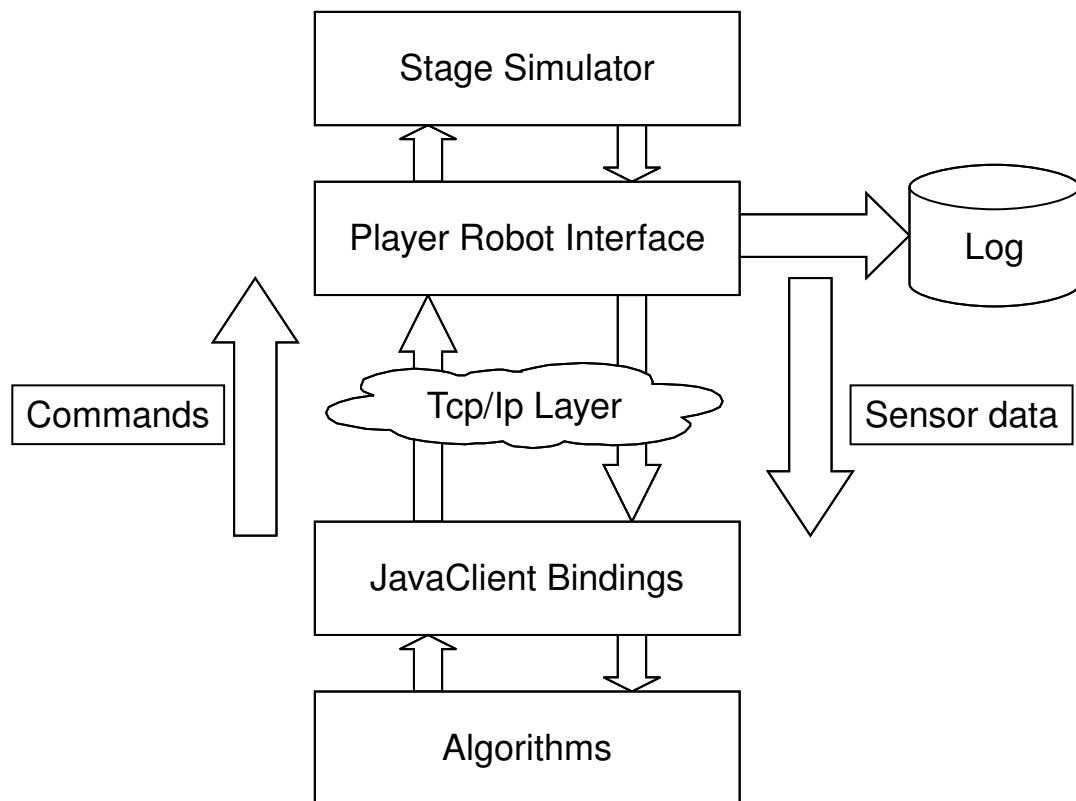


Figure 3.8: Data Flow Architecture

The structure of the system is detailed in Figure 3.9. The lower section of the diagram shows the modules directly involved with supporting the algorithms, such as providing them with sufficient data and control mechanisms to achieve their goals. The upper box contains the modules involved in the simulation. The “Java client bindings” in this figure is the same as referred to in Figure 3.8. The simulation module simply refers to the collection of interfaces and simulator detailed in the previous figure. Inside the “Road Interface”, the roads layout is provided as prior information in the form of a sequence of road nodes, as described in section 3.2. Shortest path searching algorithms are also provided to allow searching the graph formed by the road nodes for paths through the road network. A data storage area is provided for keeping up to date information about the state of the robot itself, as well as limited information about its peers. The following data is stored:

- Data about the robots peers (provided by inter-robot communications):
  - ✧ Robot locations.
  - ✧ Robots destination location (the position that they are currently heading for).
  - ✧ Location of a target if one is observed.

- Data provided by sensors:
  - ✧ Current position and orientation.
  - ✧ Laser scan.
  - ✧ Bumper contact.
  - ✧ Sonar readings.
  - ✧ Target location (assuming it is within the field of view).

The Inter-robot communications module takes the appropriate data from the Data Storage and shares it with the team via the JavaClient Interface. This simulates a WiFi style network where the data is broadcast by the robot to all others. Inter-robot communications are for information purposes only, there are no explicit commands for robots to instruct each other. The Sensor Data Interface merely takes the data fed back from the simulation and inserts it into the Data Storage in an appropriate form. Movement Command Interface takes the movement signals from the Algorithms, generally a signal to head in a given direction, and then sends the signal to the simulation to produce the required movement. This then uses data from the Data Storage about the robot's current state to achieve the correct speed and heading, as well as avoid obstacles.

This effectively forms a Layered control system (Brooks, 1986; Simmons et al., 1997), with the Algorithm at the top layer setting high level behaviours of searching and tracking and the Movement Command Interface at a lower layer, performing the lower level obstacle avoidance and movement towards a desired positions. With both layers gaining sensor data to help in their task via the Data Storage. The layers execute asyn-



chronously to each other and the sensors.

It also bares similarity to a hybrid control system (Low et al., 2002; Connell, 1992), with the Algorithm performing the deliberative long term planning and the Movement Command Interface performing the reactive short term actions.

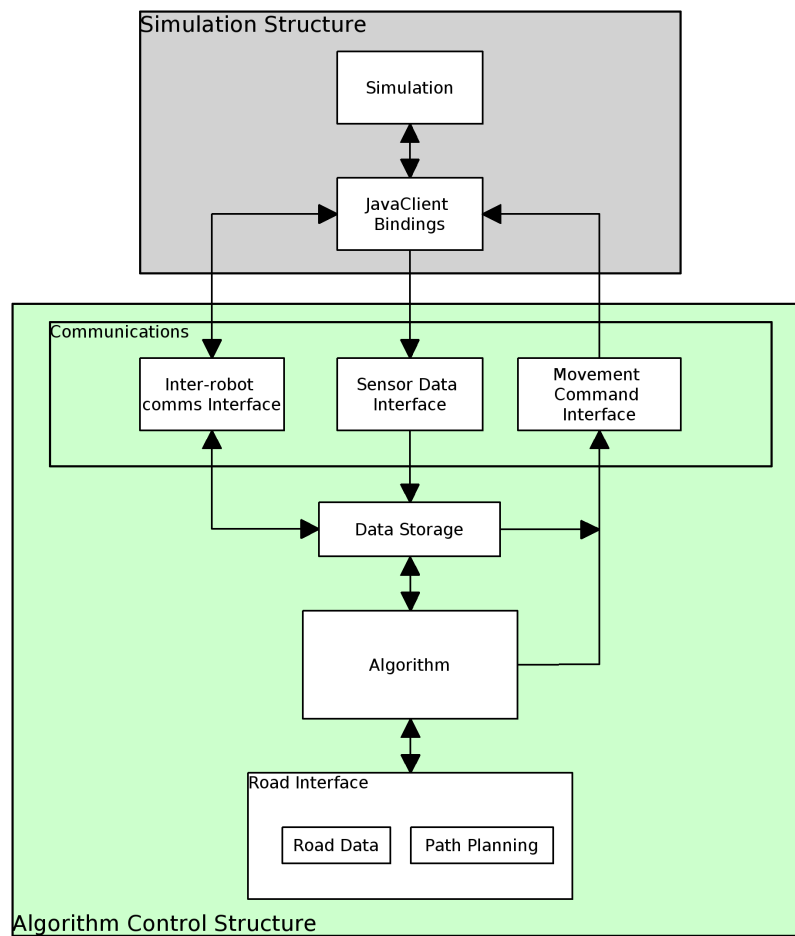


Figure 3.9: Control Program layout

# Chapter 4

## Tracking Algorithms

To address this problem three distinct algorithms have been developed. The first is an extension of the two approaches A-CMOMMT and the Jung approach as discussed in the literature review, this approach is named the Combined Urban Tracker (CUT). The second is a custom made algorithm called the Short Cut Path (SCP) algorithm, this also incorporates the Shortest Escape Path algorithm discussed in section 2.3.2. A third algorithm is then created that is based on the concept of positioning the robots at key positions within the road network. Finally a no movement strategy is also tested as a baseline comparison. These four algorithms are to be individually tested and their performance compared against each other. The results of this testing are shown in the following chapter.

Throughout this chapter a number of symbols are used, these are summarised in Appendix D.

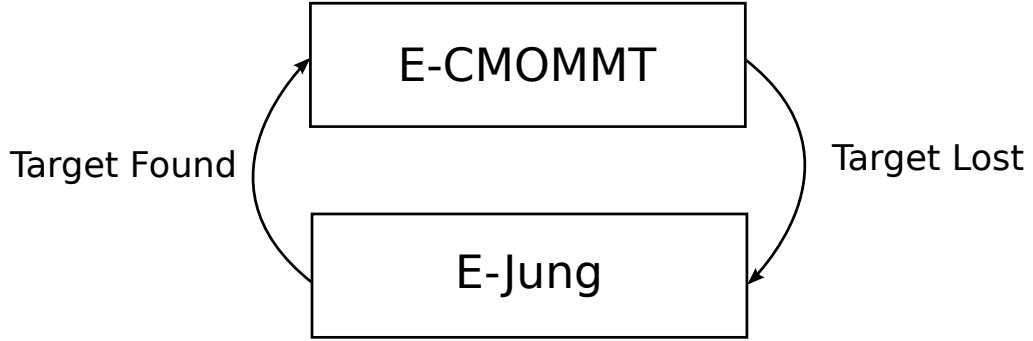


Figure 4.1: Switching behaviour of the Jung/A-CMOMMT algorithms

## 4.1 Combined Urban Tracker

The Jung and A-CMOMMT algorithms detailed in Sections 2.4.1 and 2.3.1 provide the complementary functions of searching and tracking. These functions are being combined to produce a complete tracking algorithm CUT (Combined Urban Tracker). Were the algorithms to be taken verbatim, they would be unsuitable for this environment due to a lack of understanding of the road network and the concept that there are areas that a target can traverse but the robot cannot. Both the A-CMOMMT and Jung algorithms are therefore modified to create the CUT algorithm, these are referred to as E-CMOMMT and E-Jung due to being extensions of the algorithms.

While the target is currently being observed, a modified version of the E-CMOMMT algorithm is used for the whole team. Then when lost a E-Jung style system performs the searching to re-acquire the target (Figure 4.1).

### 4.1.1 E-CMOMMT

$$V = \sum_{k=1}^n \omega_{lk} f_{lk} + \sum_{i=1, i \neq l}^M \psi_{li} g_{li} \quad (4.1)$$

The standard A-CMOMMT algorithm (as described in section 2.3.1) is designed to cope with multiple targets and is as shown in equation 4.1, where  $V$  is the movement vector that the robot takes,  $f_{lk}$  is the force that acts upon the robot produced by the target  $o_k$ ,  $g_{li}$  is the force produced by the robot  $v_i$ .  $\omega_{lk}$  and  $\psi_{li}$  are weighting functions that modify the influence of each target and robot depending upon their distance from the robot. The weighting function  $\psi_{li}$  was not included in the work cited in the literature review, however was included in earlier work on the A-CMOMMT problem (Parker, 1999), it is reintroduced here as it allows for a greater degree of control over the robots. The movement vector is generated by summing the weighted force of each robot and target, the summation of these forces and resulting movement vector as shown in Figure 4.2. The forces ( $f_{lk}$  and  $g_{li}$ ) are based upon the trajectory of the robot or target. Due to the weighting of the robots force ( $\psi_{li}$ , Figure 4.3) being negative, the robots force is repulsive encouraging the robots to space out. Where as the targets force ( $\omega_{lk}$ , Figure 4.3) transfers from negative, to positive to zero; by remaining negative at close range this promotes keeping a certain distance from the target while still in general acting as an attractive force. The values of the key points are specified based upon the robots sensing capabilities (Parker, 1999), in this work they have been hand coded to reasonable values based on the given robots sensing capability.

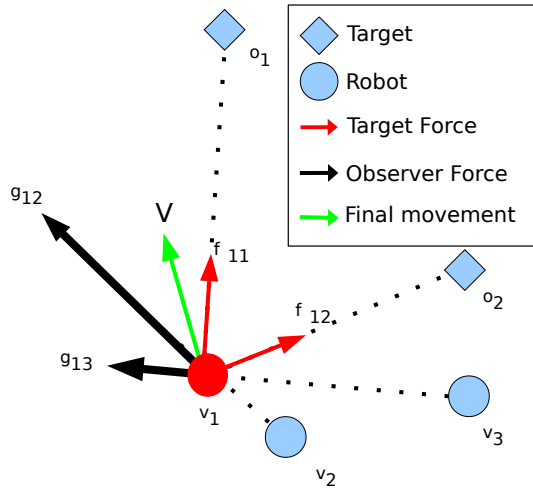


Figure 4.2: The summation of forces using the A-CMOMMT algorithm

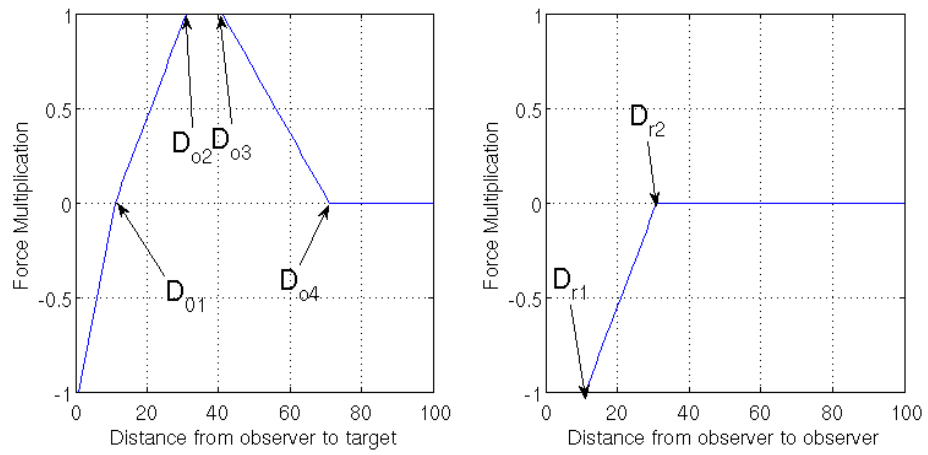


Figure 4.3: Left: Robot to target weighting function  $\omega_{lk}$ . Right: robot to robot weighting function  $\psi_{li}$ . Taken from (Parker, 1999).

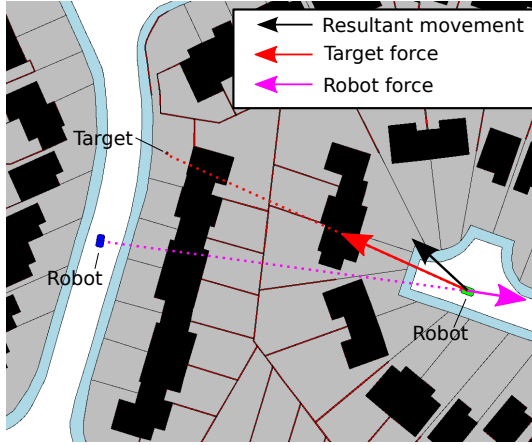


Figure 4.4: An example of a local maxima forming due to the restrictions of A-CMOMMT. Ordnance Survey© Crown Copyright. All rights reserved.

This strategy of directly summing forces, however, is extremely prone to producing a local maximum on the road network in which the robot can become stuck. As shown in Figure 4.4, the force directly pulls the robot towards the target. However as the obstructions generate opposing forces a local maxima will form, trapping the robot. Additionally, superfluous movement tangential to the direction of the road would be encouraged that, in addition to being relatively unhelpful, encourages the robot to leave the road network.

The algorithm is amended, projecting all forces along the road network in order to prevent movement tangential to the direction of the road. An additional pull in the direction of the target along the road network was also added to allow the robot to overcome the frequent local maxima present within a road network ( $F_{tt}$ ).

The final equations are shown in equation 4.2-4.9. The equations culminate in equation 4.2, where  $F$  is the resultant force that acts upon the

robot.  $R$  is the vector of the current road's direction,  $F_c$  is a vector to the centre of the road.  $F_{tc}$  and  $F_{tt}$  are the forces produced by the target,  $F_{tc}$  uses a straight Cartesian vector from the target to the robot,  $F_{tt}$  uses the topological road network to pull along the road network in the direction of the target.  $F_o$  is the force applied by other robots. As can be seen, the target and robot forces are summed as before:  $F_{tc}$  being equivalent to the term  $\sum_{k=1}^n \omega_{lk} f_{lk}$  and  $F_o$  equivalent to  $\sum_{i=1, i \neq l} \psi_{li} g_{li}$ . These are, however, then, projected along the road network via the scalar product with the roads trajectory ( $R$ ). An additional force  $F_c$  is also added that pulls the robot towards the centre of the road in order to keep the robot upon it. Finally as previously stated a force is added ( $F_{tt}$ ) that uses the topological road network to pull towards the target along the road network. The shortest distance from the target to the robot via the road network is used, thus pulling towards the target along the road network. Due to this force already being in the direction of the road there is no need for it to be projected along the road network.

$$\vec{F} = \left[ (\vec{F}_o + \vec{F}_{tc}) \cdot \vec{R} \right] \hat{R} + \vec{F}_{tt} + \vec{F}_c \quad (4.2)$$

The  $F_o$  function can be seen in equation 4.3. This is simply a summation of vectors from all robots to the robot performing the calculation. Each of these vectors are then scaled by the weighting function  $\Theta_{rrf}(d)$  (Equation 4.4), where  $d$  is a distance.  $\Theta_{rrf}(d)$  is an implementation of the weighting function  $\psi_{li}$  (shown on the right of Figure 4.3).

The distance functions traditionally used in the A-CMOMMT algorithm are also replaced with  $\delta(A, B)$  when calculating the distance between robots. This is the topological distance from  $A$  to  $B$  using the road network as opposed to the Cartesian distance that is traditionally used.

$P_x$	The position of robot $x$ .
$\delta(A, B)$	Topological distance from $A$ to $B$ on the road network.
$D_{rx}$	Constants that define the profile of the robot to target weighting function.
$j$	Index of robot.

$$\vec{F}_o = \sum_{i=0, i \neq j}^{i \leq n} P_i \vec{P}_j \Theta_{rrf}(\delta(P_i, P_j)) \quad (4.3)$$

$$\Theta_{rrf}(d) = \begin{cases} -1 & d < D_{r1} \\ \frac{d-D_{r1}}{D_{r2}-D_{r1}} - 1 & D_{r1} < d < D_{r2} \\ 0 & otherwise \end{cases} \quad (4.4)$$

$F_{tc}$  is implemented as a single vector from the position of the robot ( $P_j$ ) to the position of the target ( $\Gamma$ ) as show in in equation 4.5. This is weighted by the function  $\Theta_{rtc}$  that is an implementation of  $\omega_{lk}$  (shown on the left of Figure 4.2). Note that due to this being the Cartesian force the Cartesian distance function,  $\Delta(A, B)$ , is used. The weighting function essentially encodes the concept of pulling towards the target, however keep a



reasonable distance, then at a long distances diminish the influence to zero. At these long distances the topological function ( $F_{tt}$ ) remains active however, still pulling towards the target. However, using the efficient shortest path route.

$$\vec{F}_{tc} = \Gamma \vec{P}_j \Theta_{rtc}(\Delta(\Gamma, P_j)) \quad (4.5)$$

$$\Theta_{rtc}(d) = \begin{cases} \frac{d}{D_{o1}} - 1 & d < D_{o1} \\ \frac{d-D_{o1}}{D_{o2}-D_{o1}} & D_{o1} < d < D_{o2} \\ 1 & D_{o2} < d < D_{o3} \\ 1 - \frac{d-D_{o3}}{D_{o4}-D_{o3}} & D_{o3} < d < D_{o4} \\ 0 & otherwise \end{cases} \quad (4.6)$$

$F_{tt}$  is calculated using a single vector in the direction of the closest node, on the road network, to the target ( $R_\Gamma$ ). This is then again scaled by a weighing function. That essentially produces a diminishing force, when close to the target the topological vector has no influence, thus allowing the original forces used to dominate. However at large distances, where the robot is more prone to get stuck in local minima, the topological force dominates and thus pulls along the road network avoiding these areas.

$$\vec{F}_{tt} = \vec{R}_\Gamma \Theta_{rtt}(\Delta(\Gamma, P_j)) \quad (4.7)$$

$$\Theta_{rtt}(d) = \begin{cases} 0 & d < D_{p0} \\ \frac{d-D_{p0}}{D_{p1}-D_{p0}} & D_{p0} < d < D_{p1} \\ 1 & otherwise \end{cases} \quad (4.8)$$

The effects of the forces  $F_o$ ,  $F_{tc}$ , and  $F_{tt}$  can be seen in Figure 4.5.

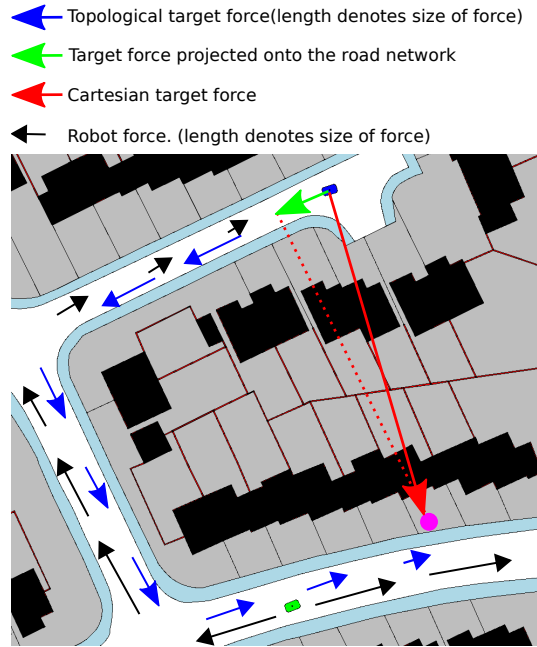


Figure 4.5: Robot force( $F_o$  ←), Cartesian target force( $F_{tc}$  ←) projected along the road network and topological target force( $F_{tt}$  ←). Ordnance Survey© Crown Copyright. All rights reserved.

Finally  $\vec{F}_c$  is a vector pulling towards the centre of the road simply achieved as the vector from the robots current location to the centre of the road ( $C_j$ ).

$$\vec{F}_c = C_j - P_j \quad (4.9)$$

### 4.1.2 E-Jung

The original equations are detailed in section 2.4.1. As noted in this section the algorithms in their original form are unsuitable for this new environment due to the algorithm having no accounting for targets changing region on the topological map. Thus the urgency at a given region is independent of the density of targets in adjacent regions. For this environment this omission is significant since the target will regularly move between adjacent areas. Due to the target also no longer being constrained to the topological map, areas that are close in a Cartesian sense are also of higher risk. Additionally, the existing model only uses the robot's current location when calculating the robot density. In an environment such as the one at hand where travel time is significant, multiple robots could identify the same area as urgent and expend a significant amount of time travelling to a perceived urgent area. This could be significantly redundant, therefore the nodes that each of the team are currently travelling towards in order to make an observation are also taken into account when calculating the robot density. The algorithms are as shown in equations 4.10-4.14. The urgency at a given point is defined as  $U$  in equation 4.10 as a function on the density of the targets ( $D_t$ ) and the density of the robots ( $D_r$ ).

$$U(R) = u(D_r(R), D_t(R)) \quad (4.10)$$

$$u(D_r, D_t) = \begin{cases} \frac{D_t}{D_r} & D_r \neq 0 \\ a_0 D_t & otherwise \end{cases} \quad (4.11)$$

Since this is a single target problem, the concept that an area with no robots that has not been observed in a long time needs to be checked for targets is removed explicitly from the urgency  $u$ , since this will now naturally happen through the definition of  $D_r$  and  $D_t$ . If a robot is present to make an observation, the urgency becomes the target density scaled by the robot density and otherwise proportional to the target density.

$$D_r(R) = \sum_{i=0, i \neq j}^{i < n} D_{rf}(R, P_i) + \sum_{i=0, i \neq j}^{i < n} D_{rf}(R, \Phi_i) \quad (4.12)$$

$$D_{rf}(R, P_i) = \begin{cases} \frac{1}{\delta(R, P_i)} & 1 < \delta(R, P_i) < a_1 \\ 1 & \delta(R, P_i) < 1 \\ 0 & otherwise \end{cases} \quad (4.13)$$

Where  $\Phi_i$  is the current destination of robot  $i$  ( $j$  is the index of the robot performing the calculation) and as before  $P_x$  defines the position of a robot. The robot density ( $D_r$ ) is now inversely proportional to the distance to all the robots as well as their destinations. This therefore takes into account that a robot is heading towards a given area, preventing other robots heading towards the area and making redundant observations. These again use the topological distance calculation  $\delta$ . The inverse proportion is calculated through the robot force ( $D_{rf}$ ).  $D_{rf}$  caps the maxi-

mum density for any given robot to one and also drops the density to zero beyond a given point.

$$D_t(R) = \begin{cases} A & \text{Target lost \& } D_r > a_3 \\ D_{t-1}(R) + \phi & \text{Target lost \& } D_r \leq a_3 \\ \frac{1}{\Delta(R,\Gamma)} & \text{otherwise} \end{cases} \quad (4.14)$$

Where  $A$  is the area covered at that node.

$D_t$  again produces an urgency measure, representing the proximity of the target. While there are no robots present to make an observation, the urgency gradually increases to represent the uncertainty in that location. However if the target is currently being observed, the urgency is reset to being inversely proportional to the distance to the target. Finally if the target is lost but there is a robot in the vicinity to make an observation (or on its way to the location), the urgency is reset to a value relative to the road area at that node ( $A$ ) indicating that an observation has been made and nothing found.  $A$  is used as this encourages large areas to be checked more frequently.

## 4.2 Short Cut Path Algorithm (SCP)

This algorithm was based upon the initial observation that current algorithms are most vulnerable at positions where the target can take “short cuts” between sections of the road network (Figure 4.6). This produces a

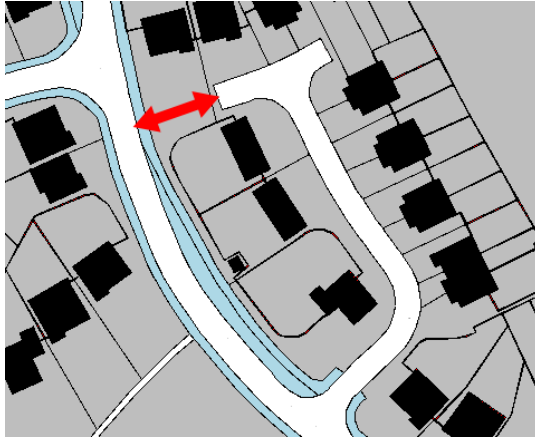


Figure 4.6: An example of possible short cut path. Ordnance Survey© Crown Copyright. All rights reserved.

particularly difficult situation for a naive algorithm due to the assumption that the robot is able to follow the target. Therefore, the areas of the road network that offer potential shortcuts need to be identified and accounted for in the tracking algorithm. To this end Short Cut Paths are identified and added to a graph containing the road network and the additional paths. This graph is referred to as the Short Cut Graph.

The shortcut paths are identified using the prior road map provided to the robot. The robot continually scans for sections of road that are within the field of view. A shortcut is then identified if equation 4.15 evaluates true. Essentially this tests if the topological distance is much greater than the Cartesian distance between the robot's current node, and a road node within the field of view. If so then a shortcut has been found. When a shortcut is identified, a notional link is made between the section of road that the robot is making the observation from, and the road at the other side of the shortcut. As will be shown later, due to these areas being notionally closer together, robots will be attracted to that area to

cover the escape route.

$$\frac{\delta(n_o, n)}{\Delta(n_o, n)} \geq S_d \quad (4.15)$$

The algorithm is based upon developing a cost map at key nodes across the road network then assigning robots to explore the areas of highest risk. Risk is assigned to each node as defined in equations 4.16-4.20.

$t$  Time.

$r(n)$  Immediate node cost.

$u_t(n)$  Urgency for node  $n$  at time  $t$ .

$$u_t(n) = \frac{u_{t-1}(n) + r(n)}{2} \quad (4.16)$$

The immediate cost ( $r(n)$ ) is the urgency calculated at this time. This is filtered using equation 4.16, causing the final cost to gradually converge to the immediate cost so that step changes in the input variables, such as a robot changing its destination does not cause a drastic change in other's cost maps immediately, meaning that they in turn change their plans and drive the system unstable.

$$r(n) = U_o(n)U_e(n)\tau(n) \quad (4.17)$$

The criteria to generate the immediate cost of a node ( $r(n)$ ) is a product of three competing factors, those attributed to the robots ( $U_o(n)$ ), the target ( $U_e(n)$ ) and cost of travelling ( $\tau(n)$ ).

$$\tau(n) = e^{-\delta(n_o, n)} \quad (4.18)$$

The travelling cost associated with a node is simply the topological distance from the robot's current node ( $n_o$ ) and the node in question ( $n$ ). This is scaled non-linearly via the exponential function, to indicate that closer nodes are significantly preferable to distant nodes.

$a_x$                       Normalising constants.

$\gamma$                         Time since the target was observed (represents uncertainty at node).

$\Delta(p_1, p_2)$         Cartesian distance between  $p_1$  and  $p_2$ .

$$U_e(n) = a_3 \left( e^{-a_4 \frac{\Delta(n_e, n)^2}{\gamma}} + e^{-a_4 \frac{\delta(n_e, n)^2}{\gamma}} \right) \quad (4.19)$$

The target cost is formed as inversely proportional to both the topological distance ( $\delta$ , using a graph containing shortcut links) and the Cartesian distance ( $\Delta$ ). The target cost is where the shortcut paths come into play, these paths link nodes that are not adjacent on the road network. However, they are adjacent on the modified network. Thus, when calculating the topological distance, these nodes appear close and thus have a higher



urgency. Additionally, the links' topological distances are shorter than a link on the road network. The distance between two adjacent nodes on the road network is defined as a straight Cartesian distance between the two, however for the short cut links a weighting is applied to this distance causing the link to appear shorter. This makes nodes traversed to via a short cut path more urgent. This is to represent the fact that these areas are of higher risk. The shorter links will thus favour a node that has been traversed to via a shortcut path over an equidistant node that is traversed to purely via the road network. This higher urgency at the opposing side of the link attracts robots to the high cost to cover the route. The target cost at a given node is also scaled by the time since the target was observed, this represents the growing uncertainty about the target's position as time passes. Immediately after an observation is made, the influence of the target is narrow. Then as time increases it becomes broader and broader, encouraging robots to scan wider and wider regions, Figure 4.7 shows how the area of the influence changes over time. As shown, soon after the target was lost (labelled  $t = 0.1$ ) the urgency is focused closely towards the location that that target was last observed. Then as time increases the distribution widens increasing the search area.

$N_i$                       Current destination node of robot  $i$ .

$$U_o(n) = \prod_{i=1, i \neq c}^{i \leq \text{numRobots}} (1 - e^{-a_1 \delta(n, n_i)})(1 - e^{-a_1 \delta(n, N_i)}) \quad (4.20)$$

The cost attributed to the robots ( $U_o(n)$ ) is the product of the topological distances from the node in question to each robot's current position and

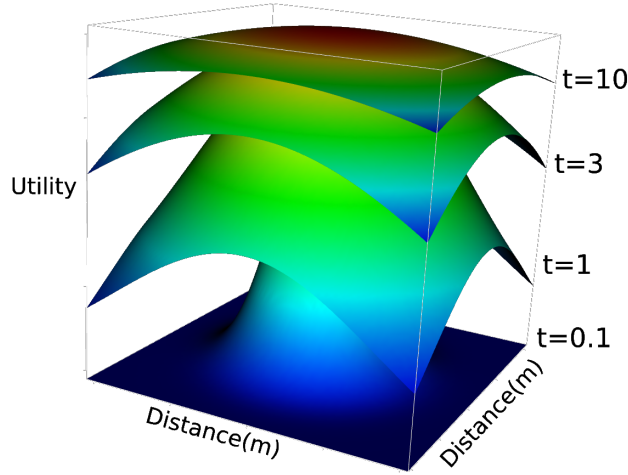


Figure 4.7: Target urgency over time. X,Y axis show Cartesian distance from the centre, where the target was last observed.

destination (the robots current destination will have been decided by previous applications of the algorithm). The closer a node is to a robot or the destination of a robot, the lower its urgency will be.

As with the previous algorithm, once the target is within the field of view of a robot, an algorithm will be needed to manoeuvre the robot to keep it within the field of view. For this the Shortest Escape Path (SEP) algorithm was used, that is described in Section 2.3.2. However, this will again produce a movement with no regard for the environmental restrictions placed upon the robot, the SEP's movement vector is therefore again projected onto the roads vector to keep movement in the direction of the road. This differs slightly from the way that the CUT algorithm works. For the CUT algorithm, once the team has the target within the field of view, the entire team switches over to the E-CMOMMT style algorithm.

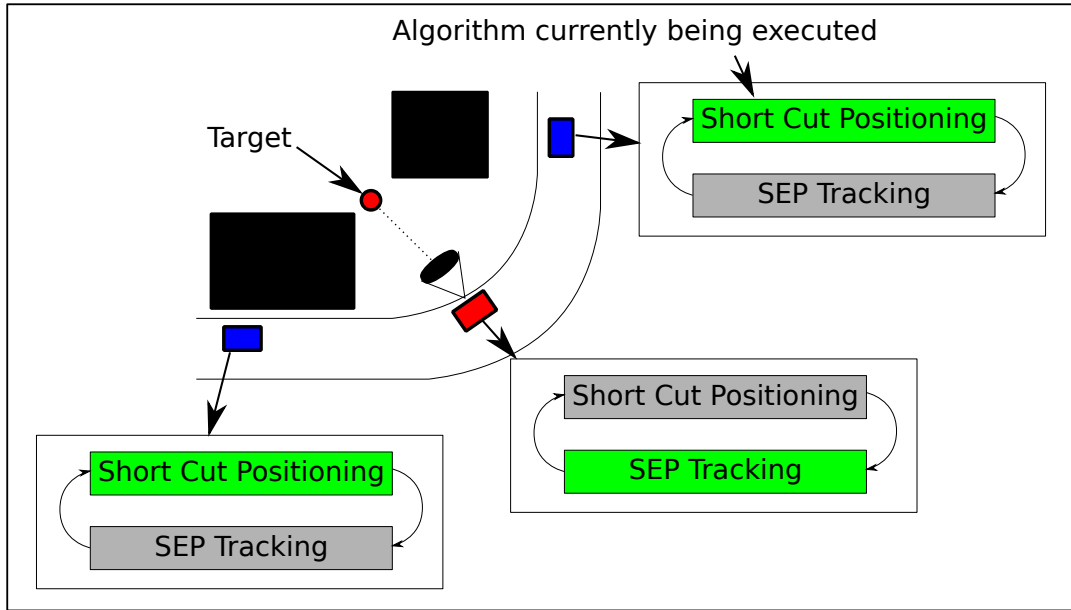


Figure 4.8: Illustration of the switching between algorithms for the SCP tracker.

However for the SCP algorithm only the robots that are currently viewing the target run the SEP algorithm, the others continue as normal. As shown in Figure 4.8 the robot immediately with the target in sight has switched to running the SEP algorithm while the others remain executing the positioning strategy, if either of the other robots movement were to bring them within sight of the target they would also switch to the SEP tracking algorithm.

### 4.3 Branch

This algorithm attempts to place robots such that they are in key positions of the road network, in order to prevent the target from leaving its current area of the network without being observed. This also works

upon a Shortcut Graph, therefore trying to avoid allowing the target to escape by travelling between sections of the road network. The equations are detailed in equation 4.23-4.21 and are built upon the principle:

- The closer a node is to the root node the more urgent it is.
- The more road that is accessible through a given node the more urgent it is.
- Junctions in the network tend to cover critical areas(i.e. maintain views of 2 roads from a single point) thus are more urgent.
- Robots should spread out to cover as much area as possible.

$$u(n, g) = D(n, g)\Lambda(n)S(n, N)B(n) \quad (4.21)$$

The urgency at a given node ( $u$ ) is calculated using four competing factors each representing one of the principles given. In equation 4.21,  $n$  refers to the node for whom the urgency is being calculated.  $g$  is a set of nodes that have previously been assigned for inspection by the algorithm.  $N$  is the “root” node of the network, this is assigned as the node closest to the target.

$D(n, g)$  is a measure of separation between node  $n$  and each of the nodes that have already been assigned for searching ( $g$ ). This is normalised by the radius of the search ( $M$ ). As shown in equation 4.22, the metric

becomes proportional to the distance from the node in question to all of the already marked nodes. Therefore this term represents the concept that “Robots should spread out to cover as much area as possible”.

$$D(n, g) = \prod_{i=1}^{i \leq j} \frac{\delta(n, g_i)}{M} \quad (4.22)$$

$\Lambda(n)$  indicates the amount of the road network that can be accessed by travelling through that node, starting from the target’s position. This therefore represents the concept that “The more road that is accessible through a given node the more urgent it is.”. This is calculated by executing the recursive function  $A(n, P)$  (equation 4.23), which measures the area of network accessible via node  $n$ . The function  $A(n, P)$  is initiated with the arguments  $A(N, [])$  (i.e it is started at the node closest to the target). It then recursively searches the graph in a depth first manner, by exploring the adjacent nodes on the road network.  $L_{in}$  refers to the  $i^{th}$  node adjacent to  $n$  on the road network. Calculating the amount of road network accessible via the road node, this area is then assigned to  $\Lambda(n)$  for each node. Since the road network is a graph with cyclic routes, the argument  $P$  is added to the area calculation so that it can detect that a node has been previously processed and avoid looping infinitely.

$$A(n, P) = \sum_{i=1}^{i \leq j} \begin{cases} \delta(n, L_{in}) + A(L_{in}, [P, L_{in}]) & \delta(n, L_{in}) < M \& L_{in} \notin P \\ 0 & else \end{cases} \quad (4.23)$$

```

while robots are available
   $\eta$ =Most urgent node given the node closest...
    to the target ( $N$ ) and placed nodes( $g$ )
  Assign  $\eta$  to the closest robot to the node
  Add  $\eta$  to list of placed nodes  $g$ 
end loop

```

Table 4.1: Algorithm for assigning branch nodes to robots

$S(n, N)$  is a normalised distance from the node  $n$  to the node  $N$ . This is implemented to favour nodes that are closer to the target, representing “The closer a node is to the root node the more urgent it is.”.

$$S(n, N) = \left( 1 - \frac{\delta(n, N)}{M} \right) \quad (4.24)$$

Finally  $B(n)$  measures the number of leaf nodes each node has, therefore incorporating the concept that junctions are more critical areas of the network.

Nodes are assigned to the robots using the algorithm described in Table 4.1. This method assigns the most urgent node to the closest robot to that node. It then recalculates the urgencies, given the already assigned node, and then assigns the next node to the closest robot (excluding the robot that has already been assigned a node) and so on until all robots have been assigned nodes.

The effects of this algorithm are shown in Figure 4.9. The size of the square around each road node indicates its urgency. As shown, the junctions in the network produce the largest urgencies and thus attract robots to the location, placing them in a good position to respond to the target’s

movement as well as blocking off potential routes that the target can take to escape the team's collective field of view.

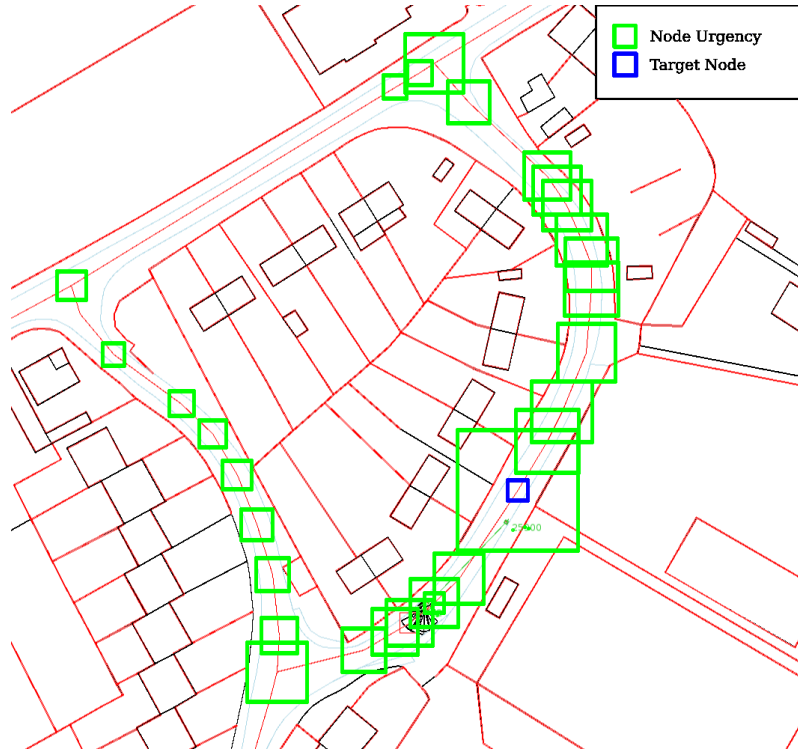


Figure 4.9: Urgency of nodes using Branch positioning. Larger nodes are more urgent. Ordnance Survey© Crown Copyright. All rights reserved.

## 4.4 No Movement

The no movement strategy is provided as a baseline for comparison. This simply leaves each of the robots in their starting positions with their sensors active and tracking the target if it comes within the field of view.

## 4.5 Contribution

In this chapter a number of novel contributions have been made. Firstly three algorithms have been developed for the purposes of target tracking in an urban environment, CUT, SCP and Branch.

CUT is based on two existing algorithms A-CMOMMT and the Jung algorithm. These are modified from their original form so that they are suitable for operation on a road network. These modifications involve projecting the existing forces that are used in the algorithms so that they work in the direction of the road network. This prevents the robot from firstly leaving the road network but also helps it to be more efficient by not performing redundant movement. Additionally a centring force was added to push the robots towards the road network again preventing it from leaving the road network. Topological forces were also introduced to these algorithms in order to prevent the robots becoming trapped in areas of the road network and to manoeuvre along the road network instead.

Two more algorithms are developed from scratch specifically for this environment. The problem of “Shortcuts” has been identified which are particularly dangerous to a team of road based observers. Both of the new algorithms identify and account for shortcut paths.

The SCP algorithm attempts to configure the team in order to account for the Shortcut Paths, and thus reduce their danger. It does this by forming a graph on top of the road network containing these shortcut paths, when calculating cost maps upon this altered graph the areas near these shortcuts are highlighted as significant and thus attract robots to



cover the area. This is used in conjunction with the Shortest Escape Path algorithm, who's movements were again projected to work in the direction of the road network.

Finally the Branch algorithm is defined. This again works with the novel shortcut graph used in the previous algorithm, however, it uses it in a different manner. This attempts to station robots at key areas of the road network in hope that the robots are in good positions to move as the target moves and to block the most likely routes that the target can use to escape observation.

# Chapter 5

## Experiments and Results

### 5.1 Aims

Overall the aim of the experiments is to quantify the performance of each algorithm. In detail the aims are to discover the following:

- The overall performance of the algorithms over a number of environments.
- Identify features of the environment that cause problems to one or more of the algorithms.
- Contrast the performance of each algorithm in problematic areas.
- Quantify how the algorithms are affected by the speed of the target.
- Quantify how the algorithms perform with various numbers of robots.

The effectiveness of an algorithm is defined as the percentage of time that the target remains under observation ( $\frac{\text{Time samples in which the target was observed}}{\text{Total simulation time}}$ ).

## 5.2 Design of experiments

The experiments performed involve simulating the robots in a number of environments. A variety of maps are used to achieve a number of the goals. Firstly varying the maps removes any bias that a particular algorithm may have towards a specific map and allows us to achieve the goal of producing an overall performance for each algorithm. Exploring various maps will also allow the identification of particular common features that the algorithms find challenging and to contrast their performance in these areas.

The aim of quantifying how the algorithms cope with various speeds of the target is achieved through the variation of the target's speed between trials. Six different speeds of the target relative to the robots are used (0.31, 0.375, 0.43, 0.518, 0.75 and 1.0), the speeds are relative to the robot's maximum speed (i.e.  $speed = \frac{targetSpeed}{robotSpeed}$ ). The speeds are noted in this way since obviously the absolute speed of the target is irrelevant. However, what is important is the relationship between the target's and robot's speed. As a point of reference however, the robots absolute speed was fixed at  $4m/s$  while the targets ranged from  $1.25m/s$  to  $4m/s$ . The lower speed ratio was chosen due to the fact that at this speed ratio the results become highly skewed towards 100% effectiveness, a range of speed ratios are then tested up to equal speed with the target. The speed ratio

of one is a hard limit due to the fact that once the target becomes faster than the robot the task for tracking becomes ultimately impossible as the target will always be able to out pace the robots and escape their field of view irrespective of the tracking algorithm. The specific speed ratios were chosen while tuning damping factors on the targets control mechanism. However preliminary testing showed that at the higher speeds the algorithms tended towards a final value (as will be shown later, Figure 5.18). More samples were therefore taken at the lower speeds where larger changes can be observed, than in the higher linear region.

Tracking is impossible if the target is faster than the robots over an indefinite period of time due to the targets ability to eventually move beyond any of the team's sensor range. However, with a larger collective sensor area, larger teams should be able to increase efficiency in such a scenario. Additionally in a scenario where the target is evasive it would potentially be possible to trap the target in a local area despite its speed.

Similarly to assess the effects that the number of robots makes, the number of robots was simply varied between trials. This was varied between 2 and 5. Five was found as an upper limit due to the memory limitations of the computers executing the algorithms.

The parameters were varied between trials. At least 100 trials of each possible combination of map, speed ratio and number of robots was run. For each of these trials the starting positions of the robots and target are randomised in order to remove any bias from a single configuration.

When producing the starting configuration it is assumed that the ini-

tial location of the target is known and at least one member of the team initially has the target in sight. The initial location of the target is a randomly chosen road node, the remaining robots are then placed randomly at road nodes between  $20m$  and  $400m$  from the initial target.

In summary the fixed variables for each trial are as follows:

- Six different speed ratios of the target relative to the robot (0.31, 0.375, 0.43, 0.518, 0.75 and 1.0)
- Four maps noted as Map 1 - Map 4
- Between 2 and 5 robots.

The only randomised variable is the starting positions of the target and robots.

### **5.2.1 Test Areas**

The maps range from  $966,000m^2$  to  $2,017,000m^2$  and are from areas around Warwickshire and Birmingham, sourced from Ordnance Survey©. These are shown in Figures 5.1-5.4. Significant aspects of these maps have been labelled as follows:

- |   |  |
|---|--|
| A | Large open areas such as parks and fields.   |
| B | Areas of the map that are entirely enclosed and thus not accessible to the target. |

- C            Small areas of road network that due to the limits of the map are not accessible to the robots.
- D            Particularly notable short cut paths.
- E            Areas that are almost entirely obscured from view of the road network.
- F            Long straight roads with few junctions.

These significant areas were identified in initial testing as significant as either corresponding to areas that produce significantly high or low effectiveness. The maps were generally chosen so that each provides a variety of these significant areas. This variety allows for the testing of how each algorithm copes with a number of examples of each type of problem, and to compare their performance.

The areas are generally of suburban areas of towns, except for map 3 that is an entire village. An inherent problem with simulating this environment is that due to memory constraints the region must be bounded. However this commonly produces areas of the map that are inaccessible to the robot by cutting out a section of the road network (these are labelled C on the annotated maps). This has lead to choosing specific areas that contain a minimum of such areas.

Rural areas were not chosen due to the sparse nature of roads in such areas providing little scope for the algorithms to perform. Town and city centres were also avoided due to heavy pedestrianisation limiting the algorithms scope for movement. Such areas also have fairly complex road

structures such as overpasses and ring-roads, which are impossible to account for in a 2 dimensional simulation, and also difficult to produce algorithms that are able to navigate such structures.

A number of significant areas have been labelled on the maps. As will be shown in the following sections these are generally areas that cause particular difficulty. That is with the exception of the areas labelled B. Such areas appear conspicuously absent in the results for having no readings however this is due to them not being accessible to the target due to being entirely enclosed by fences. A number of areas that are not entirely enclosed have been labelled as inaccessible due to the entrance of such an area being so small that a randomly moving target did not manage to enter the area. Inaccessible areas of road network (labelled C) also correspond to areas of low loss due to the fact that these are generally cut off, by walls and buildings, from the target too. However in cases where the target can access the region losses can be found.

## **Map 1**

Map 1 is shown in Figure 5.1. The dimensions of this map are  $1316 \times 940m$ . This provides a number of interesting aspects, in particular there are a number of long straight roads. There are few short cut paths, however, the path labelled D1 is particularly significant due to the fact that the distance required to reach the other side via the road network is extremely long. The most significant area that this map provides is the large open space labelled A2. This is a park with a number of easily

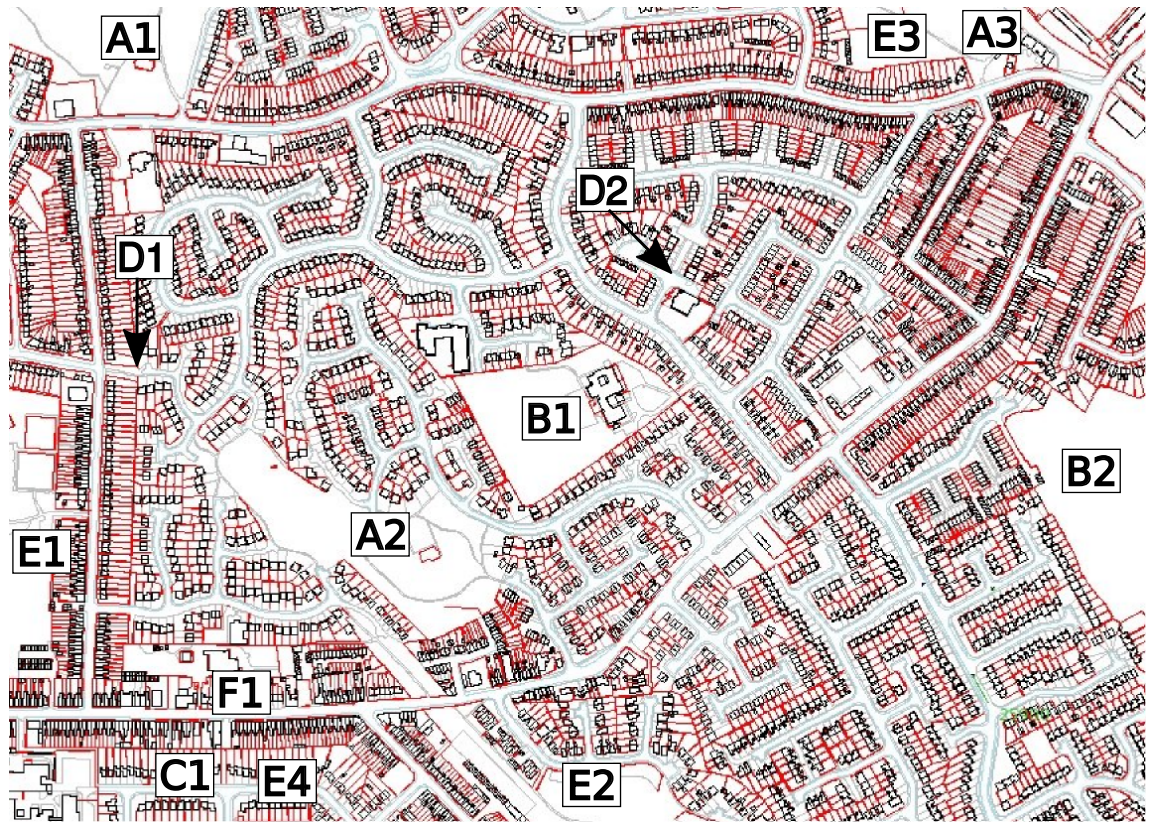


Figure 5.1: Labelled image of Map 1. Ordnance Survey© Crown Copyright. All rights reserved.

- A Large open areas such as parks and fields.
- B Areas of the map that are entirely enclosed and thus not accessible to the target.
- C Small areas of road network that due to the limits of the map are not accessible to the robots.
- D Particularly notable short cut paths.
- E Areas that are almost entirely obscured from view of the road network.
- F Long straight roads with few junctions.



accessible entrances and exits meaning that the target can easily enter and leave through any number. This essentially forms a shortcut path, however, would not be detected due to the distance between both sides meaning that the other side cannot be seen. This produces a particularly dangerous area. There are also a number of areas obscured from the road network (labelled E).

## **Map 2**

Map 2 is shown in Figure 5.2. The dimensions of this environment are  $1316 \times 914m$ . This has a significant number of shortcut paths within it giving opportunity to see their effects. The shortcut paths are also fairly highly concentrated. There are again a large number of areas occluded from the road network (labelled E) that the target may gain access to and avoid detection. In contrast to the previous map, these areas have quite large entrances meaning that a randomly moving target is more likely to reach such areas, unlike the previous map where the target must travel through relatively narrow entrances. This map also contains no particularly large open areas in the centre of the map, instead the only large open area is at the top right. Due to being on the edge of the map it is not surrounded by road, this means that getting close to the target is even more challenging and even impossible.

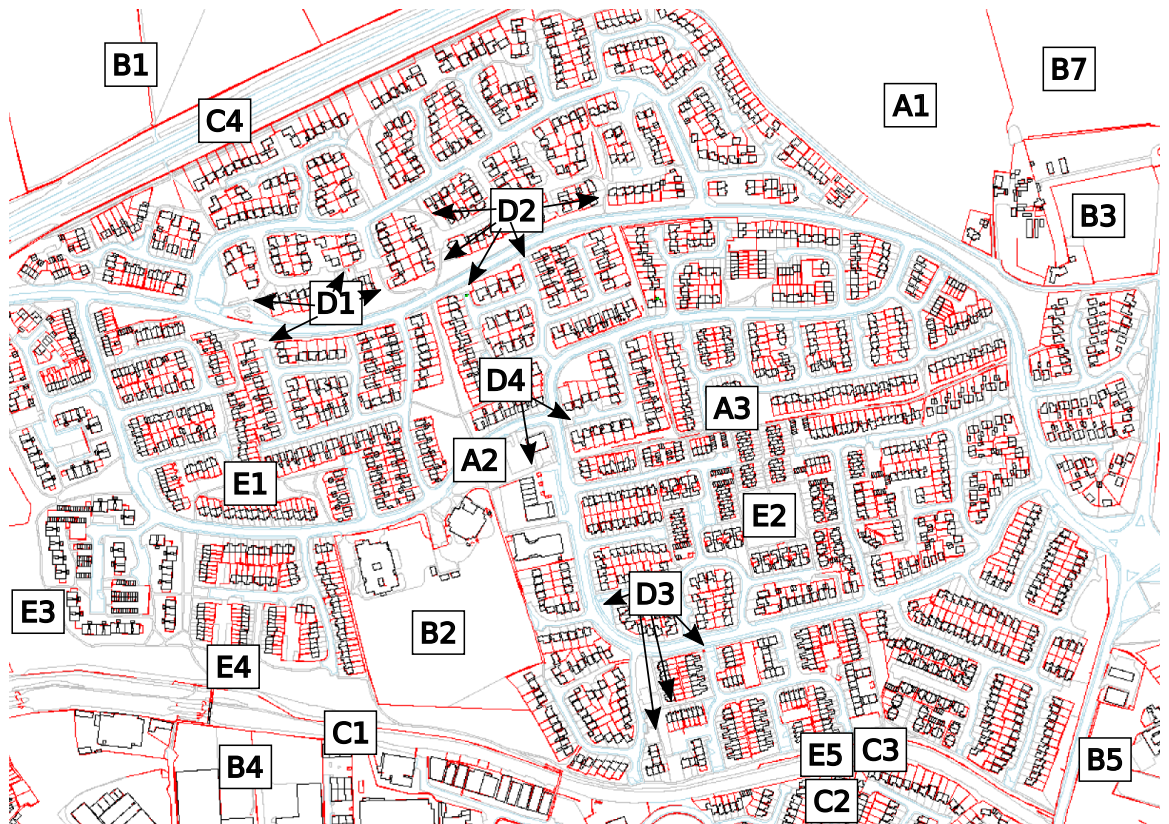


Figure 5.2: Labelled image of Map 2. Ordnance Survey© Crown Copyright. All rights reserved.

- A Large open areas such as parks and fields.
- B Areas of the map that are entirely enclosed and thus not accessible to the target.
- C Small areas of road network that due to the limits of the map are not accessible to the robots.
- D Particularly notable short cut paths.
- E Areas that are almost entirely obscured from view of the road network.
- F Long straight roads with few junctions.

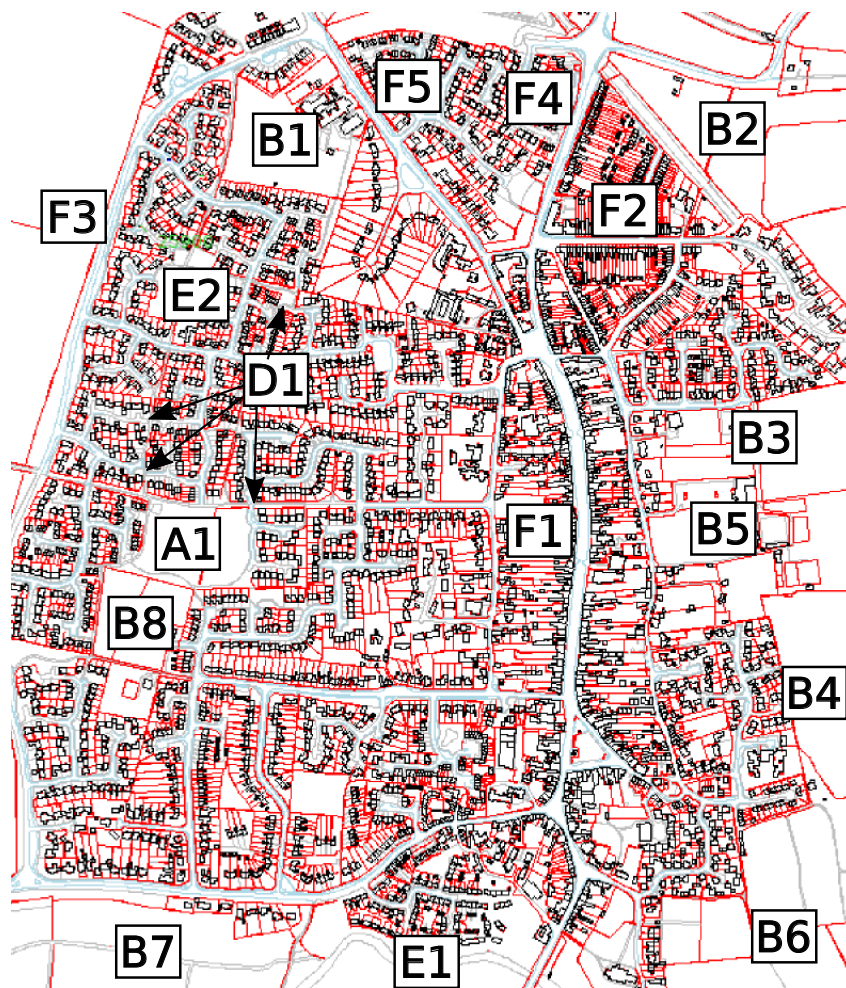


Figure 5.3: Labelled image of Map 3. Ordnance Survey© Crown Copyright. All rights reserved.

- A Large open areas such as parks and fields.
- B Areas of the map that are entirely enclosed and thus not accessible to the target.
- C Small areas of road network that due to the limits of the map are not accessible to the robots.
- D Particularly notable short cut paths.
- E Areas that are almost entirely obscured from view of the road network.
- F Long straight roads with few junctions.

### **Map 3**

Map 3 is shown in Figure 5.3. The dimensions of this environment are  $1338 \times 1508m$ . The intention of this map was to look at an entire urban area as opposed to a small section as in the other environments. This is therefore an entire village and the largest of the environments. This also provides the densest concentration of buildings and walls, consequently there is little free and open space. A number of shortcut paths also exist. Again due to the complexity of the road network these are particularly threatening due to the distance required to travel to cover them. A large number of long roads are also present within this map.

### **Map 4**

Map 4 is shown in Figure 5.4. The dimensions of this environment are  $1174 \times 832m$ . The dominating feature to this map is the central open area in a similar manner to map 1. This area again is easily accessible due to the lack of walls around the area. This again forms an area in which the target can easily enter and leave, effectively forming a short cut path as well as an area that the robots will find hard to get close to and track. This map also contains a fairly high concentration of shortcuts in the D1 and D2 regions. Despite in general not containing a large number of straight roads this does contain one of the longest roads with no junctions in the area of F1.

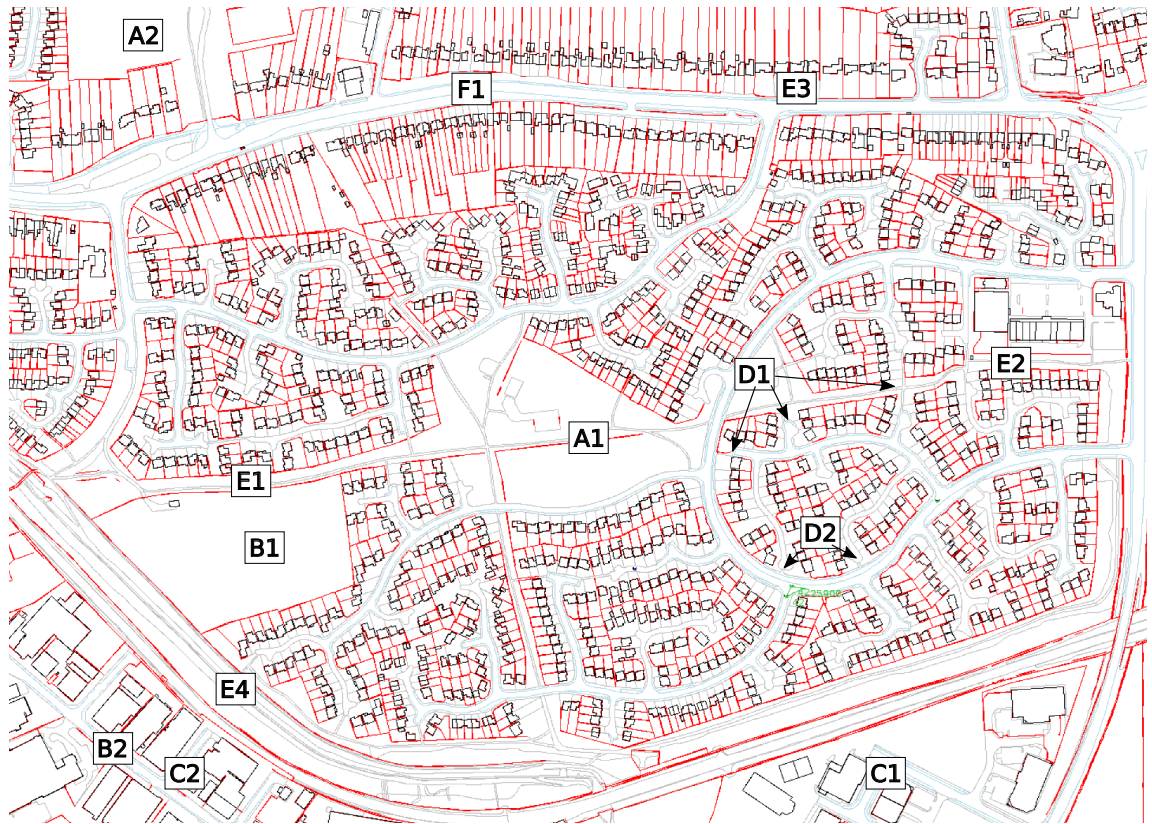


Figure 5.4: Labelled image of Map 4. Ordnance Survey© Crown Copyright. All rights reserved.

- A Large open areas such as parks and fields.
- B Areas of the map that are entirely enclosed and thus not accessible to the target.
- C Small areas of road network that due to the limits of the map are not accessible to the robots.
- D Particularly notable short cut paths.
- E Areas that are almost entirely obscured from view of the road network.
- F Long straight roads with few junctions.

### **5.2.2 Trial Structure**

The results were run over 8,000 trials. These were executed on a cluster with roughly 100 computers. In each trial the target and robots are placed on the road network and the simulation is executed for 10 minutes. The data that the robots' sensors collect was logged throughout the trial. The effectiveness metric is subsequently calculated after the simulation using the recorded logs.

## **5.3 Results**

### **5.3.1 Analysis of Maps**

In the proceeding sections the results for each map are presented. In each section the effectiveness for each run is calculated and presented in the tables and charts. The tables present the average effectiveness calculated for each simulation run, these results are then compared by plotting the effectiveness against the number of robots used. The effectiveness for a given configuration is calculated as the mean effectiveness from all trials performed, where the starting positions were randomised between trials. Appendix B shows the histograms of the effectiveness. As can be seen the effectiveness is skewed towards 100% at the lower speeds, particularly for the better performing algorithms CUT and SCP. Then as the target reaches the same speed as the robots this becomes skewed towards 0% for all algorithms.



A heat map is also shown in each section (figures 5.7, 5.10, 5.12 and 5.17), these indicate the locations where the target tends to become lost for each map. On these heat maps blue denotes an area where the target was never lost, then transferring through yellow to red denoting an area where the target was lost for a significant amount of time. Further maps are shown in Appendix A, these have been normalised to show the percentage of time that the target spent in the region in which it was not observed (i.e.  $\frac{\text{time spent in region while not observed}}{\text{total time spent in region}}$ ) this was also filtered to remove areas in which the robot spent a negligible amount of time. The heat maps displayed in the following sections denote the areas of most importance due to displaying the areas in which the most overall time was spent lost. However, since certain areas are significantly bad, a disproportionate amount of readings are made in these areas and obscure others when shown on a single scale. The graphs in Appendix A therefore show areas that incur a high probability of losing the target whether or not a significant amount of time was spent there.

## Map 1

The table of results derived from the trials is shown in Table 5.1, these are then visualised in the plots in Figure 5.5. Overall, for this map the SCP algorithm performs best with an average performance of 58.5%, closely followed by CUT at 55.8% and Branch fairly significantly behind on 48.4%. The baseline of No Movement achieving 35.9%.

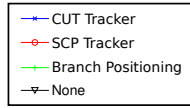
A heat map that shows the areas in which the target spent the majority of its time lost is shown in Figure 5.7. One of the most notable common

Number of Robots	CUT	SCP	Branch	None
Speed Ratio 0.31				
2	0.730	0.758	0.621	0.387
3	0.773	0.748	0.634	0.419
4	0.778	0.790	0.658	0.467
5	0.795	0.775	0.665	0.515
Speed Ratio 0.375				
2	0.614	0.649	0.522	0.328
3	0.681	0.697	0.544	0.405
4	0.748	0.741	0.600	0.468
5	0.827	0.817	0.666	0.506
Speed Ratio 0.43				
2	0.533	0.558	0.458	0.299
3	0.585	0.644	0.538	0.363
4	0.647	0.658	0.562	0.413
5	0.645	0.671	0.549	0.430
Speed Ratio 0.518				
2	0.456	0.536	0.430	0.276
3	0.563	0.581	0.457	0.338
4	0.560	0.594	0.480	0.369
5	0.615	0.627	0.532	0.434
Speed Ratio 0.75				
2	0.326	0.387	0.302	0.256
3	0.418	0.477	0.389	0.289
4	0.387	0.432	0.357	0.286
5	0.434	0.475	0.385	0.343
Speed Ratio 1.0				
2	0.268	0.280	0.238	0.203
3	0.318	0.372	0.340	0.243
4	0.356	0.401	0.351	0.285
5	0.341	0.377	0.328	0.292

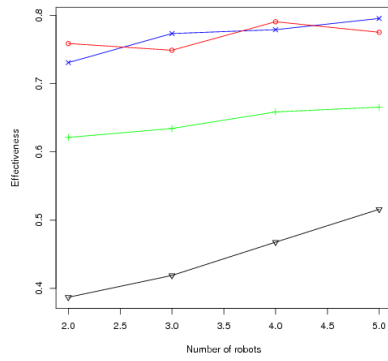
Table 5.1: Effectiveness for map 1.

$$Effectiveness = \frac{\text{Time samples in which the target was observed}}{\text{Total simulation time}}$$

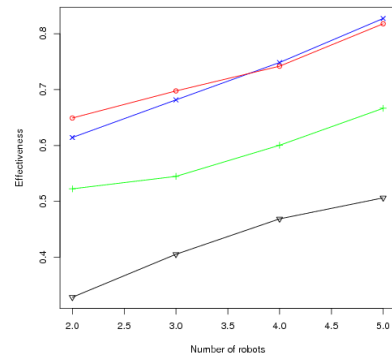




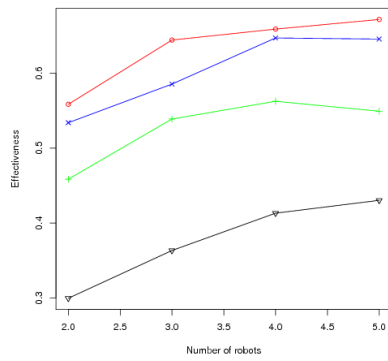
0.31



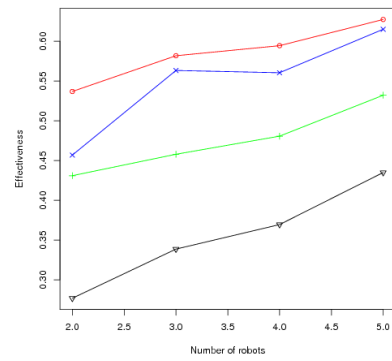
0.375



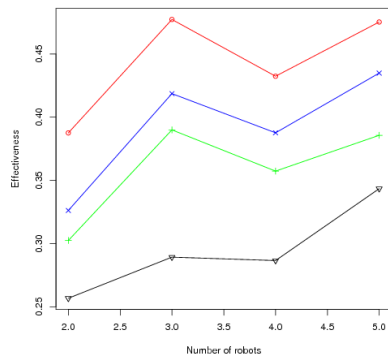
0.43



0.518



0.75



1.0

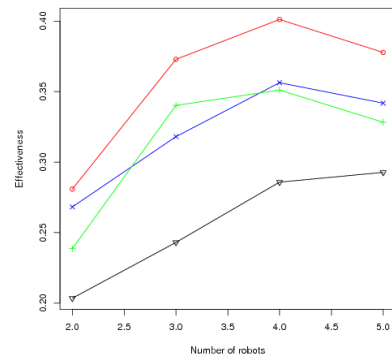


Figure 5.5: Effectiveness against number of robots for map 1. Speed relative to the robots labelled above each graph.

failure areas of all algorithms appears to be the large open space labelled A2 and A3. These areas will obviously always have a higher than normal probability of losing the target due to the lack of roads in the area of the target for robots to travel down and maintain a field of view. It should be noted that the ability of the SCP and Branch algorithms to implement their strategy is particularly impaired due to the reliance on the method used to generate the short cut paths. Due to the lack of a prior map of the environment, the short cut paths are not known and thus generated on the fly as described in section 4.2. The algorithm relies upon being able to view both sides of a shortcut at the same time. However for large open spaces, even though a shortcut does exist the other side is outside the sensor radius, thus not observed and a shortcut is not formed. Consequently the SCP and Branch algorithm will not necessarily move to the opposing side of the open space to cover if the target moves out of the opposing side. This impairment seems to have meant that in these areas the two algorithms based upon Shortcut Paths fair no better than the CUT algorithm. In the isolated example in the area of A3 (shown in Figure 5.6), the Branch algorithm fairs worse in open spaces. This is due to the Branch algorithm not attempting to keep a robot in direct view of the target, merely posting robots at fixed positions. If these fixed positions do not then naturally have a good view of the open space it will be highly impaired. The lack of readings in the area A1 is due to the narrow entrance to the area, meaning that the target seldom entered.

The results also highlight that dead ends seem to coincide with areas of the road network that are occluded from the road network. Thus dead

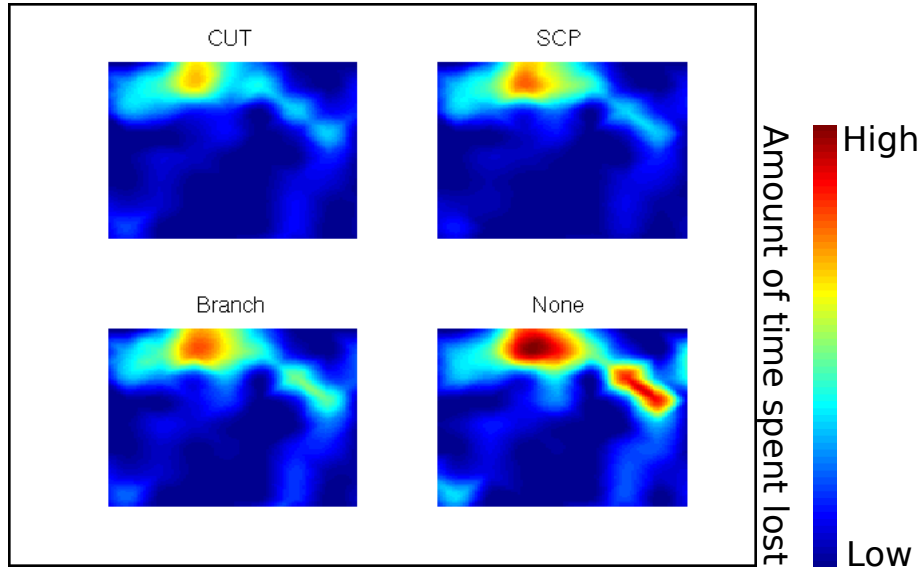


Figure 5.6: Close up of area A3 on Map 1 (Figure 5.1). Colour map scaled to highlight the differences in performance.

ends are also coincidental to high losses. From observation of the various searching algorithms, generally as robots approach dead ends the proximity of a robot leads to the area gradually becoming less and less urgent. Eventually other areas become more significant and the robot leaves to explore that area before having fully explored the local vicinity.

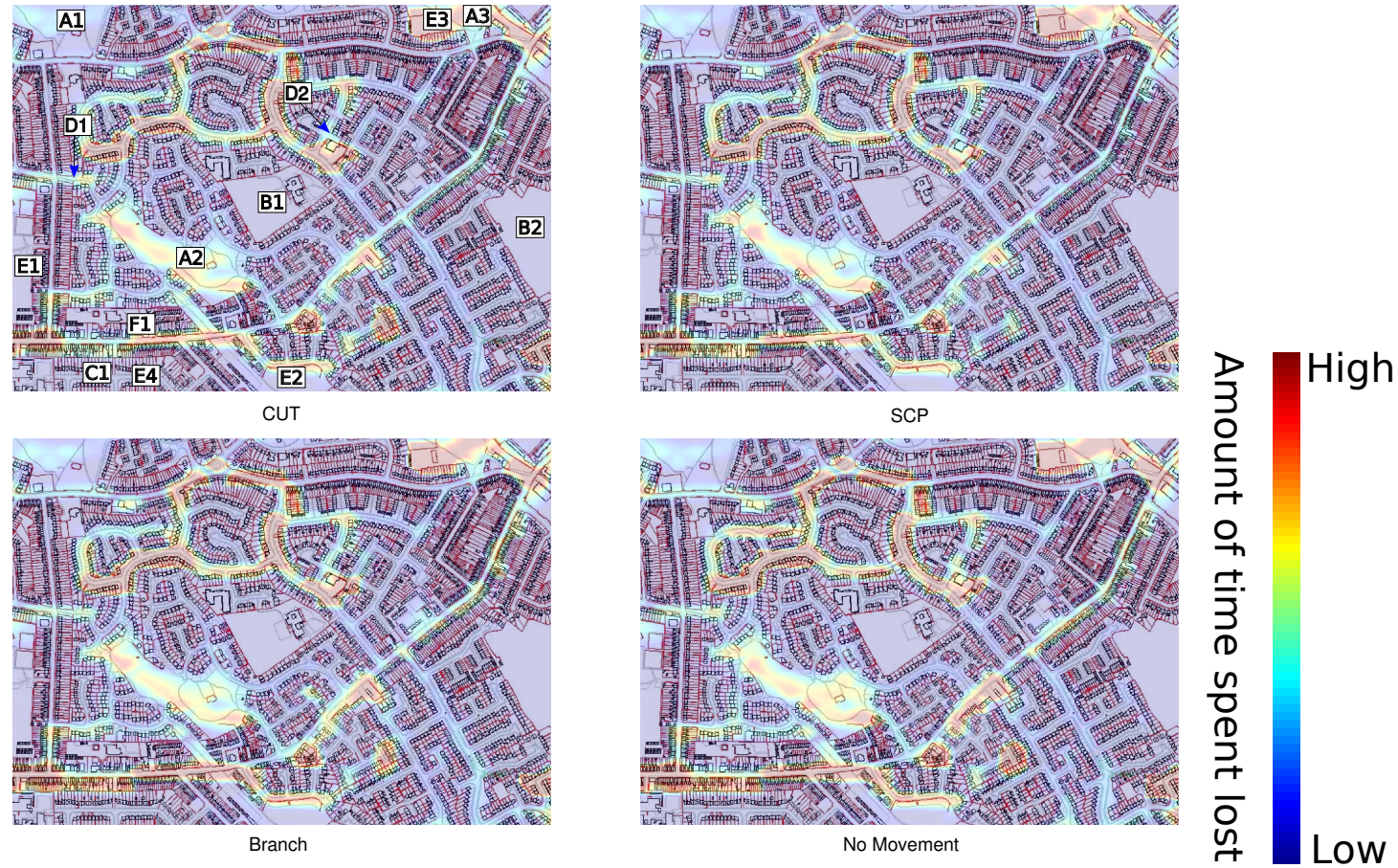


Figure 5.7: Areas of highest risk for map 1(Figure 5.1) and various speed ratios (highest risk is denoted by red).  
 Ordnance Survey© Crown Copyright. All rights reserved.

The long straight road in the area of F1 also show a higher than normal loss. Due to straight roads containing fewer nodes than curved roads, then all algorithms will be at a slight disadvantage as the searching becomes coarser. However the Branch algorithm will be particularly disadvantaged due to it merely trying to place the robots at fixed positions. Therefore as the nodes become coarser the robots will be positioned further away from the target. This can again be seen on the heat maps in Appendix A.1, where the SCP and CUT do not register significantly high readings in these areas until the high speed ratios of 0.75 or 1. The Branch algorithm however starts to see losses sooner at the 0.43 speed ratio. One notable feature of the results is the presence of a number of long straight roads that do not correlate with a high loss rate. This is due to the fact that the entry to these roads is relatively narrow. Consequently a randomly moving target with obstacle avoidance is unlikely to find the entrance and go down such roads. Consequently the target did not enter such areas and few readings were made.

Higher areas are also present in the vicinity of the shortcut paths D1 and D2. The significance of these areas are discussed in later maps as these types of area are more prominent.

## **Map 2**

As shown in Table 5.2 and Figure 5.8, similar results are seen as for map 1 with CUT and SCP overall achieving the better results with the Branch algorithm significantly behind. Overall CUT performs best achieving an

effectiveness overall speed ratios of 47.5%, SCP 46%, Branch 32.3% and No Movement 28.6%.

The graphs in Figure 5.8 show that the effectiveness is far more dependant upon the number of robots as the target reaches higher speed ratios. At lower speed ratios the gradient of the line is far shallower than at the high speed ratios. This is to be expected as at low speed ratios the effectiveness tends towards one, thus the robots are able to keep track of the target. As the speed ratio increases the target escapes observation of a single robot more often, the number of robots in the team becomes important as a larger area of the environment can be covered.

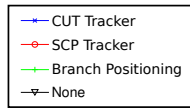
Again the only large dominant space of this area (A1, Figure 5.10) produces an area that the target spends a significant amount of time not under observation.

An additional failure common to all algorithms are areas that are mostly obscured from the road network, these are labelled E and are particularly visible on a normalised map as shown in Figure 5.9. This shows that even in these areas the probability that the robots will lose sight of the target is extremely high. Unfortunately, due to the limitations of the robots there is little possibility for improvement in obscured areas. The best that could be expected would be to identify if the target is heading to such an area and then indicate that a different form of tracking will be required. One possibility is using boundary coverage to attempt to encompass the area such that as soon as the target leaves, it will be observed. An alternative would be if robots that were able to enter such areas were available activate them.

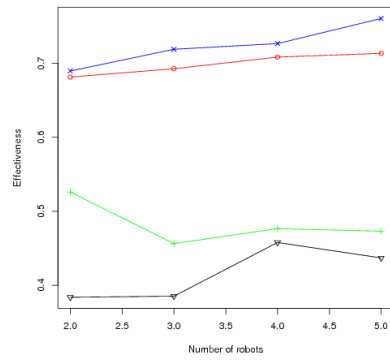
Number of Robots	CUT	SCP	Branch	None
Speed Ratio 0.31				
2	0.689	0.681	0.526	0.383
3	0.718	0.692	0.456	0.385
4	0.726	0.708	0.476	0.457
5	0.760	0.713	0.473	0.437
Speed Ratio 0.375				
2	0.592	0.600	0.429	0.342
3	0.651	0.612	0.428	0.380
4	0.654	0.634	0.424	0.362
5	0.656	0.608	0.410	0.417
Speed Ratio 0.43				
2	0.516	0.519	0.366	0.277
3	0.555	0.538	0.376	0.314
4	0.564	0.542	0.385	0.343
5	0.603	0.541	0.351	0.334
Speed Ratio 0.518				
2	0.460	0.415	0.293	0.227
3	0.473	0.449	0.317	0.257
4	0.459	0.440	0.285	0.279
5	0.493	0.473	0.304	0.307
Speed Ratio 0.75				
2	0.249	0.226	0.168	0.158
3	0.243	0.265	0.206	0.161
4	0.285	0.291	0.209	0.199
5	0.303	0.320	0.217	0.228
Speed Ratio 1.0				
2	0.150	0.153	0.139	0.132
3	0.188	0.190	0.156	0.154
4	0.198	0.198	0.169	0.164
5	0.208	0.226	0.180	0.173

Table 5.2: Effectiveness for map 2.

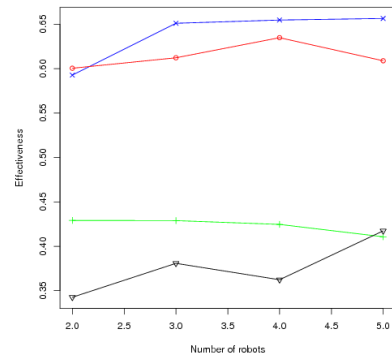
$$Effectiveness = \frac{\text{Time samples in which the target was observed}}{\text{Total simulation time}}$$



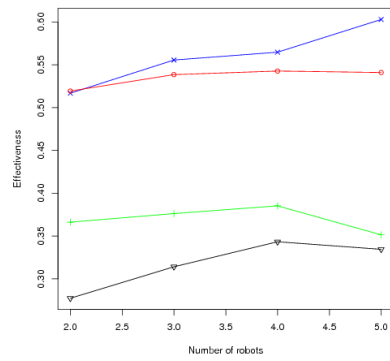
0.31



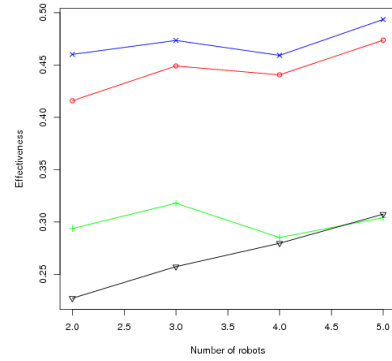
0.375



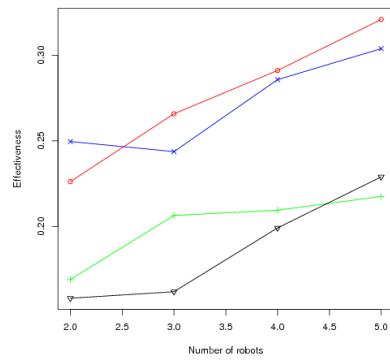
0.43



0.518



0.75



1.0

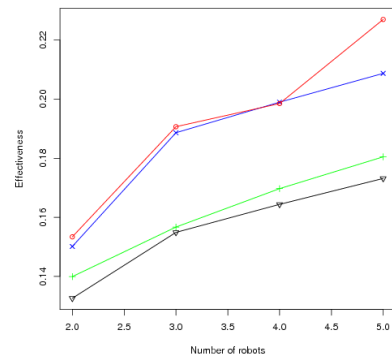


Figure 5.8: Effectiveness against number of robots for map 2. Speed relative to the robots labelled above each graph.



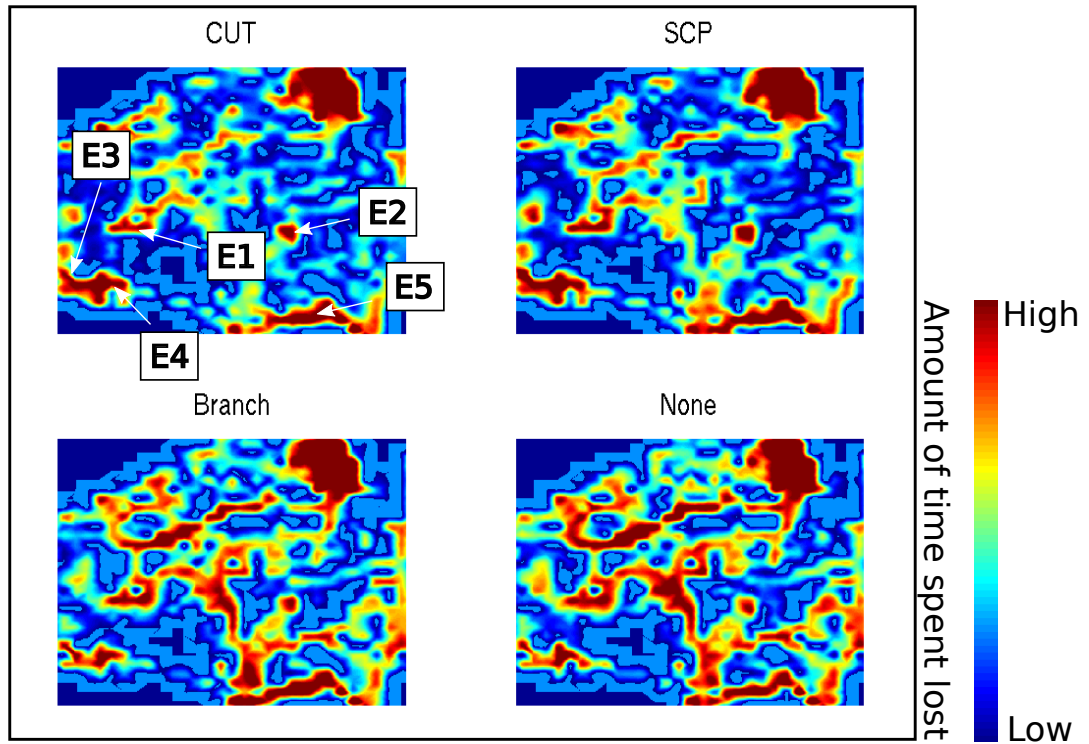
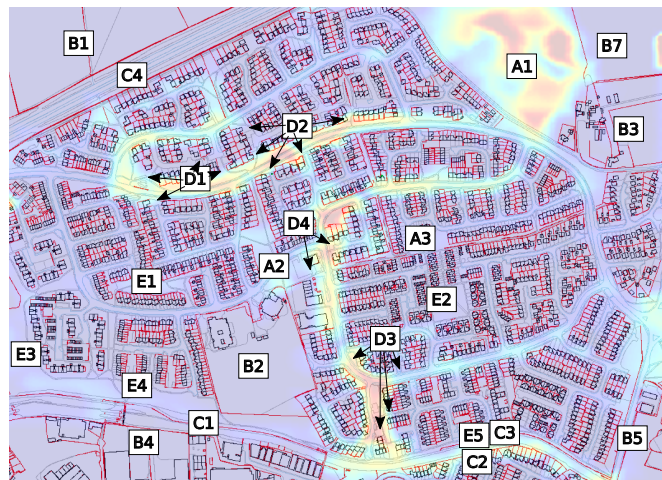
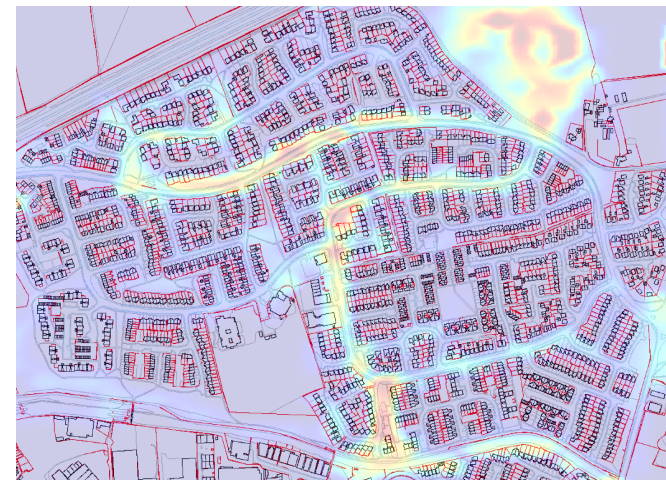


Figure 5.9: Normalised heat map of map 2 with heavily obscured areas labelled.

As expected, high losses are found in the areas that contain clusters of shortcut paths (labelled D). The differences between each algorithms performance in these areas is discussed fully in Section 5.3.2.



CUT



SCP



Branch



No Movement

Amount of time spent lost

High

Low

Figure 5.10: Areas of highest risk for map 2(Figure 5.2) and various speed ratios (highest risk is denoted by red). Ordnance Survey© Crown Copyright. All rights reserved.

### **Map 3**

Overall the CUT algorithm performs best in this map with an effectiveness of 55.3% again closely followed by SCP at 54.7%. Once more Branch falling significantly behind at 45.6% and finally No Movement at 36.8%.

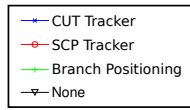
This is a particularly dense map, with few open spaces and short cut paths. Also a significantly high percentage of the environment is inaccessible to the target due to being occupied by buildings or fences. As a result no single area is singled out as being particularly poor. Looking at the normalised maps in Appendix A.3, we see again the development of long roads being a particular problem, especially for the Branch algorithm.

Other than the common straight roads on this map, the other areas in which the target gets lost is similar to previous maps. The most prominent areas that produce a high risk of losing the target are mostly occluded areas. However due to the dense nature of this environment there are few particularly large areas. At higher speed ratios (as shown in Appendix A.3) there is some development around the cluster of short cut paths labelled D1.

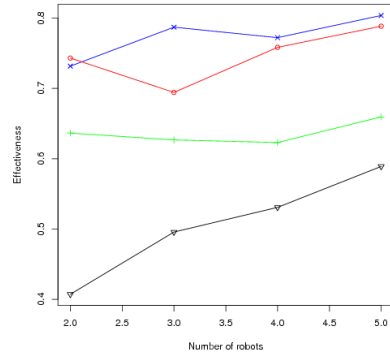
Number of Robots	CUT	SCP	Branch	None
Speed Ratio 0.31				
2	0.731	0.743	0.636	0.407
3	0.787	0.694	0.626	0.495
4	0.772	0.758	0.622	0.530
5	0.803	0.788	0.659	0.588
Speed Ratio 0.375				
2	0.619	0.659	0.562	0.387
3	0.691	0.606	0.542	0.391
4	0.702	0.719	0.590	0.449
5	0.714	0.700	0.602	0.478
Speed Ratio 0.43				
2	0.537	0.510	0.427	0.328
3	0.575	0.553	0.432	0.350
4	0.580	0.593	0.466	0.386
5	0.593	0.593	0.495	0.429
Speed Ratio 0.518				
2	0.449	0.456	0.378	0.270
3	0.528	0.559	0.455	0.348
4	0.505	0.551	0.412	0.341
5	0.549	0.559	0.441	0.423
Speed Ratio 0.75				
2	0.471	0.490	0.377	0.280
3	0.502	0.537	0.401	0.299
4	0.521	0.532	0.457	0.390
5	0.578	0.564	0.504	0.422
Speed Ratio 1.0				
2	0.206	0.202	0.171	0.144
3	0.263	0.243	0.212	0.210
4	0.276	0.254	0.239	0.238
5	0.319	0.264	0.234	0.252

Table 5.3: Effectiveness for map 3.

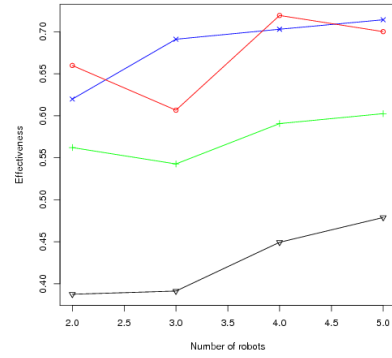
$$Effectiveness = \frac{\text{Time samples in which the target was observed}}{\text{Total simulation time}}$$



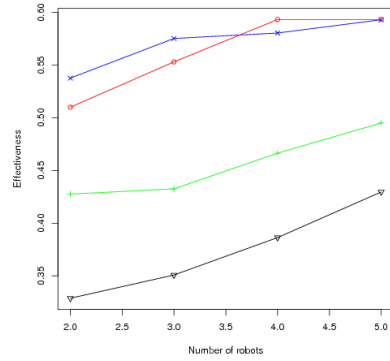
0.31



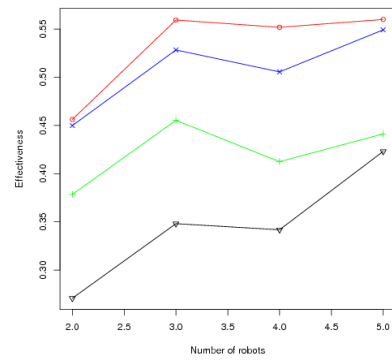
0.375



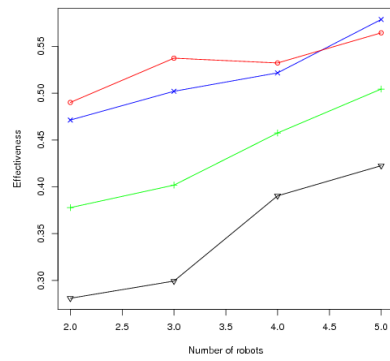
0.43



0.518



0.75



1.0

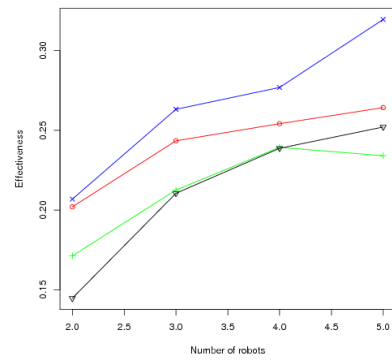


Figure 5.11: Effectiveness against number of robots for map 3. Speed relative to the robots labelled above each graph.



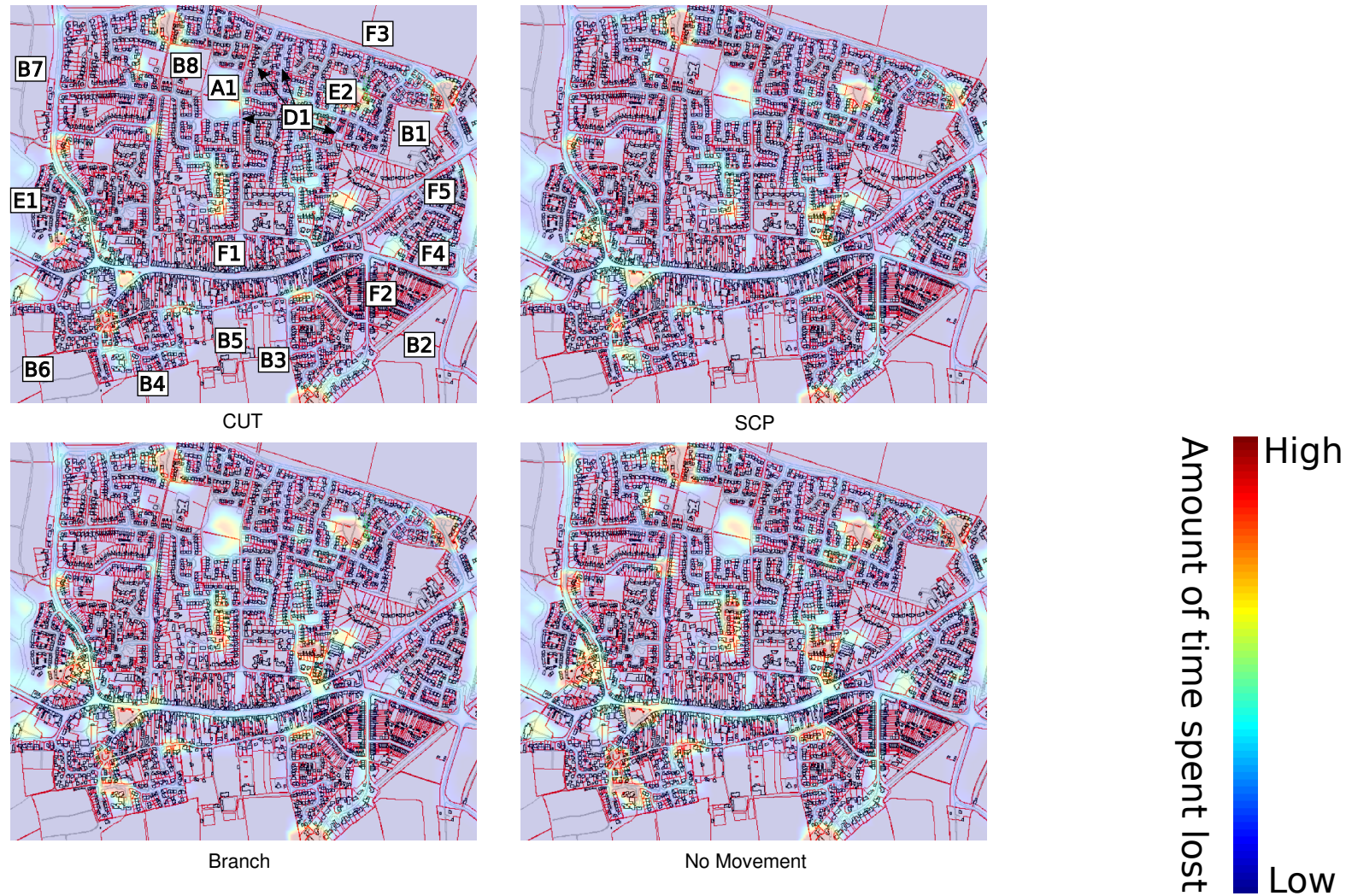


Figure 5.12: Areas of highest risk for map 3(Figure 5.3) and various speed ratios (highest risk is denoted by red). Ordnance Survey© Crown Copyright. All rights reserved.

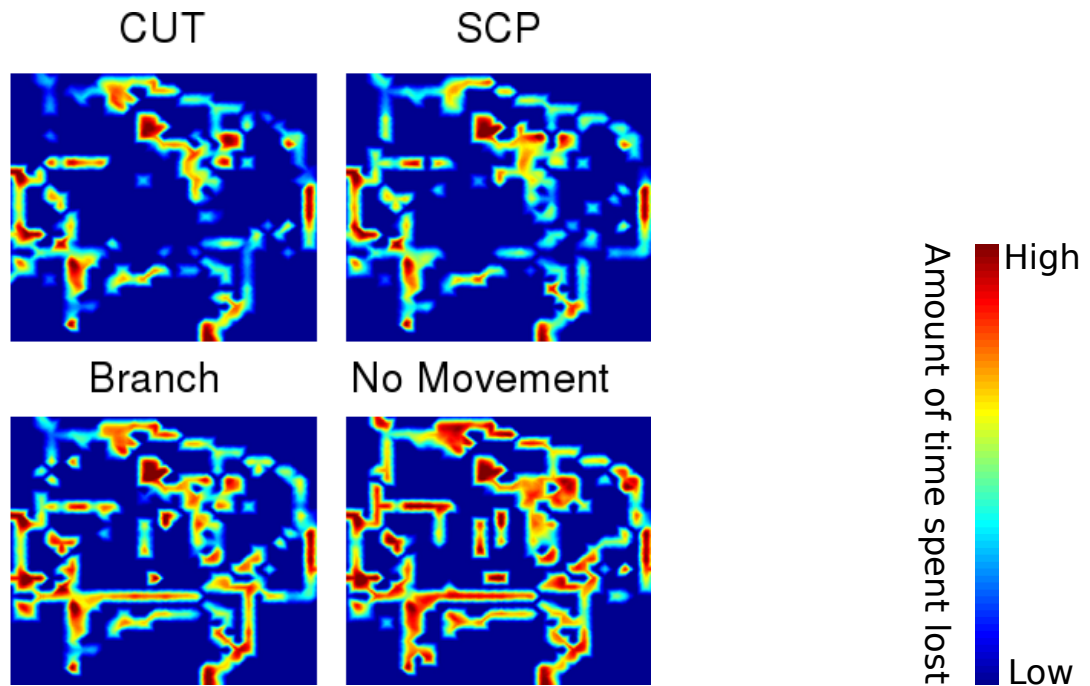


Figure 5.13: Normalised heat map of map 3.

#### Map 4

In this map the SCP algorithm again comes out slightly better at 56.4% CUT falling at 55.7%, Branch 44.6% and No Movement 39.8%.

As seen in previous maps, open areas are again a problem. Similarly to map 1 the Branch algorithm fairs worse in the open space, as show in the close up of area A1 in Figure 5.14.

This map again highlights the problem of obscured areas (labelled E), shown on the normalised heat map in Figure 5.15. The particularly notable area of C1 is also caused by an area that is effectively obscured from the road network, since the area of the road network in that area is inaccessible due to the boundary of the map.

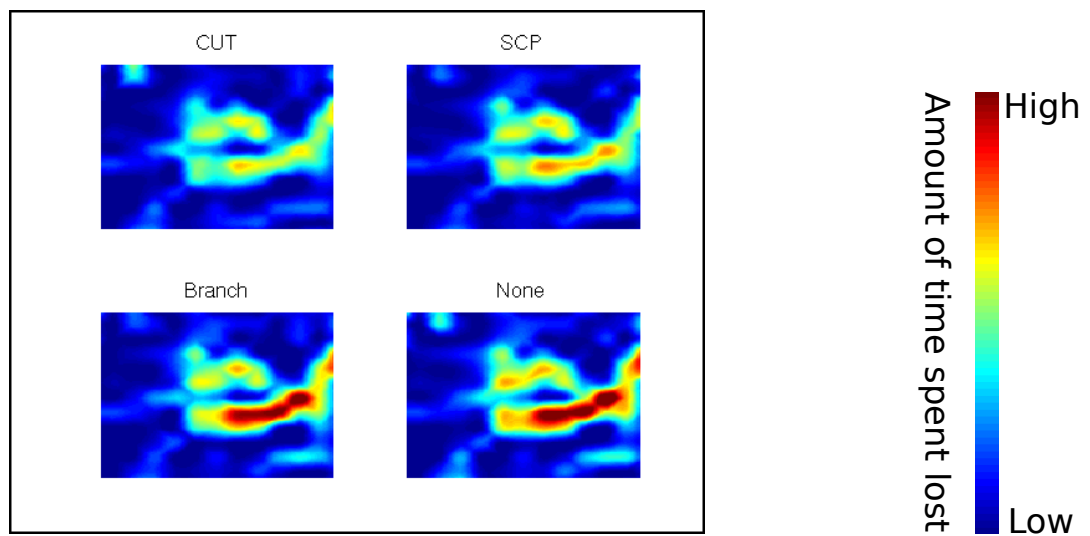


Figure 5.14: Close up of area A1 on Map 4. Colour maps have been scaled to highlight the differences in performance.

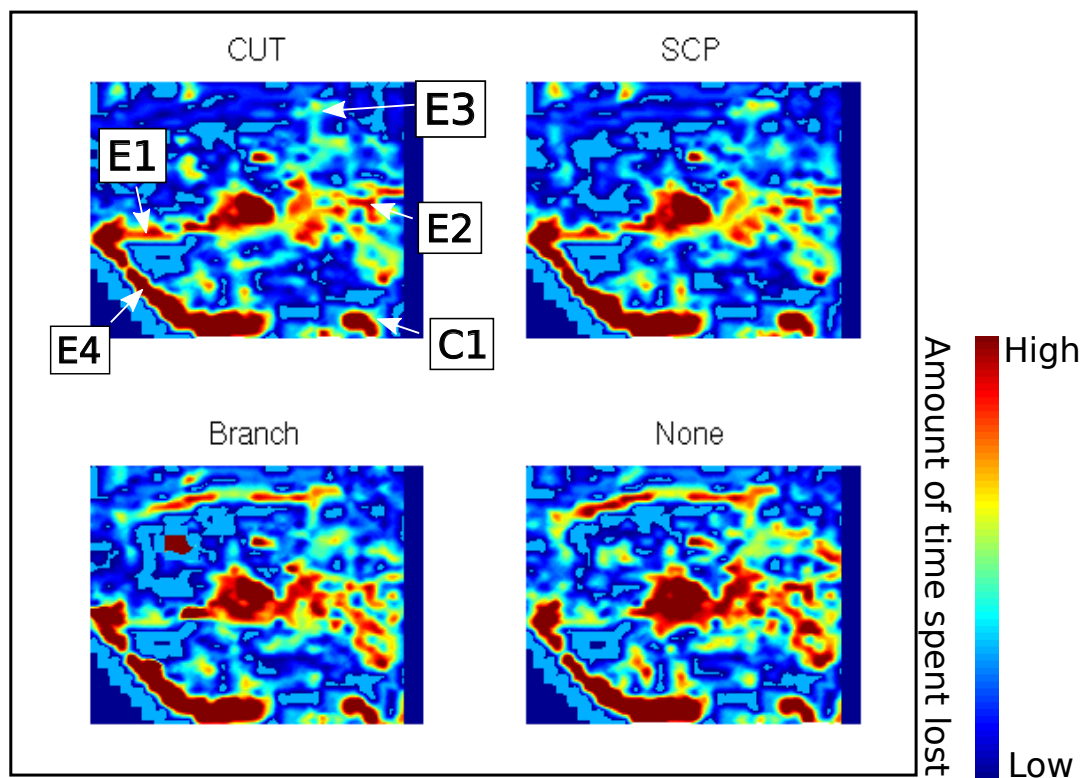


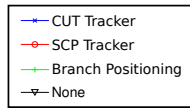
Figure 5.15: Normalised heat map of map 4 with heavily obscured areas labelled.



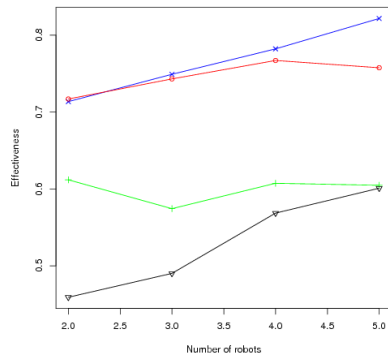
Number of Robots	CUT	SCP	Branch	None
Speed Ratio 0.31				
2	0.713	0.716	0.611	0.459
3	0.749	0.743	0.574	0.490
4	0.782	0.767	0.607	0.568
5	0.821	0.757	0.604	0.600
Speed Ratio 0.375				
2	0.681	0.706	0.567	0.458
3	0.685	0.697	0.498	0.435
4	0.732	0.710	0.522	0.500
5	0.702	0.678	0.530	0.489
Speed Ratio 0.43				
2	0.611	0.668	0.473	0.368
3	0.612	0.625	0.485	0.389
4	0.655	0.670	0.495	0.430
5	0.706	0.670	0.528	0.521
Speed Ratio 0.518				
2	0.496	0.516	0.412	0.323
3	0.519	0.535	0.462	0.379
4	0.515	0.542	0.442	0.358
5	0.561	0.512	0.438	0.397
Speed Ratio 0.75				
2	0.388	0.406	0.301	0.306
3	0.404	0.431	0.343	0.335
4	0.443	0.427	0.364	0.355
5	0.403	0.444	0.356	0.356
Speed Ratio 1.0				
2	0.253	0.300	0.253	0.228
3	0.281	0.311	0.270	0.248
4	0.315	0.348	0.291	0.255
5	0.351	0.365	0.283	0.300

Table 5.4: Effectiveness for map 4.

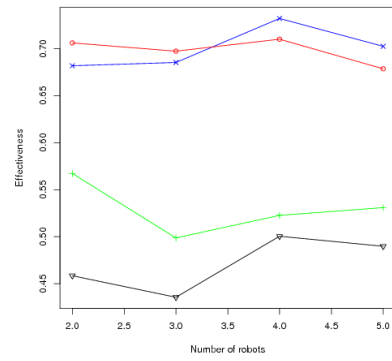
$$Effectiveness = \frac{\text{Time samples in which the target was observed}}{\text{Total simulation time}}$$



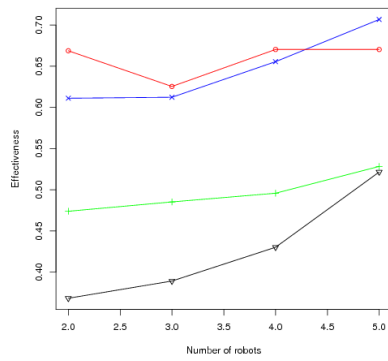
0.31



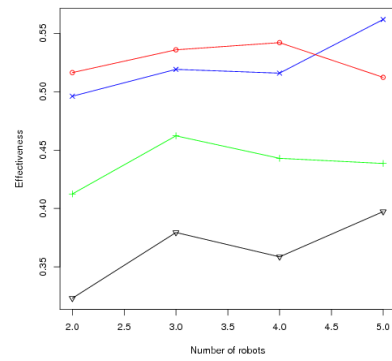
0.375



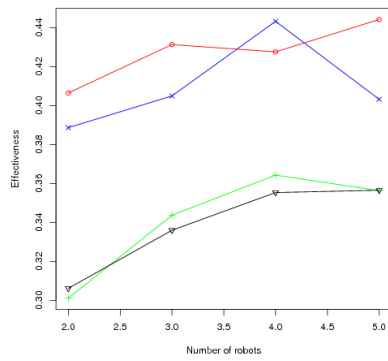
0.43



0.518



0.75



1.0

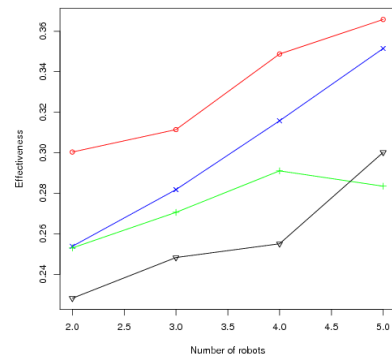
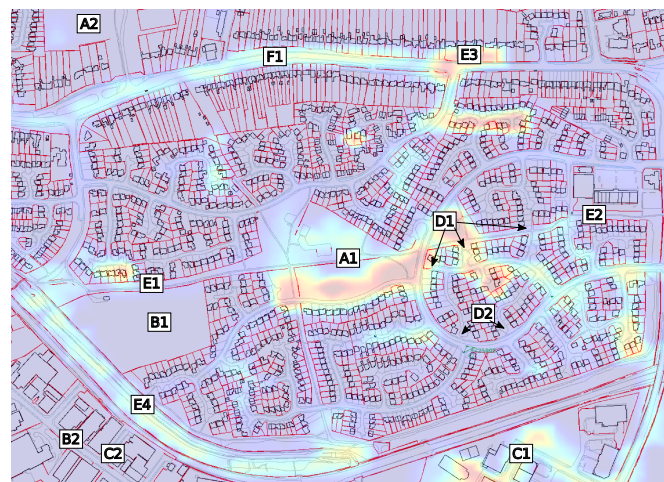
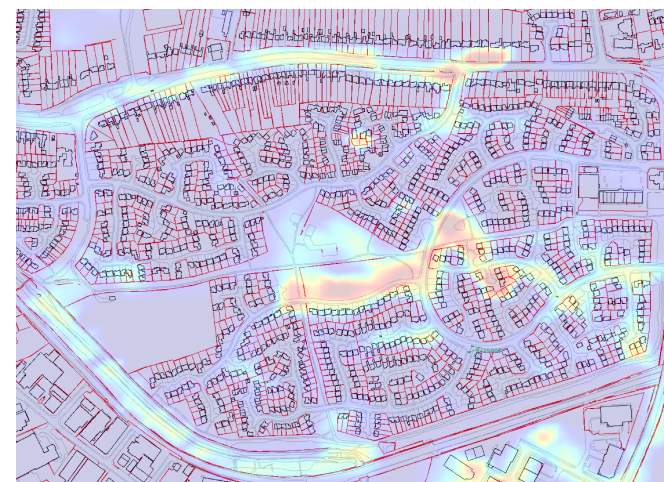


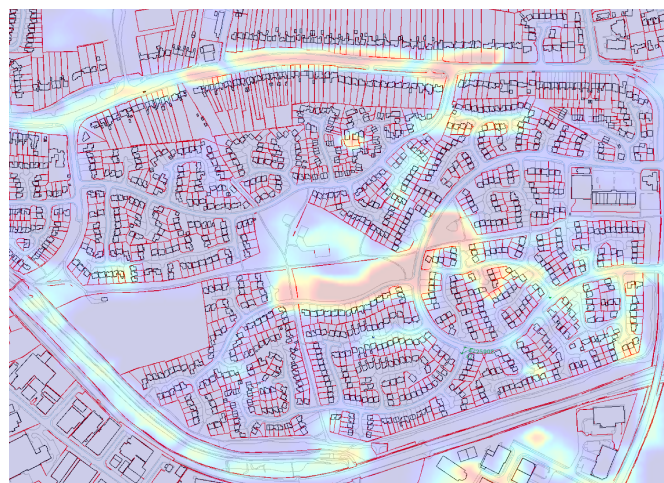
Figure 5.16: Effectiveness against number of robots for map 4. Speed relative to the robots labelled above each graph.



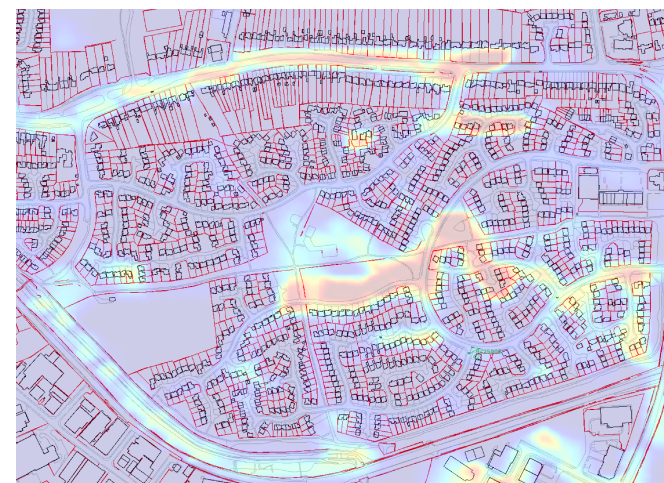
CUT



SCP



Branch



No Movement

Amount of time spent lost

High

Low

Figure 5.17: Areas of highest risk for map 4(Figure 5.4) and various speed ratios (highest risk is denoted by red). Ordnance Survey© Crown Copyright. All rights reserved.

Algorithm	Mean
CUT	0.536
SCP	0.539
Branch	0.427
None	0.353

Table 5.5: Mean effectiveness over over all trials

### Map Summary

As shown from the results so far, overall there is no significant advantage to either the CUT algorithm or the SCP algorithm, both averaging roughly the same throughout all the trials performed. This is also shown in Table 5.5, with only a 0.3% difference between the SCP and CUT algorithm overall. The Branch positioning however fared significantly worse averaging roughly 11% behind the others and on occasions falling below the no movement strategy (Figure 5.8), however averaging 9.3% above the no movement algorithm overall.

As can be seen from Figure 5.18 and Table 5.6 the algorithms are fairly substantially effected by the speed of the target relative to the robots. With the CUT and SCP experiencing a drop in effectiveness of 49.0% and 45.9% respectively between the speed ratios of 0.31 and 1.0. However both CUT and SCP remain roughly 40% effective, even with the targets only 25% slower than the robots. As the target reaches similar speeds to the robot however the effectiveness of all the algorithms seem to converge towards that of the no movement, with the better algorithms remaining only 5 – 6% better than a No Movement strategy. This shows that the target’s advantage of increased mobility and comparable speed ratios make effective tracking by the team extremely difficult.

Algorithm	Mean					
Speed Ratio	0.31	0.375	0.43	0.518	0.75	1.0
CUT	0.758	0.684	0.595	0.512	0.397	0.268
SCP	0.739	0.677	0.597	0.522	0.419	0.280
Branch	0.590	0.527	0.462	0.409	0.333	0.241
None	0.474	0.424	0.373	0.333	0.291	0.220

Table 5.6: Comparison of algorithms over all maps, comparing the speed ratio of the target relative to the robots.

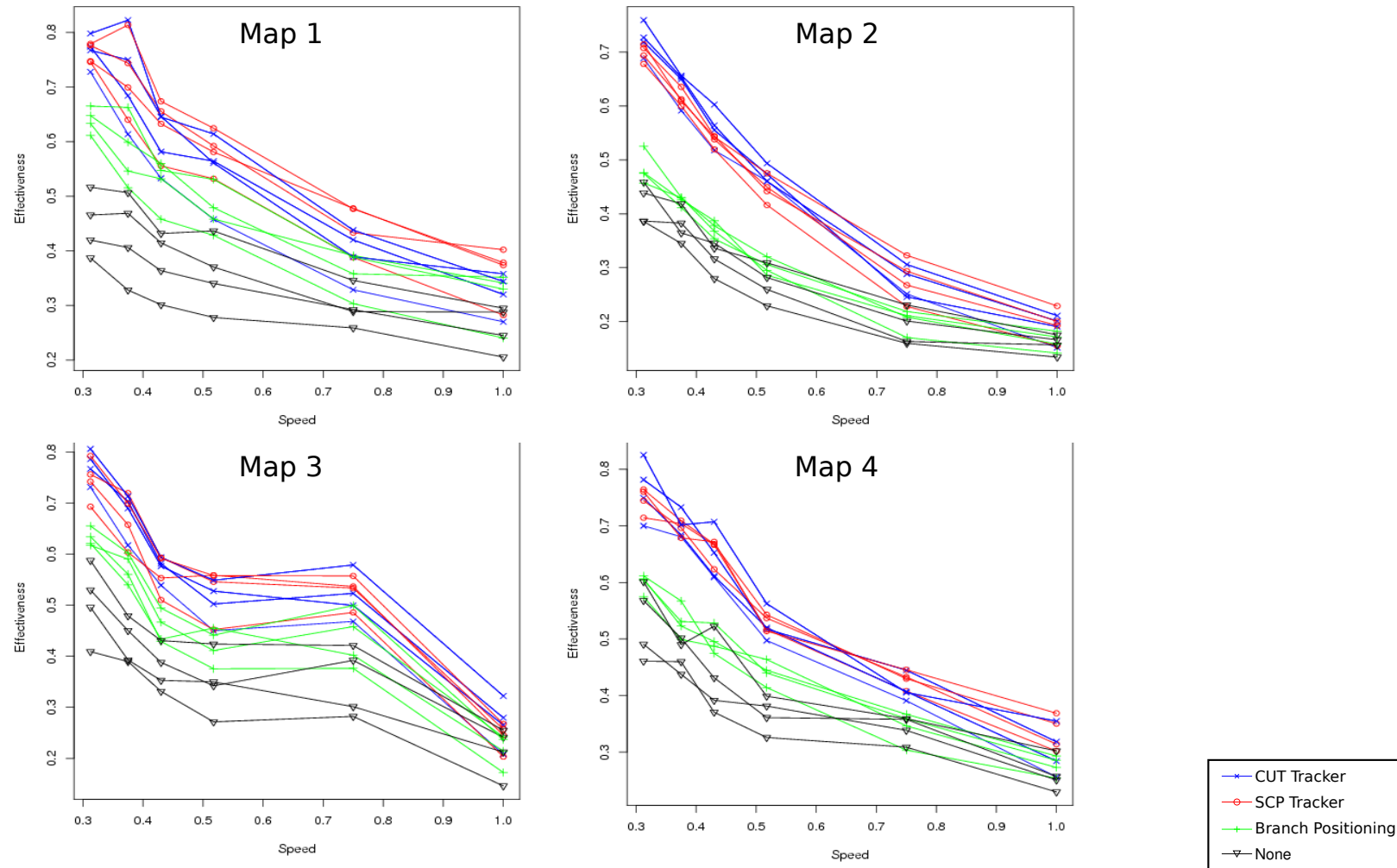


Figure 5.18: Effectiveness of algorithms against speed ratio of robot. Differing numbers of robots are shown on different lines.

### 5.3.2 Analysis by Algorithm

This section includes the results presented in the previous section however concentrates on contrasting the algorithms performance overall.

As seen, the overall trend shows that CUT and SCP have similar performances followed by the Branch positioning that fares significantly worse.

The Branch positioning is unique in being particularly bad in regions containing long relatively straight roads as shown in Figure 5.21. This is due to the Branch algorithm attempting to place itself at key places on the network. It will tend to stay at the ends of roads where junctions naturally occur, this means that no robot is being directly placed close to the target. By comparison the other algorithms have some mechanism for keeping a robot within a reasonable distance. This effect can be seen on maps 3 and 4 where a number of long straight roads caused problems particularly for the Branch algorithm (shown in Figure 5.20).

As can be seen on Map 2 where the road network that contains a significant number of shortcut paths (Figure 5.19), Branch also performs particularly badly. The algorithm merely tries to position the robots such that the target can be observed no matter what route it takes from its current location. It does not however incorporate any mechanism to make long journeys early, in order to preempt the target taking such a route. When it encounters a shortcut path it is viewed as acceptable to locate itself on the node which is closest to the target since this covers a similar amount of the network, it is however also classed as more urgent due to being closer to the target. Therefore when the target traverses this shortcut

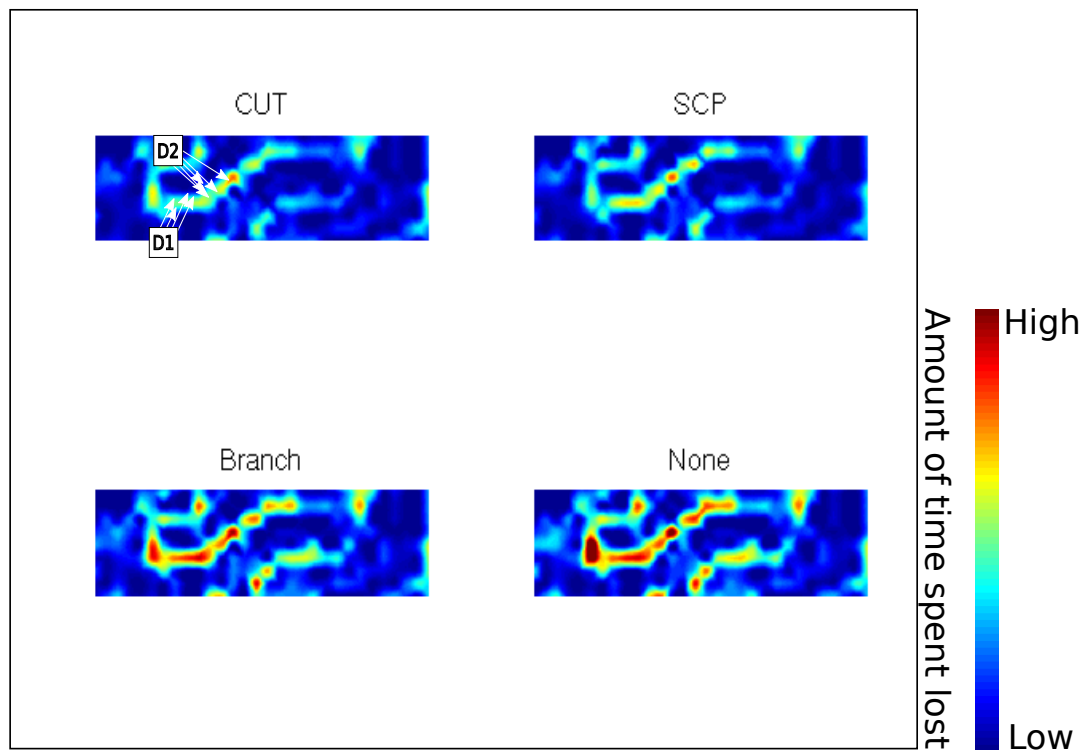


Figure 5.19: Heat map of Map 2 (Figure 5.2) illustrating Branch positions inability to cope with shortcut paths

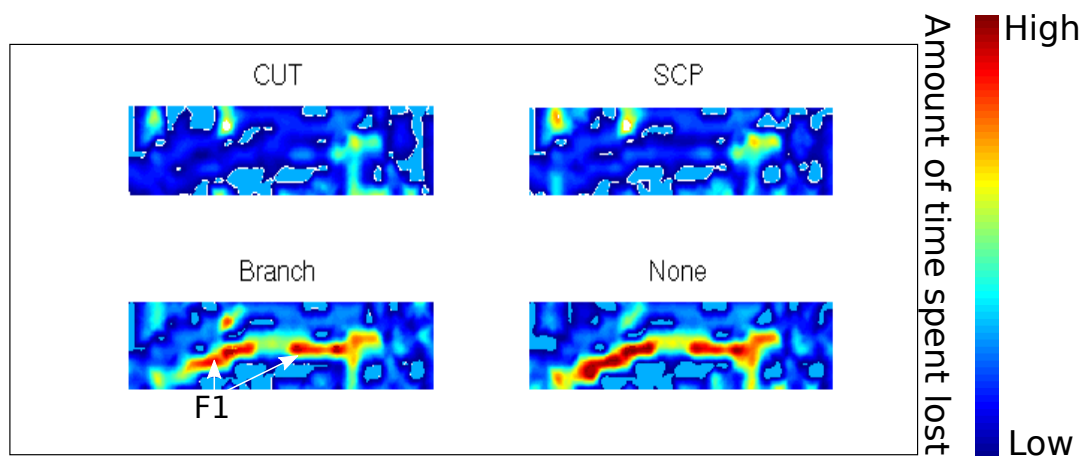


Figure 5.20: An area of Map 4 that proved particularly problematic for the Branch algorithm (Figure 5.4).



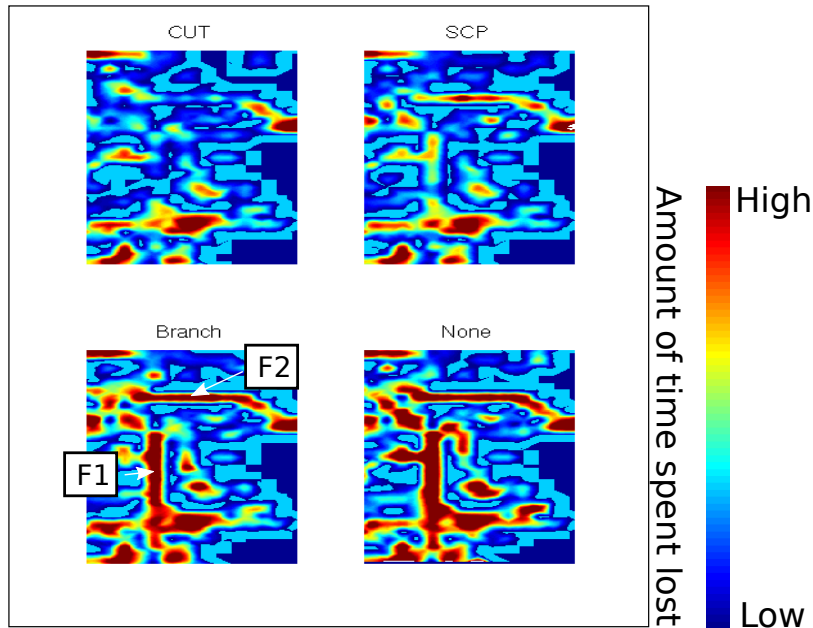


Figure 5.21: An area of Map 3 that proved particularly problematic for the Branch algorithm(Figure 5.3).

the robot finds itself unable to follow and having to make a significantly long movement in order to follow.

Figure 5.22 shows the positions at which targets became lost and then remained lost for a significant amount of time. The path in black shows the movement immediately before the target became lost (thus showing the targets trajectory, then a marker is placed at the position at which it became lost and then a line in colour for where it travelled after. The trails displayed are specifically the ones where the target got lost due to travelling through a shortcut. For clarity, some of the paths after becoming lost have been removed due to the large number of paths in the area, particularly for the Branch algorithm. The SCP and CUT algorithms do cope with these situations better, as shown by there generally being fewer instances of the targets after having travelled through such

areas. Note that again for clarity not all instances are shown however the general level of losses is representative. This can be seen in Figure 5.23 that shows the positions at which robots became lost. In areas where too many paths were present to display, or the path was not due to a shortcut, some were removed before plotting in Figure 5.22. However, all the positions are included in Figure 5.23. As can be seen, in general there is still a similar level between the CUT and SCP algorithms however significantly more from the Branch algorithm.

Comparing the SCP and CUT algorithms in Figure 5.19 in the areas of D1 and D2 both are vastly improved over the Branch and no movement strategy. However the SCP algorithm is showing little benefit over the CUT in this area despite the specific accounting for the shortcuts contained in this area. This indicates that the use of Cartesian cost maps in conjunction with the topological map is sufficient to attract robots to these areas without the need to specifically identify and account for the short cuts. Overall this shows that short cut paths are not a significant enough problem to cause the SCP algorithm specifically coded to cope with them to perform significantly better than the naive algorithm.

### **5.3.3 Comparison by Problem**

This section is mostly a summary of the previous two sections to summarise the findings.



Figure 5.22: Positions at which the targets became lost and their movements before(Black) and after becoming lost in an area of Map 2. Top: CUT algorithm, Middle: SCP algorithm, Bottom: Branch algorithm. Ordnance Survey© Crown Copyright. All rights reserved.

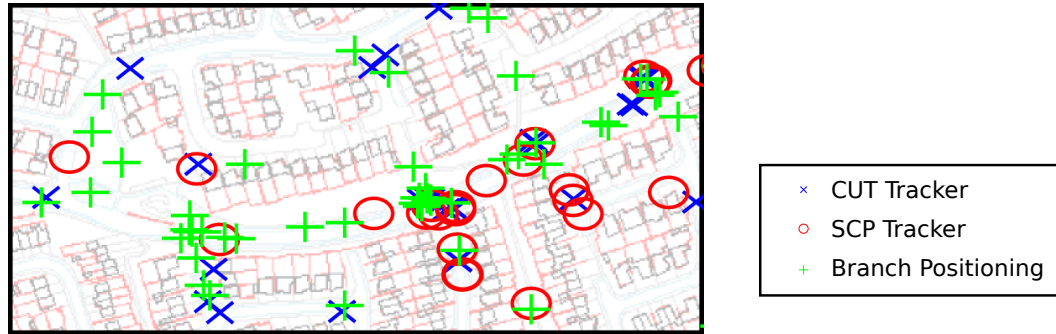


Figure 5.23: All positions at which the target became lost for the area on Map2. Ordnance Survey© Crown Copyright. All rights reserved.

### Open Spaces and Occluded Spaces

As shown these problems seem to effect all algorithms, causing them all to perform badly. However in isolated examples the Branch algorithm is particularly at a disadvantage in open spaces due to it not explicitly trying to keep a robot within visual contact of the target.

Occluded spaces have shown to be a significant remaining problem for all algorithms. This demonstrates that these are significant enough a problem that for complete coverage of the environment a variety of robots is required, including those that can explore these regions.

### Short Cut Paths

Generally the shortcut paths do not provide a significant enough challenge that the algorithms specifically designed to cope with the problem have a particular advantage. The addition of combining Cartesian and topological forces to the algorithms is sufficient to attract robots to such nodes without the explicit identification of the short cut. The approach

of CUT is comparable to the SCP algorithm in these areas. The Branch algorithm however performs significantly worse in such areas due to its algorithm viewing it as acceptable not to proactively cover such areas.

### **Long Roads**

Long roads are specifically a problem for the Branch algorithm and generally do not effect the other algorithms. This is due to the algorithm keeping the robots at fixed positions at the end of roads and not directly within line of sight of the target.

# **Chapter 6**

## **Critical Assessment and Further Work**

### **6.1 Contribution**

A particularly unique aspect of urban surveillance has been identified in this work that has previously been unexplored, namely a problem where the robots and targets are unable to traverse similar spaces. A literature review identified relevant algorithms to this domain, particularly focusing on simple algorithms that scale well with respect to the number of robots present in the system. A testing environment has then been defined in which to explore this problem. This environment is specified as an extension to a pre-existing definition used for target tracking. The environment chosen has three types of spaces, defining space that cannot be traversed, space inaccessible to the robots and free space. Although only

two types of obstacle are used in this work, the principle could easily be extended to allow for more complex forms of environment, such as allowing the target to traverse smaller obstacles such as fences or walls while preventing access to buildings. This can provide a more realistic scenario in the future.

It was also identified that the new environment necessitates changes in the existing algorithms in order to account for the restricted space in which the robots traverse. These include the frequent local maxima that are produced by a road network when using previous algorithms that have little understanding of the structure of the environment. The presence of “Short Cut Paths” was also identified that could allow targets to easily escape observation. Highly occluded areas and large open spaces are also identified. These are later shown, through testing, as one of the most problematic of all areas.

Three new algorithms were produced as a result of the observations CUT, SCP and Branch. SCP is primarily original, however it incorporates the existing Shortest Escape Path algorithm. Branch is an original algorithm.

The CUT algorithm is primarily based on existing work from (Jung, 2002, 2005) and (Parker, 2002). These were modified firstly to project the movements to work along the direction of the road network. Secondly the distance metrics used in the calculations were purely based upon a Cartesian space. These were modified to incorporate knowledge of distance using the topological road network in order to overcome the limitations

previously stated, where due to the naive forces the robots becomes stuck at certain points in the road network.

The SCP algorithm incorporated the SEP tracker for the tracking of a target in view and is built on top of a custom made algorithm for the positioning of robots. The unique aspect of this algorithm is its detection of short cut paths through which targets can escape. The team is then organised in order to account for these paths and attempt to preempt the target using such a route. It therefore classifies which routes are of highest risk and from this information positions the robots to attempt to minimise the risk. This identification of Short Cut Paths, incorporation into the Short Cut Graph, and use in calculating the movements of the robots is particularly unique to this work.

The Branch algorithm is then developed as an entirely new algorithm based on the concept of placing robots at key positions on the road network in order to be best positioned for future movement. This again uses the novel Short Cut Path and Short Cut Graph in its calculation of the best position to move to.

A simulation has been developed based upon the Player/Stage (Player, 2008) platform. This was used to test the three algorithms and the baseline comparison in a variety of environments and configurations. Of the three algorithms tested, the CUT and SCP algorithms perform reasonably comparably. Over all there is less than 0.3% difference in effectiveness between these two algorithms: SCP achieving 53.9% and CUT 53.6%. The branch algorithm however fares particularly worse at an average of 43.8% overall.



There was the initial observation that targets escape through shortcut paths which lead to the specific modification of algorithms to account for them. However as shown, due to high risk shortcuts generally being closer, in a Cartesian sense, this is enough for the CUT algorithm to attract robots to the area, and thus the specific coding does not produce enough benefit for a significant improvement to be seen overall.

Finally the outstanding issues that remain problematic have been identified. Mostly the highly occluded areas of the map and the large open spaces. In such areas it would probably be impossible to resolve the issues due to much of the limitation being due to the robots restricted movement. This therefore shows that, for a comprehensive tracker across the whole environment, a variety of sensors are needed and that road based sensors would need to be supplemented in such areas.

## **6.2 Critical Assessment**

There are a number of factors that would be present in real life that would affect an actual implementation and that were not possible to incorporate into the test environment. These factors will have an impact on the algorithms effectiveness and may require them to be modified. However the simulation is still viewed as an accurate representation of reality.

## **6.2.1 Environment Construction**

### **Additional Agents**

Additional traffic to the road is likely to be a significant factor, the addition of which is a natural continuation of this work. Agents present that are not directly involved in the scenario are likely to significantly impact the robots ability to manoeuvre but also provide dynamic occlusions to the field of view. This could incorporate both pedestrian and road traffic. Rules of the road such as speed limits, one way streets and road lanes are also factors that would be a natural addition to the current simulation model.

### **Target Ability**

The intelligence of the target is a factor that will significantly impact upon the robots ability to track the target. This work was stated as non-adversarial and thus this was not addressed. However inclusion of this intelligence is a natural extension to the work.

### **Environmental Parameters**

Due to limited memory the environments were constructed with bounds to the environment. Although there will always be a requirement for bounds to any environment, larger areas may give the target more freedom and thus impact on the overall performance.

Latency and data loss within the system was also not taken into account particularly with respect to communications.

The prior knowledge of the targets starting position is an element that benefits the robots that could also be removed to make the scenario more realistic.

Simulations were also kept at a fixed duration. This decision was taken again in order to keep the number of trials to an acceptable level. However longer simulations may well show issues between the algorithms ability to run indefinitely. Longer trials should potentially show that certain algorithms may be adept at tracking a target that is in view. However, once lost is not adept at re-acquiring it. Short trials would favour such an algorithm by limiting the amount of time a target can spend permanently lost.

The number of robots was limited due to the amount of processing power available. Due to the effectiveness tending towards 100% at lower speeds, increasing the number of robots will only have a small impact on these results. Particularly due to the fact that the most significant areas are those not visible from the road network, a problem that cannot be addressed with numbers. However, it is likely to see increases in effectiveness at higher target speeds, as shown in the results, at higher target speeds the effectiveness is far more dependant on robot numbers. Ultimately as numbers increase the target will rarely be lost while visible from the road network, meaning the effectiveness will converge towards those seen at lower speeds where the target is also rarely lost.

## **Maps**

An accurate road map is assumed to be available. This type of data is readily available however unexpected changes to the road network could be present such as road works or minor inaccuracies to the data.

The impact of 3 dimensions is also likely to impact the robots ability to maintain a view of a target. Due to the limitations of the environment obstructions had to be classified as purely occluding or non-occluding to all sensors. There was no representation of how the obstacles blocked the sensors and if they only partially blocked the sensor view such as low walls.

### **6.2.2 Robot construction**

A number of simplifications were also made in the specification of the robots. A localisation system with low noise is assumed to be present. This was viewed as acceptable since many localisation systems using GPS are accurate to less than 1m as well as allowing localisation and navigation in urban environments. However taking into account the effects of noisy sensors is also a possibility.

The target identification system is also assumed to be a perfect system in this work and thus could be revised to represent a more realistic system which fails to identify and also possibly produces false positive matches. Noise could also be applied to the laser and sonar sensors as well as a more limited range.

Inter-robot communications are currently assumed to be of unlimited bandwidth and range. Limitations upon these devices could also be explored within the context of this problem.

The robots were constructed using a simple differential drive model, a more complex system such as car like drive system could be incorporated to see how this effects the robots ability. Additionally only the speed of the robots was varied in the trials, other dynamics such as the robots acceleration and turning speed could also be varied.

### **6.2.3 Data Limitations**

The data provided by Ordnance Survey has a few limitations that affects the simulations representation of reality. These are viewed as relatively minor, however they should be noted. The data used in the simulations is quoted as accurate to 1m and includes features greater than a few meters in any dimension. This is a possible limitation as to its applicability to real life. It is however unlikely that this level of accuracy will greatly affect the robots as objects of such small dimensions are rarely present on the road network and would also provide little obstruction to the target or opportunity for occlusion. As previously noted the data does not have any description of the visual appearance of objects. This is particularly important for obstacles that may be visually transparent such as chain link fences, railings or low walls. It was assumed that these are opaque in these simulations. This naturally impedes the robots as more occlusions make their task harder. One would therefore assume that including this

information would produce a corresponding increase in efficiency.

#### **6.2.4 Control Problem**

A number of issues were experienced in the running of the simulations. These were mainly due to the large amount of processing power and memory required in order to simulate a large environment and multiple robots. This then led to control problems, as the server and client side has trouble keeping their messages synchronised as one may become starved of processing power. In this situation, the control of the robot becomes difficult since, if the algorithms side becomes starved of processor time, the frequency of the control loop is reduced thus making tasks like obstacle avoidance difficult. This could then cause robots to crash during the periods in which control was reduced. This was minimised by reducing the simulation speed and memory usage. However it still had an impact upon the results particularly when simulating 5 robots. The effects of this, however, can be noticed as some of the graphs show decreasing efficiencies relative to the number of robots. This appears to be due to the ability of the simulation to keep all robots functional.

### **6.3 Further Work**

This work has attempted to provide a platform in which to progress towards producing a system that is capable of autonomous surveillance in an urban environment. Obviously there are many steps in between

this work and actually achieving this. Many of the challenges are hardware related such as producing autonomous platforms that are capable of traversing the road network autonomously. These technologies are being explored and should be available in the near future.

### **6.3.1 Sensor Noise and Uncertainty**

Modifying the simulations to include a measure of uncertainty in the sensors readings or the possibility of mistaken or missed identification by the target identification system is a fairly natural progression of the current simulation. For instance the camera system was assumed to identify the target with 100% accuracy if it was within the field of view. This is obviously not possible with a real system.

### **6.3.2 Additional Agents**

The most interesting work would be to include additional agents that are not directly involved in the scenario, such as general traffic and pedestrians. The impact of these agents on the ability of a team to perform a surveillance task appears to be unexplored. The main limiting factor that prevented the inclusion of such agents in this work was the limitations of memory and processing power. However in the future as more processing power, or a more efficient simulation implementation becomes available, exploring this area would be possible.

### **6.3.3 Additional Robots**

Currently the algorithms also have a large scope for improvement particularly in the areas identified as troublesome. The particular problem area of large open spaces and areas that are occluded to the road network could also be addressed by potentially integrating robots of different capabilities into the simulation such as cooperative air and ground surveillance (Grocholsky et al., 2006).



# Chapter 7

## Conclusions

This work has explored the problem of target tracking in an urban environment. The motivation for this work is to increase safety to personnel, specifically for military operations in urban environments where visibility is extremely limited. By providing methods to track targets in urban environments, this provides more complete information about the location of objects of interest to the user. Allowing the user to make more informed plans, reducing the uncertainty and risk involved.

An environment in which to explore target tracking in an urban environment was defined and a simulation for this environment developed. A literature review of algorithms relevant to the task of tracking a target in an urban environment, specifically looking at algorithms that scale well with the number of robots taking part in the exercise has been performed. Three algorithms were developed for the task: CUT, SCP and Branch. These are based on existing algorithms and incorporating measures to account for the challenges of this new environment: notably the highly restricted space that the robots traverse as well as the inability of the

robots to follow the target in a similar manner to previous algorithms; the newly identified problem of short cut paths was also accounted for. These algorithms were tested and their effectiveness obtained in a number of environments using real map data based on urban environments in the UK. The general results show that the SCP and CUT algorithms perform reasonably comparably while the Branch algorithm performs significantly worse. Additionally that the short cut path problem does not seem to present a significant enough challenge that the specific coding for these situations gives the algorithm any significant advantage over the naive algorithms. The particular places within the environment in which these algorithms have trouble have been identified, generally consisting of open spaces and areas significantly occluded from the road network and, in the case of the Branch algorithm, particularly long straight roads. It has been shown that the remaining areas of significant difficulty are mostly areas that are not visible from the road network and that future work should be focused towards addressing such areas, as well as making the simulation a progressively more realistic scenario through additions such as non-participatory agents and more realistic target movement.

The contributions of this thesis have been as follows:

- Definition of the environment in which to explore target tracking in urban environments.
- Literature review of relevant algorithms to the task of tracking a target in an urban environment.
- Identification of particularly difficult aspects of urban environments

that make road based surveillance difficult.

- ✧ Highly restricted robot movement
  - ✧ Frequent local minima
  - ✧ Short Cut Paths
  - ✧ Highly Occluded areas
  - ✧ Open spaces
- Development of 3 algorithms for the tracking of a target in an urban environment
    - ✧ SCP algorithm
      - ❑ Specifically accounts for the situation of short cut paths.
      - ❑ Identifies areas of the map in which a short cut path exists and incorporates this knowledge into its planning.
      - ❑ Attempts to cover the paths that present the highest risk then take preemptive movement.
    - ✧ CUT algorithm
      - ❑ Based on a combination of two existing algorithms A-CMOMMT and Jung.
      - ❑ Both algorithms were modified to be suitable for a road network by incorporating additional forces that act along the road network, as well as projecting the existing forces onto the road network.

- ❑ Due to the nature of the environment both topological and Cartesian distances were incorporated into the algorithms to account for the fact that the robots essentially traverse a topological space and the targets traverse a Cartesian space.
- ✧ Branch Algorithm
  - ❑ Attempts to place robots at key areas of the road network such that an robot cannot leave the current area without being observed.
  - ❑ Also incorporates shortcut paths.
- Simulation, evaluation and comparison of the algorithms.
  - ✧ Overall the SCP and CUT algorithms perform comparably:
    - ❑ Only 0.3% difference between the two. Both achieving roughly a 53% – 54% effectiveness overall.
    - ❑ At lower target speed ratios SCP and CUT achieve 74% and 76% effectiveness respectively falling to 42% and 40% at higher speed ratios and then 28% and 27% at the same speed as the target .
  - ✧ with the Branch algorithm performing significantly worse
    - ❑ Achieved a 43% effectiveness overall.
    - ❑ At lower speed ratios seeing an effectiveness of 59% falling to 33% at the higher speed ratios and 24% at the same speed as the target.

- Identification of outstanding issues and the type of areas that remain problematic for the algorithms developed.
  - ✧ Primarily the need to account for large and occluded areas either by blocking the exits to such areas and thus keeping the target in a known area even though not observed or calling in a more capable robot.

# References

- Bailey, T. and Durrant-Whyte, H. (2006). Simultaneous localization and mapping (slam): part ii. *Robotics & Automation Magazine, IEEE*, 13(3):108–117.
- Borenstein, J., Everett, H. R., Feng, L., and Wehe, D. (1997). Mobile robot positioning: Sensors and techniques. *Journal of Robotic Systems*, 14(4):231–249.
- Borenstein, J. and Koren, Y. (1991). The vector field histogram-fast obstacle avoidance for mobile robots. *Robotics and Automation, IEEE Transactions on*, 7(3):278–288.
- Brooks, R. (1986). A robust layered control system for a mobile robot. *Robotics and Automation, IEEE Journal of*, 2(1):14–23.
- Cardei, M. and Wu, J. (2006). Energy-efficient coverage problems in wireless ad-hoc sensor networks. *Computer Communications*, 29(4):413–420.
- Chakrabarty, K., Member, S., Iyengar, S. S., Qi, H., and Cho, E. (2002). Grid coverage for surveillance and target location in distributed sensor networks. *IEEE Transactions on Computers*, 51:1448–1453.

- Cheng, P. (2003). A short survey on pursuit-evasion games. Technical report, Department of Computer Science, University of Illinois at Urbana-Champaign.
- Collins, R., Lipton, A., Fujiyoshi, H., and Kanade, T. (2001). Algorithms for cooperative multisensor surveillance. *Proceedings of the IEEE*, 89(10):1456–1477.
- Condor (2009). Condor high throughput computing. [online] accessed 3rd march 2009. url <http://www.cs.wisc.edu/condor>.
- Connell, J. (1992). Sss: a hybrid architecture applied to robot navigation. In *Robotics and Automation, 1992. Proceedings., 1992 IEEE International Conference on*, pages 2719–2724 vol.3.
- Culler, D., Estrin, D., and Srivastava, M. (Aug. 2004). Guest editors’ introduction: Overview of sensor networks. *Computer*, 37(8):41–49.
- Durrant-Whyte, H. and Bailey, T. (2006). Simultaneous localization and mapping: part i. *Robotics & Automation Magazine, IEEE*, 13(2):99–110.
- ELRT (2008a). European land-robot trial, camp security trial requirements. [online] accessed 27th december 2008. url <http://www.elrob.eu/files/cs.pdf>.
- ELRT (2008b). European land-robot trial. [online] accessed 7th may 2008. url <http://www.elrob.org>.
- Feyrer, S. and Zell, A. (1999). Detection, tracking, and pursuit of humans with an autonomous mobile robot. In *Intelligent Robots and Systems*,

1999. *IROS '99. Proceedings. 1999 IEEE/RSJ International Conference on*, volume 2, pages 864–869 vol.2.
- Filliat, D. and Meyer, J.-A. (2003). Map-based navigation in mobile robots:: I. a review of localization strategies. *Cognitive Systems Research*, 4(4):243–282.
- Foresti, G., Micheloni, C., Snidaro, L., Remagnino, P., and Ellis, T. (2005). Active video-based surveillance system: the low-level image and video processing techniques needed for implementation. *Signal Processing Magazine, IEEE*, 22(2):25–37.
- Frew, E. (10-14 April 2007). Cooperative standoff tracking of uncertain moving targets using active robot networks. *Robotics and Automation, 2007 IEEE International Conference on*, pages 3277–3282.
- Fuentes, L. M. and Velastin, S. A. (2006). People tracking in surveillance applications. *Image and Vision Computing*, 24(11):1165–1171. Performance Evaluation of Tracking and Surveillance.
- Gage, D. (1992). Sensor abstractions to support many-robot systems. *Proceedings of SPIE Mobile Robots VII*, Volume 1831:pp 235–246.
- Gerkey, B. P., Thrun, S., and Gordon, G. (2006). Visibility-based pursuit-evasion with limited field of view. *Int. J. Rob. Res.*, 25(4):299–315.
- Gerkey, B. P., Vaughan, R. T., and Howard, A. (2004). Player version 1.5 user manual. [online] accessed 14th nov 2009. url <http://playerstage.sourceforge.net/doc/player-manual-1.5.ps.gz>.



- Gonzalez-Banos, H. C.-Y. L. L. J.-C. (2002). Real-time combinatorial tracking of a target moving unpredictably among obstacles. *Robotics and Automation, 2002. Proceedings. ICRA '02. IEEE International Conference on*, 2:1683–1690 vol.2.
- Grocholsky, B. P., Keller, J., Kumar, V., and Pappas, G. (2006). Cooperative air and ground surveillance. *IEEE Robotics & Automation Magazine*, 13(3):16 – 25.
- Guibas, L. J., Latombe, J.-C., LaValle, S. M., Lin, D., and Motwani, R. (1999). A visibility-based pursuit-evasion problem. *International Journal of Computational Geometry and Applications*, 9(4/5):471–.
- Hahnel, D., Burgard, W., Fox, D., and Thrun, S. (2003). An efficient fast-slam algorithm for generating maps of large-scale cyclic environments from raw laser range measurements. In *Intelligent Robots and Systems, 2003. (IROS 2003). Proceedings. 2003 IEEE/RSJ International Conference on*, volume 1, pages 206–211 vol.1.
- Haritaoglu, I., Harwood, D., and Davis, L. (2000). W4: Real-time surveillance of people and their activities. 22(8):809–830.
- Hegazy, T. A. (2004). *A Distributed Approach to Dynamic Autonomous Agent Placement for Tracking Moving Targets with Application to Monitoring Urban Environments*. PhD thesis, School of Electrical and Computer Engineering Georgia Institute of Technology.
- IAIS (2008). 3dls-k continuously rotating 3d laser scanner. marketing material. [online] accessed 29th october 2008. url [http://www.3d-scanner.net/datasheet/3dls\\_flyer\\_kont\\_eng.pdf](http://www.3d-scanner.net/datasheet/3dls_flyer_kont_eng.pdf).

- Isler, V., Kannan, S., and Khanna, S. (2005). Randomized pursuit-evasion in a polygonal environment. *IEEE Transactions on Robotics*, 5(21):864–875.
- Javaclient (2008). Javaclient for player/stage. [online] accessed 7th may 2008. url <http://java-player.sourceforge.net/>.
- Jung, B. (2005). *Cooperative target tracking using mobile robots*. PhD thesis, University of Southern California, Los Angeles, CA.
- Jung, B. (November 2002). Tracking targets using multiple robots: The effect of environment occlusion. *Autonomous Robots*, 13:191–205(15).
- Kammel, S., Ziegler, J., Pitzer, B., Werling, M., Gindele, T., Jagzent, D., Schröder, J., Thuy, M., Goebel, M., von Hundelshausen, F., Pink, O., Frese, C., and Stiller, C. (2008). Team annieway’s autonomous system for the 2007 darpa urban challenge. *J. Field Robot.*, 25(9):615–639.
- Kolling, A. and Carpin, S. (2007). The graph-clear problem: definition, theoretical properties and its connections to multirobot aided surveillance. *Intelligent Robots and Systems, 2007. IROS 2007. IEEE/RSJ International Conference on*, pages 1003–1008.
- Lee, C.-Y., Gonzalez-Banos, H., and Latombe, J.-C. (2-5 Dec. 2002). Real-time tracking of an unpredictable target amidst unknown obstacles. *Control, Automation, Robotics and Vision, 2002. ICARCV 2002. 7th International Conference on*, 2:596–601 vol.2.
- Leonard, J., Barrett, D., How, J., Teller, S., Antone, M., Campbell, S., Epstein, A., Fiore, G., Fletcher, L., Frazzoli, E., Huang, A., Jones, T.,

- Koch, O., Kuwata, Y., Mahelona, K., Moore, D., Moyer, K., Olson, E., Peters, S., Sanders, C., Teo, J., and Walter, M. (2007). Team MIT urban challenge technical report. Technical report, Massachusetts Institute of Technology.
- Lipton, A., Fujiyoshi, H., and Patil, R. (1998). Moving target classification and tracking from real-time video. In *Applications of Computer Vision, 1998. WACV '98. Proceedings., Fourth IEEE Workshop on*, pages 8–14.
- Lipton, A., Heartwell, C., Haering, N., and Madden, D. (2003). Automated video protection, monitoring & detection. *Aerospace and Electronic Systems Magazine, IEEE*, 18(5):3–18.
- Low, K. H., Leow, W. K., and Ang, J. M. H. (2002). A hybrid mobile robot architecture with integrated planning and control. In *AAMAS '02: Proceedings of the first international joint conference on Autonomous agents and multiagent systems*, pages 219–226, New York, NY, USA. ACM.
- Lubecke, V., Boric-Lubecke, O., Host-Madsen, A., and Fathy, A. (3-8 June 2007). Through-the-wall radar life detection and monitoring. *Microwave Symposium, 2007. IEEE/MTT-S International*, pages 769–772.
- MasterMap (2008). Os mastermap topography layer technical specification annexe d v1.6. Â© crown copyright.
- Maurin, B., Masoud, O., and Papanikolopoulos, N. (March 2005). Tracking all traffic: computer vision algorithms for monitoring vehicles, individuals, and crowds. *Robotics & Automation Magazine, IEEE*, 12(1):29–36.

- MaxBotix (2005). Lv-maxsonar -ez4 high performance sonar range finder datasheet. copyright 2005 - 2007.
- Meguerdichian, S., Koushanfar, F., Potkonjak, M., and Srivastava, M. (2001). Coverage problems in wireless ad-hoc sensor networks. *INFO-COM 2001. Twentieth Annual Joint Conference of the IEEE Computer and Communications Societies. Proceedings. IEEE*, 3:1380–1387 vol.3.
- Meguro, J., Hashizume, T., Takiguchi, J., and Kurosaki, R. (2005). Development of an autonomous mobile surveillance system using a network-based rtk-gps. In *Robotics and Automation, 2005. ICRA 2005. Proceedings of the 2005 IEEE International Conference on*, pages 3096–3101.
- Meyer, J.-A. and Filliat, D. (2003). Map-based navigation in mobile robots:: Ii. a review of map-learning and path-planning strategies. *Cognitive Systems Research*, 4(4):283–317.
- MOD (2006). Mod grand challenge. [online] accessed 7th may 2008. url <http://www.challenge.mod.uk/>.
- Oh, S., Schenato, L., Chen, P., and Sastry, S. (2007). Tracking and coordination of multiple agents using sensor networks: System design, algorithms and experiments. *Proceedings of the IEEE*, 95(1):234–254.
- Parker, L. E. (1999). Cooperative robotics for multi-target observation. *Intelligent Automation and Soft Computing*, 5:5–19.
- Parker, L. E. (2002). Distributed algorithms for multi-robot observation of multiple moving targets. *Auton. Robots*, 12(3):231–255.

- Player (2008). [online] accessed 7th may 2008. url <http://playerstage.sourceforge.net/>.
- Poduri, S. and Sukhatme, G. S. (2004). Constrained coverage for mobile sensor networks. In *IEEE International Conference on Robotics and Automation*, pages 165–171.
- Reif, J. H. and Wang, H. (1999). Social potential fields: a distributed behavioral control for autonomous robots. *Robotics and Autonomous Systems*, 27:171–194.
- Reisman, P., Mano, O., Avidan, S., and Shashua, A. (14-17 June 2004). Crowd detection in video sequences. *Intelligent Vehicles Symposium, 2004 IEEE*, pages 66–71.
- Rybski, P., Papanikolopoulos, N., Stoeter, S., Krantz, D., Yesin, K., Gini, M., Voyles, R., Hougen, D., Nelson, B., and Erickson, M. (2000). Enlisting rangers and scouts for reconnaissance and surveillance. *Robotics & Automation Magazine, IEEE*, 7(4):14–24.
- SensComp (2003). 6500 series ranging module. datasheet.
- Shi, X., Hu, J., Xu, Y., and Song, J. (2005). Architecture and simulation of sensor network system for urban surveillance. *Robotics and Biomimetics (ROBIO). 2005 IEEE International Conference on*, pages 640–645.
- Sibley, G. T., Rahimi, M. H., and Sukhatme, G. S. (2002). Robomote: A tiny mobile robot platform for large-scale sensor networks. In *Proceedings of the IEEE International Conference on Robotics and Automation*, Washington, DC, USA.

- SICK AG (2006). LMS200/211/221/291 laser measurement systems. datasheet. copyright SICK AG.
- Siebel, N. T. (2003). *Design and Implementation of People Tracking Algorithms for Visual Surveillance Applications*. PhD thesis, Department of Computer Science, The University of Reading, Reading, UK.
- Siebel, N. T. and Maybank, S. (2002). Fusion of multiple tracking algorithms for robust people tracking. In Heyden, A., Sparr, G., Nielsen, M., and Johansen, P., editors, *Proceedings of the 7th European Conference on Computer Vision (ECCV 2002), København, Denmark*, volume IV, pages 373–387.
- Simmons, R., Goodwin, R., Haigh, K. Z., Koenig, S., and O’Sullivan, J. (1997). A layered architecture for office delivery robots. In *Proceedings of the First International Conference on Autonomous Agents, Marina del Rey*, pages 245–252. ACM Press.
- Spletzer, J. R. and Taylor, C. J. (2003). Dynamic sensor planning and control for optimally tracking targets. *I. J. Robotic Res.*, 22(1):7–20.
- Stone, P., Beeson, P., Meriçli, T., and Madigan, R. (2007). Austin robot technology. Technical report, DARPA Urban Challenge.
- Sukhatme, G. S., Montgomery, J. F., and Vaughan, R. T. (2001). Experiments with aerial-ground robots. In Balch, T. and Parker, L., editors, *Robot Teams: From Diversity to Polymorphism*. AK Peters.
- Sukkarieh, S., Nebot, E., and Durrant-Whyte, H. (1999). A high in-

- tegrity imu/gps navigation loop for autonomous land vehicle applications. *Robotics and Automation, IEEE Transactions on*, 15(3):572–578.
- VFDFLA (2009). Vector field diagram and field line applet. [online] accessed 2nd january 2009. url <http://web.mit.edu/jbelcher/www/java/vecnodyncirc/vecnodyncirc.html>.
- Voth, D. (2004). A new generation of military robots. *Intelligent Systems, IEEE*, 19(4):2–3.
- Wang, G., Cao, G., and Porta, T. F. L. (2006). Movement-assisted sensor deployment. *IEEE Transactions on Mobile Computing*, 5(6):640–652.
- Wilhelm, T., Bohme, H. J., and Gross, H. M. (2004). A multi-modal system for tracking and analyzing faces on a mobile robot. *Robotics and Autonomous Systems*, 48(1):31–40. European Conference on Mobile Robots (ECMR '03).
- Yamauchi, B. (2004). Packbot: A versatile platform for military robotics. In *Proceedings of SPIE 5422*, pages 228–237.
- Zhang, Y., Wu, H., Cheng, X., and Liu, C. (2008). Accuracy evaluation of three dimensional laser range scanner based on field calibration. *8th International Symposium on Spatial Accuracy Assessment in Natural Resources and Environmental Sciences*, pages 119–126.

# **Appendix A**

## **Heat Maps**



## A.1 Map 1

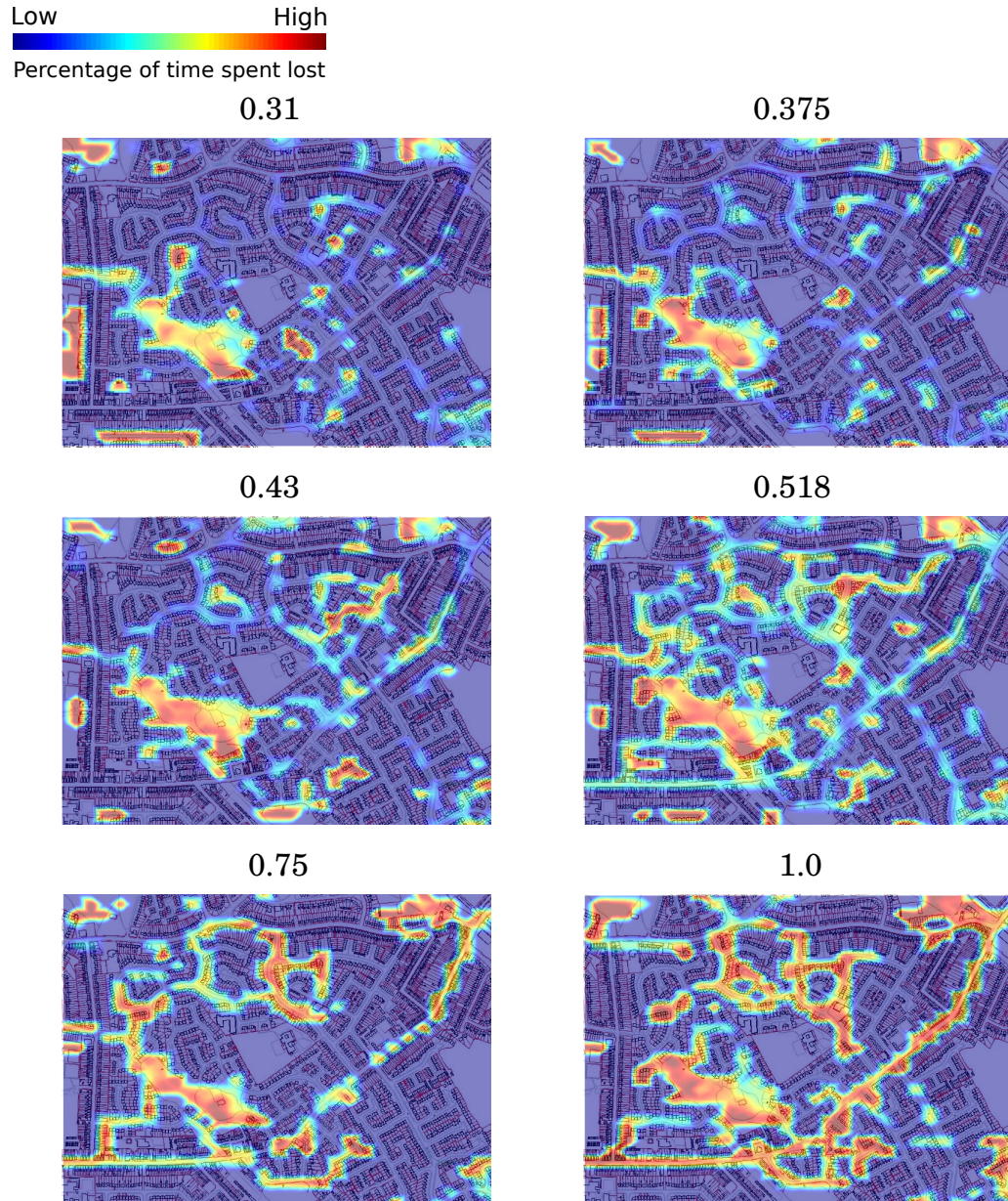


Figure A.1: Heat map of lost positions for the CUT algorithm on map 1. Speed of the target relative to the robots listed above each diagram. Values have been normalised to highlight areas that have a high probability of losing the target as opposed to areas that the target spent a significant amount of time unobserved. Ordnance Survey© Crown Copyright. All rights reserved.



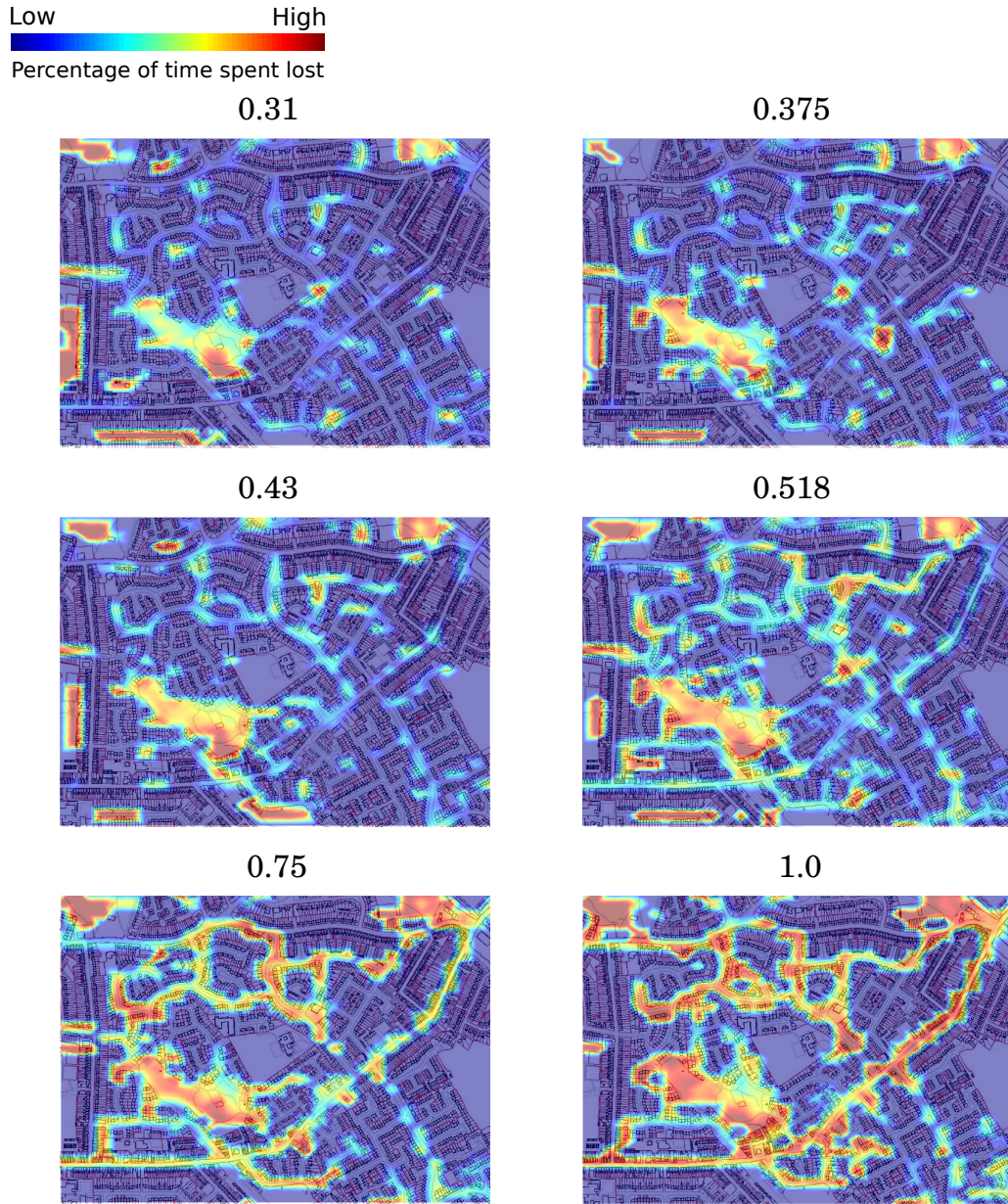


Figure A.2: Heat map of lost positions for the SCP algorithm on map 1. Speed of the target relative to the robots listed above each diagram. Values have been normalised to highlight areas that have a high probability of losing the target as opposed to areas that the target spent a significant amount of time unobserved. Ordnance Survey© Crown Copyright. All rights reserved.



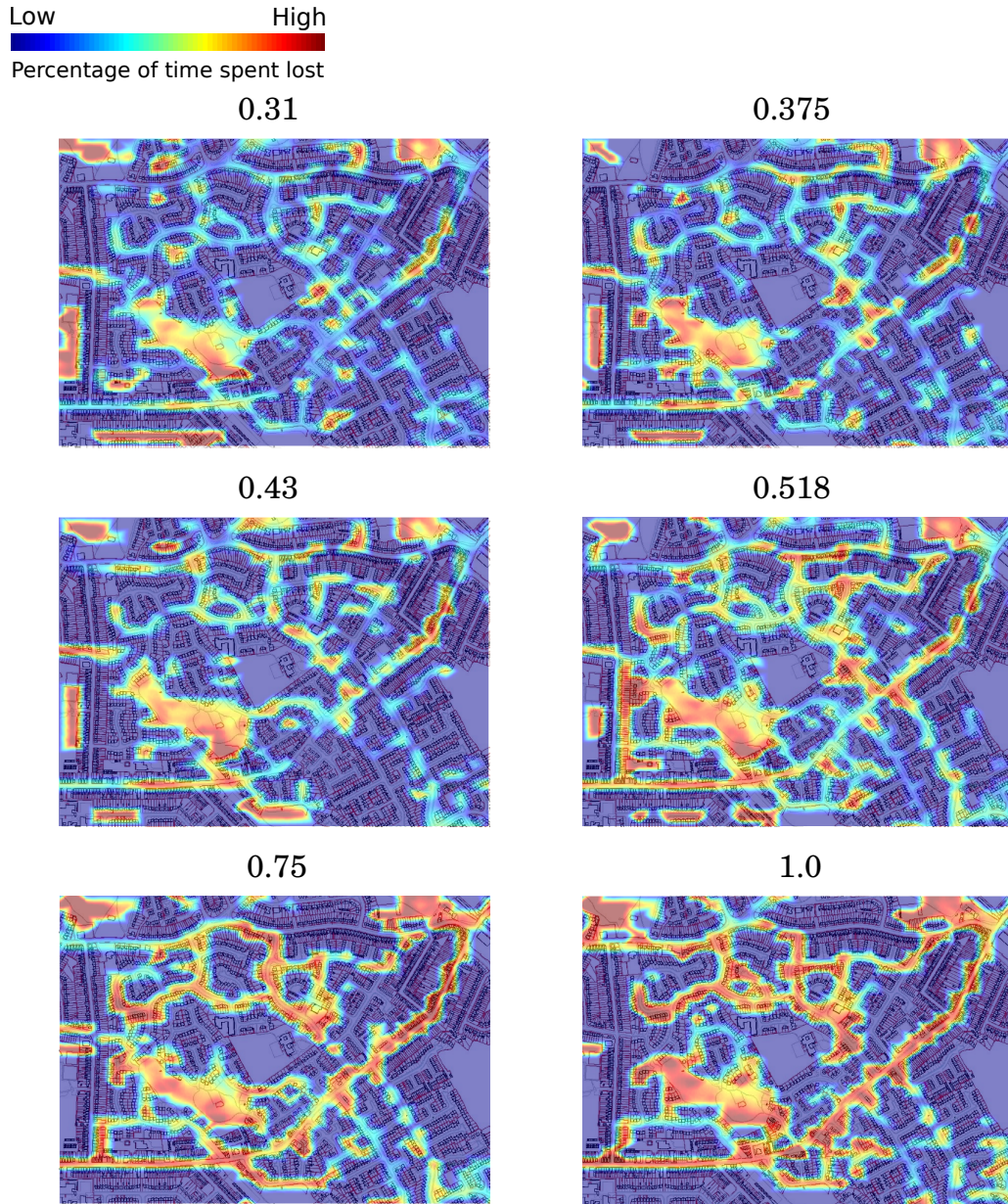


Figure A.3: Heat map of lost positions for the Branch algorithm on map 1. Speed of the target relative to the robots listed above each diagram. Values have been normalised to highlight areas that have a high probability of losing the target as opposed to areas that the target spent a significant amount of time unobserved. Ordnance Survey© Crown Copyright. All rights reserved.



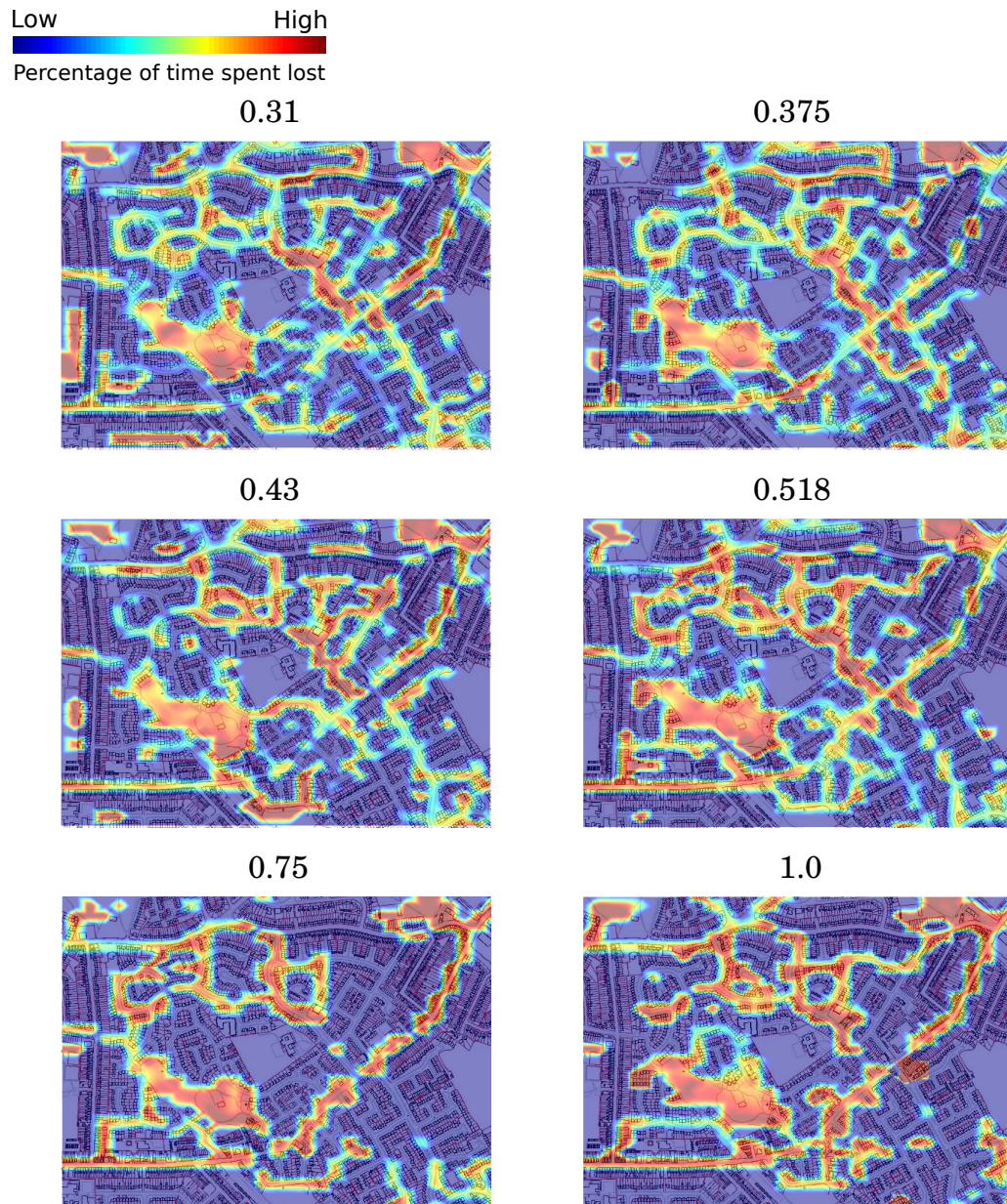


Figure A.4: Heat map of lost positions for the No Movement on map 1. Speed of the target relative to the robots listed above each diagram. Values have been normalised to highlight areas that have a high probability of losing the target as opposed to areas that the target spent a significant amount of time unobserved. Ordnance Survey© Crown Copyright. All rights reserved.

## A.2 Map 2

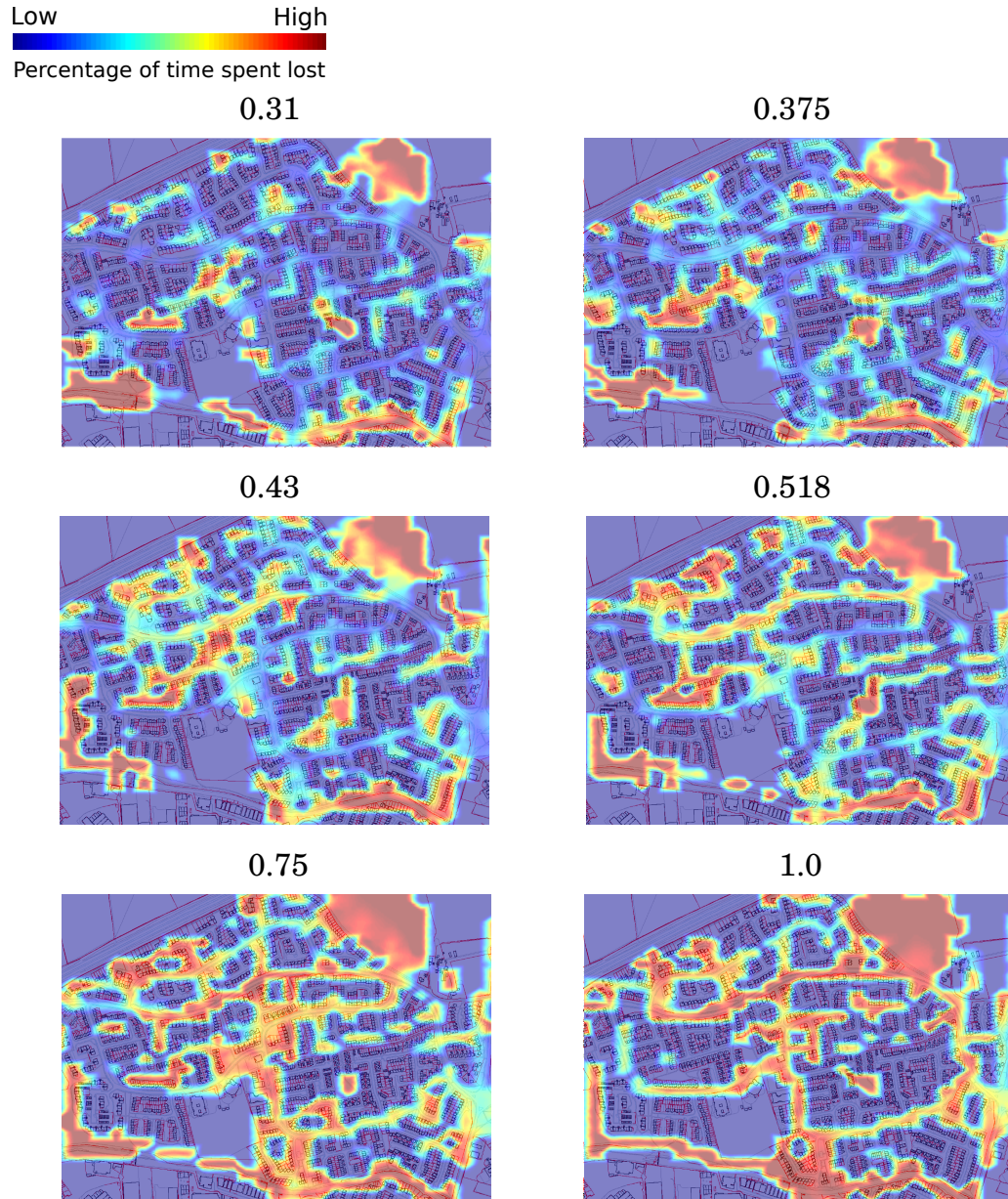


Figure A.5: Heat map of lost positions for the CUT algorithm on map 2. Speed of the target relative to the robots listed above each diagram. Values have been normalised to highlight areas that have a high probability of losing the target as opposed to areas that the target spent a significant amount of time unobserved. Ordnance Survey© Crown Copyright. All rights reserved.



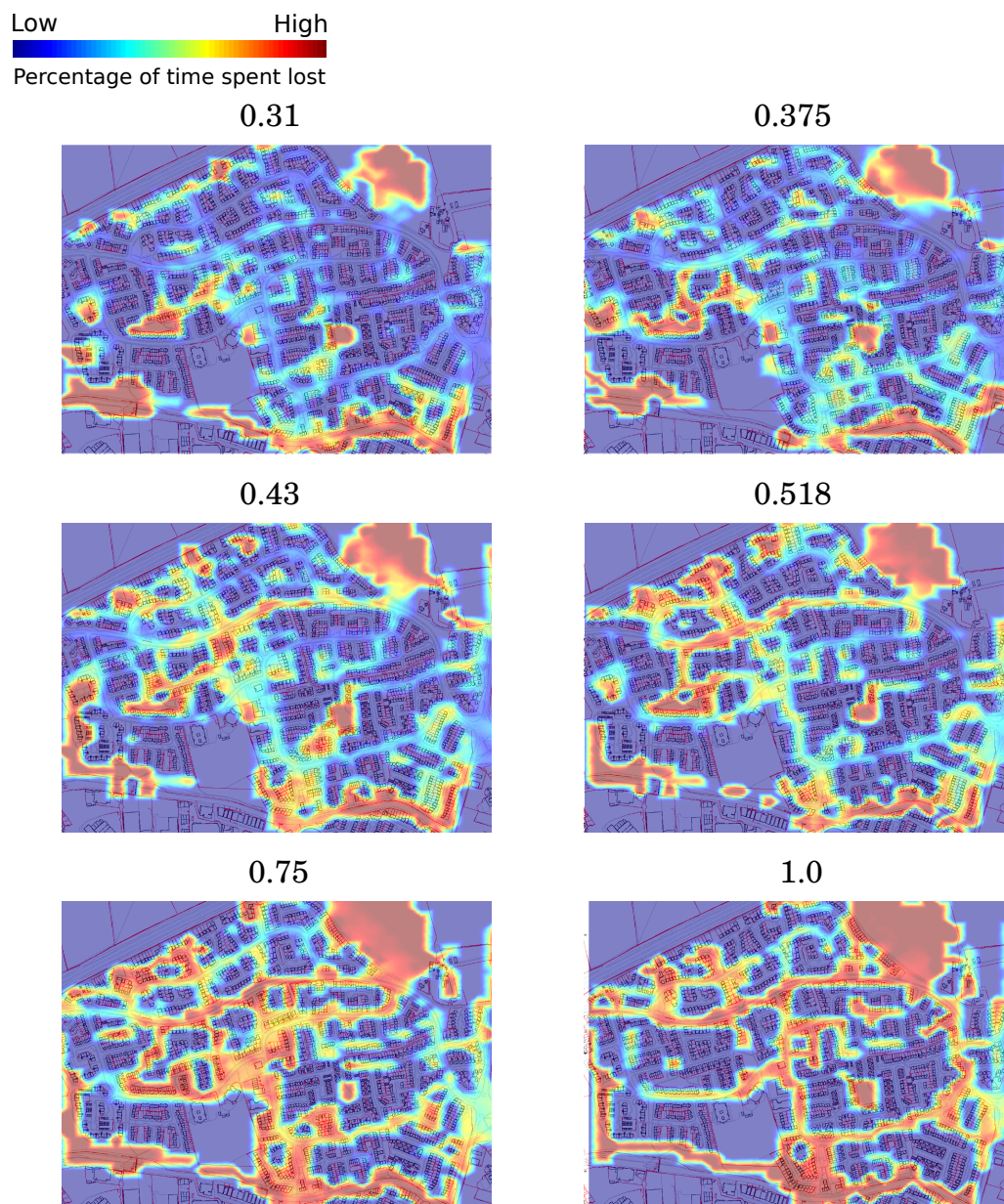


Figure A.6: Heat map of lost positions for the SCP algorithm on map 2. Speed of the target relative to the robots listed above each diagram. Values have been normalised to highlight areas that have a high probability of losing the target as opposed to areas that the target spent a significant amount of time unobserved. Ordnance Survey© Crown Copyright. All rights reserved.

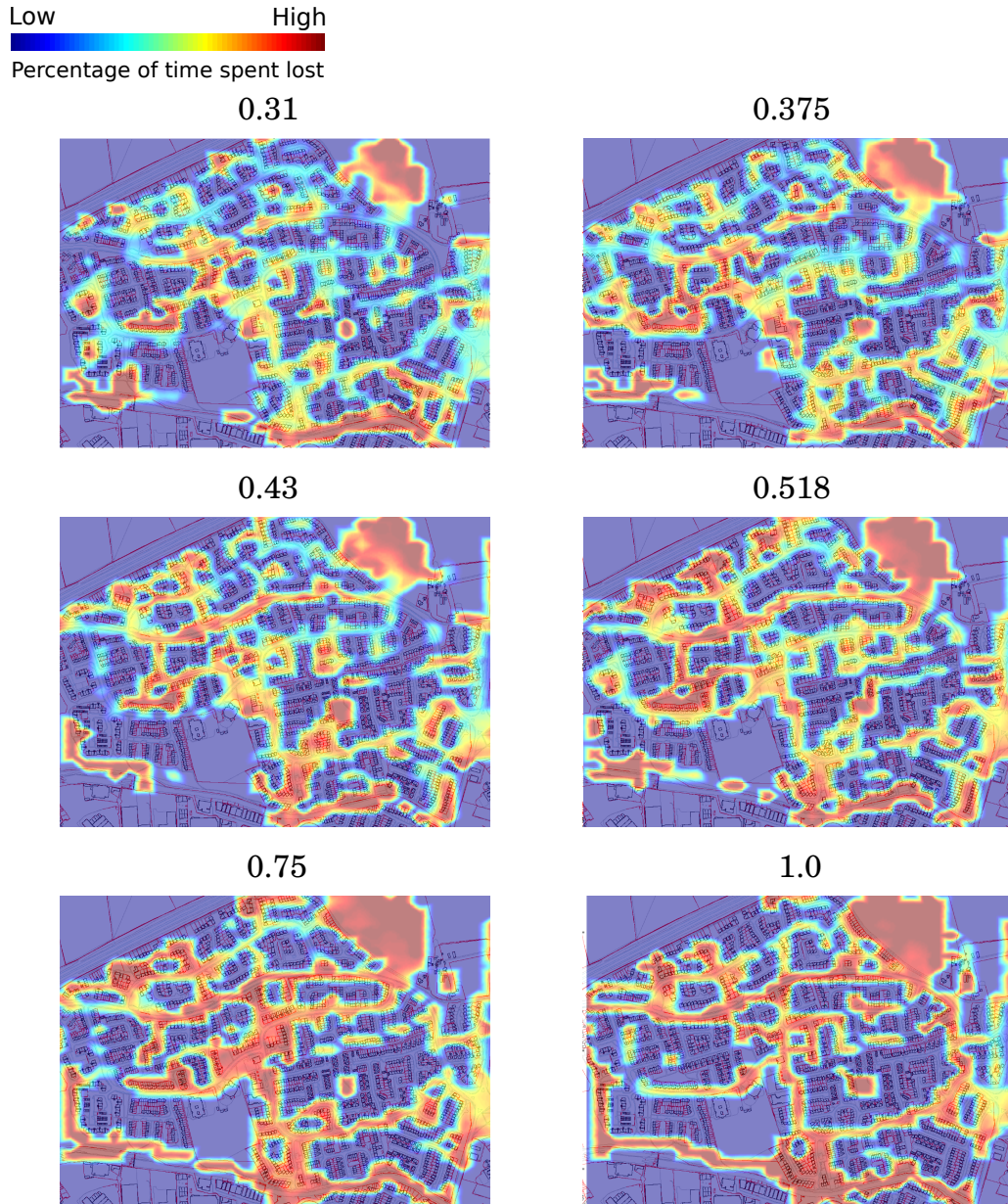


Figure A.7: Heat map of lost positions for the Branch algorithm on map 2. Speed of the target relative to the robots listed above each diagram. Values have been normalised to highlight areas that have a high probability of losing the target as opposed to areas that the target spent a significant amount of time unobserved. Ordnance Survey© Crown Copyright. All rights reserved.



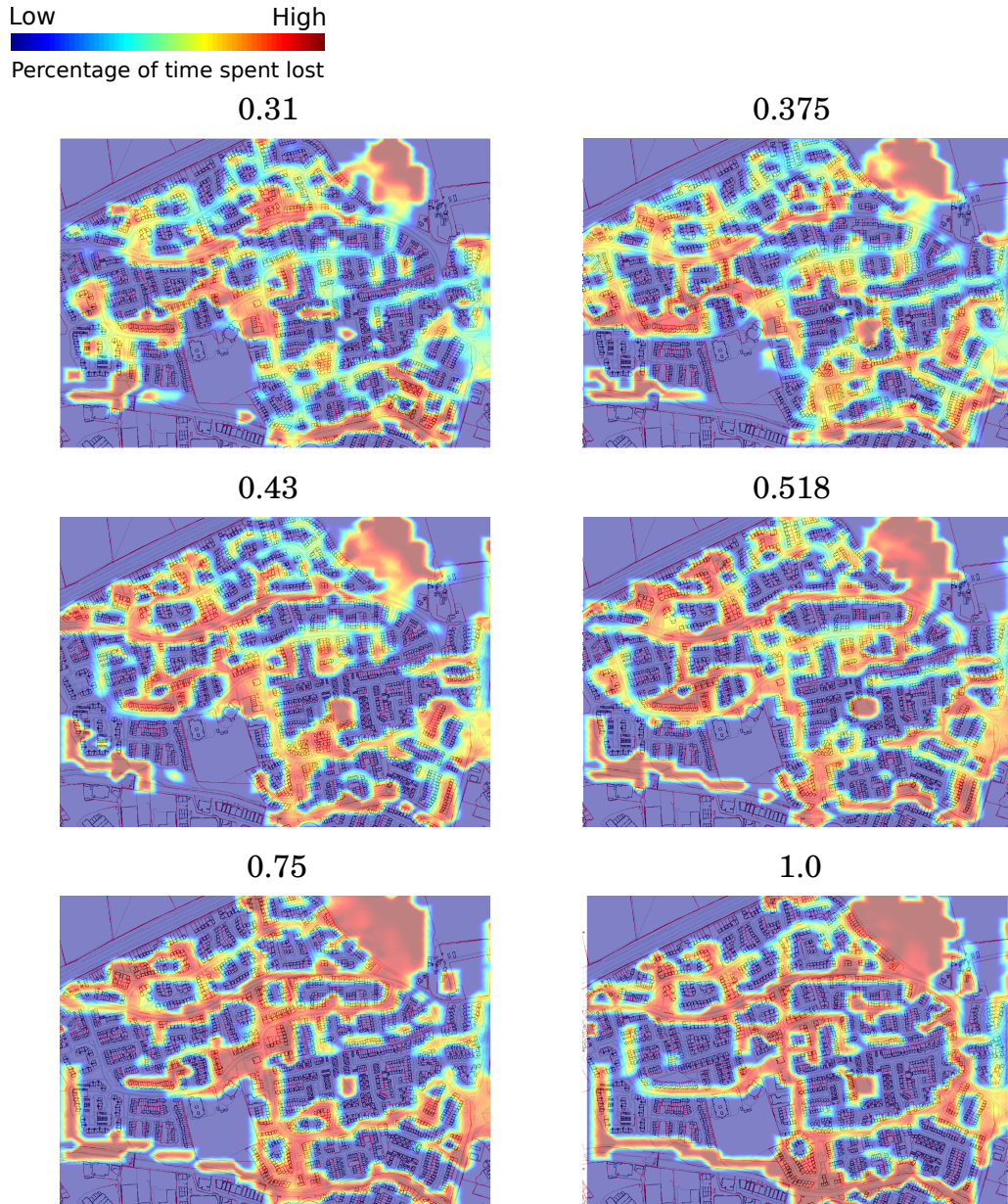


Figure A.8: Heat map of lost positions for the No Movement on map 2. Speed of the target relative to the robots listed above each diagram. Values have been normalised to highlight areas that have a high probability of losing the target as opposed to areas that the target spent a significant amount of time unobserved. Ordnance Survey© Crown Copyright. All rights reserved.



## A.3 Map 3

Low High  
Percentage of time spent lost

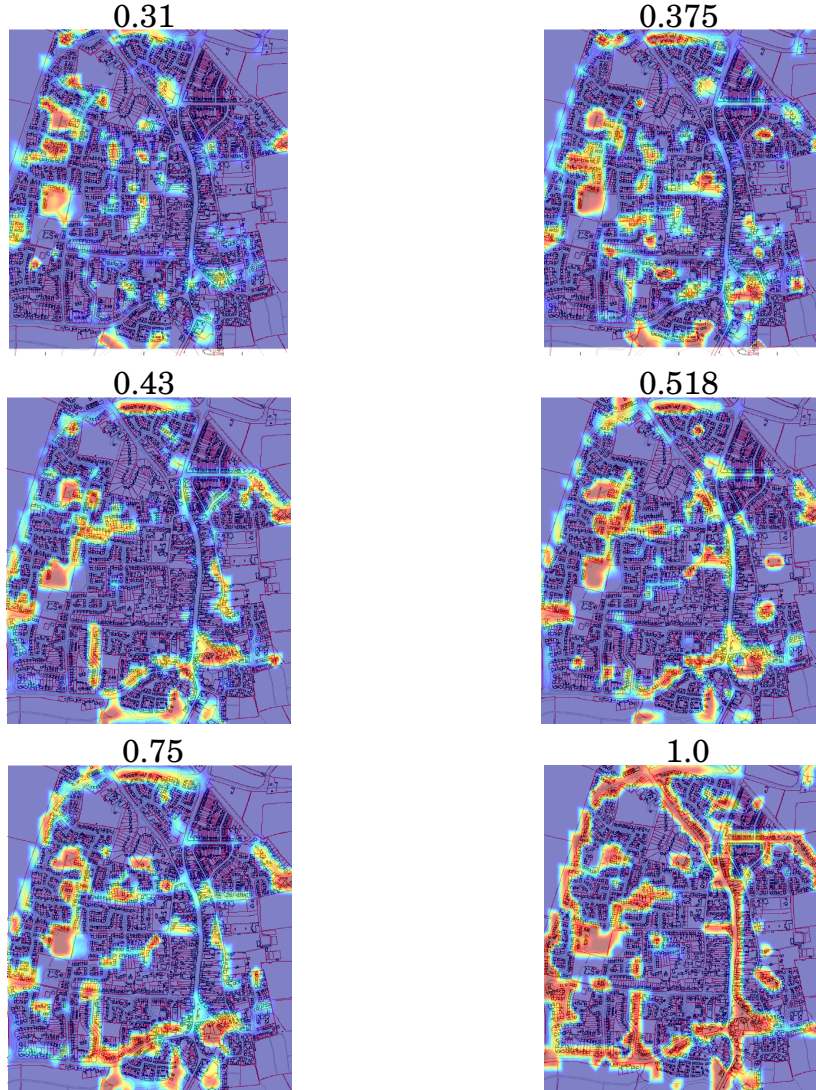


Figure A.9: Heat map of lost positions for the CUT algorithm on map 3. Speed of the target relative to the robots listed above each diagram. Values have been normalised to highlight areas that have a high probability of losing the target as opposed to areas that the target spent a significant amount of time unobserved. Ordnance Survey© Crown Copyright. All rights reserved.

Low High  
Percentage of time spent lost

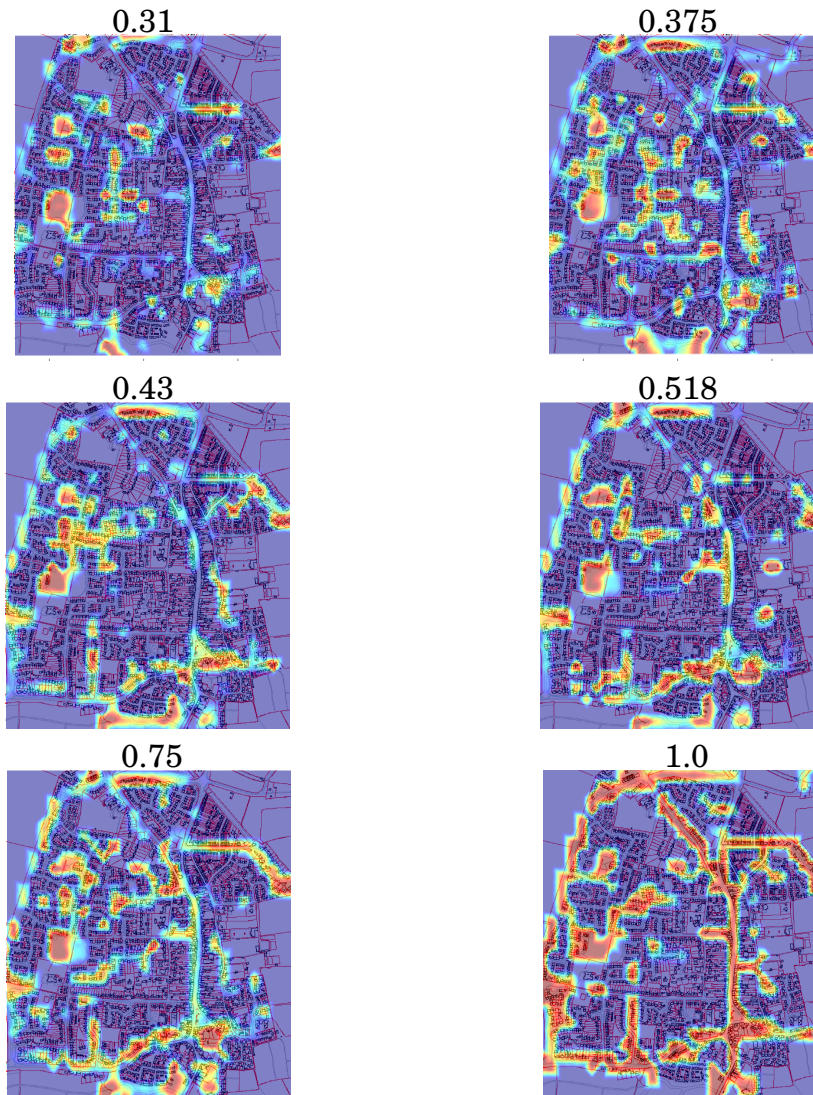


Figure A.10: Heat map of lost positions for the SCP algorithm on map 3. Speed of the target relative to the robots listed above each diagram. Values have been normalised to highlight areas that have a high probability of losing the target as opposed to areas that the target spent a significant amount of time unobserved. Ordnance Survey© Crown Copyright. All rights reserved.

Low High  
Percentage of time spent lost

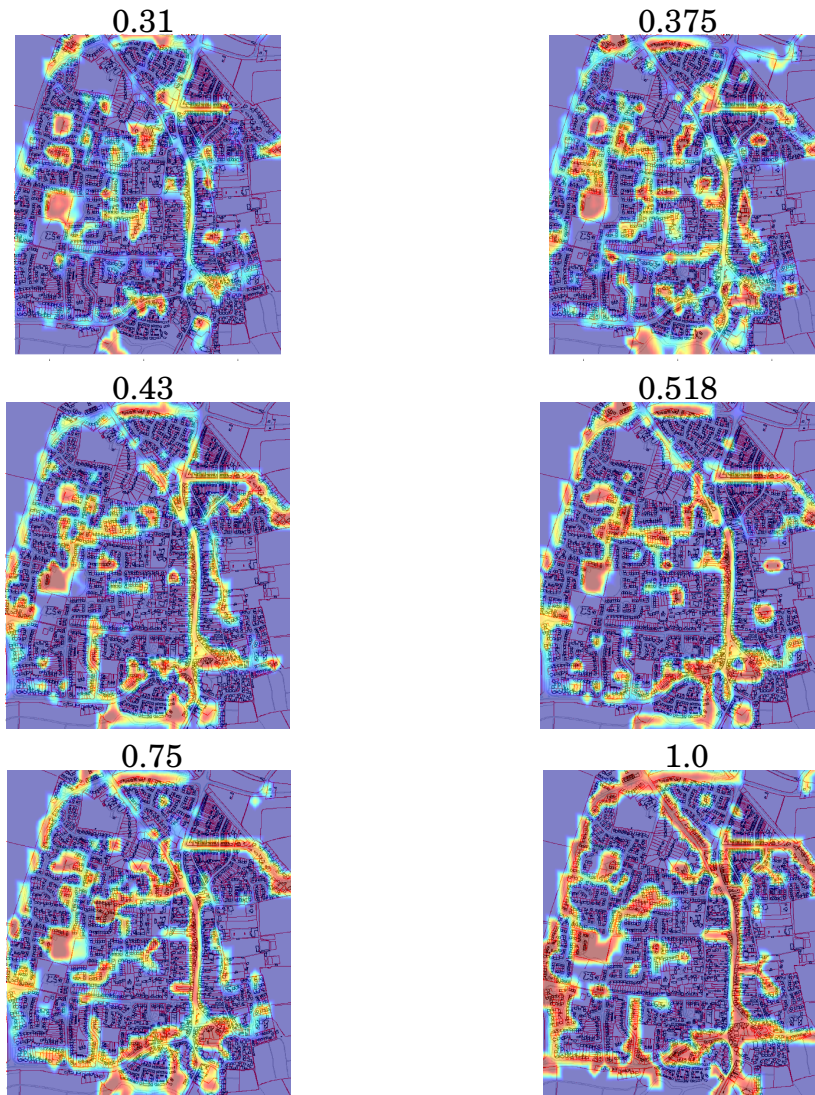


Figure A.11: Heat map of lost positions for the Branch algorithm on map 3. Speed of the target relative to the robots listed above each diagram. Values have been normalised to highlight areas that have a high probability of losing the target as opposed to areas that the target spent a significant amount of time unobserved. Ordnance Survey© Crown Copyright. All rights reserved.



Low High  
Percentage of time spent lost

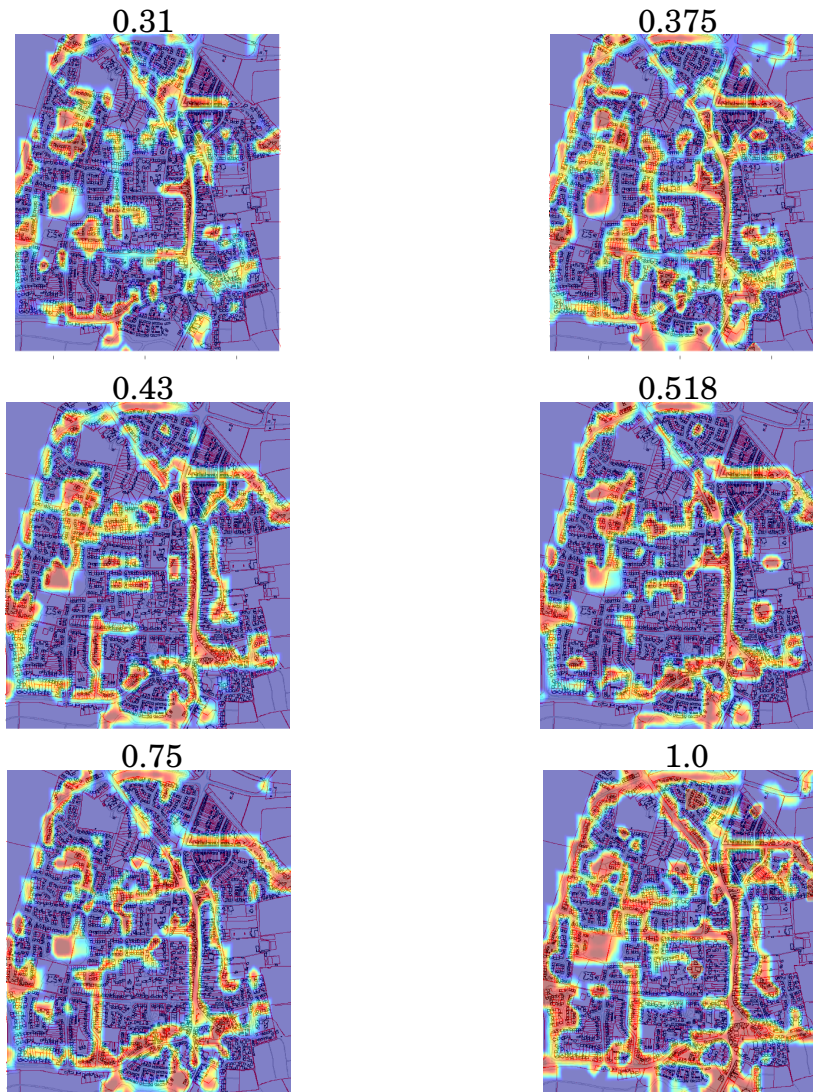


Figure A.12: Heat map of lost positions for the No Movement on map 3. Speed of the target relative to the robots listed above each diagram. Values have been normalised to highlight areas that have a high probability of losing the target as opposed to areas that the target spent a significant amount of time unobserved. Ordnance Survey© Crown Copyright. All rights reserved.

## A.4 Map 4

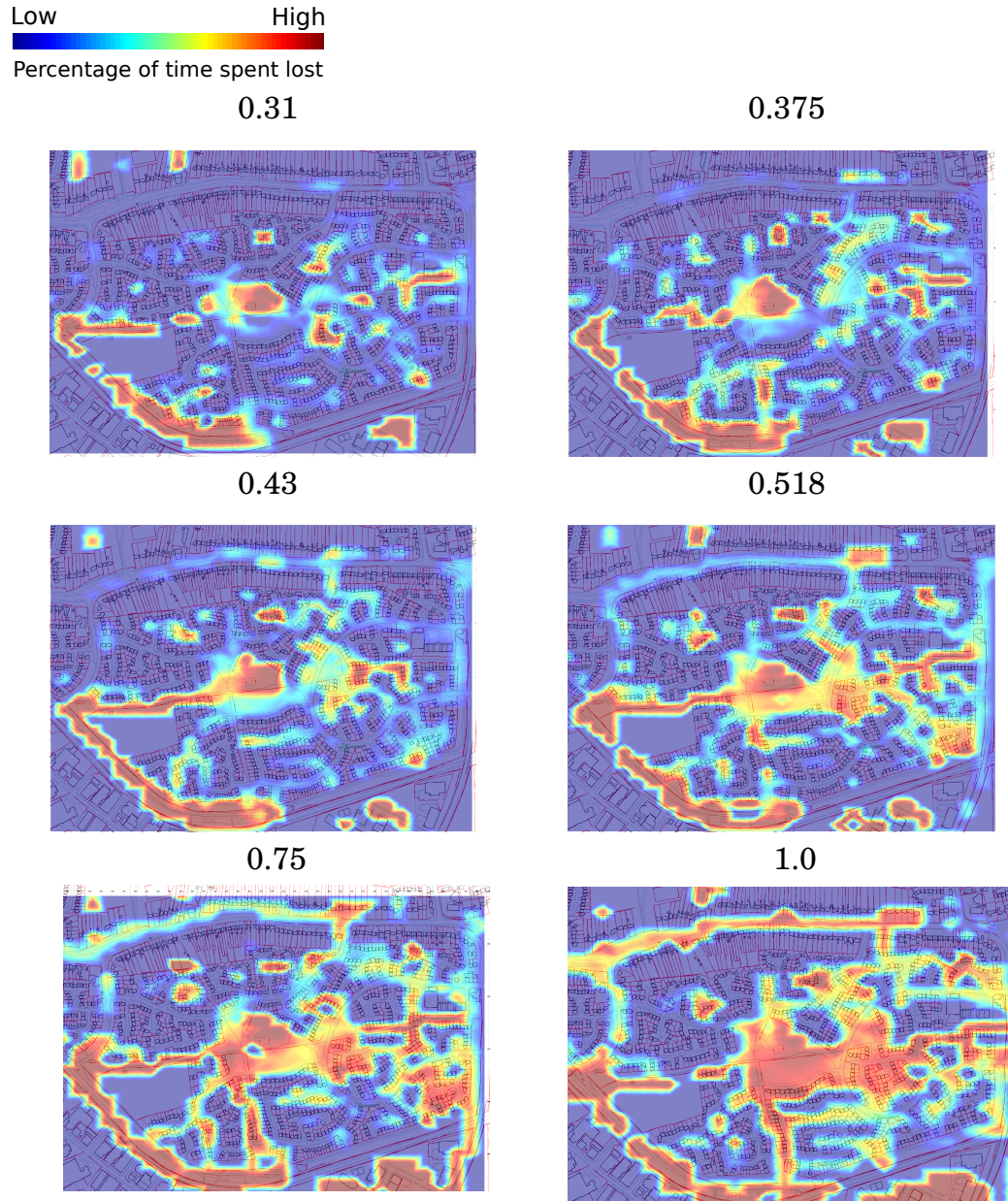


Figure A.13: Heat map of lost positions for the CUT algorithm on map 4. Speed of the target relative to the robots listed above each diagram. Values have been normalised to highlight areas that have a high probability of losing the target as opposed to areas that the target spent a significant amount of time unobserved. Ordnance Survey© Crown Copyright. All rights reserved.



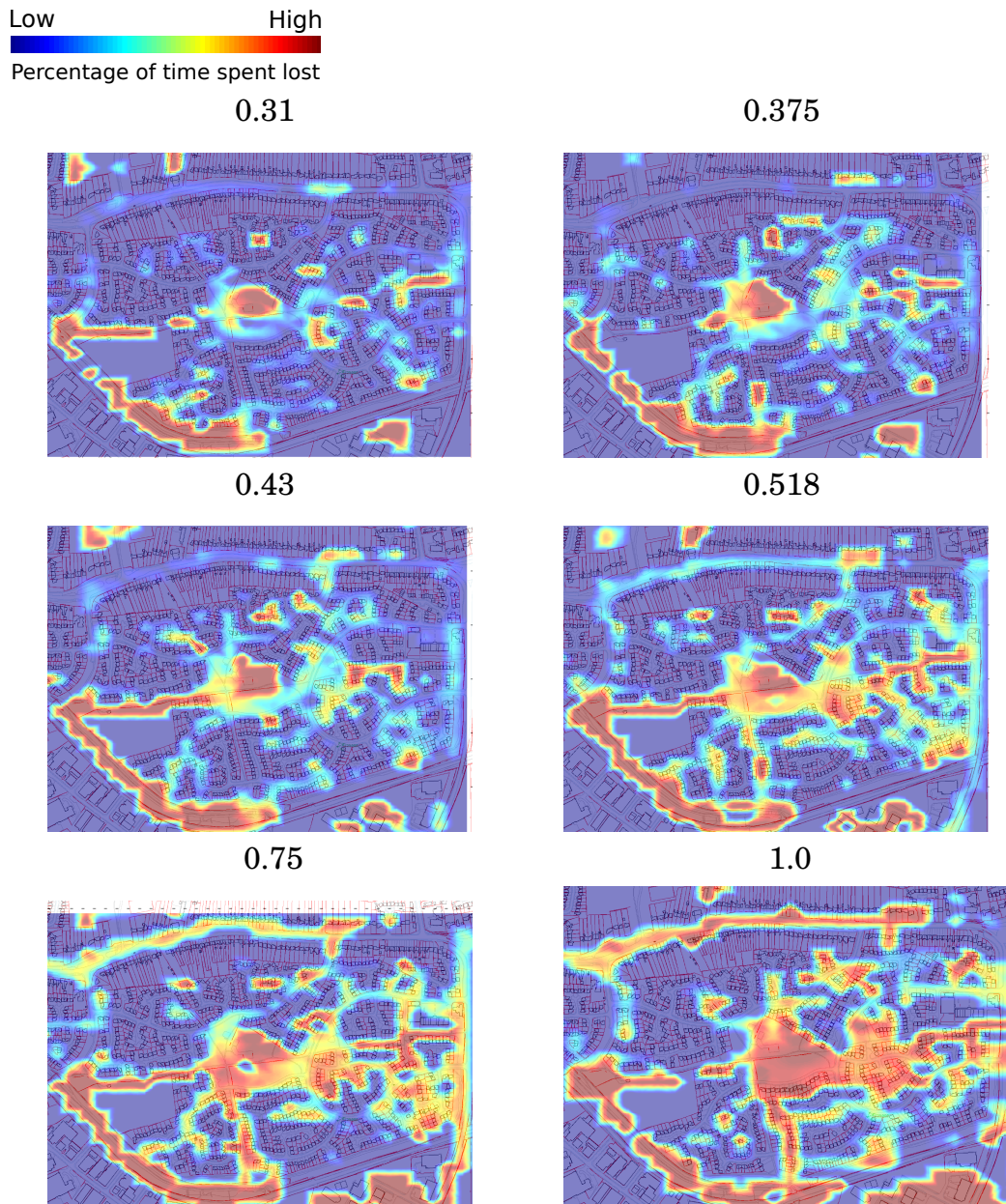


Figure A.14: Heat map of lost positions for the SCP algorithm on map 4. Speed of the target relative to the robots listed above each diagram. Values have been normalised to highlight areas that have a high probability of losing the target as opposed to areas that the target spent a significant amount of time unobserved. Ordnance Survey© Crown Copyright. All rights reserved.

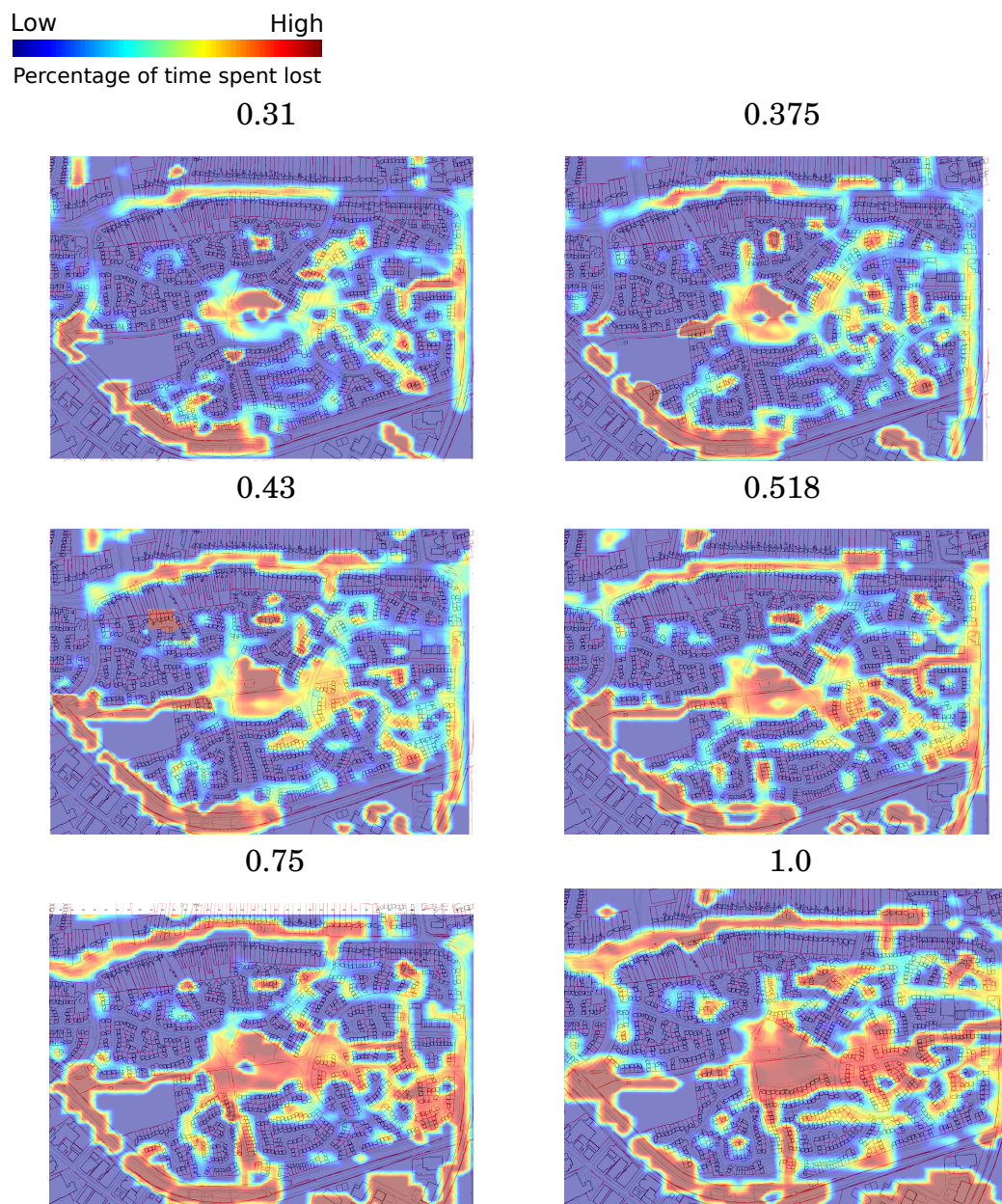


Figure A.15: Heat map of lost positions for the Branch algorithm on map 4. Speed of the target relative to the robots listed above each diagram. Values have been normalised to highlight areas that have a high probability of losing the target as opposed to areas that the target spent a significant amount of time unobserved. Ordnance Survey© Crown Copyright. All rights reserved.



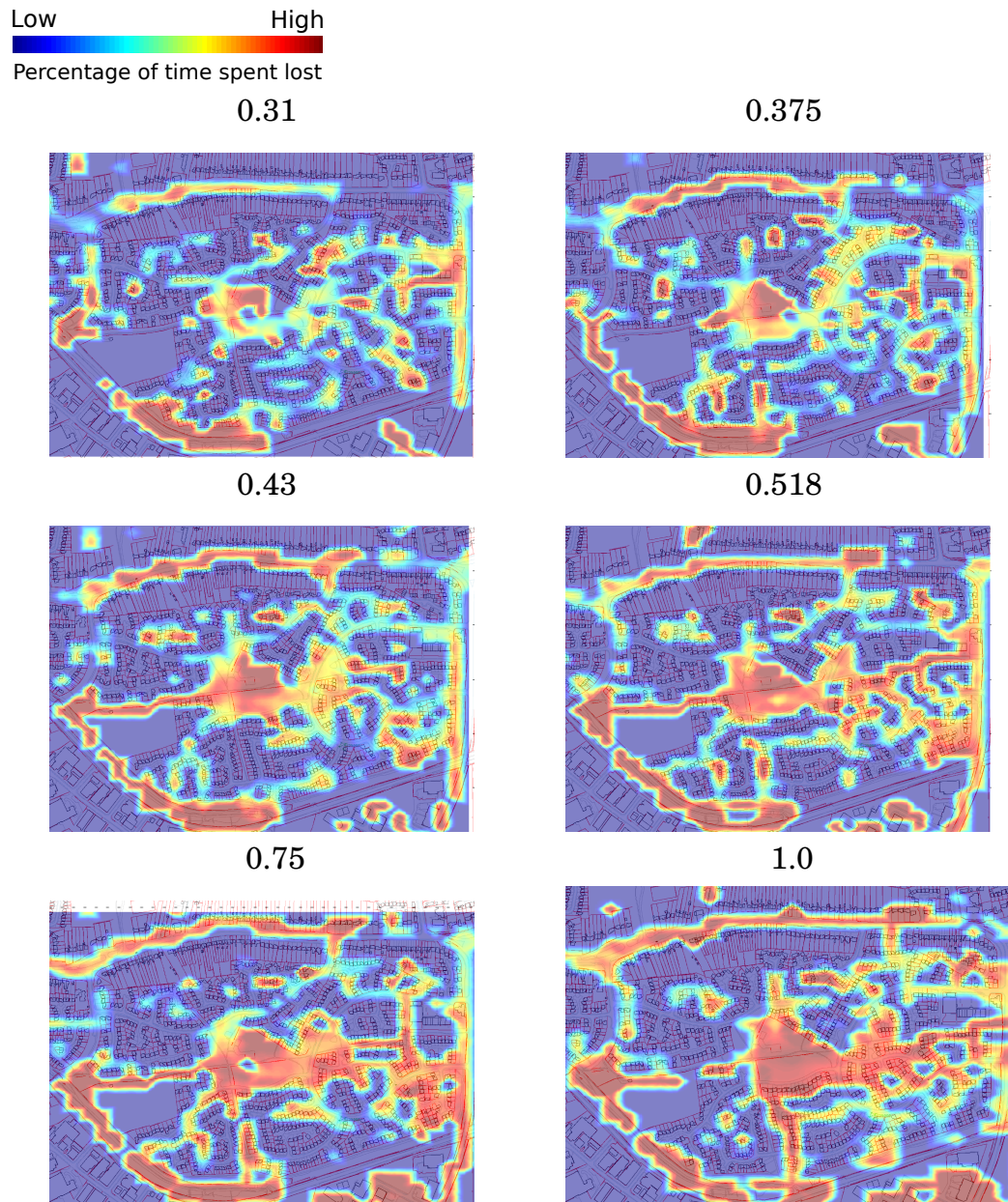


Figure A.16: Heat map of lost positions for the No Movement on map 4. Speed of the target relative to the robots listed above each diagram. Values have been normalised to highlight areas that have a high probability of losing the target as opposed to areas that the target spent a significant amount of time unobserved. Ordnance Survey© Crown Copyright. All rights reserved.



## **Appendix B**

### **Histograms of Effectiveness**

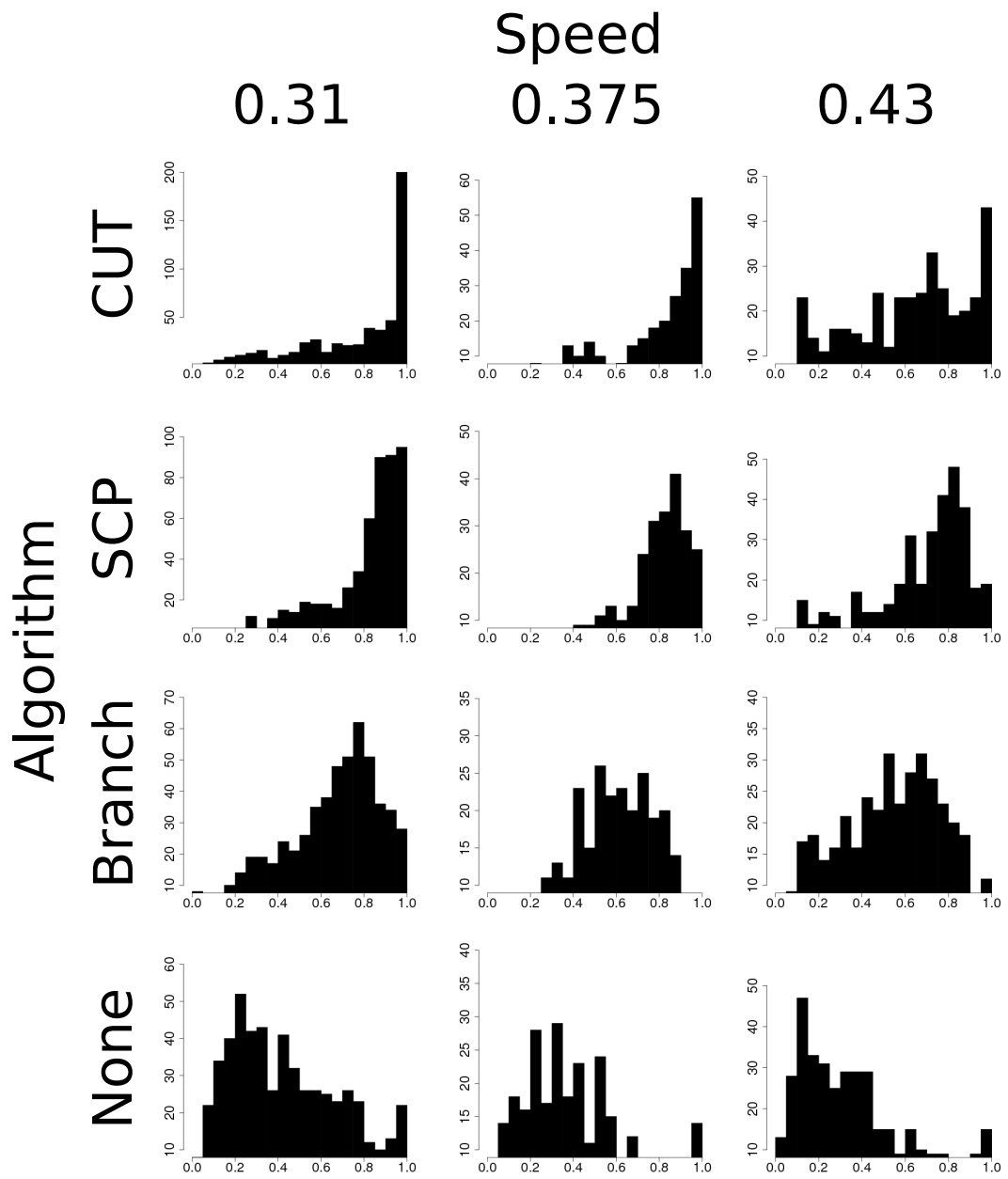


Figure B.1: Map 1 results. Horizontal axis denotes effectiveness, vertical denotes frequency.

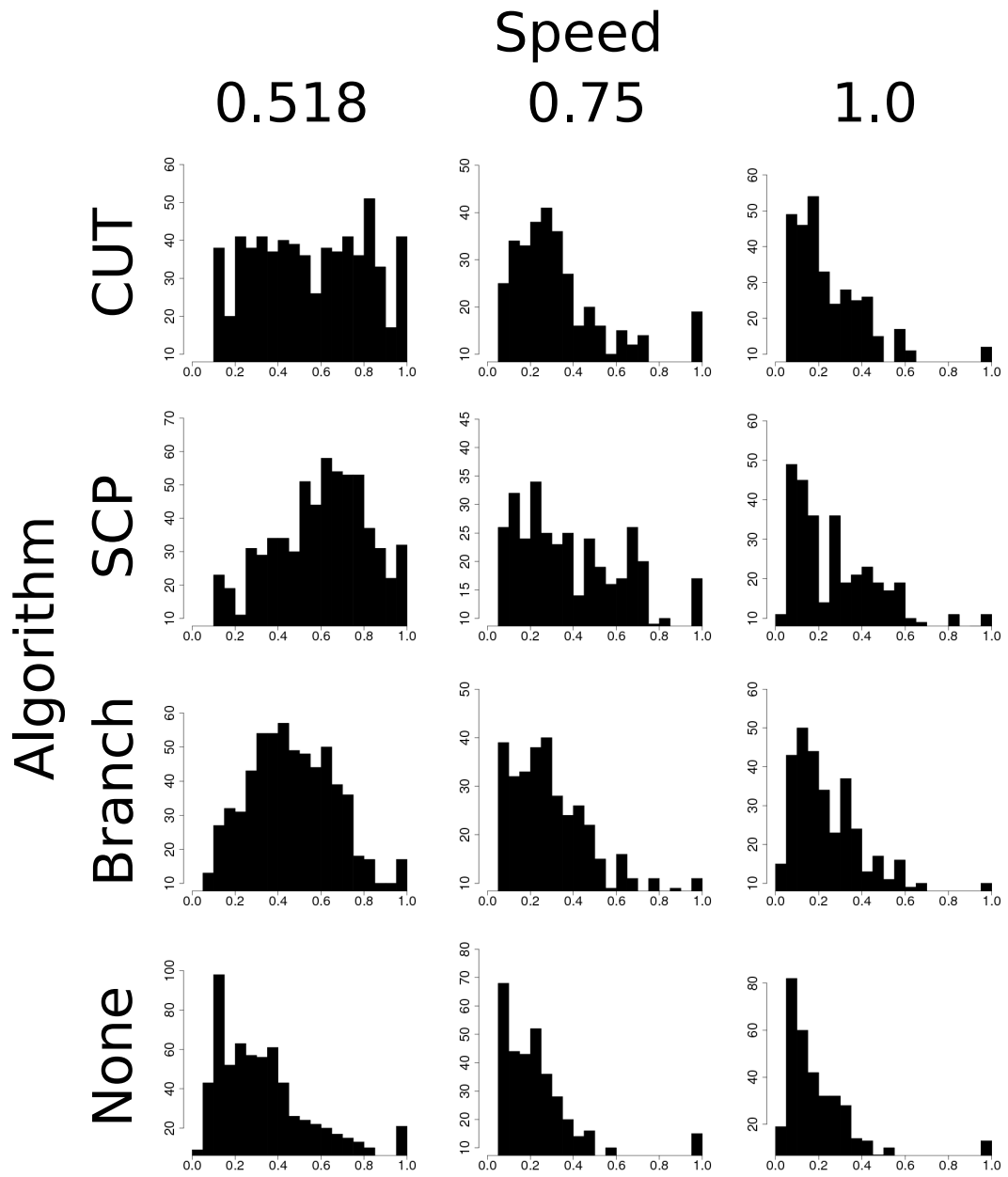


Figure B.2: Map 1 results. Horizontal axis denotes effectiveness, vertical denotes frequency.

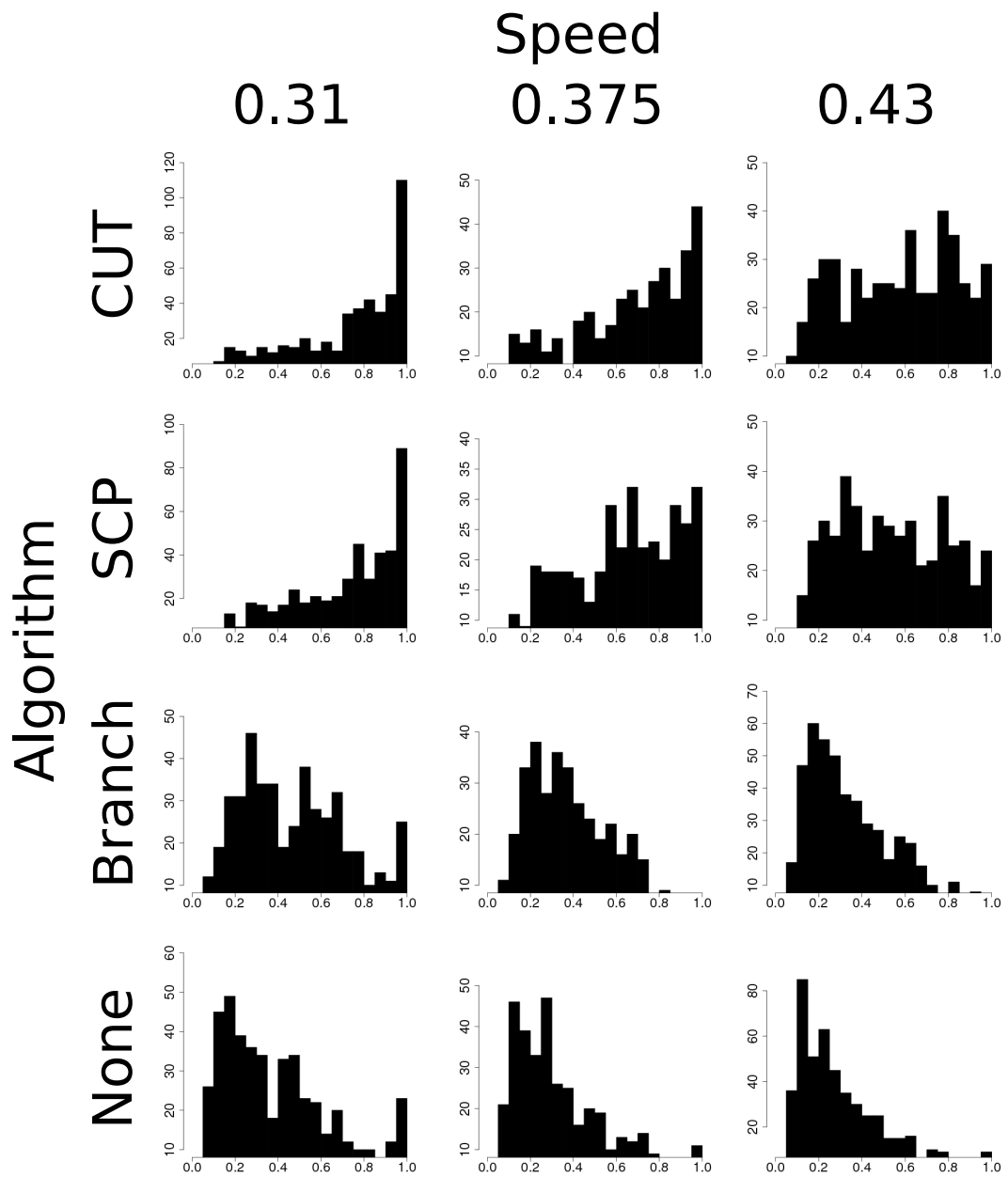


Figure B.3: Map 2 results. Horizontal axis denotes effectiveness, vertical denotes frequency.

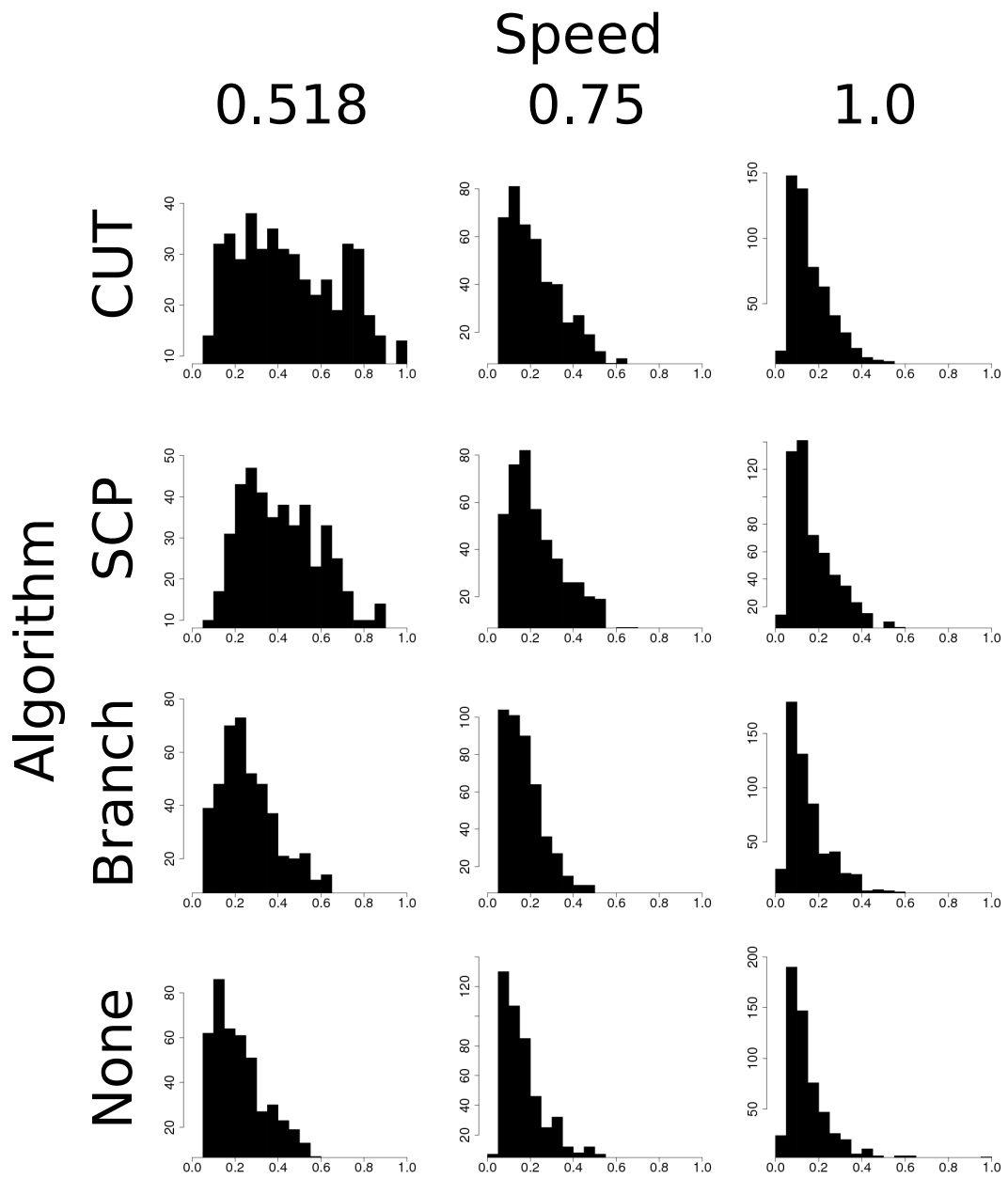


Figure B.4: Map 2 results. Horizontal axis denotes effectiveness, vertical denotes frequency.

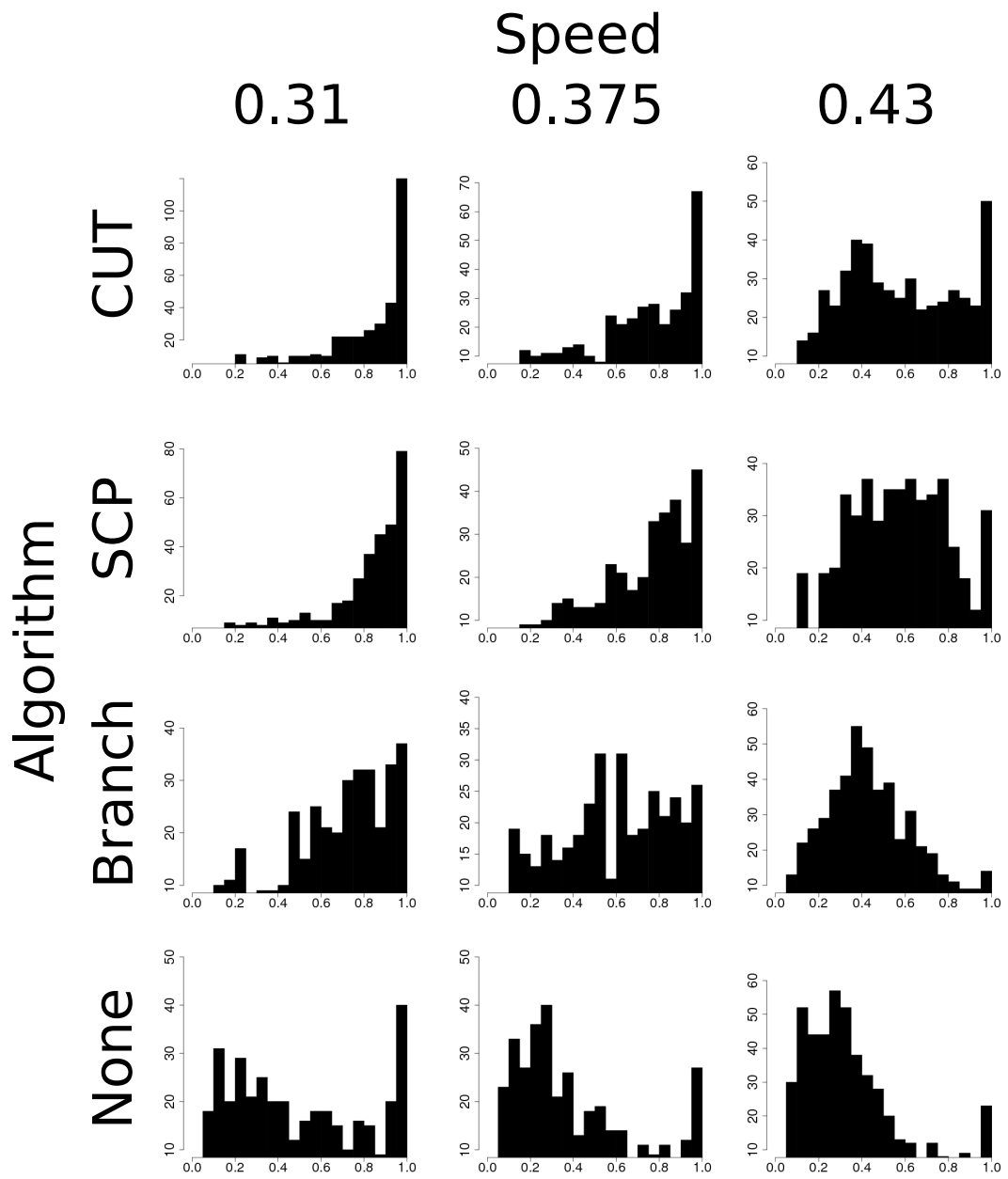


Figure B.5: Map 3 results. Horizontal axis denotes effectiveness, vertical denotes frequency.

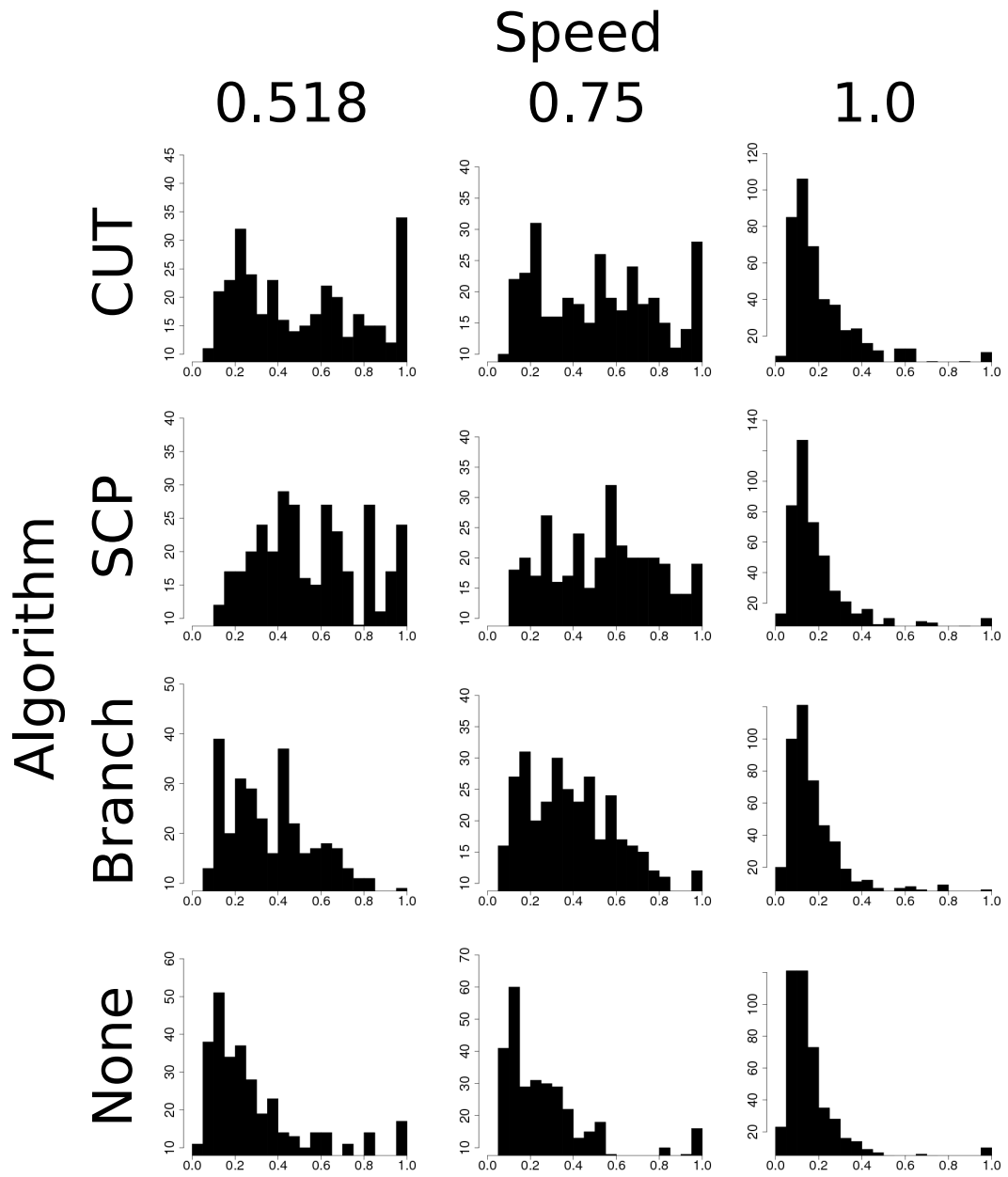


Figure B.6: Map 3 results. Horizontal axis denotes effectiveness, vertical denotes frequency.

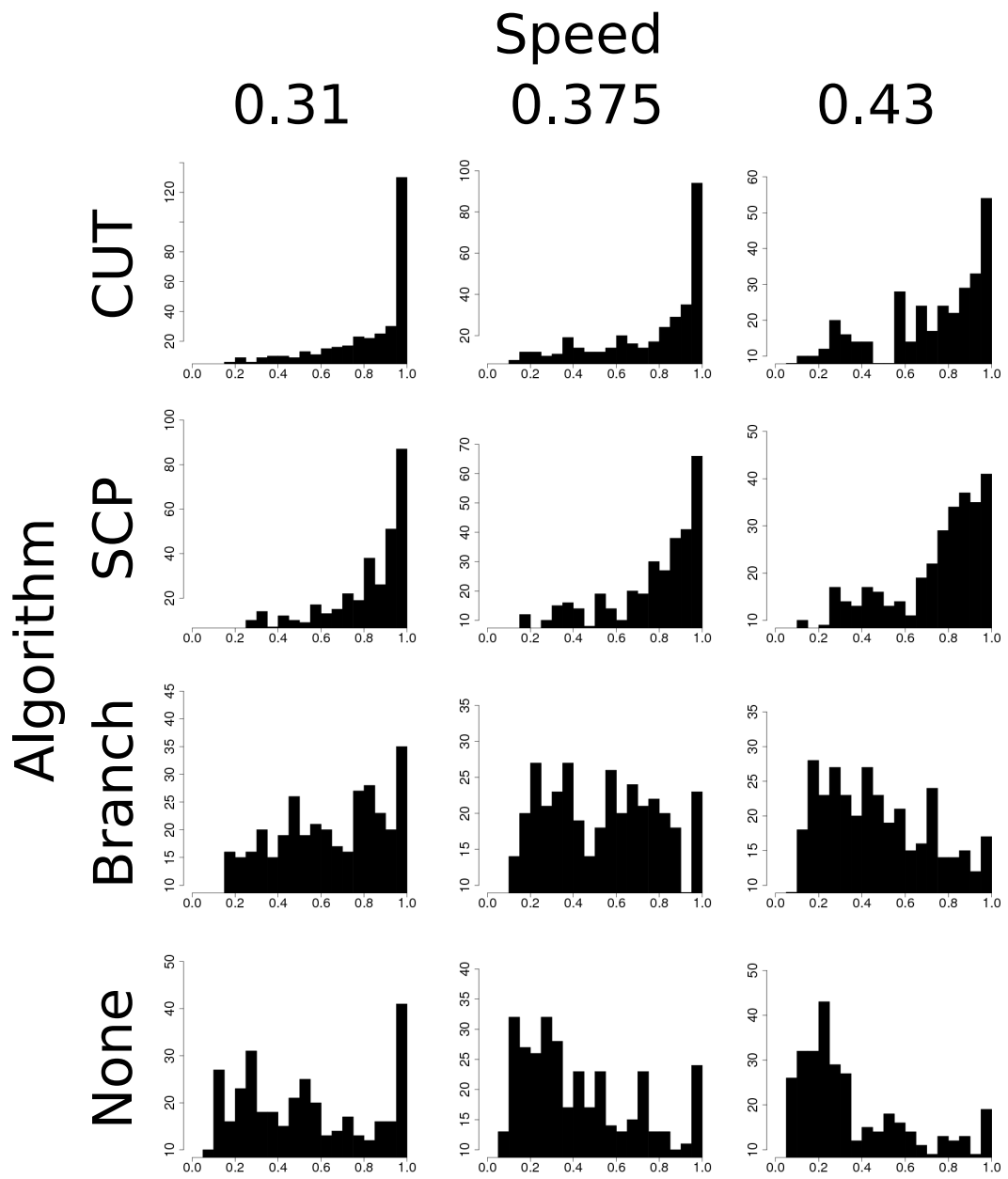


Figure B.7: Map 4 results. Horizontal axis denotes effectiveness, vertical denotes frequency.



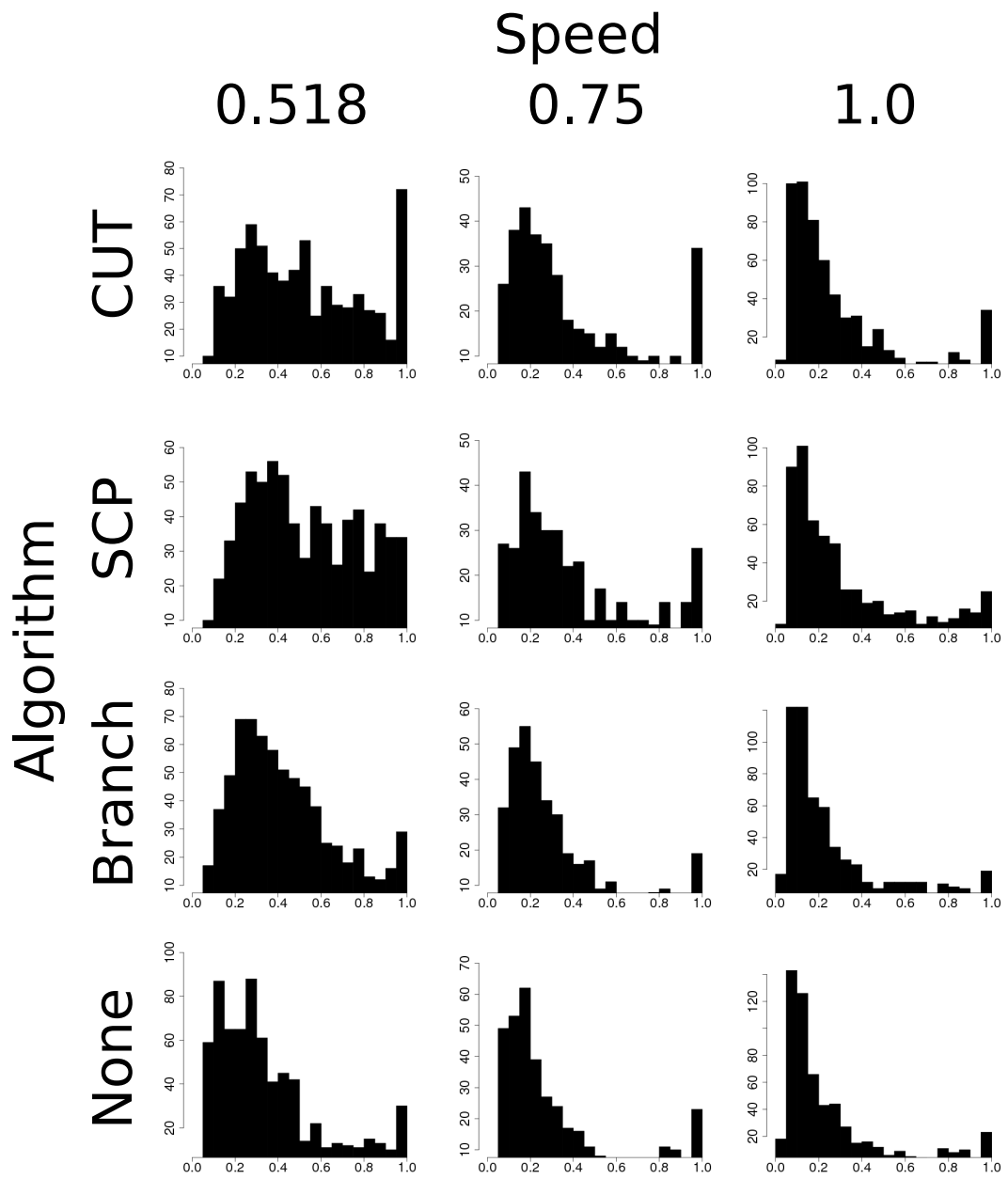


Figure B.8: Map 4 results. Horizontal axis denotes effectiveness, vertical denotes frequency.

## **Appendix C**

### **Example of Map Data**

<pre> &lt;osgb:RoadLink fid='osgb4000000015407464'&gt;   &lt;osgb:theme&gt;Road Network&lt;/osgb:theme&gt;   &lt;osgb:descriptiveGroup&gt;Road Topology&lt;/osgb:descriptiveGroup&gt;   &lt;osgb:descriptiveTerm&gt;Motorway&lt;/osgb:descriptiveTerm&gt;   &lt;osgb:natureOfRoad&gt;Dual Carriageway&lt;/osgb:natureOfRoad&gt;   &lt;osgb:polyline&gt;     &lt;gml:LineString srsName='osgb:BNG'&gt;       &lt;gml:coordinates&gt;         426459.000,304140.000         426296.000,304000.000         426238.000,303949.000       &lt;/gml:coordinates&gt;     &lt;/gml:LineString&gt;   &lt;/osgb:polyline&gt;   &lt;osgb:directedNode orientation='-' xlink:href='#osgb4000000015231358'/&gt;   &lt;osgb:directedNode orientation='+' xlink:href='#osgb4000000015231171'/&gt;   &lt;osgb:referenceToTopographicArea xlink:href='#osgb1000000114532371'/&gt;   &lt;osgb:referenceToTopographicArea xlink:href='#osgb1000001793275732'/&gt;   &lt;osgb:referenceToTopographicArea xlink:href='#osgb1000002100378879'/&gt; &lt;/osgb:RoadLink&gt; </pre>	<p>A unique number identifying the road Some terms defining the nature of the road</p>
	<p>A series of x,y co-ordinates that define the location of the road nodes</p>
	<p>Links to roads connected to this road</p> <p>Links to the data that defines the physical shape of the road</p>

Figure C.1: Example of OSMasterMap road node data. (Ordnance Survey ©Crown Copyright. All rights reserved)

Figure C.2: Example of OSMasterMap road dimensions. (Ordnance Survey ©Crown Copyright. All rights reserved)

# Appendix D

## List of Symbols

### D.1 General

$\Delta(a, b)$  A function calculating Cartesian distance between the points  $a$  and  $b$ .

$\delta(a, b)$  A function calculating topological distance between the points  $a$  and  $b$ .

$d$  Distance.

$P_x$  The position of robot  $x$ .

$T$  Total simulation time.

$t$  Time.

$j$  Index of the robot performing the calculation.

$n$  Road node.

## D.2 Symbols Used in E-CMOMMT Algorithm

$\vec{F}$  Movement vector of the robot.

$\vec{F}_o$  Force produced by robots.

$\vec{F}_{tc}$  Cartesian force produced by the target.

$\vec{F}_{tt}$  Topological force produced by the target.

$\vec{R}$  Vector in the direction of the road that the robot is on.

$\vec{F}_c$  Centring force, pulling in the direction of the centre of the road.

$\Theta_{rrf}$  Robot to robot weighting function.

$\Theta_{rtc}$  Robot to target weighting function for Cartesian force.

$\Theta_{rtt}$  Topological weighting function.

$D_{rx}$  Constants that define the profile of the robot and target weighting function.

$\Gamma$  Location of the target.

$R_\Gamma$  Road node closest to the target.

$C_j$  Point at the centre of the road parallel to the robot.

## D.3 Symbols Used in E-Jung Algorithm

$R$  Region.

$U(R)$  Urgency at  $R$ .

$D_r(R)$  Robot density at  $R$ .

$D_t(R)$  Target density at  $R$ .

$u(D_r, D_t)$  Urgency function, calculates  $U(R)$ , given  $D_r$  and  $D_t$ .

$D_{rf}(P_j, P_i)$  Density at a region, given the current robots position ( $P_j$ ) and that of another robot ( $P_i$ ).

$\Gamma$  Location of the target.

$A$  The area covered at that node.

$\phi$  Rate at which Target density increases.

## **D.4 Symbols Used in SCP Algorithm**

$r(n)$  Immediate node cost.

$u_t(n)$  Urgency for node  $n$  at time  $t$ .

$S_d$  Constant defining the ratio of topological to Cartesian distance needed to be a shortcut.

$r(n)$  Urgency at node  $n$ .

$U_o(n)$  Urgency at node  $n$  attributed to the robots.

$U_e(n)$  Urgency at node  $n$  attributed to the target.

$\tau(n)$  Travel cost.

$a_x$  Normalising constants.

$\gamma$  Time since the target was observed.

$N_i$  Current destination node of robot  $i$ .

## D.5 Symbols Used in Branch Algorithm

$g$  Nodes that have already been assigned to a robot for searching.

$M$  Search radius.

$u(n, g)$  Urgency at a node  $n$ , given the already assigned nodes  $g$ .

$D(n, g)$  A measure of separation between node  $n$  and each of the nodes that have already been assigned for searching ( $g$ ).

$\Lambda(n)$  Indicates the amount of the road network that can be accessed by travelling through that node.

$N$  Root node of the network.

$S(n, N)$  Normalised distance from the node  $n$  to the node  $N$ .

$B(n)$  The number of leaf nodes of node  $n$ .