Aston University

Improving the Fidelity of Abstract Camera Network Simulations

---

Arezoo Vejdanparast

Doctor of Philosophy

December 2019

Aston University

Improving The Fidelity of Abstract Camera Network Simulations

Arezoo Vejdanparast

Doctor of Philosophy

December 2019

This thesis studies the impact of augmenting an abstract target detection model with a higher degree of realism on the fidelity of the outcomes of camera network simulators in reflecting real-world results. The work is motivated by the identified trade-off between realistic but computationally expensive models and approximate but computationally cheap models. This trade-off opens the possibility for an alternative to augment abstract simulation tools with a higher degree of realism to capture both benefits, low computational expense with a higher fidelity of the outcomes.

For the task of target detection, we propose a novel decomposition method with an intermediate point of representation. This point is the core element of our model that decouples the architecture into two parts. Decoupling brings *flexibility* and *modularity* into the design. This empowers practitioners to select the model's features individually and independently to their requirements and camera settings. To investigate the fidelity of our model's outcomes, we build models of three detectors and apply on our lab-based image data set to create ground truth confidences. By incorporating only a few more properties of realism, the fidelity of our model's outcomes improved significantly when compared to the initial results in reflecting the ground truth confidences.

Finally, to explore the implication of our high fidelity target detection model, we select a case study from coverage redundancy in smart camera networks. Highlighting the performance of a coverage approach strongly relies on the reliability of target detection results. An underestimation in the performance of studied coverage approaches is determined by employing the standard abstract detection model when compared to the results of our model.

The identified underestimation in this study is one example of the general open concern in agent-based modelling about the unclear impact of simplified abstract models on the ability of the simulator to capture real-world behaviours.

Additional Keywords

(Simulators, Target Detection, Agent-based Modelling)

Acknowledgements

This work would not have been possible without the support of many people that helped me to go through this journey in many different ways, and to whom I am sincerely grateful.

First, I would like to express my gratitude to my supervisor Dr Peter Lewis and my associate supervisor Dr Aniko Ekart for the time they invested in my research and all their support and guidance during these years. I also would like to thank Dr Lukas Esterle for all his support and encouragement during the last two years of my PhD and Dr Harry Goldingay for his support and guidance through the first year of my PhD.

I would like to thank my PhD examiners Prof. Sven Tomford and Dr Diego Faria for their helpful advice and discussions.

My special thank is to my lovely father-in-law, Hossein Karami, who financially supported me during this PhD.

Thanks to my lab mates, who are also my friends, Noa, Deepeka, Aamir, Thomas, Vania and Reham for all the lunches together, all chats and laughs.

My sincere thanks to my family, for all their support through these years, to my gorgeous son, Mohammad H. for kindly sharing his toys with mummy for her lab experiments, being patient with mummy to work even during weekends while I was writing this thesis! And for my best friend and husband Ali, for being amazingly patient and encouraging and putting up with me through this journey unconditionally. I simply couldn't have done this without you.

Finally, I would like to thank my parent Mohammad, Kobra who set me off on the road to this PhD a long time ago and for always supporting me.

Contents

List of Abbreviations

- FoV - Field of View
- GT - Ground Truth
- Hi-Fi - High Fidelity
- Lin - Linear
- MSE - Mean Squared Error
- ORB - Oriented FAST and Robust BRIEF
- PIP - Patch Image Proportion
- RL - Reinforcement Learning
- RMSE - Root Mean Squared Error
- SCN - Smart Camera Networks
- SD - Standard Deviation
- SIFT - Scale Invariant Feature Transform
- SR - Symbolic Regression
- SSE - Sum of Squared Error
- SSR - Sum of Squared Residual
- SURF - Speeded Up Robust Features
- SVR - Support Vector Machine

List of Figures

List of Tables

Publications arising from this thesis

[1] Arezoo Vejdanparast, Peter Lewis, Lukas Esterle (2018). Online Zoom Selection Approaches for Coverage Redundancy in Visual Sensor Networks. In: *12th International Conference on Distributed Smart Cameras.*

[2] Arezoo Vejdanparast (2018). Coverage Redundancy in Visual Sensor Networks. In: *the 12th International Conference on Distributed Smart Cameras.*

[3] Arezoo Vejdanparast, Peter Lewis (2019). Learning and Sharing for Improved k-Coverage in Smart Camera Networks. In: *4th International Workshops on Foundations and Applications of Self* Systems (FAS* W)- In conjunction with SASO2019*

[4] Arezoo Vejdanparast, Peter Lewis, Lukas Esterle (2020), Improving the Fidelity of Camera Network Simulations *Under review of ACM Transactions On Sensor Networks (TOSN) journal.*

Part I

Introduction and Motivation

Chapter 1

Introduction

Over the last few decades, remarkable infrastructure growths have been noticed in security and safety-related issues. With an increased demand for security and safety, surveillance systems become an important domain that attracts researchers attention. Deployment of large-scale surveillance systems in the real world is a significant undertaking that often faces several difficulties. It is both cost and time-intensive, which might even prohibit establishing such a surveillance system from the beginning. Moreover, in some cases, available empirical data is limited due to legal impediments (e.g., [131]) for the purpose of target detection and tracking applications.

Sanmiguel et al. described, "*the success of smart camera networks (SCNs) depends on the availability of simulators that facilitate design, prototyping, and validation of performance objectives before deployment*" [92]. Smart cameras are embedded devices able to observe their environment, process the acquired images on-board, and communicate aggregated information and extracted knowledge with other devices. This enables them to detach from central components, analysing imagery locally, making decisions and acting on them autonomously.

Existing camera network simulation tools often reflect real-world information in

different ways (the spectrum represented in Figure 2.1). At one end of the spectrum, realistic or complex models aim to represent a specific phenomenon in the real-world, while taking many properties of realism into account. Properties of realism refer to the levels of details, (e.g. environmental factors or constraints) in which a simulator reflects real-world operations. At the other end of the spectrum, simple and abstract models only capture a limited number of real-world properties within an abstract environment [14].

While using the complex (i.e.realistic) models can improve the accuracy and ty of the outcomes in the sense of reflecting real-world operations, incorporating many realism properties can make simulation tools cumbersome and slow. Hence, this can limit their scalability to support larger scenarios (e.g. [82]). On the other hand, using abstract models allows for development and verification of new theories with the results easy to interpret. However, due to the simplified nature of these abstract models, which remove details and hence can introduce errors, the outcomes can be imprecise and have room for improvement in terms of their fidelity.

In general, it is not clear what impact making such simplified abstract models has on the ability of the simulator to capture real-world behaviour. This is also a general open concern in agent-based modelling [36].

This gives rise to an important trade-off between realistic but computationally expensive simulations and approximate but computationally cheap simulations. This trade-off opens the possibility for an alternative to augment abstract simulation tools with a higher degree of realism. Thereby creating solutions that capture both benefits, low computational expense with a higher fidelity of the outcomes. This thesis aims to contribute to this approach.

1.1 Scenario

Within surveillance systems, target tracking, and coverage analysis are two important applications that their performance highly relies on the reliability of target detection results [129]. These results can provide valuable information about the location of targets, their temporal correspondences, and movement pattern over time. Methods based on background subtraction, frame differencing, and optical flow, are commonly used in video-based surveillance systems, where high quality real or synthetic videos (or images) of the scene are available. However, abstract simulation environments often do not have access to real-world, high-quality imagery of the scene. Therefore, they ignore the details of the scene and model objects simply as moving points (e.g. vertices of a grid) across the surveillance field. Therefore, in the case of using an abstract simulation environment, an alternative is required.

Esterle et. al [31], explored the impact of incorporating one physical property, i.e. *cameras' zoom* on object tracking performance using such abstract simulation environment, i.e. CamSim smart camera network simulation environment [28]. Throughout a set of profiling experiments, they showed there is a simple linear correlation between the pixel density of a region of interest and classification success rate. A *pixel* defines the size of the smallest, clearly observable object with distinct boundaries. This model deployed across CamSim environment is referred to as the *CamSim standard model of target detection.* CamSim standard detection model only incorporates the camera's current zoom as a property of realism. In this sense, the model is extremely abstract.

Inspired by their work, in the light of the identified trade-off, we augment the extremely abstract CamSim standard target detection model with a higher degree of realism, aiming to capitalise on both benefits, low computational expense with

a higher fidelity of the outcomes. The studied target detection task is related to a *confidence* representing the probability of the target being correctly detected in the right location within the camera's field of view across an abstract simulation environment.

The term *model* we used across this thesis, refers to its general sense; "*any abstraction of the system and its environment that captures some knowledge and may be used for reasoning with respect to the system goals*" [60].

Throughout this thesis, we propose a novel decomposition method by establishing an intermediate point of representation called Patch Image Proportion, PIP. Within proposed architecture, PIP is a core element, capturing a ratio of the pixel density of a patch (i.e. projection of a target on the image sensor of a camera) to an entire image. PIP decouple the architecture into two partial models, namely *feature abstraction models*, represented by f, and *detector models*, represented by g. Decoupling is useful in bringing *flexibility* and *modularity* within the design of the model. This empowers practitioners to select the model's features individually and independently to their requirements and camera settings. In other words, this decomposition enables composability of different functionalities required for detection and tracking in smart camera applications. This, in turn, lifts the limitations imposed by models focussing on certain optical properties of specific camera types and models relying on a particular classifier, i.e. detector. Indeed, the sufficiency of the selected middle point in undertaking the decomposition process is a vital question we address throughout this thesis.

Given camera's pixel density, we investigate the impact of only three physical parameters, the size of the target, the distance from the camera, and the camera's current zoom on the pixel deviation of PIP. The proposed model is purposefully,

abstract and generic. This helps to support its applicability to a broader range of applications that face the trade-off between fidelity and corresponding computational expense. Indeed, in realistic modelling of the target detection, it is necessary to incorporate more specific environmental and camera factors imposed by real-world constraints such as camera's aperture, lens distortion, and lighting.

Further, the implication of our proposed model, *high fidelity abstract target detection model* will be explored across a case study from coverage redundancy domain in smart camera networks.

1.2 Overarching Research Questions

This thesis is concerned with two overarching research questions:

1. Within the development of the target detection model, is the selected intermediate point of representation, sufficient to undertake the decomposition? More specifically, how accurate it is in predicting the detectors outcomes (as the ground truth across this study)?

2. What is the implication of employing the high fidelity target detection model on the results of a selected case study? More specifically, what is the implication of the augmented target detection model on the performance of coverage approaches when compared to CamSim initial results?

These questions are studied in building an augmented target detection model, with a higher degree of realism, across an abstract SCN simulation environment. The first question more specifically is discussed in Chapter 5, where, the sufficiency of ground truth PIP as an intermediate point of representation is explored in predicting the outcomes of three detector models (SURF, SIFT, ORB). To explore the

implication of our developed model, we select a case study from coverage redundancy in self-organised smart camera networks. Highlighting that the performance of coverage approaches are affected by the reliability of target detection results. The second research question specifically is studied in Chapter 8 of this thesis. Within the selected case study, the performance of coverage approaches is compared while employing two different target detection models, i) proposed high fidelity target detection model ii) CamSim standard target detection model and discuss our findings.

1.3 Contributions of the Thesis

The major contributions of this thesis are as follows:

- A novel method for decomposing the modelling of target detection into two partial models, by establishing an intermdiate representation point. The aim of this method is to bring flexibility and modularity into the design of the model. This empowers practitioners to be able to select the model's features individually and independently to their requirements and camera settings.
- The description of an intermdiate representation point, PIP, within the decomposition method. PIP is a core element of the model, capturing a ratio of the pixel density of a patch (i.e. projection of a target into camera's image sensor) to an entire image which undertakes the decoupling role.
- A lab-based image dataset, created using a real camera, with 480 images. The image dataset is used to establish the ground truth PIP values, as well as to build three sets of ground truth confidences employing three detector models, i.e. ORB, SIFT, and SURF.
- An analysis of the sufficiency of the three physical properties, distance from

the camera, size of the target, and camera's current zoom in predicting ground truth PIP values.

- A comparison between fidelity of our model's outcomes, and the results of standard model of CamSim in approximating the ground truth confidences.

- A case study is selected from the coverage redundancy domain of smart camera networks to explore the implication of our proposed high fidelity target detection model. Highlighting that the performance of coverage approaches are affected by the reliability of target detection results. A comparison conducted across studied models while employing: i) our model, ii) CamSim standard model. A previously unknown underestimation in the performance of coverage approaches is determined by employing CamSim's standard detection model.

It is important to note that this thesis is not investigating the best computer vision classifier for the task of target detection. Instead, by developing models of three detectors using three well-established feature extraction methods, we establish ground truth confidences, to explore the sufficiency of PIP in predicting detector's outcomes.

Additionally, in the line of questions distilled in this thesis, three particular types of objectives are studied as follows.

1. Easy to interpret and implement, meaning that by looking to the mathematical form of the solution, the relationship between inputs and the outcome become understandable. While this can ease the debugability of the solution, it also facilitates the implementation of the model further across simulation environments.

2. Accucary and fidelity of the outcomes are evaluated in predicting/reflecting the

ground truth variations.

3. Low computational overhead, which is ideal for a simulator to support run-time and online computations/applications.

1.4 Overview of the Thesis

The remainder of this thesis is structured as follows. Chapter 2 introduces smart camera networks and a need for simulation environments to facilitate the design and prototyping of new objectives. It surveys a list of widely used smart camera network simulation tools in research. A spectrum is drawn across studied simulators, highlighting an important trade-off between the fidelity of simulators outcomes and the corresponding computational expense. This chapter also introduces fundamental techniques for feature extraction as will be used to build the three detector models. Finally, it briefly introduces the coverage redundancy problem in smart camera networks as a case study in this thesis. In chapter 3, an architecture of a high fidelity target detection model is described. The core element of the model is established and formulated. The process of creating the image dataset using a real camera is described. Chapter 4 focuses on the feature abstraction models, represented by f. The aim is to predict ground truth PIP values, accurately from three physical parameters. Three state-of-the-art analytical approaches are used to obtain predictions. The accuracy of each prediction set is analysed using different evaluation metrics, and the distribution of the residuals. Chapter 5 focuses on the detector models, represented by g. By exploring the sufficiency of ground truth PIP in approximating three detectors outcomes. In this regard, first, three models of detectors are developed and applied to our image data set to create ground truth confidences. Next, the sufficiency of PIP in predicting ground truth confidences investigated by

developing a linear regression. Chapter 6 combines two partial model's f and g to be further deployed across the CamSim simulation environment. By exploring the sufficiency of predicted PIP in approximating the ground truth confidences. Also, it describes the standard detection model of CamSim and investigates the sufficiency of its outcomes in approximating the ground truth confidences. Chapter 7 explores the implication of our high fidelity target detection model using a case study from coverage redundancy in smart camera networks. Highlighting that the performance of coverage approaches are affected by the reliability of target detection results. Given smart cameras are equipped with an adjustable zoom lens, a set of coverage approaches reviewed and employed to maximise the redundancy network-wide. Finally, chapter 8 compares the outcomes of the high fidelity detection model with the CamSim standard detection model outcomes across studied coverage behaviours and discusses our findings. Finally, Chapter 8 concludes the thesis by reviewing the contributions of the preceding chapters and discusses prospects and directions for further research.

Chapter 2

Literature Review

This chapter surveys a list of Smart Camera Networks (SCNs) simulation tools that have been extensively used in research. A spectrum is drawn across the studied simulation environments, considering two different perspectives, *realism* and *generalism*. After that, the corresponding computational expense of each simulator is inferred from its identified degree of realism and generalism. A list of the main features of smart camera networks that often supported by simulation tools is described. According to the number of features that each simulator supports, they are categorised into three groups, subject-specific, multi-subject, and broad-subject or holistic. Depending on the level of details in which a simulator reflects real-world operations, their degree of realism is inferred. An important trade-off is identified between realistic but computationally expensive simulations and approximate but computationally cheap simulations. This trade-off opens the possibility for an alternative to augment abstract simulation tools with a higher degree of realism. Thereby creating a solution that captures both benefits, low computational expense with higher fidelity.

The fidelity of proposed models is studied in reflecting three *detectors' outcomes* as ground truth confidences. To build our models of detectors for the purpose of

this study, this chapter provides an overview of the standard techniques used in the feature extraction process. A combination of a feature extraction technique with an efficient distance metric as a (visual) similarity function forms the detector models for the purpose of this study. Feature extraction is the process of transforming visual information in the images into feature vectors, and these vectors are then compared against each other using a standard distance metric as a similarity function.

Finally, to explore the implication of the proposed model, we select a case study from *coverage redundancy* domain in smart camera networks. A brief introduction to this problem is provided with a highlight of the design considerations — a review of studied approaches provided in Chapter 7 of this thesis.

This chapter proceeds as follows. Section 2.1, first, introduces the smart cameras in the real world applications and motivates the need for camera network simulation environments to facilitate design and prototyping of the new models. Next, the main features that are often supported by existing simulation tools described and categorised in this section. Finally, a spectrum drawn across the listed SCN simulation tools under two different perspectives. Section 2.2, describes what the detector models in this study are? And how are they built? It introduces the fundamental techniques involved in feature extractions. Also, it provides a discussion comparing these techniques. Along with describing a similarity technique is used for building the models of detectors. Finally, Section 2.3, introduces the coverage redundancy problem in smart camera networks. This application domain is selected as a case study across this thesis. The review of the coverage approaches provided later in Chapter 7. In Section 2.4, we summarise this chapter.

2.1 Smart Camera Networks Simulation Tools

Smart cameras are embedded devices able to observe their environment, process the acquired images on-board, and communicate aggregated information and extracted knowledge with other devices. By operating in networks, their ability to adapt to changing conditions makes them robust, flexible, and resilient [132]. These multi-camera systems create an interdisciplinary field lies at the intersection of Computer Vision and Sensor Networks, raising research problems in the two fields that need to be addressed simultaneously [29], [82].

Deployment of large-scale surveillance systems in the real world is a significant undertaking that often faces several difficulties. It is both cost and time-intensive, which might even prohibit establishing such a surveillance system from the beginning. Moreover, in some cases, available empirical data is limited due to legal impediments [131]. To tackle these obstacles, and simulate a wide range of application scenarios, smart camera network simulation tools help to facilitate design, and prototyping of the models to be employed in real-world and have been extensively used in research [32, 59, 83, 9, 135, 30, 50].

A list of widely used camera network simulators inspired by [91], is surveyed under two different perspectives, **i.** degree of realism, and **ii.** degree of generalism with definition of each term descibed as follows.

- ***Degree of Realism*** refers to the levels of details in which a simulator reflects real-world operations [81].
- ***Degree of Generalism*** refers to the available camera features, i.e. functionalities supported by the simulator. This includes a range from subject-specific simulators which focus on a particular feature of camera networks, to a holistic

simulator which provides support across a wide range of key features.

2.1.1 Supported Features

The main features supported by the existing smart camera network simulation tools are divided into four classes within this study as follows.

i. *Computer Vision* features, refers to a list of image processing techniques, e.g. face recognition, pedestrian detection, and tracking supported by a simulator using both real-world and synthetic datasets [117].

ii. *Network Protocols* features, concerns with evaluating the networking aspect of the SCNs, where several cameras communicating with each other via single hop or multiple hops. By focusing on deep network protocols, it supports protocols such as routing, TCP/IP, and multi-casting, across both wired/wireless communication platforms [121].

iii. *Communication-Control* features, simulators often support different communication techniques for data exchange among cameras. Either they are having direct communication, i.e. unsynchronised and instantaneous without accounting for realistic problems,(e.g. transceiver-related collisions of data packets) or supporting realistic communication channels. For controlling, they often support either tracking hand-off of objects over multiple cameras [78, 48], or proactive controlling [108].

iv. *Resource Management* features, assuming cost-free data exchange or unlimited bandwidth or memory resources considered in some SCNs tools is ideal (not realistic). However, to reflect real-world SCNs operation, it is necessary for simulators to consider the constraints imposed by resource-limited platforms (e.g. battery-powered cameras). Thus, resource management feature usually supports power-consumption models for SCN hardwares [91].

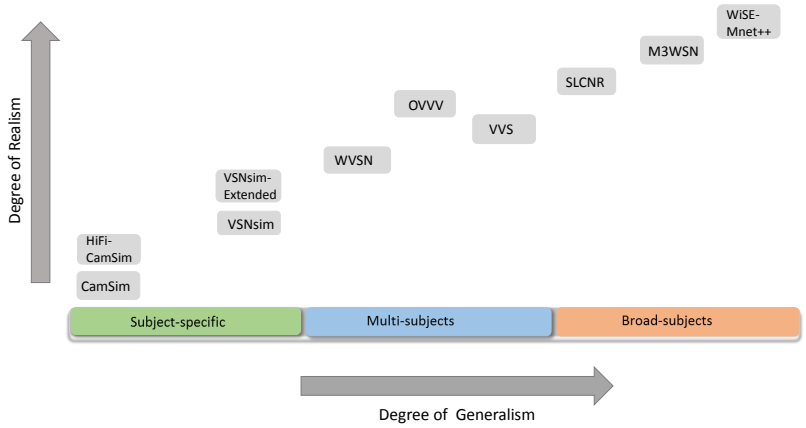


Figure 2.1: An illustration of a spectrum across a list of surveyed camera network simulation tools. Simulators organised according to two different perspectives including degree of realism, and degree of generalism

A summary of these simulators with the main features they support, demonstrated in Table 2.1.

2.1.2 Subject-Specific to multi-Subject Simulators

First, we review a set of available simulators with *subject-specific* to *multi-subject* degree of generalism. As described, these simulators often focus on testing/evaluating a particular or limited feature/s of camera networks while keeping other features abstract. Indeed, abstracting the details of other features results in incorporating fewer properties of realism in reflecting real-world operations compared to broad-subject simulations. Then, we move on towards more broader simulators focusing on realistic camera networking, supporting a more comprehensive range of key features in camera networks. Finally, according to the level of generalism and realism is supported, we further infer the corresponding computational overhead of these simulation tools.

Table 2.1: Classification of camera network simulators in terms of their main features.

Environment	Computer Vision	Networking Protocols	Communication-Control	Resource Management
VSNSim	–	–	Yes	–
VSNSim(Extended)	–	–	Yes	–
VVS	Pedestrian recognition/Tracking	–	Camera assignments/Handoff	–
OVVV	Video analysis	–	–	–
SLCNR	Pedestrian tracking	–	proactive controlling/Handoff	–
CamSim	–	–	Communication graph Vision graph	–
HiFi-CamSim	High Fidelity target detection	–	Communication graph Vision graph	–
WVSN	Area Coverage	–	direct communication	Energy consumption
M3WSN	Real-video transmission	Yes	Realistic-comm. channel	Energy consumption, Memory usage, CPU states
WiSE-Mnet++	Video analysis/ Tracking	Yes	Realistic-comm. channel	Energy consumption

The Object Video Virtual Video (OVVV) [114], is a deep computer vision simulation which use highly realistic, i.e. life-like virtual 3D scenes to emulate real-world operations. It operates on a commercial game engine to generate synthetic images of virtual scenes [46]. To increase the fidelity of the outcomes, they model some real video noise such as pixel noise, video ghosting(e.g. [116]) and employ them to synthetic video streams. It also supports a wide range of camera platforms including fixed, mobile, aerial. To support performance evaluation, it also generates automatic ground truth for each frame, including target centroids. However, by focusing on computer vision algorithms, the other features of a camera network mostly remain abstract or details of them is not provided, e.g. networking protocols. Furthermore, due to incorporating a large number of properties of realism, creating a new scenario is not straightforward and often requires to define an extensive amount of physical properties such as static objects, adding controls to define their behaviours, artificial lights, shadows.

In the context of subject-specific simulators, the Visual Sensor Network Simulator (VSNSim) [97] is another example; it focuses on coordination and control strategies in camera networks supporting only static camera platforms. Further, the extended version of the simulator [39] is released, which supports mobile camera platforms as well. The functionality of both tools is focused on the implementation of coordination and control algorithms such as market-based approach [30] while keeping details of other features such as networking protocols, image processing algorithms abstract. CamSim simulator [28] has got an abstract 2D environment with simulating moving targets as mobile points (e.g. vertices of a grid) in the field. Its main focus is on the development of collaborative algorithms to facilitate implementation and testing of distributed algorithms for self-adaptation and self-organisation of the

network. The simulator is also extendable to support the implementation of more sophisticated online learning applications (e.g. task exchange approaches) result in dynamic adaptation during runtime [58].

An extension to CamSim simulation tool is proposed in this work, for the task of target detection called Hi-Fi CamSim (High Fidelity CamSim) across this spectrum. While the proposed target detection approach retains the extendability and flexibility of original CamSim, it aims to improve the fidelity of the outcomes in reflecting real-world results. To achieve this, we incorporate a few more properties of realism, Euclidean distance of the target from the camera, camera's current zoom.

The Wireless Video Sensor Network (WVSN) [80] is another example of multi-subject simulator based on OMNeT++ platform [121]. Assuming cameras with a limited field of view, it focuses both on the area (k)coverage algorithms and efficient scheduling of visual sensors to reduce energy consumption. Assuming a mission-critical surveillance application, they considered a static 2D scene within objects as moving points, and the visual coverage of the scene is provided through 2D images (rather than video streams). Moreover, by assuming an ideal direct communication among camera neighbours, the realistic channel problems are relaxed across this simulator.

So far, we reviewed a set of subject-specific and multi-subject camera network simulators, focussing on a particular or limited feature/s of camera networks. On the one hand, keeping details of other features of camera networks abstract results in incorporating fewer properties of realism. It can be inferred that these simulators often come with low computational overhead. This motivates a generation of new scenarios and testing a new hypothesis. Also, the extension of the functionalities of these simulators to a practitioner's requirement is often not difficult. Although,

an exception of this statement is found in the OVVV simulator. Due to supporting realistic models for computer vision analysis, it is resource-intensive. Thereby, we infer the corresponding computational overload can be higher than other studied multi-subject tools. Moreover, due to *bundle-package* nature of this simulator, designing new scenarios is not straightforward and can come at extra cost.

On the other hand, they often lack *realistic* models for other camera features, such as camera resources, communication channels, which makes it challenging to implement and test our approaches under realistic networking conditions. This motivates to study more broader simulators, supporting realistic models across a wide range of SCN key features.

2.1.3 Broad-Subjects or Holistic Simulators

While alleviation of the real operating environment of camera networks allows for quick development and testing of distributed/collaborative algorithms, real-world SCNs need to account for a range of constraints imposed by resource-limited platforms. Hence, to predict the performance of the models under realistic conditions, a set of more sophisticated simulator developed. Often the aim is to represent a specific phenomenon in the real world while taking real camera networks constraints into account. This achieved by providing realistic models to support key functionalities/features of camera networks. In this context, the Mobile Multi-media Wireless Sensor Network (M3WSN) [85] simulator, which is based on OMNeT++ and Castalia frameworks developed to supports video transmission, control and verification in a set of mobile and fixed scenarios. By offering a variety of functionalities such as real-video processing, real communication channel models, resource management, and networking protocols, this environment becomes a network-oriented,

i.e. holistic simulator for smart camera researchers. The simulator supports a wide scope of IoT (Internet of Things), and smart cities applications that require visual and audio information, e.g. traffic monitoring, personal health care. It also offers some performance evaluation metrics for multimedia transmissions such as QoE (Quality of Experience), MOS (Mean Opinion Scores) to emulate the real-world operations. However, these evaluation metrics are not offered in other studied frameworks. Within the real-video transmission, the simulator also supports object detection and movement.

In the context of virtual vision simulators with slightly broader features support, Starzyk et al. presented a Software Laboratory for Camera Networks Research (SLCNR) [109] simulation environment, built on top of a game engine (Panda3D). Virtual cameras deployed in virtual environments generate synthetic video feeds from the scene that is fed into a vision processing module. A focus of this simulator is on target, i.e. pedestrian detection and tracking algorithms while supporting advanced rendering effects including shadows, lightening, and transparency for synthetic video streams. The simulator also implemented a range of communication strategies supporting inter-camera communications among the cameras, and hand-off using a synchronisation module.

The architecture of the simulator is described as a collection of modules that can communicate with each other over the network. Hence, the simulator can be deployed across a network of machines to support more significant scenarios (with a large number of virtual cameras) or more complex scenes.

While a direct relationship between the degree of realism and generalism identified across the studied simulation environment as it is illustrated in Figure 2.1, it is inferred from these categories that there is an inverse relation between them and

the computational overload of simulators.

Another example of a holistic simulator is the WiSE-Mnet++ [92] simulation environment, developed to support a variety of key features of smart camera platforms utilising both real-world and synthetic videos. To model energy consumption, each camera node operates with three-state duty cycles (active, idle, and sleep) model [93], each of which can be selected upon demand, e.g. when a processor is required to complete a task. While it supports direct communication for data exchange among cameras, using Castalia framework [11], it also supports realistic communications, e.g.transceiver models, and channel models. By extending the functionality of the sensing module, the simulator further supports real-world videos as well as synthetic videos generated via game engines.

Given broad-subject simulators, with a higher degree of realism, it is inferred that the coressponding computational overhead of these simulators would be higher than subject-specific tools. While, having more realistic, broad-subject simulators improve the accuracy and fidelity of the outcomes, As the size of the network grows, the computational overhead increases noticeably in a way that it can not be run on one machine with given hardware resources.

An instance of this case can be found in Virtual Vision Simulator, (VVS) [82]. A particular focus is to emulate the real-world surveillance system in a virtual 3D scene by including realistic models for cameras, video processing, and pedestrian tracking parts. Even though the simulator is categorised as a multi-subject class, due to incorporating a high amount of realism properties, the simulator only can scale to a network with a maximum size of 16 cameras to be able to run on a single machine with given hardware resources. Although, this problem is tackled later in their following work [109], by distributing the computational load across

multiple machines rather than one machine. However, the proposed solution is not cost-efficient.

2.1.4 The Trade-off between Fidelity and Computational Expense

So far, we studied different functionalities of some widely used SCN simulation tools according to their degree of realism and degree of generalism. It was found that increasing the degree of realism and generalism in SCN simulators often leads to higher computational overhead. While using the realistic models can improve the accuracy and fidelity of the outcomes in reflecting real-world operations, incorporating many realism properties can make simulation tools cumbersome and slow. For an standard use case this could vary e.g. taking up to three times slower than abstract one. On the other side, using abstract models with low computational expense allows for the development of new theories with the results easy to interpret. However, due to the simplified nature of these models, which remove details and hence can introduce errors, the outcomes can be imprecise and have room for improvement in terms of their fidelity.

There is a trade-off between abstract simulators with approximate models, and thereby, low computational overhead, and more realistic simulators with more realistic models, thus, high computational overhead.

Although, fidelity is the degree of similarity between the simulation and reality, it is critical to have a detailed and precise capability to measure that fidelity. In terms of fidelity quantification, there are several types of metrics available [61]. Objective measurement of simulation fidelity is a metric that attempts to compare the simulated objects/tasks with the corresponding referent or real-world environment. Due to the scope of this work, where only a few number of physical parameters (i.e.

properties of realism) incorporated, we apply the objective measurment to quantify the fidelity. A review of mathematical models on fidelity measurements can be found in [98]. In this way, fidelity is measured by a binary scoring system. Simulation conditions are evaluated by either 0 or 1, with 0 meaning that simulation does not duplicate the real-world conditions, and 1 indicating that the simulation does reproduce the real-world conditions. Simply, averaging those ratings together provides an assessment of the overall fidelity. It is important to note that there are several factors that affect the computational complexity of these simulations, e.g. the optimisation algorithms utilised, the number of inputs/properties of realism incorporated, and/or the number of nested loops. However, for an standard use case, there is limited information available in the literature for quantifying the actual computational complexity. The corresponding computational expense of simulations is then inferred according to their degree of realism and generalism.

2.2 Detector Models for Target Detection

This section is an introduction to general feature extraction techniques for the task of target detection. Within this thesis, target detection is considered as a classification task [113], where the posterior probability of similarity returned by the classifier/detector is interpreted as the camera's confidence of correct identification. Then the location of the object is selected based on the density of re-identified features.

Feature extraction methods that studied in this work are reviewed in Chapter 5. A combination of a feature extraction method with a visual similarity technique (i.e. Euclidean distance metric) form a *detector model* for our study.

2.2.1 Feature Extraction

Feature extraction refers to the process of transforming visual information in images into compact vectors, known as features, i.e. descriptors. In this way, it is desired that the visual features be fairly invariant to scaling, rotation, and illumination changes [34].

Feature extraction methods can be divided into three main classes. Methods based on *local features* which detect the interesting regions of each image and describe the local information of them into features (i.e. visual vectors) using human-engineered techniques. To represent the visual information in a more compact way, *global features* aggregate local visual information into a single image representation. Finally, deep features that are based on convolutional neural networks.

Local Features

To identify the similarities between two images using local features, first, relevant patches, i.e. interesting regions of a given image is identified. Then, its visual content is transformed into features, i.e. descriptor vectors. Second, features (relevant patches) are compared against each other to find a common pattern within different images [66]. To have a more robust comparison between features, these methods rely on practitioner's expertise such as edge detection, and corner detection algorithms to describe the visual content of relevant patches into invariant features.

For the purpose of this study, we selected lthree ocal feature methods, i.e. SIFT (Scale Invariant Image Transform) [62], SURF (Speeded Up Robust Feature) [6], and ORB (Oriented FAST and Robust BRIEF) [88]. The details of them will be reviewed later in Chapter 5 of this thesis.

Global Features

Within local feature methods, each image may contain a significant amount of interesting regions, which increase memory requirements. To reduce the memory requirement of local features across large image datasets, global features aggregate multiple local features into one single global vector such as bag-of-words (BOW) [103].

Deep Features

Deep features are visual representations of an image obtained from Convolutional Neural Networks (CNN). CNN's architecture typically consists of an input layer, which processes input images, an output layer, which provides the output results, and some hidden layers (i.e. known as nodes or neurons) in between. Each hidden layer takes the output of the previous layer and makes some transformation and forwards the new data to the next layer. In this way, the learning is performed by computing the prediction error (between the outputs of the CCN and the expected results) and backpropagating it to the network to improve the accuracy of the outcomes.

Although, CNNs are powerful techniques that have pushed the boundaries of what is possible in many computer vision domains, e.g. image recognition, image classification, speech recognition, however, there are some considerations need to be taken into account. A complete list of ldeep feature's imitations reviewed in [63].

Deep features are learnt directly from observations of the high-resolution input images (i.e. training set). Meaning that these features are specific to the training set. Therefore, special care must be taken in the selection and size of the training dataset to ensure it performs well with new images (different from the training set).

This type of generalisation, also referred to as *extrapolation*, which requires going beyond a space of known training examples [64]. While applying limited training dataset, may result in the risk of *overfitting* to the training data, thereby, not to generalise for the task at hand [77], having a huge training sets requires substantial computational power during the training phase.

Meanwhile, in the context of generalism, local features such as SIFT algorithm benefits from being general, and not class-specific, meaning that they perform the same for any given images [77]. This empowers these methods to be used in applications such as image-stitching, 3D recognition, where CNNs yet perform poorly.

Local feature methods have full transparency, where one can judge whether the solution will work outside the training set. Although they require a practitioner's knowledge in extracting features, if a solution fails, the parameters can be easily adjusted to perform well across a broad range of images. However, due to *black box* nature of neural networks, the relative opacity, where the contribution of each hidden layer in a complex network is not clear, is still unsolved. In contrast to local features, with CNNs, manually tweaking of the model's parameters would be too difficult.

Furthermore, there are some emerging application domains in computer vision such as 3D vision, panoramic stitching, 360° cameras, that CNNs are not yet well-established.

A broad comparison between local feature methods, (i.e. referred to as traditional computer vision), and deep feature methods,(i.e. referred to as deep learning) provided in their recent work [77].

2.2.2 Visual Similarity

To compare how alike are two features from two different images, a similarity function is evaluated between their descriptors. With this study, we consider a standard Euclidean distance metric as a similarity function, which is easy and fast to implement. In this way, assuming x, and y as two features, the similarity between them is computed as follows.

$$d_E = \sqrt{\sum_i (x_i - y_i)^2}$$

In this way, the smaller the distance is, the more similar the two vectors are.

A more complete review on visual similarity metric learning approaches can be found in [19].

In this thesis, we use local feature methods as feature extraction part and combine it with a standard distance metric (i.e. Euclidean distance) as a visual similarity technique to build up three models of detectors termed as *detector models*.

2.3 Coverage Redundancy

A case study is selected from coverage redundancy problem, formalised as k-coverage, in SCNs, to explore the implication of our proposed high fidelity abstract target detection model. This section provides an introduction to coverage redundancy problem in camera networks and a review of some proposed approaches to either ensure the specific level of redundancy or maximise it across the network.

Coverage redundancy problems concern with covering a region of interest, i.e. target with at least k cameras at any given time. The provided redundancy is vital for fault tolerance and the acquisition of multiple perspectives of targets across the

network. Given cameras are equipped with an adjustable zoom lens, we studied the impact of off-line, on-line, and on-line (reinforcement) learning-based approaches on the improvement of redundancy across all targets network-wide. The details of studied approaches, along with the obtained results, are described in Chapter 6 of this thesis.

2.3.1 Design Factors

In general, for coverage problems in camera networks, there are some design factors that need to be taken into account as follows.

- ***Coverage type***. In general, the coverage problem in camera networks can be classified into three main types; point (object) coverage, area coverage, and barrier coverage [128]. Within the case study of this thesis, we are interested in the point coverage problems in an abstract camera network simulation environment, where objects are often modelled as a set of discrete points within a surveillance field.

- ***Deployment strategy***. Deployment methods usually concern with how a cameras network is constructed. Generally, these methods lay across two main classes. Deterministic camera placement, which can be ideal for small to medium size scenarios. Random deployment, which often applied to the larger networks with more than one hundred cameras, or in the case of hostile environments.

- ***Degree of coverage***. In point coverage problems, the degree of coverage describes the number of cameras covering a specific point, i.e. region of interest.

- ***Modelling Objects Movement Pattern***. The term, *movement pattern* can be attributed to high-level process knowledge derived from low-level trajec-

tory data as stated in [57]. The term*trajectory or pattern* itself, refers to the representation of a point object's movement as described in [41]. Within the surveillance field, objects can adopt different mobility patterns, such as flocking movement pattern, scripted movement pattern, uniform movement pattern.

In our selected case study, the type of coverage is point coverage, i.e. target coverage, where all targets in the networks can take different mobility patterns. The deployment strategy is deterministic with three different camera layouts described in Chapter 7. The particular concern is to cover each target with as many cameras as possible. Therefore there is not a certain degree of coverage is defined in our problem; instead, we focus on achieving the highest possible level of k-coverage across the network.

2.3.2 A Review of Coverage Redundancy Approaches

While surveillance is still an important aspect of smart camera networks, other application areas have also emerged. (k-) coverage optimisation is one important application of smart camera networks. Besides *coverage optimisation* smart camera networks can follow goals such as object tracking and recognition, optimal placement of cameras in the field which are not in the scope of this thesis. Typically in k-coverage problems, a desired fixed value of k is used, and the challenge is to ensure that at least k sensors cover all objects with sufficient confidence. To ensure a certain level of coverage, k across the network all time, some researchers translate the k-coverage problem to a *SET-COVER* problem [44], [1], [17]. Thereby, the problem is to determine the minimal set of active visual sensors that provide required k-coverage in the network through a central controlling system. However, since the problem modelled as an optimisation problem which is proven to be NP-hard [35],

a set of approximation algorithms proposed to solve the SET-COVER problem [17]. With a slightly different perspective to the problem, Huang et al. [47] study the k-coverage problem as a decision-making problem in sensor networks. Their work aims to evaluate for a given k whether the sensor network is k-covered. This was studied by exploring the perimeter coverage of the visual sensors, considering both fixed and adjustable sensing ranges. The authors claim that the whole area is k-covered if each sensor in the network is k-perimeter covered. In their recent work, Esterle and Lewis [27] investigate k-coverage on an object level. Where the goal is to coordinate a set of mobile cameras with a directed field of view to maximise the number of targets for which the network achieves k-covered, over time.

In our case study in Chapter 7, the main concern for each directional camera is to determine an appropriate zoom in a way the coverage redundancy is maximised across all available mobile targets network-wide. Generally, the performance of a coverage strategy is highly influenced by the level of detail captured by a camera. Thereby, the high fidelity target detection model developed and analysed across this thesis becomes a fundamental requirement for this application in correctly detecting a target within a camera's FoV prior to performing coverage strategies.

2.4 Summary and Conclusions

This chapter first provided a survey across a set of smart camera simulation environments. The simulators are studied under two different perspectives; the degree of realism, and degree of generalism, from which the computational overhead of each simulator is inferred. An important trade-off was identified between accurate but computationally expensive simulation environment and approximate but com-

putationally cheap simulations. At one end of the spectrum, Figure 2.1, realistic models aim to represent a particular phenomenon in the real-world, while taking many properties of realism into account. At the other end of the spectrum, simple and abstract models supporting limited features by capturing a limited number of real-world properties within an abstract environment.

The identified trade-off opens the possibility for an alternative to augment abstract simulation tools with a higher degree of realism. Thereby, creating a solution that captures both benefits, low computational expense with higher fidelity in reflecting real-world outcomes.

To achieve this we augment CamSim simulation environment as an abstract SCN simulation tool with a high fidelity target detection model, and further, explore the implication of the employed new model across a case study from coverage redundancy application of smart camera networks.

In the context of building detector models as ground truth for our further evaluations, the chapter provided an introduction to feature extraction techniques, by dividing them to three main classes. Although convolutional neural networks were powerful techniques that pushed the boundaries forward in many fields of computer vision, however, there was still some limitation with these techniques that need to be taken into account. Furthermore, there are some emerging application domains in computer vision such as panoramic imaging, a 3D vision that CNNs can be supplemented by other techniques, while local features are well-established.

Thus, we select three local feature methods (with details reviewed in Chapter 5) for feature extraction part and combine them with a brute-force search with an efficient Euclidean distance metric to form our three detector models. Since these detectors later in chapter 5 will be applied across our lab-based image dataset, their

outcomes are considered as ground truth for our study.

Finally, a case study is selected from *coverage redundancy* domain in smart camera networks to explore the implication of our proposed model. Highlighting the importance of having reliable target detection results as a prerequisite to coverage redundancy applications. The chapter provided a brief introduction to the coverage redundancy problem in smart camera networks. A review of studied coverage approaches is provided in Chapter 7 of this thesis.

Part II

Model Development

Chapter 3

A High Fidelity Abstract Target Detection Model

In this chapter, the architecture of a *high fidelity abstract* model of target detection is established and described. Target detection task studied in this thesis is related to confidence representing the probability of the target being correctly detected in the right location within the camera's field of view in an abstract simulation environment. As described in Chapter 1, the abstract simulation environments often ignore the details of the scene by simply modelling targets as moving points (e.g. vertices of a grid) across a surveillance field. We aim to produce an accurate estimation of detection, incorporating only a small number of physical properties, capitalising on both benefits, low computational overhead and high fidelity of the outcomes in reflecting the real-world results.

To achieve this, we propose a novel architecture in this chapter, by introducing an intermediate point of representation, called Patch Image Proportion, PIP. Within the architecture, PIP is a core element, capturing a ratio of the pixel density of a patch (i.e. projection of a target on the image sensor of a camera) to an entire image. PIP *decouples* the architecture into two partial models as depicted in Figure 3.1.

Figure 3.1: An architecture of a high fidelity abstract target detection model. Each box in the diagram represents a set of available properties and the connecting arrows in between represent functions that approximate models of mapping input data to output data. PIP, as an intermediate point of representation, decouples the architecture to feature abstraction models, represented by f and detector models, represented by g.

The feature abstraction models, represented by the function f, explicitly focuses on predicting the ground truth PIP using three physical parameters, size of the target, distance from the camera, and camera's current zoom. The detector models represented by g, mainly focused on evaluating the sufficiency of PIP — both ground truth and predicted — in building high fidelity models of three selected classifiers, also termed as *detectors*.

Decoupling the architecture brings some crucial benefits to the design of the model. It allows for significant *flexibility* and *modularity* in the design, which empowers practitioners to be able to select the model's features individually and independently to their requirements and camera settings. Furthermore, the modularity in the design of the model facilitates the extension of the model's functionalities for further network requirements.

In other words, this decomposition enables composability of different functionalities required for detection and tracking in smart camera applications. This, in turn, lifts the limitations imposed by models focussing on certain optical properties of specific camera types and models relying on specific tracking and detection

algorithms, e.g. a particular classifier.

These benefits, however, come at the cost of adding an extra layer of prediction errors to the outcomes of the model. Impacts of these extra errors on the fidelity of the predictions will be analysed in the next three chapters of this thesis.

In decoupling the architecture, it is important to ask ***is PIP, as an intermediate point of representation, sufficient to undertake the decomposition?***

Throughout this thesis, the sufficiency of PIP is explored in building high fidelity models of three selected detectors' outcomes as ground truth confidences.

In Section 3.1, we first highlight the need for augmented abstract simulation tools to capitalise on both benefits, low computational expense with higher fidelity in reflecting real-world outcomes. In Section 3.2, we describe the studied target detection task within camera network simulation environments together with camera network terminology. An insight into the imaging geometry of the profiling experiments along with a model of a camera's field of view is described and formalised in section 3.3. Section 3.4 includes the architecture of a high fidelity abstract model, and a set of four follow on research questions raised from this architecture. In Section 3.5 we describe our profiling experiments conducted with a real camera to establish a ground truth PIP for further statistical analysis, and finally, in Section 3.6, we summarise the findings of this chapter.

This chapter provides both an introduction to the problem and a theoretical underpinning for the results presented later in the thesis.

3.1 Motivation

Existing camera network simulation tools often reflect real-world information in different ways. As described earlier in Chapter 2, at one end of the spectrum, complex

models aim to represent a particular phenomenon in the real-world, while taking many properties of realism into account. At the other end of the spectrum (e.g., Figure 2.1), simple and abstract models only capture a limited number of real-world properties within an abstract environment [14]. While using the complex models can improve the accuracy and fidelity of the outcomes in the sense of resembling real-world operations, incorporating many realism properties can make simulations of the environment cumbersome and slow.

On the other hand, using abstract models allows for development and verification of new theories with easy to interpret results. However, due to the simplified nature of these abstract models, details are removed. Hence, errors are introduced, which can lead to imprecise outcomes and room for improvement in terms of their fidelity. In general, it is not clear what impact the creation of such simplified abstract models has on the ability of the simulator to capture real-world behaviour. This unclear impact is also a general open concern in agent-based modelling [36]. This gives rise to an important trade-off between realistic but computationally expensive simulations and approximate but computationally cheap simulations. This trade-off opens the possibility for an alternative to augment abstract simulation tools with a higher degree of realism. It is thereby creating solutions that capture both benefits, the low computational expense with higher fidelity. This study incorporates three physical parameters, as properties of realism, for target detection estimation task within the *abstract* simulation environment, CamSim.

3.2 An Introduction to the High Fidelity Abstract Model

The studied target detection task is related to a *confidence* representing the probability of the target being correctly detected in the right location within the field of view of the camera. In general, target detection is a classification task [113], where the posterior probability returned by the classifier is interpreted as the camera's confidence of correct identification. However, the idea behind developing a high fidelity abstract target detection model is to estimate this probability across abstract simulation environments accurately. Indeed, incorporating more properties of realism such as lens distortions, camera's aperture, environmental lighting leads to have more realistic models. However, in this thesis, we are not aiming to reflect a particular phenomenon of the real world by incorporating many properties of realism. Hence, produce a realistic target detection model. Instead, by isolating the model to a small number of relative physical parameters aim to produce accurate estimations, while keeping the computational expenses low and improve the fidelity of the outcomes.

3.2.1 Preliminaries

In this work, we consider the terminology proposed by Greenleaf [38], e.g. focal length, and angle of view.

The *Focal Length (fl)* refers to the distance between the lens and the focal point. In turn, the focal point is the point at which the parallel light rays converge to form a sharp image of an object observed through the convex lens.

The *Angle of View (AoV)* of a camera describes the amount of a given scene that is captured by the respective camera. To compute the angle of view we utilise the

camera's focal length fl and the dimensions of the image sensor res_i.

The term *target* within the real-world, refers to a generalised form, including human-made objects (e.g. vehicles, toys, buildings) that have sharp boundaries and are independent of background environment [20].

However, the definition of the terms target and focal length within the abstract simulation environment, CamSim are slightly different. Within CamSim, for simplicity, details of targets are ignored, i.e. as vertices of a gride with a unique radius. Within other abstract environments, the details could be ignored by using a bounding box with the edge of q, which denotes the region of interest.

The term focal length is interpreted as the optical zoom in the real-world operations. Assuming cameras are equipped with an adjustable zoom lens, within CamSim, the camera's current zoom corresponds to the radius of its FoV (i.e. r_i as defined in Equation 3.2).

Assuming a directional camera c_i with a given image sensor resolution of res_i, the physical properties of our target detection model include the camera's current zoom, z, the size of the region of interest, i.e. a target, q, the distance between the camera and the object, d. These physical properties are inspired by 2D modelling a camera's FoV within CamSim environment as described in the next section.

We call the projection of the target inside a camera's FoV to the camera's image sensor, a *patch* and its resolution is denoted by res_j. res_j expresses a pixel count of that particular patch on the image plane. A pixel is defined as the size of the smallest, clearly observable object with distinct boundaries.

The proportion of image sensor surface, res_i that is occupied by the patch surface res_j, is called *PIP* and simply defined as the ratio of $\frac{res_j}{res_i}$. In this study, the PIP metric, capturing a ratio of the pixel density of a patch to an entire image is

a core element. It is an intermediate point of representation across our proposed architecture, which decouples the architecture into two partial models.

3.3 Camera's Field of View

Each camera c_i has its own FoV modelled as a circular sector representing the portion of the environment observed by that camera [2]. A visualisation of the imaging geometry of our profiling experiments in a three-dimensional environment and a two-dimensional modelling of a camera's FoV is illustrated in Figure 3.2.

In general, the camera's current zoom has an important impact on the total number of acquired pixels on the target. While the total number of pixels for an entire image acquired by the camera stays the same, using a narrow zoom, i.e., zooming in, leads to having relatively high pixel count across the target region compared to the covered area. The pixel count, in turn, drops when zooming out and hence widening the angle of view. Across the simulation, we assume all cameras to be mounted at the same height, allowing us to simplify the model for further analysis in two dimensions only.

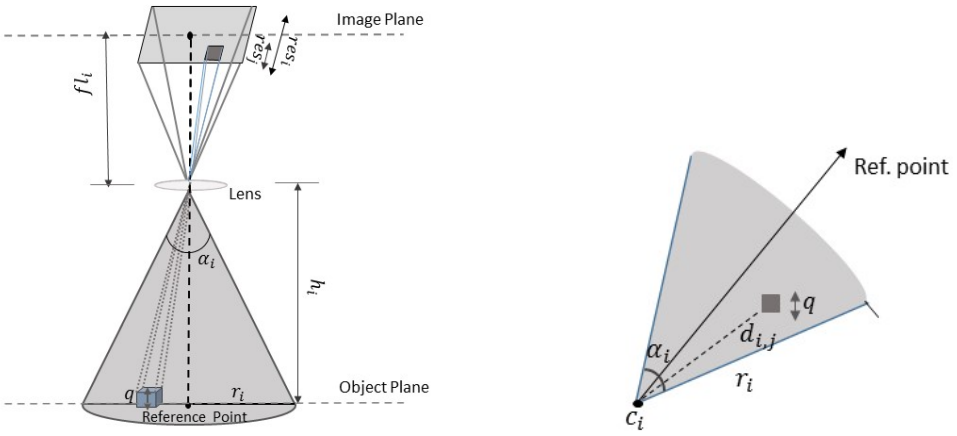
Thus, a camera's FoV with regard to the fixed reference point on the object plane is determined by its angle-of-view α_i and the range r_i representing the depth of the camera view in the 2D modelling of the FoV. The angle of view of a camera α_i at a given discrete time interval of t is defined by

$$\alpha_i(t) = 2 \times \arctan(\frac{res_i}{2 \times fl_i(t)}) \tag{3.1}$$

At the same time the range of FoV r_i is defined by

$$r_i(t) = h_i \times \tan(\frac{\alpha_i(t)}{2}) \tag{3.2}$$

Where, h_i refers to the distance of a camera from the object plane.



(a) 3D (b) 2D

Figure 3.2: Figure(a), an illustration of the imaging geometry of the lab-based experiments using a real camera in a 3D environment. The image plane demonstrates the projection of the patch on the surface of the image sensor from a front view. The object plane refers to the standard coordinate system on which objects move, e.g. ground. Figure(b), a 2D modelling of a circular sector of a cameras FoV within a simulation environment. an arbitrary object inside

Across the simulation, CamSim, each object o_j on the object plane/surveillance field, (i.e. a common coordinate system), has a location $\vec{o_j} = (x_j, y_j)$ and moves in straight vector with a constant velocity $\vec{v_j}$ all time.

$$\vec{o_j}(t+1) = \vec{o_j}(t) + \vec{v_j} \tag{3.3}$$

As shown in Figure 3.2)(b), for simplicity, the appearance details of an arbitrary object are abstracted by adding a bounding box around it. The distance between the target and the camera is considered as Euclidean distance metric, which is demonstrated as $d_{i,j}$ in Figure 3.2(b).

3.4 Architecture of the Model

Within abstract simulation environments, the important question is, how to estimate the probability of correctly detecting a target within a camera's FoV. To answer this question, we develop a high fidelity target detection model that captures both benefits, i.e. low computational expense with a higher fidelity of predictions.

Here, we establish and describe an architecture of a high fidelity abstract model of target detection using only a limited set of physical properties as inputs to the model. A visualisation of foundation components of the model's architecture is illustrated in Figure 3.1. Each box in the diagram represents a set of available properties and the connecting arrows in between representing functions that approximate models of mapping input data to output data. Within this architecture, there is a PIP component as an intermediate point of representation of the architecture. This point, decouples the architecture into two partial models, namely *feature abstraction models*, and *detector models*. As shown under each arrow of the Figure 3.1, decomposing different camera settings from a variety of computer vision classifiers (as shown, local features, deep features) brings significant flexibility and modularity in the design of the model. As described earlier in this chapter, this flexibility then empowers practitioners to swap these features individually and independently to their requirements and camera settings.

Before exploring a solution space in our approach, we take a closer look at the individual components of the model as illustrated in Figure 3.1:

- ***Physical Properties*** component is comprised of a set of inputs to the diagram including *the target size, distance to the camera, the camera's zoom.*

- ***PIP*** component, is Patch Image Proportion, is a core element in our approach,

capturing a ratio of the pixel density of a patch to an entire image.

- ***f, Feature Abstraction Models*** are the approximation models that predict ground truth PIP values from physical properties (q,d, and z). Where the aim is to compress the properties of three inputs into only one output, PIP.

- ***g, Detector Models***, is a set of three detector models for each of ORB, SIFT, SURF i.e. feature-extraction techniques.

- ***Predicted Confidence*** component is a combination of f and g two partial models that forms the high fidelity abstract model of target detection to be deployed across the CamSim.

We can, therefore, distil the follow on research questions that we aim to answer in the next three chapters of this thesis as follows:

- **i.** Does any correlation exist between physical parameters as a set of independent variables and ground truth PIP as a dependent variable?

- If, the answer to the question is positive, **ii.** How well can physical parameters approximate the empirically verified PIP value?

- **iii.** Does any linear correlation exist between both ground truth and predicted PIP (produced by f) with each set of ground truth confidences?

- **iv.** Is PIP sufficient in building high fidelity models of detector's outcomes (i.e. ground truth confidences)?

3.4.1 Feature Abstraction Models, f

To address the first two research questions **i** and **ii** above, we propose the first part of the architecture. Given the ground truth PIP (obtained from the lab-based

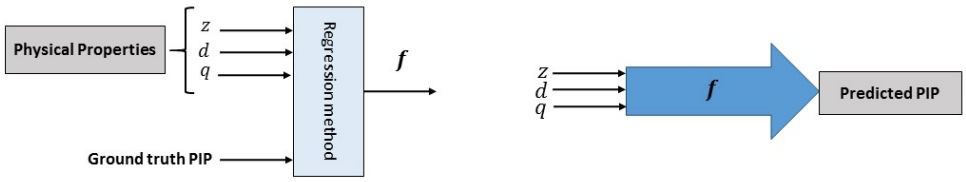


Figure 3.3: Block diagram of a process of predicting (ground truth) PIP using three physical properties, (z, d, q). Three main regression methods studied here; namely, Support Vector Regression, Symbolic Regression, and Multi-linear Regression will be replaced with the regression method box.

experiments described In Section 3.5), we first focus on the development of a set of approximation functions, called f. With f, we aim to accurately predict ground truth PIP values using three predictors known as physical properties. To achieve this, a set of three state-of-the-art regression methods were applied on the data set, aiming to map the physical properties box to the (ground truth) PIP box.

A visualisation of the approximation process, given three physical parameters, is shown in Figure 3.3. Throughout Chapter 4, the performance of each regression method with more details on the accuracy of the predictions will be analysed and described. The whole process is termed as feature abstraction, since it abstracts/compacts three input features into one output feature, PIP.

3.4.2 Detector Models, g

We investigate the sufficiency of PIP — both the ground truth and predicted — (obtained from f), in reflecting/predicting the detectors outcomes, i.e. ground truth confidences in the second part, represented by g. To achieve this, first, we develop three models of detectors, i.e. ORB, SIFT, and SURF. This includes selecting a feature extraction method and combine it with a visual similarity function (i.e. a brute-force search with an efficient evaluation of the Euclidean distance). Next, we

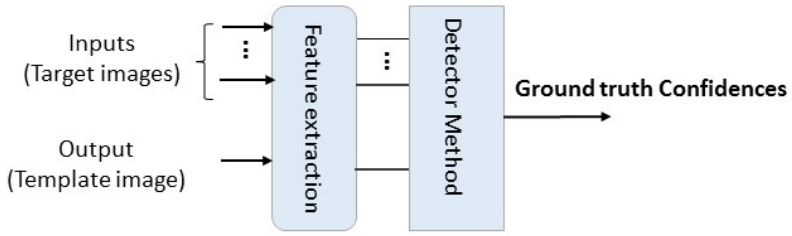


Figure 3.4: Block diagram of a process of obtaining the ground truth confidence as a probability of detecting an object; namely, a template image across the entire target image. The detectors run on pure images captured by a real camera at lab environment. Thereby, the outcomes of the process produce the ground truth confidences.

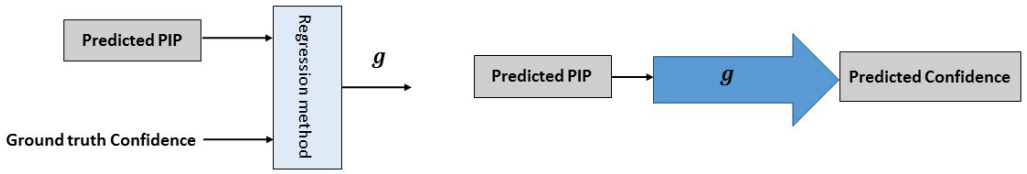


Figure 3.5: A Block diagram is demonstrating the process of obtaining predicted confidence values from predicted PIP (obtained from f partial models). An ordinary linear regression developed between predicted PIP values as an independent variable and each set of ground truth confidences as a dependent variable.

apply them on our image data set, to produce ground truth confidence values as depicted in Figure 3.4.

We first investigate the sufficiency of the ground truth PIP, in reflecting three sets of ground truth confidences. This investigation is useful when the high-quality real or synthetic images of the scene are available. For the case of abstract simulators, where the high-quality images of the scene are not available, we explore sufficiency of predicted PIP in approximating the ground truth confidences.

A combination of f and g is deployed across the CamSim abstract simulator as a high fidelity abstract model of target detection. A visualisation of the partial model g is depicted in Figure 3.5.

3.5 Impact of Physical Properties

In this section, with a particular focus on the first part of the architecture (Figure 3.1), we establish the ground truth PIP to explore the impact of physical properties. To achieve this, we conduct a set of profiling experiments with a real camera in a lab environment. Across our experiments, we use eight distinct objects with varying sizes q, at the range of $[0.66m - 0.10m]$, each of which forms a different size of the patch. The camera is equipped with six discrete optical zooms to correspond to 6 varying focal lengths in the range of $[10mm, 15mm, 24mm, 50mm, 70mm, 85mm]$.

The camera used across all our experiments is Canon EOS 7D, including a CMOS colour image sensor with the size of $22.4mm \times 15.0mm$ and a resolution of 5184×3456 pixels. In our experiments, we consider ten equally increasing distances in the range of 1-10 meters (e.g. $1m, 2m, ..., 10m$).

We placed the camera in a laboratory room free of obstacles; each experiment starts by selecting a certain object with a known size and is followed by capturing pictures of the object at six different optical zooms by employing six varying focal lengths. The entire experiment is repeated ten times trying the different distances in the range of 1-10 meters (i.e. $1m, 2m, ..., 10m$) resulting in 60 different images with varying pixel densities for that certain object. An example of this setup is illustrated in Figure 3.6.

To create the ground truth PIP, given all captured images, a small region including an object of interest is extracted from each image representing a *patch*. In our profiling experiments, patches have got different pixel density from each other, employing varying focal lengths, distances and sizes. An example is demonstrated in Figure 3.7 six distinct optical zooms applied on a camera while keeping distances to the object and the size of the object the same. Thus, the numerator of the PIP

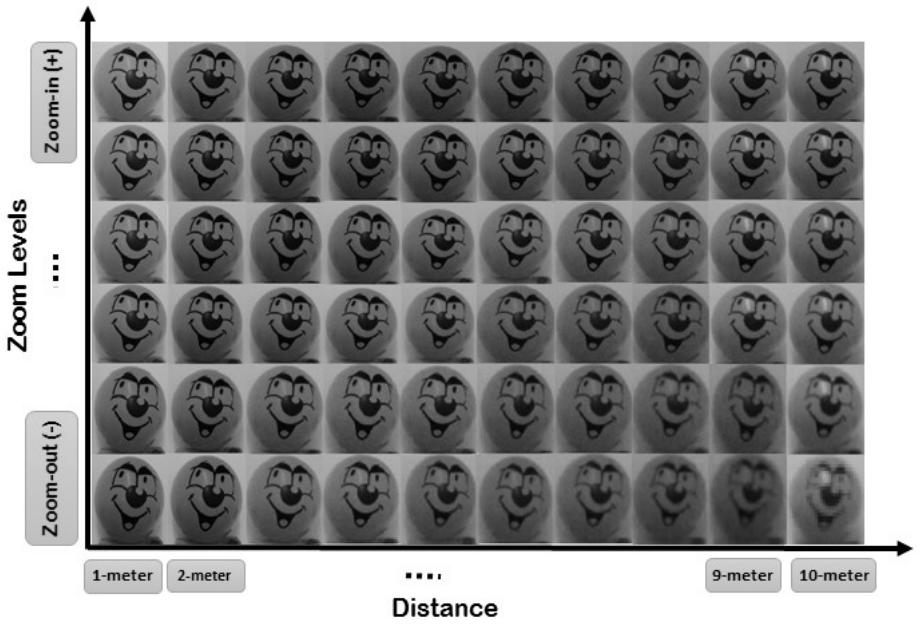


Figure 3.6: A mosaic of 60 different images of the same object within varying pixel count. The x axis shows the distances in 1-meter steps and for ten steps and the y axis demonstrates six employed optical zoom levels. As the distance increases and the camera's current zoom gets wider the pixel density of the patch, i.e. the ball, drops noticeably.

res_j, determines the patches pixel density and a fixed value of res_i as the resolution of the utilised image sensor, which is 5184×3456 pixels, forms the denominator of the PIP. In this way, each image has its own unique PIP value; creating our *ground truth* PIP across the dataset.

Indeed, the pixel density of the obtained ground truth PIP across these experiments relies on the given camera's specification, such as camera's image sensore resolution, optical zoom lens distortion, etc. Therefore, using different type of camera may results in slight difference in the value of ground truth PIP. Note, the flexibility in design of our model allows to select more specific characteristics of a camera at the time of thses calculations.

(a) $fl = 85mm$ (b) $fl = 70mm$ (c) $fl = 50mm$

(d) $fl = 35mm$ (e) $fl = 24mm$ (f) $fl = 15mm$

Figure 3.7: An example of six images of the same object of interest (i.e. the ball), employing six different optical zooms from left to right, image (a) with the narrowest zoom and longest focal length to image (f) with the widest zoom and the shortest focal length. The distance from the camera is 1-meter for all images. The pixel density of the region of interest from left to right is, 2904×2850 pixels, 2646×2563 pixels, 1946×1898 pixels, 1413×1353 pixels, 1023×990 pixels, and 666×627 pixels.

3.6 Conclusions and Discussion

In this chapter, an architecture for a high fidelity abstract model of target detection has been established and described. The model aims to produce an accurate estimation of detection across abstract camera network simulation environments using only a small number of physical parameters. These parameters were inspired by 2D modelling of a camera's field of view across the simulation environment. Indeed, incorporating more properties of realism leads to having more realistic models. However, in light of the trade-off, this could also risk the interpretability of the model and increase the computational expense. Therefore, by isolating the model to a small number of realism properties, we aim to capture both low computational overhead and high fidelity of the outcomes.

An intermediate point of representation, PIP metric was introduced, which cap-

tured a ratio of the pixel density of a patch to an entire image. Impacts of the physical parameters were investigated on the pixel density of the ground truth PIP by conducting a set of lab-based experiments using a real camera. In this way, the ground truth PIP were established for further analysis across this thesis.

It was demnstrated that PIP as a core element of the model decoupled the proposed architecture into two partial models, feature abstraction models, represented by f, and detector models, represented by g. The decomposition is useful in bringing flexibility and modularity within the design of the model. It also lifts the limitations imposed by models focussing on certain optical properties of specific camera types and models relying on specific tracking and detection algorithms, such as a particular classifier.

In addition to the research question in Chapter 1, a set of four follow on research questions were distilled. The next two chapters of this thesis will be focused on the two separate partial models by exploring explore their functionalities while addressing the relevant research questions. In Chapter 6 a combination of these two partial models is developed to be deployed further across the CamSim simulator.

Chapter 4

Feature Abstraction Models

This chapter builds upon the architecture described in the previous chapter, explicitly focusing on the feature abstraction functions, f. As illustrated in Figure 3.1 Chapter 3, within this part, a set of predictive models are developed to accurately predict the ground truth PIP values using three inputs, i.e. physical parameters. The three physical parameters are the camera's current zoom, the distance between the target and the camera (Euclidean distance) and the size of the target (i.e.region of interest).

To explore the sufficiency of these physical parameters in predicting the ground truth PIP values, a set of three state-of-the-art regression methods, i.e. Multi-linear, Support Vector Machine, and Symbolic regression are developed and analysed within this chapter. By applying a Multi-linear regression, we explore the existence of a linear correlation between physical parameters and ground truth PIP. Applying Support Vector Regression (with RBF kernel), we explore the existence of non-linearity in the relationship, and finally, Symbolic regression leads to induce the structure of the model as well as the regression coefficients from data itself without a *priori* assumptions. It is important to note, while the primary aim of the feature abstraction models is to generate accurate predictions, the secondary interest for

them is to be easy to interpret. In a sense that by looking to the mathematical form of the produced model understand how each input (i.e. predictor) relates to the outcomes (i.e. response values) or how changes in each predictor affect the model's outcomes [54]. The importance of interpretability in our solutions can be thought of this way: If solutions are robust and self-contained, the interpretability of them might not be critical; however, if it is required to use them in the context of larger systems, it could be crucial for debuggability [63]. Therefore, the accuracy of the predictions and the interpretability of the outcomes are two evaluation metrics across this chapter.

The term, *feature abstraction* in this study represents the process of abstracting properties of the three inputs as three features into only one feature, PIP.

Throughout this chapter, we focus on answering the following research questions:

1. Does any correlation exist between physical parameters as a set of independent variables (inputs) and the ground truth PIP as a dependent variable (output)?
2. How accurate can physical parameters approximate the empirically verified PIP value?

With this in mind, the chapter proceeds as follows.

Section 4.1 describes feature abstraction process in the light of the main architecture and also explains the terminologies used within this chapter. Section 4.2, provides some quantitative insights into the data set. Section 4.3 discusses potential data pre-processing methods to apply across the data set before training the models. Section 4.4 describes our validation method and the process of splitting the data to the training and testing sets. In section 4.5, the predictive models are built and analysed using three different regression methods; Multi-Linear regression, Support Vector Regression, and Symbolic Regression across the data set. Finally, section 4.6,

concludes the chapter with a discussion, highlighting the predictive ability of each regression function according to the accuracy and standard error metrics and raises an important question emerging from this.

Throughout the analysis, three main regression methods are considered for building feature abstraction models. Their effects on the quality/accuracy of the outcomes of the model are compared and investigated. The simplicity and interpretability of the obtained models are described. As such, this chapter provides both the results for the f functions of the architecture and an introduction to the next chapter's contributions.

4.1 An Introduction to Feature Abstraction Models, f

In addition to the ground truth PIP establishment, the next step is to predict empirically verified PIP values using three physical parameters inspired by modelling a camera's FoV, previously described. For a camera with a given image sensor resolution, i.e. pixel density, changing each of these physical parameters affect the number of pixels used to represent the region of interest, i.e. patch. Therefore, to profile the pixel deviation of PIP incorporating these parameters, a set of regression analyses conducted across our lab-based dataset, to produce the approximation functions. Indeed, the exact pixel density of a specific patch varies depending on camera type, environmental factors such as lighting. Here, for a given image sensor resolution, we investigate the impact of only a small number of physical properties on the pixel deviation.

To achieve this, first, we explore the existence of simple linear relation, which is highly interpretable between three physical parameters as inputs and the ground

truth PIP as output. Linear regression-type models are appropriate when the relationship between the input and output falls along a straight line.

The second method is symbolic regression, which offers data-driven regression models by discovering the *structure* as well as *coefficients* within that structure. It produces closed-form solutions without making *a priori* assumptions about the structure of the model. Therefore, the relationship among inputs can be further interpreted through the estimated coefficients.

Finally, we developed a support vector regression, as state-of-the-art robust prediction tool using both linear and non-linear (RBF) kernels, to explore the existence of any non-linearity in the relationship as well as linearity between predictors and response.

The quality/accuracy of the induced models using these regression methods evaluated by running *K-fold Cross Validation* technique with K equals to five.

Before, exploring relationships within data for predicting some desired outcomes, the terminologies used in this chapter are briefly described as following.

- ***data point***, refers to a single independent unit of data.
- ***independent variables, predictors***, are the data used as input for the prediction function.
- ***dependent variable, response***, refers to the quantity being predicted.
- ***training data set***, refer to the data used to develop models.
- ***test data set***, a set of data points that have not been used prior, and only used for evaluation the performance of the final model.
- ***outliers***, refer to data points that are exceptionally far from usual stream of the data.

4.2 Data Description

To have some quantitative insights across the data set obtained from a set of profiling lab-based experiments with details described in Section 3.5, a list of descriptive statistics summarised in Table 4.1. Standard deviation (SD) metric indicates the spread of each variable across the data set. The more spread out the data distribution is, the higher is SD. The data skewness indicates how symmetric the distribution is around its mean point.

With these in mind, the results of the Table indicates; first, the data points of the zoom predictor tend to be closer to its mean value while with distance predictor, the data points tend to spread out further from its mean value.

Table 4.1: A summary of characteristics of the data categorised as predictors (inputs) and the response (output).

VARIABLES	PREDICTOR	RESPONSE	MEAN	SD	MIN	MAX	SKEWNESS
Zoom	✓	–	0.046	0.024	0.015	0.085	0.28
Object size	✓	–	0.31	0.20	0.10	0.66	0.61
Distance	✓	–	5.07	2.74	1.00	10.00	0.35
PIP	–	✓	0.054	0.12	0.0001	0.77	3.71

Second, the input variables are on different scales which requires us to re-scale them on a common scale for further analysis. This, can make some improvement towards the numerical stability of some calculation according to [54].

This transformation, however, comes at a loss of interpretability of the individuals since the data are no longer in the original units.

Third, a useful observation derived from data is to explore the minimum and maximum values of each variable. In general, if the ratio of the maximum value to the minimum value is higher than 20, the data tend to have *skewness* [54]. The sample skewness statistic is formalised as follows.

$$skewness = \frac{\sum(x_i - \bar{x})^3}{(n-1) \times v^{\frac{2}{3}}},$$
$$v = \frac{\sum(x_i - \bar{x})^2}{(n-1)}$$

Where x is a variable, n is the number of values, and $\bar{x}$ is the sample mean of the variable. If the distribution of a variable is approximately symmetric, the skewness values will be close to zero. The symmetric distribution refers to the probability of falling on either side of the distribution's mean is approximately equal.

Looking at the skewness values of the variables indicates that while the distribution of three predictor values is approximately symmetric, the response data which the row is shown in grey colour in the table, exhibit a strong skewness of 3.71. The details of tackling the data skewness issue are discussed in the next section.

In addition to the description of data demonstrated in Table 4.1, a visualisation of the distribution of each variable across data set shown in Figure 4.1.

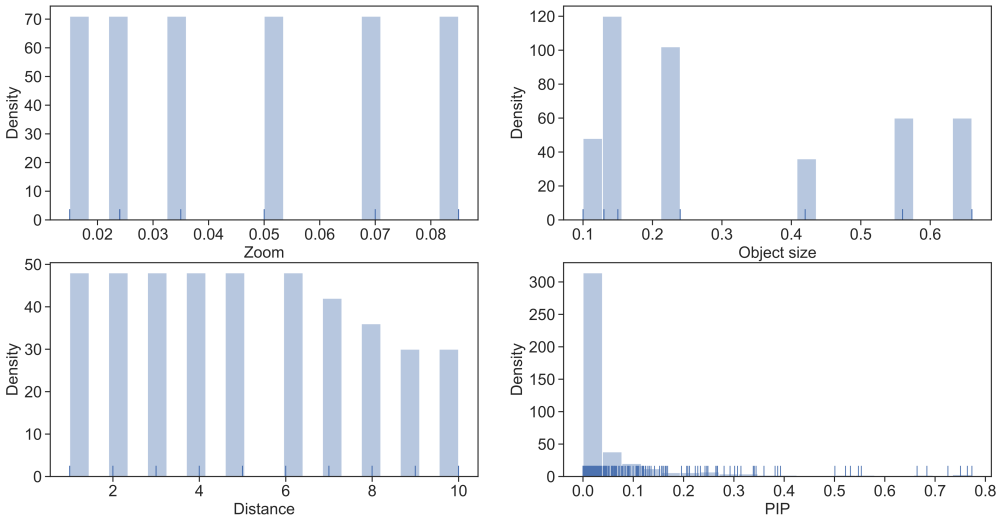


Figure 4.1: Histograms of the standard deviation of the three predictors and the one response. From top to bottom and left to right, zoom, object size, and distance and PIP. The response value has a strong right skewness with a concentration of data points with low values (around 0 - 0.2). For this variable, the ratio of the largest value to the smallest value is 7700 (way larger than a usual amount of 20) and a skewness value of 3.71. Small vertical ticks at each histogram bin, show values of each observation fall in each bin.

The distribution of three predictors and one response variable in Figure 4.1 confirms a strong right-skewness of the PIP values, with a large number of data points accumulated on the left side of the distribution graph. The next section looks at data pre-processing techniques that can be applied across the dataset to mitigate the impacts of such problems.

4.3 Data Pre-processing

The requirement for a data pre-processing often depends on the type of model being used. For example, it is described later in this chapter that the linear regression is sensitive to the characteristics of predictor and response variables. While tree-based regressions are insensitive to these characteristics. As it can be observed from Figure 4.1, the predictor's values are on different scales. Thus, the first step towards

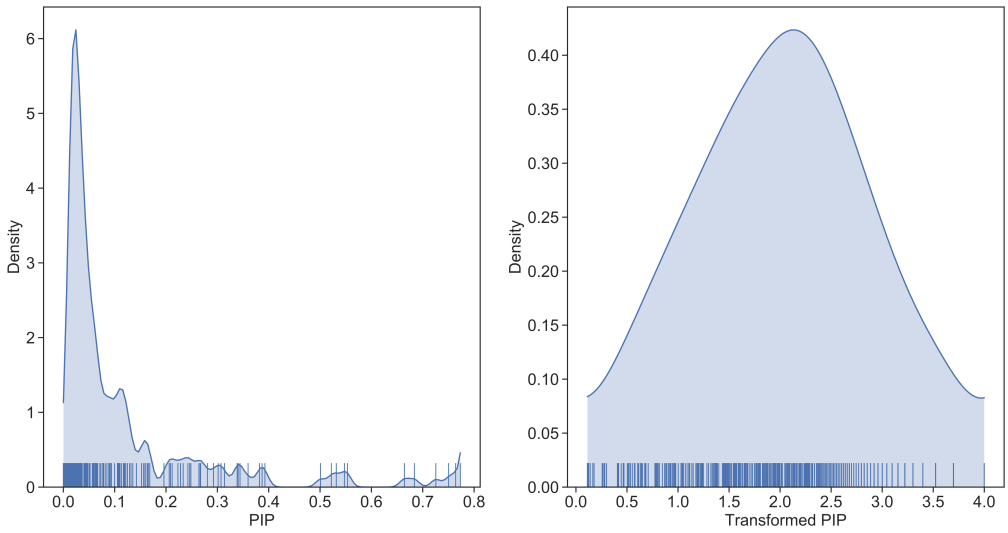


Figure 4.2: *Left:* A Kernel Density Estimation (KDE) graph of the density of observations of the PIP values with a strong right-skewness value of 3.71. *Right:* The density of observations of the same variable after a $(-log)$ transformation. The skewness value of the transformed PIP is equal to -0.009. The solid blue ticks demonstrate the value of each observation.

data pre-processing is to *normalise* the predictors to bring them on a common scale. Normalisation is considered as re-scaling real-valued numeric attributes into the range 0 and 1.

To remove the distributional data skewness (occurred across the response values), as was determined in the previous section, one possible alternative is to apply a *transformation* techniques. Here, the data values of PIP is transformed to the $-log$, that help to remove the skewness. Figure 4.2, demonstrates the distribution of the PIP before and after transformation. A standard non-parametric Kernel Density Estimation technique [127, 99] used to estimate the probability of distribution of PIP values before and after transformation. Although, after applying the transformation, the distribution of the response values is not entirely symmetric, a significant improvement achieved in the distribution of data when compared to the non-transformed values. The new skewness value is equal to -0.009. The negative value indicates the appearance of a very slight left-skewness, that is negligible.

In the subject of *between-predictors correlation*, which becomes a concern when a model has more than one predictors, it is essential to note that there is no *collinearity* observed among our three predictors. Collinearity problem [71] refers to the situation where there is a substantial correlation between a pair of predictor variables. This, can leads to instability when the statistical techniques are applied [10]. Due to the independent nature of the three selected physical properties (also known as predictors of the model) from each other, the collinearity would not be a concern in this study.

4.4 Data Splitting to the Training and Test Sets

The heart of predictive modelling can be attributed to the process called *data splitting*. Before moving on to the regression analysis, one important decision to make is *how to split data points into the train and test sets?* As described earlier in Section 4.1, a test set is used to evaluate the predictive performance of a model, which ideally (that data points) were not used in the process of building that model.

A simple common way to achieve this is to split the data to a static ratio of train and test sets, often by applying a random sampling method. This splitting method works best with a large amount of data, where a reasonable size of samples can be set aside to qualify the performance of the model, while still a large number of training samples are left to use for creating a model.

However, in this study, the number of data samples is not significant (total number of 426). Thus, given the method, the small size of the test set may not have a *adequate precision* to make a reasonable judgement on the performance of the model. This problem has also been discussed in several research works [65, 43, 67].

Another drawback to this method appears in the presence of noisy data and uncertainty problem. A model may be estimating the correct value of the test set but may pay a high cost on *uncertainty* [54]. This means, changing the test set (i.e. resampling) may produce a very different value. Resampling methods such as cross-validation can help to detect the noise or uncertainty across the data points.

Therefore, instead of a conventional train/test splitting method, in this study, a well-established resampling method called *K-Fold Cross-Validation* with $k = 5$ is applied to estimate the performance of the model in the presence of new data points or the model *generalise* to new data points. Another advantage raises from applying resampling methods, is by avoiding a single test set, every single data point is used for building a model, which is vital when the size of data set is relatively small.

In this approach, the data points are uniformly partitioned into k folds with approximately equal size. In the first iteration, the first fold is set aside as a test set (i.e. hold out) and a model is trained across $k - 1$ folds. The test samples then will be predicted by the trained model, and the estimation performance is measured. In the next iteration, the first fold is returned to the training set, and the whole process repeats for the second fold and so on. Each fold comes up with an accuracy score, indicating the proportion of the information in the data that is captured by the model's predictors.

Finally, the k resampled performance estimations are summarised for understanding the general predictive performance of the model.

4.5 Regression Analysis

To determine the relationship between three physical parameters as predictor variables, and PIP as the response variable, a set of three different regression analysis performed with details described as following. It is important to note that we are not claiming to employ the best regression technique across our data set. Instead, we apply three well-established techniques to explore existence of different possible relations, e.g. linear, non-linear, induced from data (with no priori assumption about the mathematical form of the relation).

Thereby, coming back to our research questions, this section mainly focuses on answering the following research questions for each set of obtained predictions:

1. Does any correlation exist between physical parameters as a set of independent variables and ground truth PIP as a dependent variable?

2. if, the answer is positive, how well can physical parameters approximate the empirically verified PIP value?

4.5.1 Multiple Linear Regression

Multiple linear regression is an extension of ordinary linear regression when the number of predictors is more than one [54]; thereby the goal is to estimate the regression coefficient in a way to minimise the sum of squared errors (SSE) between the observed and predicted response.

$$SSE = \sum_{i=1}^{n} (y_i - \bar{y_i})^2 \tag{4.1}$$

Therefore, the regression model is sensitive to the appearance of significant outliers within the data, which can lead to skewing away the linear regression model

from the true primary relationship. Following the terminology proposed in [25], the "*linear*" term refers to the fact that the model is linear in its coefficients, β_j. This assumption addresses the functional form of the model. A form of this type of models is presented as below.

$$Y_i = \beta_0 + \beta_1 X_{i1} + \beta_2 X_{i2} + ... + \beta_P X_{iP} + e_0 \tag{4.2}$$

Where, Y_i represents the numeric response for the ith data point, β_0, is the regression intercept, β_j is the regression coefficient for the jth predictor (in the case of multiple linear regression), X_{ij}, represents the value of ith predictor for the jth data point, and finally e_0 refers to the random error that can not be described by the model.

It is important to note, before exploring the predictive performance of a linear model, a set of preliminary analysis were already performed in Section 4.3, to ensure there is no violation of the assumptions of the normality and the colinearity [70, 54] across the dataset.

The benefits of transforming the ground truth PIP before training a linear model was demonstrated in Figure 4.3 concerning the accuracy, R^2-score of the regression. There are several formulas for calculating the R^2-score of a regression [55] also known as the coefficient of determination [72]; the simplest one refers to the proportion of the total variance of the response variable that is captured by the model [45].

Figure 4.4, illustrate the distribution of residuals of the multiple linear regression model. Where, *residuals*, refers to the difference between actual observation and predicted values by the model, which is a useful metric to measure the predictive performance of the model [133]. To achieve the residual graph while applying cross-validation, the built-in *cross-val-predict* method from the scikit-learn library [79] is used. Utilising this method, each data point on the graph belongs to one test set,

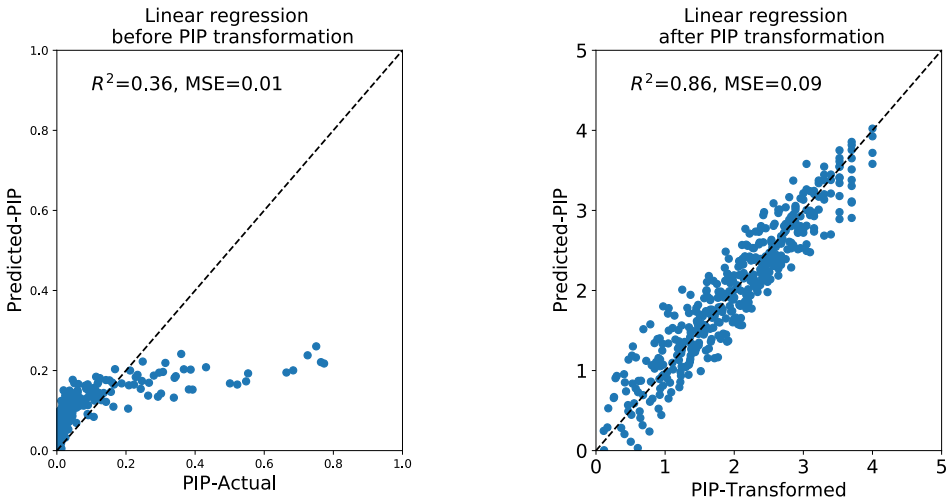


Figure 4.3: A visualisation of an impact of transforming the response variable on the linear regression results. The y-axis of both graphs demonstrates the outcomes of multiple linear regression (predicted-PIP). *left*: The x-axis, refers to the PIP before-transformation. *right*: shows the benefit of transforming the response values before training a linear model on the accuracy of predictions. The x-axis shows the values of PIP after transformation.

and its prediction is calculated using a model fitted on the corresponding training set.

As it can be observed, the residuals scatter uniformly across the regression line, assuming a linear model. The variance of the errors $(y_i - \widehat{y_i})$, seems to be consistent across all observations. In order to quantify the quality of the trained (i.e. fitted) multiple linear regression model, a set of metrics are evaluated. The mean accuracy of five folds is $0.85(+/-0.08)$. The highest accuracy score explained by the model is 86%, with a mean-squared-error (MSE) value of 0.09, formally described at Equation 4.3, and the root-mean-squared error of 0.30.

$$MSE = \frac{1}{n}\sum_{i=1}^{n}(y_i - \bar{y_i})^2 \tag{4.3}$$

Where accuracy estimation representing the average standard error (plus/minus standard deviation). The estimated coefficients and intercept of the linear model ob-

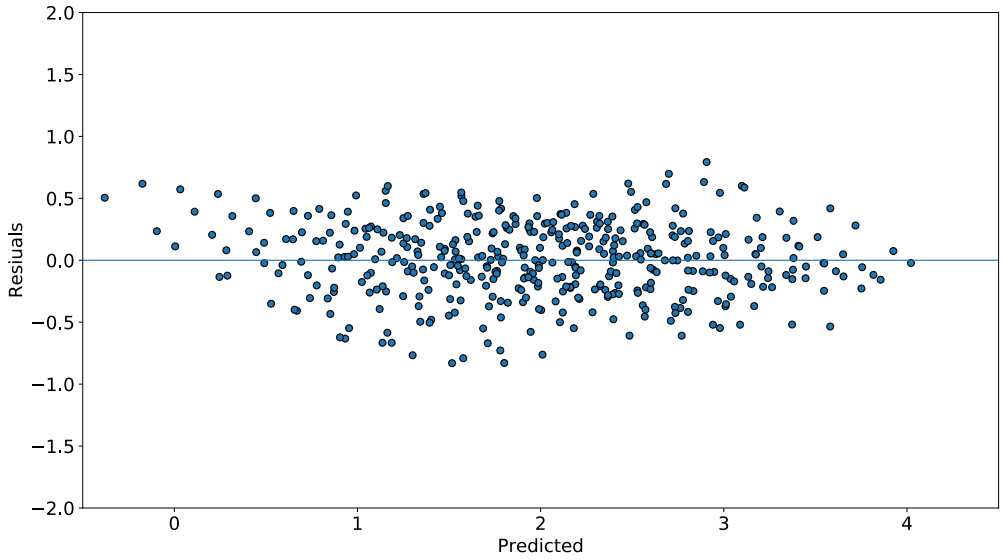


Figure 4.4: A visual demonstration of the distribution of residuals, assuming a linear relationship. The black dots on the graph represent data points. The solid blue line represents the regression line. The x-axis refers to the predicted values of the linear model, and the y-axis indicates the residuals with respect to the regression line.

tained, using the fold with highest accuracy score applying a 5−fold cross-validation). The regression function is represented in the following regression equation.

$$f_{lin}(q, d, z) = -18.82\ z - 2.0\ q + 0.2\ d + 2.47 \tag{4.4}$$

The mathematical nature of the obtained model indicates the outcome is highly interpretable. For example, one unit increase in a predictor leads to how much change in the response values. As mentioned earlier, these models are appropriate when the relationship between predictors and the response fall along a straight line. Based on the accuracy and standard error metrics, it is shown that a linear correlation exists between three inputs and ground truth PIP as output at 95% confidence bound.

4.5.2 Symbolic Regression

Unlike the standard multiple linear regression model, where a model's structure (e.g., Equation 4.2) is hypothesised and fit across all data points, *symbolic regression* offers data-driven regression models by discovering the *structure* as well as *coefficients* within that structure [104, 126]. One way of achieving this, is to use Genetic Programming (GP) [53].

In this way, a population of naive random formulas, i.e. programs are set up to represent a relationship between independent and dependent variables to predict new data. The successive generation of programs optimised and evaluated in terms of how well they fit the observed data points. In the process of evolution, this information then used to decide which program to use as parents for the next generation [126, 56, 4, 52].

Symbolic regression using GP can become a very useful method when models need to be simple and interpretable with reasonable generalising behaviour, while they are accurate with the outcomes. The regression itself, defined as a set of statistical processes for underlying the mathematical expression that best describes relationships between the predictors and the response pairs [84]. However, this, then raise a trade-off between models complexity degree and the accuracy of the fit, which is considered as one of the challenges ahead of this method [104].

Furthermore, symbolic regression makes no or limited *a priori* information about the process and no assumptions on the models. This means symbolic regression allows the data pattern itself reveals structures of variables, functions and constants that are appropriate to describe the observed behaviour. This benefit of the method makes it suitable for this study.

With this in mind, here, a standard symbolic regression method using GP is de-

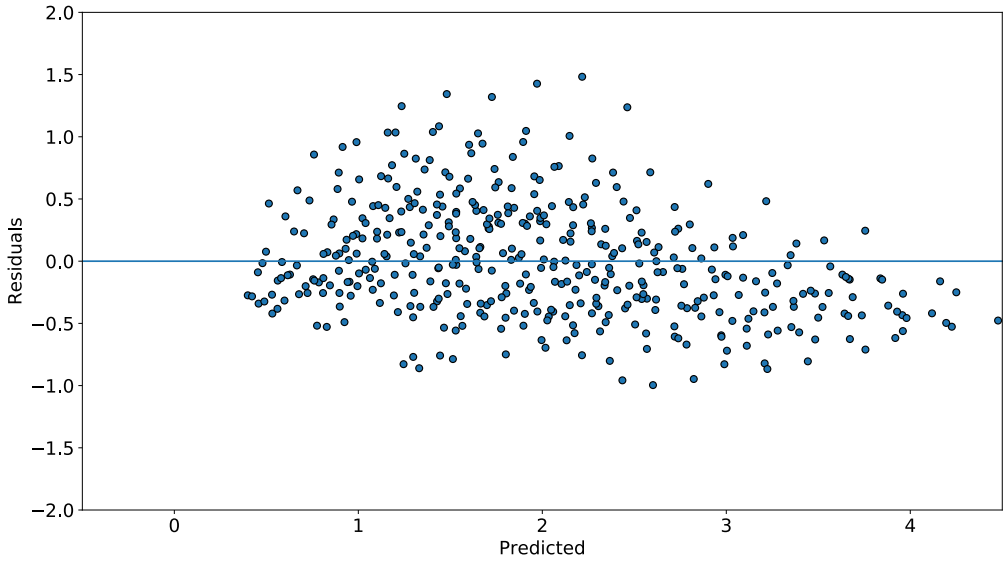


Figure 4.5: A visual demonstration of the distribution of residuals, developing a symbolic regression using GP. The black dots on the graph represent data points. The blue solid line, represents the regression line. The x-axis refers to the predicted response values obtained from the symbolic model and the y-axis refers to the residuals with respect to the regression line.

veloped across all available data points. The gplearn library[1] is used to implement Genetic Programming (GP) based models. The parameter setting of the implemented symbolic regression is as follows; the size of the population is 1000, and the number of generations is 30, and finally, the parsimony coefficient value adjusted to 0.001. The parsimony coefficient is a constant value that regularises the size of a program. By adjusting the over-sized program's fitness to be less favourable for selection, this metric can control a phenomenon called *Bloat* [101]. Bloat is when the size of the program is increasing through evolution without a significant increase in fitness. While this phenomenon results in increasing computational overhead, it can also make the final results to be challenging to interpret. Larger values of the parsimony coefficient penalise the program more and can control this phenomenon. A residual's distribution graph demonstrated in Figure 4.5.

[1]gplearn library extends the scikit-learn machine learning library to perform GP with Symbolic Regression.

The process of obtaining the residuals of the Symbolic Regression model is the same as described in the previous section.

The residuals appear to be uniformly scattered around zero (i.e. the regression line) with respect to the predicted values. The mean accuracy of all five folds is $0.70(+/-0.12)$, with a mean-squared-error (MSE) value of 0.21 and the highest accuracy score of 73% captured by the model.

Comparing the measurements with the (multiple) linear regression results indicates that the predictive ability of the model induced by symbolic regression is slightly less than linear regression across the same data points with regards to the standard error and R squared values. The estimated coefficients and intercept of the symbolic model are represented in the following regression equation.

$$f_{SR}(q, d, z) = (z - 0.51) \times (-(2 \times z - 0.51) \times (-d - 0.51) - 0.36 \times q^{-1}) \quad (4.5)$$

As it is represented in Equation 4.5, the advantage of these kinds of models is that relationships among predictors can be further interpreted through the estimated coefficients.

The mathematical form of the model is not difficult to interpret. However, based on accuracy and standard error metrics, a comparison with the linear regression outcomes suggests that the predictive ability of the obtained model seems to be slightly lower than the linear model results.

4.5.3 Support Vector Regression

In addition to the results of Multiple linear, and Symbolic regression methods, to explore the existence of non-linear relationships between the independent and depen-

dent variables, this section studies the Support Vector regression. Support Vector Machines (SVM) [119] are state-of-the-art, powerful and highly flexible modelling techniques. SVMs build a *maximum margin separator*, as a decision boundary with the largest possible distance to the sample points. This helps the method to generalise well. The theory behind SVMs was originally developed in the context of classification models with a focus on tasks such as object recognition [94, 95]. In recent years, SVMs have been successfully applied to regression problems because of their robustness and simplicity [26, 68, 21].

Unlike Neural Networks, where the architecture has to be determined a priori or modified while training by some heuristic, which can become a more challenging problem for multilayer networks [23, 69], in Support Vector Regressions (SVRs), the architecture of the system does not have to be determined before training. Moreover, with the ability to embed the data into a higher-dimensional space through using a *kernel function*, the strength of SVRs are then in building a separating hyperplane at only a linear cost.

Following the support vector regression framework studied in [105, 26], we consider the ϵ-*insensitive technique* of SVM for our regression problem where focuses on minimising the effect of outliers on the regression equations [119].

In this way, no penalty is associated (within the training loss function) to the predicted points within the distance epsilon from the actual value. Outliers, generally defined as data points that are exceptionally far from the mainstream of the data [54].

In other words, the effect of outliers in the regression equations is minimised by fitting the error $(y_i - \bar{y_i})$, within a certain threshold, ϵ.

To describe this formally, let $(x_i, y_i) \in \mathbb{R}^m \times \mathbb{R}$, $i = 1, 2, ..., N$ be a set of training

data points with predictors (i.e. inputs), $x_i \in \mathbb{R}^{\mathrm{m}}$ and the response (i.e. output) $y_i \in \mathbb{R}$. As mentioned earlier, in the ϵ-insensitive SVR the goal is to find a function $f(x)$ (as the predicted result for data point x), that has at most ϵ deviation from the observed response values y_i across all the training data. By constructing a linear function f, taking the form of

$$f(x) = \omega.x + b \quad with \quad \omega \in \mathbb{R}^{\mathrm{m}}, \mathrm{b} \in \mathbb{R} \tag{4.6}$$

In order to obtain the coefficient and the intercept of this function as well as to ensure maximum possible flatness depicted in Figure 4.6, we can translate it to the form of convex optimisation problem [12], as represented in the Equation 4.7.

$$\begin{aligned} & min\ \frac{1}{2}\|\omega\|^2 \\ s.t. \quad & \begin{cases} \omega.x_i + b - y_i \leq \epsilon \\ y_i - \omega.x_i + b \leq \epsilon \end{cases} \end{aligned} \tag{4.7}$$

Where ϵ is a threshold specifying the width of the ϵ-insensitive tube. Having Equation 4.7, assumes that such function f actually exists that approximates all pairs (x_i, y_i) with ϵ precision, or in other words, that the convex optimisation problem is feasible [106].

However, not all of the residuals may fall in the ϵ boundaries. Thus, no such function $f(x)$ exists to satisfy these constraints for all data points. In this case, the *soft margin* loss function [8] can be used by introducing $\xi_i, {\xi_i}^*$ slack variables to deal with data points fall outside the ϵ boundaries.

Figure 4.6 depicts a schematic of a linear SVR with a soft margin. The loss

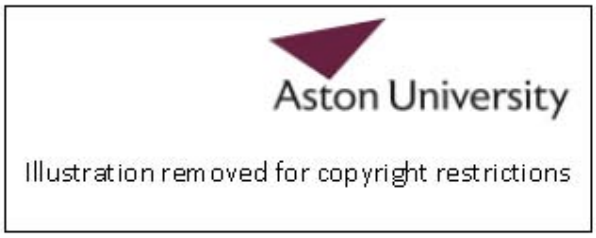


Figure 4.6: A schematic of the soft margin loss setting for a linear SVR [96]

function can be transformed as stated in [120] as following.

$$min\ \frac{1}{2}\|\omega\|^2 + C\sum_{i=1}^{N}(\xi_i + {\xi_i}^*)$$
$$s.t.\ \begin{cases} \omega.x_i + b - y_i \leq \epsilon + {\xi_i}^* \\ y_i - \omega.x_i + b \leq \epsilon + \xi_i \\ {\xi_i}^*, \xi_i \geq 0 \end{cases} \tag{4.8}$$

Where $\xi_i, {\xi_i}^*$ are slack variables and C is a penalty coefficient that determines the trade off between the flatness of f and the amount up to which deviations larger than ϵ are tolerated.

In order to explore existence of non-linear correlation between dependent and independent variables, a kernel-based transformation is performed. By mapping the data points into a high dimensional feature space, $\mathcal{F}$, $\Phi : x_i \to \mathcal{F}$, a kernel function performs a dot product $\kappa(x, x') = (\Phi(x).\Phi(x'))$, and then a linear regression is performed. For this problem space, two commonly used kernels are studied.

We explored the existence of linear relationship utilising a linear kernel of SVR and for exploration of the existence of any non-linear relationship a *Radial Basis Function* (RBF) from LIBSVM library [18], defined as follows.

- ***Linear Kernel***

$$\kappa(x, x') = x.x' \tag{4.9}$$

- ***Radial Basis Function Kernel (RBF)***

$$\kappa(x, x') = exp(-\frac{\|x - x'\|^2}{2\sigma^2}), \quad \sigma \in \mathbb{R} \tag{4.10}$$

Where x, x' are two data points from training dataset. The outcomes of applying these kernels in predicting the response value illustrated in Figure 4.7. Within our experiments, applying a range of different values for epsilon and C parameters, the configuration of $\epsilon = 0.1$, and the penalty coefficient value $C = 1$ provides highest accuracy of predictions. The σ value also set to $\frac{1}{n_{features}}$, $n_{feature}$ is the number of features. It is important to note, the value of ϵ can affect the number of support vectors used to construct the regression function. The bigger ϵ, the fewer support vectors are selected. On the other hand, bigger ϵ-values results in more flat estimates.

The statistical analysis of the goodness of fit of each kernel function summarised in Table 4.2.

Table 4.2: A summary of the goodness of fit analysis of a linear and an RBF kernel functions of SVR.

Kernel	MSE	R^2	Mean R^2-scores of all folds
Linear	0.19	0.74	$0.70(+/-0.11)$
RBF	0.05	0.92	$0.92(+/-0.04)$

In the case of using SVR with a linear kernel, the model's coefficient with the interception of the regression line is obtained as follows.

$$f_{SVR(lin)}(q, d, z) = -6.74z - 2.04q + 0.20d + 1.88 \tag{4.11}$$

LIBSVM only supports output probabilities for SVR models [18]. Hence, using

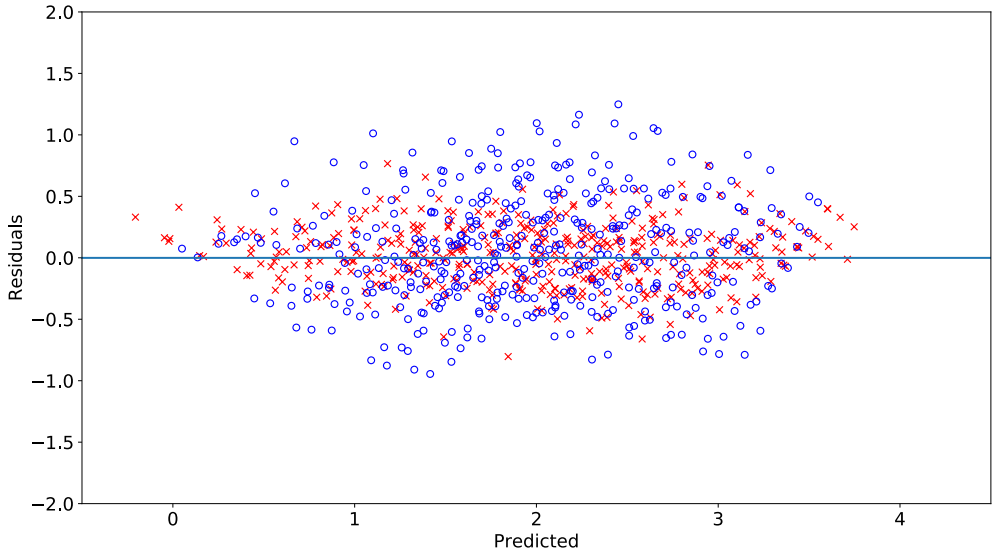


Figure 4.7: A visual demonstration of the distribution of residuals, developing an SV regression using i) a linear kernel with results demonstrated as blue circles and ii) an RBF kernel with results shown as a red cross. The solid blue line represents the regression line. The x-axis refers to the predicted response values obtained from each kernel, and the y-axis indicates to the residuals concerning the regression line. The graph clearly demonstrates the advantages of employing a non-linear kernel over linear with respect to residual amounts.

non-linear kernel (RBF), there is no probabilistic explanation for the regression. According to the distribution of residuals along with goodness of fit analysis, it becomes evident that the predictive ability of SVR with the RBF kernel function is considerably higher than linear kernel results. This confirms that a non-linear correlation is more likely to exist between the physical parameters and PIP.

While the predictive ability of the SVR with RBF kernel is the highest among all studied methods, there are some limitations to take into account. The first limitation lies in the choice of kernel according to [15], where the best choice of the kernel for a problem at hand is still a research question.

The second limitation could be the speed and size both in training and testing, which can be problematic for large datasets (millions of support vectors).

Finally, applying these methods do not directly provide a closed-form solution of the produced model; as such, they generate black box models [76]. In a sense, they

are not able to explain the process of obtaining the predictions in an understandable (interpretable) form.

The last limitation can further challenge the *easy implementation* concern of our high fidelity target detection model across an abstract simulation environment. This challenge will be discussed later in the Chapter 5 with more details.

4.6 Conclusions and Discussion

This chapter explicitly studied the feature abstraction modelling, f of the proposed architecture previously described in Chapter 3. These predictive models aimed to explore the sufficiency of the three physical parameters in predicting the ground truth PIP values. A set of three regression methods was selected for this purpose, i.e. Multilinear, SVR, and Symbolic regression. Before, applying these methods, a collection of data pre-processing techniques was used to ensure there was no violation of the assumptions of the selected methods across the data set.

A *K-Fold Cross-Validation* with $k = 5$ is applied to estimate the performance of each obtained/trained model in the presence of new data points. Moreover, a set of performance metrics such as MSE (Mean Squared Error), r^2(squared correlation coefficient) were introduced to evaluate the accuracy of the predictions produced by each regression method.

According to the performance metrics and the distribution of the residuals graphs, a set of predictions that provided by SVR with a non-linear (RBF) kernel are the closest to the ground truth PIP, with an accuracy of predictions equal to 92%. This could confirm that it is more likely that there is a non-linear correlation between three physical parameters and ground truth PIP. However, while SVR provides

predictions with the highest accuracy demonstrating the high chance of existing a non-linear relationship, applying this method with RBF kernel does not provide a closed-form solution. This means that from interpretability perspective, it is not clear how changes in each predictor affect the outcomes of the model. The second accurate set of results were obtained through multiple linear regression method with the predictions capturing up to 86% of the variation of the ground truth. Applying this method, while the predictions are reasonably accurate, by providing closed-form solutions, relationships among predictors can be further interpreted through the estimated coefficients.

Finally, It was demonstrated that the predictive ability of the outcomes of Symbolic regression with capturing up to 73% of the ground truth was the lowest accuracy compared to the results of the other two methods.

Recalling the research questions, a non-linear correlation is more likely to exist between the three physical parameters as independent variables and the ground truth PIP as dependent variable according to the 92% of the accuracy of predictions. However, due to black-box nature of this method, a closed-form solution is not at hand for further analysis of the relationship between the independent variables (i.e.predictors) and dependent variable (i.e.response).

In order to facilitate the implementation of these models further across abstract simulation environments such as CamSim, to support runtime applications, according to the results, it is recommended to select the produced model of Multiple linear regression. In this way, while the predictions are reasonably accurate (86% of the ground truth), a closed-form solution is provided with the results easy to interpret. With the closed-form solution at hand, further, the implementation of the model across abstract simulation environments could be facilitated as a one-line task. More

discussion on this is described in the next chapter of this thesis.

Following the main architecture of a high fidelity abstract model, as depicted in Figure 3.1, the next chapter looks at the detector models, g of the diagram.

A particular focus will be on exploring the sufficiency of PIP — both ground truth and predicted — in building high fidelity models of three selected computer vision detectors.

Chapter 5

Computer Vision Detector Models

Following the architecture of a high fidelity target detection model described in Chapter 3 (depicted in Figure 3.1), the main focus of this chapter is on developing the second part, computer vision detector models g. This part itself breaks down into two chapters. The current chapter focuses on developing three models of detectors and exploring the sufficiency of *ground truth* PIP in reflecting the detectors' outcomes. This investigation is useful when the high-quality real or synthetic images of the scene are available. For the case of abstract simulators, where the high-quality images of the scene are not available, we explore sufficiency of *predicted* PIP in approximating the ground truth confidences in the next chapter.

Within this chapter, first, we build three models of detectors and apply them on our image dataset to create the ground truth confidences for further analysis. Next, we investigate the sufficiency of the ground truth PIP in predicting the ground truth confidences by developing a least square linear regression. Throughout this chapter, we focus on answering the following research question:

1. Is the ground truth PIP as an intermediate point of representation of our architecture sufficient to build a high fidelity model of each of ground truth confidences?

A wide variety of feature detectors are available in literature for the task of object classification such as Histograms of Oriented Gradients (HOG) [24], wavelet-based features [125]. To build models of detectors, we selected three widely used local feature extractors with a simple architecture and farely high performance across wide range of vision tasks in literature, i.e. SIFT (Scale Invariant Image Transform), SURF (Speeded Up Robust Feature), and ORB (Oriented FAST and Robust BRIEF). Given extracted features, the similarity of the two features is investigated using a brute-force search with an efficient evaluation of the Euclidean distance. The confidence of each detector in correctly detecting a target of interest across a given image is evaluated by its matching precision. Because each detector runs on pure real image features, the outcomes create ground truth confidences for this study.

Previously, in their work [31], Esterle et al. showed there is a *linear* correlation between pixel density and the classification success rate. In this regard, we develop a linear regression between the ground truth PIP (as an independent variable) and the outcomes of each detector, i.e. ground truth confidences (as a dependent variable).

It is important to note that this chapter is not investigating the best computer vision classifier for the task of object detection. Instead, by building three models of detectors using three well-established feature extraction methods, we establish ground truth confidences, to explore the sufficiency of PIP in predicting detector's outcomes.

Through simulation results, it is shown that the accuracy of ground truth PIP in predicting the detectors' outcomes varies depending on the selected detector. Based on an accuracy evaluation metric, there is a strong linear correlation exists between the ORB detector's success rate and ground truth PIP when compared to two other detectors.

The chapter proceeds as follows. Section 5.1 highlights the fidelity concern of the simulation environments models, in the light of the trade-off previously identified in Chapter 2. Section 5.2, describes the feature extraction process across the target and template images. The confidence of detecting the object correctly within a given image defined in Section 5.3, in the line of potential false matches problem and how to tackle this issue. Section 5.4, describes the performance of ORB object detector model. The sufficiency of the ground truth PIP in predicting the outcomes of ORB is also explored in this section. Section 5.5, describes the performance of the SIFT detector, along with exploring the existence of a linear correlation between ground truth PIP and SIFT outcomes. Section 5.6, analyses the performance of the SURF detector model across the image dataset, and draws a linear regression between ground truth PIP and SURF outcomes. Finally, Section 5.7, concludes with a discussion.

5.1 Motivation

An important trade-off introduced and described in Chapter 2 between accurate but computational expensive and imprecise but computationally cheap simulations.

Due to the simplified nature of abstract models, which remove details and hence can introduce errors, the outcomes can be imprecise and have room for improvement in terms of their fidelity. An example of this will be observed later in our case study when comparing our results against CamSim standard detection model's results. In general, it is not known what impact making such simplified abstract models has on the ability of the simulator to capture real-world behaviour. This still remains as a general open concern in agent-based modelling [36].

In this chapter, by building accurate models of the three detectors, the fidelity of PIP (ground truth) is evaluated in reflecting real-world outcomes (i.e. ground truth confidences). This evaluation has been carried out through a set of profiling experiments and developing a linear regression.

5.2 Computer Vision Detectors: Feature Extraction

To represent visual information from raw pixels in images, the first step is called feature extraction. As described in Chapter 2, feature extraction is a process of representing visual information of images into compact vectors known as features or descriptors. Methods based on local features identify relevant patches of the image and represent the local visual information of these patches into descriptor vectors [34].

In the feature extraction process as depicted in Figure 5.1, a large number of *handcrafted features* are extracted from each template image. This may involve a set of computer vision algorithms such as edge detection, corner detection, or threshold segmentation as well [75].

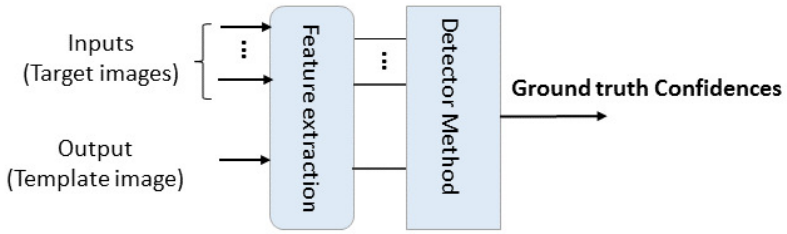


Figure 5.1: A schematic of the process of ground truth confidence formation. Each of the three detectors can replace the Detector Method box.

Next, a set of highly distinctive features from the template image are searched for other images, i.e. target images in a brute-force manner with an efficient evaluation of the Euclidean distance as a visual similarity function. Finally, if a significant

number of features from a target image matches with the template image as formally defined in the next section, the target image is classified as containing that specific object (e.g. the ball image, Figure 3.7 in our profiling experiments). The location of the object is then selected based on the density of re-identified features.

5.3 Computer Vision Detectors: Confidence of Detection

To perform reliable matching between template and target images, as described in the previous section, we require to extract distinctive invariant features from each image. This helps to ease the impact of undesired rotation noises might have occurred while conducting the lab-based profiling experiments.

Given a camera equipped with an adjustable zoom lens, as the employed zoom on the camera gets wider, the more of background get involved in the image. This leads to the size of the object of interest get smaller within the entire image. Therefore, the number of correct matches (between the template and target images) drops significantly and some *false matches* rise from the background in addition to the correct ones. A good match for local image features, to an extensive database of features from target images, is considered the one with minimum Euclidean distance between features descriptors [33]. To tackle the false match problem and discard the features that do not have a good match to the database of target images, a well-known distance ratio test is performed as proposed by Lowe [62] across all matches. In this method, the probability of having a correct match is determined by taking the ratio of the (Euclidean) distance from the nearest neighbour to the (Euclidean) distance of the second closest. In our image matching techniques, the matches with a distance ratio greater than 0.8 are rejected in this study. The Threshold set up

in this study is based on a range of experiments conducted using different values of $0.6, 0.7, 0.8, 0.9$. While considering smaller values leads to increased number of false positive matches, increasing it to 0.9 results in dramatic drop on the number of true positive matches. Setting the threshold to 0.8 results in a rational match accuracy across the studied techniques.

In order to quantify the performance of each detector in correctly detecting an object, a precision metric also known as confidence is computed using the following Equation 5.1:

$$\zeta(j) = \frac{feature_{det}(j)}{feature_{total}(j)} \tag{5.1}$$

Where, $feature_{det}(j)$ refers to the features from the template image that positively matched with the detected features in a target image (positive match). $feature_{total}(j)$ refers to the total number of features extracted from a template image. Thus, continuous values of $\zeta(j)$, indicate the confidence of a particular detector model in correctly detecting an object within a given image.

5.4 Computer Vision Detectors: ORB

Oriented FAST and Robust BRIEF (ORB) uses FAST keypoint detector [86], by efficiently computed orientations based on the intensity centroid moment. FAST keypoints are computationally fast and suitable for real-time visual feature matching systems. However, keypoints are variant to image scale and rotation. In addition to FAST keypoints, ORB also uses a recent feature descriptor, BRIEF [16]. It describes the visual information of images by using binary features and run a simple binary

test between pixels to train a set of classification trees. Similar to FAST, BRIEF is sensitive to in-plane rotation. To tackle these issues, some modifications have done toward both methods to finally make ORB features invariant to image scale and rotation, the details of the work can be found in [88].

Running ORB as a local feature descriptor along with a brute-force search across the template and target images, a typical feature matching results using OpenCV library [13] demonstrated in Figure 5.2. The target image ($focal\ length = 85mm$) with a scale of 2904×2850 pixels on the right and the template image of the same scene ($focal\ length = 24mm$) and with a scale of 5184×3456 pixels on the left side of this figure.

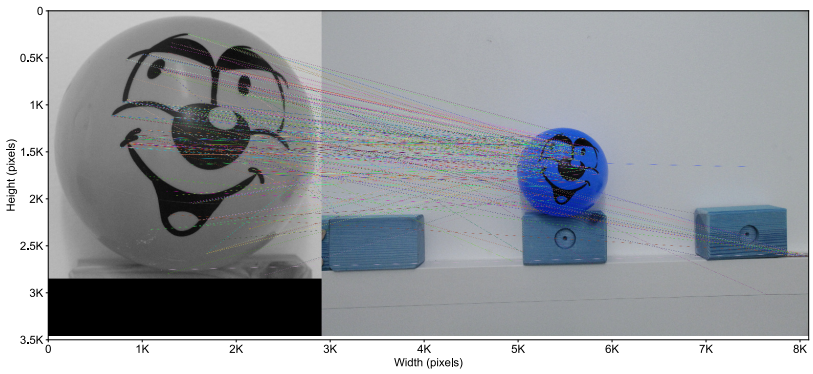


Figure 5.2: Typical feature matching result using ORB features on real camera images. The template image on the left, with a scale of 2904×2850 pixels and a target image on the right, with a scale of 5184×3456 pixels using the same viewpoint. Coloured dash lines indicate all matches, which includes both valid and invalid matches.

Although, as it can be observed from figure 5.2, the majority of matches seems true matches. However, there are still false matches with a growing number as i) the distance increase ii)the zoom gets wider (the focal length decreases), iii) the size of the object under the experiment get smaller.

Given the ground truth confidences of ORB detector technique, the next step is to evaluate the sufficiency of the ground truth PIP in predicting ORB results. To

demonstrate this, a *Ordinary Least Square*linear regression [100] developed between the PIP values as the independent variable and the ORB results as the dependent variable. The least-square method involves finding a mathematical expression for the relation between two variables (e.g. PIP and ORB), such that the sum of the squared deviations from the mathematical relationship is minimised. Thus, by choosing the regression line that is the *closest,* line to all data points, e.g.(x_i, y_i) the sum of the squared deviations of ORB to the regression line are minimised [107]. The outcome of this analysis demonstrated in Figure 5.3.

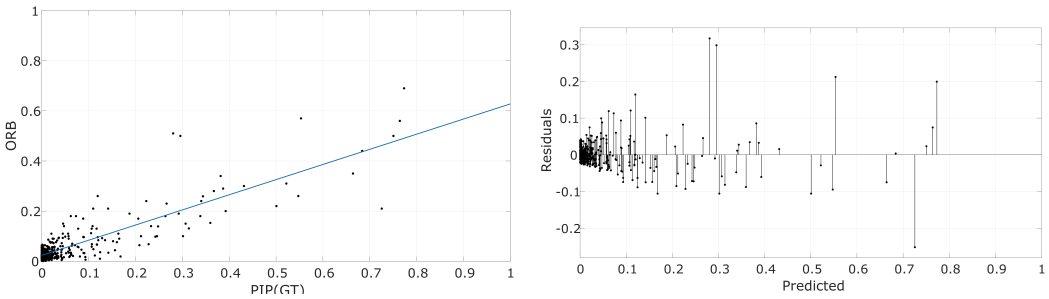


Figure 5.3: Figure (a), shows a correlation between the x-axis, PIP (ground truth) and the y-axis, ORB results. The solid blue line is the regression line between two variables which suggest a simple degree one polynomial relation can be derived from this correlation: figure (b), a visual demonstration of the distribution of residuals relative to the regression line. The x-axis refers to the predicted values. y-axis indicates the residuals.

A mathematical expression of the correlation between PIP and ORB is obtained as follow.

$$g_{ORB}(PIP_{GT}) = 0.60 \times PIP_{GT} + 0.02 \tag{5.2}$$

Looking at the results space, the solid blue line on the Figure 5.3(a), suggests a simple degree one polynomial relation between ORB and PIP. The residuals relative to the fit appear randomly scattered around the zero line, which suggests the model describes the data well.

As described earlier, in the least square method to determine the best possible regression coefficient, the sum of the squares errors (SSE) must be minimised. The

regression SSE value is **SSE=0.76** and the R-square evaluation metric is equal to **0.77** according to a t-test at the 95% confidence level.

Coming back to the research questions of this chapter, it is shown that there is a degree one polynomial relationship exists between the ground truth PIP and ORB results with accuracy (i.e. R-squared value) of the regression up to 77%.

5.5 Computer Vision Detectors: SIFT

Scale Invariant Feature Transform (SIFT) proposed by Lowe [62] has proven considerably successful in a range of applications using visual features. It has solved a set of issues such as image rotation, scaling, affine distortion, and view change in feature matching field of research [42]. One drawback to the robust and distinctive SIFT features is that they are computationally expensive which makes the technique become slow. Later in this chapter, an extension to this method also is studied which is considered speeded up version of SIFT.

Following the trend of previous section, first SIFT detector technique is applied on the image dataset. An example of typical results of SIFT features matching result demonstrated in Figure 5.4 for the same scene setting as explained in previous section.

In order to investigate existence of a linear correlation between ground truth PIP and SIFT, an ordinary least square regression is developed between these variables.

The results of the regression along with the residual graph of distribution demonstrated in Figure 5.5.

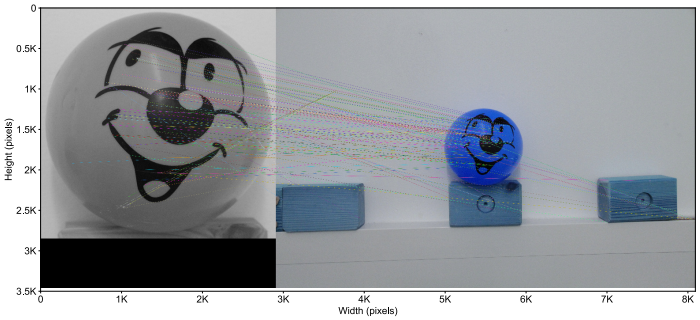


Figure 5.4: Typical feature matching result using SIFT features on real camera images. The template image on the left, with a scale of 2904×2850 pixels and a target image on the right, with a scale of 5184×3456 pixels using a same viewpoint. Coloured dash lines indicate all matches, which includes both valid and invalid matches.

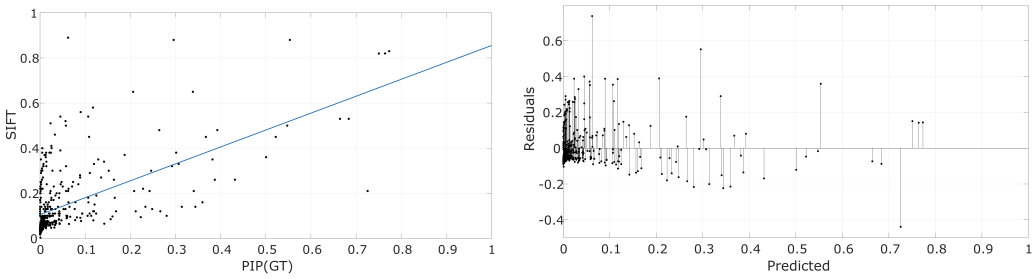


Figure 5.5: Figure (a), shows a correlation between the x-axis, PIP (ground truth) and the y-axis, SIFT results. The blue solid line is the regression line between two variables which suggest a simple degree one polynomial relation can be derived from this correlation. Figure(b),a visual demonstration of the distribution of residuals relative to the regression line. The x-axis refers to the predicted values. y-axis refers to the residuals.

A mathematical expression of the correlation between the ground truth PIP and SIFT results obtained by the regression is defined as follow.

$$g_{SIFT}(PIP_{GT}) = 0.75 \times PIP_(GT) + 0.10 \tag{5.3}$$

Analysing the outcomes of the regression through residuals indicates that despite of having residuals scattered randomly around the zero line the appearance of outliers in the residual graph is undesirable. It is observed that generally the residuals error is higher with the value of **SSE= 5.97** when compared to the results of the ORB detector model. Furthermore, **R-square =0.36** demonstrate that a fitted line

can at most capture 36% of the variation of SIFT outcomes. Coming back to our research questions, the results of goodness of fit along with the visual demonstration of residuals attribute the insufficiency of a linear correlation in descibing the variation of SIFT results.

5.6 Computer Vision Detectors: SURF

Speeded Up Robust Features (SURF) [5] technique that reduces the computational cost of SIFT by using integral images [22]. It is based on gradient orientations which try to keep the quality of detected keypoints. It uses SURF keypoints to detect features in an image [6]. In addition to the results of ORB and SIFT methods described earlier, here, we explore sufficiency of ground truth PIP in explaining the variation of SURF results. Therefore, by developing a linear regression between these variables, we investigate the existence of a linear correlation between PIP and SURF results.

An example of typical results of the SURF feature matching is demonstrated in Figure 5.6 for the same scene-setting described in two previous sections. As it can be observed, the number of features extracted and matched employing SURF technique, is considerably higher than ORB and SIFT outcomes while it is performing faster than SIFT algorithm. However, this leads to an increase, in the number of false matches as well.

The same as two previous approaches, an ordinary least square regression is developed to explore sufficiency of PIP in building a good model of SURF outcomes. A regression line that is the closest to all data points is shown in Figure 5.7.

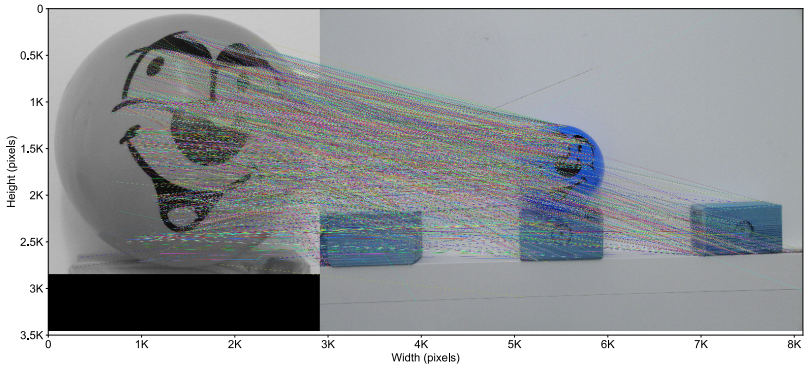


Figure 5.6: Typical feature matching result using SURF features on real camera images. The template image on the left, with a scale of 2904×2850 pixels and a target image on the right, with a scale of 5184×3456 pixels using the same viewpoint. Coloured dash lines indicate all matches, which includes both valid and invalid matches.

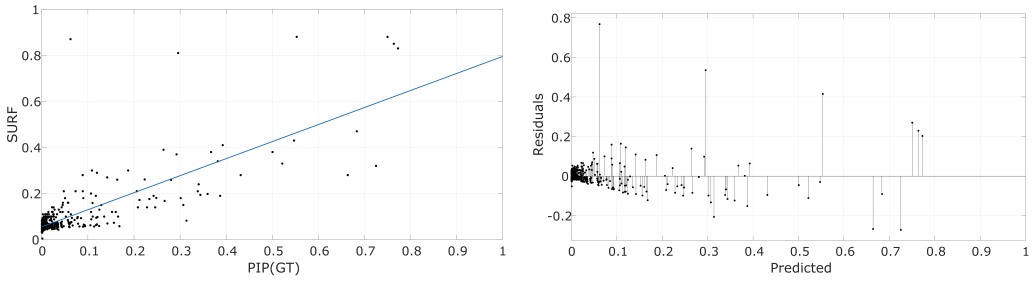


Figure 5.7: Figure (a), shows a correlation between the x-axis, PIP (ground truth) and the y-axis, SURF results. The solid blue line is the regression line between two variables which suggest a simple degree one polynomial relation can be derived from this correlation: figure (b), a visual demonstration of the distribution of residuals relative to the regression line. The x-axis refers to the predicted values. y-axis indicates the residuals.

The regression function obtained as follow.

$$g_{SURF}(PIP_{GT}) = 0.73 \times PIP_{GT} + 0.05 \tag{5.4}$$

Analysing residuals from a graphical display shown in Figure 5.7(b), indicates, while residual errors are generally lower than the SIFT results, still few outliers appeared in the graph.

In order to decide if the suggested model at Equation 5.4 is acceptable as a description of SURF results, the goodness of fit statistics of the regression is examined. The SSR value of the regression is **SSE = 2.03**, which is lower than SIFT results but still higher than ORB results. The success rate of the fitted line in explaining the variance of SURF results is equal to **R-square = 0.62**, which is reasonably higher than SIFT results. Therefore, addressing our research questions, we speculate by neglecting few outliers PIP can also build a good model of SURF confidence based on statistic analysis of the obtained regression line.

5.7 Conclusions and Discussion

This Chapter explicitly studied the second part of the proposed architecture, represented by g, while addressing our main research question on exploring sufficiency of ground truth PIP in reflecting three detectors' outcomes.

In this regard, three detector models were developed, a combination of three feature extraction methods with a brute-force search with an efficient evaluation of the Euclidean distance. The precision of each detector was interpreted as the confidence of that detector in correctly detecting an object. Three sets of ground truth confidences were created by applying each detector on our image dataset.

A simple linear regression was developed to investigate the sufficiency of PIP as an intermediate point of representation, in reflecting the ground truth confidences.

The best results according to the accuracy evaluation metric obtained between ground truth PIP and ORB features outcomes, with 76% of accuracy in capturing the variation of ORB-detector model. According to the accuracy metric, PIP can capture up to 62% of SURF-detector and 36% of SIFT-detector's outcomes.

Based on the accuracy evaluation metric, it was observed that the ground truth PIP could build a good model of ORB-detector outcomes developing a linear regression. The next accurate results produced between PIP and SURF-detector's outcomes. However, it was observed that ground truth PIP was less successful in capturing the variations of SIFT-detector's outcomes.

The investigation carried out in this chapter is useful when the high-quality images of the scene are at hand to establish the ground truth PIP. However, in the case of abstract simulation environments, where high-quality imagery is not available to the simulator, an alternative is required. With this in mind, we move on to the next chapter to investigate the sufficiency of predicted PIP, produced by feature abstraction models, f in capturing the variations of three detectors' outcomes.

Chapter 6

Putting it All Together: Combination of Two Partial Models, f and g

So far, we studied two partial models represented by f and g of the proposed architecture. However, to deploy the final *high fidelity abstract target detection model* across an abstract simulation environment, where the ground truth PIP values are not available, an alternative is required.

This motivates the work of this chapter, to investigate the sufficiency of predicted PIP produced by feature abstraction models, f, in predicting each detector's outcomes.

Furthermore, for the purpose of comparison, a description of CamSim's standard model of detection is provided, and it is applied to the image dataset. Next, the sufficiency of the outcomes explored in predicting a set of three ground truth confidences.

Therefore, in this chapter, we finalise the mathematical form of the high fidelity target detection model to be deployed across the abstract simulation environment, CamSim. Highlighting the improvement in the fidelity of our model's outcomes when compared to the CamSim standard model's results in capturing the ground

truth confidences.

This short chapter proceeds as follows. Section 6.1 characterises the fidelity of the predicted PIP (produced by f) in capturing the variation of ground truth confidences. A comparison is conducted in Section 6.2 between the outcomes of our model and the standard detection model of CamSim in reflecting the ground truth. Finally, we summarise the findings of this chapter in Section 6.3.

6.1 Combination of Two Partial Models

Addressing our main research question distilled in Chapter 3, so far, we explored sufficiency of ground truth PIP in explaining the variations of detector's results, performing linear regression. In this section, we are particularly interested in evaluating the fidelity of predicted PIP (obtained from feature abstraction models, Chapter 4) in predicting the ground truth confidences. Therefore, by combining two partial models, f, g to build accurate models of the three detectors, *predicted detector* models denoted as $g(f(q, d, z))$.

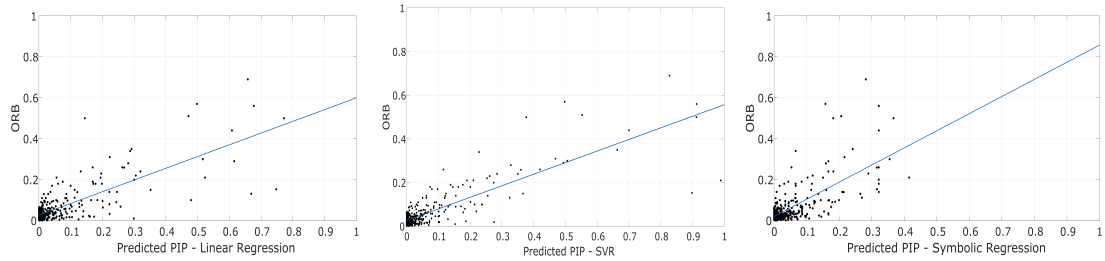
For this exploration applying linear regression, three sets of predictions (f) from three regression methods create three sets of independent variables, and the three ground truth confidences create three sets of dependent variables. Therefore, the total number of nine models are developed applying a simple linear regression between each of these independent and dependent variables.

Further, according to the r-squared, and standard error values of each obtained model, the high fidelity target detection model will be selected to deploy across the simulation environment for target detection estimation.

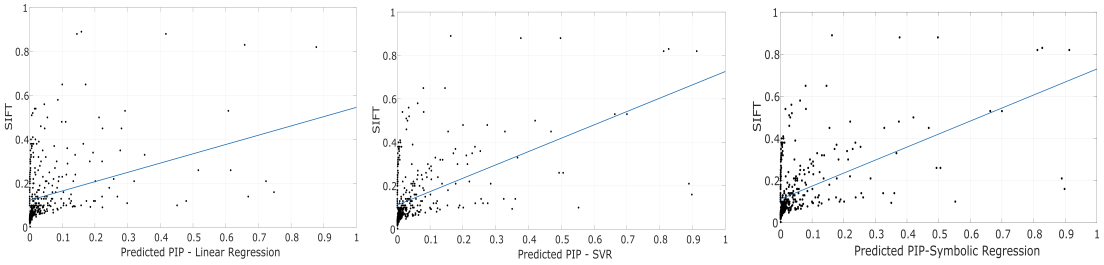
The results demonstrated in Figure 6.1.

The x-axis of each graph is one of the three independent variables, and the y-axis

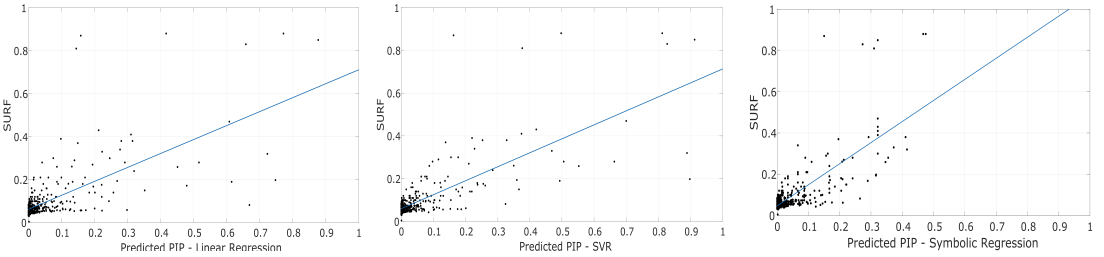
of each graph is one of the three sets of ground truth confidences.



(a) ORB



(b) SIFT



(c) SURF

Figure 6.1: A selection of total nine graphs demonstrating the existence of a linear correlation between the ground truth confidences, ORB, SIFT, and SURF, with the predictions obtained running three regression methods, Linear, SVR-rbf kernel, and SR. The black dots represent the data points, and the solid blue line is the linear regression line.

A summary of goodness of fit analysis of each graph along with the mathematical expression of the relative linear model is listed in the Table 6.1.

The same as ground truth PIP results described in the previous chapter, it is observed, there is a *higher chance* of existing a linear correlation between the ORB-detector outcomes and the predictions of all three regression methods.

Based on r^2 accuracy metric, all three sets of predicted PIP can build a good model of ORB features. However, as it can be observed the accuracy drops when using SURF and SIFT features.

Table 6.1: A summary of the goodness of fit evaluation of the total nine graphs along with the mathematical form of each produced model,$g(f(q,d,z))$ at 95% confidence bound. Each row of the table represents the predictive ability of a linear correlation obtained between each pair of detector-prediction outcomes using three evaluation metrics, r^2, RMSE, and SSE

Graph ID	r^2	**RMSE**	**SSE**	**Closed-form Solution**	$g(f(q,d,z))$
ORB-Linear	60%	0.052	1.19	✓	$0.571 \times f_{lin}(q,d,z) + 0.026$
ORB-SVR (RBF)	71%	0.044	0.85	✗	$0.538 \times f_{SVR(rbf)}(q,d,z) + 0.025$
ORB-SR	50%	0.059	1.51	✓	$0.835 \times f_{SR}(q,d,z) + 0.020$
SIFT-Linear	23%	0.130	7.22	✓	$0.608 \times f_{lin}(q,d,z) + 0.114$
SIFT-SVR (RBF)	27%	0.127	6.92	✗	$0.519 \times f_{SVR(rbf)}(q,d,z) + 0.116$
SIFT-SR	29%	0.125	6.67	✓	$0.618 \times f_{SR}(q,d,z) + 0.111$
SURF-Linear	46%	0.082	2.87	✓	$0.649 \times f_{lin}(q,d,z) + 0.061$
SURF-SVR (RBF)	57%	0.069	2.30	✗	$0.653 \times f_{SVR(rbf)}(q,d,z) + 0.059$
SURF-SR	55%	0.073	2.34	✓	$1.022 \times f_{SR}(q,d,z) + 0.046$

The results of the table demonstrate, while utilising a Support Vector regression with a non-linear kernel (rbf) can capture the most of the variation of almost all of the detectors outcomes, there is no closed-form solution available using this method.

As described in Chapter 4, due to the black-box nature of the method, the coefficient of the regression function $f_{SVR}(q,d,z)$ is not provided. Thereby, further implementing the high fidelity detection model utilising SVR (with rbf kernel) across CamSim (built-in Java) would not be straight forward processed.

To do so, we either require to export the trained SVR model from (in this case) Python programming language using some standard language such as Predictive Models Markup Language *PMML*, which enables an exchange of predictive models between different tools and environments [40]. However, using this tool, the issue of the compatibility over different models and Python versions is a concern that should be taken into account.

The second alternative is to train the model within CamSim, directly implementing the SVR regression. However, depending on the resources of the utilising machine, this approach might not suit the runtime applications. However, the implementation challenge of SVR, in the case of two other regression methods with a

closed-form solution at hand, $f_{lin}(q, d, z)$, and $f_{SR}(q, d, z)$, can be easily sorted as a one-line task.

Therefore, to facilitate the implementation of the high fidelity detection model across abstract simulation environments such as CamSim to support runtime applications, (e.g. the studied k-coverage case study), we select an alternative based on accuracy and prediction error metrics. In this context, according to r^2 accuracy score and SSE standard error of predictions, the *ORB-Linear* model as a second accurate model with a closed-form solution at hand is selected to implement across the CamSim Simulation environment.

In this way according to the results of Table 6.1, ORB-Linear row, the mathematical form of the high fidelity abstract target detection model, providing the $f_{lin}(q, d, z)$ from Equation 4.4 Chapter 4 is represented as follows.

$$\begin{aligned} g(f_{lin}(q, d, z)) &= 0.571 \times (-18.82\ z - 2.0\ q + 0.2\ d + 2.47) + 0.026 \\ &= -10.74 \times z - 1.14 \times q + 0.11 \times d + 1.43 \end{aligned} \tag{6.1}$$

Coming back to the main requirements in developing a high fidelity detection model, while accurate predictions are desirable, the obtained models required to be easy to interpret, i.e. in a sense by looking at the mathematical form of the model understand why the model works. This tension between the accuracy of the predictions and the model's interpretability described in [54] also as a trade-off when prediction accuracy is the primary goal.

With this in mind, the selected model, produces reasonably accurate predictions, with capturing around 60% of the ground truth ORB confidence variation, it is highly interpretable and easy to implement across a simulation environment.

Although adding an intermediate point of representation, PIP in the design of the model leads to an extra layer of prediction errors. This, generally decrease the maximum possible achievable accuracy of the outcomes. However, the question is how much improvement the proposed model could bring to the simulator in terms of fidelity when compared to its standard model of detection?

6.2 Fidelity Evaluation

So far, the mathematical form of our high fidelity abstract target detection model, to be implemented on CamSim simulation environment is formalised. Its fidelity evaluated in predicting the variations of ground truth confidences. It becomes evident that based on the r-squared metric, the selected model can capture up to 60% of the ground truth ORB confidences while incorporating only three physical parameters (i.e. properties of realism).

To have a better understanding of the obtained fidelity, here we explore fidelity of the CamSim standard model of detection in predicting ground truth confidences. To achieve this, CamSim standard detection model represented in Equation 6.2 is described and implemented across our lab-based dataset to produce a set of target detection predictions. The outcomes of the model are denoted as *CamSim Conf.* values.

CamSim standard off-the-shelf confidence of target detection model is an extreme abstract layout that only takes one dimension of realism, i.e. camera's current zoom, into account at the time of confidence of detection calculation. More details on the development process of the model can be found [31]. The simplified relationship between camera's current zoom and confidence of target detection obtained through

Table 6.2: A summary of the goodness of fit evaluation of the total three graphs along with the mathematical form of each obtained model at 95% confidence bound. Each row of the table represents the predictive ability of a linear correlation obtained between each pair of detector-CamSim Conf. outcomes using three evaluation metrics, r^2, RMSE, and SSE

Graph ID	r^2	RMSE	SSE	Closed-form Solution	Obtained Model
ORB-CamSim Conf.	9%	0.081	2.85	✓	$0.092 \times \kappa(c_i) + 0.028$
SIFT-CamSim Conf.	6%	0.144	8.79	✓	$0.16 \times \kappa(c_i) + 0.098$
SURF-CamSim Conf.	7%	0.108	5.01	✓	$0.122 \times \kappa(c_i) + 0.059$

a set of profiling experiments of their work and described as below,

$$\kappa(c_i) = 0.95 \times (1 - \frac{r(c_i)}{argmax(r(c_i))}) - 0.15 \tag{6.2}$$

Where, $\kappa(c_i)$ refers to the confidence of a virtual camera c_i in detecting a target of interest inside its FoV, and $r(c_i)$ is the radius of the camera's FoV based on its zoom level, and $argmax(r(c_i))$ defines the maximum radius based on the maximum zoom possible for the same camera. However, the radius of the FoV in their study attributed to the current zoom. This definition of the zoom is in *reverse* relation from the zoom, i.e. *focal length* of a real camera, that considered in our lab-based experiments (as described in Chapter 3). Thus, to have a fair comparison with a unique definition for the zoom parameter, the order of the CamSim model outcomes is reversed across the six discrete optical zooms of the profiling experiments described in Chapter 3.

Given the CamSim conf. predictions, next, we develop a linear regression between CamSim conf. as an independent variable and each of three ground truth confidences as a dependent variable.

A summary of the goodness of fit measures of the results demonstrated in Table 6.2. Generally, the results presented in table 6.2 indicate the performance of both models varies depending on the selected detector model. Comparing the goodness of fit evaluation metrics across two Tables 6.2, 6.1, both models are shown to

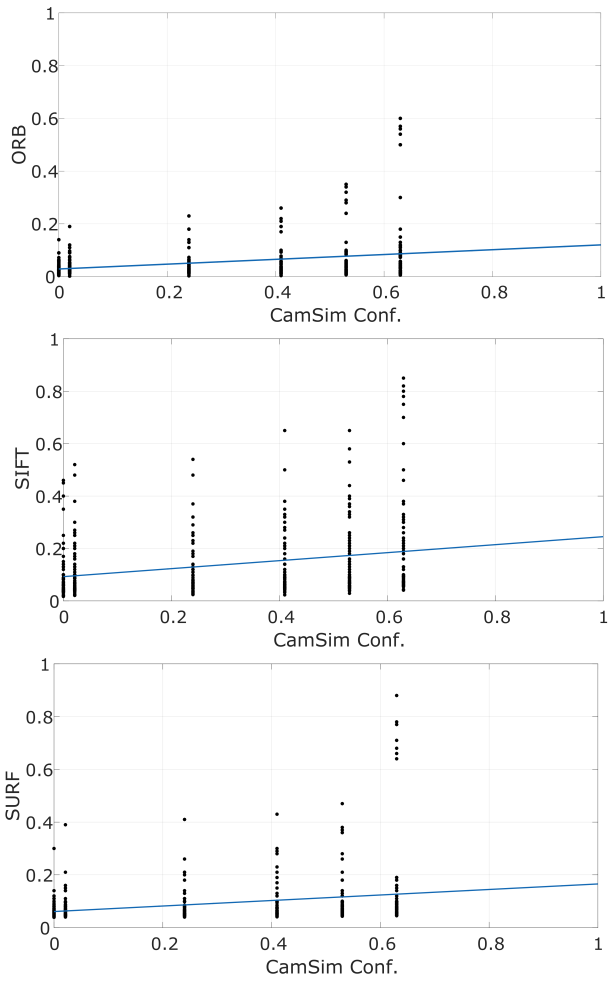


Figure 6.2: A comparison between the performance of CamSim standard detection model across three ground truth confidences of ORB, SIFT, and SURF outcomes. The x-axis of the graphs shows CamSim Conf. The y-axis of the graphs demonstrate results of ORB, SIFT, and SURF from top to bottom, respectively.

be more successful in capturing the variation of ORB, rather than two other detectors. It is important to note that, the fidelity of the obtained target detection model ($g(f_{lin}(q, d, z))$), significantly improved based on R-Squared evaluation metric when compared to the results of CamSim standard detection model. In this way, our detection model with predicting 60% of the variations of the ground truth ORB, made around 50% improvement in comparison to the CamSim detection results which can predict only 9% of the ground truth results.

6.3 Conclusions and Discussion

We explored the sufficiency of the predicted PIP obtained from feature abstraction models, f Chapter 4, in reflecting the outcomes of three detector models.

The combination of three sets of predictions (f), and three sets of ground truth confidences (g) produced nine approximation models, developing a linear regression. Based on the accuracy and standard error evaluation metrics, the mathematical form of the high fidelity target detection model to be deployed across CamSim formalised.

The fidelity of our model's outcomes was compared against the results of CamSim standard model of detection, in reflecting three ground truth confidences. The results confirm the fidelity of the outcomes of our model improved substantially, with almost 50% when compared to CamSim outcomes.

Throughout the results, it becomes apparent that incorporating more properties of realism (e.g. size of the target, distance from the camera) leads to a substantial improvement in the fidelity of the model's outcomes in predicting real-world operation. The implication of this improvement will be explored across our selected case study from coverage redundancy application of smart camera networks in the next chapter of this thesis.

Part III

A Case Study

Chapter 7

A Case Study: Coverage Redundancy in a Network of Smart Cameras

In order to explore the implications of the high fidelity abstract target detection model on real world applications, this chapter establish a case study from coverage redundancy management in smart camera networks domain. More specifically, the self-organising ability of smart camera networks in maximising the coverage redundancy across all moving targets is studied across CamSim simulation environment.

Generally, the performance of a coverage approach is highly influenced by the level of details captured by a camera. This, can provide valuable information about the location of targets, their temporal correspondences, and movement pattern over time. Thereby, the high fidelity target detection model developed and analysed across previous chapters becomes a fundamental requirement in correctly detecting a target within a camera's FoV before performing coverage strategies.

When a network of cameras with adjustable zoom lenses is tasked with object coverage, an important question is how to determine the optimal zoom level for each camera. More specifically, is it possible for each camera to determine its own zoom, based on local information in order to achieve highest possible k-coverage

for all objects in the system at all times? While covering a smaller area allows for higher detection likelihood, overlapping fields of view introduce a redundancy which is vital to fault tolerance and acquisition of multiple perspectives of targets. [123]. Considering a Pan-Tilt-Zoom (PTZ) camera, the main idea is to adapt an appropriate zoom (i.e. focal length) to maximase the coverage redundancy across moving targets. Indeed, adapting an appropriate orientation (e.g. pan and tilt) in addition to the zoom can maximise the geometrical coverage of the entire area using the individual FoV. This could improve the accuracy of a target detection models in correctly detecting a target of interest within a camera's FoV.

We categorised the studied coverage behaviours into three main classes. First, we look at offline behaviour, i.e. greedy approach, the second class studies online behaviours divided into; baseline and learning based approaches, and the third class describes the impact of more coordinated strategies on coverage redundancy.

All studied approaches, first employ the *high fidelity abstract target detection model* as a posterior probability to reason about the number of targets being correctly detected within a camera's FoV.

A visualisation of some fundamental elements in the coverage redundancy management in the self-organising smart camera networks depicted in Figure 7.1.

The results were previously presented in [123], [122].

The chapter proceeds as follows. Section 7.1 defines the coverage redundancy problem itself, as used throughout this chapter, and formulate it as k-coverage problem to be studied. Section 7.2, describes properties of a smart camera node as self-organising agents, with the ability to process the sensory inputs on board and communicate with other devices across the network. To evaluate performance of the coverage approaches, a set of test scenarios designed with details described in

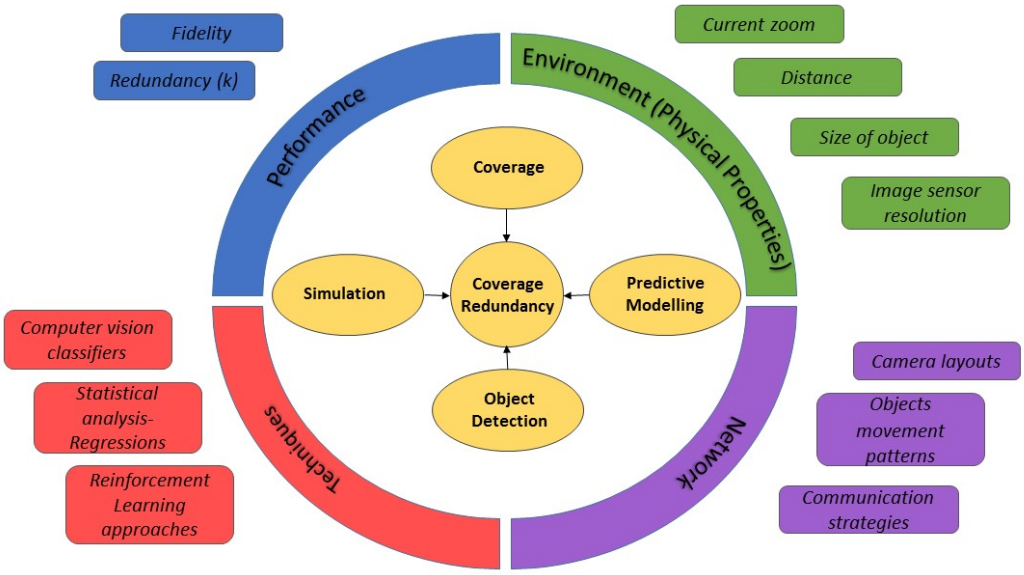


Figure 7.1: Foundational elements in our self-organising smart camera network. These elements drive the cooperation of smart cameras in finding an appropriate zoom configuration across the network in a way to maximise the possible redundancy across all mobile targets network-wide.

Section 7.3. Section 7.4, describes and analyses the coverage redundancy behaviours under three different categories, *offline-greedy*, baseline approaches (online-intuitive heuristics),*online-learning*. Section 7.5 describes the simulation results, including a comprehensive comparison of the performance of all discussed coverage behaviours across all designed test scenarios. Finally, section 7.6 concludes with a discussion.

7.1 Problem statement

In this section, we consider a smart camera network (SCN) of directional cameras $C = \{c_1, c_2, \ldots, c_i, \ldots, c_n\}$ each equipped with an adjustable zoom lens. The network is tasked to cover a set of moving objects $O = \{o_1, o_2, \ldots, o_j, \ldots, o_m\}$. The current zoom level has an inherent impact on the quality of the covered objects. A trade-off arises between the size of the camera's FoV and the quality of acquired

information. A narrower zoom covers less physical space but results in a higher pixel count for the region of interest. In contrast, a wider zoom covers more of the environment but uses fewer pixels on a given square-unit. Employing our high fidelity abstract target detection, if the detection probability of the target drops from a certain threshold, *the target will not be detected even though they are within the camera's FoV.*

Within the CamSim simulation environment, an object o_j is *covered* by a camera if that object satisfies two following conditions,

1. The object, lies within the camera's FoV. In 2D modelling of cameras FoV described in Equations 3.2, 3.1, Chapter 3. Where, the Euclidean distance between camera and the object $d_{ij} \geq r_i(t)$.

2. The probability of target detection, as formalised in the previous chapter, $g(f_{lin}(q, d, z))$ over o_j is above a certain threshold τ. Where τ, can be defined by a practitioner.

Here, we are not only interested in simply maximising the number of covered objects in the environment, but covering each object with as many cameras as possible. However, we are not trying to cover static points in the environment but rather mobile points that may change their position over time. Typically in $k-$coverage problems (e.g. [47, 44, 1]), a desired fixed value of k is used, and the challenge is to ensure that all objects are covered by at least k sensors with sufficient confidence. Here, this translates to $\forall o_j,\ k_j \geq k = \sum_{i=1}^{n} \kappa_{ij}$, where

$$\kappa_{ij} = \begin{cases} 1, & \text{if } g(f_{lin}(q, d, z)) \geq \tau \\ 0, & \text{otherwise} \end{cases}$$

Here, we focus on the problem of achieving the highest level of k-coverage across the network, requiring us to maximise the minimum value of k_j across all objects $o_j \in O$. We denote this minimum k value as k_{min} at time t, as defined in Equation 7.1.

Therefore, the goal is to maximise the k_{min} value across the network by exploring the impact of adopting different local behaviours at an individual camera level. Bearing in mind, the probability of detecting an object correctly within a camera's FoV is the posterior probability obtained from the high fidelity target detection model.

$$k_{min}(t) = min(k_1(t), k_2(t), ...k_j(t), ..., k_m(t)) \tag{7.1}$$

Further, given the online nature of the problem, as objects may move about, we are also interested in maximising k_{min} over time.

A discrete time window $t = t_1...t_{max}$ is considered and called *time step*, therefore this may be defined as:

$$performance = \sum_{t=1}^{t_{max}} k_{min}(t). \tag{7.2}$$

7.2 Cameras Properties

In contrast to traditional cameras, smart cameras are embedded devices able to perceive their environment, process this acquired knowledge on board, and communicate with other devices. By operating in networks, their ability to adapt to changing conditions makes them more robust, flexible, and resilient. Within the simulation environment, a smart camera node considered as a learning agent that can perceive its environment through an adjustable zoom lens (i.e.focal length).

Table 7.1: Summary of Scenarios Used in our Study

ID	Layout	No. of Cameras	Object Movement Pattern	Area Coverage
1	Lattice	9	Random	100%
2	Ring	8	Random	92%
3	Cluster	14	Random	66%
4	Lattice	9	Scripted	100%
5	Ring	8	Scripted	92%
6	Cluster	14	scripted	66%

Where the term *precept* refers to sensory inputs to a camera node, within the simulation environment, this can be translated to the number of detected objects at a given zoom. The *environment* that a camera node operates in, is a surveillance field with known boundaries and comprised of several moving targets with a constant velocity that can adopt different movement patterns. Since the environment changes while a camera node is acting, it is considered as a *dynamic environment*. The *action* refers to switching between available zooms.

Self-organising applications are able to dynamically change their functionality and structure without direct user involvement to meet changes in their environment. Within this case study, the adaptation capabilities are transferred to the individual cameras themselves.

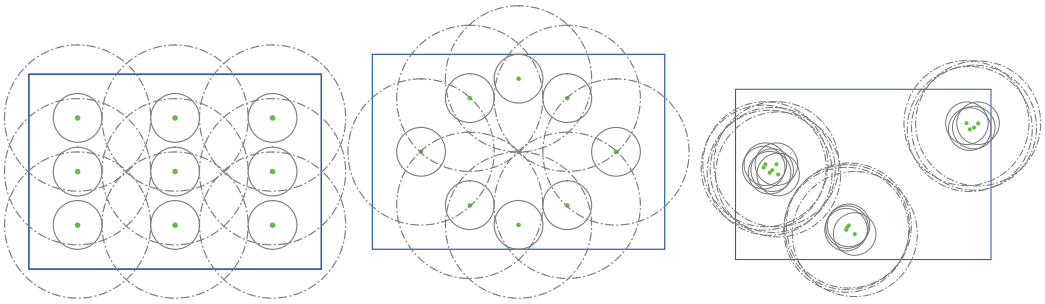
7.3 Test Scenarios

To evaluate the performance of our decentralised coverage behaviours in a dynamic environment, we construct six qualitatively different scenarios using CamSim simulator. A summary of the scenarios is provided in Table 7.1. The six scenarios are composed of three different camera layouts, each of which can represent a set of real-world applications. An overview of them is depicted in Figure 7.2.

Across our experiments, two different movement patterns defined for objects within each scenario. These include *random/semi-random* movement pattern, where

all objects move in straight vectors inside the surveillance field until they hit the boundaries then they bounce back in a random direction. *Pre-defined trajectory* (i.e.scripted) movement pattern, where all objects follow a rectangle trajectory placed around the centre of each camera layout. As a property of each scenario, the last column of the table indicates the ratio of the covered area (excluding multiple overlaps of camera's FoV) to the total size of the surveillance area. Decreasing area coverage increases the coverage holes across the surveillance field, which leads to increasing the risk of having *un-covered* objects for a given time interval.

In our experiments, the possible zoom length for a single camera is discretised into five levels ($\mathsf{z} = 5$). The distances is calculated as the 2D Euclidean distance between the camera and the location of the object. The small time slots used to calculate k_{min}, correspond to discrete time steps which are assumed to be synchronised across all cameras in the scenario.



(a) Lattice (b) Ring (c) Cluster

Figure 7.2: Camera layouts tested with the *CamSim* simulation tool [123]. A green dot represents a camera, and the associated grey inner circle demonstrate the minimum FoV when the camera is zoomed in, and the dashed circle represent the maximum FoV associated with zoom out.

7.4 Coverage Approaches

To maximise the value of k_{min} at any time, first, we employ the high fidelity target detection model to reason about the varying abilities of cameras to cover objects within their current FoV. Next, a set of coverage behaviours categorised into three

classes, i.e. offline, online divided into baseline and learning approaches, and more coordinated approach are studied. Their impact on the overall coverage redundancy (i.e.k_{min} values) is also performed across the network. Given the dynamic nature of the problem, in which objects always move about, the question is how to select an appropriate zoom for each camera such that the value of k_{min} is maximised over time?

7.4.1 Greedy Approach

Although cameras make decisions based on local information, we are primarily interested in performance at the global level. This forms a top boundary across the results space for further evaluations and comparison across different approaches. We call a specific set-up of cameras with corresponding zooms a *configuration*. The *Greedy approach* analyses all potential configurations in the current time step reachable from the current zoom level. Given each camera is equipped with $Z = \{z_1, z_2, ..., z_z\}$ discrete zoom levels, allows us to select the zoom configuration which provides the highest k_{min} at each time step.

However, determining an optimal k_{min} values using exhaustive offline search at every single time step is time consuming and computationally expensive with complexity increases exponentially with the number of cameras (Z^C). By the time this computation completes, a moving object has probably left the FoV of the camera. Moreover, doing so assumes that the characteristics of the scenario are known in advance; this includes cameras layout, objects movement pattern, cameras current zoom. Indeed, this lack of *a priori* scenario knowledge is a crucial problem characteristic motivating the online coverage approaches that use only local information

to provide near-optimal outcomes. Therefore, we next study some online intuitive heuristics do not require a priori knowledge of a scenario and do not involve in learning. These form our baseline approaches and then extend the idea, where individual cameras learn behaviours online during run time.

7.4.2 Baseline: Simple Intuitive Heuristics

Here, we extend the idea of maximising the k_{min} where each camera autonomously decides to select the zoom level independent from others at each time interval. In this regard, two simple online intuitive heuristics described as following.

- ***random approach***, is a simple distributed approach operating on local information alone. Each camera selects a random zoom at each time step. The zoom is sampled from a uniform distribution across all potential zooms.

- ***zoom out approach***, in this approach all cameras select the widest zoom for all time steps to provide the highest possible k_{min} across all available objects. This corresponds to the largest FoV fixed for all cameras throughout the simulation. The position of a target has no impact on the performance of this approach.

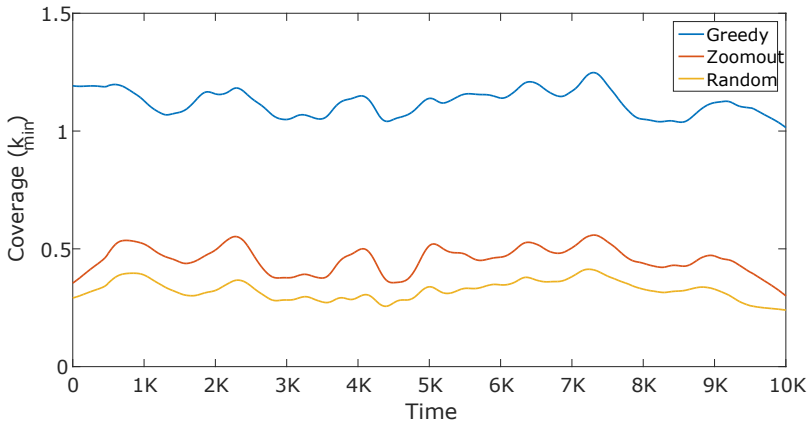


Figure 7.3: A graph comparing the performance of zoom out in red and random in yellow colours as baseline approaches with the greedy results in blue across the scenario one. The x-axis of the graph shows the simulation time steps, which is T=10000 and y-axis shows the coverage performance.

Due to stochasticity, all the experiments have been repeated 30 times, and the mean We smooth the results, using the locally weighted polynomial regression (LOWESS) filter [87], to aid visualisation of the achieved values for k_{min}, result is shown.

Employing the high fidelity target detection model before coverage computation, the global metric, k_{min} is used to evaluate the performance of three approaches network-wide. Assuming greedy behaviour guarantees that each object is at least covered by one camera all the time. However, the baseline approaches performed considerably weaker and failed to provide any redundancy across a given scenario. These results produce the lower and upper boundaries of the possible solution space.

7.4.3 Online Learning Approaches: Multi-armed-bandit solver

From a camera's perspective, the task is to select a discrete zoom from those available, which maximises its expected number of covered objects over time. Thus, this problem can be attributed as a variant of the multi-armed bandit problem [3].

The stochastic multi-armed-bandit problem has been used extensively in research to model the trade-off between exploration and exploitation faced by individual aims to gather new knowledge of its environment.

In the bandit framework, each discrete zoom can be considered as an arm of a bandit machine, and a resulting reward is received per pulled arm. Each camera can select a zoom (i.e. pull an arm) at each time step and achieve a local reward derived from the number of covered objects. In this way, a camera learns which discrete zoom performs well given the current state of the scenario and exploit its knowledge to provide a near-optimal performance.

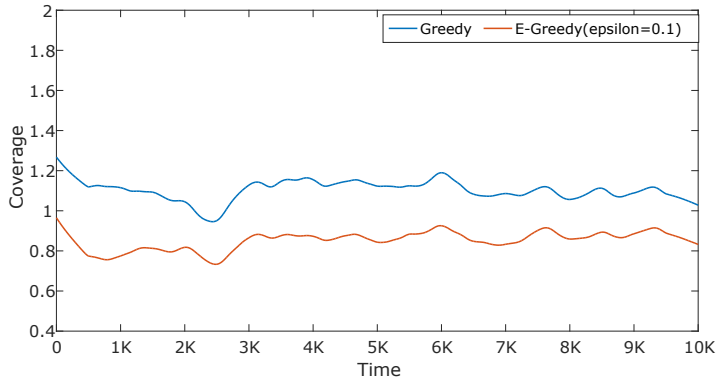
There are several so-called bandit solvers proposed in the literature. Here, we

studied ϵ-greedy [124], which requires a ϵ value to determine the amount of exploration. To apply a bandit solver to zoom selection behaviour at the local level in a self-organising system, a reward function is required to be defined, such that the network-wide goals are achieved. The reward function defined here is a linear combination of the local metric (the number of covered objects),

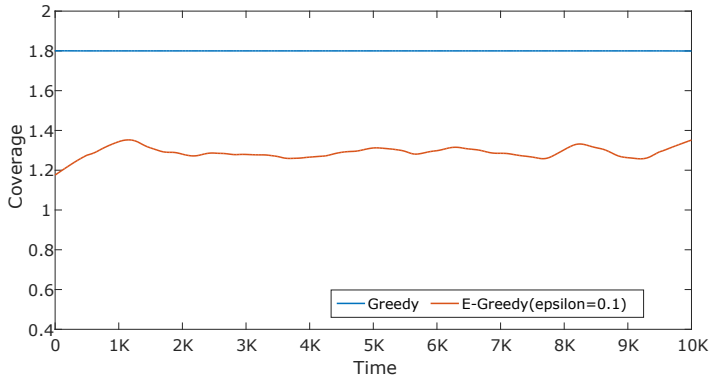
$$\begin{aligned} reward &= \lambda \times U_i(O_i) \\ U_i(O_i) &= \sum_{j \in O_i} u_i(j) \\ \phi_i : O_i \rightarrow \{0,1\} \quad s.t. &\begin{cases} 1, & \text{if } conf_{orb} \geq \tau \\ 0, & \text{otherwise} \end{cases} \end{aligned} \tag{7.3}$$

Where, O_i, refers to a set of objects laid within camera c_i FoV. The utility function U_i sums covered objects over all objects lied within the camera's FoV. λ parameter allows for tuning the reward value, and it is used for direct local learning such that outcomes at the global level are near to optimal.

For the ϵ-greedy approach, a set of various ϵ values from 0.1, 0.01, to 0.001 is explored. In all scenarios, $\epsilon = 0.1$ obtained the closest outcomes to the optimal results. Therefore, the value of $\epsilon = 0.1$ is used for the results demonstrated in Figure 7.4. Employing a scripted pattern increases the chance of learning for ϵ-greedy, which leads to a noticeable enhancement in the performance. It is observed that changing the objects movement pattern from a random to the scripted across the same camera layout, leads to a noticeable improvement in the total coverage redundancy values achieved network-wide. It becomes evident that performance of online learning algorithm across the deterministic environment (scenario 4) with all objects following the same trajectory all time is considerably higher than the stochastic environment,



(a) Scenario 1



(b) Scenario 4

Figure 7.4: Employing a scripted pattern increases the chance of learning for ϵ-greedy, which leads to a noticeable enhancement in the performance. Coverage redundancy across scenario one with all objects following a random pattern and scenario four, with all objects following a scripted pattern over time. The blue line shows the performance of the greedy approach, while red line shows the ϵ-greedy results.

where each object follow a random direction. The dynamic zoom selection (i.e. those which change over time, in this case through online learning) learnt from ϵ-greedy behaviour can obtain near-optimal results with capturing up to 81% of the greedy performance.

7.4.4 Online Learning Approaches: Reinforcement Learning

The task of *reinforcement learning* [112], [49] is to observe immediate rewards in order to learn an optimal or (near to optimal) policy for the environment. Whereas, the agent has no *prior knowledge* of neither the model of the environment nor the

reward function. Unlike, *supervised learning*, the reinforcement learning, (RL) agent never sees examples of good or bad behaviours. Instead, it receives some feedback, i.e. positive and negative rewards for the taken action [90]. Figure 7.5 outlines the basic reinforcement learning mechanism, depicting the interactions between the agent and the environment.

In this case study, a fully observable environment is assumed, where each precept of the agent provides the current state. Therefore, the framework of a standard reinforcement learning algorithm can be mathematically modelled as a Markov Decision Process (MDP) [89], [7]. A five-tuple defines an MDP: $\langle S, A, T, R, \gamma \rangle$, where S is a set of environment states, A is a set of agent actions, T is the state transition probability function, which defines the probability of moving between the different environment states, R is the reward function and γ, $(0 \leq \gamma \leq 1)$ is the discount factor, which models the relevance of immediate and future rewards. By assuming an MDP framework, the agent's policy can be represented as $\pi : A \leftarrow S$, a mapping from each state of the environment to a probability distribution of available actions.

The RL agent, learns an action-value function, or Q-values as demonstrated in Equation 7.6, giving the expected reward of taking a given action in a given state [90]. In this manner, the agent can compare the expected rewards for its available choices, i.e. actions without needing to know their outcomes. Thus, it does not require a model of the environment.

Nevertheless, this lack of the knowledge of the actual outcomes of the taken actions, where "*they do not know where their actions lead*" [90] (i.e. an agent can not look ahead), can restrict individual agent's learning ability.

In this context, when learning a model is not feasible the agent can still *learn to predict* its future behaviour using *temporal-difference* methods [110].

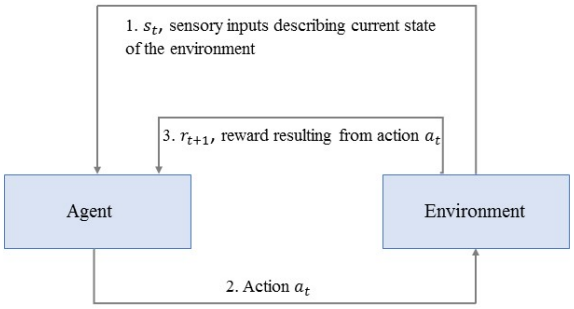


Figure 7.5: A schematic of Reinforcement Learning mechanism, demonstrating the agent-environment interaction

Unlike, the conventional prediction-learning methods that are derived by the error between predicted and actual outcomes, TD methods are derived by the error, i.e. difference between temporally successive predictions. In this manner, the learning appears to happen whenever a change occurs in prediction over time.

At each time interval, the agent selects an action and receives an immediate reward. The reward is used to update estimates of its action-value function, which then is used to predict the long-term discounted reward it will receive if it takes a given action in a given state.

The RL agent aims to learn a policy that maximises the total (i.e. long-term) expected discounted reward, i.e. optimal policy. The discounted expected reward, R_t , at time t is represented in Equation 7.4, where $\mathbb{E}$, denotes the expectation of the discounted reward and k denotes the number of actions.

$$R_t = \mathbb{E}[\sum_{k=0}^{\infty} \gamma^{k} R_{t+k+1}] \tag{7.4}$$

The RL algorithm can be decomposed into two components. The *update policy*, whose value is being learned or simply how the agent learns the optimal policy. *Behaviour policy*, which is used to control the agent during the learning. In an *off-policy* algorithm following [112] the update policy is different than behaviour

policy. Q-learning is an off-policy TD method [130]. In this manner, the agent takes advantage of employing exploratory behaviour to gather diverse data while it learns how to behave greedily with no exploration required.

In a *on-policy* algorithm the update and behaviour policies are identical. Therefore, it can not separate exploration from learning. Thereby, it should confront the exploration problem directly.

Here, the on-policy TD SARSA (State Action Reward State Action) algorithm [89], [111] is employed at individual camera level to learn best actions at each state of the environment to enhance network-wide coverage redundancy.

Our action space includes five discrete actions, indicating what zoom to select.

Using SARSA prediction method, a transition at time step t, $\langle s_t, a_t, r_{t+1}, s_{t+1}, a_{t+1}\rangle$ takes place from one state-action pair to the next. Where, the current state and action s_t and a_t, the immediate reward r, and the next state and action s_{t+1} and a_{t+1}. However, as mentioned earlier, using a_{t+1} introduces an extra variance to the Q-value update when the updated policy has got some stochasticity. This is a typical challenge ahead of on-policy methods such as SARSA [118]. This additional variance can slow the convergence [102], [118]. In order to evaluate the impact of different amount of exploration on the performance of SARSA approach, here we employed three widely used setups for $\epsilon = 0.1, 0.2, 0.3$. The results of this comparison on the coverage performance of on-policy SARSA is demonstrated in Figure 7.6.

Looking at the results space indicates that increasing the chance of exploration within the behaviour policy of on-policy SARSA from $\epsilon = 0.1$ to 0.3 leads to a noticable drop in the performance of the method across the network. This, also results in the method's convergence to the optimal performance (i.e.greedy) becomes

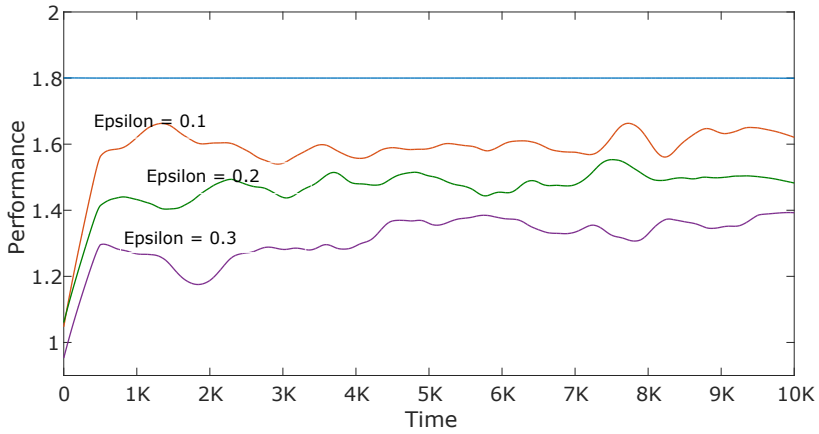


Figure 7.6: Impact of increasing chance of exploration through various epsilon values, on the performance of TD onpolicy SARSA within a deterministic environment.

slower.

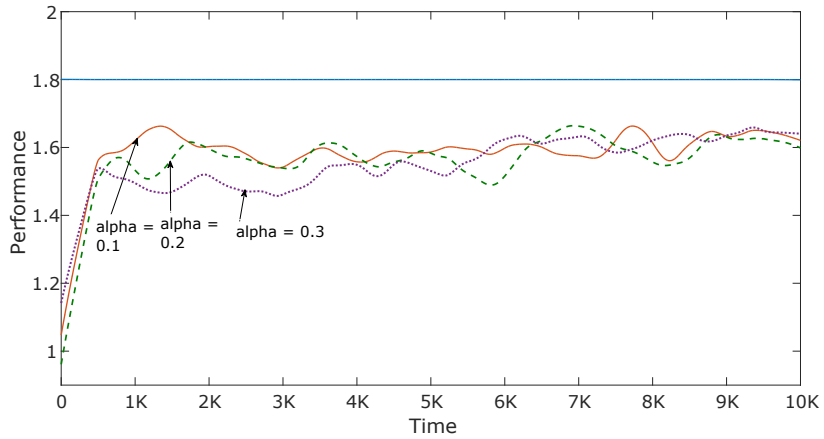
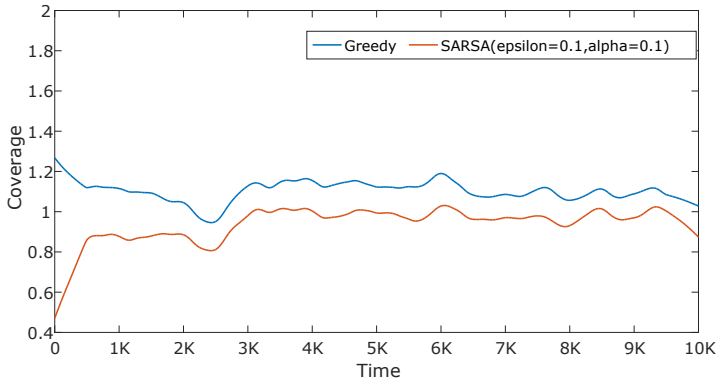
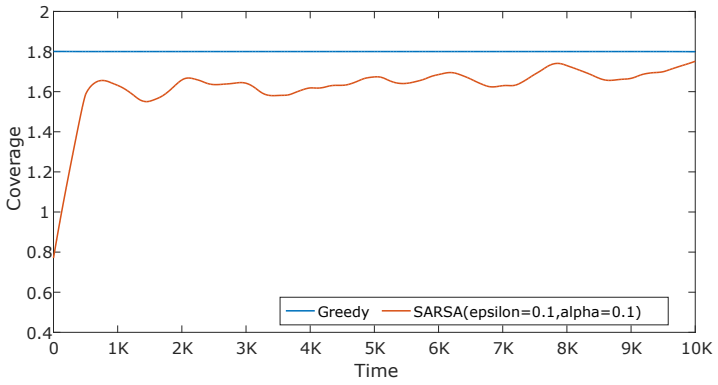


Figure 7.7: Impact of various learning rates on the performance of SARSA algorithm using scenario four.

Another interesting investigation carried out on the impact of employing different learning rate, $\alpha = 0.1, 0.2, 0.3$ on the coverage performance of the method with the results demonstrated in Figure 7.7. While all three different α values converge to almost the same coverage value at the end of the simulation run $T = 10,000$, however, with the learning rate of $\alpha = 0.1$, on-policy SARSA converge faster to that certain coverage value. According to these results, the TD on-policy SARSA method's implementation set up in this case study is considered as; $\alpha = 0.1$, and



(a) Scenario1



(b) Scenario4

Figure 7.8: A comparison of the coverage redundancy between SARSA in the solid red line, and greedy in the solid blue line across scenario one and four overtime.

$\epsilon = 0.1$.

In order to evaluate the desirability of each state-action pair, (s_t, a_t), a numerical reward is allocated, based on which the action-value function that represented in Equation 7.6 get updated. Our reward function is defined as following,

$$reward = \lambda \times U_i(O_i)$$

$$\lambda = \begin{cases} 1, & \text{if } O_{i(t)} < O_{i(t+1)} \\ 0, & \text{if } O_{i(t)} = O_{i(t+1)} \\ -1, & \text{if } O_{i(t)} > O_{i(t+1)} \end{cases} \tag{7.5}$$

Finally, after every transition from one state-action pair $Q(s_t, a_t)$ to the next $Q(s_{t+1}, a_{t+1})$ the Q-value is regularly updated using the following equation:

$$Q(s_t, a_t) \leftarrow Q(s_t, a_t) + \alpha[r_{t+1} + \gamma Q(s_{t+1}, a_{t+1}) - Q(s_t, a_t)] \tag{7.6}$$

Where, α is a learning parameter and γ $(0 \leq \gamma \leq 1)$ is the discount factor. In this manner, each camera aims to maximise the total reward while following a ϵ-greedy as both update and behaviour policies.

The performance of the algorithm evaluated in comparison to the greedy performance under two different type of environments. A stochastic environment with an example of scenario one when all objects adopt a random direction, versus a deterministic case, with an instance of scenario four where all objects follow a pre-defined trajectory all time.

Figure 7.8 demonstrate the results. The results from scenario four, indicate that learning environmental constraints in this case objects movement pattern over time, captured by online learning methods (e.g., a reinforcement learning approach) can lead to a dynamic zoom-selection behaviour at runtime. This turns to improve the coverage redundancy network-wide in a way to capture up to 81%, and 87% of the greedy performance in the cases of ϵ-greedy and SARSA respectively.

7.4.5 Coordinated Coverage Approach: Knowledge Sharing

So far, this chapter studied a set of online behaviours at the individual camera level, where there was no inter-camera communication between camera agents in improving coverage redundancy network-wide. Within this section, cameras are able to share their built-up knowledge, i.e. according to their Q-values using on-policy TD SARSA approach across a local neighbourhood group. A k-NEAREST strategy for

inter-camera communication is considered, which relies on the Euclidean distance between cameras. This approach termed as *Query Based-Sarsa* that benefits from both online reinforcement learning and inter-camera communication to utilise camera neighbour's knowledge as well at the time of decision making [122].

In this manner, first each individual camera adopts the on-policy TD SARSA prediction method, as discussed in Section 7.4.4. At a given time, a camera sends its state-action pair to its neighbours and requests a response. On the other side, by mapping the received state-action information to its own Q-table, each neighbour comes up with an action, i.e. a zoom which, based on its own updated Q-values, is the best-known action to take.

Decision Making Process

As a response model, the camera that receives all the responses from k neighbours, under the assumption of equal priors, runs a *majority voting* scheme across its own and the k responses [51]. In this way, the camera counts the votes received for this query from the individual neighbours. The zoom which receives the largest number of votes is then selected as the consensus (majority) decision. As a result, the camera exploits the outcome of the majority voting with a probability $1 - \epsilon$ while with a probability of ϵ, it still has a chance to explore other zooms.

The major difference between pure SARSA and QB-Sarsa happens in the exploitation part of the ϵ-greedy policy. In addition to pure *SARSA*, where a camera node employs a ϵ-greedy policy across its *own* local observations, (i.e. exploits the best-known zoom based on its learnt Q-vales so far). In QB-Sarsa, the ϵ-greedy policy exploits the outcome of majority voting scheme employed across $\mathsf{k}+1$ locally updated Q-values, with a probability of $1 - \epsilon$.

In this manner, the camera's decision in selecting its next zoom relies not only on

its own knowledge but also on the shared knowledge of the neighbours. This allows for a local camera neighbourhood itself comprising of cameras with limited observation to learn the environment constrains, i.e. objects movement pattern, faster than individuals. Consequently, in this coordinated approach, the results converge to the maximum possible coverage redundancy considerably faster than running pure learning on each individual camera agent.

7.5 Simulation results

In this section, the performance of four studied coverage approaches are compared in terms of k-coverage network-wide and discuss the advantages and disadvantages of them across all six test scenarios described in Table 7.1. It is essential to remind that prior to k-coverage calculations, all studied approaches, first employ the high fidelity abstract target detection model across the simulation environment to reason about the number of correctly detected targets at each zoom. An overview of this comparison demonstrated in Figure 7.9.

Looking at the results across different scenarios, it becomes apparent that the coverage redundancy is highly reliant on scenario properties. Based on the results, generally, the redundancy provided by almost all coverage approaches within the deterministic scenarios (with objects following the scripted pattern) is higher than the results of random ones.

A comparison across the outcomes indicates that applying even a simple learning scheme can make a substantial improvement on the performance of the online local approach, this can be found in a comparison between *Zoomout* results with three other online learning approaches. Given deterministic scenarios, the learning ap-

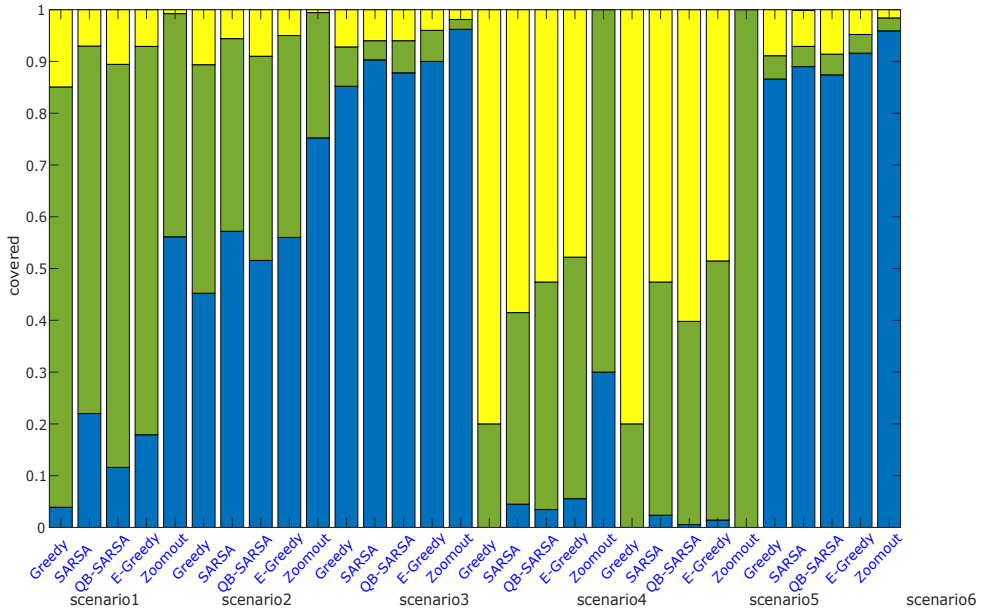


Figure 7.9: Illustration of the performance of the coverage approaches, *Greedy*, *SARSA*, *QB-SARSA*, *e-Greedy*, and *Zoomout* network-wide, across all six scenarios. The bottom blue bar represents 0-coverage, second green bar represents 1-coverage, and the top yellow bar illustrates k-coverage, where $k > 1$.

proaches can capture up to 87% of the offline greedy performance while performing during runtime.

The results show that enabling a knowledge-sharing among camera neighbours and combining it with our rescribed response model as a reactive behaviour can improve the performance of the pure learning approach (i.e. TD on-policy SARSA) even further. This can be inferred by comparing the zero coverage proportion (the blue bar) of both *SARSA* and *QB-SARSA* across all test scenarios.

It becomes evident that simply putting all cameras on the widest zoom (i.e.Zoomout) doesn't necessarily provide the highest coverage redundancy. Employing our high fidelity detection model, if the probability of detecting a target within a camera's FoV is less than a certain threshold, the target considered as *uncovered* across the given time interval. As described in Section 7.1, having only one uncovered target at each time slot drops the k-coverage value to zero across that time interval.

7.6 Conclusions and Discussion

In this chapter, a case study was established from coverage redundancy domain in smart camera networks to explore the implication of our proposed high fidelity target detection models. Emphasising the performance of a coverage approach highly relies on the reliability of target detection results. The fundamentals of the selected case study recognised. The problem of coverage redundancy was formalised along with the properties of camera nodes as self-organising agents in a dynamic environment. The coverage behaviours under this study were categorised to three different classes, offline, online, and more coordinated approaches.

All studied approaches, first employ the *high fidelity abstract target detection model* as a posterior probability to reason about the number of targets being correctly detected within a camera's FoV. It is important to note, keeping the computational expenses low in our proposed target detection model, makes it suitable for these type of runtime applications and online performance.

The on-policy TD SARSA learning method presented in this chapter is to our knowledge the most widely used reinforcement learning method due to its simplicity and being computationally cheap, which makes them suitable for online/runtime performance.

Our findings across the case study are listed as follows:

1. The coverage redundancy as a global metric is heavily reliant on scenario properties, in a way that environmental factors such as objects movement pattern, camera layout can easily affect the total quantity across the network.

2. Simply setting all the cameras to their widest zoom does not necessarily provide the highest coverage redundancy. Across the simulation, employing our

proposed model, if the outcomes of the model are below a certain threshold, the target was marked as *uncovered*.

3. While the *greedy* approach could provide the highest possible coverage redundancy across the network, it is computationally expensive and is not scalable or feasible for larger scenarios.

4. It becomes apparent that learning environmental constraints, i.e. objects movement pattern at the individual camera level, can leads to a dynamic zoom-selection behaviour at runtime. Thereby, the zoom selection behaviour of a camera can adapt to objects movement pattern over time; this then leads to a noticeable improvement in the redundancy provided network-wide. This becomes more apparent within deterministic scenarios where objects follow a scripted pattern all time.

5. It was shown that enabling knowledge sharing among camera neighbours can improve the k-coverage performance even further than the pure online learning approach.

A comprehensive comparison conducted across the results obtained from employing our proposed model against the CamSim standard model of detection over the studied coverage approaches in the next chapter.

Chapter 8

Implication of the High Fidelity Target Detection Model

Our particular focus in this short chapter is on exploration of the implication of utilising our high fidelity target detection model on the problem of k-coverage in smart camera networks as a case study initiated and studied in the previous chapter. To demonstrate this, we compare our results obtained in the previous chapter with the results of CamSim standard model of target detection across studied coverage approaches. As described in Chapter 6, the standard detection model of CamSim is an extremly abstract model of target detection, capturing only the camera's current zoom [31]. To achieve this, we replace the high fidelity detection model with the CamSim's standard detection model and repeat all the experiments conducted in Chapter 7. The target detection task studied in this thesis is a fundamental prerequisite of k-coverage calculation network-wide.

This short chapter proceeds as follows. Section 8.1 explains the implication of High Fidelity target detection model on studied k-coverage approaches. Comparing our results against initial results we discuss our findings. Section 8.2, concludes the chapter with a discussion.

8.1 Implication of High Fidelity Detection Model on k-coverage

Three coverage approaches, studied in the previous chapter, namely, Zoomout, Epsilon-greedy, and SARSA are selected for the purpose of this study. The outcomes of each approach compared applying, **i)** our high fidelity model, **ii)** CamSim standard model of detection. The results of this comparison demonstrated in Figure 8.1 across all test scenarios.

Analysing the results indicates that, while the results of CamSim detection model in the blue line, retain the variations of the previous results shown in red line **qualitatively** across all test scenarios, the results are **quantitatively** different.

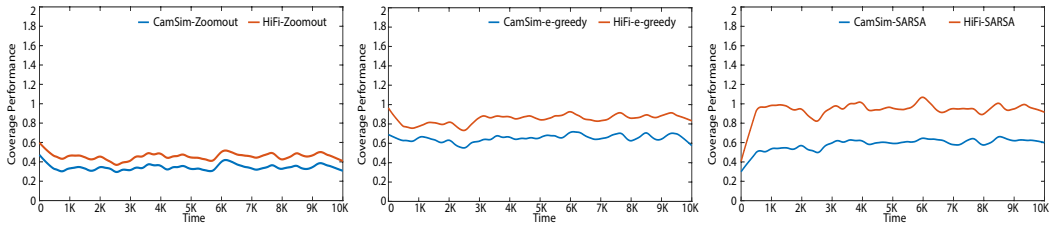
It can be observed that the performance of all selected coverage approaches improved in the presence of the high fidelity target detection model when compared to the results of CamSim standard model. This becomes apparent across all test scenarios. However, the amount of improvement obtained varies across the approaches and scenarios.

In some scenarios (e.g. scenario 3), this difference is minimal, however in others, CamSim's standard detection model underestimates the outcomes more substantially when compared to our high fidelity model of target detection. This becomes evident across all the test scenarios while using the same simulation settings.

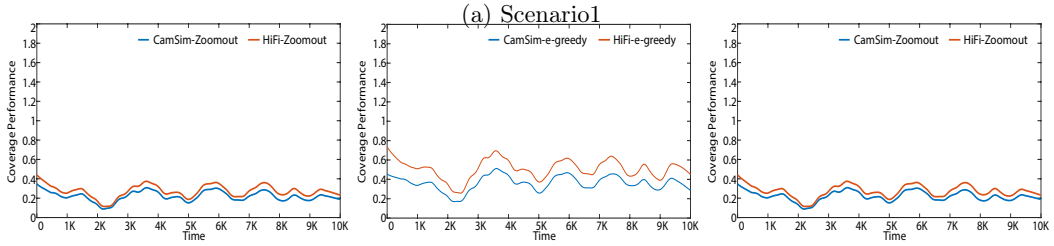
In order to quantify the impact of each detection model on the performance of the coverage approaches, we compute the proportion of the greedy results, achieved by each of these approaches.

The greedy approach analyses all potential configurations in the current time step, aiming to provide the maximum possible achieveable k-coverage using global knowledge of the network.

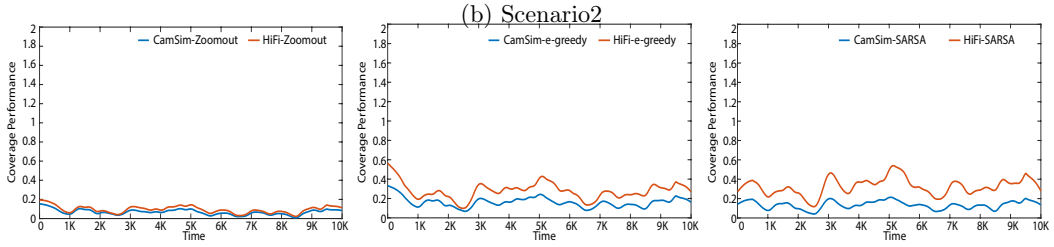
The results demonstrated in Table 8.1. Each cell of the table represents the



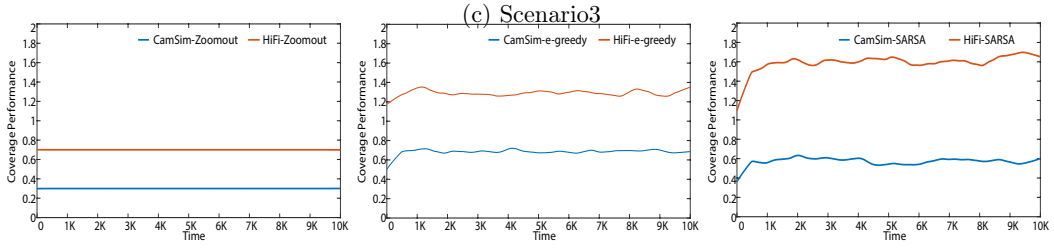
(a) Scenario1



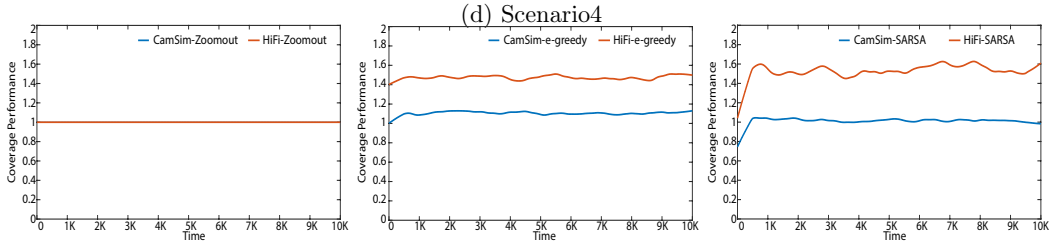
(b) Scenario2



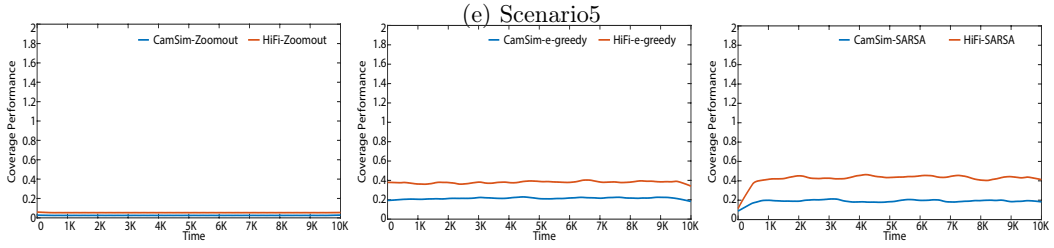
(c) Scenario3



(d) Scenario4



(e) Scenario5



(f) Scenario6

Figure 8.1: Graphs show a comparison between the performance of coverage approaches, *Zoomout,e-greedy,SARSA* utilising i. *CamSim*, the CamSim standard target detection model in a blue solid lines and ii. *HiFi* our proposed high fidelity target detection model in red solid lines across all test scenarios. The x-axis of all graphs shows the simulation time, $t = 10,000$ timesteps, and y-axis, demonstrates coverage performance across each scenario. From top to bottom, each row shows the results of each scenario. Also, from left to right each column shows the results of Zoomout, epsilon-greedy, and SARSA approaches respectively.

Table 8.1: The proportion of the greedy results achieved by each of the coverage approach across three test layouts under two different target detection models.

	Zoomout		E-Greedy		SARSA	
Layout	CamSim	$g(f_{lin}(q,d,z))$	CamSim	$g(f_{lin}(q,d,z))$	CamSim	$g(f_{lin}(q,d,z))$
Lattice	0.18	0.50	0.49	0.92	0.45	0.60
Ring	0.58	0.61	0.68	0.94	0.63	0.73
Cluster	0.05	0.44	0.64	0.93	0.57	0.70

proportion of the greedy results, achieved by each of these approaches across all test scenarios under employing two different target detection estimation models. Based on the results of the table, the performance of all studied coverage approaches with a *cluster* layout are the most affected by this underestimation.

8.2 Conclusions and Discussion

To explore the implication of our proposed target detection model, a comprehensive comparison was conducted employing i) high fidelity target detection model, ii) CamSim standard detection model across the studied coverage approaches It was shown that while the results of both detection models were qualitatively similar across all test scenarios, however, they were quantitatively different. The performance of all selected coverage approaches improved in the presence of the high fidelity target detection model when compared to the results of CamSim standard model. This becomes apparent across all test scenarios.

However, the amount of improvement obtained varies across the approaches and scenarios. In some scenarios, this difference was minimal, however in others, CamSim's standard off-the-shelf detection model underestimated the outcomes more substantially when compared to our high fidelity model of target detection. This becomes evident across all the test scenarios while using the same simulation settings.

So far, it was found that augmenting extremely abstract detection model of CamSim, to few more relative physical parameters leads to an improvement in the fidelity

of the model in reflecting ground truth confidences. Throughout the simulation results presented in this chapter, the improvement in the fidelity leads to detect an underestimation was accrued in the performance of the studied coverage approaches employing extremely abstract detection model of CamSim.

Coming back to the general concern in the agent-based modelling (described in the Introduction Chapter), a critical impact of simplified extreme abstract models (e.g. CamSim standard detection model) could be on the underestimation of the outcomes. This underestimation challenges the fidelity of the simulator's outcomes in reflecting real-world results. This problem identified and becomes evident throughout the study of this thesis.

Part IV

Conclusion and Final Remarks

Chapter 9

Conclusions and Future Work

Within this thesis, the abstract standard target detection model of CamSim SCN simulation environment was augmented with a higher degree of realism. The aim was to improve the fidelity of the simulator's outcomes in reflecting real-world's results while keeping the computational expense low.

The work was motivated by the identified trade-off between the fidelity of a simulator's outcomes and the corresponding computational overhead. The trade-off opened the possibility for an alternative to augment abstract simulation tools with a higher degree of realism. Thereby creat solutions that capture both benefits, low computational expense with a higher fidelity of the outcomes.

For the task of target detection across the abstract SCN simulator (CamSim), a novel decomposition method was proposed by introducing an intermediate point of representation. It was shown that establishing such an intermediate point brings flexibility and modularity into the design of the target detection model. This empowers practitioners to be able to select the model's features individually and independently to their requirements and camera settings. Further, it was illustrated that the established point could capture 76% of variations of ORB-ground truth confidences. Although adding an intermediate point in the design of the model came at

the cost of adding an extra layer of prediction errors. However, it was shown that still, a significant improvement (by almost 50%) was achieved when compared the fidelity of our model's outcomes against CamSim standard model of detection in predicting ground truth confidences.

Within surveillance systems, target tracking, and coverage analysis are two important applications that their performance highly relies on the reliability of target detection results. To explore the implications of our proposed model, we selected a case study from coverage redundancy domain of smart camera networks as an example of real-world applications. Emphasising that the performance of a coverage approach is highly influenced by the level of details captured by a camera. Thus, having reliable target detection results is an important prerequisite to coverage redundancy applications.

A comprehensive comparison was conducted across the performance of studied coverage approaches while employing i) high fidelity target detection model, ii) CamSim standard detection model. An underestimation was determined in the performance of the coverage approaches employing CamSim abstract detection model. The underestimation was quantified across the studied approaches. It was illustrated that depending on the scenarios, the underestimation of the outcomes could be substantial when compared to our high fidelity model of target detection.

9.1 Summary of Contributions

More specifically, this thesis provides the following contributions:

- In Chapter 3, we established and described the decomposition method by introducing PIP parameter, as an intermediate point of representation. PIP is a core element of the model that decouples the architecture into two partial

models.

- Within Chapter 3, we also created our image dataset using a real camera to establish ground truth PIP and a set of three ground truth confidences for further analysis.

- Chapter 4 explored the sufficiency of three physical properties in predicting ground truth PIP. We explored the existence of both linear and non-linear correlations using three state-of-the-art statistical analysis techniques.

- Chapter 5 three models of detectors were developed, combining three feature extraction methods with the visual similarity function. The sufficiency of the ground truth PIP was analysed in predicting the outcomes of three detectors, developing a linear regression. It was illustrated that there is a linear correlation exists between PIP and results of ORB features with higher accuracy when compared to SIFT and SURF features.

- Within Chapter 6 explored the sufficiency of predicted-PIP in reflecting three ground truth confidences. Fidelity of the predictions explored against CamSim detection model's outcomes in predicting ground truth confidences. A significant improvement was achieved in the fidelity of our results when compared to CamSim initial results.

- Chapter 7, a case study was selected from coverage redundancy domain of smart camera networks to explore the implication of our high fidelity target detection model. Highlighting the importance of having reliable target detection results as a prerequisite to coverage redundancy applications.

- In Chapter 8, a comprehensive comparison conducted employing i) high fidelity target detection model, ii) CamSim standard detection model across the

coverage approaches. An underestimation was detected in the performance of all studied approaches employing CamSim standard abstract detection model when compared to our results.

Addressing the general concern in agent-based modelling about the unclear impact of making abstract models on the fidelity of the simulator's outcomes, it was found that one example of this impact could be an underestimation in the simulation's results. Throughout the selected case study, employing the extremely abstract model of CamSim, capturing only one property of realism led to a substantial underestimation in the performance of studied coverage approaches. Coming back to the main objectives of the thesis, it was shown augmenting abstract simulation tools such as CamSim with a higher degree of realism, could improve the fidelity of the outcomes by almost 50% in reflecting real-world outcomes when compared to the initial results.

With our proposed approach, given the resolution of a camera's image sensor, the impact of only three physical parameters, size of the target, distance from the camera, and the camera's current zoom investigated in pixel deviation of the introduced intermediate point, PIP. Nevertheless, in realistic modelling of target detection, the real-world constraints imposed by environmental factors and camera setting such as the camera's aperture, lens distortion, environment lightening must be taken into account. The proposed model is purposefully generic and abstract with the objective to capture both low computational expense and high fidelity in reflecting real-world outcomes. These objectives are useful in supporting the applicability of the model to a broader range of agent-based application domains facing the identified trade-off. Indeed, within simulation environments, incorporating more properties of realism,

aiming to represent a particular phenomenon in the real world, often leads to higher accuracy and fidelity of the outcomes in reflecting real-world operations.

It is important at this stage to also identify weaknesses with the approach and analysis presented. As described earlier, establishing an intermediate point of representation in the design of the model comes at the cost of adding an extra layer of prediction errors to the final outcomes of the model. This leads to a slight decrease (around 6%) in the total predictive ability of the model in capturing the ground truth confidences. This impact varies depending on the employed regression method within the first part of the architecture.

9.2 Future Work

The future directions identified for this work fall into a number of distinct areas. The composability principle behind the proposed model development architecture opens an essential direction for future work. In this context, to investigate the impact of a wide range of different classification and regression techniques on the fidelity of the model's outcomes. Within the second part, including deep features obtained from powerful convolutional neural networks or even more complex hybrid techniques that combine the local features and deep feature (with supporting benefits of both methodologies [134, 74]) could be an interesting direction to extend the work.

As described earlier, in this thesis, we introduced a new parameter, PIP, capturing a ratio of the pixel density of a patch to an entire image. However, given the resolution of a camera's image sensor, we studied the impact of only three physical properties on the pixel deviation of PIP. This confirms that the model presented in this thesis is purposefully generic and abstract to support the applicability to those

class of systems that face the trade-off between fidelity and corresponding computational expense. Nevertheless, to develop realistic detection models, it is necessary to consider the specific constraints imposed by other important environmental and camera factors, such as the camera's aperture, lens distortion, environment lightening, on the pixel density of PIP. It is attributed that incorporating these factors in the model's development could increase the accuracy of the predictions.

One important future study could be in the concept of finding the right amount of fidelity for the problem at hand. Based on the results of this study, we observed that adding only a few more physical properties could lead to significant improvement in the accuracy of the simulation's outcomes. However, we speculate that from a certain point adding more and more details won't add a significant gain towards the accuracy. Therefore it is important to conduct a systematic research to understand the identical elements between real world and simulation and their impact on the accuracy of the outcomes. In this way, we could have a cost-efficient design, where the extra degrees of fidelity are eliminated and the accuracy of the outcomes are desirable.

Finally, the unclear impact of making simplified abstract models on the ability of the simulator to capture real-world behaviours is an important open question which is also a general open concern in agent-based modelling [36]. The underestimation identified within the outcomes of the simulator across the selected case study, employing an extremely abstract model of detection, is one important example of this statement. This was determined when compared to the results of our high fidelity detection model. The coverage redundancy in smart camera networks is one example case study to explor the implication of our model. Nevertheless, this exploration can be extended to other survillence applications such as target tracking.

Augmenting the extremely abstract model of detection to capture a few more properties of realism using a novel decomposition method with an intermediate point of representation, found to be successful in dealing with the identified trade-off across the selected case study.

This opens an interesting avenue to extend the work to apply to other agent-based modelling/simulation application domains [115] that face with the identified trade-off between fidelity of the system's outcomes and corresponding computational expense. Examples of this includes, agent-based applications in health care [73], social science [37] amongst others. The generic nature of the approach presented in this thesis will likely be applicable across these scenarios were the identified trade-off is faced.

Bibliography

[1] Zoë Abrams, Ashish Goel, and Serge Plotkin. Set k-cover algorithms for energy efficient monitoring in wireless sensor networks. In Proc. of Int. Conf. on Information Processing in Sensor Networks, pages 424–432, 2004.

[2] Hamid Aghajan and Andrea Cavallaro. Multi-camera networks: principles and applications. Academic press, 2009.

[3] Peter Auer, Nicolo Cesa-Bianchi, and Paul Fischer. Finite-time analysis of the multiarmed bandit problem. Machine learning, 47(2-3):235–256, 2002.

[4] Wolfgang Banzhaf, Peter Nordin, Robert E Keller, and Frank D Francone. Genetic programming: an introduction, volume 1. Morgan Kaufmann San Francisco, 1998.

[5] Herbert Bay, Andreas Ess, Tinne Tuytelaars, and Luc Van Gool. Speeded-up robust features (surf). Computer vision and image understanding, 110(3):346–359, 2008.

[6] Herbert Bay, Tinne Tuytelaars, and Luc Van Gool. Surf: Speeded up robust features. In European conference on computer vision, pages 404–417. Springer, 2006.

[7] Richard Bellman. A markovian decision process. Journal of mathematics and mechanics, pages 679–684, 1957.

[8] Kristin P Bennett and Olvi L Mangasarian. Robust linear programming discrimination of two linearly inseparable sets. Optimization methods and software, 1(1):23–34, 1992.

[9] Niccoló Bisagno, Nicola Conci, and Bernhard Rinner. Dynamic camera network reconfiguration for crowd surveillance. In Proceedings of the 12th International Conference on Distributed Smart Cameras, page 4. ACM, 2018.

[10] Galen Bollinger. Book review: Regression diagnostics: Identifying influential data and sources of collinearity, 1981.

[11] Athanassios Boulis et al. Castalia: A simulator for wireless sensor networks and body area networks. NICTA: National ICT Australia, 83, 2011.

[12] Stephen Boyd and Lieven Vandenberghe. Convex optimization. Cambridge university press, 2004.

[13] Gary Bradski and Adrian Kaehler. Learning OpenCV: Computer vision with the OpenCV library. " O'Reilly Media, Inc.", 2008.

[14] Elizabeth Bruch and Jon Atwell. Agent-based models in empirical social research. Sociological methods & research, 44(2):186–221, 2015.

[15] Christopher JC Burges. A tutorial on support vector machines for pattern recognition. Data mining and knowledge discovery, 2(2):121–167, 1998.

[16] Michael Calonder, Vincent Lepetit, Christoph Strecha, and Pascal Fua. Brief: Binary robust independent elementary features. In European conference on computer vision, pages 778–792. Springer, 2010.

[17] Mihaela Cardei and Ding-Zhu Du. Improving wireless sensor network lifetime through power aware organization. Wireless Networks, 11(3):333–340, 2005.

[18] Chih-Chung Chang and Chih-Jen Lin. Libsvm: A library for support vector machines. ACM transactions on intelligent systems and technology (TIST), 2(3):27, 2011.

[19] Gal Chechik, Varun Sharma, Uri Shalit, and Samy Bengio. Large scale online learning of image similarity through ranking. Journal of Machine Learning Research, 11(Mar):1109–1135, 2010.

[20] Gong Cheng and Junwei Han. A survey on object detection in optical remote sensing images. ISPRS Journal of Photogrammetry and Remote Sensing, 117:11–28, 2016.

[21] Nello Cristianini, John Shawe-Taylor, et al. An introduction to support vector machines and other kernel-based learning methods. Cambridge university press, 2000.

[22] Franklin C Crow. Summed-area tables for texture mapping. In ACM SIGGRAPH computer graphics, volume 18, pages 207–212. ACM, 1984.

[23] G Cybenko. Continuous valued neural networks with two hidden layers are sufficient, department of computer science. Trfts. University, 1988.

[24] Navneet Dalal and Bill Triggs. Histograms of oriented gradients for human detection. In 2005 IEEE computer society conference on computer vision and pattern recognition (CVPR'05), volume 1, pages 886–893. IEEE, 2005.

[25] Norman R Draper and Harry Smith. Applied regression analysis, volume 326. John Wiley & Sons, 1998.

[26] Harris Drucker, Christopher JC Burges, Linda Kaufman, Alex J Smola, and Vladimir Vapnik. Support vector regression machines. In Advances in neural information processing systems, pages 155–161, 1997.

[27] Lukas Esterle and Peter Lewis. Online Multi-object k-coverage with Mobile Smart Cameras. In Proc. of Int. Conf. on Distributed Smart Cameras, pages 1–6, 2017.

[28] Lukas Esterle, Peter R Lewis, Horatio Caine, Xin Yao, and Bernhard Rinner. Camsim: A distributed smart camera network simulator. In Proc. of Int. Conf. on Self-Adaptation and Self-Organizing Systems, pages 19–20, 2013.

[29] Lukas Esterle, Peter R Lewis, Richie McBride, and Xin Yao. The future of camera networks: Staying smart in a chaotic world. In Proceedings of the 11th International Conference on Distributed Smart Cameras, pages 163–168. ACM, 2017.

[30] Lukas Esterle, Peter R Lewis, Xin Yao, and Bernhard Rinner. Socio-economic vision graph generation and handover in distributed smart camera networks. ACM Transactions on Sensor Networks, 10(2):20, 2014.

[31] Lukas Esterle, Bernhard Rinner, and Peter R Lewis. Self-organising zooms for decentralised redundancy management in visual sensor networks. In Proc. of Int. Conf. on Self-Adaptive and Self-Organizing Systems, pages 41–50, 2015.

[32] Lukas Esterle, Bernhard Rinner, Peter R Lewis, and Xin Yao. Improved adaptivity and robustness in decentralised multi-camera networks. 2012.

[33] David A Forsyth and Jean Ponce. Computer vision: a modern approach. Prentice Hall Professional Technical Reference, 2002.

[34] Noa Garcia Docampo. Spatial and temporal representations for multi-modal visual retrieval. 2018.

[35] Michael R Garey and David S Johnson. A guide to the theory of np-completeness. WH Freemann, New York, 70, 1979.

[36] Nigel Gilbert. Agent-based models. Number 153. Sage, 2008.

[37] Nigel Gilbert and Pietro Terna. How to build and use agent-based models in social science. Mind & Society, 1(1):57–72, 2000.

[38] Allen R Greenleaf. Photographic optics. Macmillan, 1950.

[39] Michael A Gruber, Melanie Schranz, and Bernhard Rinner. The extended vsnsim for hybrid camera systems. In Proceedings of the 9th International Conference on Distributed Smart Cameras, pages 203–204. ACM, 2015.

[40] Alex Guazzelli, Michael Zeller, Wen-Ching Lin, Graham Williams, et al. Pmml: An open standard for sharing models. The R Journal, 1(1):60–65, 2009.

[41] Joachim Gudmundsson, Patrick Laube, and Thomas Wolle. Movement patterns in spatio-temporal data. Encyclopedia of GIS, pages 726–732, 2008.

[42] Richard Hartley and Andrew Zisserman. Multiple view geometry in computer vision. Cambridge university press, 2003.

[43] Douglas M Hawkins, Subhash C Basak, and Denise Mills. Assessing model fit by cross-validation. Journal of chemical information and computer sciences, 43(2):579–586, 2003.

[44] Mohamed Hefeeda and Majid Bagheri. Randomized k-coverage algorithms for dense sensor networks. In Proc. of Int. Conf. on Computer Communications, pages 2376–2380, 2007.

[45] Inge S Helland. On the interpretation and use of r2 in regression analysis. Biometrics, pages 61–69, 1987.

[46] Berthold KP Horn and Brian G Schunck. Determining optical flow. Artificial intelligence, 17(1-3):185–203, 1981.

[47] Chi-Fu Huang and Yu-Chee Tseng. The coverage problem in a wireless sensor network. Mobile Networks and Applications, 10(4):519–528, 2005.

[48] Younggwan Jo and Joonhee Han. A new approach to camera hand-off without camera calibration for the general scene with non-planar ground. In Proceedings of the 4th ACM international workshop on Video surveillance and sensor networks, pages 195–202. ACM, 2006.

[49] Leslie Pack Kaelbling, Michael L Littman, and Andrew W Moore. Reinforcement learning: A survey. Journal of artificial intelligence research, 4:237–285, 1996.

[50] Sohaib Khan and Mubarak Shah. Consistent labeling of tracked objects in multiple cameras with overlapping fields of view. IEEE Transactions on Pattern Analysis and Machine Intelligence, 25(10):1355–1360, 2003.

[51] Josef Kittler, Mohamad Hatef, Robert PW Duin, and Jiri Matas. On combining classifiers. IEEE transactions on pattern analysis and machine intelligence, 20(3):226–239, 1998.

[52] John R Koza. Genetic programming ii: Automatic discovery of reusable subprograms. Cambridge, MA, USA, 13(8):32, 1994.

[53] John R Koza and John R Koza. Genetic programming: on the programming of computers by means of natural selection, volume 1. MIT press, 1992.

[54] Max Kuhn and Kjell Johnson. Applied predictive modeling, volume 26. Springer, 2013.

[55] Tarald O Kvålseth. Cautionary note about r 2. The American Statistician, 39(4):279–285, 1985.

[56] William B Langdon and Riccardo Poli. Foundations of genetic programming. Springer Science & Business Media, 2013.

[57] Patrick Laube, Matt Duckham, and Thomas Wolle. Decentralized movement pattern detection amongst mobile geosensor nodes. In International Conference on Geographic Information Science, pages 199–216. Springer, 2008.

[58] Peter R Lewis, Lukas Esterle, Arjun Chandra, Bernhard Rinner, Jim Torresen, and Xin Yao. Static, dynamic, and adaptive heterogeneity in distributed smart camera networks. ACM Transactions on Autonomous and Adaptive Systems (TAAS), 10(2):8, 2015.

[59] Peter R Lewis, Lukas Esterle, Arjun Chandra, Bernhard Rinner, and Xin Yao. Learning to be different: Heterogeneity and efficiency in distributed smart camera networks. In Self-Adaptive and Self-Organizing Systems (SASO), 2013 IEEE 7th International Conference on, pages 209–218. IEEE, 2013.

[60] Peter R Lewis, Marco Platzner, Bernhard Rinner, Jim Tørresen, and Xin Yao. Self-Aware Computing Systems. Springer, 2016.

[61] Dahai Liu, Nikolas D Macchiarella, and Dennis A Vincenzi. Simulation fidelity. Human factors in simulation and training, pages 61–73, 2008.

[62] David G Lowe. Distinctive image features from scale-invariant keypoints. International journal of computer vision, 60(2):91–110, 2004.

[63] Gary Marcus. Deep learning: A critical appraisal. arXiv preprint arXiv:1801.00631, 2018.

[64] Gary F Marcus. Rethinking eliminative connectionism. Cognitive psychology, 37(3):243–282, 1998.

[65] J Kent Martin and Daniel S Hirschberg. Small sample statistics for classification error rates I: Error rate measurements. Information and Computer Science, University of California, Irvine, 1996.

[66] Krystian Mikolajczyk and Cordelia Schmid. A performance evaluation of local descriptors. 2005.

[67] Annette M Molinaro, Richard Simon, and Ruth M Pfeiffer. Prediction error estimation: a comparison of resampling methods. Bioinformatics, 21(15):3301–3307, 2005.

[68] K-R Müller, Alexander J Smola, Gunnar Rätsch, Bernhard Schölkopf, Jens Kohlmorgen, and Vladimir Vapnik. Predicting time series with support vector machines. In International Conference on Artificial Neural Networks, pages 999–1004. Springer, 1997.

[69] Noboru Murata. An integral representation of functions using three-layered networks and their approximation bounds. Neural Networks, 9(6):947–956, 1996.

[70] Raymond H Myers and Raymond H Myers. Classical and modern regression with applications, volume 2. Duxbury press Belmont, CA, 1990.

[71] Tormod Næs and Bjørn-Helge Mevik. Understanding the collinearity problem in regression and discriminant analysis. Journal of Chemometrics: A Journal of the Chemometrics Society, 15(4):413–426, 2001.

[72] Nico JD Nagelkerke et al. A note on a general definition of the coefficient of determination. Biometrika, 78(3):691–692, 1991.

[73] John Nealon and Antonio Moreno. Agent-based applications in health care. In Applications of software agent technology in the health care domain, pages 3–18. Springer, 2003.

[74] Rahul Nijhawan, Josodhir Das, and Balasubramanian Raman. A hybrid of deep learning and hand-crafted features based approach for snow cover mapping. International journal of remote sensing, 40(2):759–773, 2019.

[75] Mark Nixon and Alberto S Aguado. Feature extraction and image processing for computer vision. Academic Press, 2012.

[76] Haydemar Núñez, Cecilio Angulo, and Andreu Català. Rule extraction from support vector machines. In Esann, pages 107–112, 2002.

[77] Niall O'Mahony, Sean Campbell, Anderson Carvalho, Suman Harapanahalli, Gustavo Velasco Hernandez, Lenka Krpalkova, Daniel Riordan, and Joseph Walsh. Deep learning vs. traditional computer vision. In Science and Information Conference, pages 128–144. Springer, 2019.

[78] Johnny Park, Priya C Bhat, and Avinash C Kak. A look-up table based approach for solving the camera selection problem in large camera networks. In Workshop on Distributed Smart Cameras, volume 2, 2006.

[79] Fabian Pedregosa, Gaël Varoquaux, Alexandre Gramfort, Vincent Michel, Bertrand Thirion, Olivier Grisel, Mathieu Blondel, Peter Prettenhofer, Ron Weiss, Vincent Dubourg, et al. Scikit-learn: Machine learning in python. Journal of machine learning research, 12(Oct):2825–2830, 2011.

[80] Congduc Pham and Abdallah Makhoul. Performance study of multiple cover-set strategies for mission-critical video surveillance with wireless video sensors. In 2010 IEEE 6th International Conference on Wireless and Mobile Computing, Networking and Communications, pages 208–216. IEEE, 2010.

[81] Andreas Pyka and Giorgio Fagiolo. 29 agent-based modelling: a methodology for neo-schumpeterian economics'. Elgar companion to neo-schumpeterian economics, 467, 2007.

[82] Faisal Qureshi and Demetri Terzopoulos. Smart camera networks in virtual reality. Proceedings of the IEEE, 96(10):1640–1656, 2008.

[83] Faisal Z Qureshi and Demetri Terzopoulos. Surveillance in virtual reality: System design and multi-camera control. In Computer Vision and Pattern Recognition, 2007. CVPR'07. IEEE Conference on, pages 1–8. IEEE, 2007.

[84] Hossein Izadi Rad, Ji Feng, and Hitoshi Iba. Gp-rvm: Genetic programing-based symbolic regression using relevance vector machine. arXiv preprint arXiv:1806.02502, 2018.

[85] Denis Rosário, Zhongliang Zhao, Claudio Silva, Eduardo Cerqueira, and Torsten Braun. An omnet++ framework to evaluate video transmission in mobile wireless multimedia sensor networks. In Proceedings of the 6th International ICST Conference on Simulation Tools and Techniques, pages 277–284. ICST (Institute for Computer Sciences, Social-Informatics and, 2013.

[86] Edward Rosten and Tom Drummond. Machine learning for high-speed corner detection. In European conference on computer vision, pages 430–443. Springer, 2006.

[87] Patrick Royston. Lowess smoothing. Stata technical bulletin, 1(3), 1992.

[88] Ethan Rublee, Vincent Rabaud, Kurt Konolige, and Gary Bradski. Orb: An efficient alternative to sift or surf. In Computer Vision (ICCV), 2011 IEEE international conference on, pages 2564–2571. IEEE, 2011.

[89] Gavin A Rummery and Mahesan Niranjan. Online Q-learning using connectionist systems, volume 37. University of Cambridge, Department of Engineering Cambridge, England, 1994.

[90] Stuart J Russell and Peter Norvig. Artificial intelligence: a modern approach. Malaysia; Pearson Education Limited,, 2016.

[91] J. C. SanMiguel and A. Cavallaro. Networked computer vision: The importance of a holistic simulator. Computer, 50(7):35–43, 2017.

[92] Juan C SanMiguel and Andrea Cavallaro. Networked computer vision: the importance of a holistic simulator. Computer, 50(7):35–43, 2017.

[93] Juan Carlos SanMiguel and Andrea Cavallaro. Energy consumption models for smart camera networks. IEEE Transactions on Circuits and Systems for Video Technology, 27(12):2661–2674, 2016.

[94] PB Schiilkop, Chris Burgest, and Vladimir Vapnik. Extracting support data for a given task. In Proceedings of the 1st international conference on knowledge discovery & data mining, pages 252–257, 1995.

[95] Bernhard Schölkopf, Chris Burges, and Vladimir Vapnik. Incorporating invariances in support vector learning machines. In International Conference on Artificial Neural Networks, pages 47–52. Springer, 1996.

[96] Bernhard Schölkopf, Alexander J Smola, Francis Bach, et al. Learning with kernels: support vector machines, regularization, optimization, and beyond. MIT press, 2002.

[97] Melanie Schranz and Bernhard Rinner. Vsnsim-a simulator for control and coordination in visual sensor networks. In Proceedings of the International Conference on Distributed Smart Cameras, page 44. ACM, 2014.

[98] Bradley C Schricker, Robert W Franceschini, and Timothy C Johnson. Fidelity evaluation framework. In Proceedings. 34th Annual Simulation Symposium, pages 109–116. IEEE, 2001.

[99] David W Scott. Multivariate density estimation: theory, practice, and visualization. John Wiley & Sons, 2015.

[100] George AF Seber and Alan J Lee. Linear regression analysis, volume 329. John Wiley & Sons, 2012.

[101] Sara Silva and Ernesto Costa. Dynamic limits for bloat control in genetic programming and a review of past and current bloat theories. Genetic Programming and Evolvable Machines, 10(2):141–179, 2009.

[102] Satinder Singh, Tommi Jaakkola, Michael L Littman, and Csaba Szepesvári. Convergence results for single-step on-policy reinforcement-learning algorithms. Machine learning, 38(3):287–308, 2000.

[103] Josef Sivic and Andrew Zisserman. Video google: A text retrieval approach to object matching in videos. In null, page 1470. IEEE, 2003.

[104] Guido F Smits and Mark Kotanchek. Pareto-front exploitation in symbolic regression. In Genetic programming theory and practice II, pages 283–299. Springer, 2005.

[105] Alex J Smola et al. Regression estimation with support vector learning machines. PhD thesis, Master's thesis, Technische Universität München, 1996.

[106] Alex J Smola and Bernhard Schölkopf. A tutorial on support vector regression. Statistics and computing, 14(3):199–222, 2004.

[107] George W Snedecor and Witiiam G Cochran. Statistical methods, 8thedn. Ames: Iowa State Univ. Press Iowa, 1989.

[108] Wiktor Starzyk and Faisal Z Qureshi. Learning proactive control strategies for ptz cameras. In 2011 Fifth ACM/IEEE International Conference on Distributed Smart Cameras, pages 1–6. IEEE, 2011.

[109] Wiktor Starzyk and Faisal Z Qureshi. Software laboratory for camera networks research. IEEE Journal on Emerging and Selected Topics in Circuits and Systems, 3(2):284–293, 2013.

[110] Richard S Sutton. Learning to predict by the methods of temporal differences. Machine learning, 3(1):9–44, 1988.

[111] Richard S Sutton. Generalization in reinforcement learning: Successful examples using sparse coarse coding. In Advances in neural information processing systems, pages 1038–1044, 1996.

[112] Richard S Sutton and Andrew G Barto. Introduction to reinforcement learning, volume 135. MIT press Cambridge, 1998.

[113] Andrew Symington, Sonia Waharte, Simon Julier, and Niki Trigoni. Probabilistic target detection by camera-equipped uavs. In 2010 IEEE International Conference on Robotics and Automation, pages 4076–4081. IEEE, 2010.

[114] Geoffrey R Taylor, Andrew J Chosak, and Paul C Brewer. Ovvv: Using virtual worlds to design and evaluate surveillance systems. In Computer Vision and Pattern Recognition, 2007. CVPR'07. IEEE Conference on, pages 1–8. IEEE, 2007.

[115] Takao Terano, Hajime Kita, Toshiyuki Kaneda, Kiyoshi Arai, and Hiroshi Deguchi. Agent-Based Simulation: From Modeling Methodologies to Real-World Applications: Post Proceedings of the Third International Workshop on Agent-Based Approaches in Economic and Social Complex Systems 2004, volume 1. Springer Science & Business Media, 2006.

[116] Matthew Uyttendaele, Ashley Eden, and Richard Skeliski. Eliminating ghosting and exposure artifacts in image mosaics. In Proceedings of the 2001 IEEE Computer Society Conference on Computer Vision and Pattern Recognition. CVPR 2001, volume 2, pages II–II. IEEE, 2001.

[117] Maria Valera and Sergio A Velastin. Intelligent distributed surveillance systems: a review. IEE Proceedings-Vision, Image and Signal Processing, 152(2):192–204, 2005.

[118] Harm Van Seijen, Hado Van Hasselt, Shimon Whiteson, and Marco Wiering. A theoretical and empirical analysis of expected sarsa. In 2009 IEEE Symposium on Adaptive Dynamic Programming and Reinforcement Learning, pages 177–184. IEEE, 2009.

[119] V Vapnik and Vlamimir Vapnik. Statistical learning theory wiley. New York, pages 156–160, 1998.

[120] Vladimir Vapnik. the nature of statistical learning theory. Springer science & business media, 2013.

[121] András Varga. Using the omnet++ discrete event simulation system in education. IEEE Transactions on Education, 42(4):11–pp, 1999.

[122] Arezoo Vejdanparast and Peter R Lewis. Learning and sharing for improved k-coverage in smart camera networks. In 2019 IEEE 4th International Workshops on Foundations and Applications of Self* Systems (FAS* W), pages 80–85. IEEE, 2019.

[123] Arezoo Vejdanparast, Peter R Lewis, and Lukas Esterle. Online zoom selection approaches for coverage redundancy in visual sensor networks. In Proceedings of the 12th International Conference on Distributed Smart Cameras, page 15. ACM, 2018.

[124] Joannes Vermorel and Mehryar Mohri. Multi-armed bandit algorithms and empirical evaluation. In ECML, volume 3720, pages 437–448, 2005.

[125] Paul Viola, Michael J Jones, and Daniel Snow. Detecting pedestrians using patterns of motion and appearance. International Journal of Computer Vision, 63(2):153–161, 2005.

[126] Ekaterina J Vladislavleva, Guido F Smits, and Dick Den Hertog. Order of nonlinearity as a complexity measure for models generated by symbolic regression via pareto genetic programming. IEEE Transactions on Evolutionary Computation, 13(2):333–349, 2008.

[127] Matt P Wand and M Chris Jones. Kernel smoothing. Chapman and Hall/CRC, 1994.

[128] Bang Wang. Coverage problems in sensor networks: A survey. ACM Computing Surveys, 43(4):32, 2011.

[129] Junxian Wang, George Bebis, and Ronald Miller. Robust video-based surveillance by integrating target detection with tracking. In 2006 Conference on Computer Vision and Pattern Recognition Workshop (CVPRW'06), pages 137–137. IEEE, 2006.

[130] Christopher JCH Watkins and Peter Dayan. Q-learning. Machine learning, 8(3-4):279–292, 1992.

[131] William H Widen. Smart cameras and the right to privacy. Proceedings of the IEEE, 96(10):1688–1697, 2008.

[132] Wayne Wolf, Burak Ozer, and Tiehan Lv. Smart cameras as embedded systems. Computer, 35(9):48–53, 2002.

[133] Chong Ho Yu. Exploratory data analysis. Methods, 2:131–160, 1977.

[134] Guohang Zeng, Jiancan Zhou, Xi Jia, Weicheng Xie, and Linlin Shen. Hand-crafted feature guided deep learning for facial expression recognition. In 2018 13th IEEE International Conference on Automatic Face & Gesture Recognition (FG 2018), pages 423–430. IEEE, 2018.

[135] Huiyu Zhou, Murtaza Taj, and Andrea Cavallaro. Target detection and tracking with heterogeneous sensors. IEEE Journal of Selected Topics in Signal Processing, 2(4):503–513, 2008.