Modeling and Optimizing the Coverage of Multi-Camera Systems

Aaron Mavrinac
University of Windsor

Follow this and additional works at: https://scholar.uwindsor.ca/etd

Recommended Citation

This online database contains the full-text of PhD dissertations and Masters' theses of University of Windsor students from 1954 forward. These documents are made available for personal study and research purposes only, in accordance with the Canadian Copyright Act and the Creative Commons license—CC BY-NC-ND (Attribution, Non-Commercial, No Derivative Works). Under this license, works must always be attributed to the copyright holder (original author), cannot be used for any commercial purposes, and may not be altered. Any other use would require the permission of the copyright holder. Students may inquire about withdrawing their dissertation and/or thesis from this database. For additional inquiries, please contact the repository administrator via email (scholarship@uwindsor.ca) or by telephone at 519-253-3000 ext. 3208.
MODELING AND OPTIMIZING THE
COVERAGE OF MULTI-CAMERA SYSTEMS

by

Aaron Mavrinac

A DISSERTATION SUBMITTED TO THE FACULTY OF GRADUATE
STUDIES THROUGH ELECTRICAL AND COMPUTER ENGINEERING IN
PARTIAL FULFILLMENT OF THE REQUIREMENTS FOR THE DEGREE
OF DOCTOR OF PHILOSOPHY AT THE UNIVERSITY OF WINDSOR

Windsor, Ontario, Canada
2012

Copyright © 2012 Aaron Mavrinac
MODELING AND OPTIMIZING THE COVERAGE OF MULTI-CAMERA SYSTEMS

by
Aaron Mavrinac

APPROVED BY:

---

Dr. H. H. Liu, External Examiner
Institute for Aerospace Studies, University of Toronto

---

Dr. B. Boufama, Outside Department Reader
School of Computer Science

---

Dr. J. Wu, 1st Departmental Reader
Department of Electrical and Computer Engineering

---

Dr. M. Ahmadi, 2nd Departmental Reader
Department of Electrical and Computer Engineering

---

Dr. X. Chen, Advisor
Department of Electrical and Computer Engineering

---

Dr. J. P. Coyle, Chair of Defense
School of Social Work

August 9, 2012
# Declaration of Previous Publication

This thesis includes four original papers that have been previously published/Submitted for publication in peer reviewed journals, as follows:

<table>
<thead>
<tr>
<th>Thesis Chapter</th>
<th>Citation</th>
<th>Status</th>
</tr>
</thead>
<tbody>
<tr>
<td>Chapter 3</td>
<td>A. Mavrinac and X. Chen, &quot;Modeling Coverage in Camera Networks: A Survey,&quot; submitted to <em>Int. J. Computer Vision</em>, manuscript no. VISI2069R2.</td>
<td>Accepted</td>
</tr>
<tr>
<td>Chapter 8</td>
<td>A. Mavrinac and X. Chen, &quot;Optimizing Load Distribution in Camera Networks with a Hypergraph Model of Coverage Topology,&quot; in <em>Proc. 5th ACM/IEEE Int. Conf. Distributed Smart Cameras</em>, 2011. DOI:10.1109/ICDSC.2011.6042903</td>
<td>Published</td>
</tr>
</tbody>
</table>

I certify that I have obtained a written permission from the copyright owner(s) to include the above published material(s) in my thesis. I certify that the above material describes work completed during my registration as graduate student at the University of Windsor.

I declare that, to the best of my knowledge, my thesis does not infringe upon anyone’s copyright nor violate any proprietary rights and that any ideas, techniques, quotations, or any other material from the work of other people included in my thesis, published or otherwise, are fully acknowledged in accordance with the standard referencing practices. Furthermore, to the extent that I have included copyrighted material that surpasses the bounds of fair dealing within the meaning of the Canada Copyright Act, I certify that I have obtained a written permission from the copyright owner(s) to include such material(s) in my thesis.

I declare that this is a true copy of my thesis, including any final revisions, as approved by my thesis committee and the Graduate Studies office, and that this thesis has not been submitted for a higher degree to any other University or Institution.
This thesis approaches the problem of modeling a multi-camera system’s performance from system and task parameters by describing the relationship in terms of coverage. This interface allows a substantial separation of the two concerns: the ability of the system to obtain data from the space of possible stimuli, according to task requirements, and the description of the set of stimuli required for the task. The conjecture is that for any particular system, it is in principle possible to develop such a model with ideal prediction of performance. Accordingly, a generalized structure and tool set is built around the core mathematical definitions of task-oriented coverage, without tying it to any particular model.

A family of problems related to coverage in the context of multi-camera systems is identified and described. A comprehensive survey of the state of the art in approaching such problems concludes that by coupling the representation of coverage to narrow problem cases and applications, and by attempting to simplify the models to fit optimization techniques, both the generality and the fidelity of the models are reduced. It is noted that models exhibiting practical levels of fidelity are well beyond the point where only metaheuristic optimization techniques are applicable.

Armed with these observations and a promising set of ideas from surveyed sources, a new high-fidelity model for multi-camera vision based on the general coverage framework is presented. This model is intended to be more general in scope than previous work, and despite the complexity introduced by the multiple criteria required for fidelity, it conforms to the framework and is thus tractable for certain optimization approaches. Furthermore, it is readily extended to different types of vision systems.

This thesis substantiates all of these claims. The model’s fidelity and generality is validated and compared to some of the more advanced models from the literature. Three of the aforementioned coverage problems are then approached in application cases using the model. In one case, a bistatic variant of the sensing modality is used, requiring a modification of the model; the compatibility of this modification, both conceptually and mathematically, illustrates the generality of the framework.
To my Lady Crystal,
Duncan “Bugger” Idaho & Long John Silver,
and all the cool cats who sailed the stream with me.
First and foremost, I thank Dr. Xiang Chen. In his capacity as my advisor and mentor over the past six years, his excellent training and unfailingly high standards have shaped me into the researcher I am today. In particular, his guidance throughout the work surrounding this thesis has been invaluable. I also thank the other members of my committee, Dr. Majid Ahmadi, Dr. Jonathan Wu, and Dr. Boubakeur Boufama. In addition to important training and mentoring within and beyond coursework, all provided helpful insights toward my research, and generously spent their time reviewing my work.

Numerous fellow students in the Advanced Control Systems Laboratory, both those preceding me and those following me, offered their own advice and assistance on occasion. In particular, Jose Luis Alarcon Herrera became the first other researcher to employ the results presented in this thesis in his own work, and has since contributed much to the development of Adolphus and collaborated with me on important experimental work.

I am indebted to the exceptional team at Vista Solutions for providing a great opportunity to root my work in industry for most of my doctoral studies. Mike Sirizzotti and Dean Scarlett have been excellent mentors and highly accommodating managers, and I am very grateful for their having taken a genuine interest in my career. Keith deGroot, Peter Denzinger, and Aaron Bouchard, along with the rest of the technical team, are among the most knowledgeable and skilled engineers I have had the pleasure of collaborating with, and the work presented here owes much to their insights.

I thank Dr. Frank P. Ferrie and Andrew Phan for welcoming me for a brief but inspirational visit to McGill University’s Centre for Intelligent Machines. Many other fellow researchers worldwide have provided valuable feedback on various aspects of this work, in the form of email exchanges, discussions at conferences, and paper review comments.

A special nod to my late friend Clayton Brewer, with whom I spent much of my childhood building things and taking things apart, a big part of what inspired me to go into engineering.

My deepest gratitude goes out to the friends and family who supported me in more ways than I can possibly enumerate. Three people in particular stand out. My father, Robert, who inspired my passion for science and technology from an early age, and taught me to do things the right way, whether big or small. My mother, Joan, who taught me that life is best lived with integrity and honesty, and believed in me even when I didn’t believe in myself. My wife, Crystal, who has been my best friend and companion through my entire academic career, and understands my ideas better than anyone else in the world.

Finally, I would like to acknowledge the generous financial contributions of the Natural Sciences and Engineering Research Council of Canada, Mitacs, Ontario Centres of Excellence, and Vista Solutions toward this research.
Contents

Declaration of Previous Publication iii
Abstract iv
Acknowledgements vi
List of Figures x
List of Tables xii
List of Algorithms xiii

1 Introduction 2
  1.1 Origins: Multi-Camera Systems .............................................. 2
  1.2 Motivations: Problems of Coverage ........................................... 3
  1.3 Approach: Modeling Task-Oriented Coverage ................................. 6
  1.4 Thesis Outline ........................................................................ 7

Part I Representing Coverage

2 Sensor System Coverage 10
  2.1 Overview ........................................................................... 10
  2.2 The Coverage Model ............................................................ 10
  2.3 Coverage Topology ............................................................... 15

3 State of the Art 18
  3.1 Overview ........................................................................... 18
  3.2 Geometric Models of Visual Coverage ......................................... 18
  3.3 Topological Models of Visual Overlap ......................................... 30
  3.4 Topological Models of Visual Transition ...................................... 36

4 A Visual Coverage Model 42
  4.1 Overview ........................................................................... 42
  4.2 Visual Stimulus Space ............................................................ 42
Part III Appendix

A Mathematical Background 109
   A.1 Motion of Rigid Bodies 109
   A.2 Hypergraphs 112
   A.3 Miscellaneous 112

B Geometry of Computer Vision 113
   B.1 Image Formation 113
   B.2 Optical Effects 116
   B.3 Camera Calibration 116
   B.4 Active 3D Triangulation 118

C Adolphus Simulation Environment 120
   C.1 Introduction 120
   C.2 Architecture 121
   C.3 Obtaining the Software 123

D Equipment Specifications 124
   D.1 Digital Cameras 124
   D.2 Camera Lenses 126
   D.3 Miscellaneous 127

Glossary of Terms 129

Theorems and Definitions 134
   List of Theorems 134
   List of Definitions 134

Bibliography 135

Vita Auctoris 148
# List of Figures

1.1 Task-Oriented Coverage ............................................. 7
3.1 Basic Visual Criteria .................................................. 19
3.2 View Angle .......................................................... 27
3.3 Vision Graph Example ............................................... 30
3.4 Vertex Granularity in the $CN$-Complex ......................... 34
3.5 General Transition Graph Example ............................... 37
3.6 Possible Cases of Transition ...................................... 41
4.1 Coordinate Convention of $\mathbb{D}^3$ ......................... 43
5.1 HALCON Calibration Plate ....................................... 55
5.2 Validation of Field of View Criterion ............................. 56
5.3 Validation of Resolution and Focus Criteria .................... 56
5.4 Validation of View Angle Criterion ............................... 57
5.5 Ground Truth Face Pose .......................................... 59
5.6 Layout of Comparison Surveillance Network ....................... 60
5.7 Adolphus Simulation of Comparison Surveillance Network ........... 61
5.8 Vision Graph for Comparison Surveillance Network ............... 61
5.9 Comparison of Coverage Model Fidelity .......................... 62
6.1 Jitter in View Selection ............................................ 70
6.2 Layout of Simulated Surveillance Network ..................... 71
6.3 Visualization of Simulation Environment ...................... 72
6.4 Agent Paths for Simulation Experiments ....................... 72
6.5 Varying $J$ and $r$ with Ground Truth $C_i$ and $R$ ............. 73
6.6 Vision Graph for Simulation Experiments ..................... 75
6.7 Controlled Camera Data Experiment ............................ 76
7.1 Coordinate Frames in Active Vision ............................ 80
7.2 Design Variables for Automatic Refinement ................... 86
7.3 Simulated Validation Configuration ............................. 91
7.4 Structured Target Scan ........................................... 91
7.5 Validation of Scanning Density Criterion ........................ 92
List of Tables

3.1 Comparison of Selected Visual Coverage Models ................................................. 23
3.2 Comparison of Selected Topological Coverage Overlap Models ......................... 32
3.3 Comparison of Selected Topological Transition Models ..................................... 39

4.1 Sensor Model Parameters for Visual Coverage .................................................... 44
4.2 Task Requirement Parameters for Visual Coverage ............................................. 47

5.1 Requirements for Target Pose Estimation Task ................................................... 55
5.2 Requirements for Face Detection Task ............................................................... 60

6.1 Requirements for Simulated Surveillance Task .................................................... 70
6.2 Effects of Calibration Error (Path 1, J = 10) ....................................................... 74
6.3 Effects of Calibration Error (Path 2, J = 20) ....................................................... 74
6.4 Effects of Target Tracking Error (Path 1, J = 10) ................................................. 74
6.5 Using Robot Position Data ................................................................................... 76

7.1 Task Requirement Parameters for Active 3D Vision ............................................ 82
7.2 Laser Parameters ................................................................................................. 83
7.3 Design Impact Matrix for Automatic Refinement ................................................ 87
7.4 Bounds on Design Variables .............................................................................. 88
7.5 Experimental Tasks and Results ......................................................................... 93

8.1 Hyperedges in $\mathcal{H}_C$ ..................................................................................... 99

C.1 External Dependencies for Adolphus ................................................................. 123

D.1 Prosilica EC-1350 Specifications ....................................................................... 124
D.2 NET iCube NS4133BU Specifications ............................................................... 125
D.3 SICK IVP Ranger D50 Specifications ................................................................ 125
D.4 Computar M3Z1228C-MP Specifications ......................................................... 126
D.5 Computar H10Z1218-MP Specifications ........................................................... 127
D.6 NET SV-0813V Specifications .......................................................................... 127
D.7 ASUS Eee Box EB1007-B0410 Specifications .................................................... 128
List of Algorithms

1. Real-Time View Selection with Hysteresis ............................................. 69
2. Particle Swarm Optimization ................................................................. 96
3. Minimum Maximum Indegree Orientation Heuristic ................................. 100
“That would be the highest thing for me”—so saith your lying spirit unto itself—“to gaze upon life without desire, and not like the dog, with hanging-out tongue: to be happy in gazing: with dead will, free from the grip and greed of selfishness—cold and ashy—grey all over, but with intoxicated moon-eyes! That would be the dearest thing to me”—thus doth the seduced one seduce himself,—“to love the earth as the moon loveth it, and with the eye only to feel its beauty. And this do I call immaculate perception of all things: to want nothing else from them, but to be allowed to lie before them as a mirror with a hundred facets.”

Friedrich Wilhelm Nietzsche (1844–1900), *Thus Spoke Zarathustra*
Introduction
CHAPTER 1

Introduction

If you have built castles in the air, your work need not be lost; that is where they should be. Now put the foundations under them.

Henry David Thoreau (1817–1862), Walden

1.1 Origins: Multi-Camera Systems

Computer vision is the science and technology of artificial systems that see. Cameras, like all sensors, engage in the acquisition of data; it is the endeavour of computer vision to convert this data into useful information. According to Marr [1], a vision system, as an information processing system, must be understood on three levels: the computational level, which identifies the task to which the system is set, and its purpose; the algorithmic/representational level, which describes how the system represents and acts on information to achieve the task; and the physical level, which describes the “hardware” implementing the system.

Nearly half a century ago, the field of computer vision emerged as distinct from earlier work in digital image processing in that it sought to recover information about the three-dimensional world from the raw data of images. Motivations for this research included new possibilities for automated control and metrology (cf. photogrammetry), and the obvious richness of visual information toward a human-like artificial understanding of the world.

A multi-camera system is one in which the images from multiple cameras are analyzed jointly. Such systems share theoretical properties with single-camera systems using data from multiple distinct viewpoints, which have featured in computer vision work since early on. The first true multi-camera systems arose from the use of parallax as a cue for three-dimensional shape, for which the natural approach was stereo vision, inspired by biology and having precedent in photographic stereoscopy. Extension of this approach, along with other applications combining data from multiple views, pushed the number of cameras to three, four, and more, but the practical limitations of central processing and the high cost of camera hardware kept systems relatively small.
1.2 Motivations: Problems of Coverage

In recent years, the cost of digital camera devices has dropped precipitously. Meanwhile, advances in embedded computing have spurred a trend toward increased in-network processing in sensor networks [2]. The convergence of these developments, together with progress in computer vision, have led to the emergence of distributed networks of smart cameras [3, 4] as an important new field with numerous applications. Systems tasked with establishing situation awareness [5] based on multiple sources of visual information, previously the domain of human operators, are becoming increasingly automated under this paradigm.

While the communication and processing paradigms of camera networks introduce a variety of problems not previously seen in either computer vision or sensor networks, research in this field has also exposed a class of problems arising more broadly from the multi-camera paradigm, for which there has not previously existed a need for general (multi-camera) solutions. At the most fundamental level, characterization of the performance of multi-camera systems with respect to a task is an open problem [6]; lacking a unified theoretical framework, the aforementioned problems have been addressed rather haphazardly and in isolation from one another. It is the ambition of this thesis to propose such a framework, solve the issue of performance evaluation, and provide a common basis for approaching a large class of multi-camera coverage problems.

1.2 Motivations: Problems of Coverage

In order to successfully accomplish a task using a particular set of representations of visual information and algorithms which manipulate these representations, it is tautological that the available visual data, along with any ancillary knowledge (geometric information about the imaging system, shape models, etc.), must meet some set of requirements. Identifying these requirements and establishing their interaction with the system has clear value to design, enabling quantitative evaluation and optimization.

The fundamental approach of this thesis is to describe this relationship between task and system in terms of coverage. In this model, the physical level consists of a set of stimulus, which are intrinsic to the world and ostensibly encode the information needed by the task, and the sensor system, which transforms some of the stimuli into data. The question then becomes whether the data acquired is sufficient, given the representations, algorithms, and applicable constraints from ancillary knowledge, to achieve the task.

A model predicting, to some degree of fidelity, a sensor system’s performance with respect to a task, given a priori information about both, is valuable in and of itself. For example, it may be used to evaluate a design in simulation, closing the loop without the need for costly physical implementation. However, it also provides a quantitative characterization of the objective in a family of important problems of sensor systems, which are connected in that they all involve optimization over some aspect of coverage.

In the sense that the a priori model of coverage predicts the a posteriori information available to the task, the actual acquisition of the latter can provide new information for the former, closing a feedback loop. The literature on the various coverage problems to date exhibits something of a dichotomy between offline, open-loop, model-based approaches and online, closed-loop, information-based approaches. Explicitly unifying these sources of knowledge under a common model framework
clarifies the high-level identity of these problems. It is important to note, however, that as a posteriori information is inherently incomplete and noisy, and especially since its behaviour in that respect is itself a function of coverage, the extent to which it can be integrated into a given a priori model varies.

In actuality, Marr’s tri-level hypothesis generalizes to any information processing system. Similarly, the basic theoretical framework of coverage, as presented in Chapter 2, applies to any sensor system, irrespective of the sensing modality. However, as the main scope of this thesis is limited to multi-camera systems, a set of distinct and relatively concrete coverage problems which have been identified in that context are described here.

1.2.1 View Planning

The view planning problem—also variously known as sensor planning, camera planning, and optimal camera placement in the literature—is concerned with finding an adequate or optimal view of a scene for a given task. A view, in this context, is a set of one or more camera viewpoints. Generally, the space of views is continuous, although in practice a discrete approximation may be used to fit certain optimization methods. Specific problem formulations vary depending on what is constrained; typically, the objective is either to find the camera parameters (including pose) which maximize coverage given a fixed set of cameras or maximum cost, or else to minimize the cost of the system while meeting some minimum coverage requirement.

The single-camera case is, by definition, an offline, open-loop subproblem, as the camera must be placed at a single viewpoint based on the available a priori knowledge. Consequently, this problem engendered the early model-based work, such as that of Cowan and Kovesi [7] and Tarabanis et al. [8]. The general approach is to derive, from an analysis of image formation and quantitative task requirements on predetermined scene features, an indicator function over the space of viewpoints encoding adequacy.

Cases requiring multiple viewpoints to obtain the necessary information for the task give rise to the multi-camera case. However, early work on such problems tended, partly for reasons discussed in Section 1.1, to be limited to single cameras, the objective thus becoming to find an adequate or optimal sequence of viewpoints for the camera. Since the information obtained from each viewpoint is thus available in planning subsequent viewpoints, this is naturally an online, closed-loop subproblem, commonly known as the next best view problem. The approaches of Connolly [9], Hutchinson and Kak [10], and Maver and Bajcsy [11] exemplify the information-based approach to this problem, which ultimately influenced work on autonomous exploration based on the same principles, such as that of Whaite and Ferrie [12].

With the advent of multi-camera systems, the offline view planning problem has attracted renewed attention. The existing single-camera methods model coverage over the viewpoint space, making them unwieldy for large multi-camera systems, since the dimensionality is a multiple of the number of cameras, and inapplicable to problem cases where the number of cameras is variable. Meanwhile, the models developed for next best view problems tend to be strongly—and in most cases, exclusively—oriented toward encapsulating online information as feedback. The most attractive approach initially available to camera network researchers was drawn from sensor networks, adapted by Erdem and Sclaroff [13], Hörster and Lienhart [14, 15], and others into relatively realistic two-dimensional
representations of multi-camera coverage, and eventually by Zhao et al. [16, 17, 18] and others into three dimensions.

With coverage modeled over the stimulus space rather than the viewpoint space, there is no longer an immediate analytic solution to the problem of planning the viewpoints. Instead, view planning is formulated as an optimization problem, whose difficulty depends on the tractability of the mathematical model. As multi-camera vision is a complex phenomenon, there is a tradeoff between the fidelity and tractability of the model: simple models may allow the use of powerful, efficient optimization methods, but may not yield very good solutions, while complex models can accurately identify good solutions, but may be quite difficult to optimize. Approaches to multi-camera view planning span the gamut, from convex programming on extremely simple models to metaheuristics on highly complex models.

1.2.2 View Reconfiguration

The view reconfiguration problem is a limited, online derivative of the view planning problem introduced in the previous section. Again, the objective is to find an adequate or optimal view for a given task. The two major differences are temporal considerations, including processing time and delays in realizing views, and generally tighter restrictions on the parameters which can be controlled. There is some overlap with next best view problems. In the literature, the most common target systems are networks of pan-tilt-zoom cameras, eye-in-hand robotic manipulators, and mobile robots with vision. In most specific problem formulations, the coverage objective is relatively localized (typically one or more targets, such as people) and dynamic.

The very fact that the cameras are capable of online reconfiguration is a hint that this is a closed-loop problem. Such applications as surveillance are the typical arena in which the problem arises: as new information about the dynamic scene is obtained—frequently from the cameras themselves—the optimal configuration changes accordingly.

Piciarelli et al. [19, 20] illustrate the architecture of a pan-tilt-zoom camera network which updates its model of task relevance based on activity information from the cameras. This kind of representation supports the notion that the underlying phenomenon of interest is coverage: as with the next best view problem, it is simply a matter of how closely the information can be mapped to a purely a priori model of the system.

1.2.3 View Selection

The view selection problem is a distinct coverage problem in the online domain. Here, there is no control over the cameras themselves; rather, the number of cameras from which information may be transmitted, processed, or viewed simultaneously is constrained. The instantaneous subproblem involves the selection of an adequate or optimal view of a scene for a given task from a discrete set of possible views; the overall problem is to find an adequate or optimal view sequence. As with view reconfiguration, there is some overlap with next best view problems, and the coverage objective is typically one or more localized, dynamic targets.

One hallmark of this problem is that the view sequence has some property of optimality distinct from the optimality of the individual views. This echoes certain next best view formulations, such
as that addressed by Chen and Li [21], who minimize the length of the robot path required to reach a discrete set of views. The usual motivation for seeking a view sequence in multi-camera systems is for tracking the best view of dynamic agents in the scene over time. While the desirable property of a view at any particular instant is, of course, its quality of coverage, the desirable property of the overall sequence is its smoothness, which is in general a competing objective. This is quantified to some extent in the work of Jiang et al. [22] and Daniyal et al. [23].

Since the agent dynamics constitute online information, and since the most common means of obtaining this information is from the cameras themselves, most approaches to view selection are heavily information-based. Some information models depart radically from the intuitive concept of coverage, particularly those based on abstract image entropy, but coverage remains the underlying phenomenon. Others attempt to glean more familiar structure, such as object poses and occlusions, from the scene, which map more readily to an a priori model of the system. A few approaches to view selection are explicitly based on such models, assuming (and, in some cases, supplying) a means for the system to obtain information in an appropriate form.

1.2.4 Resource Distribution

Issues of load and storage distribution arise in smart camera networks, and turn out to be strongly related to coverage. The problem is one of allocating consumption of some resource—usually processing or storage, but also possibly communication, energy, etc.—efficiently among the nodes in a camera network, given some information about the task.

The volume of activity, of the sort that consumes resources, at any given node in a camera network tends to be directly proportional to the intersection of coverage and task relevance. As these concepts are made more concrete in the chapters to follow, this will become almost self-evident: in some of the aforementioned cases, such as Piciarelli et al. [19, 20], this is true by definition. Where a set of nodes are suitable candidates for carrying out some activity which need only be performed by some subset thereof, an opportunity for optimal assignment of the activity, guided by coverage information, exists.

As with the previous two problems, the task relevance information is likely to be unknown or uncertain a priori, and this is thus often a closed-loop problem.

1.3 Approach: Modeling Task-Oriented Coverage

At the core of all of the problems presented in the previous section, and arguably quite central to the design and operation of multi-camera systems overall, is one fundamental question: how can the expected performance of a multi-camera system be quantified in terms of the parameters of the system, the environment it inhabits, and the task to which it is set?

Despite its apparent simplicity, this thesis—particularly in the survey of the state of the art in Chapter 3, and in the comparison experiments presented in Section 5.3—makes the case that no single answer to this question has yet exhibited sufficient generality to encompass the scope of problems, nor sufficient accuracy with respect to actual performance to solve them well. The major cause of this deficiency is that while many researchers have approached cases of the various coverage problems in isolation, none have attempted to generalize to the common core of the problem.
The framework presented by this thesis owes its success to the methodology behind its development. From the outset, the common issue, viz. coverage, relating several problems was identified. Without initially concentrating on any particular problem, two weaknesses of existing approaches, underlying their lack of both generality and fidelity, became clear. First, the representation of coverage tends to be strongly coupled to the prevalent applications giving rise to the specific problem; without a clear separation between the system and task, models are incompatible across problems, and even ill-fit to many applications within the same problem class. Second, the models tend to be streamlined to fit certain optimization techniques; the assumptions implied lead to further restriction of application scope and loss of fidelity.

Those efforts which do apply to a reasonable subset of problems, while retaining anything approaching a practical level of fidelity, are already beyond the point where effectively only “black-box” metaheuristic optimization techniques are applicable. At that point, additional complexity in the model only impacts the performance of optimization insofar as the time taken to evaluate the objective function increases. Therefore, this thesis assumes optimization techniques allowing an objective function of arbitrary complexity, with the exception of a bounding requirement. Under this relaxed requirement of tractability, the development of a general, high-fidelity model of coverage is possible.

The concept of a coverage model for sensor systems is presented in a highly general form—simply a bounded function over an arbitrary stimulus space—and thus any optimization approach developed for such a model generalizes trivially to any other model of the same basic form. This yields a powerful framework for approaching coverage problems, as the nature and complexity of the model may vary according to the particular application and the necessary level of fidelity. In particular, a flexible model specification for the modality of vision is presented and validated.

An important and novel feature of this framework is the explicit separation of the system and task specifications. The identification of the stimulus space for a particular sensing modality provides the interface between the two: it is possible to quantify which hypothetical stimuli in the space are covered by the system sufficiently to meet task requirements, then to specify a concrete set of relevant stimuli for the task itself, and finally identify the proportion of the latter covered by the former. This allows the same coverage model to apply to a diversity of applications and problems.

1.4 Thesis Outline

Part I of the thesis opens, in Chapter 2, with a formal description of the concept of coverage and of the
coverage model. Although actually developed by generalizing from material presented subsequently in the thesis, imparting an understanding of the core framework first allows the remainder of the thesis to be framed in its terms, which is valuable in exposing the strengths and shortcomings of existing work, and in developing the more concrete model for multi-camera networks.

Chapter 3 reviews the state of the art in approaches to the coverage problems described in Section 1.2, with a strong focus on the forms and features of the various coverage models. Along with geometric models, work on two types of topological model is surveyed.

Drawing from this prior work, a general, high-fidelity model of visual coverage is presented in Chapter 4. This model incorporates most of the identified criteria for monocular vision, and issues relating to multi-camera coverage specific to vision are discussed. While the model as presented does, of course, have some restrictions in its scope of applicability—for example, it does not characterize random occlusion or illumination, and does not address stereo vision—it covers the majority of multi-camera applications, and provides a good foundation for extension to cases outside its scope (as demonstrated in Section 7.3).

Chapter 5 provides empirical evidence of the soundness and generality of this model formulation. The individual criteria for monocular vision are validated in terms of performance prediction for a task with general properties. Then, the overall performance prediction of the model is compared to two of the more promising models from the literature, with favorable results.

Part II of the thesis demonstrates three applications to coverage problems using metaheuristic optimization techniques. In Chapter 6, a relatively simple greedy algorithm with hysteresis addresses the problem of real-time view selection. In Chapter 7, an instance of the view planning problem—in this case, for active triangulation inspection systems, requiring a modification of the base coverage model of Chapter 4—is approached with a particle swarm optimization technique. Finally, in Chapter 8, a heuristic drawn from the field of scheduling is adapted to hypergraphs, and leveraged, by way of a topological coverage model, against the general camera network load distribution problem.

Concluding remarks are presented in Chapter 9. These include a review of the contributions presented in this thesis, descriptions of a number of potential future research directions, and some final reflections on the work.

The appendices in Part III cover several ancillary but related topics. Appendix A reviews several mathematical concepts and conventions used throughout the thesis. Appendix B reviews the geometry of computer vision, including image formation and calibration. Appendix C presents an accessible and detailed description of Adolphus, the simulation environment implementing the core work of this thesis. Appendix D lists specifications for the equipment used in the various experiments.
Part I

Representing Coverage
CHAPTER 2

Sensor System Coverage

Our intelligence cannot wall itself up alive, like a pupa in a chrysalis. It must at any cost keep on speaking terms with the universe that engendered it.

William James (1842–1910), A Pluralistic Universe

2.1 Overview

In this chapter, the fundamental framework within which this thesis treats the coverage of a sensor system is described. The general form does not prescribe any particular representation for stimuli, nor any means of valuation of coverage; it is ignorant of sensing modality and model thereof. In essence, it should be seen as an abstract base class for representing the coverage of multi-sensor systems. The primary aim is to introduce the vocabulary of concepts and its interrelationships, so that they may be used with some formality in subsequent chapters.

Although this framework has been derived, in part, by generalizing from the large volume of sensor system coverage models in the literature, specific reference to these models is deferred to Chapter 3. This order of presentation not only allows for a clean, direct exposition of the framework, but also provides a common language with which to describe the surveyed works.

2.2 The Coverage Model

A sensor system is an entity which detects stimuli for the purpose of executing a task. In general, this system may physically comprise a single sensor, multiple sensors, or part of one or more sensors’ ranges, with one or more sensing modalities. Here, the term sensor refers to an atomic unit, one or more of which comprise the sensor system.

Stimuli are uniquely defined in a stimulus space $S$; in some cases, stimuli are simply 2D or 3D points, so that the stimulus space is equivalent to $\mathbb{R}^2$ or $\mathbb{R}^3$, respectively. However, with more complex sensing modalities or higher-level sensor models, characteristics of the stimulus other than its
geometric position may affect coverage as well. It may also be necessary to consider complex phenomena not localized at a single point in space. While the nature of $S$ may therefore become quite abstract, it may be helpful to imagine simple spatial stimuli for the purpose of intuitively understanding the concept of the coverage function.

A stimulus $p \in S$ is considered covered by a sensor system if it yields a response sufficient to achieve the given task. An ideal coverage function, therefore, is a bivalent mapping $C : S \rightarrow \{0, 1\}$, where $C(p) = 1$ indicates that a point in $S$ is covered. Equivalently, one may speak of the set $C \subset S$ of covered stimuli. A more general definition $C : S \rightarrow \mathbb{R}^+$ encompasses models which handle uncertainty and/or grade coverage quality. In any particular case, this may be bounded as $C : S \rightarrow [0, 1]$ without loss of generality.

1 Definition (Coverage Function)

Given a stimulus space $S$, a coverage function is a mapping $C : S \rightarrow [0, 1]$, for which $C(p)$, for any $p \in S$, is the grade of coverage at $p$, according to some definition of coverage.

Extension of the subset notion is possible if one considers $C$ a fuzzy subset [24] of $S$; that is, $C$ is the (fuzzy) set of stimuli which are covered. For convenience, the same symbol will be used to denote the coverage function and the fuzzy subset of which it constitutes the membership function. This also allows the use of the standard fuzzy union and intersection operators, which are defined, respectively, as

$$C_i \cup C_j (p) = \max(C_i(p), C_j(p)) \quad (2.1)$$

and

$$C_i \cap C_j (p) = \min(C_i(p), C_j(p)). \quad (2.2)$$

Another useful set construct is the coverage hull, the set of all points in $S$ for which $C$ is nonzero (in the fuzzy set characterization, the support of $C$).

2 Definition (Coverage Hull)

The set $(C) = \{p \in S | C(p) > 0\}$ is the coverage hull of a coverage function $C$.

Definition 2, along with (2.1) and (2.2), implies that $(C_i \cup C_j) = (C_i) \cup (C_j)$ and $(C_i \cap C_j) = (C_i) \cap (C_j)$.

A coverage model describes the valuation of $C$ over $S$ in the context of information about the complete closed system in question. A fundamental conjecture of this thesis is that for any specific sensor system, environment, and task, it is in principle possible to formulate a coverage model for $C$ which reflects perfectly the desired information for the given application.¹ As with any model of a physical phenomenon, one may trade fidelity for simplicity and generality, which have their own benefits. Accordingly, the objective is to formulate a model whose nature makes it tractable for some optimization approach for a given application class, of sufficient generality to encompass the desired application scope, yet exhibiting sufficient fidelity to effectively solve problem instances.

¹Here, application refers to the application of the coverage model to a coverage problem (such as those described in Section 1.2), as opposed to the task. From a pragmatic standpoint, the ultimate arbiter of the validity of a coverage model is, of course, its utility in solving the problem in question.
2.2.1 Generalized Model Structure

Although model formulations, even within the same sensing modality, vary widely in practice, it is possible to derive a generalized fundamental structure beyond Definition 1. It is beyond the scope of this work to demonstrate, by way of a comprehensive literature survey, that all existing coverage models for all sensing modalities are subsumed by this structure; the survey presented in Chapter 3 does, however, make this case specifically for vision.

The phenomenon of sensor coverage is, in general, dependent upon the intrinsic and spatial characteristics of the sensors themselves, the structure and properties of the environment they inhabit, and the task requirements. Thus, formally, \( C(p) \) is a shorthand for \( C(p, S, E, T) \), where \( S \), \( E \), and \( T \) are contextual parameter vectors describing the sensor system, environment, and task, respectively, which collectively define the coverage grade of \( p \in S \) according to a specific model. For a given geometric global coordinate frame and sensing modality (with a corresponding definition of \( S \)), \( S \), \( E \), and \( T \) are, in principle, independent of one another. Proper decoupling of this information in a model formulation yields important benefits.

Sensor Model

The sensor model encapsulates the characteristics of the sensing modality and the capabilities of the individual sensors. In many cases, the sensor model is homogeneous within a sensor system (with parameters varying between sensors), but as individual sensor-level coverage functions may be combined arbitrarily, this is not necessarily so, especially for systems with heterogeneous modalities.

Simplified models are defined for many sensing modalities, such that each individual sensor can be described in terms of a set of generic parameters; these are termed intrinsic parameters, and are normally either specified by the manufacturer or recovered experimentally. As the qualifier *intrinsic* suggests, this information is strictly endemic to the sensor.

The position and orientation of a sensor in \( \mathbb{R}^n \) (its pose) are described by another set of parameters, termed extrinsic parameters. In general, the formulation of a coverage function for an individual sensor is simplified by operating on stimuli within the sensor’s local coordinate frame; the pose defines a mapping for stimuli localized in the global frame to the sensor frame. In the simplest case, this is a rigid transformation in \( SE(n) \) applied to any dimensions of \( \mathbb{R}^n \) which are also dimensions of \( S \) (it is often sensible, though by no means necessary, for \( \mathbb{R}^n \) to be a subspace of \( S \)). In cases where one or more non-Euclidean dimensions of \( S \) are also dependent upon the reference frame, it is necessary to define a generalization of \( SE(n) \) including these dimensions.

Typically, a sensor will define a coverage function over \( S \) within its own local coordinate frame, based on a set of coverage criteria which define their own simpler functions over \( S \). The framework makes no prescription as to the nature of these component functions or how they are combined to obtain the sensor coverage function, but in many cases each criterion \( r \) can be expressed in the form \( C^r : S \rightarrow [0, 1] \)—in essence, a partial coverage function—and the sensor coverage function might be constructed from these functions by taking their product or minimum value,

\[
C(p) = \prod_r C^r(p)
\]

(2.3)
2.2. The Coverage Model

\[ C(p) = \min_r C^r(p) \]  \hspace{1cm} (2.4)

respectively, depending on the effects of the criteria on task performance. The objective is to define a function for the sensor which correlates strongly to the actual performance, so that the relative and absolute values of coverage are meaningful; as will be seen, it is possible to do so for complex sensing modalities with such a relatively simple scheme.

Environment Model

A description of the structure and contents of the environment is also typically necessary for an accurate expression of coverage. The use of this information in formulating \( C \) is dependent on its effect on the particular sensing modality being modeled, so the inclusion and representation of information will normally be tailored accordingly. Examples of physical phenomena which might be modeled in the environment model include:

- static objects (e.g. walls),
- deterministic dynamic objects (e.g. robots with closed control loops),
- probabilistic dynamic objects (e.g. people, vehicles),
- sources of non-task stimuli which affect sensing (e.g. ambient noise), and
- properties of the stimulus medium (e.g. temperature, density, reflectivity).

Geometric information in the environment model is described in the global coordinate frame, so as with stimuli, this information is mapped to the local frame of the individual sensor to simplify the formulation of the coverage function.

Task Model

Finally, a model of the task to be performed is required. This consists of two major components: a relevance function over the stimulus space, and a set of task requirements. The implicit task specifications found in many sources can be explicitly described in these general terms.

The relevance function indicates the relevance of the coverage of points in the stimulus space to the task. This information may be related to the environment insofar as the stimuli are associated with physical entities, but is nonetheless independent in principle. As a bivalent mapping \( R : S \rightarrow \{0, 1\} \), this specifies a subset of \( S \) which may be continuous or discrete; in the very simplest case, this may consist of a single point, implied by the context. It may also be useful to prioritize the set according to each point’s relevance to the task; in this case, \( R \) maps \( S \) to \( \mathbb{R}^+ \) which, as with the coverage function, may be bounded to \([0, 1]\) without loss of generality.

\(^{\text{Note that if all } C^r \text{ are bivalent, that is, if } C^r : S \rightarrow \{0, 1\}, \text{ then (2.3) and (2.4) are equivalent.}}\)
3 Definition (Relevance Function)

Given a stimulus space $S$, a relevance function is a mapping $R : S \rightarrow [0, 1]$, for which $R(p)$, for any $p \in S$, is the relevance of the coverage of $p$ to the task with which it is associated.

It is convenient (and trivial) to extend to relevance functions the fuzzy set conceptualization, the union and intersection operations of (2.1) and (2.2), and the notion of the coverage hull from Definition 2. For the last, a more context-appropriate term for $(R)$ is the task point set.

While there is no formal difference between them, two distinct conceptual interpretations of the relevance function exist, depending on the nature of the application:

- **Concrete relevance function**: The objective is to cover a particular stimulus or set thereof, usually associated with one or more actual target objects; relatively localized in the stimulus space.

- **Abstract relevance function**: The objective is to cover a field of potential stimuli over a range; relatively dispersed in the stimulus space.

The task requirements are a set of prescriptions on the data to be obtained by the sensor system from the stimuli in $R$. These need not be requirements in the strict sense; they are arbitrary, sensor-independent desirable properties of the data, which the coverage function will treat according to the sensing model, generally affecting the valuation of coverage in terms of whether (and to what extent) the data, as modeled, exhibit these properties.

### 2.2.2 Multi-Sensor Coverage

It is often helpful to construct a coverage function by mathematically relating a set of “lower-level” coverage functions. The obvious case is the coverage function for a sensor system being composed of coverage functions for the individual sensors, though there could conceivably be an arbitrary number of levels in such a hierarchy. The definition of sensor system and sensor are general enough to state that $C(p)$ for a sensor system is some arbitrary function of $C_i(p)$ for each sensor $i$ mapping to $[0, 1]$, and that a sensor system at the $n$th level of the hierarchy is a sensor at the $(n + 1)$th level. The coverage model defines $C$ for the topmost level entirely, recursively including all lower levels. The levels referred to as sensor system and sensor will normally be clear from the context.

Although the sensor system’s coverage function is arbitrary in terms of the sensor’s functions, a useful definition of the coverage of the sensor system is often simply the combined coverage of the individual sensors, as given by

$$C_N^1(p) = \max_{i \in N} C_i(P_i^{-1}(p)).$$

(2.5)

where $N$ is the set of sensors in the sensor system, and $P_i$ is the transformation from the local sensor coordinate system in which $C_i$ is expressed to the global coordinate frame (normally equivalent to the pose of sensor $i$). If the $C_i$ functions are bivalent, $C_N^1$ indicates which stimuli in $S$ are covered by at least one sensor; in other words, $C_N^1$ defines the union of $C_i$ for all $i \in N$. As is clear from (2.5), this notion extends to the standard fuzzy union [24] for real-valued $C_i$. 

---

14
2.3. Coverage Topology

The above form defines 1-coverage, and is a special case of the more general k-coverage, for \( k \geq 1 \), which indicates whether a stimulus in \( \mathcal{S} \) is covered by at least \( k \) sensors. With real-valued \( C_i \), there is no single valid expression of the coverage of a point by a group of \( k \) sensors, but the standard form follows from the standard fuzzy intersection \([24]\) as

\[
C_M(p) = \min_{i \in M} C_i(P_i^{-1}(p))
\]

where \( |M| = k \). The \( k \)-coverage function for the sensor system, then, is the union of (2.6) for all \( k \)-combinations in \( N \), expressed as

\[
C_N^k(p) = \max_{M \in \binom{N}{k}} C_M(p)
\]

for which \( C_N^1 \) clearly reduces to the form stated in (2.5).

2.2.3 Task-Oriented Coverage Evaluation

A coverage function for the sensor system as a whole quantifies the coverage of a single point in \( \mathcal{S} \), according to the task requirements. As discussed in Section 2.2.1, the task also specifies a relevance function, which induces a task point set \( (R) \) that, in general, contains multiple points. The final component of the coverage model is the specification of a bounded scalar coverage metric for the entire task.

Given a sensor system coverage function \( C \) for a task with relevance function \( R \) inducing a finite, discrete task point set \( (R) \), the coverage performance of the sensor system with respect to the task is

\[
F(C, R) = \frac{\sum_{p \in (R)} C(p) R(p)}{\sum_{p \in (R)} R(p)}.
\]

It is not generally feasible to compute \( F(C, R) \) as an integral for continuous \( (R) \), due to the arbitrary complexity of \( C \) and \( R \). In this case, (2.8) may be evaluated to any precision by sampling \( \mathcal{S} \) as a grid of discrete points, and thereby obtaining a discrete \( (R) \).

2.3 Coverage Topology

With the division of a sensor system into sensors, various topological relationships among the sensors related to their coverage can be represented. In this abstracted form, combinatorial analysis can be applied for certain purposes. These topological models are essentially derivative of the information encapsulated by the coverage model and task, but may involve additional related considerations.

Two such models particularly useful in approaching sensor system coverage problems represent, respectively, the coverage overlap and the transition of dynamic targets. This section describes a generic form for each type of model, and relates them to the coverage model framework of Section 2.2.
2.3. Coverage Topology

2.3.1 The Coverage Overlap Hypergraph

An overlap model describes the topology of a sensor system in terms of mutual coverage of some subset of $S$ by the sensors, with respect to a relevance function (whether implicit or explicit). It is often desirable to capture not only the fact, but also the degree, of overlap.

The general form is a weighted undirected hypergraph $\mathcal{H} = (N, E, w)$, where the vertex set $N$ is the set of sensors in the sensor system, $E \subseteq \mathcal{P}(N)$ (where $\mathcal{P}$ denotes the power set) is a set of hyperedges, and $w : E \to \mathbb{R}^+$ is a weight function over $E$. The existence of a hyperedge $e \in E$ indicates that the sensors in $e$ share mutual coverage of the stimulus space, with a $k$-hyperedge corresponding to $k$-coverage. The hyperedge weight $w(e)$ quantifies the degree of shared coverage among the sensors in $e$.

The definition of mutual coverage and the characterization of edges can vary widely depending on the available and desired information in a particular application. However, it is possible to define an explicit general form derived from the coverage function, which all related models approximate in whole or in part; this form is an ideal model of coverage overlap with respect to a task to the extent that the coverage function and relevance function ideally quantify coverage.

The coverage hypergraph of a sensor system comprising a set of sensors $N$ is the hypergraph $\mathcal{H}_C = (N, E_C, w)$. Its hyperedge set is defined as

$$E_C = \{M \in \mathcal{P}(N) | (C_M \cap R) \neq \emptyset\}$$  \hspace{1cm} (2.9)

where $C_M$ is computed by (2.6) for a given task, $R$ is the relevance function of the task, and $\mathcal{P}$ denotes the power set. Intuitively, $M \in E_C$ indicates that sensors $M$ have mutual coverage of some part of $S$ with respect to $R$.

1 Theorem ($E_C$ is an Abstract Simplicial Complex)

The hyperedge set $E_C$ of a coverage hypergraph is an abstract simplicial complex; that is, for every $M \in E_C$, and every $L \subseteq M, L \in E_C$.

Proof If $n \in M$, then by (2.6), $C_M = C_{M \setminus n} \cap n$. From (2.2), for all $p \in S$, $C_M(p) \leq C_{M \setminus n}(p)$. Then, from Definition 2, clearly $(C_M) \subseteq (C_{M \setminus n})$, and $(C_M \cap R) \subseteq (C_{M \setminus n} \cap R)$. Thus, for every $M \in E_C$, and every $M \setminus n \subset M, M \setminus n \in E_C$.

In practice, pairwise overlap is by far the most commonly sought topological information. The 2-uniform subgraph of $\mathcal{H}_C$—by analogy with the $k$-coverage notation in (2.7), denoted $\mathcal{H}_C^2$—is a (weighted) graph encapsulating pairwise overlap. Since, by Theorem 1, $E_C$ is an abstract simplicial complex, $\mathcal{H}_C^2$ is isomorphic to the primal graph of $H_C$.

2.3.2 The Transition Graph

A transition model describes the topology of a sensor system in terms of the probability and/or timing of dynamic agents transitioning from one region of coverage to another. While an overlap model

\footnote{Hypergraphs and their properties are reviewed in Section A.2.}
2.3. Coverage Topology

captures a physical topology, a transition model captures a more abstract functional topology of agent activity.

In the most general form, such a model is a weighted directed graph \( G = (N, A, w) \), where \( N \) is the set of sensors in the sensor system (noting that it may be advantageous, in this case, for “sensors” to represent subdivisions of the actual physical sensors), \( A \) is a set of arcs, and \( w : A \rightarrow \mathbb{R}^+ \) is a weight function over \( A \). \( N \) may also include a special source/sink node to collectively represent the uncovered portions of the stimulus space. The existence of an arc \( a \in A \) indicates that agents may transition from the tail region to the head region. In a weighted model, \( w(a) \) is a quantitative metric encapsulating the probability and/or duration of the transition.
3.1 Overview

A comprehensive literature survey of geometric and topological coverage models for multi-camera systems is presented. The models are analyzed and compared in the context of their intended applications. The general model form presented in Chapter 2, as well as the visual coverage model of Chapter 4, derive, in part, from the various properties and features of these models.

It is assumed that the reader is familiar with the fundamentals of computer vision, including perspective projection, basic optics, and camera calibration. Appendix B reviews these topics, and establishes the associated terminology, notation, and conventions employed in this and following chapters.

3.2 Geometric Models of Visual Coverage

The first type of coverage model surveyed is that which attempts to provide a coverage function valuation over physical visual stimuli—hence the “geometric” appellation—which corresponds to the form presented in Section 2.2.

3.2.1 Anatomy of a Visual Coverage Model

Visual stimuli considered in the work reviewed herein can be reduced to point features [7, 8], which have a single point of origin in Euclidean space and possibly other characteristics. Vision is an inherently three-dimensional sensing modality, though it is frequently modeled in two dimensions for simplicity. Thus, in all cases, the stimulus space is (or is a superset of) $\mathbb{R}^2$ or $\mathbb{R}^3$, lending a relatively concrete sense to discussion of the coverage of points.
A clear distinction between the environment and task models, as described in Section 2.2.1, is not always made in the cited sources, as there is often strong interaction between the two sets of information. For example, the relevance function is implied by a bounded environment structure in several of the sources reviewed here. Nevertheless, they will be treated independently for the sake of consistency.

**Sensor Model: Coverage Criteria for Vision**

Based on well-studied geometric imaging models (as described in Appendix B), a number of criteria for monocular visual coverage have been identified in the literature, and some or all are incorporated into various coverage models.

The first three criteria [25] depend only upon the viewpoint and a point feature in $\mathbb{R}^3$ (two-dimensional coverage models can be thought of as projecting these criteria onto the plane).

- **Field of view**: The bounds on the infinite subspace of $\mathbb{R}^3$ which can theoretically be imaged by the camera: a quadrilateral pyramid determined by the horizontal and vertical apex angles (in turn, by the optics and physical image sensor size) and the pose of the camera.

- **Resolution**: An upper and/or (less commonly) lower bound on the length projecting onto a single pixel in the image; translates directly into upper and lower limits on depth along the optical axis.

- **Focus**: A constraint on the acceptable sharpness of the image; given a maximum blur circle diameter, imposes upper and lower limits on depth along the optical axis about the subject distance (the depth of field).

Combining these criteria—i.e., truncating the field of view by the depth constraints of resolution and/or focus—produces a frustum, termed the viewing frustum after the analogous concept in computer graphics.

**Figure 3.1**: Basic Visual Criteria – The three basic visual criteria (field of view, resolution, and focus) are entirely defined by the location of the stimulus in space, and collectively define the viewing frustum.

Considering the view angle to the feature, the direction of the surface normal point feature with respect to the camera, adds a fourth possible coverage criterion.
3.2. Geometric Models of Visual Coverage

- **View angle**: A constraint on the maximum angle of the surface normal with respect to either the optical axis or the ray joining the optical center with the point.

A point feature may also be occluded, and thus not covered, if the ray from the feature to the optical center of the camera is interrupted by an opaque physical object. Occlusion is modeled in two distinct ways, depending on the type of information about the scene available.

- **Deterministic occlusion**: A bivalent criterion depending on whether the ray is interrupted by static objects or objects with known dynamics (e.g. walls).

- **Random occlusion**: A criterion depending on the probability that the ray to the point is interrupted by stochastic objects (e.g. humans).

In theory, deterministic occlusion could be treated as a special case of random occlusion with occupancy probabilities in \{0, 1\}, but in practice the information is modeled sufficiently differently to warrant the distinction; in fact, all surveyed models with a random occlusion criterion also consider deterministic occlusion separately. Self-occlusion is typically handled by a combination of the above and an upper bound of $\pi/2$ on the maximum view angle.

**Environment Model**

A nearly universal feature of environment models for visual coverage is some representation of static occluding structures, especially walls. Originating with the classic art gallery problem [26], the two-dimensional representation is often in the form of one or more polygons; this generalizes to polyhedrons in three-dimensions. Naturally, these are used in computing the deterministic occlusion criterion.

In some cases, a probabilistic model of occupancy and/or agent dynamics is also provided, allowing for the computation of the random occlusion criterion. This model may be informed in part by the static scene model, if one is available, in e.g. the imposition of constraints on agent motion.

**Task Model**

The relevance function $R$ takes the general form of Definition 3. As $S \supseteq \mathbb{R}^N$, task points in \langle R \rangle are located at physical points in the environment. Most commonly, $R$ represents a relatively large volume of the environment to be observed, or some localized feature or object to be inspected or tracked. Frequently, $R$ is not specified explicitly: rather, it is implied by the structure of the static environment model and/or the probability distributions of agent dynamics.

A recurring motif in the literature is that the quantification of visual coverage depends as much on the task as it does on the imaging system. Generally, given a computer vision algorithm used in a task, it is possible, at least in principle, to quantify soft or hard requirements on the criteria described in the sensor model. These typically include minimum and maximum resolution, maximum acceptable blur circle diameter (or equivalent focus criterion), and maximum acceptable view angle, depending on whether these criteria are observed by the model.
3.2.2 Geometric Coverage Models by Application

**View Planning**

The single-camera view planning problem received a significant amount of attention in the late 1980s and 1990s, prior to the advent of multi-camera networks. With the problem so posed, the objective is to find a viewpoint which adequately covers the target feature or features—which, in general, may be defined by a relevance function—according to a set of task requirements. The output of such methods is in the form of a (possibly empty) set of suitable viewpoints. It is generally straightforward to invert the criteria to obtain the coverage function over $S$ for any particular viewpoint. Tarabanis et al. [27] present an excellent survey of the earlier work on this topic. Typically, the target systems employ a single camera observing a relatively well-controlled scene, and both require and can afford high-fidelity coverage models. The work of Cowan and Kovesi [7] and Tarabanis et al. [8] are quintessential examples from this period.

This exact approach does not scale well to systems with more than a few cameras at most, and the multi-camera context introduces additional design variables, including, of course, the number of cameras (possibly with a competing cost objective). Nonlinear optimization techniques and metaheuristics are the tools of choice, encouraging the use of much simpler coverage models. Typically, the objective is to search for either the solution with maximum coverage given a fixed cost or number of cameras, or the solution with minimum cost or number of cameras yielding some minimum coverage.

A basic formulation is equivalent to the classic art gallery problem [26]; González-Banos and Latombe [28] frame it so, with their model assuming omnidirectional visibility and infinite range. Significantly higher fidelity can be achieved simply by limiting visibility and range, but this fundamentally changes the problem. Drawing on the sensor network literature, Ma and Liu [29, 30] propose a so-called boolean sector coverage model (derived from the common 2D disc model [31]), enabling them to treat view planning as a set cover problem [32, 33]. Qian and Qi [34], Wang et al. [35], and Jiang et al. [36] further develop this direction. Erdem and Sclaroff [37, 13] approach the problem with a more realistic two-dimensional model; subsequent results using different coverage model and optimization techniques but an overall similar method have been reported by Hörster and Lienhart [14, 15], Angella et al. [38], and Zhao et al. [16, 17, 18]. Malik and Bajcsy [39] address view planning for stereo camera nodes similarly. Yao et al. [40] adapt this type of approach to surveillance networks with tracking and handoff tasks, adding a “safety margin” to their coverage model to enforce the necessary coverage overlap. The work of Mittal and Davis [41, 42, 43] extends the set of constraints to include random occlusion, important in a significant subset of applications involving relatively high densities of dynamic agents.

**View Reconfiguration**

Coverage models and optimization techniques used in approaching view reconfiguration problems reflect the need for real-time online performance. Bodor et al. [44, 45] and Fiore et al. [46] seek to optimize the configuration of cameras mounted on mobile robots for global scene coverage. Piciarelli et al. [19, 20] address reconfiguration of pan-tilt-zoom (PTZ) cameras, common in surveillance applications. Ram et al. [47] and Erdem and Sclaroff [13] both also touch on PTZ reconfiguration; the latter do so by introducing a time constraint to the view planning. Chen et al. [48] focus on the view angle...
criterion in optimizing the configuration of rotating (panning) cameras.

**View Selection**

With a smaller solution space and usually more loosely defined task requirements, coverage model used in view selection focus on high-fidelity, usually graded quantization of coverage. Reed and Allen [49] and Chen and Li [21] approach the related next best view problem using coverage model similar to those used in view planning. Park et al. [50] use a relatively simplistic three-dimensional coverage model for view selection, acknowledging that a more sophisticated model could be substituted. The approach of Shen et al. [51] is notable for assigning a scalar coverage metric to the stimulus space and for allowing task-specific weighting of the individual criteria; they also touch on a version of the view planning problem. Soro and Heinzelman [52] approach a slightly different problem: given a desired viewpoint directly—as opposed to a coverage objective in the form of a relevance function—their algorithm attempts to find the closest actual available viewpoint, subject to energy costs.

**Load and Storage Distribution**

For completeness, it is worth mentioning the geometric component of the topological coverage overlap model of Kulkarni et al. [53], which differs from other models surveyed here in that it is not analytically derived from a camera model. Instead, it is purely empirical: through a Monte Carlo process whereby a structured target is placed at an arbitrary number of random points in the scene, each camera with a view to the target at a given position estimates its pose, and each Voronoï cell [54] around a target position forms a part of the geometric coverage of each camera that observed that position. In combination with the topological model, it is applied to load scheduling problems.

**3.2.3 Analysis and Comparison of Models**

Table 3.1 compares the nature and properties of a number of camera network coverage models from the literature, grouped by application. Since most of these models have been developed with specific applications in mind (indicated in the first column), it should be interpreted as a comment on the generality, and not necessarily the validity or quality, of the models. The second column indicates the dimensionality of the model; a dimensionality of 2.5 indicates that the final representation is two-dimensional, but is derived from three-dimensional characteristics of the sensor system and environment. The third column indicates whether the coverage function is graded, i.e. whether it assigns to a point a scalar measure of coverage in some form (weighted, probabilistic, fuzzy, etc.); non-graded functions are bivalent. The following four columns indicate which of the imaging coverage criteria—field of view, resolution, focus, and view angle—are observed. The final two columns indicate which type of occlusion models—deterministic and/or random—are used. It should be noted that, in some cases, the authors do not provide quantitative descriptions of some criteria or means of obtaining the information required to derive them.
### Table 3.1: Comparison of Selected Visual Coverage Models

<table>
<thead>
<tr>
<th>Model</th>
<th>App.</th>
<th>Properties</th>
<th>Imaging Criteria</th>
<th>Occlusion</th>
</tr>
</thead>
<tbody>
<tr>
<td>Cowan and Kovesi [7]</td>
<td>VP</td>
<td>● ● ● ●</td>
<td>● ● ● ●</td>
<td>● ● ● ●</td>
</tr>
<tr>
<td>Tarabanis et al. [8]</td>
<td>VP</td>
<td>● ● ● !</td>
<td>● ● ● ●</td>
<td>● ● ● ●</td>
</tr>
<tr>
<td>González-Banos and Latombe [28]</td>
<td>VP</td>
<td>● ●</td>
<td>● ● ● ●</td>
<td>● ● ● ●</td>
</tr>
<tr>
<td>Wang et al. [35]</td>
<td>VP</td>
<td>● ●</td>
<td>● ● ● ●</td>
<td>● ● ● ●</td>
</tr>
<tr>
<td>Jiang et al. [36]</td>
<td>VP</td>
<td>● ●</td>
<td>● ● ● ●</td>
<td>● ● ● ●</td>
</tr>
<tr>
<td>Erdem and Sclaroff [13]</td>
<td>VP</td>
<td>● ●</td>
<td>● ● ● ●</td>
<td>● ● ● ●</td>
</tr>
<tr>
<td>Hörster and Lienhart [15]</td>
<td>VP</td>
<td>● ●</td>
<td>● ● ● ●</td>
<td>● ● ● ●</td>
</tr>
<tr>
<td>Angella et al. [38]</td>
<td>VP</td>
<td>● ● ● ●</td>
<td>● ● ● ●</td>
<td>● ● ● ●</td>
</tr>
<tr>
<td>Zhao et al. [17]</td>
<td>VP</td>
<td>● ● ● ●</td>
<td>● ● ● ●</td>
<td>● ● ● ●</td>
</tr>
<tr>
<td>Malik and Bajcsy [39]</td>
<td>VP</td>
<td>● ● ● ●</td>
<td>● ● ● ●</td>
<td>● ● ● ●</td>
</tr>
<tr>
<td>Mittal and Davis [43]</td>
<td>VP</td>
<td>● ● ● ●</td>
<td>● ● ● ●</td>
<td>● ● ● ●</td>
</tr>
<tr>
<td>Bodor et al. [45]</td>
<td>VR</td>
<td>● ● ● ●</td>
<td>● ● ● ●</td>
<td>● ● ● ●</td>
</tr>
<tr>
<td>Piciarelli et al. [20]</td>
<td>VR</td>
<td>● ● ● ●</td>
<td>● ● ● ●</td>
<td>● ● ● ●</td>
</tr>
<tr>
<td>Park et al. [50]</td>
<td>VS</td>
<td>● ● ● ●</td>
<td>● ● ● ●</td>
<td>● ● ● ●</td>
</tr>
<tr>
<td>Shen et al. [51]</td>
<td>VS</td>
<td>● ● ● ●</td>
<td>● ● ● ●</td>
<td>● ● ● ●</td>
</tr>
</tbody>
</table>

---

### Dimensionality of the Stimulus Space

Although vision is an inherently three-dimensional phenomenon, many coverage models in the literature are two-dimensional. In such cases, to simplify the problem at hand, it is assumed (either implicitly or explicitly) that

- all cameras are positioned in a common plane,
- all targets are constrained to a common plane, and
- the scene consists of occluding vertical “high walls.”

In models derived from the art gallery problem formulation, e.g. González-Banos and Latombe [28], the choice reflects the fact that three-dimensional generalization of the problem is NP-hard [55]. The vast majority of work on sensor network coverage problems [56] has employed two-dimensional disc models [31] (although the three-dimensional case has been studied [57]), assuming a roughly planar environment. Some camera network models, including those of Ma and Liu [29, 30, 33], Wang et al. [35], and Jiang et al. [36], follow directly from this tradition, simply restricting the disc to a sector [31] for directionality. Erdem and Sclaroff [13] and Hörster and Lienhart [15] do not appear to share this lineage, and explicitly cite the complexity of their respective optimization methods as motivating their restriction to two dimensions. The model of Yao et al. [40] appears to be heavily influenced by that of Erdem and Sclaroff. In all of the preceding cases, the domain of camera coverage is explicitly planar.

In contrast, some two-dimensional models are not developed from the ground up as such. Bodor et al. [44, 45] and Mittal and Davis [43] begin with three-dimensional analytic treatments of their respec-
tive constraints, but subsequent assumptions about the scene and viewpoint restrictions effectively reduce their models to the plane without loss of information. Shen et al. [51] present a similar treatment of view angle—in particular, including the inclination angle between the sensor and a human subject’s head with respect to the ground plane—in an otherwise two-dimensional model. Piciarelli et al. [20] account for a three-dimensional field of view criterion by projecting the elliptical cross-section of their conical visible region onto the plane.

Early coverage models used in single-camera view planning, such as those of Cowan and Kovesi [7] and Tarabanis et al. [8], are fully three-dimensional: the gains in generality and fidelity clearly outweigh the added complexity in the single-camera case. These advantages have induced a number of multi-camera coverage models across the application spectrum to follow suit. Cerfontaine et al. [58] describe a multi-camera method employing a three-dimensional coverage model presumably derived from the pinhole camera model, but give no details on the criteria. Park et al. [50] fully describe their model with a three-dimensional viewing frustum; the multi-camera complexity is handled by dividing the covered volume into discrete parts and generating look-up tables for the graded coverage function. Angella et al. [38] employ a three-dimensional model drawing heavily on the single-camera view planning literature. The model of Malik and Bajcsy [39] is also fully three-dimensional. While Zhao et al. [17] use a three-dimensional model and do not restrict camera pose, targets—and therefore, the relevance function—are constrained to lie in a plane.

**Coverage Function Valuation**

View planning applications typically have well-defined task requirements, and the objective is simply to find any viewpoint which satisfies these requirements. Accordingly, models such as that of Cowan and Kovesi [7] are bivalent: either the viewpoint is adequate or it is not. Tarabanis et al. [8] discuss not only the adequacy of a viewpoint, but also its optimality, proposing an overall coverage quality metric based on the robustness (e.g. with respect to positioning error) in individual criteria.

In solving the view selection problem, one is interested in finding the best view of a relevance function, to which a real-valued coverage function clearly lends itself. In Park et al. [50], the quality of coverage of a point \( \mathbf{p} \in \mathbb{R}^3 \) from a camera is considered to vary inversely with the distance from \( \mathbf{p} \) to the center of the viewing frustum. The authors point out that developing a high fidelity coverage metric is not their focus, and allow that a more sophisticated definition could be substituted for their own. Shen et al. [51] explicitly set out to define such a metric for the restricted problem case of human surveillance; theirs takes the form of a real-valued coverage function.

Soro and Heinzelman [52] study several coverage-based valuations of viewpoints for view selection, but as previously mentioned, their formulation is notably different than others discussed here. Roughly speaking, each valuation can be thought of as a distance metric on the space of admissible viewpoints. Were one to assign an ideal viewpoint to every \( \mathbf{p} \in \mathcal{S} \), these metrics would effectively constitute a coverage function per Definition 1.

By contrast, in solving the multi-camera view planning and view reconfiguration problems, bivalent coverage functions are used almost exclusively, to enable the use of certain optimization techniques (e.g. binary integer programming) that otherwise would not apply. Wang et al. [35] provide one counterexample, applying a multi-agent genetic algorithm over a graded coverage function simple enough to make the optimization computationally feasible. Continuous coverage functions defined
3.2. Geometric Models of Visual Coverage

by Yao et al. [40] assign reduced values to regions near the limits of field of view and resolution, in order to encourage their optimization process to yield solutions with a substantial margin of overlap between cameras for improved tracking and handoff. Notably, Shen et al. [51] use their graded coverage function as a constraint in solving a restricted case of the view planning problem using a greedy algorithm.

Field of View

The coverage model employed by González-Banos and Latombe [28] is unique among those surveyed in assuming omnidirectional viewing capabilities, and thus not observing a field of view criterion. The directional nature of camera coverage is a recurring key point in the literature, and field of view is the most commonly modeled constraint. The simple sector-based models of Ma and Liu [29, 30, 33], Qian and Qi [34], Wang et al. [35], and Jiang et al. [36] describe the field of view with a single angle parameter, which corresponds roughly to the horizontal apex angle. The boundary rays are symmetric about the optical axis, implying an assumption of non-oblique projection (see Section B.1.2). Otherwise, this turns out to be a satisfactory definition in two dimensions; Erdem and Sclaroff [13] and Hörster and Lienhart [15] arrive at the same by way of the pinhole camera model, perhaps elucidating how its value should be determined from the configuration of a given camera.

Erdem and Sclaroff [37] also describe the three-dimensional field of view using two apex angles. Malik and Bajcsy [39] and Mittal and Davis [43] handle the field of view similarly.

Cowan and Kovesi [7] and Tarabanis et al. [8] both effectively limit the field of view to the smaller of the two apex angles, and assume non-oblique projection. Piciarelli et al. [20] model the field of view as a cone, presumably with aperture angle equal to the smaller apex angle. While this representation facilitates their algorithm by projecting to a circle of constant radius on a transformation of the scene plane, it lacks fidelity and no justification is given in the context of their application.

The apex angles are derived from a more elementary characterization of the field of view. By definition, a point \( p \in \mathbb{R}^3 \) is within the field of view of a camera if its projection lies somewhere on the physical sensor surface. Zhao et al. [17] use this constraint directly, and can therefore, in theory, handle oblique projection. The field of view can also be thought of as an infinite set of rectangles similar to the sensor surface and orthogonal to the optical axis; Park et al. [50] simply assume that the dimensions of those at the near and far depth of field limits—in other words, the parallel faces of the viewing frustum—are known, and that they are centered at the optical axis (implying non-oblique projection).

Resolution

The sector-based models proposed by Ma and Liu [29, 30, 33], Wang et al. [35], and Jiang et al. [36] have a radial range limit; although there is no explicit relationship to a resolution criterion, it seems its most likely justification. Cowan and Kovesi [7] model their resolution criterion as an arc in two dimensions and as a spherical cap in three dimensions.

In fact, this circular/spherical representation unnecessarily complicates the matter: since the projected image is planar and orthogonal to the optical axis, resolution is a function of depth along the optical axis rather than distance along the ray from the optical center [25]. The triangle-shaped model
of Hörster and Lienhart [15] is a more accurate two-dimensional representation of the resolution criterion, although it is not explicitly parameterized as such. Erdem and Sclaroff [13], Bodor et al. [45], Malik and Bajcsy [39], Yao et al. [40], and Mittal and Davis [43] all use distance along the optical axis as the single variable for the resolution criterion. The last also suggest that such a resolution criterion could be used as a “soft” constraint informing a graded quality measure.

Zhao et al. [17] model resolution in combination with view angle, for their specific application, as the projected length of a target (name tag), to which they apply a threshold.

Focus

While focus is a staple constraint in coverage models for single-camera view planning [27], it is not observed by most models developed for other purposes. Angella et al. [38] mention it, but as with their other imaging criteria, they provide no details. Park et al. [50] are the other exception; their model is bounded in depth along the optical axis by the near and far depth of field limits.

Park et al. also use focus as part of their coverage function valuation, to some extent: if the center of the viewing frustum is taken as an approximation of the subject distance, the distance of a point along the optical axis from the center varies approximately proportionally to the blur circle diameter. A similar interpretation can be applied to the coverage function valuation of Wang et al. [35].

View Angle

A view angle criterion is observed by some coverage models for single-camera view planning [27], including that of Cowan and Kovesi [7]. In the multi-camera context, it is observed where the target task clearly depends on view angle. For example, the application task addressed by Zhao et al. [17] is the identification of planar tags, the performance of which degrades with increasing view angle. Similarly, Shen et al. [51] are interested in surveillance tasks such as face tracking, so view angle features prominently in their model. Mittal and Davis [43], drawing on the earlier single-camera view planning models, include the criterion for the sake of generality, anticipating that some tasks will have such a requirement.

Special cases of task constraints on view angle give rise to a few alternate, yet equivalent, forms of the criterion. Bodor et al. [45] are interested in observing paths, where foreshortening effects due to the view angle to a path degrade performance; their criterion is based on both the angle between the path normal and the camera position, and the angle between the path center and the optical axis. Some applications, such as those of Malik and Bajcsy [39] and Chow et al. [59], require 360° coverage of a target, and define a maximum view angle for mutual coverage of a point by two cameras. If the view angle to a feature on an opaque surface exceeds 90°, the surface occludes the feature from view; this phenomenon is known as self-occlusion, and is sometimes treated as a separate criterion, such as by Chen and Li [21] and Zhao et al. [17]. In the latter case, the view angle criterion is used for self-occlusion; recall that the authors consider view angle along with resolution by computing the projected tag length.

An interesting question that arises in defining this criterion is whether to measure the view angle between the surface normal (at the point feature) and the optical axis, or between the surface normal and the ray from the camera’s optical center to the point feature. Both approaches have merit in terms
of validity with respect to task requirements. The former is taken by Chen and Li [21] and Bodor et al. [45]; the latter, by Cowan and Kovesi [7], Shen et al. [51], Malik and Bajcsy [39], and Zhao et al. [17].

Soro and Heinzelman [52], in one of their models, base the value of the coverage function primarily on view angle.

**Deterministic Occlusion**

Occlusion by static scene objects factors heavily in most multi-camera coverage work. Malik and Bajcsy [39], whose model does not include a static occlusion criterion, assume a simple rectangular room with \( (R) \) somewhere near its center, which suits their target task, but in most multi-camera applications the scene is allowed to be more complex. The polygonal “high wall” occlusion model common in two-dimensional approaches has its origin in the art gallery problem, exemplified by González-Banos and Latombe [28]. This constraint is enforced as follows: given a scene model consisting of line segments in the plane, a point \( p \in \mathbb{R}^2 \) is occluded (not covered) if the ray from the camera’s optical center to \( p \) intersects any such line segment. Erdem and Sclaroff [13] propose an algorithm to construct a continuous “visibility polygon” set which contains all non-occluded scene points. Hörster and Lienhart [15], Mittal and Davis [43], and Shen et al. [51] check for ray intersection on a discrete \( (R) \). Jianget al. [36] approximate occlusion by simply excluding obstacle regions from the field of view of a camera; in confined spaces and using cameras with realistic field of view, this approach exhibits low fidelity.

The three-dimensional analog to the line segment scene model is composed of opaque surfaces; without loss of generality, triangles. A continuous, analytic solution analogous to that of Erdem and Sclaroff [13] for the three-dimensional case is presented by Tarabanis et al. [60]. Maver and Bajcsy [11] compute a similar structure in their approach to the next best view problem. In the multi-camera context, checking for ray intersection on a discrete \( (R) \) is more common, as is done by Angella et al. [38] and Zhao et al. [17].

Piciarelli et al. [20] handle static occlusion directly in the relevance function. Each camera node has its own copy of the global relevance function, in which the values for all occluded points (determined via two-dimensional line of sight) are set to 0.
Random Occlusion

Mittal and Davis [43] have pioneered the handling of random occlusion in a visual coverage model. They use a probabilistic model of agent occupancy and some assumptions about agent height and allowable camera viewpoints to formulate a probabilistic visibility criterion, which is then integrated with their other constraints (including deterministic occlusion). Angella et al. [38] use this model. Chen and Davis [61] independently propose their own probabilistic model for random occlusion, under similar assumptions about the agents and cameras. Qian and Qi [34] also propose a probabilistic model, with agents modeled as two-dimensional discs (analogous to Mittal and Davis’ representation) and using a simple sector-type coverage model.

Zhao et al. [17] include a “mutual occlusion” criterion in their model, which approximates worst-case random occlusion by specifying a range of view angles within which a point is assumed to be occluded by another agent.

Combining Criteria and Multi-Camera Coverage

Cowan and Kovesi [7] treat coverage criteria as constraints on the viewpoint, so in order to find the solution set which satisfies all constraints (i.e., the set of viewpoints which adequately cover the task), it suffices to intersect the solution sets for each individual criterion. Other bivalent coverage models have taken much the same approach, intersecting the sets of covered points generated by each criterion, exemplified by the “feasible region” result of Erdem and Sclaroff [13]. In the multi-camera context, the overall coverage of the scene is of interest; this is usually found by taking the union of the coverage sets for each individual camera, as Erdem and Sclaroff also demonstrate.

Mittal and Davis [43] integrate their random occlusion criterion with their other (deterministic) constraints to obtain an overall graded coverage function for each point and orientation.

Several models also provide mechanisms to compute the overall $k$-coverage of a scene. Erdem and Sclaroff [13] imply this capability in their experimental figures, but none of their experimental problem statements require $k$-coverage with $k > 1$. Liu et al. [62] use an intersection-union approach similar to that described in Section 2.2.2 in their work, which focuses specifically on $k$-coverage.

Mittal and Davis [43] discuss hypothetically more complex “algorithmic constraints” involving the interplay of various constraints between multiple cameras, for such tasks as stereo matching. In the framework presented in Section 2.2, this implies that the coverage function for the sensor system would be more complex than the basic $k$-coverage function, and might involve higher-level task parameters. To some extent, particularly on the view angle criterion, this is realized in the $k$-coverage model of Shen et al. [51].

Task Model Specification

A relevance function defined over a relatively wide volume of $S$ is most commonly used in view planning and view reconfiguration applications, where it comprises the coverage objective. Often, $⟨R⟩$ is implicitly the working volume (as in the art gallery problem); in order to support a general problem definition, however, the coverage objective should be explicit, and independent of the environment model and the admissible viewpoint set. Jiang et al. [36], Hörster and Lienhart [15], Angella et al. [38],...
Zhao et al. [17], and Malik and Bajcsy [39] all allow explicit specification of a relevance function in some form.

The aforementioned applications can also benefit from a real-valued relevance function. Hörster and Lienhart [15] use a continuous weighted relevance function in their problem instance definition; in the actual discrete domain of their algorithm, this informs the density of control points (essentially, the discrete $\langle R \rangle$). Jiang et al. [36] optimize directly over the continuous function, using distinct regions with integer weights to simplify the weighted coverage computation. Piciarelli et al. [20] define a relevance function as a mapping of discrete points to real values.

The values of the imaging constraints in such models as those of Cowan and Kovesi [7] and Tarabanis et al. [8] are direct task requirements. Erdem and Sclaroff [13] emphasize the task-specific nature of the criteria in the multi-camera view planning context. One of their experiments specifies a higher minimum resolution requirement on certain parts of their $\langle R \rangle$, showing how a mixture of task requirements can be handled (in this case, as a more precise alternative to a weighted relevance function).

One form of the view planning problem constrains the minimum required proportion of $\langle R \rangle$ covered by the solution (while maximizing or minimizing some other variable, such as the cost of the sensor system), a task-specific requirement. This is one of the four variations studied by Hörster and Lienhart [15]. The weighted form of this proportion, termed the “coverage rate” by Jiang et al. [36], may fill a similar role, as in the view planning problem case studied by Shen et al. [51].

### 3.2.4 State of the Art and Open Problems

To date, no geometric model has fully captured the phenomenon of visual coverage in a representation suitable for the general multi-camera context. While some of the single-camera view planning models surveyed exhibit sufficient fidelity and generality to apply to a wide range of tasks, they are ill-suited to modeling typical systems and environments involving multiple cameras, and in their present form would likely put prohibitive computational requirements on optimizations involving even relatively small sets of cameras. Conversely, in expressly designing multi-camera models in forms suitable for specific optimization techniques, the remainder of the authors mentioned have restricted applicability to relatively specific problem classes. Mittal and Davis [43] appear to have designed the most accurate and general model to date suitable for multi-camera optimization, but it is still somewhat restricted by certain assumptions, notably its two-dimensional final representation, and its lack of a focus criterion.

The ideal geometric coverage model would not only accurately model visual coverage in a form convenient for multi-camera systems and their environments, with as few assumptions as possible and allowing for generalized task requirements, but also provide this information in a form tractable for powerful optimization techniques. It is clear from the preceding discussion that the factors involved in a model achieving the former goal would be highly complex, complicating success in the latter goal. The prevailing approach to this problem has been to design the model to be as accurate and general as possible for one specific optimization technique from the outset, but this has failed to produce the ideal model. This suggests that attempting to achieve the first goal in isolation could, at the very least, produce a tool for evaluation, but may also yield new insights into the nature of multi-camera coverage that may lend the model, or some derivative thereof, to an appropriate optimization scheme.

Significantly, with every model surveyed here approaching a practical level of fidelity over a reasonably general application scope, the authors have resorted to metaheuristic optimization techniques,
suggesting that the nature of the general problem is such as to be tractable only for such an approach. If so, the model design should embrace this fact, and concentrate on fidelity, generality, and complexity of evaluation. The remainder of this thesis hypothesizes that this is the case, and the model derived in Chapter 4 is developed accordingly.

Most sources surveyed have assumed that the coverage model employed reflects a posteriori task performance, with little or no validation of the model itself. In order to evaluate fidelity and generality, a generic scheme for relating the coverage metric to a task performance metric should be developed and adopted. A simple statistical measure, such as the Pearson product-moment correlation coefficient, might suffice; depending on the nature of the coverage and performance metrics, other measures might be more illuminating.

### 3.3 Topological Models of Visual Overlap

The second type of model surveyed is the topological representation of coverage overlap, corresponding to the form presented in Section 2.3.1.

#### 3.3.1 Anatomy of a Visual Overlap Model

A coverage overlap model describes the topology of a multi-camera system in terms of mutual coverage of some part of the scene. Typically, the camera node is the atomic entity, and of interest are the node-level coverage overlap relationships. The most general form is a hypergraph, as described in Section 2.3.1. A much more common form is the vision graph (Figure 3.3), which is an ordinary graph (a 2-uniform hypergraph) and thus considers only pairwise coverage overlap.

![Vision Graph Example](image)

**Figure 3.3:** Vision Graph Example – From 2D coverage geometry (left) to pairwise overlap topology (right). The vision graph is the most common form of topological model of coverage overlap, owing to its usefulness in various applications.

#### 3.3.2 Overlap Models by Application

The earliest examples of visual coverage overlap models are found in multi-view registration applications, including video sequence registration and 3D range image registration. Since the objective is to align visual data from multiple views, it is clearly useful to know which views overlap and thus might have some corresponding features for registration. Sawhney et al. [63] propose a graph formalism
of the coverage overlap relationships between multiple views for video sequence registration, with each frame (view) represented by a vertex, addressing the fact that frames which are not temporally adjacent may still be adjacent in terms of overlap topology. Kang et al. [64] construct a similar graph representation of the overlap topology of frames, in which edges indicate either temporal or spatial (overlap) adjacency. Their algorithm searches for an optimal path in this graph to minimize error in global registration. Huber [65] constructs a graph for registration of partial 3D views using an overlap criterion on the range images, analyzes the registration problem through its connectivity properties, and performs reconstruction over a spanning tree. Sharp et al. [66] also study 3D range image registration using a similar graph formalism, which they assume exists a priori. They approach the global registration problem by first considering registration over basis cycles within the graph, then merging the results using an averaging technique.

Knowledge of multi-camera system topology in terms of coverage overlap is a useful precursor to full metric multi-camera calibration. Antone and Teller [67] require, as input to their calibration algorithm, a graph of camera adjacency; although the criterion for edge presence is based on position (from GPS), since the algorithm targets omni-directional cameras, this is supposed to approximate coverage overlap and is thus a vision graph. Brand et al. [68] further develop this work, using directionally-constrained graph embeddings. Devarajan et al. [69, 70] name and explicitly describe the vision graph, pointing out its distinctiveness from the communication graph (a notable departure from traditional sensor networks), and demonstrating its usefulness in informing a full calibration algorithm as to which camera pairs should attempt to find a homography. However, they offer no means of obtaining the vision graph automatically, instead making the temporary assumption that it is available a priori. Cheng et al. [71] address this issue by approximating the vision graph via pairwise point feature matching, and describe a full calibration algorithm also employing the feature data following the procedure of Devarajan et al. Kurillo et al. [72] construct a weighted vision graph based on the number of shared calibration points, then optimize the set of calibration pairs by finding a shortest-path spanning tree. Bajramovic et al. [73] perform multi-camera calibration over connected components of their vision graph, which they construct independently using the normalized joint entropy of point correspondence probability, one of several methods described by Brückner et al. [74]. Mavrinac et al. [75] describe the vision graph as a theoretical upper bound for the connectivity of their grouping and calibration graphs.

Overlap topology can be used to help establish direct tracking correspondence, a subproblem of tracking correspondence involving agents simultaneously visible in multiple cameras. This is useful for camera handoff among overlapping cameras [76, 77]. In this context, overlap topology is usually considered to be a subset of a more general transition topology (described in Section 2.3.2, surveyed in Section 3.4). Stauffer and Tieu [78] describe a “camera graph” which identifies with the vision graph, estimating camera overlap from sets of likely correspondences between tracks. This graph is then used as feedback to improve tracking correspondence. Mandel et al. [79] use a probabilistic approach on motion correspondence to establish overlap topology for tracking purposes. In a series of papers on the topic, Van Den Hengel, Detmold, Hill, and various co-authors [80, 81, 82, 83, 84] describe the “exclusion” approach, whereby the vision graph begins complete and edges are removed based on contradictory occupancy observations, with a target application of tracking correspondence in surveillance networks. Lobaton et al. [85, 86, 87] propose a simplicial complex representation of overlap topology dubbed the “CN-complex,” primarily targeted at tracking applications. Overlap topology is employed
by Song et al. [88] as part of their consensus approach to tracking and activity recognition.

Camera networks are often composed of devices with limited computational and energy resources. Knowledge of overlap topology can help inform efficient scheduling of node activity. Ma and Liu [89] estimate the correlation between views using their geometric coverage model (mentioned in Section 3.2), to improve the efficiency of video processing in camera networks with partially redundant views. However, the information used is not strictly topological, and the method applies specifically to two-camera systems. Dai and Akyildiz [90] address the latter issue by extending the correlation problem to multiple cameras, but their model is also not strictly topological. Kulkarni et al. [53] construct a vision graph using a Monte Carlo feature matching technique with a geometric model component, and demonstrate its use in duty cycling and triggered wake-up. Mavrinac and Chen [91] propose a coverage hypergraph derived directly from their geometric coverage model, and apply it to the optimization of load distribution using a parallel machine scheduling algorithm.

### 3.3.3 Analysis and Comparison of Models

Table 3.2 compares the nature and properties of a selection of topological coverage overlap models from the literature, grouped by application (indicated in the first column). The second column identifies the combinatorial structure used (whether explicit or interpreted), and the following three columns indicate which additional properties are exhibited: edge weighting, $k$-view modeling, and modeling of partial views, respectively. The remaining five columns specify which type of data is used to estimate the model: geometric coverage information, registration results, local feature matching, occupancy correlation, or motion correlation.

<table>
<thead>
<tr>
<th>Model</th>
<th>App.</th>
<th>Properties</th>
<th>Estimation</th>
</tr>
</thead>
<tbody>
<tr>
<td>Sawhney et al. [63]</td>
<td>R</td>
<td>G</td>
<td></td>
</tr>
<tr>
<td>Huber [65]</td>
<td>R</td>
<td>G</td>
<td></td>
</tr>
<tr>
<td>Sharp et al. [66]</td>
<td>R</td>
<td>G</td>
<td></td>
</tr>
<tr>
<td>Cheng et al. [71]</td>
<td>C</td>
<td>G</td>
<td></td>
</tr>
<tr>
<td>Kuriillo et al. [72]</td>
<td>C</td>
<td>G</td>
<td></td>
</tr>
<tr>
<td>Bajramovic et al. [73]</td>
<td>C</td>
<td>G</td>
<td></td>
</tr>
<tr>
<td>Mavrinac et al. [75]</td>
<td>C</td>
<td>G</td>
<td></td>
</tr>
<tr>
<td>Stauffer and Tieu [78]</td>
<td>DT</td>
<td>G</td>
<td></td>
</tr>
<tr>
<td>Mandel et al. [79]</td>
<td>DT</td>
<td>G</td>
<td></td>
</tr>
<tr>
<td>Van Den Hengel et al. [80]</td>
<td>DT</td>
<td>G</td>
<td></td>
</tr>
<tr>
<td>Lobaton et al. [87]</td>
<td>DT</td>
<td>SC</td>
<td>✓</td>
</tr>
<tr>
<td>Kulkarni et al. [53]</td>
<td>S</td>
<td>HG</td>
<td>✓</td>
</tr>
<tr>
<td>Mavrinac and Chen [91]</td>
<td>S</td>
<td>HG</td>
<td>✓</td>
</tr>
</tbody>
</table>

¹The publication cited is a prior version of Chapter 8. The coverage hypergraph first published for this work is described, in more general terms, in Section 2.3.1.
3.3. Topological Models of Visual Overlap

Combinatorial Structure

Although not all of the coverage overlap models surveyed are explicitly formalized as graphs (or hypergraphs), they can be cast as cases of the general model described in Section 2.3.1 without loss of information. The original descriptions given by the authors are summarized here, and instances where ancillary information not captured by the graph representation is present are highlighted.

The vision graph as described by Devarajan et al. [69, 70]—an undirected, unweighted graph with vertices representing cameras and edges indicating sufficient coverage overlap for the purposes of the task—is the simplest and most common combinatorial structure for models of coverage overlap topology seen in the literature. This is the explicit form of the models of Cheng et al. [71], Bajramovic et al. [73], Mavrinac et al. [75], and Stauffer and Tieu [78]. The graphs of Sawhney et al. [63] and Kang et al. [64] describe temporal and spatial adjacency, but since in their application temporally adjacent frames are assumed to be spatially adjacent also, they are effectively describing the vision graph structure. The graphs described by Huber [65] and Sharp et al. [66] are also essentially vision graphs; although edges are annotated with pairwise relative pose and other relations, this information is not part of the overlap model proper. Mandel et al. [79] and Van Den Hengel et al. [80, 81] do not explicitly present graph formalisms, but maintain sets of hypotheses about coverage overlap which correspond to edges in the vision graph.

Some recent models extend the captured topology from pairwise overlap to general $k$-overlap, requiring a hypergraph-like structure to accomodate the relationships. Lobaton et al. [85, 86, 87] partially achieve this with a simplicial complex representation. This choice of representation, over a more abstract structure such as a hypergraph, seems to stem from the focus being more on geometrical properties and operations and less on combinatorial optimization. They are interested in overlap topology only up to 2-simplices (or 3-simplices in a hypothetical extension to three dimensions), so their model does not capture general $k$-overlap. Kulkarni et al. [53] model the full $k$-overlap topology of the camera network, although they do not explicitly formalize this model in a hypergraph representation or use any combinatorial techniques. Mavrinac and Chen [91] present an explicit hypergraph representation of $k$-overlap topology, to which combinatorial optimization is applied.

The assignment of one vertex to each camera is sensible for most purposes, but a few models eschew this paradigm and subdivide vertex assignment into cells of coverage. Motivations for doing so vary. Van Den Hengel et al. [80, 81] subdivide views into an arbitrary number of “windows” to handle partial coverage overlap of cameras (due to the specifics of their estimation method). Mandel et al. [79] divide views into regions for a similar reason. In both cases, it appears that the model of interest to the eventual application recombines the coverage cells (which are each associated with a specific camera) to the more usual granularity of one vertex per camera. Lobaton et al. [85, 86, 87] divide the two-dimensional geometric coverage of each camera at rays to occlusion events, which they call “bisecting lines,” allowing their model to accurately capture some geometric properties of overlap, such as static occlusions within the field of view, as shown in Figure 3.4. This increased granularity is preserved, and shown to be beneficial in optimizing tracking applications.

Calibration and scheduling applications of overlap models often make use of graph optimizations related to path length. In such cases, weighting vision graph edges proportionally to the degree of coverage overlap can yield better results than the unweighted graph. Given an edge $e_{AB} = \{A, B\}$ linking cameras $A$ and $B$, Kurillo et al. [72] assign the weight $w(e_{AB}) = 1/N_{AB}$, where $N_{AB}$ is the
number of common reference points detected by cameras A and B. Kulkarni et al. [53] similarly compute the degree of $k$-overlap from the number of common reference points of $k$ cameras. Both describe methods of handling non-uniform spatial distributions of reference points. Mavrinac and Chen [91] theoretically use the volume of intersection between $k$ cameras’ geometric coverage models to weight hyperedges, but in practice, the required polytope intersection procedure is NP-hard, so they use a uniform distribution of points to compute a discrete approximation.

**Estimation from Visual Data**

In theory, overlap topology is a derived property of the geometric coverage of the camera network, as described in Section 2.3.1. However, since it is often employed in applications where geometric coverage information is unavailable, especially in calibration initialization, it is often necessary to estimate it using visual data. Finding correspondences between visual data in some form among views is the obvious means—if camera A matches a piece of its own information to one from camera B, then a hypothesis of mutual coverage between A and B can be made or strengthened.

For tasks which already make use of correspondences between local image features, the same information, or some subset thereof, can be used to recover overlap topology. This is the approach originally suggested by Devarajan et al. [69, 70]. Because their algorithm works in an offline centralized context, Kang et al. [64] are able to directly correlate image features to infer topology. Cheng et al. [71], addressing the camera network calibration problem, attempt to make such an approach scalable in an online distributed context by instead sharing “feature digests” of SIFT [92] descriptors among camera nodes. Bajramovic et al. [73, 74] use the pairwise joint entropy of point correspondence probability distributions, based on SIFT feature descriptors, as a measure of overlap. Kurillo et al. [72] use direct matching of a more sparse but more accurate set of features, obtained from a structured calibration target. A similar approach is taken by Kulkarni et al. [53], although the structured target is only used for topology inference and is unrelated to their application. In their case, both the degree and geometry of coverage overlap are estimated using a Monte Carlo technique, whereby the target is imaged at random reference points, and the $k$-covered Voronoi cells around each point contribute to the estimate for each of the $k$ cameras covering it.

Registration-based applications are typically iterative, and some overlap models are updated using
new correspondence data available in each iteration. Sawhney et al. [63] infer global overlap topology iteratively, using feedback from a local coarse registration stage to recover graph edges, and subsequently performing local fine registration on adjacent views. An analogous three-dimensional process is employed by Mavrinac et al. [75] in a distributed calibration algorithm, with coarse registration results iteratively building a grouping graph which then informs pairwise fine registration. Huber [65] also uses candidate registration matches to iteratively infer overlap topology.

Camera networks often have wide baselines and large rotational motion between cameras, over which local feature detectors generally have poor repeatability and matching performance [93, 94]. Fortunately, they offer the possibility of matching online motion data instead of static features, which can be more robust under some circumstances. Stauffer and Tieu [78] argue that the descriptiveness, spatial sparsity, temporal continuity, and linear increase in volume over time of tracking correspondences make them more reliable in matching than static features. They correlate local tracks between cameras over time, and infer a visual graph edge where the expectation of a match exceeds a threshold. Mandel et al. [79] take a slightly different approach, detecting local motion and attempting to correlate it with motion observed in other cameras, via a distributed algorithm. Lobaton et al. [87] automatically decompose cameras into coverage cells by locally finding “bisecting lines” at which occlusion events occur (e.g. walls), then, with a distributed algorithm, globally estimate cell overlap by matching concurrent occlusion events over time.

Van Den Hengel et al. [80, 81] take the reverse approach to those described thus far. Their so-called “exclusion” algorithm begins by assuming all camera nodes have overlapping coverage, thus a complete visual graph, and eliminates edges over time using occupancy data to rule out coverage overlap. This method does not require any correspondence between observations; if camera A is occupied (currently observing an object) and camera B is unoccupied, this is evidence that A and B do not have mutual coverage, which through observation ratio calculations and thresholding contributes to the final model. Partial overlaps are handled by dividing camera coverage into an arbitrary number of coverage cells. Hill et al. [84] describe a number of potential shortcomings in real-world operation, along with ways of mitigating the adverse effects on performance. Detmold et al. [82, 83] extend the approach into an online distributed context for scalability and dynamic updating of the model.

One direct route to an overlap model well-suited to the task at hand is to use the very visual data used by the task itself to estimate the model, if this data (or similar data) is available. This can clearly be seen in most cases of registration, feature-based calibration, and tracking applications in Table 3.2. In a distributed camera network, depending on the nature of the data and the amount of it required to establish accurate overlap estimates, there is a potential scalability issue since, initially, the data must effectively be broadcast to all other nodes. As mentioned in the preceding section, Cheng et al. [71] address this using digests of the SIFT features to establish overlap topology, then share the substantially larger full feature data pairwise only among cameras with sufficient overlap for calibration. Kurillo et al. [72] also use calibration feature points to estimate overlap; scalability is less of an issue because they use a structured calibration target, which yields a set of features both sparse enough to distribute among many cameras and robust enough to achieve accurate metric calibration. In the algorithm of Stauffer and Tieu [78], overlap topology estimation is part of the closed-loop tracking correspondence task itself. The scheduling applications of Kulkarni et al. [53] and Mavrinac and Chen [91] use occupancy correlation and geometric coverage, respectively, in an attempt to obtain the same fundamental information, viz. the degree of content pertinent to the task in each k-view.
3.3.4 State of the Art and Open Problems

The vision graph is a well-established concept and theoretical tool in multi-camera networks. In the application classes of multi-view registration and calibration, which (in the surveyed cases) involve pairwise coverage relationships exclusively, it has proven useful in its basic form. Additional optimizations are possible with appropriate use of edge weights and related combinatorial techniques, as demonstrated by Sharp et al. [66] and Kurillo et al. [72] for their respective applications.

When used in direct tracking correspondence, the limitations become apparent. Arbitrary subdivisions of camera nodes into partial coverage cells appears to improve performance, but this is unsatisfying from a theoretical standpoint and raises scalability concerns. Lobaton et al. [85, 87] present an explicit departure from the graph model, allowing them to represent 2-coverage and 3-coverage in a simplicial complex; however, presumably since their application does not require it, general k-coverage modeling is absent. Kulkarni et al. [53] and Mavrinac and Chen [91] use more general hypergraph (or equivalent) models explicitly designed for general k-coverage, suitable for scheduling in distributed camera networks, but ignore the coverage subdivisions needed by tracking applications.

The generalized coverage hypergraph model presented in Section 2.3.1 appears to include all of the information necessary to fit the needs of each of the applications covered here, and being a relatively straightforward combination of existing concepts from the literature, should be backwards-compatible with all of the reviewed sources. In the absence of task-specific geometric coverage information, it is sensible to use the task data directly to approximate the model. It remains an open question whether the nature of the information contained in edge weights, and the additional combinatorial optimizations they make possible, can be incorporated into such a unified framework.

3.4 Topological Models of Visual Transition

The third and final type of model surveyed is the topological representation of transition probabilities and/or timings, corresponding to the form presented in Section 2.3.2.

3.4.1 Anatomy of a Visual Transition Model

A transition model describes the topology of a multi-camera system in terms of the probability and/or timing of dynamic agents transitioning from one region of coverage to another. Relationships may exist among camera nodes with no mutual scene coverage (non-overlapping cameras). Since the target application class is agent tracking, the granularity of the topology may extend down to subsets of camera nodes’ coverage: entry and exit points and regions of overlap are often considered individually.

The general form is a weighted directed graph, as described in Section 2.3.2. The vertex set N, in this case, consists of coverage cells, which may represent an individual camera node’s coverage hull or some portion thereof, such as an entry or exit zone (note that a coverage cell may be both an entry zone and an exit zone).

3.4.2 Transition Models by Application

Transition models are largely aimed at one particular application class: predictive tracking in (generally) non-overlapping camera networks. For a locally tracked agent leaving one coverage cell, the
3.4. Topological Models of Visual Transition

The objective is to predict in which other coverage cell(s) the agent will reappear, possibly to inform camera handoff. A special case occurs when the cameras have coverage overlap, which is addressed by several models of overlap topology as the direct tracking correspondence problem (covered in Section 3.3.2). Javed et al. [95] show that, in the context of non-overlapping tracking correspondence, transition probabilities and durations are dependent on individual correlations of entry and exit zones, of which each camera may have a number. Their geometric counterparts are coverage cells, and in a combinatorial transition model, they comprise the vertex set. Various techniques have been applied to this type of model to aid in tracking agents across non-overlapping views (i.e. through unobserved regions).

The model presented by Ellis et al. [96] exemplifies this approach. Their method automatically identifies entry and exit zones in each camera (a problem previously addressed by Stauffer [97]), then finds the transition topology by temporally correlating a large number of local trajectories between cameras, requiring no actual tracking correspondence. Makris et al. [98] extend this method and further develop its theoretical basis. Stauffer [99] operates on a closely related model, but presuming the availability of a coverage overlap model—Stauffer cites his own previous work with Tieu [78]—treats cliques of overlapping cameras (connected components in the vision graph) as the larger coverage structure containing entry and exit zones, on the premise that the overlapping case is better handled by robust direct correspondences. The aforementioned methods ascribe to observations an implicit correspondence, and assume a unimodal statistical distribution of transitions. Tieu et al. [100] address this with a method capable of handling multimodal distributions.

Marinakis et al. [101, 102] consider cameras with full coverage of widely-separated sections of hallways in a building, so that transitions are constrained to the hallway topology. Due to these constraints, the entry and exit zone coverage cells (transition graph vertices) are the cameras themselves, and the cameras need only be capable of detecting an agent’s presence with reasonable fidelity for their method to successfully estimate the topology. Niu and Grimson [103] target a vehicle tracking application, using appearance to match observations between, and infer the topology of, non-overlapping
3.4. Topological Models of Visual Transition

cameras. Erdem and Sclaroff [104] appear to make similar assumptions in modeling a hybrid camera network including active cameras; again, the entry and exit zone coverage cells are the cameras themselves. In their case, event correlation is assumed to be known.

Dick and Brooks [105] approach the predictive tracking problem with a Markov model which captures transition topology after a training phase, albeit not in an explicitly combinatorial form, dividing the view into blocks over which the topology is found. The method of Gilbert and Bowden [106] incrementally learns the topology between recursively subdivided blocks of the views; their method does not require a training phase and can adapt to changes in the configuration of the multi-camera system. Both yield a probabilistic topological model which can be used in conjunction with appearance-based matching to track across disjoint views.

Zou et al. [107] are interested in tracking humans, and integrate appearance-based agent correspondence based on face recognition into the inference method of Ellis, Makris, and Black, for improved robustness in their target instance. Nam et al. [108] also specifically track humans, and an appropriate appearance model is integral to their estimation method. The method of Farrell and Davis [109] falls within this category as well, and is notable for its expressly distributed approach, which affords scalability to large, distributed surveillance networks.

The coverage overlap model developed by Van Den Hengel, Detmold, Hill, et al. [80, 81, 82, 83, 84] can be extended, as the authors explain, to capture non-overlapping transition topology by adding a temporal padding window to the exclusion method.

3.4.3 Analysis and Comparison of Models

Table 3.3 compares the properties of a selection of topological transition models from the literature. Interpreting each model as a graph, the first and second columns indicate whether the graph is directed and/or weighted, respectively. The third column indicates whether vertices of the graph represent individual entry/exit points, of which each camera may have several; the implication otherwise is that the granularity is at the level of cameras only. The fourth column indicates whether the model includes an explicit source/sink vertex, for agents entering or leaving the scene. The fifth column indicates whether the graph models transitions between overlapping cameras, thus implicitly modeling coverage overlap to the extent described with direct tracking correspondence applications in Section 3.3. The final two columns specify which type of data is used to estimate the model: statistical correlation between temporal events, or correlation via an appearance model.

Combinatorial Structure

Relatively few of the transition models surveyed are explicitly presented as graphs resembling the generalized model described in Section 2.3.2. Marinakis et al. [101, 102] model the topology in a directed, unweighted graph, in which vertices represent camera nodes and arcs represent possible transitions. Transition probabilities and durations are captured separately in an agent model. Erdem and Sclaroff [104] define a directed graph with similar structure, albeit with arc weights as random variables representing the expected transition time. The graph of Nam et al. [108] also has a vertex for each camera, but also has intermediate vertices representing either an overlapping or non-overlapping transition point and a source/sink vertex; since individual entry and exit zones are not represented, the graph is undirected. Zou et al. [107] use essentially the same model as Ellis, Makris, and Black [96, 98].
but treat it explicitly as a weighted, directed graph, with vertices representing entry and exit zones and arcs indicating possible transitions. Trivially, related models, such as those of Stauffer [99] and Tieu et al. [100], could be treated similarly. The transition matrix of Dick and Brooks [105] can be interpreted as an incidence matrix for the transition graph. In general, it is not difficult to apply a graph interpretation to any of the models surveyed here.

As discussed in Section 3.3.3, coverage overlap models typically represent each camera node as a vertex, a structure which offers useful combinatorial properties in most applications. Some transition models employ this structure as well. Marinakis et al. [101, 102] assume widely separated cameras and wish to avoid dealing with complex local tracking, so this is the sensible representation for their case. Niu and Grimson [103] and Farrell and Davis [109] also consider transitions only between strictly non-overlapping cameras. In scenes of even moderate complexity, however, a transition topology among individual entry and exit points is more germane to predictive tracking. This structure is described by Ellis, Makris, and Black [96, 98] and used by a plurality of the models surveyed [99, 100, 107, 108]. Dick and Brooks [105] do not automatically determine entry and exit points, but do divide the cameras into coverage cells, which would induce the vertices in a graph interpretation of their model.

Makris et al. [98] include a source/sink vertex (which they call a “virtual node”), in addition to the entry and exit zone vertices, to handle the probabilistic paths of agents entering or leaving the overall coverage of the camera network. Marinakis et al. [102] and Nam et al. [108] also include such a vertex in their models.

Among explicit graph models with arc weights, the definition of the weighting function varies. Makris et al. [98] annotate arcs in the graphical representation of their model according to the probability of transition, computed from the cross-correlation of the temporal sequences of departure and arrival events at each entry and exit zone (vertex), but do not operate on it as a weighted graph. Zou et al. [107] explicitly apply this weighting to the graph representation, with a modified correlation function based on both identity and appearance (as opposed to identity only). In contrast, Nam et al. [108] weight arcs based on the mean duration of transitions between cameras.

---

**Table 3.3: Comparison of Selected Topological Transition Models**

<table>
<thead>
<tr>
<th>Model</th>
<th>Properties</th>
<th>Estimation</th>
</tr>
</thead>
<tbody>
<tr>
<td>Ellis et al. [96]</td>
<td>✓ ✓ ✓ ✓ ✓ ✓</td>
<td>●</td>
</tr>
<tr>
<td>Makris et al. [98]</td>
<td>✓ ✓ ✓ ✓ ✓ ✓</td>
<td>●</td>
</tr>
<tr>
<td>Dick and Brooks [105]</td>
<td>✓ ✓ ✓ ✓ ✓ ✓</td>
<td>●</td>
</tr>
<tr>
<td>Marinakis et al. [101]</td>
<td>✓ ✓ ✓ ✓ ✓ ✓</td>
<td>●</td>
</tr>
<tr>
<td>Stauffer [99]</td>
<td>✓ ✓ ✓ ✓ ✓ ✓</td>
<td>●</td>
</tr>
<tr>
<td>Tieu et al. [100]</td>
<td>✓ ✓ ✓ ✓ ✓ ✓</td>
<td>●</td>
</tr>
<tr>
<td>Niu and Grimson [103]</td>
<td>✓ ✓ ✓ ✓ ✓ ✓</td>
<td>●</td>
</tr>
<tr>
<td>Nam et al. [108]</td>
<td>✓ ✓ ✓ ✓ ✓ ✓</td>
<td>●</td>
</tr>
<tr>
<td>Zou et al. [107]</td>
<td>✓ ✓ ✓ ✓ ✓ ✓</td>
<td>●</td>
</tr>
<tr>
<td>Farrell and Davis [109]</td>
<td>✓ ✓ ✓ ✓ ✓ ✓</td>
<td>●</td>
</tr>
<tr>
<td>Erdem and Sclaroff [104]</td>
<td>✓ ✓ ✓ ✓ ✓ ✓</td>
<td>●</td>
</tr>
</tbody>
</table>
3.4. Topological Models of Visual Transition

Estimation from Visual Data

It is normally assumed that the camera network is uncalibrated and that information about the scene and agent dynamics is unavailable a priori. For the purposes of this discussion, we will approach the estimation methods assuming that entry and exit zones are known, either estimated separately [97, 96, 106], specified a priori, or implicit, as in the case where each camera is a single entry/exit zone. If agents can be uniquely identified and reliably matched between all generally non-overlapping views, and a sequence of their arrival and departure events is obtained over a period of time, distributions of the probabilities and durations of transitions can be established. From this information, all of the parameters of the general transition model can be obtained.

Unfortunately, visual correspondence of agents of arbitrary appearance between generally disjoint views is notoriously difficult. Ellis, Makris, and Black [96, 98] sidestep this challenge with a method of estimation based on pure temporal correlation of otherwise unmatched observations. Essentially, they assume implicit correspondence between all pairs of arrival and departure events, and seek a single mode of temporal correlation between each pair of entry and exit zones within a time window (positive and negative); every peak above a certain threshold induces an arc in the transition graph between the associated vertices. Stauffer [99] employs a similar method, but considers transitions between overlapping cameras separately, so the transition time window is positive only. Tieu et al. [100] handle more general statistical dependencies, capturing richer multi-modal transition distributions rather than simply a mean transition duration, and thus permitting topology estimation from more complex agent behavior. Marinakis et al. [101, 102] also avoid direct correspondence. They assume that the dynamics of an agent is a Markov process, and estimate the parameters of this process—the probabilities and durations of transitions—using a Monte Carlo expectation-maximization method.

Methods which do rely on appearance-based agent correspondence normally have a narrower application focus. Dick and Brooks [105] require a training phase for their Markov model which relies on colour-based correspondence. Niu and Grimson [103] rely on correspondence of tracked vehicles using an appearance model based on colour and size. The estimation method of Nam et al. [108] centers around correspondence based on background subtraction and a human appearance model. Farrell and Davis [109] employ an information-theoretic appearance matching process, and infer the expected transition model from the accumulated evidence using a modified multinomial distribution. Their method is also notable for its distributed design: its “semi-localized” processing yields a scalable algorithm for which the authors demonstrate successful results in networks up to 100 nodes.

Zou et al. [107] integrate correspondence based on face recognition into the previously described statistical method of Ellis, Makris, and Black, resulting in a hybrid approach which they claim outperforms methods based purely on either identity or appearance.

Transitions Between Overlapping Cameras

There is a question as to how transitions between cameras with overlapping coverage should be handled in transition models. Referring to the example agent paths in Figure 3.6, it is clear how to handle the transition between non-overlapping cameras shown in Figure 3.6(a), as the surveyed methods unanimously agree: an arc from A or its exit zone to B or its entry zone, with a positive transit duration. However, in the transition between overlapping cameras shown in Figure 3.6(b), the agent passes
through the entry zone of $B$ before passing through the exit zone of $A$, and the agent is observed by one or both cameras during the entire transition. Transitions from one entry or exit zone to another within a single camera’s coverage can be thought of as a special case of this scenario.

![Figure 3.6: Possible Cases of Transition – Dark ellipses denote entry and exit zones, and the dotted line indicates the agent path. Some transition models treat these as distinct cases.](image)

Ellis, Makris, and Black [96, 98] deal with the overlapping case as with the non-overlapping case. For a given departure event at time $t_1$, they check for arrival events at time $t_2 \in [t_1 - T, t_1 + T]$, where $T$ is a temporal search window. Thus, in Figure 3.6(a), $t_2 > t_1$, whereas in Figure 3.6(b), $t_2 < t_1$. The advantage of this approach is that it does not require prior estimation of overlap topology, and uses a single process to estimate transition topology for a general-case camera network with overlapping and/or non-overlapping cameras.

Stauffer [99] argues that the overlapping case is best handled by more robust direct tracking correspondence, and proposes first estimating overlap topology [78], then treating connected components in the vision graph as single “cameras”—in general, with multiple entry and exit zone vertices—in the transition model. The advantage of this approach is improved robustness in estimating the overlapping portions of the transition topology, assuming a reliable means of finding inter-camera correspondences of agents and/or their tracks is available.

### 3.4.4 State of the Art and Open Problems

Numerous researchers have converged on the structure described in Section 2.3.2, to varying degrees. As with coverage overlap models, it is safe to say that this generalized model subsumes all existing cases; individual models have left out certain properties (arc directivity and weights, node subdivision, source/sink node) either because they are unnecessary for the particular application case or else to facilitate optimization. Given the clear focus on a single application class, future optimization efforts should adopt such a unified model, if possible, for the sake of general applicability.

Estimation of the graph from visual data is split between statistical temporal correlation and appearance-based correlation. Given the complementary strengths of both methods, the way forward seems to be a hybrid approach in the vein of Zou et al. [107]. If agent dynamics are being modeled probabilistically for the purposes of random occlusion, as by Mittal and Davis [43], this may also be informative for transition model approximation.

One point of contention, to which the answer is not yet clear, is whether the graph should model transitions strictly between non-overlapping coverage cells, with overlapping transitions handled separately as proposed by Stauffer [99], or all transitions. If the relative reliability of the approximations for overlapping transitions is the issue, implementation of the aforementioned hybrid approximation approach may favor the latter unified model.
A Visual Coverage Model

An unnoticed corner of the world suddenly becomes noticed, and when you notice something clearly and see it vividly, it then becomes sacred.

Allen Ginsberg (1926–1997)

4.1 Overview

This chapter presents the full formulation of a general, high-fidelity coverage model for multi-camera systems. This model is developed in response to the observations of Chapter 3, in which a general perspective on model-based approaches to coverage problems illuminates the effects of design decisions across a broad spectrum of multi-camera system applications. In keeping with the philosophy of Chapter 2, the sensor system, environment, and task models maintain good decoupling, and the parameters for each are direct and intuitive.

The presentation in this section relies heavily on an understanding of the model of image formation, optical effects, and calibration procedures reviewed in Appendix B.

4.2 Visual Stimulus Space

A description of a point feature including its position in space and the normal of the surface on which it lies is sufficient to subsume all of the vision tasks considered by authors in Chapter 3 [7, 8]. This assumes that all tasks are, or can trivially be made, invariant to rotation about the optical axis (within the image plane); since an image may be rotated with very little loss of the original information, this assumption is almost universally valid. Complex features not localized at a point [25] may be described by sets of such point features. The directional space encodes this information, and is thus an appropriate stimulus space for vision.
4 Definition (Directional Space)

The directional space $\mathbb{D}^3 = \mathbb{R}^3 \times [0, \pi] \times [0, 2\pi)$ consists of three-dimensional Euclidean space plus direction, with elements of the form $(p_x, p_y, p_z, p_\rho, p_\eta)$.

A point $p \in \mathbb{D}^3$ is termed a directional point. For convenience, its spatial component is denoted $p_s = (p_x, p_y, p_z)$ and its directional component $p_d = (p_\rho, p_\eta)$.

A pose $P \in SE(3)$, comprising rotation $R$ and translation $T$, may be extended to the directional space as $P : \mathbb{D}^3 \to \mathbb{D}^3$; for simplicity of notation, they will be considered interchangeable. For $p \in \mathbb{D}^3$, the spatial component is transformed as per (A.15), i.e. $P(p_s) = Rp_s + T$. The directional component is transformed as follows. If $d$ is the unit vector in the direction of $p_d$, then

$$P(p_d) = \begin{bmatrix} \arccos([Rd]_z) \\ \text{atan2}([Rd]_{yz}, [Rd]_{xz}) \end{bmatrix}$$

where $\text{atan2}$ is the two-argument arctangent (A.19).

4.3 Model Specification

4.3.1 Sensor Model

Vision is a relatively complex sensing modality in which three-dimensional objects are projected onto a two-dimensional image, a process known as image formation. In order to define a sensor model for visual coverage, a model of this process is required. Physically accurate models of image formation can be excessively complex, so a common geometric model with a manageable set of parameters is used, which is sufficient for the vast majority of cases. Ma et al. [110] discuss this reduction and its justification in the context of computer vision applications.

The ideal projective model in Section B.1 describes the transformation from a spatial point $p_s \in \mathbb{R}^3$ onto its image, a point $q \in \mathbb{R}^2$ in the image plane, of the form $(q_u, q_v)$. Assuming that $s_u$ and $s_v$, the
effective horizontal and vertical dimensions of pixels on the physical sensor, respectively, are known, \( q \) is given by (B.3) as

\[
q = \begin{bmatrix} \frac{f}{s_u} p_x + r_u + \frac{f}{s_v} p_y + r_v \end{bmatrix}^T
\]

where \( f \) is the effective focal length, and \( r = (r_u, r_v) \) is the image of the principal point in pixel coordinates.

Since the sensor obviously has a finite size, the actual image extends over a rectangular subset of the image plane, from the origin to pixel coordinates \((w, h)\), where \( w \) and \( h \) are the width and height\(^1\) of the sensor, respectively.

In practice, the camera aperture has a finite size, and the optics introduce effects of distortion and blur, described in Section B.2. After calibration and correction, lens distortion effects are negligible with the majority of modern optics used in computer vision; the effects of distortion are, therefore, not directly considered in modeling coverage. Conversely, blur in the image is, in general, neither negligible nor readily corrected, and can have a great deal of impact on task performance. In addition to the effective focal length, the effective aperture diameter \( A \) and the subject distance \( z_S \) are required to model blur.

The preceding parameters all describe properties endemic to the sensor and optical system, and are thus the intrinsic parameters of the camera. As discussed in general terms in Section 2.2.1, the extrinsic parameters describing the camera’s pose are also required to describe coverage. There are several possible representations, with varying numbers of parameters (see Section A.1); as they are equivalent, the pose will simply be referred to as a single “parameter” \( P \) having six degrees of freedom.

<table>
<thead>
<tr>
<th>Parameter(s)</th>
<th>Unit</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>( w, h )</td>
<td>P</td>
<td>width and height of the image sensor</td>
</tr>
<tr>
<td>( s_u, s_v )</td>
<td>L</td>
<td>physical dimensions of a pixel on the image sensor</td>
</tr>
<tr>
<td>( f )</td>
<td>L</td>
<td>effective focal length</td>
</tr>
<tr>
<td>( r = (r_u, r_v) )</td>
<td>P</td>
<td>image coordinates of the principal point</td>
</tr>
<tr>
<td>( A )</td>
<td>L</td>
<td>effective aperture diameter</td>
</tr>
<tr>
<td>( z_S )</td>
<td>L</td>
<td>subject distance</td>
</tr>
<tr>
<td>( P = (T, R) )</td>
<td>L, A</td>
<td>global pose of the camera (6 degrees of freedom)</td>
</tr>
</tbody>
</table>

**Obtaining the Camera Parameters**

The properties of the digital camera sensor itself are generally specified by the manufacturer. The width \( w \) and height \( h \) of the sensor are always available\(^2\) or can trivially be determined from a captured image. The physical pixel size \( s_u \times s_v \) is normally also specified, and often \( s_u = s_v \) (for square pixels);

\(^1\)Sometimes called the resolution of the sensor; though obviously related, not to be confused with the notion of resolution as a quantifier of precision in imaging objects.

\(^2\)Many devices allow specification of a subset of the physical sensor grid, termed a region of interest (ROI), for image capture. This analysis generalizes trivially to any constant ROI.
4.3. Model Specification

in cases where these values are unreliable or unavailable, they can be estimated as part of internal camera calibration.

The properties of the lens are difficult to specify with precision, particularly since it is normally manufactured separately from the sensor and is often manually adjustable. The effective focal length $f$ and principal point $o$ are typically determined via internal camera calibration (see Section B.3). The effective aperture diameter $A$ is somewhat more difficult to determine reliably; normally, it is necessary to rely on the specification or marking of a so-called f-number expressing the ratio of $f$ to $A$, which may not be precise. Once the optical system is calibrated, the subject distance $z_s$ may be obtained by measuring a known length in the plane at the subject distance; unfortunately, this too is imprecise as it relies on an image-based estimation of focus.$^3$

Numerous methods exist for estimating the pose of one or more cameras [110]. In general, this process normally involves finding a least squares solution to an overdetermined system obtained from a set of correspondences of image points to mutually non-coplanar object points with known relative geometry, as described in Section B.3. Some calibration methods estimate the intrinsic and extrinsic parameters simultaneously, normally in the form of a $3 \times 4$ camera matrix describing the projective mapping of the pinhole camera model; it is important to note that although the pose parameters may take any serviceable form, the intrinsic parameters are individually required to parameterize the sensor model.

4.3.2 Environment Model

The environment affects vision in two principal capacities: illumination and occlusion. The former is a complex phenomenon, involving the positions and properties of light sources as well as reflectance and other properties of materials in the environment, and furthermore, is coupled with the camera properties. In the basic visual coverage model, as with most traditional view planning models [27], it is assumed that illumination appropriate for the task is present.$^4$ Therefore, the environment model’s sole purpose is to describe the geometry of opaque bodies with the capacity to occlude the cameras’ rays of sight.

The applications addressed by this coverage model to date, including the examples presented in Part II of this thesis, are largely confined to relatively controlled environments, in which the geometry of scene objects are normally considered to be known. Therefore, the environment model is specified for deterministic occlusion. In principle, nothing beyond a redefinition of the criterion in Section 4.4.5 prevents the use of more general random occlusion, using e.g. a probabilistic occupancy grid as an environment model, but this type of occlusion is not yet well-studied [43] and is outside the scope of this work.

Three-dimensional surfaces may be represented to an arbitrary degree of precision by triangles, and ray-triangle intersection is a well-studied geometric operation by virtue of its importance in com-

$^3$An attempt was made to calibrate the depth of field in the absence of the problematic $A$ and $z_s$ parameters, using the Tenengrad criterion [111] against a well-behaved target at a fixed series of depths. Unfortunately, the results were found to be insufficiently reliable in practice, and this means of calibration remains an open problem.

$^4$The active triangulation model variant in Chapter 7 presents an example where illumination—in this case, a line laser source—is explicitly incorporated into the model, as its effects are well-defined, and the assumption of appropriate illumination is not justifiable.
puter graphics. Therefore, a simple and general form for the geometric environment model is a set of triangles $\mathcal{T}$.

**Obtaining the Environment Model**

The familiar way to obtain a model of this form in practice is to extract it from one or more CAD models of the scene. Many CAD and 3D modeling tools have the capability of triangulating polyhedral and non-polyhedral parametric solids to a specified granularity, and exporting this representation to some standard file format.

**4.3.3 Task Model**

The traditional field of view requirement is simply that the set of points to be imaged actually project within the finite physical sensor in the image plane. However, points near the edge of the image may not be covered, in the context of a particular task, for any of a number of reasons, including:

- Local feature detectors (e.g. Harris [112], SIFT [92], SURF [113]) require at least a small neighbourhood around the point feature to operate.
- Residual artifacts from lens distortion correction and other optical aberrations, typically most pronounced at the extremes of the image, may be unacceptable for certain classes of task.
- In such applications as surveillance and tracking, the quality of a view often depends directly on the proximity of a target’s image to the image center [50].

For this purpose, a parameter $\gamma$ is used, specifying the width, in pixels, of a border region around the inside edge of the image. Valid values range from 0 to half the lesser dimension of the image. As will be seen in Section 4.4.1, this is used by the field of view criterion to taper the coverage of points projecting to this region.

The other four task requirements—minimum and maximum resolution, maximum blur circle diameter, and maximum view angle—are directly related to the requirements of image processing algorithms. They are divided into acceptable and ideal values, which respectively enforce hard and soft constraints on the criteria, and are indicated by subscripts $a$ and $i$ on the parameters. The coverage model presented in this chapter specifically uses them for linear tapering of the bounded coverage values for the individual criteria, as will be formalized in Section 4.4.

Virtually all vision tasks depend, to some extent, on the resolution of the object being imaged. Most carry a minimum resolution requirement and/or favor higher resolution, hence the parameters $R_{na}$ and $R_{ni}$, respectively, the minimum acceptable and ideal resolution. Some tasks also place an upper limit on resolution and/or favor lower resolution, e.g. for privacy control in surveillance networks, hence the parameters $R_{xa}$ and $R_{xi}$, respectively, the maximum acceptable and ideal resolution. These four parameters range freely in $\mathbb{R}^+$, subject to relative constraints listed in Table 4.2. Since these parameters are expressed in units of distance per pixel, it is important to keep in mind that lower values correspond to higher resolution; though the convention is somewhat counterintuitive in this context, it is best aligned with common usage.
4.3. Model Specification

Visual tasks invariably depend on the focus of the image. Though the sensitivity to blur varies widely, explicit specifications are rare. Nevertheless, a focus requirement is quantified by a maximum blur circle diameter, hence the parameters $c_a$ and $c_i$. Measured in pixels, $c = 1$ is effectively perfect focus, since nothing smaller can be resolved; this, therefore, is the minimum value for these parameters.

The view angle to a feature can be a major factor in coverage. The maximum view angle parameters, $\zeta_a$ and $\zeta_i$, describe the extent to which the task can perform with the feature’s surface normal rotated with respect to the ray to the optical center. As a surface rotates beyond $\pi/2$, it suffers self-occlusion; this is therefore the maximum value for these parameters.

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Unit</th>
<th>Constraint</th>
<th>Default</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>$\gamma$</td>
<td>P</td>
<td>$0 \leq \gamma \leq \min(w, h)/2$</td>
<td>$\gamma = 0$</td>
<td>image boundary padding</td>
</tr>
<tr>
<td>$R_{na}, R_{ni}$</td>
<td>L/P</td>
<td>$R_{ni} \leq R_{ni} \leq R_{na}$</td>
<td>$R_{ni} = R_{na} = \infty$</td>
<td>minimum resolution</td>
</tr>
<tr>
<td>$R_{xa}, R_{xi}$</td>
<td>L/P</td>
<td>$R_{ni} \geq R_{x} \geq R_{xa} \geq 0$</td>
<td>$R_{xi} = R_{xa} = 0$</td>
<td>maximum resolution</td>
</tr>
<tr>
<td>$c_a, c_i$</td>
<td>P</td>
<td>$1 \leq c_i \leq c_a$</td>
<td>$c_i = c_a = \infty$</td>
<td>maximum blur circle diameter</td>
</tr>
<tr>
<td>$\zeta_a, \zeta_i$</td>
<td>A</td>
<td>$0 \leq \zeta_i \leq \zeta_a \leq \pi/2$</td>
<td>$\zeta_i = \zeta_a = \pi/2$</td>
<td>maximum view angle</td>
</tr>
</tbody>
</table>

Obtaining the Relevance Function and Task Requirements

As discussed in Section 2.2.1, the relevance function can be either concrete or abstract. The nature of the relevance function depends on the nature of the application.

Tasks requiring visual coverage of a specific target induce concrete relevance functions; these tend to arise in surveillance and monitoring applications where a particular individual or object is tracked in a closed loop, and in inspection, reconstruction, and exploration applications where the target geometry is known a priori or modeled simultaneously. These relevance functions are typically related to the actual surface geometry of the target, and may be dynamic in pose and configuration.

Tasks specifying a hypothetical range of possible target stimuli induce abstract relevance functions; these are the hallmark of generalized view planning for applications with future unknown targets. Such relevance functions are typically static and delimit large “zones” of coverage, possibly with priority.

Task requirements arise from the low-level image processing algorithm(s) which comprise the task. In the best cases, these are documented or derived analytically, as is often the case for the resolution parameters. The boundary padding for the field of view and the view angle parameters can, for many well-known algorithms, be derived from empirical studies of their behaviour. The focus parameters appear to be the least frequently specified or studied; as with any of the other requirements, direct empirical testing can yield good values. It should be noted that in the absence of these parameters, the default values in Table 4.2 are maximally permissive, so not all need be specified. If multiple low-level algorithms with differing requirements are employed serially in the task, the most restrictive value for each should be used.

---

5In Adolphus, the Task object subclasses Posable, so that its relevance function can be manipulated geometrically using the same facilities available to physical objects.
4.4 Monocular Visual Coverage Function

Each individual camera is modeled with a coverage function defined over its local frame of $\mathbb{D}^3$, based on the criteria discussed in Section 3.2.1. Assuming that the camera’s optical center is positioned at the origin and its optical axis is along the positive $z$-axis, as in Figure B.1, its coverage function is

$$C(p) = C^v(p) \cdot C^h(p) \cdot C^e(p) \cdot C^A(p) \cdot C^O(p),$$

(4.3)

where the five individual component functions are defined in the subsections to follow. The multiplicative combination (2.3) is used, reflecting the fact that, in practice, most computer vision tasks compound performance degradation as the inputs move farther from the optimal ranges of these criteria.

In defining these component functions, a bounding function is used to limit their values to the range $[0, 1]$, simplifying the formulation. This function is defined as

$$B_{[0,1]}(x) = \min(\max(x, 0), 1).$$

(4.4)

4.4.1 Field of View

In the absence of occlusion, the visibility of a point feature depends on whether it is within the field of view of the camera. As seen in Section 3.2, this is a nearly universal criterion of visual coverage.

A novel feature of this model is the consideration of edge effects, discussed in Section 4.3.3. The coverage near the inside surface of the field of view pyramid, corresponding to the border region induced by the task parameter $\gamma$, is tapered linearly to zero.

Assuming that $\gamma > 0$, the horizontal and vertical field of view criteria are given, respectively, by

$$C^{vh}(p) = B_{[0,1]} \left( \frac{1}{\gamma_h} \min \left( \frac{p_x}{p_z} + \tan \alpha_l, \tan \alpha_r - \frac{p_x}{p_z} \right) \right)$$

(4.5)

and

$$C^{vv}(p) = B_{[0,1]} \left( \frac{1}{\gamma_v} \min \left( \frac{p_y}{p_z} + \tan \alpha_l, \tan \alpha_b - \frac{p_y}{p_z} \right) \right)$$

(4.6)

where $\alpha_l$ and $\alpha_r$ are the horizontal field of view angles, and $\alpha_v$ and $\alpha_b$ are the vertical field-of-view angles, as given by (B.4) through (B.7), respectively, and $\gamma_h$ and $\gamma_v$ are given by

$$\gamma_h = \frac{\gamma}{w} (\tan \alpha_l + \tan \alpha_r)$$

(4.7)

$$\gamma_v = \frac{\gamma}{h} (\tan \alpha_v + \tan \alpha_b)$$

(4.8)

In the case where $\gamma = 0$, the horizontal and vertical field of view criteria are considerably simpler, easily expressed by the bivalent indicators

$$C^{vh}(p) = \begin{cases} 
1 & \text{if } -\tan \alpha_l < \frac{p_x}{p_z} < \tan \alpha_r, \\
0 & \text{otherwise}, 
\end{cases}$$

(4.9)
4.4. Monocular Visual Coverage Function

and

\[ C^V_v(p) = \begin{cases} 1 & \text{if } -\tan \alpha_l < \frac{p_x}{p_z} < \tan \alpha_r, \\ 0 & \text{otherwise}, \end{cases} \quad (4.10) \]

For either case, the combined field of view criterion is given by

\[ C^V(p) = \begin{cases} \min(C^V_h(p), C^V_v(p)) & \text{if } p_z > 0, \\ 0 & \text{otherwise.} \end{cases} \quad (4.11) \]

4.4.2 Resolution

Given a resolution \( R \) in units of distance per pixel, the depth at which that resolution occurs in the image, along an arbitrary direction, is

\[ z(R) = R \min \left( \frac{w}{\tan \alpha_l + \tan \alpha_r}, \frac{h}{\tan \alpha_l + \tan \alpha_r} \right). \quad (4.12) \]

Note in (4.12) that \( w \) and \( h \) are in pixels, so the result \( z(R) \) is in units of distance.

With \( R_{nl} < R_{na} \), the minimum resolution criterion is given by

\[ C^{Rn}(p) = B_{[0,1]} \left( \frac{z(R_{na}) - p_x}{z(R_{na}) - z(R_{ni})} \right). \quad (4.13) \]

In the case where \( R_{nl} = R_{na} \), the minimum resolution criterion is expressed by the bivalent indicator

\[ C^{Rn}(p) = \begin{cases} 1 & \text{if } p_x < z(R_{na}), \\ 0 & \text{otherwise.} \end{cases} \quad (4.14) \]

Similarly, the maximum resolution criterion for the cases where \( R_{xl} > R_{xa} \) and \( R_{xl} = R_{xa} \), respectively, are given by

\[ C^{Rx}(p) = B_{[0,1]} \left( \frac{p_x - z(R_{na})}{z(R_{xl}) - z(R_{xa})} \right) \quad (4.15) \]

and

\[ C^{Rx}(p) = \begin{cases} 1 & \text{if } p_x > z(R_{xa}), \\ 0 & \text{otherwise.} \end{cases} \quad (4.16) \]

For any of the four possible cases, the combined resolution criterion is given by

\[ C^R(p) = C^{Rn}(p) \cdot C^{Rx}(p). \quad (4.17) \]

4.4.3 Focus

Solving for \( p_x \) in (B.10) yields the two depths at which a point in the scene maps to a blur circle of diameter \( c \) in the image,

\[ z(c) = \frac{Afz_S}{Af \pm c \min(s_u, s_v)(z_S - f)}. \quad (4.18) \]
with the slight change that $c$ is here expressed in pixels, and must be multiplied by the lesser of the physical pixel dimensions to convert it to units of distance.

Let $z_n(c)$ and $z_f(c)$ represent the lesser (with $\pm$ evaluated as $+$) and greater (with $\pm$ evaluated as $-$) of these values, respectively. With $c_i < c_a$, the focus criterion is given by

$$C^F(p) = B[0,1] \left( \frac{p_x - z_n(c_a)}{z_n(c_i) - z_n(c_a)} \right) \cdot B[0,1] \left( \frac{z_f(c_a) - p_x}{z_f(c_a) - z_f(c_i)} \right).$$

(4.19)

In the case where $c_i = c_a$, the focus criterion is expressed by the bivalent indicator

$$C^F(p) = \begin{cases} 1 & \text{if } z_n(c_a) \leq p_x \leq z_f(c_a), \\ 0 & \text{otherwise}. \end{cases}$$

(4.20)

### 4.4.4 View Angle

The view angle to a point feature is defined in this model as the angle between the surface normal at the feature and the ray between the feature and the camera’s optical center. The surface normal vector is obtained from $p_\rho$ and $p_\eta$ of directional point $p$, and the view angle is given by

$$\zeta(p) = \cos^{-1} \left( \begin{pmatrix} \sin p_\rho \cos p_\eta, \sin p_\rho \sin p_\eta, \cos p_\eta \end{pmatrix} \cdot \overrightarrow{p_z} \right).$$

(4.21)

For $\zeta_i > \zeta_a$, the view angle criterion is given by

$$C^A(p) = B[0,1] \left( \frac{\cos \zeta(p) - \cos \zeta_a}{\cos \zeta_i - \cos \zeta_a} \right).$$

(4.22)

noting, for implementation purposes, that the cosine on $\zeta(p)$ cancels with the inverse cosine of (4.21).

In the case where $\zeta_i = \zeta_a$, the view angle criterion is expressed by the bivalent indicator

$$C^A(p) = \begin{cases} 1 & \text{if } \zeta(p) \leq \zeta_a, \\ 0 & \text{otherwise}. \end{cases}$$

(4.23)

### 4.4.5 Deterministic Occlusion

The criterion makes use of a function intersect($\Delta, p, q$), which returns the point of intersection between line segment $\overrightarrow{pq}$ and triangle $\Delta$, or $\emptyset$ if none exists. Möller and Trumbore [114] present an efficient means of computing this intersection.

Given the set of triangles $\mathcal{T}$ comprising the opaque surfaces in the environment, and recalling that $P$ is the pose of the camera and $T$ is the position of its optical center, the deterministic occlusion criterion is given by

$$C^O(p) = \begin{cases} 1 & \text{if intersect}(\Delta, P(p), T) \in \{\emptyset, p\} \forall \Delta \in \mathcal{T}, \\ 0 & \text{otherwise}. \end{cases}$$

(4.24)

In (4.24), it is necessary to transform $p$ to the global coordinate frame for the intersect function, because the triangles in $\mathcal{T}$ are specified globally.
4.5 Multi-Camera Coverage

The coverage function defined by (4.3) takes a directional point \( \mathbf{p} \) as its argument, where \( \mathbf{p} \) is in the local coordinate frame of the camera. Therefore, the mutual coverage function \( C_M \) for a set of \( k \) cameras \( M \) is as given by (2.6), and the coverage function \( C_N^k \) of the multi-camera system comprising a set of cameras \( N \) is as given by (2.7).

It should be noted that this \( k \)-coverage function does not apply to stereo vision, bistatic active vision, or other multi-camera modalities which have additional performance criteria beyond mutual information coverage (e.g. baseline, in the case of stereo vision). Such criteria at the sensor system level, which Mittal and Davis [43] refer to as “algorithmic constraints,” necessitate, at least, additional task requirement parameters and a redefinition of the sensor system level coverage function.

4.6 Task-Oriented Coverage Evaluation

For low-level computer vision tasks operating on point features, the coverage performance metric defined by (2.8) applies. This carries with it two implicit assumptions: that point features are the highest-level atomic data, and that such features are representable, even in the abstract, as directional points. Many applications break one or both of these assumptions, but the following two sections explain how to reconcile such cases with the existing stimulus space and coverage model.

4.6.1 Coverage of Complex Features

Many higher-level computer vision tasks operate on complex features composed of multiple point features. Complex features are atomic (operated on as a whole), so the performance of tasks operating on them are bound to the worst coverage across all constituent stimulus points. Modeled as a concrete relevance function, the coverage performance metric described in Section 2.2.3 does not capture this, as it implicitly assumes that no larger structure than individual stimuli are used by the task. Strictly following the concepts of Chapter 2, each complex feature class would induce its own stimulus space, but each type of feature—as varied as the tasks that operate on them—would then require its own (much more complex) coverage model, a patently impractical proposition.

Instead, it is possible to continue using the directional space and the visual coverage model presented in this chapter for such tasks, simply by modifying the task performance metric. Assuming that, because all points are therefore equally important to the task, the relevance function \( R \) is bivalent, the coverage performance is

\[
F(C,R) = \min_{\mathbf{p} \in \langle R \rangle} C(\mathbf{p}). \tag{4.25}
\]

Mathematically, this is a simple equivalent to defining a stimulus space and coverage model for complex features composed of the points in \( \langle R \rangle \), and evaluating (2.8) on a single stimulus point. It does not, therefore, address the issue of abstract relevance functions over complex features; clearly, an abstract \( R \) would not itself represent the atomic complex feature, and (4.25) would lose its meaning.

The extent to which the complex features are localized near a point in the stimulus space determines how closely the point feature coverage model approximates the actual task. One simple example is mentioned in Section 4.3.3: local feature detectors require a small neighbourhood of stimuli around
the immediate feature, and the \( \gamma \) task requirement parameter provides a reasonable means of partially modeling this. Zhao et al. [16, 17, 18] provide another example: their task operates atomically on name tags, which occupy a non-singular portion of the stimulus space, yet are generally small enough in the field of view that, for most of the coverage function definition, they are assumed to be point features.

4.6.2 Invariance to View Angle

The view angle criterion is unique in enforcing a constraint, even in the default (most permissive) settings of \( \zeta_a \) and \( \zeta_t \), which a task’s relevance function—particularly an abstract one—might sensibly wish to ignore. Many tasks, at least in their simplified representations, allow for the coverage of point features from any view angle. This could be addressed without modifying the coverage model, simply by relaxing the upper bounds on \( \zeta_a \) and \( \zeta_t \) and assigning the task points in question an arbitrary direction. However, such an approach is unsatisfying for two reasons: first, the arbitrary assignment of direction is not intuitive, and second, while \( \zeta_t = \zeta_a = \pi \) has a concrete meaning (invariance to view angle), \( \pi/2 < \zeta_t < \pi \) and \( \pi/2 < \zeta_a < \pi \) do not.

Rather than redefining a separate “non-directional” stimulus space (i.e. \( S = \mathbb{R}^3 \)) and associated coverage model, a simple addendum to the existing definition suffices to allow computation of \( C(p) \) with \( p \in \mathbb{D}^3 \) or \( p \in \mathbb{R}^3 \). Note that, according to (4.22), the suggested \( \zeta_t = \zeta_a = \pi \) above is effectively stating that \( C^A(p) = 1 \) for all \( p \in \mathbb{D}^3 \). Note also that (4.11), (4.17), (4.19), and (4.24) are invariant to view angle, and thus depend only on \( p_x \). Therefore, for \( p \in \mathbb{R}^3 \), \( C^A(p) = 1 \) by definition, and the remaining criteria are evaluated as written.

4.7 Tractability and Computational Complexity

The fidelity and generality of the coverage model presented in this chapter is validated and compared to the state of the art in Chapter 5. However, its tractability for optimization techniques, though addressed specifically for three problem cases in Part II of this thesis, warrants some general comments.

4.7.1 Metaheuristic Optimization

In order to define a coverage model of practical fidelity in the general case, little can be assumed about the coverage function beyond its basic definition (see Definition 1), and accordingly, the same may be said of the task-oriented coverage performance metric defined by (2.8). It is not generally possible to state a visual coverage problem of the classes described in Section 1.2 as a differentiable objective function over a continuous solution space. Therefore, neither finitely terminating algorithms nor convergent iterative methods apply, leaving only metaheuristics. While the latter class of method is able to iteratively improve the solution over any “black-box” objective, it carries two disadvantages: it does not guarantee convergence to an optimal solution, and is typically much more computationally expensive.

Fortunately, the nature of visual coverage problems and the coverage function framework is such that most problem formulations lend themselves particularly well to such methods, and can satisfactorily mitigate these disadvantages. First, the performance metric is bounded in \([0, 1]\), so it is possible to
quantify absolutely the degree to which a direct continuous optimization over (2.8) or derived function has converged. In many cases, there exists a set of solutions yielding a value of 1, and it suffices for convergence to find any one such solution. Second, many problems can be reduced to a combinatorial formulation, as the set of views, admissible camera poses, number of cameras, total system cost, and other variables often induce a discrete solution space. In some cases, topological reductions provide sufficient information. Finally, the nature of most visual coverage problems is such that fairly tight bounds can be established on the solution space given a cursory manual inspection of the problem instance, often substantially reducing computation time.

The form of the coverage model presented in this chapter does, however, present one particular challenge to the continuous case. The bivalence of the deterministic occlusion criterion defined by (4.24), coupled with the arbitrary nature of \( \mathcal{T} \), produces, in the majority of applications, a large number of discontinuities in the objective function, however derived. This can also be the case to some extent with the other four criteria, depending on how the task parameters are valued. These create many local minima, to which metaheuristic search methods are sensitive.

### 4.7.2 Complexity of Performance Metric Evaluation

The application of metaheuristic optimization techniques implies that, in some sense, (2.8) must be evaluated repeatedly. It is therefore important to understand the computational complexity of evaluating this function for the coverage model presented in this chapter.

The camera coverage function \( C_i \) must be evaluated for each camera \( i \in N \). The critical component is \( C^D \) (4.24): for each triangle in \( \mathcal{T} \), the intersect function must be computed. Some of \( \mathcal{T} \) may be culled initially for each change in \( C_i \) for a particular camera, but in general, the entire set are candidates for occlusion. Therefore, the computational complexity is \( \mathcal{O}(|N| \times |\mathcal{R}| \times |\mathcal{T}|) \).

In practice, the evaluation of (2.8) is highly parallelizable in general, and specifically, ray-triangle intersection and frustum culling, both used in evaluating (4.24), are computations to which GPU hardware is ideally suited.
We wish to pursue the truth no matter where it leads. But to find the truth, we need imagination and skepticism both.

Carl Sagan (1934–1996), *Cosmos*

5.1 Overview

This chapter presents experimental validation of the coverage model for monocular multi-camera vision proposed in Chapter 4. First, the relationship between actual task performance and the individual visual criteria is verified, using a controlled data set isolating the effects of the criteria. The results demonstrate clear correlation between predicted and actual performance using an entirely a priori specification of the task. Then, the model is compared in an experimental application against two other state of the art models from the literature, drawn from Chapter 3. Favorable results demonstrate the relative flexibility and fidelity of the model.

5.2 Validation of Monocular Criteria

The HALCON [115] computer vision software library is capable of estimating, to a relatively high degree of accuracy, the three-dimensional pose of a structured calibration plate shown in Figure 5.1 from a single image, given its geometric description. This function has well-defined requirements in each of the criteria observed by the sensor model of Section 4.4, and moreover, its parameters allow for some control over the sensitivity. These properties make it a good representative stand-in task for validation. In these experiments, a 70mm plate is used.

The relevance function for the calibration plate is defined as $R(p) = 1$ for each of 50 discrete points $p \in \mathbb{D}^3$, all with direction perpendicular to its plane: one point at the center of each of the 49 dots, and one point at the center of the triangular corner reference mark, comprising the information required by the function. Table 5.1 lists the task requirements, derived from the documented requirements of the functions involved in the pose estimation process.
5.2. Validation of Monocular Criteria

Figure 5.1: HALCON Calibration Plate – HALCON [115] is capable of estimating the three-dimensional pose of the plate from a single image, given its geometric description. The experiments in this chapter use a 70mm plate and a 200mm plate, respectively.

Table 5.1: Requirements for Target Pose Estimation Task

<table>
<thead>
<tr>
<th>Parameter</th>
<th>$\gamma$</th>
<th>$R_{na}$</th>
<th>$R_{ni}$</th>
<th>$R_{xa}$</th>
<th>$R_{xi}$</th>
<th>$c_a$</th>
<th>$c_i$</th>
<th>$\zeta_a$</th>
<th>$\zeta_i$</th>
</tr>
</thead>
<tbody>
<tr>
<td>Value</td>
<td>25</td>
<td>1.4583</td>
<td>0.0625</td>
<td>0.0625</td>
<td>0.04375</td>
<td>15.3184</td>
<td>1.0</td>
<td>1.2217</td>
<td>0.0</td>
</tr>
</tbody>
</table>

The ground truth pose of the plate, necessary both for transformation of $R$ to the global frame and for computing actual pose error in some of the experiments, is obtained by positioning the plate using a Mitsubishi RV-1A six-axis manipulator arm, whose coordinate system is the global frame. The camera, a NET iCube NS4133BU with a Computar H10Z1218-MP lens, is externally calibrated within this frame.

5.2.1 Field of View

The calibration plate is oriented perpendicular to the camera’s optical axis, and translated within its plane to 129 positions fully within, partially within, and fully outside the field of view. The ground truth pose of the plate is obtained from the robotic arm, and $F(C, R)$ is evaluated according to (4.25), with $R$ transformed to this pose and (effectively) $C(\mathbf{p}) = C^V(\mathbf{p})$. The HALCON pose estimation algorithm is executed on each image, and the presence or absence of a valid pose estimate is recorded. Figure 5.2 shows the excellent receiver operating characteristic of $F(C, R)$ with respect to actual performance, with a binary classifier threshold varied from 0 to 1. Residual false positives and false negatives are likely attributable to error in the external calibration of the camera, as the criterion itself is theoretically sound to the extent that the camera model reflects reality.

5.2.2 Resolution and Focus

The calibration plate is translated perpendicular along the camera’s optical axis to 57 positions between 1035mm and 1250mm in depth relative to the camera frame. All $\mathbf{p} \in (R)$ are within the field of view and unoccluded. The ground truth pose of the plate is obtained from the robotic arm, and $F(C, R)$ is evaluated according to (4.25), with $R$ transformed to this pose and (effectively) $C(\mathbf{p}) = C^R(\mathbf{p}) \cdot C^R(\mathbf{p})$. The HALCON pose estimation algorithm is executed on each image, and the position error is computed as the Euclidean distance between the estimate and ground truth.

The Pearson product-moment correlation coefficient between $F(C, R)$ and the position error is $r = -0.9476$, indicating a strong correlation as seen in Figure 5.3, and thus the predictive power of the resolution and focus components of the model.
5.2. Validation of Monocular Criteria

Figure 5.2: Validation of Field of View Criterion – $F(C, R)$, as calculated by (4.25) with field of view only, exhibits an excellent receiver operating characteristic in predicting the ability of HALCON to generate a valid pose estimate.

Figure 5.3: Validation of Resolution and Focus Criteria – The results exhibit a clear correlation between $F(C, R)$, as calculated by (4.25) with resolution and focus only, and the position error.

5.2.3 View Angle

The calibration plate is rotated, with its center at $z_5$ on the optical axis, through 146 angles. All $p \in \langle R \rangle$ are within the field of view and unoccluded. The ground truth pose of the plate is obtained from the robotic arm, and $F(C, R)$ is evaluated according to (4.25), with $R$ transformed to this pose and (effectively) $C(p) = C^A(p)$. The HALCON pose estimation algorithm is executed on each image, and the rotation error is computed as the angle between the estimate and ground truth.
Figure 5.4: Validation of View Angle Criterion – The results exhibit a clear correlation between $F(C, R)$, as calculated by (4.25) with view angle only, and the cosine of the rotation error.

The Pearson product-moment correlation coefficient between $F(C, R)$ and the cosine of the rotation error (where a greater value of the cosine indicates lower error) is $r = 0.9811$, indicating a strong correlation as seen in Figure 5.4, and thus the predictive power of the view angle component of the model.

5.3 Comparison with Other Models

5.3.1 Selected Model Definitions

Two models are selected from those reviewed in Section 3.2 for comparison. To select one representative model from each the two-dimensional and three-dimensional cases, the following criteria are used:

1. The publication presents details of the coverage criteria sufficient to develop an implementation reasonably close to the original.

2. The coverage model is representative of the state of the art within its application scope.

3. Task assumptions are sufficiently realistic to allow modeling of the common experimental application.

Wherever implementation details are unspecified or unclear, the default is to give the benefit of the doubt and use the equivalent component from the proposed model. A significant example is in the use of the task-oriented coverage framework presented in Sections 2.2.3 and 4.6, where neither of the selected models offers a sufficient means of representing the task as for the proposed model.
5.3. Comparison with Other Models

Hörster and Lienhart

The two-dimensional coverage model proposed by Hörster and Lienhart [14, 15] observes three criteria with bivalent indicator functions: field of view, resolution, and deterministic occlusion. Thus, it can be represented by a coverage function of the form

\[ C(\mathbf{p}) = C^V(\mathbf{p}) \cdot C^{rn}(\mathbf{p}) \cdot C^O(\mathbf{p}). \] (5.1)

The field of view criterion is modeled by a triangle with origin at the optical center and apex angle \( \alpha_h \), and is given by

\[ C^V(\mathbf{p}) = \begin{cases} 1 & \text{if } |p_x| < \frac{p_z}{2} \tan \alpha_h \\ 0 & \text{otherwise,} \end{cases} \] (5.2)

where \( \alpha_h = \alpha_l + \alpha_r \), from (B.4) and (B.5). Note that the absolute value in the conditional implicitly enforces \( p_z > 0 \).

The resolution criterion is equivalent to \( C^{rn} \) as defined by (4.14). The deterministic occlusion criterion is equivalent to \( C^O \) as defined by (4.24), with the condition that every triangle in \( \mathcal{T} \) lies in a plane perpendicular to the \( x\)-\( y \) plane and extends infinitely in the \( z \) direction (its base, therefore, being equivalent to a two-dimensional line segment in the \( x\)-\( y \) plane).

Zhao et al.

The three-dimensional coverage model proposed by Zhao et al. [16, 17, 18] observes four criteria with bivalent indicator functions: field of view, self-occlusion, deterministic occlusion, and a combined resolution and view angle “projected length” criterion. Thus, it can be represented by a coverage function of the form

\[ C(\mathbf{p}) = C^V(\mathbf{p}) \cdot C^L(\mathbf{p}) \cdot C^A(\mathbf{p}) \cdot C^O(\mathbf{p}). \] (5.3)

The field of view criterion is enforced by direct verification that \( \mathbf{p} \) projects onto the image plane. Though not explicitly specified, the authors imply they use essentially the same model of image projection presented in Section B.1, so their criterion is equivalent to \( C^V \) as defined by (4.11), with \( \gamma = 0 \).

Both resolution and view angle are handled directly as a function of the projected length of the tag, under the assumption that the tag is oriented horizontally. Lifting this assumption, a more general equivalent criterion can be derived. Given the physical length of the tag \( l \) and the threshold for its minimum projected length \( T \), the maximum acceptable depth is

\[ z_L(l, T, \mathbf{p}) = z_R\left(\frac{l \cos \zeta(\mathbf{p})}{T}\right), \] (5.4)

where \( \zeta(\mathbf{p}) \) is as defined in (4.21). The bivalent projected length criterion is then given by

\[ C^L(\mathbf{p}) = \begin{cases} 1 & \text{if } p_z < z_L(l, T, \mathbf{p}) \\ 0 & \text{otherwise.} \end{cases} \] (5.5)

The actual view angle criterion is used strictly as a self-occlusion constraint, and is thus equivalent to \( C^A \) as defined by (4.22) with \( \zeta_l = \zeta_\alpha = \pi/2 \). The deterministic occlusion criterion is equivalent to
as defined by (4.24), with the condition that every triangle in \( T \) lines in a plane perpendicular to the \( x-y \) plane.

Due to the way the criteria are computed in the original source, the centroid of \( \langle R \rangle \) is constrained to lie in a plane parallel to the \( x-y \) plane.

### 5.3.2 System and Task Description

The experimental application is a surveillance network, with a relatively simple face detection task. The camera network consists of eight wall-mounted NET iCube NS4133BU cameras with NET SV-0813V lenses, each hosted by an ASUS Eee Box EB1007-B0410 PC running Linux. All cameras are internally calibrated, and the network is externally calibrated with respect to a common reference frame (see Section B.3.2 for a description of the latter procedure).

![Figure 5.5: Ground Truth Face Pose](image)

(a) With Mask  
(b) Without Mask

Figure 5.5: Ground Truth Face Pose – A reasonably accurate estimate of the pose of the face is obtained by placing a rig consisting of a mask and a 200mm HALCON calibration plate in one of two images with the face in the same pose.

The face detection process itself is carried out on single images using a Haar cascade [116]. Still images are captured simultaneously from all eight cameras; a total of 150 such images, with the subject’s face in a representative sample of poses within the covered area, are captured for the experiment. In order to determine the ground truth pose of the face—and thereby the “pose” of the relevance function—two sets of images are taken for each pose of the face; in one, the subject wears a rig consisting of a mask and a 200mm HALCON calibration plate, as shown in Figure 5.5, providing a reasonably accurate estimate.¹

### 5.3.3 Parameterization and Results

The implementation of the face detection algorithm is used as-is, with no documentation available regarding its requirements. Therefore, the task requirements, listed in Table 5.2, are determined experimentally by independent measurement of the resolution, blur, and view angle values delimiting performance degradation in a set of controlled training images.

For the model of Zhao et al., a physical feature length of 80mm is used for a face, with a minimum projected length criterion of 20 pixels corresponding to the empirical minimum acceptable resolution requirement. Based on the approximate height of the subject, the centroid of \( \langle R \rangle \) is constrained to the plane at \( z = 1500 \text{mm} \).

¹For image sets in which the pose of the HALCON calibration plate is available from more than one image, the image in which its projected area is the largest is chosen. Incidentally, such multiple pose estimates also serve to confirm the consistency of the external calibration.
5.3. Comparison with Other Models

Figure 5.6: Layout of Comparison Surveillance Network – Cameras are mounted at varying heights and orientations around the outer wall, and the room contains two occlusions which do not extend to its full height.

Table 5.2: Requirements for Face Detection Task

<table>
<thead>
<tr>
<th>Parameter</th>
<th>$\gamma$</th>
<th>$R_{na}$</th>
<th>$R_{ni}$</th>
<th>$R_{xa}$</th>
<th>$R_{xi}$</th>
<th>$c_a$</th>
<th>$c_i$</th>
<th>$\zeta_a$</th>
<th>$\zeta_i$</th>
</tr>
</thead>
<tbody>
<tr>
<td>Value</td>
<td>25</td>
<td>4.0</td>
<td>3.0</td>
<td>0</td>
<td>0</td>
<td>10.0</td>
<td>8.0</td>
<td>1.2</td>
<td>0.4</td>
</tr>
</tbody>
</table>

Let the entire set of 1200 images be represented by a set $I$ of ordered pairs of the form $(i, j)$, where $i \in \{1, 2, \ldots, 8\}$ is the camera node and $j \in \{1, 2, \ldots, 150\}$ is the frame. The subset of images $D \subset I$ in which a face is detected is obtained by running the algorithm on each image and visually verifying the accuracy of detection. The subset of images $P \subset I$ for which a model predicts positive face detection is obtained by evaluating the coverage performance according to (4.25), over coverage function $C_i$ for the corresponding individual camera, with relevance function $R$ transformed according to the ground
5.3. Comparison with Other Models

Figure 5.7: Adolphus Simulation of Comparison Surveillance Network – The same configuration is used for all three coverage models, with Model, Camera, Task objects replaced by subclasses implementing the appropriate model.

Figure 5.8: Vision Graph for Comparison Surveillance Network – A well-connected vision graph allows for accurate external calibration and target pose estimation.

truth face pose, and binarizing according to a threshold in $[0, 1]$. The true and false positive rates are then given by

$$TPR = \frac{|P \cap D|}{|D|} \quad (5.6)$$

and

$$FPR = \frac{|P \setminus D|}{|I \setminus D|} \quad (5.7)$$

respectively. Figure 5.9 plots the resulting receiver operating characteristic, varying the binarization threshold (for the proposed model only, since the others are binary by definition).

The model of Hörster and Lienhart performs relatively poorly, in part because the system configuration and task violate several of their assumptions. In particular, the cameras deviate significantly from their two-dimensional common plane projections, and the vertical occlusions do not extend to the full height of the working volume. However, the application is typical of the type supposedly approximated by this model (and two-dimensional models in general). This result clearly demonstrates the shortcomings of a two-dimensional formulation.
5.3. Comparison with Other Models

Figure 5.9: Comparison of Coverage Model Fidelity – The receiver operating characteristic for the face detection task shows clearly that the proposed model outperforms the comparison models.

By contrast, the application is one to which the model of Zhao et al. is ideally suited, according to their assumptions. Only vertical, rectangular occlusions occur, and the task is the detection of an essentially planar object with strong similarity to the tag identification application. Accordingly, this model performs relatively well. Still, the separation of the resolution and view angle criteria and the inclusion of a focus criterion allow the proposed model to outperform this model even within its own stated scope of application.
Part II

Optimization on Coverage
CHAPTER 6

Robust Real-Time View Selection

It’s a crystal. Nothing more. But if you turn it this way and look into it, it will show you your dreams.

Jareth, Labyrinth (1986)

6.1 Overview

In this chapter, the coverage model of Chapter 4 is applied to the view selection problem. A method for real-time selection of monocular view sequences for an arbitrary task in a calibrated multi-camera system is presented. Instantaneous view quality is quantified by the coverage model based on a priori information about the sensor system, environment, and task which are generally available in the relatively controlled environments of the target application class. The criterion of transition smoothness is investigated and integrated into the overall objective function for optimal view sequences. A scalable real-time algorithm with robust suboptimal performance is presented based on this objective function. Experimental results demonstrate the performance of the method, as well as its robustness to several identified sources of non-smoothness.

6.2 Introduction

A solution to the view selection problem must balance two potentially competing objectives: the selection of the best instantaneous view at any particular moment, and the overall smoothness of the view sequence over time. Instantaneous best view selection can be thought of as one of a family of coverage optimization problems which also includes such problem classes as view planning, camera reconfiguration, and next best view. The key component in all such problems is the coverage model. View selection is unique among the problems mentioned in that the search space for the coverage optimization problem is discrete and relatively small. However, extension into the temporal domain introduces some complexity of its own in the transition smoothness objective, particularly in the real-time case.
With a quantitative coverage metric in hand, the *instantaneous* view selection problem is reduced to selecting the view whose coverage function, supplied with the system state and a coverage objective, evaluates to the largest value. In the presentation of the algorithm in Section 6.4, the feasibility of exhaustive searches within local subsets of practical multi-camera systems is discussed, and a strategy is presented, under minimal reasonable assumptions, for scaling to arbitrarily large systems.

Finding an optimal view sequence over a set of discrete instants in time—referred to as frames—is more involved than simply selecting the best instantaneous view for each frame. It is also normally desirable to have a relatively smooth set of view transitions within the sequence, even at the expense of some instantaneous view quality, both for human observers and for machine processing. In the analysis and formal statement of the problem in Section 6.3, transition smoothness is defined quantitatively, and incorporated into the objective function for optimization of the view sequence. Here, the real-time problem is specifically addressed; that is, the view is optimized at each frame without access to future observations.

### 6.2.1 Prior Work

Park et al. [50] present a distributed lookup table system for real-time view selection in a smart camera network. While their focus is primarily on the distributed aspect, their work is interesting because they define a geometric coverage model. Although their model is a simplistic stand-in for a more sophisticated measure, the idea of mapping a stimulus space to a measure of coverage based on imaging theory is there. While such models are relatively common in other coverage optimization problems, they have not been used frequently in view selection. Soro and Heinzelman [52] and Shen et al. [51] also follow this general idea, although their models are arguably even more abstract.

In contrast, the majority of view selection methods in the literature use visual information directly to determine quality of view. This approach has advantages over a priori modeling in dealing with highly unstructured scenes and uncalibrated camera systems, but involves a large amount of uncertainty, almost certainly requires more energy and computational resources for the same application (assuming the a priori approach is possible at all), and lacks generality and theoretical rigor. Vázquez et al. [117] propose an information-theoretic measure they term viewpoint entropy. In a somewhat similar vein, Snidaro et al. [118] define an unbounded metric, termed appearance ratio, based on the information contained in a segmented blob. Gupta et al. [119] base their metric on visual analysis of occlusions and appearance ambiguity. Park et al. [120] base their event detection probability measure on expert knowledge, and prioritize events for view selection. Lee, Morbee, Tessens, et al. [121, 122, 123] focus on face detection as a criterion for view quality. Guan [124] also uses a measure of information obtained from face detection as a view quality metric.

Jiang et al. [22] do not explicitly define a view quality metric. They identify the issue of transition smoothness, and introduce a term to their objective function which, as used, adds a cost for transitioning from the current view. While this achieves essentially the same effect as the transition threshold \( \tau \) in the algorithm presented here, there is no explanation as to how this relates to an actual optimal view sequence—in fact, by the smoothness criterion defined by (6.3), it is suboptimal. Using a dynamic programming approach, they optimize the view sequence over a sliding window, which includes observations of a fixed number of future frames. This makes the method unsuitable for immediate real-time view selection; though it could conceivably perform view selection with fixed
latency, it is not clear from the empirical results whether practical window sizes would yield acceptable view sequences.

At the instantaneous level, Daniyal et al. [23] grade views based on visual information: total activity, event score, number of objects, and cumulative object score. They also identify the need for transition smoothness, and use a dynamic Bayesian network (DBN) approach to smooth the view sequence. While the approach appears to effectively reduce the density of transitions with respect to the baseline instantaneous approach, the duration of subsequences per view is not reported. Additionally, the training required for the DBN parameters, which is global over the sequence period, makes this method unsuitable for real-time view selection.

Erdem and Sclaroff [104] address the problem for hybrid networks including active (pan-tilt-zoom) cameras. A transition graph model is constructed dynamically from event observations, and this is transformed into an equivalent DBN to predict the subset of cameras and their parameters in real time. However, transition smoothness is not treated, and no explicit criterion for view quality is provided. Other research in active camera networks with application to surveillance tracking which touches on this problem includes the work of Cai and Aggarwal [125], Ng et al. [126], and Isler et al. [127].

6.2.2 Application Scope

Three restrictions are placed on the class of real-time view selection applications within the scope of this chapter. First, a relatively controlled environment, such as a robot work cell, is assumed, in which camera calibration is feasible, scene geometry is to some extent known, and so forth. This isolates the approach from any application-specific vision processes for obtaining this knowledge, and from probabilistic modeling concerns. Second, only the case of monocular views is considered, as fixed and variable $k$-view selection is a complex topic of its own. Third, it is assumed that the objective is to select a single view of a single target, whose pose and configuration may vary arbitrarily. It is desirable to eventually relax all three of these restrictions in further study on the topic.

6.3 Optimal View Sequence Selection

The instantaneous optimal monocular view selection problem is formally defined in terms of the coverage model as follows:

\[
\text{Given a multi-camera system } N, \text{ coverage functions } C_i \text{ for all } i \in N, \text{ and a relevance function } R, \text{ find } \arg \max_{i \in N} (F(C_i, R)).
\]

Over time, the problem becomes one of finding an optimal sequence of such views.

5 Definition (View Sequence)

A view sequence is a mapping $Q : (0, ..., t_{\text{max}}) \rightarrow N$ assigning a view in $N$ to each frame $t$. 

---

66
Without considering transition smoothness, the optimal view sequence $Q_{\text{best}}$ is found by simply solving the instantaneous problem at each frame. In general, over the period of a view sequence, the camera system and its environment are dynamic. The sensor parameters and $T$ may change over time, so $C_i$ becomes time-dependent. Similarly, and perhaps most importantly, the target is also dynamic: both its pose and its internal configuration may change over time. Thus, $R$ also becomes time-dependent.

Denoting the time-dependent coverage and relevance functions at time $t$ as $C_i^{(t)}$ and $R^{(t)}$, respectively, the instantaneous optimal view at time $t$ is

$$Q_{\text{best}}(t) = \arg \max_{i \in N} \left( F \left( C_i^{(t)}, R^{(t)} \right) \right)$$

(6.1)

The first objective in optimal view sequence selection is to maximize the instantaneous coverage performance with respect to the maximum achievable performance. Assuming that the coverage function and the target relevance function $R$ are independently valid, the performance of a view sequence $Q$ is

$$M(Q, C_i^{(t)}, R^{(t)}) = \frac{\sum_{t=0}^{t_{\text{max}}} F \left( C_i^{(t)}, R^{(t)} \right)}{\sum_{t=0}^{t_{\text{max}}} F \left( C_{Q_{\text{best}}(t)}^{(t)}, R^{(t)} \right)}$$

(6.2)

where, by the definition of the instantaneously optimal sequence $Q_{\text{best}}$ per (6.1), $M(Q, C_i^{(t)}, R^{(t)}) \in [0, 1]$. $M(Q, C_i^{(t)}, R^{(t)})$ can be thought of as a measure of the information in $Q$ in proportion to the maximum available information from any sequence of single views per frame (i.e. $Q_{\text{best}}$).

The second objective is to maximize the smoothness of the sequence. In order to define an objective function, a formal definition of “smoothness” is needed. Jiang et al. [22], Daniyal et al. [23], and others identify high-frequency view transitions as undesirable. To clarify this notion, it is suggested that there is an upper bound on the length of a single-view subsequence detrimental to human or machine analysis, depending on the task. Accordingly, rather than assigning a cost to every transition, an undesirable transition in $Q$ is defined as one which occurs less than $J$ frames after the previous transition, and cost is assigned to such transitions only.

Let $T(Q) = \{ t \in [1, t_{\text{max}}] | Q(t) \neq Q(t-1) \}$ be the set of transition frames in $Q$. Given the ascending sequence $T^*(Q) = \langle 0, t_0, ..., t_n \rangle$ where $t_i \in T(Q) \forall i \in [0, n]$, with $T_i^*(Q)$ denoting its $i$th element, the total number of undesirable transitions in $Q$ is

$$U(Q, J) = \sum_{i=2}^{|T(Q)|+1} \begin{cases} 1 & \text{if } T_i^*(Q) - T_{i-1}^*(Q) < J \\ 0 & \text{otherwise,} \end{cases}$$

(6.3)

and the smoothness objective is the minimization of $U(Q, J)$.

For $J > 0$, it is possible that $Q_{\text{best}}$ is a Pareto optimal solution which simultaneously maximizes $M(Q, C_i^{(t)}, R^{(t)})$ and minimizes $U(Q, J)$. In practice, this is unlikely. A flexible aggregate objective func-

\footnote{Strictly speaking, $Q_{\text{best}}$ is not necessarily unique. This is irrelevant in (6.2), but here one might specify "a $Q_{\text{best}}$ which minimizes $U(Q, J)$."}
tion is defined for the multiobjective optimization problem as

\[ A(Q, C_i^{(t)}, R^{(t)}, J, \omega) = \frac{\omega \cdot M(Q, C_i^{(t)}, R^{(t)})}{U(Q, t) + \omega}, \]

(6.4)

where \( \omega > 0 \) is a tolerance factor which scales the impact of the smoothness objective; a larger \( \omega \) is more permissive of a higher proportion of undesirable transitions with respect to the total number of frames. Wherever \( C_i^{(t)}, R^{(t)}, J, \) and \( \omega \) are implicit in the context, the shorthand \( A(Q) \) will be used.

2 Theorem (Bounded Aggregate Objective Function)
For any \( Q, C_i^{(t)}, R^{(t)}, J, \) and \( \omega, A(Q, C_i^{(t)}, R^{(t)}, J, \omega) \in [0, 1]. \)

PROOF By Definition 1 and Definition 3, respectively, \( C_i^{(t)} \in [0, 1] \) and \( R^{(t)} \in [0, 1] \) for all \( t \). By (2.8), \( F(C_i^{(t)}, R^{(t)}) \in [0, 1] \) for all \( t \), so by (6.1), \( Q_{best}(t) \in [0, 1] \) for all \( t \). In (6.2), the denominator is, by definition of (6.1), equal to or greater than the numerator, so \( M(Q, C_i^{(t)}, R^{(t)}) \in [0, 1] \). By the definition of \( T(Q), |T(Q)| \leq t_{\text{max}}, \) clearly \( U(Q, J) \in [0, t_{\text{max}}], \) and \( U(Q, J)/t_{\text{max}} \in [0, 1]. \) Since \( \omega > 0, 0 \leq \omega \cdot M(Q, C_i^{(t)}, R^{(t)}) \leq \omega \leq U(Q, J)/t_{\text{max}} + \omega, \) so \( A(Q, C_i^{(t)}, R^{(t)}, J, \omega) \in [0, 1]. \)

With (6.4) in hand, the optimal monocular view selection problem is formally defined as follows:

Given a multi-camera system \( N, \) an interval of frames \( [0, t_{\text{max}}], \) time-dependent coverage functions \( C_i^{(t)} \) for all \( i \in N, \) a time-dependent relevance function \( R^{(t)} \), a transition smoothness threshold \( J, \) and a tolerance factor \( \omega, \) find a view sequence \( Q \) such that \( A(Q, C_i^{(t)}, R^{(t)}, J, \omega) \) is maximized.

Although (6.4) is intractable for analytic optimization due to the nonlinearity and discontinuity of its components, evaluability suffices for the design of a real-time suboptimal view sequence selection algorithm.

### 6.4 Real-Time View Selection Algorithm

When the problem is to select the best view in real time, each \( Q(t) \) can only be estimated from past and current observations. In terms of the global solution space for \( Q, \) it is not possible to guarantee an optimal solution, as there are multiple possible \( Q(t) \) which might minimize \( U(Q, J) \) (and therefore, maximize \( A(Q) \)) depending on the future state of the system. Algorithm 1 attempts to approximate the optimal solution by introducing hysteresis with a simple tunable transition threshold parameter \( \tau \in [0, 1]. \) This value is added to the score of last selected view in each iteration, under two conditions:

1. the view has been selected for fewer than \( J \) frames, and
2. its score is nonzero.
Condition 1 exists because after \( J \) frames the smoothness criterion has been satisfied, and condition 2 predictively attempts to lengthen the period of the next view while maximizing \( M(Q, C_i^{(t)}, R^{(t)}) \).

Algorithm 1 also reduces the search space for scalability. With the numbers of cameras used to observe common volumes in practical systems (typically well under 100), an exhaustive search over such local groups is practical. Furthermore, if the maximum size of such groups remains relatively constant, and the coverage objective (i.e. target) typically remains somewhat localized over short intervals of time—such as the inter-frame period—then the search space for most frames can be localized accordingly. It is assumed that if the target was best covered by a view in the previous frame, it is likely to be best covered by the same view, or a view which overlaps with that view, in the current frame.

**Algorithm 1 Real-Time View Selection with Hysteresis**

**Input:** \( N, \mathcal{G}, \{C_i|i \in N\}, R, J, \tau, Q(t - 1) \) if \( t > 0 \)

**Output:** \( Q(t) \)

1: \( c \leftarrow 0 \)
2: if \( t > 0 \) and \( m\left(C_i^{(t-1)}(Q(t-1)), R^{(t-1)}\right) > 0 \) then
3: \( \mathcal{V} \leftarrow N_g(Q(t - 1)) \cup Q(t - 1) \)
4: else
5: \( \mathcal{V} \leftarrow N \)
6: end if
7: for all \( i \in \mathcal{V} \) do
8: \( s_i \leftarrow F\left(C_i^{(t)}, R^{(t)}\right) \)
9: end for
10: if \( t > 0 \) and \( s_{Q(t-1)} > 0 \) and \( c < J \) then
11: \( s_{Q(t-1)} \leftarrow s_{Q(t-1)} + \tau \)
12: end if
13: \( Q(t) \leftarrow b|s_b = \max_{i \in \mathcal{V}}(s_i) \)
14: if \( Q(t) \neq Q(t - 1) \) then
15: \( c \leftarrow 0 \)
16: else
17: \( c \leftarrow c + 1 \)
18: end if
19: return \( Q(t) \)

Lines 2 to 6 select the candidate set of views. The notation \( N_g(i) \) indicates the neighbors of \( i \) on the vision graph \( \mathcal{G} = \mathcal{H}_C^2 \) (as defined in Section 2.3.1). If the last selected best view \( Q(t - 1) \) yielded nonzero coverage for \( R^{(t-1)} \), it is assumed (due to target locality over short intervals) that the current best view of the target \( Q(t) \) is a view whose coverage overlaps that of \( Q(t - 1) \) (including itself). Otherwise, if \( F(C_Q^{(t-1)}, R^{(t-1)}) = 0 \), the target is "lost" and, without further assumptions or information about transition topology, an exhaustive search of \( N \) is necessary.\(^2\) Lines 7 to 12 compute the scores for each view. For most views, this is simply the coverage performance per (2.8), but to effect hysteresis the transition threshold \( \tau \) is added to the score of the last selected best view \( Q(t - 1) \) if its previous score was nonzero and it has been selected for fewer than \( J \) frames. Line 13 selects the view with the highest

\(^2\)Note that, in practice, one might avoid necessarily searching all of \( N \) in this case by following some search heuristic, e.g. by increasing order of distance from \( Q(t - 1) \), then short-circuiting to a search of \( N_g(i) \) once some \( F(C_i, R) > 0 \) is found.
6.5. Experimental Results

6.4.1 The Transition Threshold Parameter

The jitter phenomenon, a high-frequency oscillation between two or more selected views within a sequence, is described by Jiang et al. [22], Daniyal et al. [23], and others. It is caused by error in target pose estimates, irregular or discontinuous target motion, and/or discontinuities in the occlusion function. As a nonzero \( J \) biases \( A(Q) \) toward favoring sequences with low-frequency transitions, a nonzero \( \tau \) parameter biases the real-time view selection toward favoring the current view during a period of \( J \) frames, which generally reduces transition frequency.

![Figure 6.1: Jitter in View Selection – Sequence \( Q_1 \) shows a smooth transition from view \( A \) to view \( B \); sequence \( Q_2 \) exhibits jitter.](image)

The selection of \( \tau \) for best results in \( Q \) cannot be isolated from the complex factors involved in instantaneous view selection, but in general, ceteris paribus, higher \( J \) and larger uncertainty in target pose estimates typically require higher \( \tau \). The simulation experiments in Section 6.5.1 investigate empirically, among other things, how \( \tau \) relates to \( J \) and pose error in practice.

6.5 Experimental Results

6.5.1 Simulation

To evaluate the optimality and robustness of the view selection algorithm against ground truth, a simulated surveillance network of 23 cameras is tasked with observing a single target. For simplicity, the camera system and scene are static. The objective is a simple observation task with task requirements listed in Table 6.1, where the cameras have 1360 × 1024 pixel resolution with a 2.5m subject distance.

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>( \gamma )</td>
<td>20</td>
</tr>
<tr>
<td>( R_{na} )</td>
<td>0.3</td>
</tr>
<tr>
<td>( R_{ni} )</td>
<td>0.01</td>
</tr>
<tr>
<td>( R_{xa} )</td>
<td>0.0</td>
</tr>
<tr>
<td>( R_{xi} )</td>
<td>0.0</td>
</tr>
<tr>
<td>( c_a )</td>
<td>1.72</td>
</tr>
<tr>
<td>( c_t )</td>
<td>1.0</td>
</tr>
<tr>
<td>( \xi_a )</td>
<td>( \pi/2 )</td>
</tr>
<tr>
<td>( \xi_t )</td>
<td>0.3</td>
</tr>
</tbody>
</table>

The target is a simplified human, whose relevance function—which translates into \( R(t) \) based on the target pose at time \( t \)—consists of a directional point normal to the face and three non-directional points around the head.

It is possible to vary the target’s path (which determines the “pose” of the relevance function), the transition threshold \( \tau \), and the smoothness threshold \( J \). Additionally, random fixed error (in both

\[ \text{Specifically, if } p_i(t) \text{ is the position of a point on the target over time, observations suggest that a high } \frac{\partial^3 p_i(t)}{\partial t^3} \text{, or "jerk," tends to be the culprit.} \]
6.5. Experimental Results

Figure 6.2: Layout of Simulated Surveillance Network – Cameras are mounted at various positions and orientations typical of real surveillance networks.

Position and orientation) can be added to the camera poses, as well as to the target pose at each frame. Finally, the camera pose error can be composed into the target pose to model the scenario where the target pose is obtained from the camera itself.

The target’s path is defined by a discrete set of waypoints, as shown in Figure 6.4. The position of the target at time $t$ is obtained from a three-dimensional cubic spline interpolation on the waypoints, and its orientation is tangent to the path interpolation at $t$. This ensures a relatively small difference between $R^{(t-1)}$ and $R^{(t)}$ generally, corresponding to realistic target motion, which upholds the assumption of locality over short intervals.

First, the simulation is run with ideal camera and target pose estimates, with $\omega = 0.1$, varying $J$ and $\tau$ only. Figure 6.5 shows the $A(Q)$ results for the paths shown in Figure 6.4. The general trends are clear. As $J$ increases, the performance of instantaneous view selection only (that is, where $\tau = 0$) drops significantly. Increasing $\tau$ generally reduces $A(Q)$, but can compensate for the effect of increasing $J$ for a greater overall $A(Q)$; thus, as predicted in Section 6.4.1, a higher $J$ threshold tends to be associated
6.5. Experimental Results

Figure 6.3: Visualization of Simulation Environment – A human target with a simple relevance function traverses a path through a series of rooms and hallways containing a network of 23 cameras. Visualized with Adolphus.

Figure 6.4: Agent Paths for Simulation Experiments – Paths are defined by an ordered set of waypoints, connected by a smooth three-dimensional cubic spline interpolation.

with a higher optimal value for $\tau$.

Next, random calibration error of varying controlled degree is introduced to each camera. First, a unit vector $\hat{t}$ with random direction is computed, and the camera is translated by $e_{r}\hat{t}$. Then, similarly, a unit vector $\hat{r}$ with random direction is computed, and the camera is rotated by angle $e_{r}$ about $\hat{r}$. The $C_i$ obtained from these camera poses are used to perform Algorithm 1, but the original ground truth camera poses are used to evaluate $A(Q)$. All possible values of $\tau$ to a precision of 0.01 are tested; a value of $\tau = 0.5$ is sufficient to maximize $A(Q)$ in all cases. Tables 6.2 and 6.3 show the results using $\tau = 0$ and $\tau = 0.5$, for a series of increasing $e_{r}$ and $e_{r}$. As expected, when the target pose is estimated independently from the cameras, calibration error is not a significant source of jitter, and $M(Q)$ performance degrades gracefully.
Finally, we introduce random pose estimate error of varying controlled degree to the target itself, to simulate generic tracking noise (not caused by calibration error) or locally erratic target motion. The error is computed and applied similarly to the camera calibration error in the previous two experiments, albeit separately for each frame. Again, all possible values of $\tau$ to a precision of 0.01 are tested; results for $\tau = 0$ and multiples of 0.25 found to yield the maximum $A(Q)$ are shown in Table 6.4. Note that the number of undesirable transitions $U(Q)$ is greatly reduced without significantly reducing the instantaneous performance $M(Q)$, demonstrating good robustness to the jitter introduced by target
6.5. Experimental Results

Table 6.2: Effects of Calibration Error (Path 1, \( J = 10 \))

<table>
<thead>
<tr>
<th>( e_T )</th>
<th>( e_R )</th>
<th>( \tau )</th>
<th>( M(Q) )</th>
<th>( U(Q) )</th>
<th>( A(Q) )</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>0.00</td>
<td>0.0</td>
<td>1.0000</td>
<td>2</td>
<td>0.9921</td>
</tr>
<tr>
<td>0</td>
<td>0.00</td>
<td>0.5</td>
<td>1.0000</td>
<td>2</td>
<td>0.9921</td>
</tr>
<tr>
<td>20</td>
<td>0.05</td>
<td>0.0</td>
<td>0.9908</td>
<td>2</td>
<td>0.9829</td>
</tr>
<tr>
<td>20</td>
<td>0.05</td>
<td>0.5</td>
<td>0.9905</td>
<td>1</td>
<td>0.9865</td>
</tr>
<tr>
<td>40</td>
<td>0.10</td>
<td>0.0</td>
<td>0.9713</td>
<td>6</td>
<td>0.9486</td>
</tr>
<tr>
<td>40</td>
<td>0.10</td>
<td>0.5</td>
<td>0.9708</td>
<td>3</td>
<td>0.9593</td>
</tr>
<tr>
<td>60</td>
<td>0.15</td>
<td>0.0</td>
<td>0.9504</td>
<td>7</td>
<td>0.9245</td>
</tr>
<tr>
<td>60</td>
<td>0.15</td>
<td>0.5</td>
<td>0.9514</td>
<td>2</td>
<td>0.9439</td>
</tr>
<tr>
<td>80</td>
<td>0.20</td>
<td>0.0</td>
<td>0.9062</td>
<td>3</td>
<td>0.8955</td>
</tr>
<tr>
<td>80</td>
<td>0.20</td>
<td>0.5</td>
<td>0.9061</td>
<td>2</td>
<td>0.8989</td>
</tr>
<tr>
<td>100</td>
<td>0.25</td>
<td>0.0</td>
<td>0.8723</td>
<td>5</td>
<td>0.8552</td>
</tr>
<tr>
<td>100</td>
<td>0.25</td>
<td>0.5</td>
<td>0.8718</td>
<td>1</td>
<td>0.8684</td>
</tr>
</tbody>
</table>

Table 6.3: Effects of Calibration Error (Path 2, \( J = 20 \))

<table>
<thead>
<tr>
<th>( e_T )</th>
<th>( e_R )</th>
<th>( \tau )</th>
<th>( M(Q) )</th>
<th>( U(Q) )</th>
<th>( A(Q) )</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>0.00</td>
<td>0.0</td>
<td>1.0000</td>
<td>7</td>
<td>0.9448</td>
</tr>
<tr>
<td>0</td>
<td>0.00</td>
<td>0.5</td>
<td>0.9991</td>
<td>4</td>
<td>0.9668</td>
</tr>
<tr>
<td>20</td>
<td>0.05</td>
<td>0.0</td>
<td>0.9939</td>
<td>6</td>
<td>0.9465</td>
</tr>
<tr>
<td>20</td>
<td>0.05</td>
<td>0.5</td>
<td>0.9917</td>
<td>2</td>
<td>0.9755</td>
</tr>
<tr>
<td>40</td>
<td>0.10</td>
<td>0.0</td>
<td>0.9830</td>
<td>4</td>
<td>0.9512</td>
</tr>
<tr>
<td>40</td>
<td>0.10</td>
<td>0.5</td>
<td>0.9832</td>
<td>2</td>
<td>0.9671</td>
</tr>
<tr>
<td>60</td>
<td>0.15</td>
<td>0.0</td>
<td>0.9628</td>
<td>4</td>
<td>0.9512</td>
</tr>
<tr>
<td>60</td>
<td>0.15</td>
<td>0.5</td>
<td>0.9637</td>
<td>2</td>
<td>0.9479</td>
</tr>
<tr>
<td>80</td>
<td>0.20</td>
<td>0.0</td>
<td>0.9310</td>
<td>9</td>
<td>0.8660</td>
</tr>
<tr>
<td>80</td>
<td>0.20</td>
<td>0.5</td>
<td>0.9338</td>
<td>3</td>
<td>0.9110</td>
</tr>
<tr>
<td>100</td>
<td>0.25</td>
<td>0.0</td>
<td>0.9140</td>
<td>15</td>
<td>0.8124</td>
</tr>
<tr>
<td>100</td>
<td>0.25</td>
<td>0.5</td>
<td>0.9087</td>
<td>4</td>
<td>0.8793</td>
</tr>
</tbody>
</table>

Pose estimation noise.

Table 6.4: Effects of Target Tracking Error (Path 1, \( J = 10 \))

<table>
<thead>
<tr>
<th>( e_T )</th>
<th>( e_R )</th>
<th>( \tau )</th>
<th>( M(Q) )</th>
<th>( U(Q) )</th>
<th>( A(Q) )</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>0.00</td>
<td>0.00</td>
<td>1.0000</td>
<td>2</td>
<td>0.9921</td>
</tr>
<tr>
<td>20</td>
<td>0.05</td>
<td>0.00</td>
<td>0.9975</td>
<td>51</td>
<td>0.8284</td>
</tr>
<tr>
<td>20</td>
<td>0.05</td>
<td>0.25</td>
<td>0.9963</td>
<td>2</td>
<td>0.9884</td>
</tr>
<tr>
<td>40</td>
<td>0.10</td>
<td>0.00</td>
<td>0.9944</td>
<td>133</td>
<td>0.6490</td>
</tr>
<tr>
<td>40</td>
<td>0.10</td>
<td>0.50</td>
<td>0.9881</td>
<td>2</td>
<td>0.9802</td>
</tr>
<tr>
<td>60</td>
<td>0.15</td>
<td>0.00</td>
<td>0.9902</td>
<td>181</td>
<td>0.5743</td>
</tr>
<tr>
<td>60</td>
<td>0.15</td>
<td>0.50</td>
<td>0.9795</td>
<td>8</td>
<td>0.9491</td>
</tr>
<tr>
<td>80</td>
<td>0.20</td>
<td>0.00</td>
<td>0.9838</td>
<td>275</td>
<td>0.4684</td>
</tr>
<tr>
<td>80</td>
<td>0.20</td>
<td>0.75</td>
<td>0.9704</td>
<td>12</td>
<td>0.9259</td>
</tr>
<tr>
<td>100</td>
<td>0.25</td>
<td>0.00</td>
<td>0.9803</td>
<td>312</td>
<td>0.4360</td>
</tr>
<tr>
<td>100</td>
<td>0.25</td>
<td>0.75</td>
<td>0.9678</td>
<td>22</td>
<td>0.8895</td>
</tr>
</tbody>
</table>

Figure 6.6 shows the vision graph generated from the multi-camera system in the simulation experiments, over an abstract \( ⟨R⟩ \) spanning the interior of the structure. The number of neighbours including the vertices themselves (and thus the number of \( C_t \) evaluations necessary in Algorithm 1) ranges from 2 to 14, with an average of 7.78. Even in this relatively small and dense camera network, a substantial reduction in the search space is seen.

6.5.2 Controlled Camera Data

Algorithm performance is again examined, this time using a data set obtained from a physical eight-camera system. Cameras are calibrated as described in Section B.3 against several common targets, and the resultant pose estimates are composed over a shortest-path tree of a graph in which edges and their weights represent pose estimates and their error margins, respectively. The global coordinate space is that of the Mitsubishi RV-1A manipulator. A HALCON [115] calibration plate (Figure 5.1) is affixed to the end effector of the manipulator, which is then programmed to move through a motion
Figure 6.6: Vision Graph for Simulation Experiments – The simulated camera network yields a relatively dense vision graph, with 2 to 14 neighbours per vertex.

sequence. In order to ensure precise temporal synchronization of the frames, the robot path is divided into 103 discrete positions spaced equally over time, and images of the target are captured from all eight cameras in each such position. The calibration plate itself is the target, and its relevance function—again, translating into $R(t)$ based on the target pose at time $t$—consists of a set of directional points lying in and normal to the plate’s plane.

In this experiment, the target pose estimate is obtained by querying the manipulator position directly via RS-232 serial link. For three values of $J$ and four values of $\tau$, the $A(Q)$ performance with $\omega = 0.1$ and $\omega = 1.0$ is computed. The results, recorded in Table 6.5, demonstrate that good performance can be achieved with an appropriate selection of $\tau$. The two different $\omega$ values also illustrate that depending on tolerance for short transitions, the best suboptimal $Q$ may differ.
6.6 Conclusions

This chapter has approached the problem of view selection as a coverage optimization problem, quantifying the criteria of instantaneous view quality and transition smoothness accordingly. The presented real-time suboptimal solution is relatively simple to implement, yet approaches the (non-causal) optimum very well in comparison to the baseline instantaneous approach, and offers scalability and robustness to all identified sources of jitter. Due to the very general task-oriented nature of the coverage model, the method is suitable for a variety of multi-camera applications in relatively well-controlled environments. Although the approach assumes monocular views and single targets, both the coverage model and the view selection algorithm readily generalize to $k$-ocular views and multiple targets.
7.1 Overview

In this chapter, the coverage model of Chapter 4, modified for three-dimensional vision by active triangulation, is applied to the view planning problem for high-fidelity inspection. A semi-automatic model-based approach is presented. The design process is analyzed, and the automated view planning problem is formulated only for the critically difficult aspects of design. A particle swarm optimization algorithm is applied to the latter portion, including probabilistic modeling of positioning error, using the performance metric as an objective function. The process leverages human strengths for the high-level design, refines low-level details mechanically, and provides an absolute measure of task-specific performance of the resulting design specification. The system model is validated, allowing for a reliable rapid design cycle entirely in simulation. Parameterization of the optimization algorithm is analyzed and explored empirically for performance.

It is assumed that the reader is familiar with the basic operation of three-dimensional vision by triangulation. A review oriented specifically at laser line projection systems of the sort discussed herein is presented in Section B.4.

7.2 Introduction

Optical range sensors based on active triangulation [128] are popular in industrial applications of non-contact three-dimensional metrology and inspection, owing to high fidelity and robustness in comparison to other non-contact methods, and high data acquisition rates in comparison to mechanical methods. A typical inspection task involves measuring a set of points or areas on the surface of an object at resolutions and densities on the order of tens of micrometers. Importantly, the specified
tolerances often run inside of an order of magnitude of the physical capabilities of modern devices, when other factors such as field of view and occlusion are taken into account. Hence, planning the configuration of the sensor system is not a trivial task, requiring careful balancing of a number of constraints to achieve an acceptable solution.

The prevailing trend in industry is for the end user of the inspection system to delegate this difficult task to a third-party specialist integrator, who will assemble a custom solution starting from modern off-the-shelf range cameras (see Blais’ list [129], and note that the niche has matured considerably since its publication) and software. A large part of the expertise being leveraged is in designing a physical configuration capable of meeting the client’s specifications, usually assisted by planning guidelines provided by the range camera manufacturer. Even considering only a small subset of the many factors influencing performance is very labour-intensive to do manually. Additionally, the remaining factors often have subtle and poorly-understood effects that can easily cause the result to fail to meet specifications. To compensate, the integrator often adds an error margin factor to the client specifications. However, this can greatly increase the cost of the solution (or prohibit it altogether), and lacking any more precise a model for evaluation than the planning guidelines and sample testing, the results are yet not guaranteed to meet specifications.

Automated view planning holds promise of efficiency and reliability, yet such methods are not presently in use by integrators. Scott et al. [130] present a survey of view planning, including a detailed analysis of the constraints and requirements to be considered. While a number of these are of interest mainly in reconstruction and do not apply to inspection, there is an evident gap between the remaining points and the state of the art. The authors conclude that oversimplification of the problem leads to unrealistic solutions and is the main barrier to adoption, and suggest that semi-automatic methods may yield tangible advances in the near term.

This chapter proposes such a semi-automatic design process, leveraging the strengths of both human expertise and computational efficiency. The approach addresses two key issues. First, a high-fidelity model of multi-camera bistatic coverage of active triangulation systems, based on the coverage model presented in Chapter 4, is derived. The task model is very general, and directly reflects requirements and configuration parameters. A bounded scalar performance metric predicts online performance (Section 7.3), allowing costly and time-consuming hardware implementation to be replaced with simulation in the design loop. Second, the scope of the automatic planning problem is reduced to only the critical portion that presents difficulty to human designers or expert-based heuristics (Section 7.4). The reduced problem is then analyzed and addressed using a particle swarm optimization technique (Section 7.5).

### 7.2.1 Prior Work

For a comprehensive account of work on view planning for three-dimensional reconstruction and inspection with active triangulation range sensors up to 2003, the reader is referred to the excellent survey by Scott et al. [130]. The authors conclude that the majority of proposed solutions in the literature suffer from a number of oversimplifications of the problem, including modeling the intrinsically bistatic system as a monostatic sensor, placing unrealistic constraints on viewpoints, and ignoring a variety of significant low-level performance criteria. They also conclude that as of publication, no general solution exists automated planning for high-quality inspection.
While several new methods targeted at reconstruction have been put forward since (e.g. by Larsson and Kjellander [131] and by Scott himself [132]), these attempt to solve the next best view or path planning problem for unknown object surface reconstruction. Even recent methods aimed at inspection (e.g. by Shi et al. [133], Rodrigues Martins et al. [134], and Englot and Hover [135]), assuming a priori knowledge of object geometry, still tend to focus on next best view and path planning for total surface inspection. This chapter does not address multiple-viewpoint, registration-based inspection, so these concerns are irrelevant; the aim is to achieve greater fidelity and efficiency in planning static configurations. While no existing work explicitly addresses the planning of multi-camera active triangulation systems, it is interesting to note that this may be considered a constrained case of the aforementioned multiple viewpoint planning problem.

The model-based approach has precedent in some of the more promising results to date. Of note is the work of Tarbox and Gottschlich [136], who employ a volumetric model of the coverage of a bistatic active triangulation sensor. Scott [132] proposes an improved model which makes fewer assumptions about the object; his “verified measurability” is conceptually similar to the bounded performance metric used here. Alarcon Herrera et al. [137] present an initial analysis of the viewpoint evaluation component using a precursor to the coverage model of Chapter 4. Though some aspects of the bistatic nature of the sensor are implicit, laser coverage is not modeled, and the characterization of height resolution does not translate well from task requirements. Also, view planning is based on a generate-and-test paradigm, and includes no automated component.

Scott et al. [130] observe that, in general, the view planning literature lacks clear, explicit characterization of task requirements. Scott [132] explicitly models the task with quantitative requirements for measurement precision and sampling density (roughly equivalent to the height resolution and scanning density criteria, respectively, described here), ending with quantitative performance metrics, but stops at specifying a uniform requirement over the entire target surface, though subset regions and heterogeneous criteria are mentioned. The task model used here provides a complete specification of six separate criteria for coverage of explicit surface points; heterogeneous criteria are implemented by specifying multiple tasks, over which a single performance metric may be computed.

In approaching the view planning problem, both for active triangulation inspection and generally, as discussed in Section 3.2, researchers have repeatedly attempted to model visual coverage in forms convenient for various optimization techniques. These models have invariably suffered from a loss of fidelity, resulting in poor solutions in general or restricting the scope of applicability [130]. Rather than forcing the model to fit the optimization algorithm, the approach taken here, in accord with the theme of this thesis, is instead to develop a relatively high-fidelity model with only a general consideration for optimization, then choose and parameterize an optimization algorithm based on an analysis of the objective function’s nature and landscape.

### 7.2.2 Application Scope

The majority of industrial inspection tasks can be addressed by a basic single-pass linear or rotary range scan configuration, or can be decomposed without loss of information into several subtasks of this form. The significance of this type of solution is that it entirely avoids online registration, which is prohibitively expensive computationally for realistic scanning rates in most applications [138]. Such a configuration includes exactly one line-projection light source (usually, and referred to herein as,
a laser) and one or more range cameras. Configurations with single cameras are well-studied and supported by manufacturers; multiple-camera configurations are less common, but are beginning to see use in industry. It is assumed that multiple-camera configurations are mutually calibrated: see treatments by Saint-Marc et al. [139], Vilaça et al. [140], and Mavrinac et al. [141].

![Figure 7.1: Coordinate Frames in Active Vision – The laser coordinate system (the laser plane) is parallel to—and usually coincides with—the global x-z plane, while the coordinate systems of the cameras are determined by their respective poses.]

The objective is to design a system configuration which achieves the requirements of the task. This configuration is static in the sense that laser and camera parameters are fixed, and the target object undergoes deterministic linear or rotary motion through the system. The design parameters are the poses and respective properties of the laser and cameras, as well as the number of cameras. Some of these parameters will be fixed or constrained during the manual portion of design, and the remainder will be optimized automatically.

Certain sensible restrictions are placed on the geometry of the system, based on the operating characteristics of the range cameras. The global coordinate frame is a right-handed Cartesian coordinate system of $\mathbb{R}^3$ with axes $x$, $y$, and $z$; the origin is arbitrary, and the basis depends on the motion of the object being scanned. In the case of linear motion, the $y$-axis is the transport direction, and in the case of rotary motion, the axis of rotation is parallel to the $x$-axis. The laser position is unconstrained, but the laser plane—the plane in which all projected laser lines lie—must be parallel to the $x$-$z$ plane.¹ Similarly, the camera positions are unconstrained, but they may only rotate about the $x$-axis, as rotations about $y$ and $z$ generally violate the internal assumptions of the sensor design. This rotation is subject to the requirement that the (positive) optical axis intersect the laser plane. The initial pose of the target object is arbitrary.

¹This is, in fact, a simplifying assumption: many systems allow rotation of the laser about the $x$-axis. The resulting oblique projection adds complexity to the process of range mapping. Since it is possible to solve most industrial inspection problems with the laser plane perpendicular to the transport direction, analysis is restricted to this case. However, it is certainly possible to generalize the approach to handle oblique laser configurations.
7.3 Coverage Model for Active 3D Vision

The coverage model of Chapter 4 does not translate trivially to active triangulation. At the sensor level, while most of the same fundamental coverage criteria apply, there are several fundamental differences which must be taken into account. Foremost among these is the bistatic nature of the sensing paradigm: in order for a range camera to produce any information whatsoever at a particular stimulus point, the point must first be illuminated by the laser. Thus, a coverage function for the laser itself is derived. The nature of range imaging tasks also induces some differences in the camera model, most notably, adding a height resolution criterion and modifying the resolution (called scan density here, to avoid ambiguity) and view angle criteria. Finally, since the inspection targets themselves are a major source of occlusion, it is convenient to blur the distinction between the task and environment models in this case by specifying their triangle sets separately within the task model.

To simplify the problem exposure, due to the assumption that the laser plane is parallel to the $x$-$z$ plane, $R$ is defined over $\mathbb{R}^3$ exclusively. Coverage evaluation is handled as described in Section 4.6.2, with additional details in this section. In general, the direction of a projected task point is along the negative laser projection axis. Restricting this to the $z$-axis allows simplification of (4.21) into (7.6), used for the range camera height resolution and view angle criteria.

7.3.1 Task Model

Object Geometry

Although physical object geometry is normally the domain of the environment model, for purposes of discussion, it is convenient in this case to augment the task model with its own environment model of the form described in Section 4.3.2. The target object geometry is represented as a triangulated polygonal mesh of (opaque) surfaces; the triangles form a set $\mathcal{T}_T$ (for clarity, the triangle set of the environment model will be denoted $\mathcal{T}_E$ in this chapter). It is assumed that some means exists of obtaining such a mesh a priori, at some degree of fidelity, from a CAD model or other representation of the object.

Relevance Function

An inspection task specifies a relevance function constrained to the surface of the object; formally, $\exists \Delta \in \mathcal{T}_T | t \in \Delta \forall t \in (R)$. Each task point therefore has an associated surface normal, which is the normal of the triangle in $\mathcal{T}_T$ on which it lies. If continuous areas are specified (e.g. the entire surface), these may be made discrete by sampling according to the scanning density requirement without loss of information.

Task Requirements

The image boundary padding parameter has essentially the same purpose as described in Section 4.3.3; in this case, residual lens distortion effects are the main motivation. The scan density is essentially the horizontal resolution of the range image. The height resolution is the precision to which range

---

²Adolphus implements the more general case of oblique laser projection using $\mathbb{D}^3$ and (4.21).
values are measurable at a point; this is generally the most important requirement in an inspection task. The blur circle diameter constraint is derived from the dependence of the profile interpolation algorithm of the scanner on focus, usually discussed in the hardware or software documentation. The view angle requirement derives from the sensor’s ability to image a profile, which varies inversely with the cosine of view angle relative to the laser projection, and depends on the laser, sensor, and material properties. The incidence angle cutoff occurs at a particular angle, beyond which the light received by the camera drops below a threshold [132].

Table 7.1: Task Requirement Parameters for Active 3D Vision

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Unit</th>
<th>Constraint</th>
<th>Default</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>$\gamma$</td>
<td>P</td>
<td>$0 \leq \gamma \leq \min(w, h)/2$</td>
<td>$\gamma = 0$</td>
<td>image boundary padding</td>
</tr>
<tr>
<td>$R_i, R_a$</td>
<td>L/P</td>
<td>$0 \leq R_i \leq R_a$</td>
<td>$R_i = R_a = \infty$</td>
<td>minimum scanning density</td>
</tr>
<tr>
<td>$H_i, H_a$</td>
<td>L/P</td>
<td>$0 \leq H_i \leq H_a$</td>
<td>$H_i = H_a = \infty$</td>
<td>minimum height resolution</td>
</tr>
<tr>
<td>$c_i, c_a$</td>
<td>P</td>
<td>$1 \leq c_i \leq c_a$</td>
<td>$c_i = c_a = \infty$</td>
<td>maximum blur circle diameter</td>
</tr>
<tr>
<td>$\xi_i, \xi_a$</td>
<td>A</td>
<td>$0 \leq \xi_i \leq \xi_a$</td>
<td>$\xi_i = \xi_a = \pi/2$</td>
<td>maximum view angle</td>
</tr>
<tr>
<td>$\omega_c$</td>
<td>A</td>
<td>$0 \leq \omega_c \leq \pi/2$</td>
<td>$\omega_c = \pi/2$</td>
<td>laser incidence angle cutoff</td>
</tr>
</tbody>
</table>

7.3.2 Range Camera Sensor Model

Within the application scope of this chapter, a camera’s pose is restricted, so the $P$ parameter from Table 4.1, here denoted $P_c$ to differentiate it from the laser pose, does not have six degrees of freedom. Accordingly, here, the position $T_c \in \mathbb{R}^3$, the rotation about the $x$-axis $\theta \in [0, 2\pi)$, and the rotation about the $z$-axis $\psi \in \{0, \pi\}$ are referred to explicitly.

The camera coverage function is a mapping $C : \mathbb{R}^3 \rightarrow [0, 1]$, with

$$C(p) = C^V(p) \cdot C^D(p) \cdot C^H(p) \cdot C^R(p) \cdot C^A(p) \cdot C^O(p) \quad (7.1)$$

where $C^V$ and $C^R$ are as defined by (4.11) and (4.19), and the remaining components are defined in this section.

Scanning Density

The scanning density along the $y$ direction is entirely determined by the transport pitch, and it is assumed that this is designed appropriately for the task requirements. The scanning density along the $x$ direction is a function of the horizontal resolution in units of length per pixel. As derived from (4.12), the depth at which a horizontal resolution $R$ occurs is

$$z_d(R) = \frac{Rw}{\tan \alpha_i + \tan \alpha_r}. \quad (7.2)$$

The scanning density criterion is then given by

$$C^D(p) = B_{[0,1]} \left( \frac{z_d(R_a) - p_z}{z_d(R_a) - z_d(R_i)} \right) \quad (7.3)$$

82
7.3. Coverage Model for Active 3D Vision

Height Resolution

The height resolution criterion is also computed similarly to the resolution criterion, but is additionally dependent on the cosine of the view angle due to distortion of the profile [132]. The depth along the optical axis at which a height resolution \( H \) occurs for points along the ray to a given point \( \mathbf{p} \) is

\[
    z_H(H, \mathbf{p}) = \frac{H h \sin \zeta(\mathbf{p})}{\tan \alpha_l + \tan \alpha_r},
\]

where \( \zeta(\mathbf{p}) \) is the view angle according to (7.6).

The height resolution criterion is given by

\[
    C^H(\mathbf{p}) = B_{[0,1]} \left( \frac{z_H(H_{a}, \mathbf{p}) - p_z}{z_H(H_{a}, \mathbf{p}) - z_R(H_{r}, \mathbf{p})} \right),
\]

View Angle

The view angle criterion is essentially identical to that defined in Section 4.4.4, but since the stimulus space is simplified to \( \mathbb{R}^3 \) here, a redefinition not depending on an explicit \( \mathbf{p}_d \) is required. In this case, (4.21) can be substituted with

\[
    \zeta(\mathbf{p}) = \arccos \left( \mathbf{P}_c^{-1}(\mathbf{z}) \cdot \mathbf{p} \right),
\]

and \( C^A \) is as defined in (4.22).

Deterministic Occlusion

The deterministic occlusion criterion is as defined in Section 4.4.5, with \( \mathcal{T} = \mathcal{T}_g \cup \mathcal{T}_r \).

7.3.3 Laser Model

Most commonly, sheet illumination is effected by laser sources. Within the laser’s local coordinate system, the effective sheet projection is modeled as an isosceles triangle in the \( x-z \) plane, with top angle \( \lambda \) (the so-called fan angle), and base parallel to the \( x \)-axis at distance \( z_p \) (the maximum effective depth) from the top vertex.

Given the laser’s local coordinate system with origin at the projection source and negative \( z \)-axis along the axis of projection, the laser has pose \( \mathbf{P}_L \) which, as discussed in Section 7.2.2, is restricted to translation by a vector \( \mathbf{T}_L \).

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Unit</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>( \lambda )</td>
<td>A</td>
<td>fan angle</td>
</tr>
<tr>
<td>( z_p )</td>
<td>L</td>
<td>projection depth</td>
</tr>
<tr>
<td>( \mathbf{T}<em>L = (T</em>{Lx}, T_{Ly}, T_{Lz}) )</td>
<td>L</td>
<td>position (3 degrees of freedom)</td>
</tr>
</tbody>
</table>

The laser coverage function is a mapping \( L : \mathbb{R}^3 \to \{0, 1\} \), with

\[
    L(\mathbf{p}) = L^p(\mathbf{p}) \cdot L^t(\mathbf{p}) \cdot L^d(\mathbf{p})
\]
comprising three component functions defined in this section.

**Projection**

The task point is covered only if it is within the triangle of projection.

\[
L^p(p) = \begin{cases} 
1 & \text{if } \arctan \left( \frac{|p|}{p_z} \right) \leq \lambda \text{ and } -p_x \leq z_p, \\
0 & \text{otherwise}. 
\end{cases} 
\]  

(7.8)

**Incidence Angle**

The task point is covered only if the angle of incidence within the laser plane is less than the cutoff angle.

\[
L^i(p) = \begin{cases} 
1 & \text{if } \arccos (\hat{u} \cdot \hat{n}) \leq \omega_c, \\
0 & \text{otherwise.} 
\end{cases} 
\]  

(7.9)

where \(\hat{u}\) is a unit vector in the direction of \(u = p - T\), and \(\hat{n}\) is the surface normal at \(p\).

**Occlusion**

The task point is covered only if the ray from the laser source to the point is unobstructed.

\[
L^o(p) = \begin{cases} 
1 & \text{if } \text{intersect}(\Delta, T_L, p) \in \{\emptyset, p\} \forall \Delta \in T_E \cup T_T, \\
0 & \text{otherwise}, 
\end{cases} 
\]  

(7.10)

where intersect is as defined in Section 4.4.5.

### 7.3.4 Task Performance Metric

A complete sensor system coverage function \(M(t) \in [0, 1]\) for a task point \(t \in \langle R \rangle\) is required to evaluate overall coverage performance. According to the task specification, a task point is an arbitrary point in \(R^3\), but scanning takes place exclusively within the laser plane. The transport motion of the target object carries each task point to a new location \(t' \in R^3\) such that \(t' = T_L - t_0\), from which laser and camera coverage may then be computed.

For linear transport, this motion consists simply of translating the target object by \(T = (0, T_{Ly} - t_y, 0)\).

For rotary motion, the target object is rotated about an axis parallel to the \(x\)-axis passing through \(\langle 0, T_{Ly}, z_r \rangle\). For \(t \in \langle R \rangle\), the angle of rotation is \(\phi_r = \arccos(\vec{v} \cdot \vec{z}) \) (or \(2\pi - \phi_r\), depending on the direction of rotation), where \(\vec{v}\) is a unit vector in the direction of \(v = t - (t_x, T_{Ly}, z_r)\).

Armed with an appropriate means of obtaining the transported task point \(t'\), the system coverage function is

\[
M(t) = L \left( P^{-1}_L(t') \right) \cdot \max \limits_i C_i \left( P^{-1}_{C_i}(t') \right). 
\]  

(7.11)
Given a task and system context, the performance is given by (2.8), substituting $M$ as the coverage function to yield

$$F(M, R) = \frac{\sum_{p \in (R)} M(p)R(p)}{\sum_{p \in (R)} R(p)}.$$  

(7.12)

### 7.4 View Planning

#### 7.4.1 Manual Initialization

Based on observation of the design process, the critical difficulty is encountered in planning the position and orientation of the sensor: the six camera coverage criteria are competing objectives over this space, and since focus settings must be adjusted as well, it can be very challenging to find a configuration which meets them all simultaneously. Accordingly, the manual initialization stage must determine:

1. the initial pose of the target object;
2. the transport style (linear or rotary);
3. the axis of rotation, if applicable;
4. the position and properties of the laser;
5. the specific sensors and lenses to use; and
6. the number ($n$) and $\psi$-orientations of cameras.

A complete discussion of the heuristic for designing these aspects is beyond scope, but comments on the justification for this division are warranted. Items 1 to 3 are typically dictated by fixturing, physical constraints, and the locations of the task points on the object surface; there is usually little to be gained by attempting to automate their design. The calculations for an appropriate laser configuration in item 4 are quite straightforward: the $x$-$z$ projections of $(R)$ should lie within the triangle defined by $\lambda$, $z_p$, and $T_L$, which is not normally difficult to achieve. The sensor and lens of item 5 are typically chosen from a limited pool of available products. The zoom and physical aperture settings of the lens require only coarse tuning in this stage, since other co-dependent parameters—the focus and standoff distance—will subsequently be optimized. Finally, there is a small, discrete set of feasible possibilities for the number and $\psi$-orientations of cameras in item 6, and even if the best among them is not obvious to the designer, several options may be tested.

#### 7.4.2 Automatic Refinement

In the automatic refinement stage, all parameters of the system are fixed, except the cameras’ positions, their rotation about the $x$-axis, and their focus setting (thus varying $z_s$, and by extension $f$, $r$, and $A$). Further, the focus setting is constrained such that $z_s$ along the optical axis (in camera coordinates) is always in the laser plane.
7.4. View Planning

To simplify the problem exposure, without loss of generality, the single-camera problem is discussed. For each camera, four independent values are varied, equivalent to the parametrization of a rigid transformation in the \(y\)-\(z\) plane plus translation in \(x\), but more representative of the functional design of the system. These design variables are:

- \(x\), the horizontal position of the camera
- \(h\), height \((z)\) at which the optical axis intersects the laser plane
- \(d\), distance from \(T_c\) (the optical center) to the point \((x, 0, h)\)
- \(\beta\), angle between the \(z\)-axis and the optical axis

![Figure 7.2: Design Variables for Automatic Refinement](image)

The design variables are equivalent to the permissible three-dimensional transformations, but are more intuitive and better behaved in terms of coupling.

From these variables, the position of the camera is computed as

\[
\begin{align*}
T_{cX} &= x \quad (7.13) \\
T_{cY} &= T_{L,y} \pm d \sin \beta \quad (7.14) \\
T_{cZ} &= h + d \cos \beta \quad (7.15)
\end{align*}
\]

where the sign in (7.14) depends on whether the camera is downstream (positive) or upstream (negative) from the laser plane along the transport axis. The angle of rotation is simply \(\beta + \pi\) about the negative or positive \(x\)-axis, respectively. The intrinsic parameters \(f\), \(r\), and \(A\) are obtained via linear interpolation on the lens lookup table, with \(z_s = d\).

A chart showing the impact of each design parameter on each coverage criterion, based on the concept of the design matrix [142], is shown in Table 7.3. The “minor impact” designation indicates that modifying the parameter impacts the criterion indirectly due to the ray from the point to the camera not coinciding with the optical axis; these effects are normally relatively minor, though still significant, over the visible volume. The criteria are strongly coupled with respect to the four design parameters (or any other possible set), and the nature of the bistatic range sensor design—specifically, the three-dimensional coverage of points within the two-dimensional laser plane—makes it impossible to decouple them within the scope of available parameters. Furthermore, the solution space is highly
non-linear and discontinuous in terms of the performance metric $F$, which depends directly on the six criteria. This is clearly an ill-posed problem for human design and for direct numerical optimization.

### Table 7.3: Design Impact Matrix for Automatic Refinement

<table>
<thead>
<tr>
<th></th>
<th>$x$</th>
<th>$h$</th>
<th>$d$</th>
<th>$\beta$</th>
</tr>
</thead>
<tbody>
<tr>
<td>Visibility</td>
<td>•</td>
<td>•</td>
<td>•</td>
<td>•</td>
</tr>
<tr>
<td>Scanning Density</td>
<td>•</td>
<td>•</td>
<td>•</td>
<td>•</td>
</tr>
<tr>
<td>Height Resolution</td>
<td>•</td>
<td>•</td>
<td>•</td>
<td>•</td>
</tr>
<tr>
<td>Focus</td>
<td>•</td>
<td>•</td>
<td>•</td>
<td>•</td>
</tr>
<tr>
<td>View Angle</td>
<td>•</td>
<td>•</td>
<td>•</td>
<td>•</td>
</tr>
<tr>
<td>Occlusion</td>
<td>•</td>
<td>•</td>
<td>•</td>
<td>•</td>
</tr>
</tbody>
</table>

○ – minor impact, • – major impact

For a $n$-camera configuration, the solution space is $4n$-dimensional: $x$, $h$, $d$, and $\beta$ are varied independently for each camera. $F(s)$ denotes the objective function (7.12) with respect to solution point $s \in \mathbb{R}^{4n}$, where system parameters are calculated from $s$ as described above. Given an initial manual configuration as prescribed in Section 7.4.1, the problem is to find $s$ which maximizes $F(s)$, subject to any design constraints on $s$.

### 7.5 Automatic Refinement Algorithm

#### 7.5.1 Particle Swarm Optimization

The view planning problem described in Section 7.4.2 consists essentially of maximizing a complex, irregular, non-differentiable objective function defined over a bounded, continuous solution space. Among metaheuristics, particle swarm optimization (PSO) \[143, 144\] lends itself particularly well to this form of problem. The algorithm itself has been published in various forms in numerous sources, but in order to expose the particular structure and features of this application, it is described in full as used in Algorithm 2.

Each particle’s state at a given iteration is described by a position $s_i \in \mathbb{R}^{4n}$ and a velocity $v_i \in \mathbb{R}^{4n}$. For simplicity of exposure, where any position variable $s$ is uninitialized in Algorithm 2, $F(s) = -\infty$. The lower and upper solution space bounds, respectively, are given as vectors $b_l$ and $b_u$ (values are derived in Section 7.5.2). $U(l, u)$ denotes a random uniform vector bounded by $l$ and $u$.

The parameters $S$, $\omega$, $\phi_p$, $\phi_n$, and $\kappa$ are endemic to the algorithm; they denote the size (number of particles) of the swarm, the particle momentum, the disposition toward the present best local and neighborhood solutions, and the velocity clamping factor, respectively. Appropriate values for these parameters are discussed in Section 7.5.5.

The particle state also stores $p_i$, the best solution found by particle $i$, and $n_i$, the best solution found by any particle in particle $i$’s neighborhood set $N_i \subseteq \{1, \ldots, S\}$. The best solution found by any particle $i \in \{1, \ldots, S\}$ is stored globally in $g$.

The function topology($S, i$) returns a subset of $\{1, \ldots, S\}$, describing a topological relationship between the particles. In the original PSO algorithm, all particles are considered adjacent (i.e. $\{1, \ldots, S\}$ is returned), but using other topologies can improve performance in terms of convergence and optimality \[145, 146\]. The function constrain($s_i, b_l, b_u$) modifies $s_i$ depending on the solution space bounds.
In the original PSO algorithm, no constraints are applied (i.e. $s_i$ is returned), but constraint handling mechanisms can also improve performance [147]. Both are discussed in Section 7.5.5, and selections are defined in (7.17) and (7.18), respectively.

The positions and velocities of the particles are initialized randomly (lines 2 to 6). Each iteration then consists of updating the local, neighborhood, and global best solutions based on the value of $F$ at each particle’s current position (lines 9 to 29), then updating each particle’s position and velocity (lines 30 to 41). The lines involving $F_e$, comprising the block in lines 16 through 23 and line 25, are a nonstandard feature related to positioning error compensation, explained in Section 7.5.3.

The algorithm proceeds until one of three stop conditions is met: the global best solution exceeds $F_{\text{accept}}$, a proportion $C_{\text{max}} \in (0, 1]$ of particles cluster within $\epsilon$ of $g$ (by Euclidean distance in a solution space normalized on the bounds), or the algorithm has run for $I_{\text{max}}$ iterations.

### 7.5.2 Solution Space Bounds

We derive a set of conservative bounds based on an assumption which appears to hold in practical, non-pathological cases. If $x_{\text{min}}$, $x_{\text{max}}$, $z_{\text{min}}$, and $z_{\text{max}}$ represent the $x$-$z$ bounds of $\langle R \rangle$, the proposed bounds on $x$, $h$, $d$, and $\beta$ will not exclude any solution with $M(t) > 0$ for all $t \in \langle R \rangle$ where $x_{\text{min}} \leq x \leq x_{\text{max}}$ and $z_{\text{min}} \leq h \leq z_{\text{max}}$ (the assumption being that these inequalities hold for some optimal solution). The bounds are listed in Table 7.4.

<table>
<thead>
<tr>
<th>Variable</th>
<th>Lower Bound</th>
<th>Upper Bound</th>
</tr>
</thead>
<tbody>
<tr>
<td>$x$</td>
<td>$x_{\text{min}}$</td>
<td>$x_{\text{max}}$</td>
</tr>
<tr>
<td>$h$</td>
<td>$z_{\text{min}}$</td>
<td>$z_{\text{max}}$</td>
</tr>
<tr>
<td>$d$</td>
<td>$d_{\text{min}}$</td>
<td>$d_{\text{u}}$</td>
</tr>
<tr>
<td>$\beta$</td>
<td>0</td>
<td>$\zeta_a$</td>
</tr>
</tbody>
</table>

The values for the bounds on $x$, $h$, and $\beta$ have already been defined; those for $d$ require explanation. Let $d_{\text{min}}$ and $d_{\text{max}}$ represent, respectively, the minimum and maximum values of $z_s$ in the lens lookup table. The upper bound is defined as

$$d_u = \min[z_R(R), z_H(H, P_C^{-1}(\langle x, 0, h \rangle))]$$

(7.16)

where the values of $f$ and $o$—upon which $z_r$ and $z_H$ depend, per (7.2) and (7.4)—are obtained from the lens lookup table with $z_s = d_{\text{max}}$. The values of $x$ and $h$ are irrelevant as long as they are in accord with $T_C$ for the purposes of the calculation.

Design constraints on the configuration of a camera, including limits on lens settings and physical position, are modeled by overriding the value of $F$ to $-\infty$ for all solutions violating the constraints, as proposed by Mendes [146]. These may take arbitrary form, and are not necessarily restricted to rectangular bounds in the $x$-$h$-$d$-$\beta$ design space.

### 7.5.3 Compensating for Positioning Error

Due to limitations on the physical accuracy of positioning the cameras and the target object fixture, the true poses of these entities will inevitably differ, to some extent, from the optimized design [130].

---

**Table 7.4: Bounds on Design Variables**

<table>
<thead>
<tr>
<th>Variable</th>
<th>Lower Bound</th>
<th>Upper Bound</th>
</tr>
</thead>
<tbody>
<tr>
<td>$x$</td>
<td>$x_{\text{min}}$</td>
<td>$x_{\text{max}}$</td>
</tr>
<tr>
<td>$h$</td>
<td>$z_{\text{min}}$</td>
<td>$z_{\text{max}}$</td>
</tr>
<tr>
<td>$d$</td>
<td>$d_{\text{min}}$</td>
<td>$d_{\text{u}}$</td>
</tr>
<tr>
<td>$\beta$</td>
<td>0</td>
<td>$\zeta_a$</td>
</tr>
</tbody>
</table>
If the algorithm converges on a global optimum $s$, and the actual position achieved in practice is $s_\epsilon$, then $F(s_\epsilon) \leq F(s)$. The true practical goal is to maximize the expected performance $E[F(s_\epsilon)]$. In cases where the global maximum of $F$ is non-unique, which is almost certainly the case if $F(s) = 1.0$, the algorithm may reach a solution near the “edge” of a non-singular optimal solution space region, and terminate. By considering positioning error within the algorithm, it is possible to encourage convergence to a solution with better surrounding solutions.

The relationship of pose error to the ultimate value of $F$ is complex, and deriving a closed-form probabilistic model of coverage is infeasible. Instead, compensation is achieved by directly testing the robustness of a promising candidate solution using a probabilistic model of positioning error and Monte Carlo sampling. Let $F_e(s)$ represent the mean value of $F(s)$, with the error model applied to the cameras and target object, over $n_e$ samples. The essence of lines 16 through 23 and line 25 of Algorithm 2 is that if $F(s_i)$ for one or more $i$ is at the global best for a given iteration, the algorithm begins evaluating and ranking by $F_e(s_i)$ as well. The motivation for this conditional evaluation, as opposed to simply incorporating it into the fitness function, is that $F_e$ is $n_e$ times as expensive to evaluate as $F$, and there is no reason to evaluate it for points known to be nominally suboptimal. An empirical analysis of the tradeoff between additional performance cost and improvement in optimality with positioning error is reported in Section 7.6.3.

Random error is introduced to the poses of the cameras and target object according to the following model. For the cameras, a position offset is applied to each coordinate sampled from a zero-mean Gaussian distribution with standard deviation $\sigma_{CT}$, and an orientation offset is applied as a rotation about the $x$-axis by an angle sampled from a zero-mean Gaussian distribution with standard deviation $\sigma_{CR}$. For the target object, a similar position offset is applied, with the Gaussian distribution having standard deviation $\sigma_{TT}$, and the orientation offset is applied as a rotation about a random axis uniformly sampled over the unit sphere, and with the Gaussian distribution for the angle having standard deviation $\sigma_{TR}$.

### 7.5.4 Computational Complexity of the Fitness Function

Since $F$ (7.12) is evaluated repeatedly in Algorithm 2—at least $S$ times per iteration and, in the hypothetical worst case, as many as $(n_e + 1)S$ times per iteration—it is necessary to consider its computational complexity.

The fact that all of the laser parameters are static allows for caching of the laser plane points and the results of the laser coverage function $L$ (7.7). The camera coverage function $C$ (7.1) must be evaluated for each point $t \in (R)$, for each camera $i \in \{0, \ldots, n\}$. As described in Section 4.7.2, the critical component is $C^0$ (4.24), and the computational complexity is in $O(n \times |(R)| \times |T_E \cup T_T|)$.

In practice, $n$ is quite low (usually 1 or 2), but it is not unreasonable for $|(R)|$ and $|T_E \cup T_T|$ each to be on the order of hundreds or thousands for a high-fidelity model of an inspection task.

### 7.5.5 Optimality and Convergence

Trelea [148] presents an analysis of optimality and convergence of the particle swarm optimization algorithm in general using results from dynamic system theory, and derives guidelines for parameter

---

3For completeness, if $n_e = 0$, $F_e(s) = -\infty$ for all $s \in \mathbb{R}^m$.  

---

89
selection. We seek parameters for which the algorithm will reliably converge to a global optimum within a finite number of evaluations of $F$, guided in part by these results.

It is first necessary to consider the behavior of the fitness function over the solution space. For a given camera position, the subset of $\mathbb{R}^3$ for which $C^O = 1$ is, in general, bounded by an arbitrarily complex polyhedron, depending on $T$. Since intersect is symmetric over the endpoints of the line segment, the same can be said of the subset of possible $T_c$ for which $C^O(p) = 1$, for some given $p \in \mathbb{R}^3$ (Tarabanis et al. [60] compute this subset as part of their view planning approach, and the typical complexity is evident). At such boundaries, due to the necessarily bivalent nature of $C^O$, $F(s)$ is, in general, discontinuous, and this effect produces numerous local maxima dispersed in a non-uniform fashion throughout the solution space.

Avoiding global convergence to one of these local maxima suggests the use of a loosely connected topology [145]. A ring topology is chosen, in which each particle has exactly two neighbors in addition to itself, defined by

$$\text{topology}(S, i) = \begin{cases} \{S, 1, 2\} & \text{if } i = 1, \\ \{S - 1, S, 1\} & \text{if } i = S, \\ \{i - 1, i, i + 1\} & \text{otherwise.} \end{cases}$$ (7.17)

A large $S$ improves optimality, but requires more evaluations of $F$ per iteration, and additionally, reduces the global convergence rate [148]; values near $S = 25$ perform acceptably well while converging to the global optimum for various tasks. Clamping velocity with $\kappa = 0.5$ and setting $\phi_n > \phi_p$ increase the convergence rate without overly damping exploration. Values of $\omega = 0.7298$, $\phi_p = 1.1959$, and $\phi_n = 1.7959$ yield a good tradeoff [149].

In selecting a boundary constraint handling mechanism, it is considered that the solution space is relatively low-dimensional, and that there is no particular expected large-scale structure of $F$. The nearest method [147] appears to improve convergence by keeping the particles within the subspace assumed to contain the optimum. This constraint is defined by

$$\text{constrain}(s, b_l, b_u) = \sum_{i=1}^{4n} \min(\max(s_i, b_{l,i}), b_{u,i})\hat{i}_i$$ (7.18)

where $\hat{i}_i$ is the $i$th unit vector of the solution space basis.

## 7.6 Experimental Results

Thirty black toy bricks are used to rapidly construct physical test targets for linear scans, in a diversity of shapes with known structures. Individual bricks are modeled in Adolphus with 10 triangles per brick for occlusion, for a maximum total of 300 triangles in each target model. Various sets of between 12 and 72 task points are defined on the surface of these models. Task requirements are varied, for both single- and dual-camera configurations, to cover several cases of existence and abundance of $F = 1.0$ global optima for each task.

### 7.6.1 Validation of the Performance Metric

Comprehensive validation experiments on the original visual coverage model are presented in Chapter 5. As this approach fundamentally relies on the accuracy of the modified model presented in
Section 7.3, validation results in the context of range cameras, specifically, those relating to occlusion and scan density, are summarized here. System model parameters are obtained by calibration, so positioning error does not factor in validation.

![Simulated Validation Configuration – Dual SICK-IVP Ranger D cameras with x = 0mm, h = 142mm, d = 206mm, and β = π/4 image a structured target of toy bricks.](image)

**Figure 7.3:** Simulated Validation Configuration – Dual SICK-IVP Ranger D cameras with \(x = 0\)mm, \(h = 142\)mm, \(d = 206\)mm, and \(β = \pi/4\) image a structured target of toy bricks.

![Structured Target Scan – Surface scan of the structured target (shown simulated), obtained using the real implementation of the system in Figure 7.3. Point cloud visualized using PCL [150].](image)

**Figure 7.4:** Structured Target Scan – Surface scan of the structured target (shown simulated), obtained using the real implementation of the system in Figure 7.3. Point cloud visualized using PCL [150].

The combined laser and camera occlusion, derived from the formulation of (7.11) as

\[
M^O(t) = L^O \left( P_{L}^{-1}(t') \right) \cdot \max_{i} C_i^O \left( P_{C_i}^{-1}(t') \right)
\]

may be validated against real data simply by verifying that \(M^O(t)\) is an indicator function for the existence of data at \(t\). Experiments on structured targets demonstrate perfect correlation throughout the imaged laser plane. For the object shown in Figure 7.4, of 48 task points located on the studs of the toy bricks, six are occluded to the laser, and a further six are occluded to the cameras. The 36 remaining points yielding \(M^O(t) = 1\) are fully imaged in the scan, while those yielding \(M^O(t) = 0\) are absent from the scan.

The characterization of scanning density is validated by solving for \(R\) in (7.2) for \(p_x\) of a task point in camera coordinates, \(p\), and comparing to the actual horizontal density of data in the immediate neighbourhood of \(p\) in the scan. The visible subset of the 48 task points on the object shown in Figure 7.4 are tested in this way at four calibrated positions of a single range camera. Figure 7.5 shows the strong correlation; overall mean squared error is \(6.344 \times 10^{-5}\)mm/px.
7.6. Experimental Results

Figure 7.5: Validation of Scanning Density Criterion – The black line indicates the nominal $R$ given by the relation (7.2), and red crosses indicate actual mean scan density in the point cloud in the neighbourhoods of the nominal task points.

7.6.2 Optimality and Convergence of Automated Planning

With a validated model of coverage, we conduct the remaining experiments in a closed-loop simulation of the system. Three targets of realistic complexity, shown in Figure 7.6, are designed along with several sets of task points and requirements each. 7.5 lists some details for four representative linear scan tasks, along with the statistical results of the experiments. The tasks are expressly designed to be difficult, in the sense that only a small subset of the solution space yields $F = 1.0$, and that numerous local maxima exist. This is achieved by positioning task points among occluding surfaces, and by tuning the requirements to be as demanding as possible while still allowing for at least one nominally ideal solution. In all cases, the particle swarm parameter values and functions discussed in Section 7.5.5 are used, and error compensation parameters are set to $n_e = 20$, $\sigma_{CT} = \sigma_{TT} = 2.0$, and $\sigma_{CR} = \sigma_{TR} = 0.01$. Termination conditions are $F_{\text{accept}} = 1.0$ and $I_{\text{max}} = 500$; No clustering terminator is used.

For each task, a series of 48 optimizations are conducted. The bottom portion of Table 7.5 lists, respectively, the mean value of $F$ for the best solution, the mean value of $E[F]$ for the best solution, the mean number of iterations to convergence or termination, and the mean number of evaluations of the objective function. Figure 7.7 shows statistics for $F$ and $E[F]$ per iteration.

A number of relevant observations can be made about these results. The initial $F(g)$ is virtually
always nonzero and often relatively high, indicating that, for most tasks, the bounds defined in 7.4 are a tight initial guess about the location of the optimum. Although occasional temporary stagnation is observed, in tasks 1, 2, and 3, the algorithm converges quickly to the nominal optimum, and the expected value optimum follows shortly thereafter.

The case of task 4 is shown to demonstrate what occurs when no nominal optimum with $F = 1.0$ exists; as the algorithm incrementally improves its estimate of the nominal optimum, the stochastic $E[F]$ value fluctuates slightly, and eventually settles. This fruitless behaviour can be mitigated using the clustering termination condition.

### 7.6.3 Robustness to Positioning Error

A series of 48 optimizations were conducted on a linear-transport experimental model similar to those in the previous section for each $n_e \in \{0, 5, 10, 15, 20, 25, 30\}$, with positioning error modeled with $\sigma_{CT} = \sigma_{TT} = 2.0$ and $\sigma_{CR} = \sigma_{TR} = 0.01$. All solutions converged to a nominal $F = 1.0$.

7.8(a) shows the mean value of $F$, obtained over 300 trials with the same error parameters, for each of the 48 solutions for each value of $n_e$. 7.8(b) shows the mean number of evaluations of $F$ which were conducted in the original optimizations (i.e., which were required to produce the solutions). As expected, there is a roughly linear relationship between the performance cost and the accuracy in finding robust solutions, given fixed optimization parameters and a fixed number of iterations.

### 7.7 Conclusions

This chapter has presented a novel view planning approach for high-accuracy 3D inspection systems based on active triangulation. Rather than attempting to optimize over a large set of design variables, with the assistance of expert integration engineers, the design process for this class of system has

### Table 7.5: Experimental Tasks and Results

<table>
<thead>
<tr>
<th>Task</th>
<th>1</th>
<th>2</th>
<th>3</th>
<th>4</th>
</tr>
</thead>
<tbody>
<tr>
<td>Target</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>$n$</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>1</td>
<td>2</td>
<td>2</td>
<td>2</td>
<td></td>
</tr>
<tr>
<td>$R_l$</td>
<td>0.15</td>
<td>0.20</td>
<td>0.25</td>
<td>0.50</td>
</tr>
<tr>
<td>$R_a$</td>
<td>0.19</td>
<td>0.60</td>
<td>0.30</td>
<td>2.00</td>
</tr>
<tr>
<td>$H_l$</td>
<td>0.19</td>
<td>0.20</td>
<td>0.25</td>
<td>0.10</td>
</tr>
<tr>
<td>$H_a$</td>
<td>0.23</td>
<td>0.60</td>
<td>0.30</td>
<td>1.00</td>
</tr>
<tr>
<td>$c_l$</td>
<td>1.00</td>
<td>1.00</td>
<td>1.00</td>
<td>1.00</td>
</tr>
<tr>
<td>$c_a$</td>
<td>2.00</td>
<td>2.00</td>
<td>2.00</td>
<td>2.00</td>
</tr>
<tr>
<td>$\zeta_l$</td>
<td>1.00</td>
<td>1.00</td>
<td>0.90</td>
<td>1.00</td>
</tr>
<tr>
<td>$\zeta_a$</td>
<td>1.00</td>
<td>1.00</td>
<td>0.90</td>
<td>$\pi/2$</td>
</tr>
<tr>
<td>$\omega_c$</td>
<td>1.20</td>
<td>1.20</td>
<td>1.20</td>
<td>1.20</td>
</tr>
<tr>
<td>$\gamma$</td>
<td>10</td>
<td>10</td>
<td>10</td>
<td>20</td>
</tr>
<tr>
<td>$F$</td>
<td>1.0000</td>
<td>1.0000</td>
<td>1.0000</td>
<td>0.9999</td>
</tr>
<tr>
<td>$E[F]$</td>
<td>1.0000</td>
<td>1.0000</td>
<td>1.0000</td>
<td>0.9161</td>
</tr>
<tr>
<td>Iterations</td>
<td>32</td>
<td>53</td>
<td>47</td>
<td>500</td>
</tr>
<tr>
<td>Evaluations</td>
<td>4301</td>
<td>4336</td>
<td>3166</td>
<td>15060</td>
</tr>
</tbody>
</table>
Figure 7.7: Optimization Statistics – In cases where a nominal optimum with $F = 1.0$ exists, both $F$ and $E[F]$ are observed converging to optima.

Figure 7.8: Robustness to Positioning Error – A solution with nominal $F = 1.0$ becomes suboptimal on average with positioning error; increasing $n_e$ improves the solution but increases computation time.

been analyzed, and the critically difficult portion has been reduced to only four design parameters per camera. The result is a semi-automatic process which leverages human expertise to create a basic system design, for which the camera parameters are then refined through automatic optimization.

The first major contribution is a high-fidelity task-oriented model of coverage performance, which evaluates the overall design—including manual components—with a bounded scalar metric, and also serves as the objective function for optimization. Validation experiments demonstrate its effectiveness in predicting a posteriori task performance, so that optimized designs can be reliably tested in simulation, moving costly hardware testing further outside the cycle.
The second major contribution is the identification of the critical portion of the design process, and the successful application of an algorithm based on particle swarm optimization to this reduced problem. Although the high-fidelity objective function carries a relatively high performance cost, experimental evaluation demonstrates that, with appropriate parameters, the algorithm reliably converges to optimal solutions within a small number of iterations. The algorithm also incorporates a means of directly compensating for positioning system uncertainty for the cameras and target.

Actual convergence times vary widely, depending on the task and number of cameras; typical “difficult” problems with localized optima tend to complete on the order of ten minutes, using the Adolphus-based implementation on modern (single-core) hardware. In general, at these rates, the algorithm is already expected to far outperform human experts. With further optimization and parallelization, it should be possible to substantially reduce convergence times.
Algorithm 2 Particle Swarm Optimization

\textbf{Input:} \(n, b_l, b_u, S, \omega, \phi_p, \phi_n, \kappa, I_{\text{max}}, F_{\text{accept}}, C_{\max}, \epsilon\)

\textbf{Output:} \(g\)

\begin{enumerate}
\item \(v_{\text{max}} \leftarrow \kappa (b_u - b_l)\)
\item for \(i = 1 \rightarrow S\) do
\item \(s_i \sim U(b_l, b_u)\)
\item \(v_i \sim U(-v_{\text{max}}, v_{\text{max}})\)
\item \(N_i \leftarrow \text{topology}(S, i)\)
\item end for
\item \(I \leftarrow 0\)
\item while \(I < I_{\text{max}}\) and \(F_e(g) < F_{\text{accept}}\) and \(C < SC_{\max}\) do
\item for \(i = 1 \rightarrow S\) do
\item if \(F(s_i) > F(p_i)\) then
\item \(P_i \leftarrow s_i\)
\item if \(F(s_i) > F(g)\) then
\item \(g \leftarrow s_i\)
\item end if
\item end if
\item if \(F(s_i) = F(g)\) then
\item if \(F_e(s_i) > F_e(p_i)\) then
\item \(P_i \leftarrow s_i\)
\item end if
\item if \(F_e(s_i) > F_e(g)\) then
\item \(g \leftarrow s_i\)
\item end if
\item end if
\item for all \(j \in N_i\) do
\item if \(F(s_j) > F(n_i)\) or \(F(s_j) = F(n_i)\) and \(F_e(s_j) > F_e(n_i)\) then
\item \(n_i \leftarrow s_j\)
\item end if
\item end for
\item end for
\item end for
\item \(C \leftarrow 0\)
\item for \(i = 1 \rightarrow S\) do
\item for \(k = 1 \rightarrow 4n\) do
\item \(r_p, r_n \sim U(0, 1)\)
\item \(v_{l_k} \leftarrow \omega v_{l_k} + \phi_p r_p (P_{l_k} - x_{l_k}) + \phi_n r_n (n_{l_k} - x_{l_k})\)
\item \(v_{l_k} \leftarrow \min(\max(v_{l_k}, -v_{\text{max},k}), v_{\text{max},k})\)
\item end for
\item \(s_i \leftarrow \text{constrain}(s_i + v_i, b_l, b_u)\)
\item if \(|| (s_i - g - b_l)/(b_u - b_l) || < \epsilon\) then
\item \(C \leftarrow C + 1\)
\item end if
\item end for
\item \(I \leftarrow I + 1\)
\item end while
\item return \(g\)
\end{enumerate}
CHAPTER 8

Camera Network Load Distribution

Can you fawning sycophants do more than grovel?
What of the task I set you?

Darkness, Legend (1985)

8.1 Overview

Optimal distribution of task processing in a camera network is approximated by adapting a local search heuristic for parallel machine scheduling to the hypergraph model of coverage overlap topology presented in Section 2.3.1. Simulation results are presented to demonstrate the effectiveness of the approach.

It is assumed that the reader is familiar with hypergraphs; a review of the topic, including terminology and notation, is given in Section A.2.

8.2 Introduction

Although centralized architectures for fusing and processing data from the multiple sources in camera networks are a natural extension of traditional computer vision methods, such configurations are limited in scalability and robustness. The increasingly popular distributed smart camera network [3] paradigm is the answer to this challenge. In such a system, each camera node possesses local processing capabilities, and data is increasingly abstracted (and thus increasingly compact) as it is communicated and processed farther from its original source. Zivkovic and Kleihorst [151] give an overview and analysis of smart camera node architecture illuminating the benefits of this design.

Naturally, any initial image or video processing tasks which require data only from a single node are assigned to that node. However, if the nodes themselves are also responsible for fusing and processing data from multiple sources—as they must be, in a true distributed smart camera network—it is less obvious where to assign such tasks.

Scheduling has been an active area of research for decades, and algorithms solving a variety of different problems have been used in such diverse applications as manufacturing and distributed com-
puting [152]. Formulating an appropriate scheduling problem requires domain-specific knowledge; in this case, an understanding of the underlying nature of a multi-camera task.

The scale and performance of most tasks in multi-camera networks (indeed, in sensor networks generally) are directly related to the volume of coverage of the sensor(s) in question. Chapter 4 presents a real-valued coverage model for multi-camera systems. Validation experiments in Chapter 5 demonstrate that, given a set of a priori parameters of the multi-camera system and some task requirements, this model accurately describes the true coverage of a scene in the context of the task.

The next step is to abstract this understanding into a topological structure suitable for optimization over the network. Section 3.3 surveys a variety of such models. In the context of camera networks which may be processing a coverage-bound task with data from arbitrary combinations of sensors, only the hypergraph representation described in Section 2.3.1 is sufficiently general.

The primary contribution of this chapter, detailed in Section 8.3, is the characterization of the optimal task processing distribution problem in the hypergraph framework, and the adaptation of a local search heuristic from the scheduling literature [153] which has been shown to exhibit good performance for this class of problem. Simulated experimental results demonstrating the method on a virtual network of 23 cameras are presented in Section 8.4.

8.3 Task Processing Distribution

Consider the portion of a $k$-ocular task in camera network $N$ which involves processing data from all of $M \subseteq N$, where $|M| = k$; this shall be termed an $M$-subtask. Only stimuli within $\langle C_M \rangle$ are relevant to an $M$-subtask. Given a relevance function $R$ for the task, the expected processing load for a given $M$-subtask is proportional to $|\langle C_M \cap R \rangle|$. Although this conjecture is tautological given that $R$ is arbitrary, since $R$ represents the expected distribution of the stimuli necessary to perform the task, it is reasonable to assume in general that it also reflects the distribution of the processing load incurred by said stimuli.

Assuming that $N$ consists of smart camera nodes with homogeneous local computational resources, the problem is to distribute the processing of all $M$-subtasks over the nodes such that the maximum load on any one node is minimized.

The set of eligible nodes to which $M$-subtasks may be assigned is restricted to $M$, for the following reasons:

1. **Robustness**: If a node $n \in M$ fails, the $M$-subtask can no longer be processed. Thus, assigning it to any $n \in M$ carries no risk of disrupting service for valid models.

2. **Locality**: In a large network, because the sensing range is finite, if $\langle C_M \rangle \neq \emptyset$, it is likely that nodes $M$ are physically proximate. Making no assumptions regarding the network structure, it is sensible to keep the $M$-subtask processing node physically local for communication efficiency.

The usefulness of this restriction is especially apparent in the special case $k = 1$, allowing camera-local subtasks (image preprocessing, etc.) to be included in the accounting.

Given a $K$-ocular task, where $K \subset \mathbb{Z}^+$, this problem can be solved by finding an orientation of $H^K_C$ (as defined in Section 2.3.1) which minimizes the maximum weighted indegree.
8.3.1 Minimum Indegree Orientation

The minimum maximum indegree orientation problem for hypergraphs can be stated as follows. Given a simple, undirected, weighted hypergraph \( \mathcal{H} = (V, E, w) \), find an orientation \( \Lambda \) of \( \mathcal{H} \) which minimizes \( \max_{u \in V} \delta^i_{\Lambda}(u) \).

This is equivalent to the scheduling problem of offline makespan minimization over identical parallel machines with eligibility constraints [154]; according to the three-field notation by Graham et al. [155], \( P|M_j, M_j \neq M_k \) if \( i \neq k \). This is a special case of \( P|M_j|C_{\text{max}} \), which in turn is a special case of \( R||C_{\text{max}} \) [156]. The problem is NP-hard [157], but a number of approximation algorithms and search heuristics have been proposed.

A local search heuristic based on the GR/EFF descent of Piersma and Van Dijk [153] is presented in Algorithm 3. The main differences are the use of hypergraph notation and some simplifications made possible by constraints particular to the problem.

Throughout Algorithm 3, \( \Lambda = (V, \tilde{E}, w) \). The heuristic starts with a greedy orientation (lines 2 to 5). Line 15 sorts \( V \) in nonincreasing order of indegree.

8.4 Experimental Results

Load distribution is tested on a simulated network \( N \) of 23 camera nodes arranged in a virtual environment with walls and other occlusions. Tasks are invariant to view angle; accordingly, the discussion is simplified by working exclusively in \( \mathbb{R}^3 \), as per Section 4.6.2. A top view of the environment is shown in Figure 8.1, along with the relevance function \( R \), which is uniform in \( z \) from 1.5m to 2.0m (with the floor at 0.0m, and all cameras at 2.5m), and the locations of the cameras.

The camera coverage functions are derived from real parameters of a calibrated Prosilica EC-1350 camera with a Computar M3Z1228C-MP lens. The specific task parameters used are \( \gamma = 20, R_{nl} = 0.3, R_{na} = 0.01, c_l = 1.0, \) and \( c_a = 1.72 \). Extrinsic parameters are defined manually to deploy the cameras in a reasonable arrangement covering the environment, with \( F(C, R) = 0.8242 \).

The coverage hypergraph \( \mathcal{H}_C \) for \( N \) and \( R \) is computed over a discrete sampling of \( \mathbb{R}^3 \), defined by \( \{(250x, 250y, 250z) | x, y, z \in \mathbb{Z} \} \), with coordinates in millimeters. Although it is too large to represent here graphically, Table 8.1 shows some statistics of the hyperedges in the complete \( \mathcal{H}_C \).

<table>
<thead>
<tr>
<th>Edge Size</th>
<th>Count</th>
<th>Mean Weight</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>23</td>
<td>750.51</td>
</tr>
<tr>
<td>2</td>
<td>78</td>
<td>155.66</td>
</tr>
<tr>
<td>3</td>
<td>130</td>
<td>50.13</td>
</tr>
<tr>
<td>4</td>
<td>152</td>
<td>23.49</td>
</tr>
<tr>
<td>5</td>
<td>122</td>
<td>14.09</td>
</tr>
<tr>
<td>6</td>
<td>61</td>
<td>9.37</td>
</tr>
<tr>
<td>7</td>
<td>17</td>
<td>6.40</td>
</tr>
<tr>
<td>Total</td>
<td>583</td>
<td>71.85</td>
</tr>
</tbody>
</table>

The vision graph \( \mathcal{H}_V \) for this camera network is computed similarly in Chapter 6 and shown in Figure 6.6.
Algorithm 3 Minimum Maximum Indegree Orientation Heuristic

Input: $\mathcal{H} = (V, E, w)$
Output: $\Lambda$

1: $\tilde{E} \leftarrow \emptyset$
2: for all $e \in E$ do
3: $u \leftarrow \arg\min_{v \in e} \delta_A^I[v]$
4: $\tilde{E} \leftarrow \tilde{E} \cup \{e^u\}$
5: end for
6: $v_{\text{max}} \leftarrow \arg\max_{v \in V} \delta_A^I[v]$
7: $\mathcal{R} \leftarrow \{(v, e^{v_{\text{max}}}) \mid v \in V, e^{v_{\text{max}}} \in \tilde{E}\}$
8: while $\mathcal{R} \neq \emptyset$ do
9: $\mathcal{R} \leftarrow \mathcal{R} \setminus (v, e^{v_{\text{max}}})$ for any $(v, e^{v_{\text{max}}}) \in \mathcal{R}$
10: if $\delta_A^I[v] < \delta_A^I[v^{v_{\text{max}}}] - w(e)$ then
11: $\tilde{E} \leftarrow \tilde{E} \setminus e^{v_{\text{max}}} \cup \{e^v\}$
12: go to 6
13: end if
14: end while
15: $V_s \leftarrow \langle v_0, \ldots, v_n \rangle$ such that $v_i \in V \forall i \in [0, n]$ and $i < j \Rightarrow \delta_A^I[v_i] \leq \delta_A^I[v_j] \forall i, j \in [0, n]$
16: $v_a \leftarrow v_n$
17: $v_b \leftarrow v_0$
18: $\tilde{E}_a \leftarrow \{e^{v_a} \mid v_a \in e, e^{v_a} \in \tilde{E}\}$
19: $\tilde{E}_b \leftarrow \{e^{v_b} \mid v_b \in e, e^{v_b} \in \tilde{E}\}$
20: $\mathcal{T} \leftarrow \tilde{E}_a \times \tilde{E}_b$
21: while $\mathcal{T} \neq \emptyset$ do
22: $\mathcal{T} \leftarrow \mathcal{T} \setminus (e^{v_a}_{a}, e^{v_b}_{b})$ for any $(e^{v_a}_{a}, e^{v_b}_{b}) \in \mathcal{T}$
23: if $\max(\delta_A^I[v_a] - w(e_a), \delta_A^I[v_b] - w(e_b)) < \max(\delta_A^I[v_a], \delta_A^I[v_b])$ then
24: $\tilde{E} \leftarrow \tilde{E} \setminus \langle e_{a}^{v_a}, e_{b}^{v_b} \rangle \cup \langle e_{a}^{v_b}, e_{b}^{v_a} \rangle$
25: go to 15
26: end if
27: end while
28: $v_b \leftarrow v_{b+1}$
29: if $v_b = v_a$ then
30: $v_a \leftarrow v_{a-1}$
31: $v_b \leftarrow v_0$
32: if $v_a = v_0$ then
33: return $\Lambda$
34: else
35: go to 18
36: end if
37: end if
8.4. Experimental Results

Figure 8.1: Camera Network Layout and Relevance Function – The relevance function is depicted in the shaded areas, extending vertically in $1.5m < z < 2.0m$.

For each task, events of interest are points $\mathbf{p} \in \mathbb{R}^3$ generated randomly using $\lambda^{-1}R$ as a probability density function, where $\lambda = \iiint_{\mathbb{R}^3} R \, dx \, dy \, dz$. The detection probability for event $\mathbf{p}$ by camera node $n$ is $C_n(\mathbf{p})$. Camera nodes individually detect events and are assumed to propagate their data to the appropriate nodes for processing.

8.4.1 Task 1: Generic Multi-View Processing

The first simulation experiment models a generic task in which each event is processed by every combination of camera nodes which detects it. Processing an event charges one unit of processing load to the node to which the combination is assigned (i.e., the vertex in $\mathcal{H}_c$ which is the head of the hyperedge comprising the combination).

The experiment generates 10,000 random events, and assigns their processing to nodes according to $\Lambda$, the minimum maximum weighted indegree orientation of $\mathcal{H}_c$ approximated per Algorithm 3. For comparison, the same event detections are assigned using four other orientations of $\mathcal{H}_c$: the
optimal unweighted minimum maximum indegree orientation $U$, two random orientations $R_1$ and $R_2$, and a greedy orientation $G$ (hyperedges oriented in arbitrary order to the vertex with least indegree). Figure 8.2 shows the maximum and standard deviation of processing loads (with a mean of 1378.39) for each strategy.

The $\Lambda$ distribution yields both the least maximum load and the most consistent distribution of load over the network, with improvements of 5% and 22%, respectively, over the next best strategy tested.

**8.4.2 Task 2: Best-Pair Stereo Reconstruction**

The second simulation experiment models a best-pair stereo reconstruction task. Hypothetically, upon detection of an event, camera nodes estimate their pairwise coverage of the event, then reach network-wide consensus on the pair with best coverage; the best pair then proceeds to perform a dense 3D reconstruction of the event. Each estimation of pairwise coverage charges one unit of processing load to the assigned node, and each reconstruction charges five units of processing load to the assigned node for the best pair.

The experiment generates 2,000 random events and assigns their processing to nodes according
8.5 Conclusions

The coverage hypergraph is a generalization of previous models of camera network coverage topology which fully captures node-level coverage relationships. As such, it is a useful combinatorial structure for optimization in distributed smart camera applications. Simulated experiments demonstrate its application to optimizing the distribution of task processing load, by adapting and applying an algorithm for a related scheduling problem.

This model is conceptually simple, but shows much promise as a powerful tool given that it has a strong, reliable theoretical foundation and tractability with a large volume of well-studied optimization techniques.

to Λ, the minimum maximum weighted indegree orientation of $\mathcal{H}_C^\lambda$. Again, this is compared to the unweighted solution $U$, two random orientations $R_1$ and $R_2$, and a greedy orientation $G$. Figure 8.3 shows the maximum and standard deviation of processing loads (with a mean of 373.87) for each strategy.

Again, the Λ distribution yields both the least maximum load and the most consistent distribution of load over the network, with improvements of 13% and 35%, respectively, over the next best strategy tested.
Conclusions
9.1 Summary of Contributions

On the most fundamental level, the novel framework presented in Chapter 2 embodies the concept of sensor system coverage as the interface between the system and the task. This allows problems of coverage of any modality, or of multiple modalities, to be discussed using a common, unified language. In particular, the coverage function and relevance function provide a generic interface and tool set for formulating solutions to any of a family of coverage problems, independent of the sensing modality.

The explanation in Section 1.2 of how several coverage problems in the multi-camera context—view planning, view reconfiguration, view selection, and resource distribution—are related by this approach allows Chapter 3 to survey models proposed for this purpose and compare their merits in terms of fidelity, generality, and tractability, the first such exposition. One particularly notable observation in this chapter is that with any model of practical fidelity and generality, one is restricted to the same class of optimization techniques as with any model which can be expressed in the general framework of Chapter 2.

Generalizing from the geometric models in Section 3.2, a set of criteria is derived for visual coverage, and a novel high-fidelity coverage model for multi-camera systems is developed in Chapter 4, with explicit formulation in the framework of Chapter 2. The aforementioned observation about optimization is exploited to produce a highly general formulation, which, along with the task-oriented “tricks” of Section 4.6 and the extensibility of the model demonstrated by Section 7.3, allows an unprecedented variety of multi-camera vision tasks to be represented and evaluated. While the fidelity of the model is not shown to be quantitatively optimal, it meets all of the qualitative criteria discussed in Section 3.2 (except in the area of random occlusion, which is not included).

The fidelity and generality of the model are supported experimentally by the validation and comparisons in Chapter 5, demonstrating that the model is reliable in practical prediction of task perfor-
mance. Besides validating the objective function as a basis for optimization in approaching coverage
problems, this also demonstrates the utility of the model as a pure evaluation tool, allowing costly
implementation to be replaced with simulation in the design cycle for a wide range of systems.

Part II of this thesis presents three application cases making direct use of the framework and
model developed in Part I. In Chapter 6, the performance metric is used as part of an aggregate
objective function in the multi-objective optimization problem of optimal view sequence selection. In
Chapter 7, the model is modified for active triangulation-based three-dimensional imaging, and the
performance metric (both for the nominal solution and an error model solution) is used directly as the
fitness function for a particle swarm optimization algorithm for view planning. Finally, in Chapter 8,
a combinatorial heuristic optimization is applied to a topological derivative model to solve the a priori
camera network load distribution problem.

9.2 Future Research Directions

A large subset of multi-camera system applications which is not adequately handled by the coverage
model of Chapter 4 are those employing three-dimensional vision based on stereo (or general multi-
camera) vision [6]. Although $k$-coverage is modeled, an additional criterion for the baseline(s) between
cameras [110] would be required at the sensor system level—i.e., in (2.6)—to properly quantify per-
formance. Developing such a model would be relatively straightforward, as with the modified model
presented in Section 7.3, but no application case in this thesis affords an opportunity for validation.

It is assumed in Chapter 4 that scene illumination is an external design concern, and it is thus not
modeled. In Chapter 7, it is impossible to separate illumination (a laser line, in this case) from the model
in active vision, as the sensor is explicitly bistatic. However, despite the usual treatment of traditional
vision systems as passive sensors, in reality their coverage invariably depends on illumination. In
many applications, illumination is an explicit part of the design, as with active vision, so an extension
of the coverage model to account for it would offer value in fidelity and generality. Tarabanis et
al. [27] suggest that such a model could be derived from the wide body of knowledge in photometry
and radiometry. A major challenge, and perhaps the reason this remains an open problem in the
general case, is in optimizing the tradeoff among fidelity, generality, and tractability; the framework
of Chapter 2 may offer some leverage.

The coverage model of Chapter 4 lacks a random occlusion criterion. As shown by Mittal and
Davis [43], this is an involved topic of its own, and has been considered beyond the scope of this
thesis, as the associated real-world applications are primarily in well-controlled industrial settings.
However, such a criterion is important for a significant subset of multi-camera system applications.
Conveniently, the form of a probabilistic model interacts well with the bounded model form of the
framework of Chapter 2.

Where the coverage model of Chapter 4 is used in the context of an existing physical multi-camera
system, as in Chapter 6, it requires full metric calibration to accurately model the geometry of the
system. Unfortunately, this is often prohibitive in many application areas. An interesting possibility
which could be explored for such cases is the development of an equivalent coverage model based on
affine geometry, which is generally easier to recover from a physical camera system. Such a model
would exhibit reduced fidelity, but still potentially offer adequate results for certain applications.
This thesis has employed the framework of coverage introduced in Chapter 2 primarily in an open-loop, model-based context, using a priori knowledge or design parameters to construct the model. Chapter 6 introduces a limited incorporation of visual information in a closed loop by using the images themselves to obtain the pose of a target to generate the relevance function. Many online applications of view reconfiguration, view selection, and resource distribution could benefit from (and indeed, may not be possible without) further use of rich visual information feedback, specifically, where uncertainty in the model parameters might be reduced over time by its acquisition.

The particular representation of the coverage model was deliberately chosen to ease future integration of probabilistic information, including a random occlusion criterion. The value of the coverage function or relevance function at a point in the stimulus space can be modulated by probability distributions describing the uncertainty in various factors—which are considered deterministic in this thesis—without altering the mathematical relationship of these functions to the rest of the framework or any optimization methods. In other words, there is no fundamental difference, from the perspective of evaluation and optimization, between a coverage model that does not incorporate uncertainty and one that does.
Part III

Appendix
Mathematical Background

A.1 Motion of Rigid Bodies

Rigid transformations are used extensively throughout this thesis in the form of pose. This section reviews the topic, establishes notation and conventions, and presents some useful basic operations.

A.1.1 Rotation Formalisms

The set of proper rotations in a three-dimensional Euclidean space is the special orthogonal group SO(3). Various formalisms exist to express a rotation $\mathbf{R} \in \text{SO}(3)$ as a mathematical transformation. A rotation may be uniquely described by a minimum of three parameters; though some of the following representations use more, each still only has three degrees of freedom.

Rotation Matrix

Every rotation maps one orthonormal basis of $\mathbb{R}^3$ to another. Thus, as with any linear transformation, every rotation can be represented by a matrix. A proper rotation is represented by a $3 \times 3$ orthogonal matrix with determinant 1. These matrices form a group $\text{SO}(3)$ under matrix multiplication, which corresponds to composition of rotations (thus, the group of rotations under composition is isomorphic to $\text{SO}(3)$).

Given a rotation matrix $\mathbf{R}$, a numerically stable means of obtaining the corresponding unit quaternion $\mathbf{q} = (a, b, c, d)$ is

$$
a = \frac{1}{2} \sqrt{1 + R_{11} + R_{22} + R_{33}} \quad (A.1)$$

$$
b = \pm \frac{1}{2} \sqrt{1 - R_{11} - R_{22} - R_{33}} \quad (A.2)$$

$$
c = \pm \frac{1}{2} \sqrt{1 - R_{11} + R_{22} - R_{33}} \quad (A.3)$$

$$
d = \pm \frac{1}{2} \sqrt{1 - R_{11} - R_{22} + R_{33}} \quad (A.4)$$

where $b$ has the sign of $R_{32} - R_{23}$, $c$ has the sign of $R_{13} - R_{31}$, and $d$ has the sign of $R_{21} - R_{12}$.
Axis-Angle

Every nontrivial proper rotation fixes an axis of rotation \([158]\), and acts as an ordinary 2-dimensional rotation in the plane orthogonal to this axis.

Given a rotation of \(\theta\) about the axis defined by a vector \(\mathbf{v}\), the corresponding unit quaternion \(\mathbf{q} = (a, b, c, d)\) is defined by

\[
\begin{align*}
    a &= \cos \frac{\theta}{2} \\
    (b, c, d) &= \sin \frac{\theta}{2} \mathbf{\hat{v}}
\end{align*}
\]  

(A.5) \hspace{1cm} (A.6)

where \(\mathbf{\hat{v}}\) is a unit vector in the direction of \(\mathbf{v}\).

Euler Rotations

Any rotation may be expressed as a combination of principal rotations (rotations about the axes of the coordinate system). For rotations by angle \(\theta\) about the \(x\), \(y\), and \(z\) axes, respectively, the corresponding rotation matrices are:

\[
\begin{align*}
    \mathbf{R}_x(\theta) &= \begin{bmatrix} 1 & 0 & 0 \\ 0 & \cos \theta & -\sin \theta \\ 0 & \sin \theta & \cos \theta \end{bmatrix} \\
    \mathbf{R}_y(\theta) &= \begin{bmatrix} \cos \theta & 0 & \sin \theta \\ 0 & 1 & 0 \\ -\sin \theta & 0 & \cos \theta \end{bmatrix} \\
    \mathbf{R}_z(\theta) &= \begin{bmatrix} \cos \theta & -\sin \theta & 0 \\ \sin \theta & \cos \theta & 0 \\ 0 & 0 & 1 \end{bmatrix}
\end{align*}
\]  

(A.7) \hspace{1cm} (A.8) \hspace{1cm} (A.9)

General rotations may be expressed as a product of some combination of three these matrices, wherein no two adjacent matrices express rotations about the same axis. However, since the multiplication of \(3 \times 3\) orthogonal matrices does not commute, the order in which the principal rotations are applied is significant, and is given by convention. For example, a rotation with Euler angles \(\psi\), \(\phi\), and \(\theta\) in the \(zyx\) convention is represented by the rotation matrix \(\mathbf{R}_z(\theta)\mathbf{R}_y(\phi)\mathbf{R}_x(\psi)\). The twelve possible conventions are \(xzx, xxy, yxy, yzy, zyz, xxz, xzy, xyx, yxz, zyx, zyx,\) and \(zxy\).

Quaternion

The unit quaternion is the preferred representation of a rotation. It offers a simple composition operation, greater numerical stability than the rotation matrix, and avoids the gimbal lock singularities inherent in Euler angles.

Given the unit quaternion \(\mathbf{q} = (a, b, c, d)\), the corresponding rotation is applied to a vector \(\mathbf{p} \in \mathbb{R}^3\), yielding the rotated vector \(\mathbf{p}'\), by

\[
\mathbf{p}' = \mathbf{q}\mathbf{p}\mathbf{q}^{-1}
\]  

(A.10)
where \( q^{-1} = (a, -b, -c, -d) \) is the conjugate of \( q \).

The equivalent \( 3 \times 3 \) rotation matrix is

\[
R = \begin{bmatrix}
1 - 2c^2 - 2d^2 & 2bc - 2ad & 2bd + 2ac \\
2bc + 2ad & 1 - 2b^2 - 2d^2 & 2cd - 2ab \\
2bd - 2ac & 2cd + 2ab & 1 - 2b^2 - 2c^2
\end{bmatrix}
\]  

The equivalent axis-angle rotation is a rotation of \( \theta = 2 \arccos a \) about the direction of \( (b, c, d) \).

The Euler angles in the \( zyx \) convention are given by

\[
\begin{align*}
\theta &= \text{atan2}(2(cd - ab), 1 - 2(b^2 + c^2)) \\
\phi &= -\arcsin(2(ac + bd)) \\
\psi &= \text{atan2}(2(bc - ad), 1 - 2(c^2 + d^2))
\end{align*}
\]

where \( \text{atan2} \) is the two-argument arctangent (A.19).

### A.1.2 Rigid (Euclidean) Transformations

A rigid transformation, or pose, is a transformation from a Euclidean space to itself preserving the distances between every pair of points (thus an isometry). The set of proper rigid transformations in a three-dimensional Euclidean space is called the special Euclidean group \( \text{SE}(3) \).

A pose \( P \) consists of a rotation \( R \) and a translation of the origin \( T \). Acting on a vector \( p \), it produces a transformed vector \( P(p) \) of the form

\[
P(p) = Rp + T.
\]  

**Inversion**

The inverse of a pose reverses this mapping, and is given by

\[
P^{-1}(p) = R^{-1}p - R^{-1}T.
\]

**Composition**

A succession of poses \( P_2(P_1(p)) \) can be composed into a single pose, denoted \( P_1 \circ P_2(p) \), as

\[
\begin{align*}
P_2(P_1(p)) &= R_2(R_1p + T_1) + T_2 \\
P_1 \circ P_2(p) &= R_2R_1p + (R_2T_1 + T_2)
\end{align*}
\]

Note the use of a left composition convention. This is more intuitive notation for chaining multiple poses: given \( P_{AB} \) mapping from frame \( A \) to frame \( B \), and \( P_{BC} \) mapping from frame \( B \) to frame \( C \), the composition \( P_{BC}(P_{AB}(p)) = P_{AB} \circ P_{BC}(p) \) maps \( p \) from coordinate system \( A \) to coordinate system \( C \), and can be written simply \( P_{AC}(p) \), using the source frame of the first subscript and the destination frame of the last subscript.
A.2 Hypergraphs

A hypergraph $\mathcal{H}$ is a pair $\mathcal{H} = (V, E)$, where $V$ is a set of vertices, and $E$ is a set of non-empty subsets of $V$ termed hyperedges. If $\mathcal{P}(V)$ is the power set of $V$, then $E \subseteq \mathcal{P}(V) \setminus \emptyset$.

A weighted hypergraph $\mathcal{H} = (V, E, w)$ also includes a weight function over its hyperedges $w : E \to \mathbb{R}^+$. An unweighted hypergraph may be interpreted as a weighted hypergraph for which $w(e) = 1$ for all $e \in E$.

A.2.1 Degree

The degree of a vertex in $\mathcal{H}$, denoted $\delta_\mathcal{H}(v)$ for some $v \in V$, is the total weight of hyperedges incident to the vertex:

$$\delta_\mathcal{H}(v) = \sum_{e \in E} \begin{cases} w(e) & \text{if } v \in e \\ 0 & \text{otherwise} \end{cases}$$

(A.18)

A.2.2 Directed Hypergraphs

Following the definition of Frank et al. [159], a directed hypergraph is a pair $\mathcal{D} = (V, \vec{E})$, where $\vec{E}$ is a set of hyperarcs; a hyperarc is a hyperedge $e \subseteq V$ with a designated head vertex $v \in V$, denoted $e^v$. The remaining vertices $e \setminus v$ are called tail vertices. Two additional notions of vertex degree are defined: the indegree, $\delta^i_\mathcal{H}(v)$, is the total weight of hyperarcs of which $v$ is the head vertex, and the outdegree, $\delta^o_\mathcal{H}(v)$, is the total weight of hyperarcs of which $v$ is a tail vertex.

An orientation $\Lambda$ of an undirected hypergraph $\mathcal{H}$ has the same vertex and hyperedge sets (and the same weight function, if applicable), but assigns a direction (head vertex) to each hyperedge. In an orientation of a simple hypergraph, if $e^v \in \vec{E}$, then $e^u \in \vec{E}$ implies $u = v$ (that is, $e$ is unique). Therefore, the head vertex superscript is omitted in certain circumstances; for example, the weight of $e^v$ is denoted simply $w(e)$.

A.3 Miscellaneous

A.3.1 Two-Argument Arctangent

The two-argument arctangent function, originally introduced in computer programming languages, is a variation of the arctangent function which distinguishes between opposite directions. For any $x, y \in \mathbb{R}$ not both equal to zero, $\text{atan2}(x, y)$ is the angle in radians between the positive $x$-axis of the plane and the vector $(x, y)$.

$$\text{atan2}(y, x) = \begin{cases} \text{arctan} \left( \frac{y}{x} \right) & x > 0 \\ \text{arctan} \left( \frac{y}{x} \right) + \pi & y \geq 0, x < 0 \\ \text{arctan} \left( \frac{y}{x} \right) + \pi & y < 0, x < 0 \\ \frac{\pi}{2} & y > 0, x = 0 \\ -\frac{\pi}{2} & y < 0, x = 0 \\ \text{undefined} & y = 0, x = 0 \end{cases}$$

(A.19)
Geometry of Computer Vision

B.1 Image Formation

A correspondence between three-dimensional points in a global coordinate frame and their two-dimensional image projections requires a chain of three transformations [110]:

1. a $\mathbb{R}^3 \rightarrow \mathbb{R}^3$ transformation from the global frame to the camera’s local frame,
2. a projective $\mathbb{R}^3 \rightarrow \mathbb{R}^2$ transformation from the camera frame to the image plane, and
3. a $\mathbb{R}^2 \rightarrow \mathbb{R}^2$ coordinate transformation from image plane coordinates to pixel coordinates.

The first transformation is simply the inverse of the camera’s pose, so the remainder of this section will focus on deriving the second and third, assuming that three-dimensional points are in the camera’s local coordinate frame.

B.1.1 The Pinhole Model

The pinhole camera model is a simplified ideal mathematical model of an imaging system. By assuming a thin lens and an infinitely small (point) aperture, optical concerns are neglected, and image formation is described purely by perspective projection.

Figure B.1 shows the geometry of the pinhole camera model. The optical axis intersects the focal plane at the optical center $O$, and the image plane, which is separated from the focal plane by the focal length $f$, at the principal point $R$. By similar triangles, the coordinates of a point $p = (p_x, p_y, p_z)$ and its image $q = (q_u, q_v)$ are related by

$$
q = \begin{bmatrix}
-f \frac{p_x}{p_z} \\
-f \frac{p_y}{p_z}
\end{bmatrix}^T,
$$

which is a $\mathbb{R}^3 \rightarrow \mathbb{R}^2$ transformation known as a perspective projection.

---

¹In optics, a lens whose thickness (distance along the optical axis between the two surfaces of the lens) is negligible with respect to the focal length. This approximation simplifies calculations involving rays passing through the lens.
B.1. Image Formation

The negative signs on \( f \) cause the image of \( \mathbf{p} \) to appear inverted. This is corrected by inverting the \( u \) and \( v \) axes, which is equivalent to placing the image plane at \( z = f \) rather than \( z = -f \), so that (B.1) becomes

\[
\mathbf{q} = \begin{bmatrix} \frac{p_x}{f}, \frac{p_y}{f}, \frac{p_z}{f} \end{bmatrix}^T.
\]  

(B.2)

B.1.2 Sensor Imaging

In a digital camera, the image plane is physically realized by a finite, discrete image sensor (e.g. CCD, CMOS). This introduces two additional considerations toward a practical imaging model.

First, the image plane is bounded by the dimensions of the sensor’s imaging surface, so that not every \( \mathbf{p} \in \mathbb{R}^3 \) generates an image within the image plane. The angles subtended by the horizontal and vertical extents of the sensor, with respect to the optical center, are called the apex angles. They induce a quadrilateral pyramid called the field of view (Figure B.2), with its apex at the optical center and extending infinitely along the optical axis, defining the subset of \( \mathbb{R}^3 \) in which points have a corresponding image.

Second, the origin of the image coordinate system generally does not coincide with \( \mathbf{R} \), and, as the sensor is composed of discrete light-sensitive elements, image coordinates are expressed in units of pixels rather than the units of length used in the three-dimensional space. This requires an additional \( \mathbb{R}^2 \to \mathbb{R}^2 \) coordinate transformation. Given \( s_u \) and \( s_v \), respectively, the physical width and height of a pixel on the sensor, pixel coordinates can be related to units of length in each direction. With the image coordinates (in pixels) of the principal point \( \mathbf{r} = (r_u, r_v) \), (B.2) is expressed in pixel coordinates.
B.1. Image Formation

Figure B.2: Field of View – The quadrilateral pyramid extending along the optical axis to infinity (shown truncated), defined by the horizontal and vertical apex angles $\alpha_h$ and $\alpha_v$.

Figure B.3: Pixel Coordinates – The origin of the image in pixel coordinates is at the top left corner of the image. Pixel distances are related to units of length by the physical pixel dimensions $s_u$ and $s_v$.

as

$$q = \left[ \frac{f}{s_u} p_x + r_u, \frac{f}{s_v} p_y + r_v \right]^T.$$  \hspace{1cm} (B.3)

Pixel skew is neglected in (B.3), an assumption common to virtually all practical cases.

Note that $r$ is not, in general, precisely located at the physical center of the sensor; in fact, many cameras are deliberately designed this way. With this so-called oblique projection, splitting the apex angles at the optical axis may simplify some calculations. The horizontal angles are

$$\alpha_l = 2 \arctan \left( \frac{r_u s_u}{2f} \right)$$  \hspace{1cm} (B.4)

$$\alpha_r = 2 \arctan \left( \frac{(w - r_u)s_u}{2f} \right)$$  \hspace{1cm} (B.5)

where the subscripts $l$ and $r$ denote the left (negative $x$) and right (positive $x$) halves of $\alpha_h$, as seen from the camera. Similarly, the vertical angles are

$$\alpha_t = 2 \arctan \left( \frac{r_v s_v}{2f} \right)$$  \hspace{1cm} (B.6)

$$\alpha_b = 2 \arctan \left( \frac{(h - r_v)s_v}{2f} \right)$$  \hspace{1cm} (B.7)
where the subscripts \( t \) and \( b \) denote the top (negative \( y \)) and bottom (positive \( y \)) halves of \( \alpha_v \).

## B.2 Optical Effects

The simplified model of image formation presented in Section B.1 makes some assumptions about the optical system of the lens which do not strictly hold in practice. Most of the effects of these discrepancies are negligible, but two in particular can cause the real image to deviate considerably from the predictions of (B.3).

### B.2.1 Lens Distortion

The thin lens simplification used in Section B.1 allows for rectilinear projection: straight lines in the scene map to straight lines in the image. Real lens optics only approximate this simplified model, and thus introduce distortion to the image.

Two types of distortion, radial and tangential (or decentering), are dominant. Both can be corrected using Brown’s distortion model \[160\], from which the corrected image point \((u, v)\) is obtained from the distorted image point \((u_r, v_r)\) according to

\[
u = u_r + u_0(K_1r^2 + K_2r^4 + ...) + P_1(r^2 + 2u_0^2) + 2P_2u_0v_0(1 + P_3r^2 + ...) \tag{B.8}\]

\[
v = v_r + v_0(K_1r^2 + K_2r^4 + ...) + P_2(r^2 + 2v_0^2) + 2P_1u_0v_0(1 + P_3r^2 + ...) \tag{B.9}\]

where \( u_0 = u_r - r_u, v_0 = v_r - r_v \), and \( r = \sqrt{u_0^2 + v_0^2} \).

### B.2.2 Blur

Real lens optics necessarily have an aperture of finite diameter; otherwise, no energy would reach the sensor. This deviates from the pinhole camera model used in Section B.1, which assumes an infinitely small aperture. The effect is that a lens (in a given configuration) is focused precisely at one particular depth, the subject distance \( z_s \), at which points in the scene map to points in the image; points closer or farther away map to circles\(^2\) in the image (blur circles). The phenomenon is illustrated in Figure B.4.

The diameter of the blur circle imaging a scene point \( p \) is related to the depth \( p_z \) by

\[
c = A \frac{|p_z - z_s|}{p_z} \frac{f}{z_s - f} \tag{B.10}\]

where \( A \) is the effective aperture diameter and \( f \) is the effective focal length.

### B.3 Camera Calibration

Camera calibration is the process of recovering the parameters of the image formation model. This generally includes the intrinsic parameters, minimally the effective focal length \( f \) and the image coordinates of the principal point \( r \), as well as the extrinsic parameters, the pose of the camera with respect to a global coordinate frame.

\(^2\)More precisely, to a shape similar to that of the aperture, which commonly approximates a circle.
B.3. Camera Calibration

Figure B.4: Blur Circle – A point at the subject distance $z_S$ maps to a point in the image; points closer or farther away map to blur circles, whose diameters are related to the depths of the points.

B.3.1 Single Camera Calibration

Approaches in the vein of Zhang’s method [161] estimate the parameters using partial correspondence between image points and their three-dimensional scene counterparts. This is achieved using structured targets with easily localized features, such as the HALCON calibration plate shown in Figure 5.1. Generally, the process involves finding an optimal solution (the model parameters) to an overdetermined set of linear equations, using e.g. singular value decomposition.

The parameters of the lens distortion model described in Section B.2.1 can also be recovered from image data. Since the projective transformation (without distortion) preserves straight lines, given sets of image points whose corresponding three-dimensional scene points are known to lie on straight lines, a nonlinear optimization over the resultant (B.8) and (B.9) system can recover the parameters of the lines and the distortion coefficients simultaneously [162, 163].

B.3.2 Multi-Camera External Calibration

To obtain the relative poses of a pair of cameras $A$ and $B$ whose fields of view overlap, it suffices to obtain each pose relative to a calibration target $T$ imaged simultaneously, yielding $P_{AT}$ and $P_{BT}$, then compute $P_{AB} = P_{AT} \circ P_{TB}^{-1}$ by inversion and composition (see Section A.1.2). Multiple estimates may be bundle-adjusted to improve the fidelity of the final relative pose estimate.

Extending this to multiple cameras whose fields of view do not, in general, mutually overlap is not entirely trivial. With noisy pose estimates, composition aggregates error, so a means of reducing global error is necessary.

The first step is to construct a calibration graph $G_C = (V, E, w)$. The vertex set $V$ consists of the set of cameras as well as the set of calibration targets (or unique calibration target positions). An edge $E$ exists between vertices $i, j \in V$ if $P_{ij}$ is estimated directly, implying that $j$ represents a target and $i$ represents a camera with a view of the target, or vice versa. $G$ is undirected since $P_{ji} = P_{ij}^{-1}$ with no loss of accuracy. The weight $w(i,j)$ is some measure of the expected error in the estimate of $P_{ij}$, such as the reprojection error.
3 Theorem (Connected Vision Graph)

If the calibration graph \( G_C = (V_C, E_C) \) is connected, then the vision graph \( G_V = (V_V, E_V) \) with \( V_V \subset V_C \), as induced by the calibration task, is also connected.

Proof For any \( \{i, j\} \in E_C \), without loss of generality, \( i \) must represent a camera and \( j \) a target. For \( i, k \in V_C \), where \( i \) and \( k \) represent cameras, \( \{i, j\}, \{k, j\} \in E_C \) implies that cameras \( i \) and \( k \) both view target \( j \). Since \( G_V = H^2_C \), by (2.9), this further implies \( \{i, k\} \in E_V \). Suppose that \( G_V \) has at least two connected components, containing vertices \( V_{V1} \subset V_V \) and \( V_{V2} \subset V_V \), respectively. If \( G_C \) is connected, then there must exist some \( i \in V_{V1}, k \in V_{V2}, \) and \( j \in V_C \) such that \( \{i, j\}, \{k, j\} \in E_C \), in turn implying \( \{i, k\} \in E_V \) as previously shown, violating the assumption that \( V_{V1} \) and \( V_{V2} \) are the vertex sets of connected components of \( G_V \). Therefore, if \( G_C \) is connected, \( G_V \) is connected. ■

It is necessary that \( G_C \) be connected; this is achieved by placing calibration targets within mutual view of a sufficient set of pairs of cameras so that the resultant \( E \) minimally connects \( V \). Selecting some reference frame \( R \in V \), the pose \( P_{IR} \) of any camera represented by vertex \( i \in V \) is found by composing along the shortest path from \( i \) to \( R \) in \( G \), obtained e.g. via Dijkstra’s algorithm [164], inverting poses where necessary.

![Figure B.5: Calibration Graph Example – Vertices A, B, C, D, E, and F represent cameras, and vertices W, X, Y, Z, and R represent calibration targets. Target R is the reference frame.](image)

Consider the calibration graph shown in Figure B.5. In order to obtain the pose of camera \( A \) with respect to the reference frame \( R \), supposing that \( (A, W, C, Y, E, R) \) is the shortest path between \( A \) and \( R \),

\[
P_{AR} = P_{AW} \circ P_{CW}^{-1} \circ P_{CY} \circ P_{EY}^{-1} \circ P_{ER},
\]

where all of the relative pose estimates referenced are available directly. The global pose may be computed similarly for any camera vertex reachable from \( R \).

B.4 Active 3D Triangulation

The prevalent means of acquiring three-dimensional (range) information using a single camera involves the use of structured light: knowledge of the shape of the light being projected and its geometry with respect to the camera allows for the inference of the shape of the surface it is projected upon, based on how the surface affects its shape (triangulation). A common form uses a line laser source to project a “sheet of light,” which the camera views from an offset angle. As shown in Figure B.6,
any object passing through the light will have a distorted line clearly illuminated on its surface, and a cross-section profile can be inferred from the resulting image. Moving the object (or the camera and laser) through a transport motion and capturing successive profiles allows for a complete reconstruction of the exposed surface.

Figure B.6: Scanning by Active Triangulation – A three-dimensional model of the upper surface of the object is obtained by measuring the height along a series of cross-sections of the object, yielding a series of two-dimensional profiles which are combined into a three-dimensional point cloud according to the transport pitch.

Several image processing techniques exist for estimating the offset of a well-focused laser line to subpixel accuracy, allowing for high-fidelity scanning. Some industrial sensors, such as the SICK IVP Ranger, perform this step in dedicated hardware to greatly increase the rate of profile acquisition.

Calibration of an active triangulation scanner generally involves both correcting for lens distortion and finding the homography between the laser plane and the image plane [141].
Adolphus Simulation Environment

C.1 Introduction

Adolphus\(^1\) is a three-dimensional simulation environment for multi-camera systems. Its architecture is based upon the concept of coverage introduced in Chapter 2, and its primary coverage model for vision is a precise implementation of the model described in Chapter 4. The purpose of the software is to provide a platform for managing complex descriptions of multi-camera systems, environments, and tasks, and evaluating the various model functions accordingly. It provides a feature-rich application programming interface (API) in Python \([166]\), allowing for the development of complex experiments with external logic, and is highly extensible to allow modified sensor, sensor system, environment, and task models, as well as new types of physical objects, to be incorporated with ease. It also provides a graphical user interface (GUI) with 3D visualization. All of the experimental work in this thesis employs Adolphus.

C.1.1 Comparison with Other Simulation Environments

There exist numerous other examples of software simulating multi-camera systems, broadly speaking. The aim of this subsection is to address the question of why Adolphus is needed, and how its functionality might be adapted to these other tools in the future.

Simulation of multi-camera systems is the core of the “virtual vision” experimentation paradigm, promulgated in a number of recent publications by Qureshi, Terzopoulos, Starzyk, and their colleagues \([167, 168, 169, 170, 171, 172, 173]\), and originating in Qureshi’s Ph.D. work \([174]\). As of 2012, the software is released under an open source license. The primary application is in large surveillance networks, for which the availability of real (and especially controlled) data is obviously quite limited. Accordingly, the focus is on accurate simulation of agent behaviour and imaging, to provide a realistic surrogate for such data; at present, it is ill-suited for investigating most coverage problems.

\(^1\)So named in homage to Terry Gilliam’s film *The Adventures of Baron Munchausen* \([165]\), in which the character of the same name (portrayed by Charles McKeown), an associate of the Baron’s, is a rifleman with superhuman eyesight. Incidentally, a tentatively planned high-performance reimplementation in C++ has been dubbed Berthold, after another of the Baron’s associates who is the world’s fastest runner.
Robotics simulators such as Orbital 3D [175, 176] and Gazebo [177], along with robotics platforms such as Player/Stage [178], OpenRAVE [179] and ROS [180] which include or interface with simulation components, are capable of simulating multi-sensor systems of various modalities, including vision. In general, the modeling paradigm is well-aligned with the aims of Adolphus, in that a coverage model must be established—though usually implicitly, as a sensing model—in order to simulate sensor data. Adolphus has been developed separately during the evolution of the concepts in this thesis, in part to ensure absolute faithfulness in experiments, and in part to avoid encountering any external constraints on the paradigm. With a mature model in hand, the aforementioned simulators appear to be prime candidates for integration of explicit coverage modeling, particularly given that several of them offer plugin architectures with a thin interface.

The Vision System Designer software by SensorDesk [181] has some functional overlap with Adolphus in the sense that it evaluates the task-oriented coverage of camera systems, and offers quantitative metrics of the same along with a rich, interactive 3D visualization GUI. It is primarily targeted at manual high-fidelity view planning, and offers some realistic virtual imaging functionality for direct testing, sidestepping, to an extent, the need for a theoretical relation of system and task. This software is proprietary, and does not provide an API or any extensibility outside of object and sensor definitions.

C.2 Architecture

Adolphus represents the world as a set of SceneObject objects. The base SceneObject class inherits from both the Posable class, which provides it with properties and functionality relating to its pose in three-dimensional space, and the Visualizable class, which provides it with a visual representation in one or more Displays. Each SceneObject also has a (possibly empty) set of OcclusionTriangle objects.
associated with it, which are themselves Posable and Visualizable. These triangles, whose poses are relative to their parent SceneObject, form the occlusion triangle set \( \mathcal{T} \) as defined in Section 4.3.2.

The Camera object, a special subclass of SceneObject, implements the monocular coverage model described in Section 4.4. The Model object groups a set of Cameras, together with other SceneObjects (for geometric and occlusion purposes), and provides the multi-camera coverage model described in Section 4.5. The task model described in Section 4.3.3 is implemented by the Task object, which comprises a Posable discrete relevance function representation in the form of a PointCache object, as well as the task requirements.

The Experiment object provides an interface for managing a Model and a set of Tasks. It provides a three-dimensional Display, as well as a modular command interface for interacting with and modifying the model.

### C.2.1 Geometry and the Posable Class

Adolphus includes an efficient three-dimensional geometry module, which among other things provides the important Point, DirectionalPoint, and Pose classes. A Pose object consists of a Point and a Rotation, the latter converting to and from the various representations described in Section A.1.1.

The Posable class maintains two basic properties: a parent Posable, termed its mount, and a Pose relative to its mount. If no mount is specified, the relative pose of the Posable is also its absolute pose. Otherwise, the absolute pose of a Posable is computed by composition up the chain of mounts until a parent with no mount is reached.

### C.2.2 Robots

Robot is a special subclass of SceneObject which maintains a set of RobotLink objects (also a subclass of SceneObject), and manages their relative poses by forward kinematics based on a specification using Denavit-Hartenberg parameters [182]. This allows the positions of any complex set of interrelated objects constrained by kinematic linkages to be specified by a simple set of parameters, which is much more convenient than individually specifying the poses with six degrees of freedom.
C.2.3 Multi-Camera Coverage

The base Model, Camera, and Task classes collectively implement the coverage model described in Chapter 4. Of the five criteria comprising (4.3), the first four (field of view, resolution, focus, and view angle) are implemented directly in Camera, as they do not require the environment model. The method Camera.strength() takes a global-frame point and the requirement parameters from a Task object as arguments, and returns the pre-occlusion coverage value in [0, 1]. The Model class then implements the final criterion (deterministic occlusion) using all SceneObject objects’ triangle sets, and also implements the multi-camera \( k \)-coverage as given by (2.6) and (2.7).

To implement other coverage models, these three classes can be subclassed with relative ease, and a number of facilities are provided for common operations, including various derived properties of the camera projection model.

C.3 Obtaining the Software

C.3.1 Source Code

Adolphus is hosted on GitHub. At the time of publication, full Python source code, documentation, and example models are available at https://github.com/ezod/adolphus.

<table>
<thead>
<tr>
<th>Dependency</th>
<th>Min. Version</th>
<th>URL</th>
<th>Required For</th>
</tr>
</thead>
<tbody>
<tr>
<td>Python</td>
<td>2.6</td>
<td><a href="http://www.python.org">http://www.python.org</a></td>
<td>Core</td>
</tr>
<tr>
<td>PyYAML</td>
<td>3.09</td>
<td><a href="http://www.pyyaml.org">http://www.pyyaml.org</a></td>
<td>Core</td>
</tr>
<tr>
<td>Cython</td>
<td>0.14</td>
<td><a href="http://www.cython.org">http://www.cython.org</a></td>
<td>Core</td>
</tr>
<tr>
<td>setuptools</td>
<td>0.6.25</td>
<td><a href="http://pypi.python.org/pypi/setuptools">http://pypi.python.org/pypi/setuptools</a></td>
<td>Installation</td>
</tr>
<tr>
<td>Visual</td>
<td>5.72</td>
<td><a href="http://www.vpython.org">http://www.vpython.org</a></td>
<td>Visualization</td>
</tr>
<tr>
<td>PyGTK</td>
<td>2.22</td>
<td><a href="http://www.pygtk.org">http://www.pygtk.org</a></td>
<td>Tool Panel</td>
</tr>
<tr>
<td>Epydoc</td>
<td>3.0.0</td>
<td><a href="http://epydoc.sourceforge.net">http://epydoc.sourceforge.net</a></td>
<td>Documentation</td>
</tr>
</tbody>
</table>

C.3.2 License

Adolphus is free software: you can redistribute it and/or modify it under the terms of the GNU General Public License as published by the Free Software Foundation, either version 3 of the License, or (at your option) any later version.

Adolphus is distributed in the hope that it will be useful, but WITHOUT ANY WARRANTY; without even the implied warranty of MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE. See the GNU General Public License for more details.

The GNU General Public License is distributed with Adolphus, and can alternatively be retrieved at http://www.gnu.org/licenses.
D.1 Digital Cameras

Both the Advanced Control Systems Laboratory at the University of Windsor and Vista Solutions host numerous digital camera models, for varying purposes. Those used most prominently in the experimental work in this thesis are shown here with specifications.

D.1.1 Prosilica EC-1350

The Prosilica EC-1350 is a 1.4 megapixel CCD camera based on the IIDC/DCAM specification. The Advanced Control Systems Laboratory possesses eight of these cameras, of which six are the EC-1350C variant with a Bayer filter for color imaging (the remaining two cameras are monochrome only).

![Figure D.1: Prosilica EC-1350](image)

<table>
<thead>
<tr>
<th>Specification</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>Sensor Resolution</td>
<td>1360 × 1024 pixels</td>
</tr>
<tr>
<td>Sensor Size</td>
<td>1/2”</td>
</tr>
<tr>
<td>Pixel Size</td>
<td>4.65µm × 4.65µm</td>
</tr>
<tr>
<td>Lens Mount</td>
<td>C-mount</td>
</tr>
<tr>
<td>Full Resolution Frame Rate</td>
<td>18.5 fps</td>
</tr>
<tr>
<td>Interface</td>
<td>IEEE 1394A (FireWire)</td>
</tr>
</tbody>
</table>
D.1.2 NET iCube NS4133BU

The NET iCube is a compact form factor monochrome CCD camera based on the USB 2.0 standard. The Advanced Control Systems Laboratory possesses ten of these cameras, typically used as “smart cameras” in conjunction with the NET SV-0813V lenses and ASUS Eee Box computers.

![Figure D.2: NET iCube NS4133BU](image)

<table>
<thead>
<tr>
<th>Table D.2: NET iCube NS4133BU Specifications</th>
</tr>
</thead>
<tbody>
<tr>
<td>Sensor Resolution</td>
</tr>
<tr>
<td>Sensor Size</td>
</tr>
<tr>
<td>Pixel Size</td>
</tr>
<tr>
<td>Lens Mount</td>
</tr>
<tr>
<td>Full Resolution Frame Rate</td>
</tr>
<tr>
<td>Interface</td>
</tr>
</tbody>
</table>

D.1.3 SICK IVP Ranger E/D

The SICK IVP Ranger is a high-precision 3D range imaging camera designed for active triangulation with a line laser source. Vista Solutions provided the use of these cameras for several collaborative projects, including the work presented in Chapter 7.

![Figure D.3: SICK IVP Ranger E/D](image)

<table>
<thead>
<tr>
<th>Table D.3: SICK IVP Ranger D50 Specifications</th>
</tr>
</thead>
<tbody>
<tr>
<td>Sensor Resolution</td>
</tr>
<tr>
<td>Sensor Size</td>
</tr>
<tr>
<td>Pixel Size</td>
</tr>
<tr>
<td>Lens Mount</td>
</tr>
<tr>
<td>Max. 3D Profile Rate</td>
</tr>
<tr>
<td>Interface</td>
</tr>
</tbody>
</table>

Since triangulation is performed in specialized hardware in the camera itself, the “images” returned are in fact 1536 height values in a discrete range with 8192 possible values, reflecting a resolution of 1/16 pixel (the high-resolution interpolation algorithm is accurate to roughly 1/10 pixel).
D.2 Camera Lenses

The choice of lens greatly affects the characteristics of an imaging system. The models used most prominently in the experimental work in this thesis are shown here with specifications.

D.2.1 Computar M3Z1228C-MP

The Computar M3Z1228C-MP is a high-quality varifocal lens with low distortion. It is used in applications with a relatively close working range where the focus and aperture settings remain fixed. In typical applications, the maximum aperture ratio is used, limiting the depth of field but allowing accurate modeling of focus.

Figure D.4: Computar M3Z1228C-MP

<table>
<thead>
<tr>
<th>Table D.4: Computar M3Z1228C-MP Specifications</th>
</tr>
</thead>
<tbody>
<tr>
<td>Focal Length</td>
</tr>
<tr>
<td>Max. Aperture Ratio</td>
</tr>
<tr>
<td>Max. Sensor Size</td>
</tr>
<tr>
<td>Mount Type</td>
</tr>
</tbody>
</table>

D.2.2 Computar H10Z1218-MP

The Computar H10Z1218-MP is a high-quality varifocal lens with low distortion and motorized focus, zoom, and iris controlled by an ImageLabs V1LC controller. It is used in applications with a relatively close working range where the focus and aperture settings are variable.

Figure D.5: Computar H10Z1218-MP
Table D.5: Computar H10Z1218-MP Specifications

<table>
<thead>
<tr>
<th>Specification</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>Focal Length</td>
<td>12mm - 120mm</td>
</tr>
<tr>
<td>Max. Aperture Ratio</td>
<td>1 : 1.8</td>
</tr>
<tr>
<td>Max. Sensor Size</td>
<td>6.4mm × 4.8mm</td>
</tr>
<tr>
<td>Mount Type</td>
<td>C-mount</td>
</tr>
</tbody>
</table>

D.2.3 NET SV-0813V

The NET SV-0813V lens is a compact varifocal lens with low distortion. It is used in applications with medium to far working range where the focus and aperture settings remain fixed.

Table D.6: NET SV-0813V Specifications

<table>
<thead>
<tr>
<th>Specification</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>Focal Length</td>
<td>7mm - 8mm</td>
</tr>
<tr>
<td>Max. Aperture Ratio</td>
<td>1 : 1.3</td>
</tr>
<tr>
<td>Max. Sensor Size</td>
<td>2/3&quot;</td>
</tr>
<tr>
<td>Mount Type</td>
<td>C-mount</td>
</tr>
</tbody>
</table>

D.3 Miscellaneous

D.3.1 Mitsubishi RV-1A

The Mitsubishi RV-1A is a six-axis robotic arm typical of manipulators found widely in industry. Common motion planning, programming, communication, and safety features are present. The unit itself is supported by a Mitsubishi CR1 controller and teach pendant. Several of the experiments presented in this thesis involve communication with the robot via RS-232 serial link from Adolphus and/or HALCON [115].

![Figure D.6: Mitsubishi RV-1A](image)

D.3.2 ASUS Eee Box EB1007-B0410

The ASUS Eee Box is a compact form factor general-purpose PC based on the Intel Atom CPU. This device was selected for smart camera network research owing to architecture support for Linux, Python [166], HALCON [115], and the NET iCube drivers, and as they are more powerful than typical general-purpose embedded computers, can be applied to the full range of common smart camera tasks.
The Advanced Control Systems Laboratory possesses ten of these devices, named Overlord, Zergling, Hydralisk, Lurker, Mutalisk, Devourer, Guardian, Defiler, Queen, and Ultralisk.

Figure D.7: ASUS Eee Box

### Table D.7: ASUS Eee Box EB1007-B0410 Specifications

<table>
<thead>
<tr>
<th>Component</th>
<th>Specification</th>
</tr>
</thead>
<tbody>
<tr>
<td>Processor</td>
<td>Intel Atom D410 1.66GHz</td>
</tr>
<tr>
<td>Processor Cache</td>
<td>512KB L2 Cache</td>
</tr>
<tr>
<td>Main Memory</td>
<td>1GB DDR2</td>
</tr>
<tr>
<td>Hard Drive</td>
<td>250GB 5400RPM SATA</td>
</tr>
<tr>
<td>Graphics</td>
<td>Intel GMA 3150</td>
</tr>
<tr>
<td>Power Supply</td>
<td>40W Power Adapter</td>
</tr>
</tbody>
</table>
Glossary of Terms

aperture
In an optical system, the opening through which light passes. 44, 85, 113, 116, 126

apex angle
One of the angles (horizontal or vertical) subtended by the field of view. 19, 25, 114, 115

baseline
In a stereo camera pair, the horizontal distance between the optical centers of the two cameras. 51

blur circle
An optical spot caused by a cone of light rays from the lens not coming into perfect focus when imaging a point source. 19, 20, 26, 46, 47, 49, 81, 82, 116

complex feature
An atomic feature composed of multiple (or a continuous range of) point features. 51

connected component
In graph theory, a subgraph of an undirected graph in which any two vertices are connected to each other by paths, and which is connected to no additional vertices. 31, 37

coverage function
A function dependent on the sensor system, environment, and task mapping the stimulus space to a bounded numeric range (see Definition 1). 10–16, 18, 20, 24, 26–28, 47, 50–53, 57–59, 64, 66–68, 80, 82, 84, 89, 99, 105, 107

coverage hull
The subset of points in the stimulus space with nonzero coverage, according to a coverage function. 11, 14, 36

coverage model
A set of parameters modeling the sensor system, environment, and task, along with a specification of their relations, defining a coverage function in terms of the parameters. 7, 8, 11, 14, 15, 18–22, 24, 26–29, 32, 42, 45, 46, 51–54, 57–59, 64–66, 75, 77–80, 90, 105–107, 120, 122, 123
depth of field
A range along the focal axis of a camera in which objects appear acceptably in focus (i.e. points map to acceptably small blur circles), according to some criterion. 19, 25, 26, 44, 126

deterministic occlusion
A model of occlusion caused by static objects or, more generally, objects with known dynamics. 20, 27, 45, 50, 53, 57, 58, 83, 122

directional point
A point in directional space. 43, 50, 51, 70

directional space
A space consisting of three-dimensional Euclidean space plus two additional degrees of freedom for direction (see Definition 4), serving as the visual stimulus space under the assumption of rotational invariance about the optical axis. 42, 43, 51

extrinsic parameter
A parameter of a sensor model describing a value of its pose in Euclidean space. 12, 44, 99, 116

field of view
The quadrilateral pyramid enclosing the subspace of $\mathbb{R}^3$ within which points project onto the image sensor. 19, 22–25, 27, 33, 46–49, 51, 55, 57, 58, 77, 114, 117, 122

focal length
In the pinhole camera model, the distance from the focal point to the image plane along the optical axis. Refers to the effective focal length of a complex optical system. 44, 113, 116

focal plane
In the pinhole camera model, the plane in which the theoretical thin lens and aperture point lie; the principal plane. Coincides with the $x$-$z$ plane of the camera coordinate system. 113

focus
As applied to a digital camera and its optical system, the property of sharpness in the imaging of an object, quantified by the blur circle diameter; dependent on the depth of the object and optical properties. 19, 20, 22, 26, 29, 47, 50, 55, 61, 81, 122, 126

height resolution
The smallest change in height (in the direction of projection of the laser) detectable by a range camera. Usually measured in terms of units of length per pixel, and may require multiplication by a subpixel-accuracy factor; dependent on the depth of the object being imaged, the view angle with respect to the laser projection, and optical and sensor properties. 79–83

hyperedge
An edge connecting any number of vertices in a hypergraph. 16, 33, 99, 101, 111, 112
Glossary of Terms

hypergraph
A generalization of the graph, in which an edge (hyperedge) can connect any number of vertices. 16, 32, 33, 36, 97–99, 103, 111, 112

image plane
In the pinhole camera model, the plane on which the three-dimensional scene is projected through the camera aperture (optical center). Parallel to the \( x-y \) plane (focal plane) in the camera coordinate system. 43, 46, 58, 113, 114, 119

intrinsic parameter
A parameter of a sensor model describing some property internal to the sensor. 12, 44, 86, 116

jitter
In view selection, a phenomenon whereby calibration and estimation errors cause high-frequency transitioning between two views, resulting in an undesirable view sequence. 70, 72, 75

laser plane
In active 3D vision based on triangulation from structured light in the form of a projected line (often by a laser diode with line projection optics), the plane in which the line is projected. 80, 81, 84–86, 89, 91, 119

occlusion
The effect of an opaque object obstructing the line of sight to a point beyond the object from the viewpoint. 45, 53, 77, 80, 90, 91

optical axis
The imaginary line which defines the path along which light propagates through the lens system of a camera; the principal axis or principal ray. Coincides with the axis of rotational symmetry. Considered the positive \( z \)-axis of the camera coordinate system. 19, 25, 26, 42, 47, 55, 80, 82, 85, 86, 113–115

optical center
The focal point of the optical model of the camera. In the pinhole camera model, coincides with the aperture point. Considered the origin of the camera coordinate system. 19, 20, 25–27, 47, 50, 58, 86, 113, 114

pan-tilt-zoom camera
A camera capable of modifying the direction of its optical axis and its focal length (zoom) by way of motorized mount and lens. 5

pinhole camera model
Mathematical description of the projection of a 3D point onto a 2D image plane in an ideal pinhole camera (point aperture, no lens effects). 24, 25, 45, 113, 116

point feature
A visual stimulus originating at a point in \( \mathbb{R}^n \). 18–20, 26, 31, 42, 46, 48, 50–52
pose
A rigid Euclidean transformation in SE(n). 4, 6, 12, 14, 19, 22, 24, 33, 43–45, 47, 50, 52, 54, 55, 59, 66, 70–72, 74, 75, 80, 82, 85, 88, 89, 106, 109, 111, 113, 116–118, 121, 122

positioning error
The uncertainty in the actual pose of an object (e.g., a camera) in realization of a prescribed pose. 24

power set
The set of all subsets of a set, including the empty set and the set itself. symbol 16

principal point
In the pinhole camera model, the point at which the optical axis intersects the image plane; the image center. symbol 44, 113, 114, 116

random occlusion
A probabilistic model of occlusion caused by stochastic objects with uncertain occupancy and/or dynamics, such as humans. 8, 20, 21, 27, 28, 41, 45, 105–107

relevance function
A function mapping the task point set to a bounded numeric range based on relevance to the task (see Definition 3). 13–16, 18, 20, 22, 24, 27–29, 47, 51, 52, 54, 59, 66–68, 70, 81, 98, 99, 105–107, 122

resolution
The smallest change detectable by a sensor in the quantity that it measures. For digital cameras, usually measured in terms of units of length per pixel; dependent on the depth of the object being imaged and optical and sensor properties. 19, 20, 22, 24–26, 29, 44, 46, 47, 49, 55, 57, 58, 61, 80–82, 122

scanning density
The density of points in a point cloud generated by a 3D scan. In active triangulation, dependent on the transport pitch and the horizontal imaging resolution. 79, 81, 82, 91

self-occlusion
The phenomenon whereby some part of a (complex) object interrupts the ray from a feature on some other part of its surface to a camera. 20, 26, 47, 58

stimulus
An atomic unit of information perceived by a sensor system. 3, 5, 7, 10–14, 16, 18, 19, 47, 51, 80, 98

stimulus space
The space in which vectors represent stimuli detectable by a sensor system. 7, 10, 11, 13, 14, 16, 18, 22, 42, 51, 52, 65, 83, 107
subject distance
In an optical system, the distance at which objects are projected onto the image plane in focus.
19, 26, 44, 116

task
A process to be carried out by a sensor system online, one or more of which comprise the end
objective of the system. 2–8, 10, 11, 13, 15, 16, 18, 20–22, 26–30, 33–36, 42, 44, 46–48, 51–54, 57,
59, 61, 64, 67, 70, 75, 78–82, 84, 85, 89–92, 94, 97–99, 101, 102, 105, 118, 120–122

task point set
The subset of points in the stimulus space with nonzero relevance, according to a relevance
function specified by a task. 14, 15

view
The set of configuration parameter values of a multi-camera system; the instance of a vision
sensor system model (equivalent to a viewpoint for single-camera systems). 4, 5, 30, 32, 33, 38,
39, 46, 52, 64–70

view angle
The angle between the surface normal of a stimulus point located on a surface and the ray from
the camera’s principal point through the stimulus point. 19–23, 26–28, 46, 47, 50, 52, 56, 58, 61,
80–83, 99, 122

view sequence
A discrete ordered set of views, each associated with an interval of time. 5, 64–68

viewing frustum
The pyramidal frustum of visual coverage obtained by truncating the field of view with depth
limits imposed by resolution and/or focus constraints. 19, 24–26

viewpoint
The combined set of intrinsic and extrinsic parameter values of a camera; the instance of a vision
sensor model. 2, 4, 5, 19, 20, 22–24, 28, 78

vision graph
A graph encapsulating the pairwise coverage overlap topology of a set of cameras. 30–33, 35,
37, 59, 69, 74, 118
Theorems and Definitions

List of Theorems

Theorem 1 $E_C$ is an Abstract Simplicial Complex .......................... 16
Theorem 2 Bounded Aggregate Objective Function ............................ 68
Theorem 3 Connected Vision Graph .................................................. 118

List of Definitions

Definition 1 Coverage Function ......................................................... 11
Definition 2 Coverage Hull ............................................................... 11
Definition 3 Relevance Function ......................................................... 14
Definition 4 Directional Space ......................................................... 43
Definition 5 View Sequence ............................................................. 66
Bibliography


Bibliography


Vita Auctoris

Aaron Mavrinac was born in 1982 in Windsor, Ontario. He received the degree of Bachelor of Applied Science in Electrical Engineering from the University of Windsor in 2005. After a year working in automotive manufacturing controls, he returned to the University of Windsor for graduate studies under Dr. Xiang Chen. There, he received the degree of Master of Applied Science in 2008, focusing on geometric and topological aspects of camera networks. His further work in this area culminated in this doctoral dissertation in 2012. He has accepted a position with Robert Bosch LLC’s Research and Technology Center in Palo Alto, California, in the Autonomous Technologies and Robotics Group.