Sampling-Based Coverage Path Planning for Complex 3D Structures

by

Brendan J. Englot

S.B., Massachusetts Institute of Technology (2007)
S.M., Massachusetts Institute of Technology (2009)

Submitted to the Department of Mechanical Engineering
in partial fulfillment of the requirements for the degree of

Doctor of Philosophy

at the

MASSACHUSETTS INSTITUTE OF TECHNOLOGY

September 2012

Author .....
Department of Mechanical Engineering
August 10, 2012

Certified by .....
Franz S. Hover
Finmeccanica Career Development Professor of Engineering
Thesis Supervisor

Accepted by .....
David E. Hardt
Chairman, Department Committee on Graduate Students

Sampling-Based Coverage Path Planning for Complex 3D Structures

by

Brendan J. Englot

Submitted to the Department of Mechanical Engineering
on August 10, 2012, in partial fulfillment of the
requirements for the degree of
Doctor of Philosophy

Abstract

Path planning is an essential capability for autonomous robots, and many applications impose challenging constraints alongside the standard requirement of obstacle avoidance. Coverage planning is one such task, in which a single robot must sweep its end effector over the entirety of a known workspace. For two-dimensional environments, optimal algorithms are documented and well-understood. For three-dimensional structures, however, few of the available heuristics succeed over occluded regions and low-clearance areas. This thesis makes several contributions to sampling-based coverage path planning, for use on complex three-dimensional structures.

First, we introduce a new algorithm for planning feasible coverage paths. It is more computationally efficient in problems of complex geometry than the well-known dual sampling method, especially when high-quality solutions are desired. Second, we present an improvement procedure that iteratively shortens and smooths a feasible coverage path; robot configurations are adjusted without violating any coverage constraints. Third, we propose a modular algorithm that allows the simple components of a structure to be covered using planar, back-and-forth sweep paths. An analysis of probabilistic completeness, the first of its kind applied to coverage planning, accompanies each of these algorithms, as well as ensemble computational results.

The motivating application throughout this work has been autonomous, in-water ship hull inspection. Shafts, propellers, and control surfaces protrude from a ship hull and pose a challenging coverage problem at the stern. Deployment of a sonar-equipped underwater robot on six large vessels has led to robust operations that yield triangle mesh models of these structures; these models form the basis for planning inspections at close range. We give results from a coverage plan executed at the stern of a US Coast Guard Cutter, and results are also presented from an indoor experiment using a precision scanning laser and gantry positioning system.

Thesis Supervisor: Franz S. Hover
Title: Finmeccanica Career Development Professor of Engineering

Acknowledgments

I thank Professor Franz Hover for providing guidance and insight that have influenced this thesis in numerous ways. In addition to his idea to apply coverage planning to ship hull inspection, his suggestions have aided in the design of algorithms and experiments, and his advice has improved the visualization and written presentation of this work. I also thank Professor Daniela Rus, Professor James Orlin, and Professor Doug Hart for participating in my thesis committee. Their suggestions inspired the smoothing and partitioning algorithms developed in this thesis, as well as some of the choices made in computational and theoretical algorithm analysis. Anonymous reviewers of IEEE, IFRR, and AAAI manuscripts have also made excellent suggestions over the years that have improved this work and its presentation.

Dr. Michael Kaess, Professor Ryan Eustice, Dr. Ayoung Kim, Hordur Johannsson, Paul Ozog, Dr. Jerome Vaganay, Kim Shurn, and Professor John Leonard were terrific colleagues throughout our experimental work on ship hull inspection. The synergy that emerged during our HAUV field tests and meetings led to good experimental outcomes for all. I also thank Dr. Jose Vasquez and Dr. Scott Reed for developing great software for filtering and rendering DIDSON data. I owe a debt of gratitude to Victoria Steward for organizing and managing our field experiments on Navy and Coast Guard vessels.

Dr. Geoff Hollinger has been a wonderful collaborator, and I appreciate his patient feedback over the course of our software development and paper-writing. I am grateful for early discussions with Dr. Tirtha Bandyopadhyay and Dr. Hanna Kurniawati on the subjects of path planning and view planning. I enjoyed many lunchtime conversations with Dr. Josh Taylor on optimization, the merits of heuristics, and life in academia. It is always helpful to talk shop with Georgios Papadopoulos about path planning algorithms and their implementation. I also thank the members of the Hover research group for their kindness and support, especially Brooks Reed, Eric Gilbertson, and Josh Leighton.

And finally, I dedicate this thesis to my family: Jane, Joe, Michael, and Sherry.

None of this would be possible without your tremendous love and enthusiasm. Sherry, I am in awe of the journey we have traveled together these last five years. Sharing the joys, surprises, and challenges of these years with you has been an honor, a privilege, and a never-ending thrill ride.

This work was supported by the Office of Naval Research under grant N00014-06-10043, monitored by Dr. T.F. Swean, and grant N00014-09-1-0700, monitored by Dr. M. Steinberg and Dr. J. Stack. Additional support was provided by the MIT-Singapore Alliance for Research and Technology's Center for Environmental Sensing and Modeling, the Finmeccanica Career Development Professorship in Engineering, and the MIT Sea Grant College Program's Doherty Professorship in Ocean Utilization.

Contents

List of Figures

List of Tables

Chapter 1

Introduction

In this chapter, we introduce the real-world problem that has motivated this thesis: the inspection of a ship hull by an autonomous underwater vehicle. This challenge has inspired our development of new path planning techniques that achieve sensor coverage of complex 3D structures. Our statement of the problem in Section 1.1 is followed by a statement of the contributions of this thesis in Section 1.2. To lay a foundation for the technical arguments made in this thesis, a review of relevant prior work is given in Chapter 2.

1.1 Motivation and Problem Statement

Robots improve the efficiency, economy and speed of many tasks, but their contributions are especially valuable when they can assume a mission that is dangerous to humans. In the subsea domain there are many jobs that fit this description, but one of the most compelling is the inspection of security-sensitive underwater structures, such as ship hulls. On the subject of autonomous inspection and identification, the US Navy Unmanned Undersea Vehicle Masterplan [157] states the following:

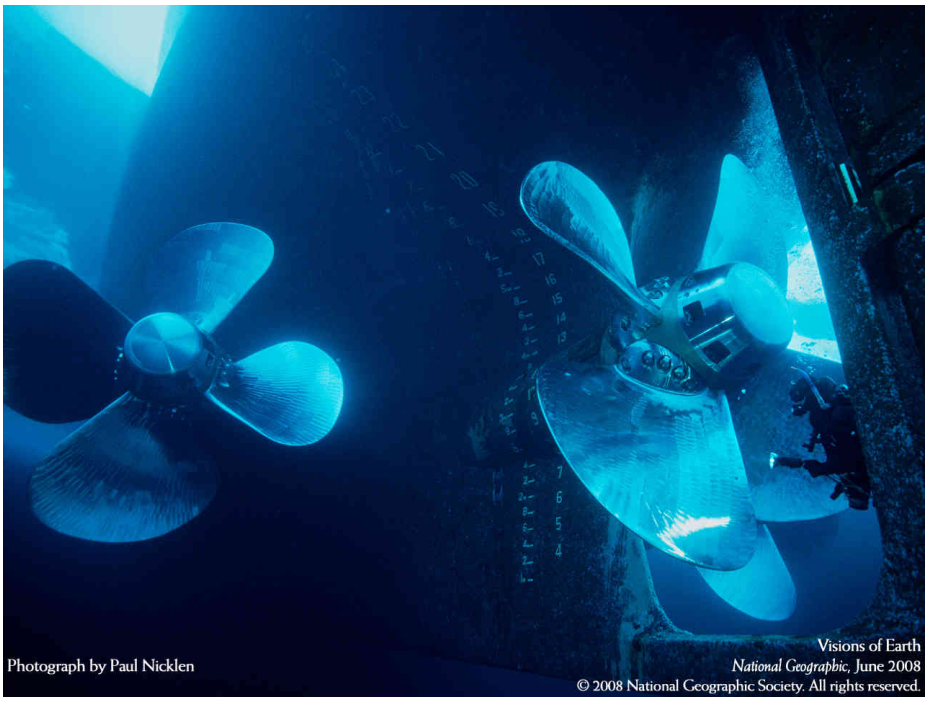
> [H]ull and pier inspection is generally the responsbility of EOD Diver teams, and it is both time and manpower intensive. The demand for security swims around piers and hulls has resulted in over a six-fold increase in these diver operations since the events of September 11, 2001.

> Additional assets beyond the available EOD Diver teams are needed to effectively meet these additional requirements for inspection.
>
> The typical targets in a hull or pier search would be unexploded ordinance, such as limpet mines or special attack charges. ... Searching for ordinance that is typically time-fused is particularly hazardous to divers. Use of an unmanned vehicle can reduce the risk to EOD technicians and divers by providing the precise location of suspicious objects, while relieving the divers of the tedious search process in cluttered environments.

When a hull inspection is performed by humans, as illustrated in Figure 1-1, not only are divers at risk of serious injury, but there is a possibility that hidden ordinance may go undetected if any portion of the hull is missed or overlooked in the inspection. The aim of an autonomous ship hull inspection is to perform the task safely and to obtain 100% coverage of every exposed surface.

Efforts to automate the inspection of the in-water portion of a ship hull have included a number of systems which physically attach to a hull using suction [69] or magnets [28], [119], and drive around its surface using wheels or treads. These systems can inspect the relatively flat, open areas of a hull at close range, and are designed primarily to assess the hull's structural integrity. The *Cetus II* autonomous underwater vehicle (AUV), which is proposed for mine detection applications, maneuvers free from the hull and uses an acoustic long-baseline navigation system for accurate state estimation [155]. Implementation of this system requires the placement of acoustic transponders on the seafloor to measure the AUV's position. Although the *Cetus II* posseses maneuverability and hovering capability superior to traditional, torpedo-shaped AUVs, its design is primarily intended for travel in a single, forward direction.

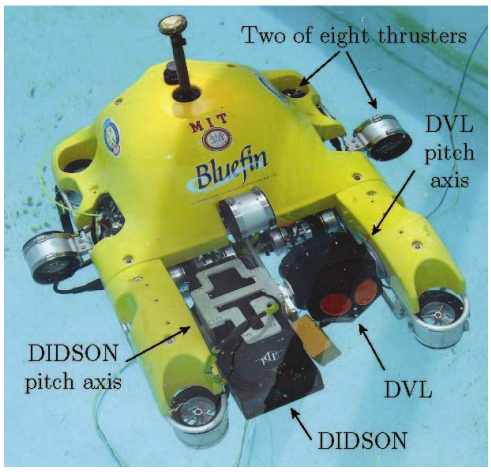
The Bluefin-MIT Hovering Autonomous Underwater Vehicle (HAUV) has been developed as a compromise between a hull-crawling vehicle and a free-floating AUV. This vehicle, pictured in Figure 1-2, possesses fully-actuated, omnidirectional hovering capability. A dual-frequency identification sonar (DIDSON) is used as the primary



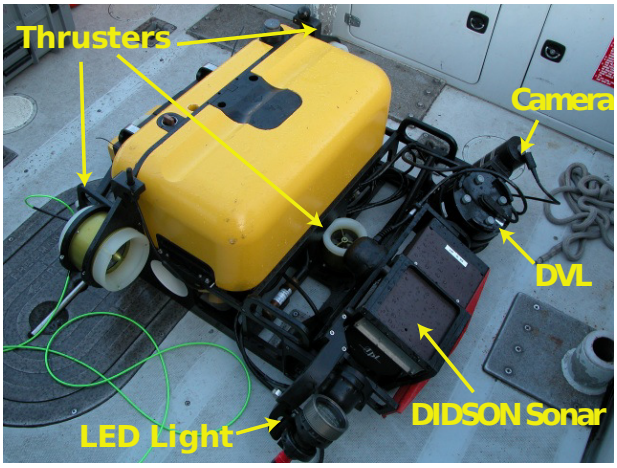
(a) A diver inspecting one of the stainless steel propellers of the *CCGS Louis S. St-Laurent*, a Canadian Coast Guard heavy Artic icebreaker ship.

(b) A US Navy diver searches for a training explosive hidden at the stern of a Coast Guard vessel.

Figure 1-1: Photographs of hull inspections in progress. Image credits: a) P. Nicklen, National Geographic Society, `http://ngm.nationalgeographic.com/wallpaper/img/2008/06/june08-02-1280.jpg` b) J. Pastoric, US Department of Defense, `http://www.defense.gov/photos/newsphoto.aspx?newsphotoid=14431`



(a) The HAUV, version 1B, used for field experiments in 2007-2010.



(b) The HAUV, version HULS3, used for field experiments in 2010-present.

Figure 1-2: Two recent prototypes of the Hovering Autonomous Underwater Vehicle (HAUV). Image credits: a) M.R. Walter *et al.*, 2008 [163] b) F.S. Hover *et al.*, 2012 [76]

sensor for inspecting the hull [16]. A monocular camera is also used when water clarity allows. To inspect flat, open areas that are easily covered by a hull-crawling vehicle, the HAUV navigates relative to the hull using its Doppler velocity log (DVL) [77]. To inspect *complex areas* that contain protruding 3D structures, the DVL is pointed at the seafloor instead.

Recent research efforts have focused on developing high-accuracy localization and mapping capability over the flat areas of the hull, hereafter referred to as the *non-complex areas*. This is a compelling problem because the non-complex areas are expansive, comprising the vast majority of surface area to be covered in an inspection. Over the course of inspecting a large vessel, the DVL and the inertial measurement unit (IMU) are subject to drift, and state-of-the-art simultaneous localization and mapping (SLAM) algorithms have been developed to correct drift, produce high-quality maps of the hull, and provide the precise location of features observed on the hull. This has been achieved by tracking observations of point features in DIDSON images [163], and also by registering pairs of camera frames [92] and DIDSON frames [84] that contain overlap. Camera and DIDSON image registration have also been integrated into a unified state estimation process [76].

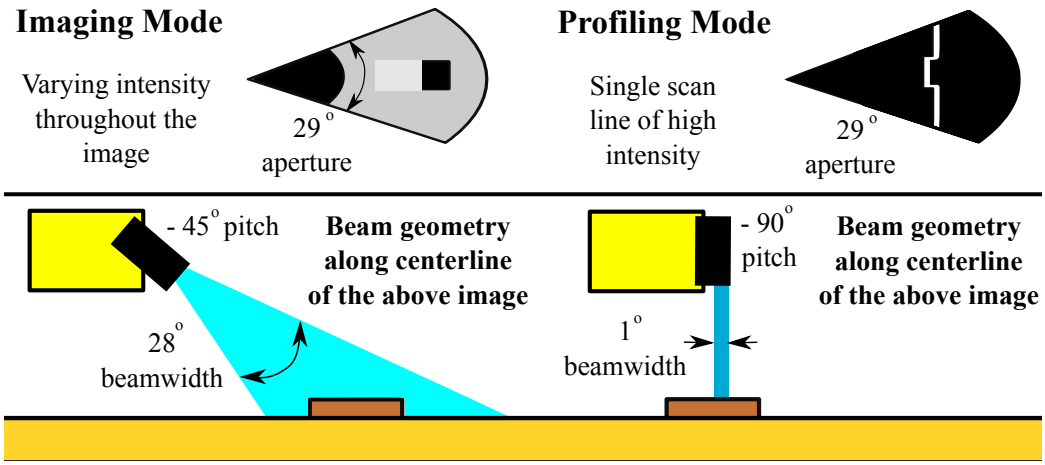
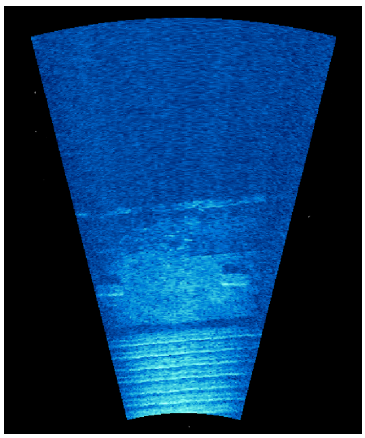


Figure 1-3: The geometry of the DIDSON field of view, shown in imaging mode at left and profiling mode at right.

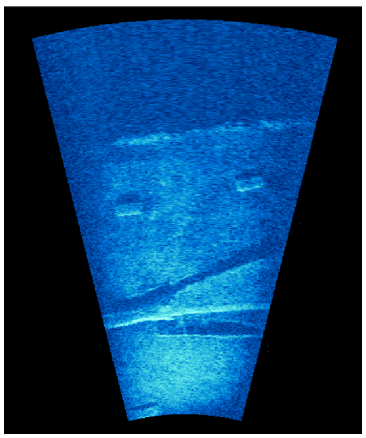
The geometry of the DIDSON field of view is depicted in Figure 1-3. The sensor has a 29-degree-wide field of view, which is spanned by 96 individual beams. The vertical aperture of the sensor is described by its "beamwidth", which can be focused to different widths using a lens mounted on the sensor. The ranges at which each beam intersects its surrounding environment are recorded in every image that is formed, along with an intensity value corresponding to every range return.

Obtaining 100% sensor coverage of a ship's non-complex area is fairly straightforward. The DIDSON is operated in "imaging" mode, in which the sonar produces 2D images of the hull that cover several square meters each. This mode uses a 28-degree vertical beamwidth; the result is a flattened, 2D depiction of structures in the sonar field of view. A planar, back-and-forth sweep path is designed for the HAUV in the hull relative coordinate frame, with conservative overlap between images to account for any navigation drift that accumulates. This method was experimentally validated in the early stages of vehicle development, and is found to reliably achieve full coverage of the non-complex areas of a ship hull [158].

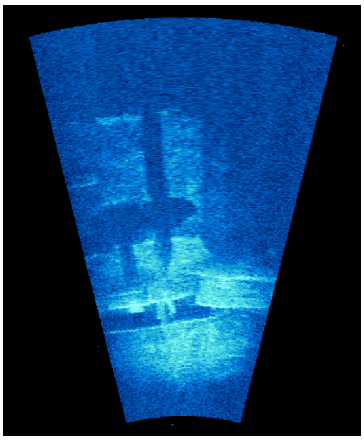
On the other hand, obtaining full coverage of the complex areas, which are primarily located at the stern of a ship, is a hard problem. The shafts, propellers, rudders, and other protruding structures lie in close proximity to one another and to the hull. A clear view of these structures is often obstructed from all but a few vantage points,



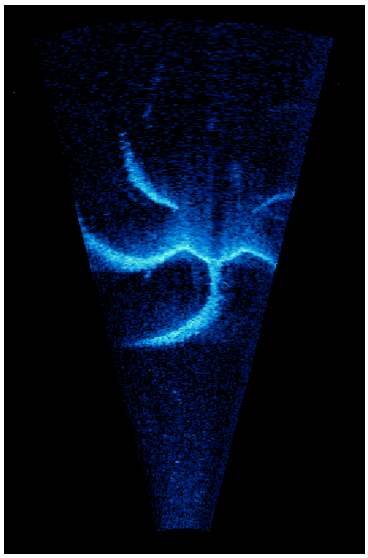
(a) Sonar image of hull featuring keel cooling pipes and zinc anodes.



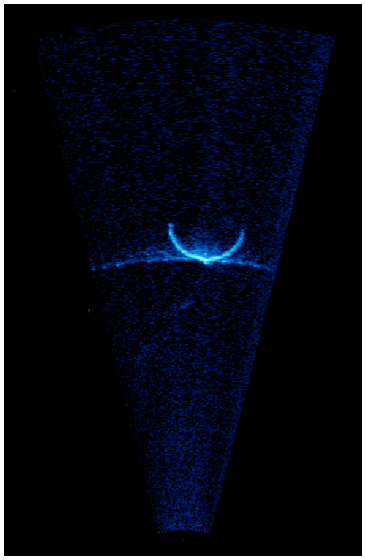
(b) Sonar image of hull featuring zinc anodes and shaft.



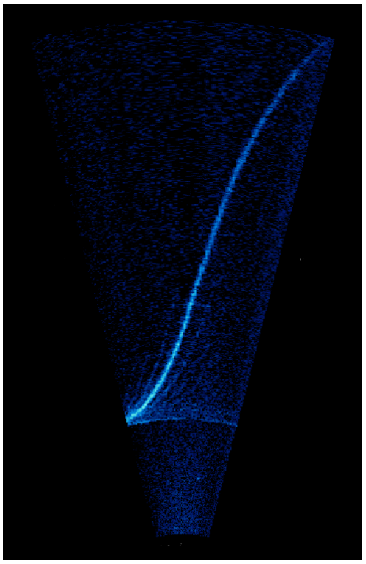
(c) Sonar image of hull featuring propeller and rudder.



(d) Sonar scan of ship propeller, which is approximately seven meters in diameter.



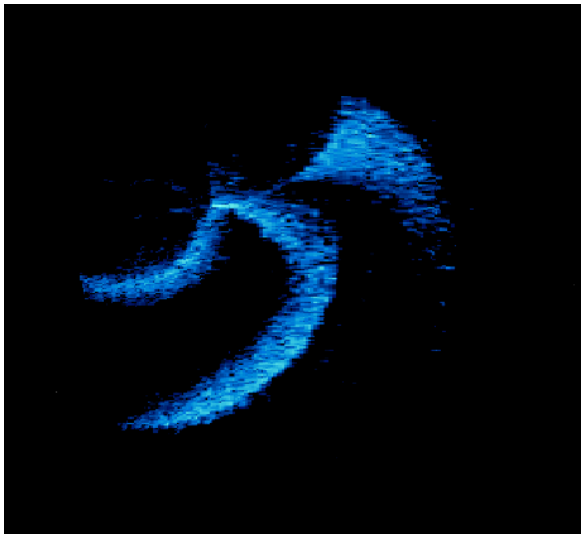
(e) Sonar scan of shaft, which is approximately one meter in diameter.



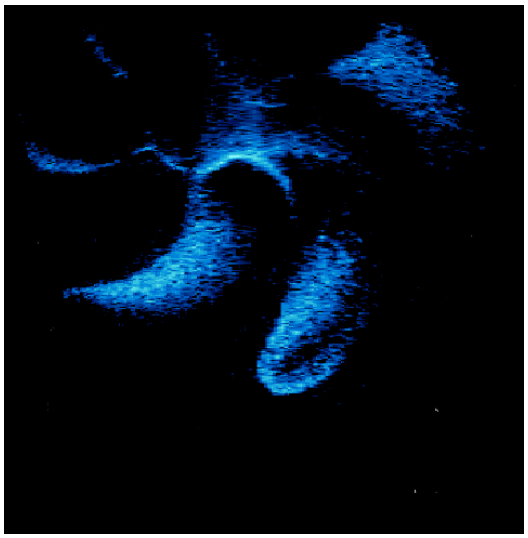
(f) Sonar scan featuring a cross-section of the hull.

Figure 1-4: Examples of DIDSON sonar data from imaging mode, at top, and profiling mode, at bottom, collected during HAUV field tests. The images at top, from the inspection of a twenty-meter-long US Coast Guard inland buoy tender, show a field of view with 2.5-7 meter vehicle-relative range. The profiling scans at bottom, from the inspection of *SS Curtiss*, a 183-meter-long aviation logistics support ship, show a field of view with 2-11 meter vehicle-relative range.

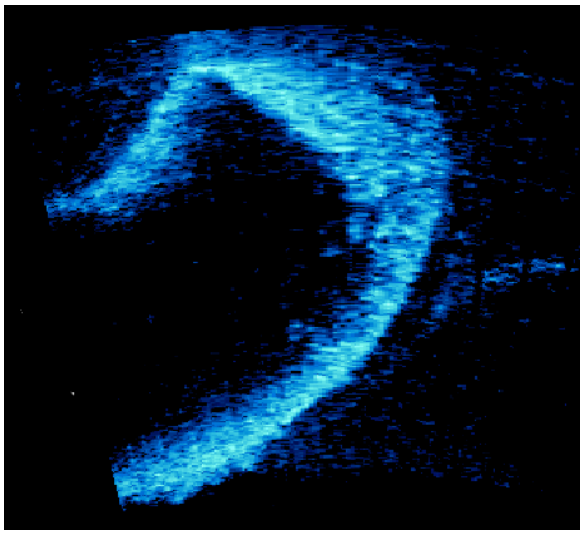
due to low clearance and complex 3D geometry. To accurately observe these structures, the DIDSON is operated in "profiling" mode, which uses a one-degree vertical beamwidth to generate an unambiguous range scan rather than a flattened 2D image.



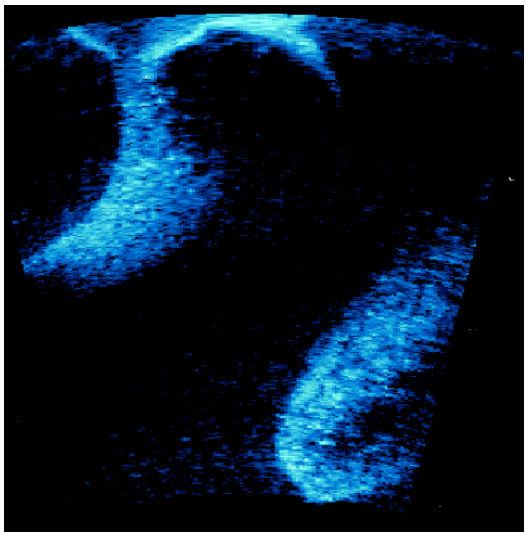
(a) View 1 of propeller, low resolution.



(b) View 2 of propeller, low resolution.



(c) View 1 of propeller, high resolution.



(d) View 2 of propeller, high resolution.

Figure 1-5: Examples of DIDSON sonar scans collected in profiling mode with different fields-of-view. Images at top are from nine meter field-of-view scans, and images at bottom are from 4.5 meter field-of-view scans. All scans depict the seven-meter-diameter propeller of the *SS Curtiss*.

Examples of data collected by the HAUV in both the imaging and profiling viewing modes are given in Figure 1-4.

The DIDSON allows improved scan resolution to be obtained by sensing at reduced range. Scans contain the same number of pixels whether they span a ten-meter range or a two-meter range. For the purposes of detecting hidden ordinance during an

inspection, it is ideal to inspect the hull using a high-resolution, short-range viewing mode, despite the reduced field of view. An example of the difference in image quality that results is given in Figure 1-5.

The goal of this thesis is to plan and execute an efficient inspection route that obtains 100% sensor coverage of a ship's complex areas. Due to the HAUV's fully-actuated control scheme and dynamics that are dominated by hydrodynamic drag, the plannning task is modeled as a geometric positioning problem. We assume that a prior model of the ship is available for planning the geometric inspection route. In the absence of a CAD model, the HAUV can sweep the perimeter of the stern at a safe distance and construct a low-resolution model sufficient to identify all ship structures. The model can be used to subsequently plan and execute a high-resolution inspection that searches for ordinance on the hull, with a significantly reduced field of view. Path planning to achieve sensor coverage of a large, complex structure, using a sensor with a small field of view, is the central focus of this thesis. Motivated by the autonomous ship hull inspection problem, we contribute new algorithms for the solution of *coverage path planning* over complex 3D structures containing low-clearance areas and visually occluded areas.

1.2 Overview and Contributions of the Thesis

Coverage path planning, the design of a collision-free path that also sweeps a robot's end effector over a required surface area, has been applied to a variety of problems in robotics and automation. Many of these problems involve path planning in support of sensing tasks, but other applications include material removal and material deposition. In Chapter 2 we review the algorithms and applications of coverage path planning, as well as other related topics in robotics. This includes the subjects of path planning, view planning, and multi-goal planning.

In Chapter 3 we introduce a new algorithm for planning a 100%-coverage collision-free inspection route. This algorithm relies on the random sampling of robot configurations to iteratively construct a full-coverage set of sensor views. The views

are then joined into a cyclical collision-free inspection route using a second phase of sampling. An accompanying analysis quantifies the likelihood that the algorithm's random sampling procedures will return a feasible solution after a specific number of samples. This *probabilistic completeness* analysis is applied to both our algorithm and a prior, competing algorithm. The relative computational performance of the two algorithms is studied, and we demonstrate that our proposed algorithm is better suited to computation over structures of complex 3D geometry. This is achieved using a state-of-the-art software implementation that applies modern data structures and combinatorial optimization tools to 3D coverage problems of unprecedented size, planning over ship models comprised of hundreds of thousands of geometric primitives. Our work on this topic is also documented in [52].

In Chapter 4 we present an iterative improvement procedure for smoothing and shortening inspection routes that are constructed using random sampling. The algorithm can be applied to an exisiting, feasible solution for long as time allows, whether several minutes or several hours are available for improving the planned inspection. This procedure also relies on random sampling, which is used to find improved sensor views that are both collision-free and satisfy the coverage constraints unique to an existing view on the inspection route. Computational results and an analysis of probabilistic completeness are also presented for this algorithm, which makes significant improvements to coverage routes designed for the HAUV complex-area inspection task. Although improvement procedures of this type have been used in standard point-to-point path planning, this is the first algorithm we are aware of that iteratively smooths paths under coverage constraints. Our work on this topic is also documented in [53].

In Chapter 5 we propose an algorithm for planning inspection routes of high regularity over complex structures. Regularity is desirable when the data from an inspection must be analyzed or processed by a human operator. We partition a structure into several pieces and design a back-and-forth planar sweep path for each segment. Any portion of the structure that is not covered by a regularized sweep path is covered by randomized views sampled at the end of the procedure to fill in

the remaining gaps in coverage. All sweep paths and randomized configurations are then joined into a single, contiguous inspection route. We analyze the probabilistic completeness of this algorithm, and we give computational results that show a tradeoff between the regularity of a route and the length of a route which can be tuned by changing the order of the segmentation. This is the first algorithm we are aware of that joins randomized and regularized paths into a single contiguous inspection route, offering flexibility when a structure cannot be covered entirely by back-and-forth sweep paths. Our work on this topic is also documented in [54].

The analysis of probabilistic completeness performed in these three chapters is the first of its kind applied to path planning under coverage constraints. By unifying techniques from prior analyses of path planning algorithms and sensor placement algorithms, we have established sharply decreasing exponential bounds that govern the convergence of all procedures in this thesis that rely on random sampling.

In Chapter 6 we show experimental outcomes from the HAUV. We first describe the use of low-resolution inspection surveys to produce *a priori* mesh models for path planning and algorithm development. Our work on this topic is also documented in [73]. We then give the results of a field experiment in which a planned, smoothed, coverage path created using one of these *a priori* models is implemented to achieve high-resolution coverage at the stern of a US Coast Guard Cutter. The resolution of the model is improved as a result. Finally, we give results from an inspection planned and executed using a laser rangefinder in air, in which the implementation of a planned path achieved full coverage and enabled the identification of mine-like targets planted on the structure of interest.

We conclude in Chapter 7 by reviewing the contributions of this thesis and identifying promising areas for future work.

Chapter 2

Background

This chapter contains a survey of prior work in coverage path planning and other related topics. We begin with an introduction to path planning in general; this includes a review of classical algorithms as well as the sampling-based paradigm that is central to the work in this thesis. We then discuss the subjects of view planning, view ordering, and multi-goal planning, which comprise foundational building blocks of the algorithms developed in this thesis. We close with a survey of coverage path planning, reviewing algorithms that synthesize many of the concepts and techniques discussed in the sections prior.

2.1 Path Planning Overview

A critical component of autonomy is an agent's ability to use a model of its environment for planning and executing tasks that require physical motion. Sometimes the model is known completely in advance, and sometimes the model is produced or refined in real-time using the agent's sensing and inference capabilities. In either case, path planning drives the agent's decision-making about how to move through its environment. Paths can be planned to minimize a variety of cost functions, but most generally the goal of path planning is to move from one *configuration* to another in as short a distance as possible. Unfortunately, this statement is deceptively simple. A robot's configuration, comprised of the state variables needed to completely describe

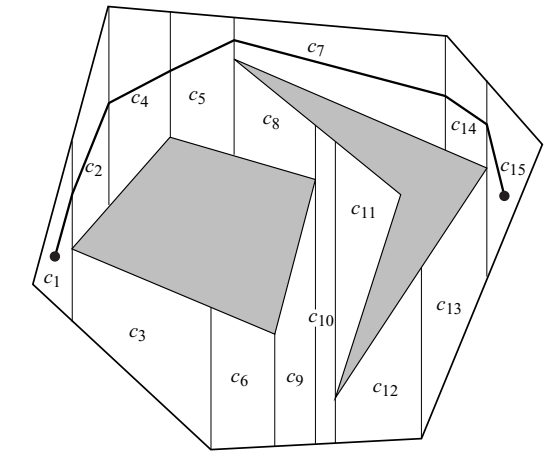
a robot's position in space at an instant in time, may contain variables from different topological spaces described by different metrics, such as the Cartesian coordinates and Euler angles that together identify the six degree-of-freedom configuration of a rigid body. Different functions may often be required to penalize movement in degrees of freedom that differ topologically [97].

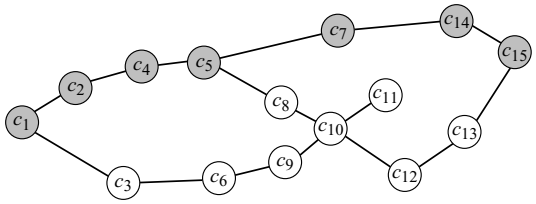
Consequently, a robot's *configuration space* (C-Space) may often be of a different topology and dimensionality than the 2D or 3D Euclidean *workspace* in which the robot is observed to operate. The instructions comprising a planned path must be expressed in C-Space for a robot to execute them unambiguously. A motivating example is the programming of a multi-link manipulator, whose end effector trajectory can be described by coordinates in Euclidean space, but requires instructions in the space of joint angles for a unique path to be specified for the entire physical robot. Describing a robot's position is made more complex still when geometric obstacles are considered. Obstacles, such as walls, are most naturally described in a 2D or 3D Euclidean workspace. For a path to be planned and evaluated in C-Space, however, the obstacle boundaries must be mapped into this space. Even when a workspace and robot C-Space are of the same dimension and topology, obstacle boundaries must be adjusted for the "girth" of the robot to evaluate whether a single point in C-Space is collision-free [114].

2.1.1 Classical Path Planning

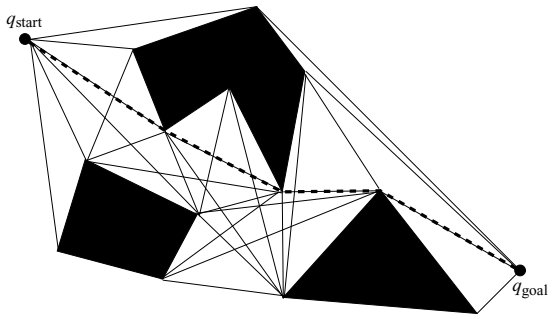
Classical approaches to path planning have focused on explicity mapping a robot's workspace obstacles into C-Space for the subsequent computation of optimal or near-optimal paths. Algorithms of this type construct a *roadmap*, a graph in C-Space whose nodes and edges represent collision-free configurations and straight-line paths that link them together. Construction of the roadmap is the major computational burden, after which optimal paths can be computed easily over the roadmap using graph search algorithms.

Cell decomposition methods such as the trapezoidal decomposition [101] divide the collision-free subset of C-Space ($\mathcal{C}_{free}$) into polygonal cells, and roadmap topology

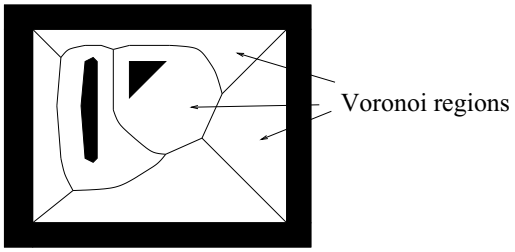




(a) A trapezoidal cell decomposition and its adjacency graph, each showing a planned path.



(b) A visibilty graph, and a path planned over the graph.



(c) A GVD over 2D polygonal obstacles.

Figure 2-1: An illustration of classical path planning tools. Image credit: H. Choset, The Robotics Institute of Carnegie Mellon University, `http://www.cs.cmu.edu/~biorobotics/book/figures`

is determined by the adjacency of cells. Roadmap nodes can be planted in the center of the cells to achieve suitable clearance from obstacles. Paths of maximum clearance can be achieved using a generalized Voronoi diagram (GVD), a graph whose edges explicitly represent the locations of maximum distance to obstacles [109]. A roadmap of shortest paths can be constructed using a visibility graph, which adds all obstacle vertices to the roadmap and creates roadmap edges between all vertices with an unobstructed line-of-sight [115]. These methods are illustrated in Figure 2-1.

Despite the great success of these algorithms in 2D Euclidean C-Space for robots translating among obstacles, optimal planning becomes more challenging as the dimensionality of C-Space increases and obstacle geometry becomes more complex. Two classical algorithms capable of planning paths for arbitrary C-Spaces, populated with arbitrary obstacles, are the cylindrical cell decomposition algorithm [43] and Canny's roadmap algorithm [27], which have worst-case exponential runtimes of $O((nd)^{3^k})$ and

$O(n^k(\log n)d^{O(k^4)})$ respectively. The dimension of C-Space, the number of polynomials required to describe the workspace obstacles, and the maximum degree among the polynomials are represented by k, n, and d, respectively.

All of the above algorithms create an efficient roadmap that uses as few nodes as possible to represent the connectivity of $\mathcal{C}_{free}$. It is possible to instead shift some of the computational burden from the roadmap construction step to the graph search step of the planning problem. In a grid-based approach, a few distinct levels of grid resolution are specified and $\mathcal{C}_{free}$ is populated with a graph of highly uniform spatial resolution [59]. A wide variety of methods have been developed for planning as a discrete state space search, motivated by robotics and many broader applications in artificial intelligence [133].

2.1.2 Sampling-Based Planning

An alternative approach for problems of high dimension or complex geometry is sampling-based planning. The goal of sampling-based planning is to find a feasible path by repeatedly probing C-Space with a sampling and collision-checking scheme [103]. This offers an alternative to explicitly constructing obstacles in C-Space and optimizing over their geometry. Instead, robot configurations can be individually projected into the workspace and checked for interference with workspace obstacles. This paradigm benefits from fast and efficient algorithms for collision detection among large polyhedra, which can be used to represent nearly any robot or workspace model in the form of a triangle mesh. Using one such method, the OBBTree, for storing polyhedral models, $O(n \log^2 n)$ time is needed to construct the required data structures, and a single collision-checking query is found in practice to be a constant-time operation [68].

Random Sampling

Many sampling-based algorithms adopt a strategy of random sampling to identify candidate configurations for collision-checking. Random sampling is observed to per-

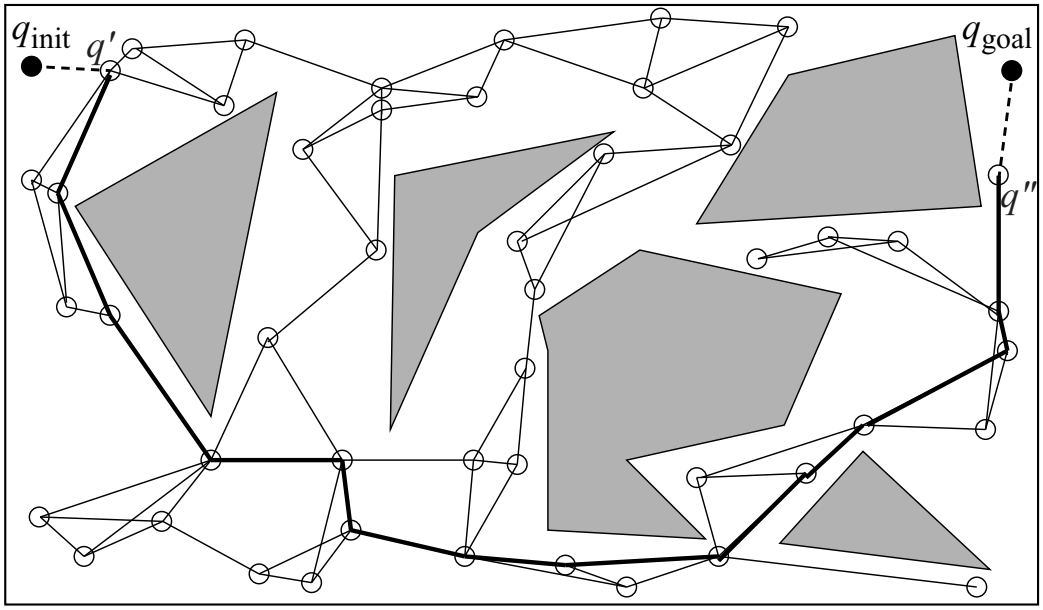


Figure 2-2: A PRM is pictured, and its use in solving a path planning query is illustrated. Image credit: H. Choset, The Robotics Institute of Carnegie Mellon University, `http://www.cs.cmu.edu/~biorobotics/book/figures`

form well in practice [65] and it enables strong theoretical guarantees with respect to algorithm performance. A key property is *probabilistic completeness*, a guarantee that if a feasible path exists, the algorithm will find one with probability that tends to one as the number of samples tends to infinity [100]. In addition to providing this guarantee, it is often possible to demonstrate appealing convergence rates for randomized sampling-based planning algorithms. This has been achieved for several algorithms by expressing the probability of failure to return a feasible solution as a sharply decreasing function of the number of samples drawn [37].

One of the earliest and most successful sampling-based algorithms is the probabilistic roadmap (PRM) [89], illustrated in Figure 2-2. Robot configurations are sampled uniformly at random, and then checked for collision with obstacles. If a sampled configuration is free of collision, it is connected by straight-line paths to the nearest nodes in the roadmap. The straight-line paths are also checked for collision as they are generated. Roadmap construction terminates when $\mathcal{C}_{free}$ is deemed by the user to be suitably connected for answering path planning queries. Because of the computational effort required for roadmap construction, the PRM is most often

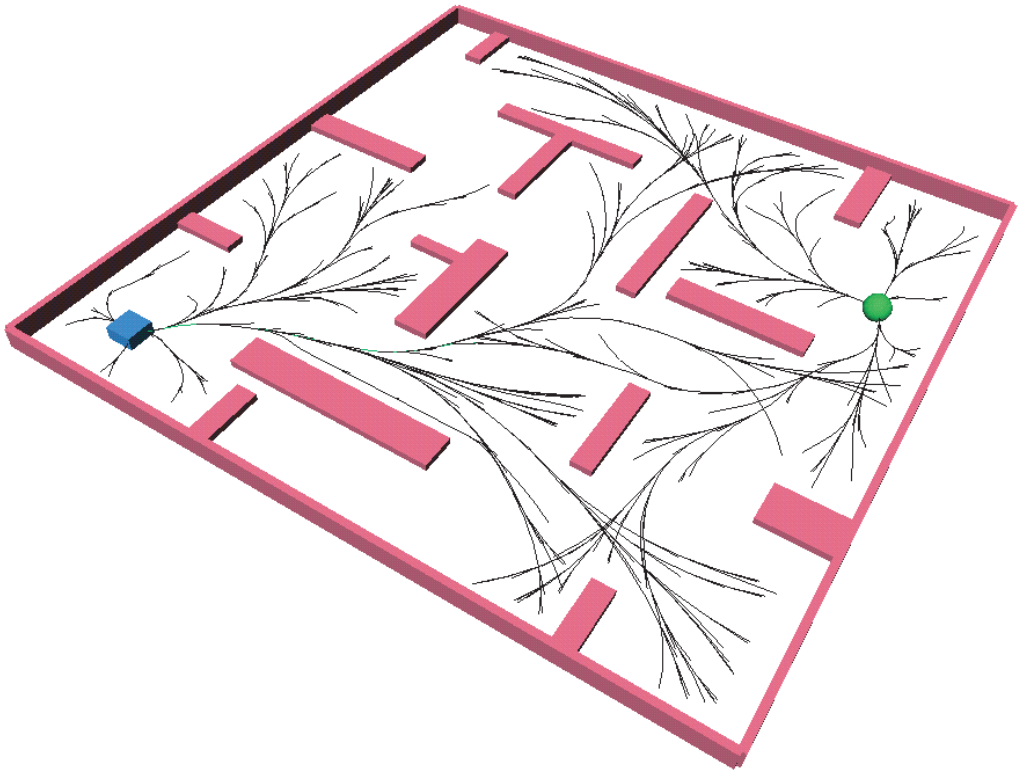


Figure 2-3: An example of a bi-directional RRT used to find a feasible path from a start configuration (blue) to a goal configuration (green). Image credit: S.M. LaValle and J.J. Kuffner, Jr., 2001 [105].

utilized when multiple path planning queries will be made for a particular robot and C-Space.

For single-query applications in which a roadmap is of no long-term utility, the rapidly-exploring random tree (RRT) [102] offers a fast and efficient alternative. This algorithm constructs a tree graph that is rooted at the start configuration, and "grows" in the direction of randomly sampled configurations until the goal configuration can also be added to the tree. A higher-performance version of the algorithm grows two trees, rooted at the start and goal respectively, and grows them until they can be connected [98]. An example is pictured in Figure 2-3.

Recent improvements to both the PRM and RRT have yielded *asymptotically optimal* variants of these algorithms, PRM* and RRT* [86]. A probabilistically complete path planning algorithm is asymptotically optimal if the shortest path found by the algorithm converges to the optimal path with probability that tends to one

as the number of samples tends to infinity. Unlike probabilistic completeness, this is a property that is achieved only in the limit, and not in a finite number of random samples. From a practical perspective this is nonetheless a highly valuable property, since near-optimal solutions can be obtained in a finite number of samples.

Deterministic Sampling

Despite the ubiquity of randomized algorithms, sampling-based path planning can also be performed using deterministic sampling schemes. In addition to quasi-Monte Carlo sequences, sampling and collision-checking configurations along a simple rectangular grid has also been found to achieve good results for certain path planning problems [104]. All samples in a grid must be checked before it is known whether the grid is of sufficient resolution to solve a planning query, and if it is not, then a grid of finer resolution can be generated. A deterministic path planning algorithm is *resolution complete* if, when a feasible path exists, the algorithm is guaranteed to generate a grid or other deterministic sampling scheme of sufficient resolution to find it [101].

2.1.3 Path Planning Under Constraints

Dynamics

In path planning problems of practical interest, additional constraints may be posed alongside the basic requirement that the path from a start configuration to a goal configuration must be collision-free. A common set of additional constraints are kinematic or dynamic differential equations that govern the behavior of the robot. The RRT and related algorithms of tree structure have been highly successful in applications requiring fast kinodynamic planning [105], [61], [87], where "growth" from one node to the next is governed by the application of a user-determined force or velocity input. If, instead of fast computation, minimization of a dynamic robot's time or energy consumption are the specific focus of a planning problem, a number of trajectory optimization methods are suitable [19]. In a notable application of trajectory

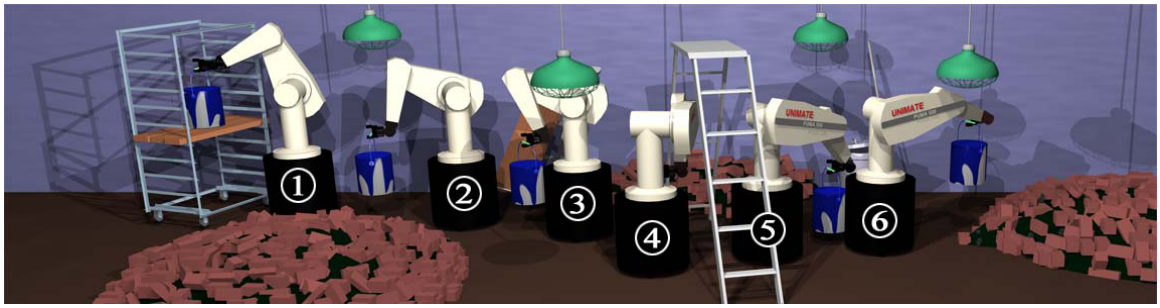


Figure 2-4: An example of a planned path with task constraints requiring the carrying and placement of an object. Image credit: M. Stilman, 2007 [148].

optimization to minimize fuel consumption, both workspace obstacles and system dynamics are expressed as constraints in a mathematical programming formulation [141].

Uncertainty

There is also an expansive body of work on achieving robust path planning in the presence of uncertainty. A fundamental way to accomplish this is through feedback motion planning, which couples a planned path with a feedback control law [132], [153]. Designing control policies to accompany a planned path improves the likelihood that a robot will reach its goal, even in the presence of exogenous disturbances. Robustness can also be achieved by planning under assumptions of sensing uncertainty [128], and in turn using feedback motion planning to manage this source of uncertainty [160]. It is also possible to plan under the assumption of an uncertain or incomplete model of the workspace [120], [26].

Task-Specific Constraints

Another practical set of constraints is the specification of a task that the robot must perform while traveling from start to goal. Carrying objects, opening doors, and maintaining contact with surfaces to operate tools or push heavy payloads are examples of task constraints to which sampling-based planning algorithms have been applied successfully [148], [17]. An example of a path planned for carrying an object is illustrated in Figure 2-4. Other families of constraints include planning simultane-

ous trajectories for multiple robots [149], [138], planning under known disturbances, such as ocean current fields [111], planning with multiple goals, which will be discussed separately in Section 2.4, and planning under coverage constraints, which is the central focus of this thesis.

Coverage constraints require a robot to sweep its end effector over some portion of the workspace surface area. A variety of tasks can be described using coverage constraints, including sensing, material deposition, and material removal. The end effector may be a paintbrush, a shovel, a cleaning device, or simply the geometric footprint of a sensor.

2.2 View Planning

A rich subject related to path planning under coverage constraints is view planning. The aim of view planning, in which the coverage task is specifically one of sensing, is to select a set of sensor views that provides full coverage of a structure in the workspace [143]. In many applications the goal is to construct a model of an unknown structure by efficienty exploring the space of sensor views. When an *a priori* model of the structure is available, the goal of view planning is to design a full-coverage set of sensor views using the model, sometimes directing the placement of a group of sensors rather than the movement of a single sensor or robot.

2.2.1 Exploratory View Planning

When no prior model of the structure is available, *next-best-view* strategies have been highly successful in covering and modeling small structures in indoor environments. Next-best-view (NBV) algorithms follow the *active perception* philosophy of using sensed information as feedback to drive real-time, exploratory sensing and decision-making [14]. After a view is acquired by a sensor, this view and all previous views are used to decide where the next view should be collected.

Most work in NBV planning can be divided into two categories, surface-based methods and volumetric methods. Surface-based methods choose where to look next

(a) A camera and laser-equipped 3D modeling sensor mounted on a robotic manipulator.

(b) A manipulator positioning objects for viewing with a depth camera.

Figure 2-5: Two examples of modern experimental apparatuses used for 3D modeling of objects via NBV planning. Image credits: a) S. Kriegel *et al.*, 2011 [96] b) S. Krainin *et al.*, 2011 [95]

by reasoning about the geometry of a structure's surface. This has been achieved by using the occluding edges of obtained views to infer which views will be unoccluded [118], [126], and by fitting parametric curves to the observed portions of the structure to infer the curvature of unobserved areas [167], [31]. Volumetric methods choose where to look next by reasoning about the workspace volume occupied by the structure. This has been achieved by modeling the workspace using a grid of voxels and selecting views based on the occupancy of the voxels [44], [117], [15], and by including occluded volumes within a polyhedral solid model and planning views of the edges of these volumes [130].

Most of the above methods plan in a low degree-of-freedom configuration space, often referred to as the *viewpoint space* in the context of NBV algorithms. This viewpoint space is often the surface of a cylinder or sphere that encloses the object being inspected. Recent developments in NBV algorithms have focused on gathering views using higher degree-of-freedom robots [96], [154], including cases in which the robot grasps the structure and moves it to obtain higher-quality views [95]. View

planning applications that use such robots are pictured in Figure 2-5.

2.2.2 Model-Based View Planning

Sensor-Specific Methods

When a prior model of the structure is available, a breadth of methods have been developed for selecting a high-quality set of views. Many of these methods model the properties of specific sensors to optimize optical parameters such as focus, magnification, and illumination [150]. Early methods of particular note include a system that concurrently plans positions for both a camera and separate accompanying light source [136], and a nonlinear optimization method in which three optical parameters are varied along with a robot's five-DOF spatial configuration. [151]. Photogrammetric network design to achieve highly accurate 3D measurements from a group of cameras has been achieved using genetic algorithms [123].

Sensor-specific planning has also focused on optimal data acquisition for laser range sensors [129]. Studies on range sensing for constructing an improved-accuracy model from an unreliable prior model have developed the concept of planning using a *measurability matrix*, a data structure that catalogs which structure surface points can be measured from selected admissible viewpoints [152], [142].

The Art Gallery Problem and The Set Cover Problem

There is also a large body of work that has emphasized the geometric and combinatorial challenges of view planning rather than those of a specific sensor. The view planning problem has often been modeled as a variant of the *art gallery problem*, in which a minimum-cardinality set of "guards" must be selected and placed to view 100% of the internal area of a polygon [124]. It is typically assumed that a guard has an infinite-range field of view that is obstructed only by blocking its line-of-sight. This problem is NP-hard, and several polynomial-time algorithms have been proposed for finding feasible, sub-optimal solutions [145].

Gonzalez-Baños and Latombe propose a practical heuristic by solving the art

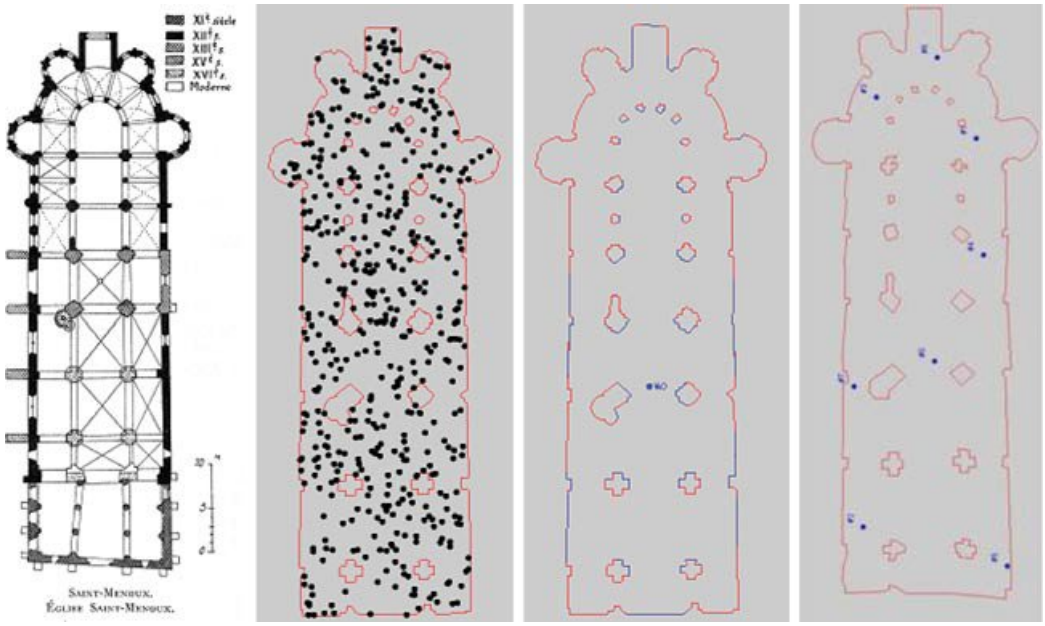


Figure 2-6: An example of the dual sampling algorithm used for view planning. Illustrated are the samples drawn and views selected for coverage of the walls of a cathedral. Image credit: P.S. Blaer and P.K. Allen, 2009 [21].

gallery problem in two phases, a sampling phase and a set cover phase [66], [67]. They propose a version of the problem in which only the edges of the polygon must be observed, and a guard's field of view is limited by range and incidence constraints. View configurations are sampled at random until the entire edge boundary is covered, and the *set cover problem* is then solved approximately to select a final set of guards.

Given a group of elements and a list of sets that contain various combinations of the elements, the goal of the set cover problem is to cover all elements using the smallest number of sets, or, if the sets are weighted, the minimum-weight combination of sets. Although the set cover problem is NP-hard, a variety of good approximation algorithms have been developed [162]. Of particular note are a polynomial-time greedy algorithm [85], [113], [40] and a linear programming rounding algorithm [72], both of which are very simple to implement in practice.

In their work on art gallery problems, Gonzalez-Baños and Latombe also propose *dual sampling*, an alternative random sampling strategy that has proven to be one of the most successful art gallery-inspired heuristics for view planning [66], [67]. Sensor configurations are sampled from an inverse-computed region of views that maps to a

specific point on the polygon boundary. This sampling region is moved to different points on the polygon boundary until a full-coverage set of views is achieved. This algorthm has been adapted for use with field robotic systems to plan views for coverage of both the interior and exterior walls of buildings [21], an example of which is pictured in Figure 2-6. Dual sampling has also been adapted to plan the placement of cameras for the coverage of 2D interior floorspace [75].

Art gallery-inspired camera placement for floorspace coverage has also been accomplished using binary integer programming [55], genetic algorithms [169], and iterative subdivision of the floorspace into convex polygonal segments [90]. The random sampling approach to sensor placement has been extended to plan a sensor field that satisfies both floor coverage and connectivity requirements [80]. Additionally, deterministic 2D art gallery algorithms have been adapted to achieve a worst-case-exponential sensor placement algorithm for optimal coverage of 3D polyedral structures [22].

Other Geometric and Combinatorial Methods

Sensor placement has also been modeled as a coverage problem over a collection of discrete point targets, which may be relevant in view planning applications with finely discretized structure models. A polynomial-time placement algorithm with a proven approximation factor has been used to place range sensors to cover a discrete set of points among occluding obstacles [5]. This algorithm has been extended to both place and orient rotating directional sensors [63].

A final notable contribution to the theory of model-based view planning is the aspect graph. Aspect graphs are data structures that store information on which continuous viewpoint sets are topologically equivalent, and how they are connected to other viewpoint sets [23]. Views are topologically equivalent if they observe the same continuous unoccluded region of a structure. Although they can be constructed in runtime polynomial in the number of geometric primitives of a model [127], the graphs are very large for structures of practical interest and they have not been adopted for use in practice [143]. Despite this, they have been used successfully as a theoretical analysis tool to illustrate the hardness of 3D view planning problems [81].

2.3 Planning and Ordering of Views: The Traveling Salesman Problem

There is a family of model-based view planning methods that not only selects sensor views, but also plans the order in which the views are gathered. These view planning algorithms differ from the coverage path planning methods discussed in Section 2.5; the step of identifying feasible interconnecting paths among the view configurations is treated here as a trivial problem. In some cases, this is because the algorithms constrain the selection of views to be conservatively distant from the structure being inspected.

A common view planning constraint is that all views must lie on the surface of a sphere that contains the structure being inspected. For one system of this type, in which an intensive pre-processing phase computes the visibility and view quality of all structure features from every discrete viewpoint on an enclosing, tesselated sphere, the structure is very small relative to the sphere that contains it [156]. Collision-avoidance is not addressed; it is likely that the view-to-view paths with collision risk are among the least efficient path segments, crossing the sphere at nearly its full diameter. A more recent view planning system initializes all views along the surface of an enclosing sphere and iteratively adjusts them using a genetic algorithm [30]. An ordering of views is computed with an emphasis on accurately describing the cost of each view-to-view path based on the required joint movements of the viewing robot. The authors state that the distance between each pair of views is computed, but a method for collision avoidance is not discussed. In both of these methods all final views are joined by line-segment paths; we are left to assume that view-to-view paths requiring more than a simple line segment are not considered.

Both of these studies emphasize the computation of a high-quality, sub-optimal solution of the *traveling salesman problem* (TSP), the problem of finding, given a list of "cities" and the distances between them, the shortest route that visits each city once and returns to the original city [106], [9]. Although the TSP is an NP-hard problem, many heuristics have been used to produce high-quality solutions in short

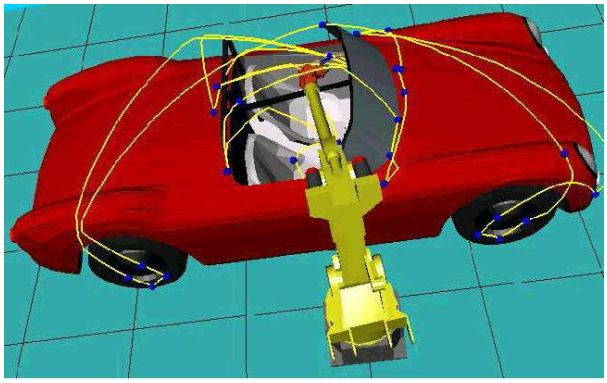
time, including the Lin-Kernighan heuristic [110], simulated annealing [93], ant colony optimization [48], and the algorithm of Christofides [39], which, for metric instances of the TSP, returns a solution within a factor of 1.5 of optimality. A metric instance of the TSP obeys the triangle inequality, which requires that no single edge of an inter-city triangle exceed the combined length of the two other edges.

Recent work in view planning proposes a method to combine the steps of view selection and view ordering into a unified optimization problem, the traveling view planning problem [164]. It is assumed that a full-coverage set of views, and the structure surfaces they observe, has already been catalogged and is available as data in the optimization problem. Weights are selected to penalize the cost of every added view and the cost of travel between views. An integer programming formulation is proposed along with a linear programming-based approximation algorithm. Unfortunately, implementation challenges exist because the approximation has an exponential number of constraints. Related classical problems are the generalized traveling salesman problem [60] and the covering salesman problem [46], which, using similar problem data, call for a minimum-length route, with no penalty on the number of views, that satisfies all coverage constraints. Solving the min-cardinality set cover problem, followed by the standard TSP, is recommended in [46] as a heuristic substitute and is supported with numerical data.

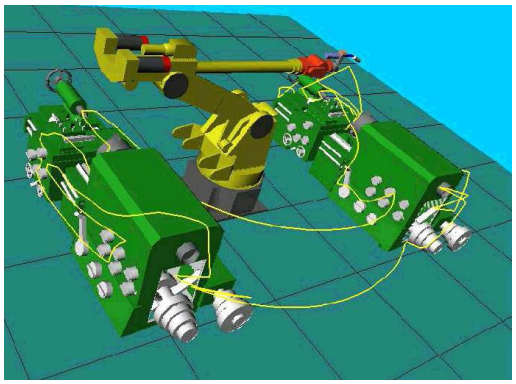
2.4 Multi-goal Planning

Multi-goal planning is the problem of planning a path or tour of minimum length that visits every configuration in a set of goals. It combines the challenges of the path planning problem introduced in Section 2.1 with the combinatorial complexity of the ordering problems discussed in Section 2.3. In this review of multi-goal planning, we focus exclusively on problems in which the identification of feasible interconnecting paths among the goal configurations is non-trivial.

Computing feasible interconnecting paths is often challenging due to the presence of obstacles in the robot workspace. The algorithm of Spitz and Requicha [146],



(a) Planned operations over an auto body, comprised of 31 goals.



(b) Planned operations over a pair of lathes, comprised of 50 goals.

Figure 2-7: Two examples of manipulator tasks requiring multi-goal planning, with the planned paths illustrated. Image credit: M. Saha *et al.*, 2006 [135].

designed for use with coordinate measuring machines, constructs a PRM in the robot C-Space until all goal configurations are connected to the roadmap. The all-pairs shortest paths problem can then be solved over the roadmap, giving the goal-to-goal distances that are required as input to the traveling salesman problem. Although feasible paths are computed in 3D Euclidean C-Space for problems of one hundred goals, the robot and obstacles possess simple geometries and only about one hundred nodes are needed to construct a PRM that reaches all goals.

The "lazy" algorithm of Saha et al. [134] is intended for multi-goal planning problems posed in high-dimensional C-Space. It is assumed that upwards of fifty goals will be required in the problem, and that the cost of computing a collision-free goal-to-goal path is high compared to the cost of computing an approximate TSP over known distances between goals. Under these assumptions, computing feasible paths between all goal pairings is prohibitively high, and so paths are only computed on an as-needed basis using a single-query sampling-based planner. On every iteration of the algorithm, a TSP tour is computed using naively assumed goal-to-goal costs based on shortest paths in the absence of obstacles. Once a goal pairing is used in the TSP tour, the true cost of a collision-free path is substituted and the algorithm is repeated until the tour length falls below a desired threshold. Paths planned using this algorithm are pictured in Figure 2-7.

The amount of "laziness" suitable in a multi-goal planning problem has been explored through a computational comparison of the above algorithms with an ant colony optimization technique [51]. The ant colony algorithm was designed to achieve a compromise between Saha's $O(n)$ path queries per iteration and Spitz and Requicha's $O(n^2)$ path queries performed in total. The number of "ants" used in the optimization serves as a multiplicative coefficient of the lazy algorithm's $O(n)$ complexity and this approach achieves a high-performance compromise offering the lowest sum of computation time and robot mission time among the three algorithms.

Work of Wurll et al. [168] and later work by Saha et al. [135] focused on multi-goal planning over *goal groups*. The goals of the planning problem are posed in the workspace rather than the C-Space, and there are many possible robot configurations that map to each goal. Under this assumption, a group of goals is generated in C-Space in which all members map to the desired goal in the workspace. The planning problem is now one of choosing only a single member from each goal group to complete the required task, which in this case is spot-welding performed by a robot manipulator arm.

Finally, in some problems computing feasible interconnecting paths is challenging because of kinematic and dynamic constraints governing the robot's behavior, rather than the presence of obstacles. Savla et al. have developed methods for solving the minimum-time TSP for robots governed by nonholonomic [140] and double-integrator [139] constraints.

2.5 Coverage Path Planning

We now review coverage path planning, a problem in which several of the techniques from prior sections are used to plan a low-cost, feasible path over which a robot sweeps the surface of a required structure using its end effector. We divide coverage path planning algorithms into two categories, discrete and continuous. Discrete coverage path planning entails view planning as a first step, followed by multi-goal planning to join the discrete set of views into a feasible path. Much of the work that con-

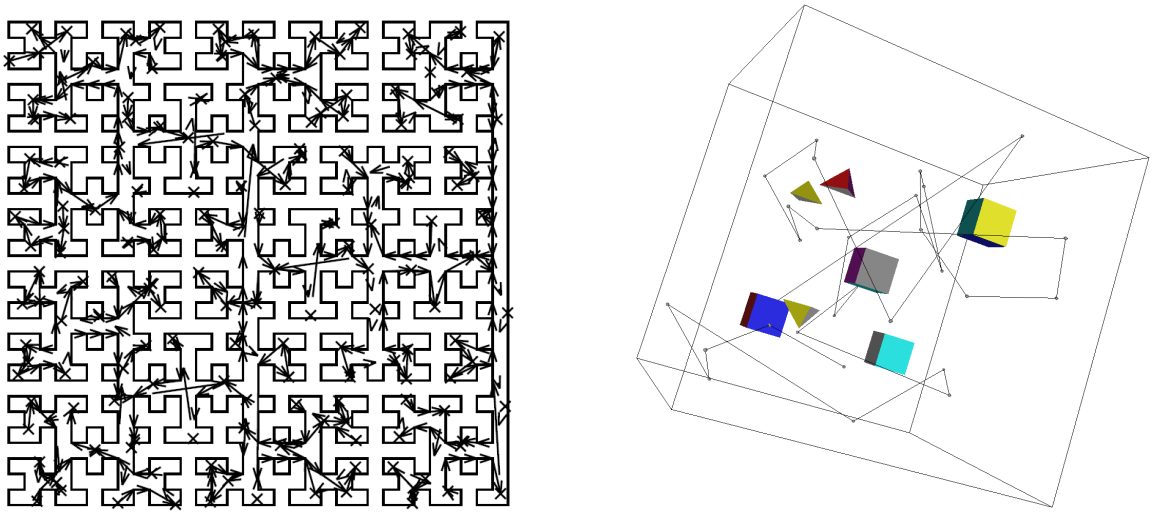
tributes to this area has already been covered in Sections 2.2, 2.3, and 2.4; here we present the few algorithms that have integrated these methods. Continuous coverage path planning employs continuous sensing or deposition by the end effector along the trajectories followed, and its methods resemble classical path planning to a greater extent than model-based view planning. Concepts such as cell decomposition, the generalized Voronoi diagram (GVD), and grid-based planning have been adapted to avoid obstacles and also satisfy coverage constraints.

2.5.1 The Watchman Route Problem

The coverage path planning problem bears some similarity to the classical *watchman route problem*. Given a polygon whose internal area must be observed by a set of infinite-range "guards", the watchman route problem calls for the shortest continuous cyclical route along which the entire polygon is viewed. A guard can collect the required views from any location along the continuous route. If the starting location of the route is specified, this problem can be solved to optimality in polynomial time over simple polygons [34]. Despite this result on simple polygons, the problem becomes NP-hard if holes are added to the interior of the polygon or if the problem is posed in 3D over a simple polyhedron [33]. Additionally, if the guards have a limited-range field of view, only approximate solutions have been found in polynomial time [122]. A version of this problem in which views are only collected at discrete locations along the route, termed the *generalized watchman route problem*, is also NP-hard [165]. As these latter cases are more representative of real-world coverage planning problems, modern algorithms often sacrifice the pursuit of optimality for fast heuristics that produce high-quality feasible solutions.

2.5.2 Discrete Coverage Path Planning

The watchman route algorithm of Danner and Kavraki [47] is a discrete planning method inspired by the classical watchman route problem. First, a set of full-coverage guards is selected using the dual sampling algorithm of Gonzalez-Baños and Latombe.



(a) Planned path for coverage of a complex 2D polygonal workspace.

(b) Planned path for coverage of several simple polyhedra.

Figure 2-8: Two examples of discrete coverage path planning using the watchman route algorithm of Danner and Kavraki. Image credit: T. Danner and L.E. Kavraki, 2000 [47].

After guards are selected, they are joined into a visibility graph in 2D instances of the problem and a PRM in 3D instances of the problem. The TSP is then solved approximately by computation of the minimum spanning tree (MST) [39] over the goals in the roadmap. This is the earliest work we are aware of that combines model-based view planning, multi-goal planning over obstacles and the ordering of views into a single integrated algorithm. Examples of planned coverage paths are pictured in Figure 2-8.

Other discrete algorithms have focused on 2D workspaces exclusively. The boundary placement heuristic of Faigl et al. maps all obstacles into C-Space and traces an initial, continuous coverage path along the obstacle boundary, offset by the robot's maximum sensing range [56]. This boundary path is populated with discrete robot configurations, the remaining gaps in coverage are iteratively filled, and obsolete guards are removed prior to planning of a final inspection path. The authors demonstrate, through computational results, that their view planning algorithm yields shorter paths on average than both the randomized dual sampling method and a geometric structure partitioning method [90]. Discrete, 2D coverage path planning

has also been performed for multi-robot deployments, in which a set of full-coverage views is planned and subsequently partitioned among a team of robots [58].

2.5.3 Continuous Coverage Path Planning

Planning in 2D Workspaces

A breadth of continous coverage algorithms have been developed for use in a 2D Euclidean C-Space populated with obstacles, with the goal of covering $\mathcal{C}_{free}$ as efficiently as possible [35]. Choset's boustrophedon cellular decomposition algorithm solves this problem over polygonal obstacles by dividing $\mathcal{C}_{free}$ into cells that are individually covered by efficient back-and-forth sweeping motions [38]. The algorithm seeks a decomposition with as few cells as possible to limit the number of times the contiguous sweeping motions are interrupted. Related work by Huang explored reducing overall path length by orienting sweep paths differently in different cells [79]. Recent work by Mannadiar and Rekleitis improves efficiency further by ensuring that no piece of terrain is covered twice within cells that must be visited twice during execution of the coverage path [116].

To generate the cells used in the solution, these and other cell decomposition strategies propagate a "slicing function" through $\mathcal{C}_{free}$ to identify critical points where the connectivity of $\mathcal{C}_{free}$ changes across the slice. Critical points, which occur when a new obstacle first intersects the slice or when an existing obstacle departs from the slice, mark ideal locations for the creation of cell boundaries. The concept of a slicing function has been generalized for obstacles of curved geometry, using a variety of functions that yield curved, radial, and spiral-shaped robot sweep patterns [3].

Cell decomposition has been used in concert with GVDs as part of a hybrid strategy for planning in workspaces that include both open areas and narrow corridors [2]. In open areas with sufficient clearance, a robot is assumed to possess enlarged boundaries that extend to the limits of its range sensor, and coverage is planned using cell decomposition. In low-clearance areas where the sensor has effectively infinite range, constructing a GVD and following its edges is sufficient to achieve coverage.

An alternative, grid-based method for 2D coverage plans a non-cyclical coverage route using a given a start configuration and goal configuration [170]. A cost is assigned to each grid cell based on its distance from the goal, and the robot moves in the direction of greatest cost increase subject to the requirement that no cell is visited more than once. The spanning tree covering method of Gabriely and Rimon also discretizes the interior of $\mathcal{C}_{free}$ using a grid of small, identically-sized squares [64]. A graph is generated representing the connectivity of the grid, and the minimum spanning tree is computed over this graph. A coverage path is generated by tracing a route around the perimeter of the spanning tree.

Algorithms have also been developed for achieving full coverage of *a priori* unknown 2D environments. The cellular decomposition of 2D C-Space has been computed incrementally as a robot explores its environment along back-and-forth sweep paths, enabling online planning and verification of sensor coverage [1]. Full coverage of unstructured environments has been achieved experimentally using this method with application to demining [4]. An algorithm designed for surveying the ocean floor using an AUV, which requires no *a priori* knowledge of the environment, achieves coverage by sweeping along the lines of a pre-determined grid and tracing the boundary of workspace obstacles when they are encountered [71]. The algorithm is also capable of planning paths that change depth along grid lines to accomodate altitude changes in seafloor terrain.

Also of note are several 2D coverage path planning algorithms designed for multi-robot deployments. Cell decomposition has been extended to multi-robot coverage planning using the novel solution of parallel paths swept by robots in formation [82]. In a related procedure, a cell decomposition of $\mathcal{C}_{free}$ is divided among a team of robots with limited communications, including the assignment of cells to different groups of robots and the team-based sweeping of individual cells [131]. Multi-robot coverage algorithms have also used the GVD as a tool for path planning, both for coverage of $\mathcal{C}_{free}$ [99] and the boundary of $\mathcal{C}_{free}$ [49].

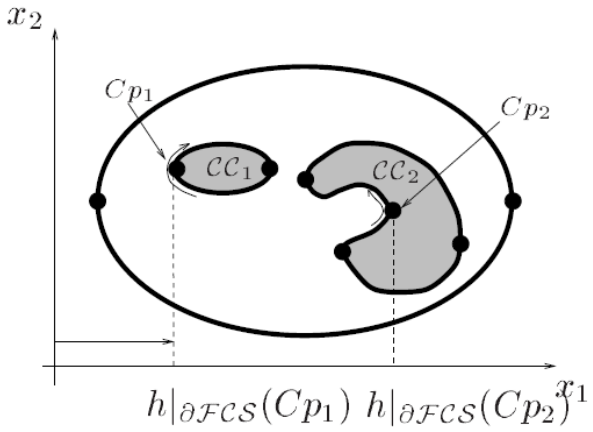
Planning in 3D Workspaces

Some algorithmic tools used for 2D coverage path planning have been adapted for use in 3D workspaces. Cell decomposition has been developed for coverage planning over 3D structures by combining a series of planar 2D "coverage loops" into a full 3D inspection [10]. A 2D slicing function is propagated through the 3D workspace to identify critical points where a change in the topology of the coverage loop is required. Examples of 2D and 3D coverage paths planned using cell decomposition are pictured in Figure 2-9. If an infinite-range sensing assumption applies to a robot, then the hierarchichal generalized Voronoi graph, an adaptation of the GVD for higher-dimensional C-Space, can be used to plan for full coverage of a 3D workspace, even if the workspace is *a priori* unknown [36].

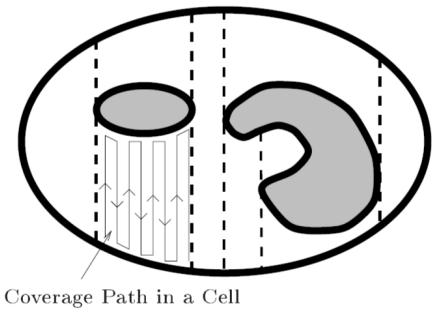
Other 3D coverage planning algorithms have been developed with specific applications in mind. A specialized planning method for auto body painting segments a car model into pieces of individually simple topology [11], and coverage paths are planned over the segments with the goal of minimizing geodesic curvature to achieve uniform paint deposition [12]. Robot dynamics are considered in an algoritm for planning minimum-time coverage of the exterior of buildings by a team of unmanned aerial vehicles (UAVs), which covers the buildings using a series of planar looping trajectories [32]. A sampling-based method for planning a marine structure inspection by an AUV iteratively constructs a full-coverage roadmap, from which a set of linear path segments is selected and pieced together into a contiguous inspection route [50]. A five-DOF robot C-Space is considered in this problem, and each line-segment path utilizes a different orientation of the limited-field-of-view acoustic sensor.

2.6 Summary

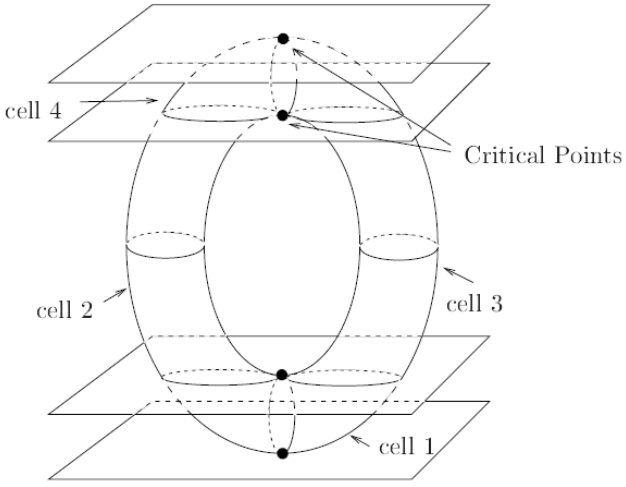
In this section we presented a comprehensive review of relevant prior work in path planning, view planning, multi-goal planning, and coverage path planning. Two clear perspectives emerge in the review of these works: coverage can be achieved using a discrete set of configurations, and coverage can be achieved over a continuous trajec-



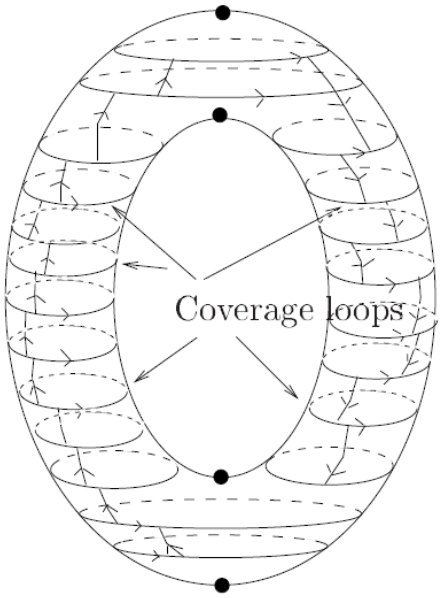
(a) Critical points computed by sweeping a vertical line across the pictured 2D C-space.



(b) Cell decomposition based on 2-9(a) with coverage of one cell illustrated.



(c) Critical points computed by sweeping a horizontal plane through the pictured 3D C-Space.



(d) Cell decomposition based on 2-9(c) with coverage loops illustrated.

Figure 2-9: Examples of 2D and 3D cell decompositions used for continuous coverage path planning. Image credit: E.U. Acar *et al.*, 2002 [3].

tory. Most often, the former perspective is motivated by a sensing task, specifically one in which collecting a sample requires non-trivial time or a stationary robot. The latter perspective is motivated by higher-bandwidth tasks in sensing, deposition, and removal, in which continuous-time assumptions are accurate. Despite this difference in application, algorithms in both categories share the burden of finding collision-free

paths, and satisfying challenging coverage constraints. In the three following chapters, we introduce new planning algorithms and discuss the merits of each in the context of this prior work. Each new algorithm satifies a compelling, unmet need motivated by the application of planning an autonomous in-water ship hull inspection.

Chapter 3

Planning Feasible Inspection Tours

3.1 Introduction

As introduced in Chapter 1, our coverage application is the autonomous in-water inspection of a ship hull, a 3D structure with challenging complexity at the stern due to shafts, propellers, and rudders in close proximity to one another and to the hull. The Bluefin-MIT Hovering Autonomous Underwater Vehicle is tasked with inspecting 100% of the surface area at the stern using a DIDSON. The vehicle is fully actuated and precision-maneuverable, but it cannot fit into the spaces between the component structures at the stern. As a result, most of the prior methods for 3D coverage path planning are unsuitable.

Several of the 2D coverage algorithms reviewed in Chapter 2 can be applied iterativley as "2-and-a-half-D" (2.5D) algorithms. A 3D structure can be partitioned into a series of 2D slices, and the boundary of each slice can be covered using an appropriate 2D path planning or view planning algorithm. Examples of 2D algorithms suitable for this purpose are a GVD-based boundary coverage algorithm [49] and the dual sampling algorithm for randomized view planning [66]. In addition, some algorithms designed explicitly for 3D coverage path planning rely on 2.5D strategies, including a 3D cell decomposition algorithm [10] and a method for the exterior inspection of buildings [32], both of which design planar looping trajectories for 2D cross-sections of a structure. Related to 2.5D algorithms are 3D coverage algorithms that parti-

tion a structure in a non-planar way for planning over each separate module, such as a car-painting strategy that relies on structure segmentation [11], [12]. The 2.5D building inspection method also relies on segmentation when multiple buildings are involved. For both of these methods, it is assumed that there is no risk of collision along a path designed for an isolated component structure.

If we adopt a 2.5D approach for planning a ship hull inspection, there is no guarantee that a single "slicing" direction will allow access to all low-clearance areas. A 2.5D plan may need to be augmented with special, out-of-plane views to grant visibility of confined areas that are occluded or inaccessible in-plane. If a 3D modular approach is implemented, paths planned for component structures are at risk of collision with neighboring structures. It may not be possible to design a series of loops that fully covers a shaft due to limited clearance between the shaft and other component structures.

In consideration of these factors, we take a global optimization approach, in which all 3D protruding structures are considered simultaneously. The constraints are determined by the geometry of the 3D model provided as input. We use a triangle mesh, typically comprised of thousands of primitives, to accurately model a ship's running gear. Rather than explicitly optimizing robot configurations over the thousands of collision and visibility constraints posed by such geometry, sampling-based planning is used, employing random sampling to find feasible means for a robot to peer into low-clearance areas from a distance.

The watchman route algorithm of Danner and Kavraki [47] uses the global, sampling-based approach described above, providing a suitable starting point for coverage path planning over complex 3D structures. This algorithm has been used to plan paths that cover very simple polyhedra, and the final details of its 3D implementation are left by its authors as an area for future work. We present an algorithm that makes several extensions to this work, including effecient means for checking the visibility of geometric primitives over structures with large models and complex geometries. Our algorithm constructs a *redundant roadmap*, in which every geometric primitive is observed by multiple robot states. To enable fast planning over a large roadmap,

tools from multi-robot [137] and multi-goal [135] planning are used to enable lazy collision-checking.

Both the redundant roadmap algorithm and the watchman route algorithm construct a discrete set of stationary views to obtain full coverage. This is preferable for planning an autonomous ship hull inspection, as the presence of ocean disturbances increases the difficulty of executing a continuous sensing path with high precision. Using discrete coverage planning, the HAUV can stabilize at each individual waypoint before collecting a view, avoiding the need to double back to collect missed observations from a continuous sensing path.

A desirable property for a sampling-based planning algorithm is *probabilistic completeness*. If a feasible solution exists for a given problem, then a probabilistically complete algorithm will find a solution with probability that tends to one as the number of random samples tends to infinity [100]. This property has been proven for a variety of sampling-based path planning algorithms, including the probabilistic roadmap (PRM) [88] and the rapidly-exploring random tree (RRT) [105]. Probabilistic completeness has not been explored, however, in the context of coverage path planning. We propose a framework for analyzing the probabilistic completeness of a sampling-based coverage path planning algorithm, and we identify quantitative bounds on the probability of obtaining a feasible solution.

Our proposed roadmap construction and collision-checking procedures are presented in Section 3.2. In Section 3.3 we discuss the methods by which the set cover problem (SCP) and TSP are approximated in sequence to build an inspection tour from a redundant roadmap. In Section 3.4 we give an analysis of probabilistic completeness that applies to both the watchman route algorithm and the redundant roadmap algorithm. We then compare the computational performance of our algorithm with the watchman route algorithm over an ensemble of trials. In Section 3.5 we examine algorithm performance over Monte Carlo trials in which randomly-sampled primitives must be inspected by a point robot in an obstacle-free 3D workspace. In Section 3.6 we apply the inspection planning algorithms to a large-scale, real-world task, planning the inspection of a ship hull by the HAUV.

3.2 Sampling-Based Planning Procedure

We develop a feasible inspection tour by solving two sub-problems in series. The first problem entails sampling feasible robot configurations that together give 100% coverage of a structure boundary, which we term the *coverage sampling problem* (CSP). The CSP differs from the classical art gallery problem [124] because it does not require the selection of a minimum cardinality set, but merely a feasible covering set. After a set of configurations from the CSP is selected for traversal, the second problem requires the linking of these configurations with feasible paths, which we refer to as the *multi-goal planning problem* (MPP).

Our roadmap construction algorithm for solving the CSP uses random sampling to create a discrete state space of tunable resolution from which the inspection path will be made. To solve the MPP, a point-to-point planner is applied iteratively to connect the configurations on the roadmap with feasible paths. The MPP algorithm is "lazy", finding a quality solution without computing paths for all point-to-point combinations. A stateflow diagram summarizing the coverage path planning procedure from start to finish is given in Figure 3-1. The watchman route algorithm is also illustrated in Figure 3-1 for the purposes of comparision.

3.2.1 Motivation

Danner and Kavraki's watchman route algorithm uses the dual sampling method of Gonzalez-Baños and Latombe as a key subroutine. In their work on the subject of sampling-based view planning [66], [67], Gonzalez-Baños and Latombe describe two strategies for achieving sampling-based coverage: sampling C-Space at random until the workspace boundary is covered, and sampling from the workspace boundary itself and selecting views that map to sightings of each boundary location. In the former case, the min-cardinality set cover is approximated over a large group of random samples. In the latter case, dual sampling, the set cover is pieced together incrementally and is not solved in a single batch step.

The dual sampling method selects in each iteration a geometric primitive that

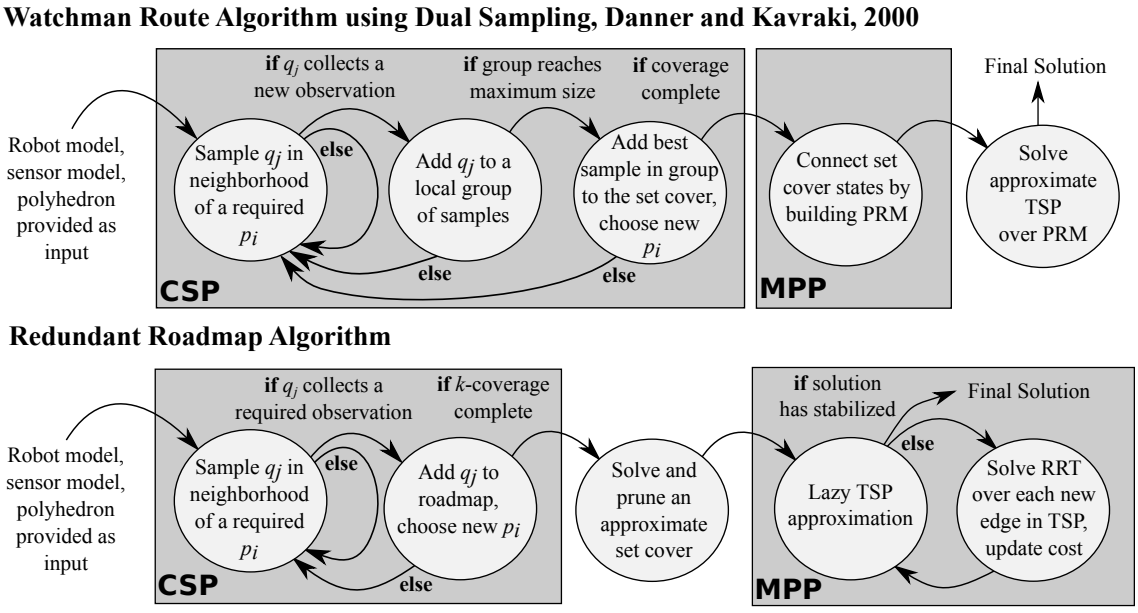


Figure 3-1: A stateflow diagram illustrating two algorithms for feasible sampling-based coverage path planning, highlighting the subroutines that solve the CSP and MPP subproblems.

has not been observed, and samples in a local neighborhood of C-Space that maps to feasible views of this primitive. Samples are drawn until a group of sufficient size is collected in which each sample observes at least one required primitive. The sample that contributes the largest quantity of new sensor information is immediately added to the set cover, and the rest of the group is discarded. Local sampling continues elsewhere until every primitive is observed at least once. The key tunable parameter of dual sampling is the *number of local samples* that is drawn in the neighborhood of each geometric primitive.

Gonzalez-Baños and Latombe also propose a tunable parameter for the batch case: a limit on the maximum number of samples drawn in C-Space. A sampling limit will allow a user to obtain improved set cover outcomes in exchange for a greater investment in sampling, but there is no guarantee that the designated number of samples will achieve full coverage of the workspace boundary. Our aim is to develop an algorithm that uses the best features of both methods. Our proposed redundant roadmap algorithm samples the workspace boundary like the dual sampling method, but it solves the set cover in a single batch step.

3.2.2 Roadmap Construction

Our proposed algorithm adds configurations to a roadmap until each geometric primitive is observed a requisite number of times, which we term the *redundancy* of the roadmap. Construction begins with the selection of a geometric primitive that has not been observed the required number of times. Robot configurations are sampled uniformly at random in a local neighborhood of this primitive, avoiding exhaustive sampling in empty portions of the workspace. A configuration is added to the roadmap if it collects at least one required observation, and if the configuration is free of collisions and occlusions. In addition to collision-checking, this requires ray shooting; casting a line segment between the robot's sensor and each of the primitives inside the sensor footprint to ensure the line of sight is clear. After a configuration is added to the roadmap, another primitive is selected, and the procedure repeats until the redundancy requirement is satisfied. The full roadmap construction procedure is detailed in Algorithm 1.

Increased redundancy is intended to create a finely discretized state space from which a smaller covering subset of robot states is chosen. This procedure stands in contrast to dual sampling, in which the final set of configurations used in the inspection is pieced together one-by-one, and many candidate samples are discarded before complete coverage is achieved. The aim of constructing a redundant roadmap is to conserve the amount of collision-checking and ray shooting required in the solution of a 3D coverage problem, while preserving a means for tuning the performance of the algorithm.

3.2.3 Lazy Point-to-Point Planning

Once a set of views is selected from the roadmap, they must be joined together into a contiguous, collision-free inspection route. The watchman route algorithm achieves this by building a PRM that joins all view configurations into a single connected component; the TSP is then approximated using the all-pairs shortest path lengths among the view configurations. This approach, formalized by Spitz and Requicha

Algorithm 1 $ConfigList = BuildRoadmap(Primitives, Obstacles, Redundancy)$

```
1: IncompletePrimitives ← Primitives
2: while IncompletePrimitives ≠ ∅ do
3:    SeedPrimitive ← ChooseRandomEntry(IncompletePrimitives)
4:    NewConfig ← FeasibleSample(SeedPrimitive, Obstacles)
5:    NewSightings ← Sensor(NewConfig, Primitives, Obstacles)
6:    NeededSightings ← NewSightings ∩ IncompletePrimitives
7:    if NeededSightings ≠ ∅ then
8:       ConfigList.add(NewCfg, NewSightings)
9:       for i ∈ NeededSightings do
10:         NeededSightings[i].incrementNumSightings()
11:         if NeededSightings[i].numSightings = Redundancy then
12:            IncompletePrimitives ← IncompletePrimitives \ NeededSightings[i]
13:         end if
14:      end for
15:   end if
16: end while
17: return ConfigList
```

[146], requires extensive sampling if the individual paths from view to view are to be well-formed, since there are $O(n^2)$ individual view-to-view paths that may be selected in a tour that traverses n views.

An alternate approach developed by Saha et al. emphasizes the construction of high-quality paths among a small subset of $O(n)$ view-to-view pairings [134], [135]. This method assumes the cost of computing an approximate TSP solution is minor compared to the cost of building feasible paths, and it is intended for problems of high dimension and complex geometry. This assumption holds true in our application of interest, in which the number of views is relatively small (about one-to-two hundred for a typical hull inspection problem), but the cost of path planning is high among complex structures with hundreds of thousands of primitives.

Efficient computation of a feasible tour is achieved with a lazy algorithm adapted from this work. As the redundant roadmap of views is constructed, an adjacency matrix is maintained in which all entries represent the Euclidean norms among roadmap nodes. Computation of a Euclidean norm is far simpler than performing collision-checking along every possible view-to-view path. An initial inspection tour is computed over this naive adjacency matrix, and only the edges selected in the tour are

Algorithm 2 $RobotTour = LazyTourAlgorithm(Nodes, Obstacles)$

$AdjMat \leftarrow EuclideanDistances(Nodes)$
$UnclearedEdges \leftarrow GetEdgePairs(Nodes)$
$ClearedEdges \leftarrow \emptyset$
while $NewTourCost \neq PreviousTourCost$ **do**
  $PreviousTourCost \leftarrow NewTourCost$
  $NewTourCost \leftarrow 0$
  $LazyTour \leftarrow ComputeTour(AdjMat)$
  for $Edge_{ij} \in LazyTour$ **do**
    if $Edge_{ij} \in UnclearedEdges$ **then**
      $FeasiblePath_{ij} \leftarrow RRT(Edge_{ij}, Obstacles)$
      $ClearedEdges \leftarrow ClearedEdges \cup Edge_{ij}$
      $UnclearedEdges \leftarrow UnclearedEdges \setminus Edge_{ij}$
      $AdjMat(i,j) \leftarrow PathCost(FeasiblePath_{ij})$
    end if
    $NewTourCost \leftarrow NewTourCost + AdjMat(i,j)$
  end for
end while
$RobotTour \leftarrow LazyTour$
return $RobotTour$

collision-checked, rather than every edge of the roadmap. The bi-directional rapidly-exploring random tree (RRT) [98] is used as the point-to-point planner. The computation of RRTs over the edges of the inspection tour increases the lengths of some edges. To address this, an iterative solution procedure, similar to that in [134], is utilized. After the first set of feasible paths is obtained, the costs in the adjacency matrix are updated, and the inspection tour is recomputed using the new costs. This procedure is repeated, and goal-to-goal costs are iteratively updated, until there is no further improvement in the length of the returned path. This procedure is detailed in Algorithm 2.

3.3 Combinatorial Optimization Procedure

In the development of the redundant roadmap algorithm, we assume that an inspection route is optimal if it minimizes the total duration of the inspection. Time is spent traveling the length of the route, and also collecting the planned sensor view at each node along the route. Stated as an integer programming problem, minimizing

the total duration of an inspection tour, given a full-coverage roadmap with assumed node-to-node distances, would require multiplicative weights on both the number of views and the length of the tour in the problem's cost function. These weights would be determined by the cost of collecting a sensor view relative to the cost of travel per unit distance. This problem, called the *traveling view planning problem* (Traveling VPP), has been studied by Wang et al. [164], who have proposed a linear programming rounding algorithm for finding an approximate solution. Their advocacy of this approach is based on the argument that decoupling the solution of the Traveling VPP into two sequential steps, the min-cardinality set cover and the TSP, can give arbitrarily poor solutions.

This is true, and becomes increasingly problematic, when the robot's field of view approaches infinite range and the boundaries of its environment are near-infinite relative to the size of the structure being inspected. Conversely, our application of interest concerns a robot with a limited sensing radius (3-5 meters) operating in a confined environment with boundaries on the order of tens of meters. As a result, we plan a full-coverage inspection route by approximating the SCP and TSP in sequence. This sequential approach is an effective heuristic for the *covering salesman problem*, a classical problem embedded with finite-range, geometric coverage constraints in which all "cities" given must lie within a required minimum distance of a city selected for the tour [46]. In our implementation of this procedure, the SCP is solved once, and the TSP is solved iteratively using the method described in Section 3.2.3. Solving the SCP only once limits the number of point-to-point path queries posed over the problem's complex geometry, which would be much greater if a Traveling VPP were solved in each iteration. Below we discuss the methods used to approximate the SCP and TSP in sequence.

3.3.1 Set Cover Subproblem

To solve the set cover subproblem, we rely on polynomial-time approximation algorithms that find solutions within guaranteed factors of optimality. We consider two such algorithms, a greedy algorithm and a linear programming (LP) rounding algo-

rithm. The greedy algorithm simply adds to the set cover, on each iteration, the roadmap node with the largest number of observed primitives not yet in the cover [85], [113], [40]. This algorithm solves the SCP within a factor of optimality that is bounded above by $\ln(m) + 1$, where m is the number of primitives required in the inspection. The rounding algorithm [72] solves the LP relaxation of the SCP, and then rounds the fractional solution according to a simple rule: if f is the largest number of roadmap nodes which share sightings of a primitive, then any roadmap node whose fractional decision variable is greater than or equal to $1/f$ is included in the cover. This method is guaranteed to return a solution within a factor f of optimality.

In the ship hull inspection example to be presented below, there are more than 10^5 primitives required in the inspection, giving a greedy algorithm approximation factor of about 12.5. At the same time, a typical value of f on a representative roadmap for this task is about twenty. Since these are both fast algorithms, and the approximation factors are of the same order, we will compare the two to assess their performance in practice.

Although both algorithms produce feasible solutions, these can often be pruned to yield feasible solutions of smaller size. Our pruning procedure, which runs in $O(n^2m)$ time, identifies configurations in the set cover that observe no geometric primitives uniquely, and in each iteration one of these configurations is randomly selected and pruned from the cover. The procedure repeats until every configuration in the cover is the unique observer of at least one geometric primitive.

3.3.2 Traveling Salesman Subproblem

To solve the TSP subproblem, we rely on another polynomial-time approximation. The algorithm of Christofides [39] computes the minimum spanning tree (MST) over a graph, and then a minimum-cost perfect matching over the odd-degree nodes of the MST, achieving an approximation factor of 1.5 when the triangle inequality holds over the roadmap. Although our lazy computation procedure may occasionally violate the triangle inequality, RRT post-optimization smoothing ensures that there are no paths from a roadmap node i to a roadmap node k such than an alternate path from i to

some node j to k is dramatically shorter. This assumption has proven successful in MST-only variants (with factor-2) for single and multi-agent coverage planning [47], [58], as well as pure multi-goal planning [134].

The Christofides approximation gives a good starting point for the TSP, but we also utilize a post-optimization improvement heuristic. Heuristics such as the Lin-Kernighan algorithm [110], which iteratively improves a TSP solution by swapping groups of edges, have succeeded in finding fast, high-quality solutions to very large TSP instances in practice [9]. We apply the chained Lin-Kernighan improvement procedure [8] for a short period of time after computation of each inspection tour.

3.4 Analysis of Sampling-Based Planning of Feasible Coverage Paths

Here we analyze the sampling-based solution of robot coverage path planning. The analysis of probabilistic completeness in this section is designed for compatibility with both our proposed algorithm and the watchman route algorithm of Danner and Kavraki. Both the watchman route and redundant roadmap algorithms solve the CSP by randomly sampling configurations until the required structure is covered, although the latter algorithm does not terminate until coverage of multiplicity k is achieved among the configurations in its roadmap. In the analysis of the CSP to follow, which is the major contribution of this section, we will assume that k-coverage is required so the analysis will apply to both algorithms.

The two algorithms also differ in their solution of the MPP. The watchman route algorithm connects the nodes in the set cover using a PRM. The redundant roadmap algorithm employs an iterative solution of the RRT over all goal-to-goal paths in the tour. Our analysis of probabilistic completeness will address both methods for solution of the MPP, drawing largely on existing results on the completeness of the individual PRM and RRT.

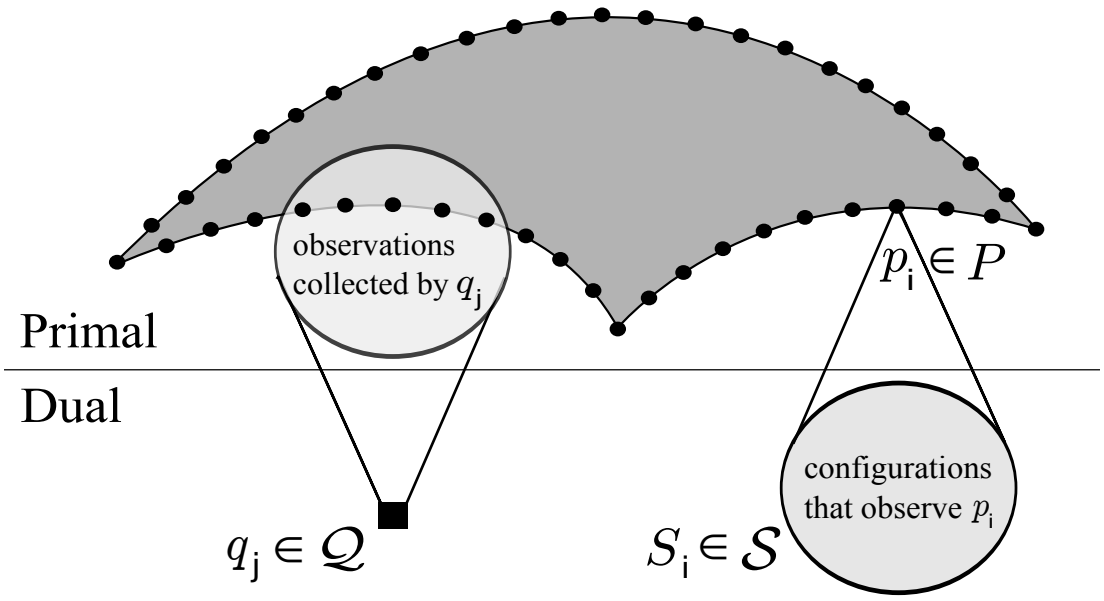


Figure 3-2: An illustration of the primal and dual set systems in the coverage sampling problem. Robot configurations q_j are used in both systems; the primal set cover problem employs the sensor observations collected at q_j and the dual hitting set problem employs the physical state of q_j. The primal (primitive) space is discrete and the dual (configuration) space is continouous.

3.4.1 Set Systems and the CSP

We will represent the coverage sampling problem using the *set system* $(P, \mathcal{Q})$, also known as a *range space*. P is a finite set of geometric primitives p_i comprising a structure that that must be covered by the robot. $\mathcal{Q}$ is the robot configuration space. Every feasible configuration $q_j \in \mathcal{Q}$ maps to a subset of P viewed by the robot's sensor. These sets of observed primitives are known as *ranges*. Given a finite set of ranges from $\mathcal{Q}$, the set cover problem calls for the minimum number of configurations q_j such that all elements $p_i \in P$ are covered.

The problem can also be modeled using the *dual set system* $(\mathcal{Q}, \mathcal{S})$, where $S_i \in \mathcal{S}$ is the set of feasible robot configurations in $\mathcal{Q}$ that obtain views of the primitive $p_i \in P$. Given a finite set of robot configurations from $\mathcal{Q}$, the *hitting set problem* calls for the minimum number of configurations q_j such that at least one configuration lies in every S_i for all $p_i \in P$. The structure of the primal and dual set systems for a robot coverage sampling problem is illustrated in Figure 3-2.

The set system modeling language was developed for analyzing the number of samples required to cover the ranges of a set system, as a function of their size [161],[70]. In the analysis most closely related to probabilistic completeness, set systems have been used to analyze the number of samples required for high-probability floor coverage by a random sensor network [80]. Set systems have also been used in the study of set cover and hitting set approximation algorithms, [24], with several applications to robot coverage and sensor placement [67],[81],[62].

Our analysis differs from prior work due to its emphasis on covering a discrete collection of primitives rather than the full continuous surface of a structure. The analysis requires only two scalar parameters to describe the difficulty of a coverage problem: the total number of geometric primitives, and a ratio comparing the volumes of the C-Space region being sampled and the smallest subset of views with a single primitive in common. A continuous analysis, on the other hand, depends heavily on the geometry of the robot sensor's field of view, the dimensionality of the workspace, and the available degrees-of-freedom for positioning the sensor in the workspace. After presenting our results on probabilistic completeness below, we will discuss the procedures required to obtain a comparable continuous result. The continuous case is also presented in greater detail in Appendix A.

We now formally define the coverage sampling problem:

Definition 1 (Coverage Sampling Problem)**.** *Let P be a finite set of discrete geometric primitives p_i comprising a structure to be inspected. Let the infinite set $\mathcal{Q}$ be the robot configuration space whose configurations $q_j \in \mathcal{Q}$ map to observations of the Euclidean workspace which contains P. Let integer k be the number of times each $p_i \in P$ must be viewed. Find a finite set of feasible configurations $N \subset \mathcal{Q}$ that obtains at least k distinct views of all $p_i \in P$.*

Let's now assume that an algorithm has been proposed for solution of the CSP using a random sampling scheme in a d-dimensional Euclidean C-Space. We define the property of *probabilistic completeness* for a CSP algorithm as follows.

Definition 2 (Probabilistic Completeness of a CSP Algorithm)**.** *Let CSA be a pro-*

posed coverage sampling algorithm for the CSP. Let $(\mathcal{Q}, \mathcal{S})$ be the dual set system over which the CSP is defined. Let $\delta = \min_{S_i \in \mathcal{S}} \mu(S_i)/\mu(\mathcal{Q})$ be the volume fraction of the smallest range in $\mathcal{S}$, where the measure μ represents the volume of the specified region of configuration space. If, when $\delta > 0$, the probabability that at least k samples have landed in every $S_i \in \mathcal{S}$ approaches one as the number of samples of $\mathcal{Q}$ drawn by CSA approaches infinity, then CSA is probabilistically complete.

This definition implies that if a feasible CSP solution exists, a probabilistically complete CSP algorithm will find a feasible solution in the limit. In fact, we employ a rather strict definition of feasibility that deems a CSP to be feasible only if the smallest range in $\mathcal{S}$ has nonzero volume. This eliminates degenerate instances of the CSP from consideration, in which some point $p_i \in P$ can only be viewed from a manifold in $\mathcal{Q}$ of lower dimension than $\mathcal{Q}$ itself.

3.4.2 Probabilistic Completeness of the CSP

We can analyze probabilistic completeness by studying the simple event of whether a randomly-sampled configuration q_j lands in a particular range $S_i \in \mathcal{S}$. We will assume throughout the analysis that some subset of the configuration space $A \subseteq \mathcal{Q}$, which is relevant for the inspection task, is chosen for sampling. A is often comprised of the region of $\mathcal{Q}$ that is within sensor viewing range of the structure. The probability of a sample q_j landing in S_i is equivalent to the ratio $\mu(S_i \cap A)/\mu(A)$. Using these preliminaries, we give the following theorem on probabilistic completeness.

Theorem 1 (Completeness and Convergence of the Discrete CSP)**.** *Any algorithm for the CSP that samples uniformly at random from an infinite subset $A \subseteq \mathcal{Q}$ such that $\mu(S_i \cap A)/\mu(A) \geq \epsilon > 0 \; \forall S_i \in \mathcal{S}$ is probabilistically complete. Additionally, the probability that a feasible solution has not been found after m samples is bounded such that*

$$Pr[FAILURE] < |P| \cdot \frac{e^k}{e^{\,m\epsilon/2}} \quad , \tag{3.1}$$

where $|P|$ is the number of geometric primitives $p_i \in P$.

Proof. The probability of m samples producing a feasible CSP solution is equivalent to the probability that at least k random samples have landed in every range $S_i \in \mathcal{S}$. This fails to occur if there is at least one S_i in which fewer than k samples have landed. To model this event, we define the binomial random variable $X_i = X_{i_1} + X_{i_2} + ... + X_{i_m}$, which gives the number of samples that have successfully landed in S_i out of m total trials. We express the probability of CSP algorithm failure as follows:

$$\begin{aligned} Pr[FAILURE] &\leq Pr\left[\bigcup_{i=1}^{|P|} X_i < k\right] \\ &\leq \sum_{i=1}^{|P|} Pr[X_i < k] \\ &\leq |P| \cdot Pr[X_{i^*} < k] \end{aligned} \tag{3.2}$$

Using the union bound, the probability that $X_i < k$ for at least one S_i is bounded above by the sum of the probabilities of this event for all $S_i \in \mathcal{S}$. This is further simplified by taking $Pr[X_{i^*} < k]$ as an upper bound on the failures of all X_i, where X_{i^*} is the binomial random variable corresponding to the range in $\mathcal{S}$ that minimizes $\mu(S_i \cap A)/\mu(A)$.

We next bound $Pr[X_{i^*} < k]$ using the Chernoff bound for the lower tail of a Poisson distribution, which accurately represents a binomial distribution for large numbers of samples:

$$Pr[X_{i^*} < \gamma \cdot \lambda] < e^{-\frac{(1-\gamma)^2}{2}\lambda}, \quad \gamma \in [0, 1) \tag{3.3}$$

The parameter $\lambda = m\epsilon$ is the expected number of Poisson successes and γ is a fractional coefficient of λ. If we choose $\gamma = k/m\epsilon$, this allows the product $\gamma \cdot \lambda$ to evaluate to k, the exact number of successes we wish to model. We can now simplify (3.3).

$$Pr[X_{i^*} < k] < e^{\frac{-m\epsilon}{2} + k + \frac{-k^2}{2m\epsilon}} \leq \frac{e^k}{e^{\,m\epsilon/2}} \tag{3.4}$$

Combining the result of (3.4) with (3.2), we obtain the desired relationship between m and the probability of failure:

$$Pr[FAILURE] < |P| \cdot \frac{e^k}{e^{\,m\epsilon/2}}, \quad \lim_{m\to\infty} |P| \cdot \frac{e^k}{e^{\,m\epsilon/2}} = 0 \tag{3.5}$$

Since $\mu(S_i \cap A)/\mu(A) > 0 \; \forall S_i \in \mathcal{S}$, $\epsilon > 0$ and the limit behaves as indicated in (3.5). □

The bounding methods used in this analysis have been used previously in other probabilistic completeness proofs. The union bound was used previously in the proof of completeness of the PRM [88], and the Chernoff bound was used in the proof of completeness of the RRT [105]. Our analysis requires *both* of these tools since we need to reach every $S_i \in \mathcal{S}$ and we must do so at least k times.

Implications of Analysis

Any algorithm to which Theorem 1 applies benefits from a probability of failure that decreases exponentially in the number of samples m. Theorem 1 applies to both the redundant roadmap algorithm and the watchman route algorithm as long as A is selected to allow $\epsilon > 0$ whenever $\delta > 0$. Both algorithms sample from a subset $A \subseteq \mathcal{Q}$ that includes all areas where the robot's geometric sensor footprint intersects at least one $p_i \in P$, so this condition will always be satisfied.

It is also true that poor selection of A can result in the failure of a CSP algorithm to attain probabilistic completeness. Consider an algorithm which chooses a manifold A of lower dimension than $\mathcal{Q}$, such as a set of cross-sections in $\Re^2$ from a set $\mathcal{Q} \subseteq \Re^3$, which is often the strategy of 2.5D coverage algorithms. Even though $\mu(S_i)/\mu(\mathcal{Q}) > 0 \; \forall S_i \in \mathcal{S}$, it may be possible that $\mu(S_i \cap A)/\mu(A) = 0 \; \exists S_i \in \mathcal{S}$ and a 2.5D algorithm does not achieve probabilistic completeness.

In the application of autonomous ship hull inspection, which we explore in detail below, sweeping the stern of a naval ship for the purpose of mine detection demands coverage of a polyhedron comprised of several hundred thousand primitives. In a worst-case representative example where $|P| = 10^6$, $\epsilon = 10^{-3}$, and $k = 10$, the

probability in (3.1) drops from unity to 10^{-12} as the number of samples grows from 10^4 to 10^5. This quantity of samples is typically drawn over the course of one to two minutes of algorithm runtime.

In a continuous coverage analysis, m must surpass a threshold number of samples before such an exponential bound applies. This threshold, which is capable of taking on large values that exceed 10^5 samples, must be derived uniquely for different sensor geometries and constraints on sensor postioning. For a 3D structure observed by infinite field-of-view cameras positioned on the surface of an enclosing sphere [81], the threshold varies in $\Theta(log(v))$, where v is the number of vertices in a polyhedron P (we have used $|P|$ to refer to arbitrary primitives that are not necessarily vertices). In Appendix A we discuss this threshold, and the additional sampling it requires, in detail, giving relevant background on the geometric properties that figure in a continuous analysis.

It is conservative in any case, however, to assume that the worst-case volume fraction of $\epsilon = 10^{-3}$ describes the difficulty of observing all one million primitives. We can employ more detailed knowledge of the ranges $S_i \in \mathcal{S}$, but the potential for improved bounds on failure probability is limited. For example, we can establish a large volume fraction ϵ_{large} and a small volume fraction ϵ_{small}, such that ϵ_{large} bounds the volume fraction $\mu(S_i \cap A)/\mu(A)$ for the vast majority of ranges $S_i \in \mathcal{S}$, and ϵ_{small} bounds the volume fraction for a small minority of ranges. Splitting the summation in (3.2) into two additive terms with different coefficients, we obtain a more descriptive result in (3.6), framed in terms of the small number $n << |P|$ of ranges described by ϵ_{small}.

$$Pr[FAILURE] < (|P| - n) \cdot \frac{e^k}{e^{m\epsilon_{large}/2}} + n \cdot \frac{e^k}{e^{m\epsilon_{small}/2}} \tag{3.6}$$

With a coefficient n much smaller in magnitude than the total number of geometric primitives $|P|$, the ϵ_{small} term, which dominates (3.6) due to its slower decay rate, will fall sufficiently close to zero over a reduced number of random samples. The ϵ_{large} term, assigned most of the weight of $|P|$, will decay away at at a much faster

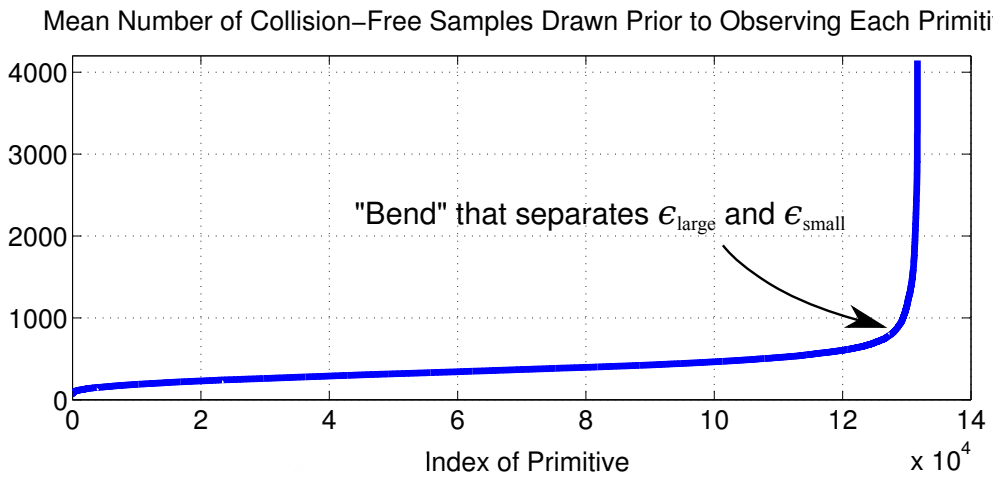


Figure 3-3: The geometric primitives in a 3D mesh model of a ship's stern are sorted according to the number of samples required to observe them over the course of constructing a redundancy-ten roadmap.

rate. It may be exhaustive to identify specific ϵ for all $S_i \in \mathcal{S}$, but identifying a small number of distinct categories may strengthen the result of the analysis.

In the one-million-primitive case described above, let us assume that a small cluster of primitives requires $\epsilon_{small} = 10^{-3}$, but the vast majority of primitives can be described by $\epsilon_{large} = 10^{-2}$. This assumption is derived from our application of interest: when the geometric primitives in the model of a ship's stern are sorted by the number of samples required to observe them, as is depicted in Figure 3-3, a "bend" empirically divides easy-to-observe and hard-to-observe geometric primitives, approximately ninety-seven percent to three percent. If we allow ϵ_{large} and ϵ_{small} to describe the easy-to-observe and hard-to-observe primitives, respectively, then the result of (3.6), assuming the problem of interest has a feasible solution, guarantees a 99.99 percent probability of successful completion after fifty thousand samples are drawn. If ϵ_{small} is applied to all primitives instead, then fifty-seven thousand samples are required to guarantee the same probability of successful completion. If ϵ_{small} applied only to ten out of the one million primitives, then only thirty-four thousand samples would be required. In any of these cases, though, ϵ_{small} dominates (3.6) to an extent that the improvements in the theoretical guarantee are not dramatic.

3.4.3 Attraction Sequences and the MPP

Next we analyze probabilistic completeness of the MPP phase of sampling-based coverage path planning. Once a covering subset of robot configurations is selected for traversal, these goal configurations must be connected by a system of feasible paths. We formally define the MPP as follows:

Definition 3 (Multigoal Planning Problem)**.** *Let $G \subset \mathcal{Q}$ be a finite set of robot configurations which comprise the set of goals selected for traversal. Find a set of feasible paths in $\mathcal{Q}$ that joins all goals into a single connected component.*

If the goals are joined into a single connected component, then a feasible closed walk of all goals in G exists, giving a feasible solution to the coverage path planning problem. Both coverage path planning algorithms depicted in Figure 3-1 generate a feasible inspection tour that is compatible with Definition 3, although the redundant roadmap method, after solving the MPP in its first iteration, adds to the connected component in each subsequent iteration to shorten the inspection tour. We now define probabilistic completeness in the context of the MPP.

Definition 4 (Probabilistic Completeness of a MPP Algorithm)**.** *Let MPA be a proposed multigoal planning algorithm for the MPP. Let $G \subset \mathcal{Q}$ be the set of goals over which the MPP is defined. If, when a set of feasible paths in $\mathcal{Q}$ exists that joins all goals into a single connected component, the probability that such a set is found by MPA approaches one as the number of samples of $\mathcal{Q}$ drawn by MPA approaches infinity, then MPA is probabilistically complete.*

Proofs of completeness are straightforward for the MPP. For both the watchman route algorithm and the redundant roadmap algorithm, we utilize the notion of an *attraction sequence* [105]. To connect a pair of goals $\{q_a, q_b\} \in G$ with a feasible path, an attraction sequence is a sequence of sets $\mathcal{A}_{a,b} = \{A_0, A_1, ..., A_n\} \subseteq \mathcal{Q}$, where $A_0 = q_a$ and $A_n = q_b$, that bridge the gap between q_a and q_b. The defining property of an attraction sequence is the following: if an existing configuration q_{l-1} lies in A_{l-1}, and new configuration q_l is generated in A_l, then a PRM or RRT edge will

be constructed that connects q_{l-1} and q_l. In general it is desirable for an attraction sequence to have as few members A_l as possible, and so all A_l other than singletons A_0 and A_n should be as large in volume as possible.

We will use $\mathcal{A}_{MPP}$ to designate the set of all attraction sequences employed in solving an instance of the multigoal planning problem, where $|\mathcal{A}_{MPP}|$ is the total number of sets A_l in $\mathcal{A}_{MPP}$. A worst-case analysis of the MPP will depend on both $|\mathcal{A}_{MPP}|$ and the volume fraction $\epsilon = \min_{A_l \in \mathcal{A}_{MPP}} \mu(A_l)/\mu(\mathcal{Q}_{free})$, where $\mathcal{Q}_{free}$ is the obstacle-free portion of the configuration space. It remains desirable for the problem to be solved using as few A_l as possible, and for the A_l to be individually as large in volume as possible. We also note that in the sampling processes used to solve the MPP, the singleton sets in $\mathcal{A}_{MPP}$ representing goal configurations do not need to be populated with new samples, as they are already finalized as part of the inspection tour. As a result, these zero-volume singletons will not be considered in the computation of ϵ, and $|\mathcal{A}_{MPP}|$ is a conservative overestimate of the number of sets that must be populated with new samples.

3.4.4 Probabilistic Completeness of the MPP

The watchman route algorithm solves the MPP by constructing a PRM that joins all goals into a single connected component. An all-pairs shortest paths algorithm can be used to determine the costs of all goal-to-goal paths, and a TSP algorithm can find a minimum-cost traversal. Unlike the typical use of the PRM, in which goal-to-goal queries are presented one at a time, the MPP presents a larger set of goals upfront and requires that all of these goals are connected to the roadmap. This can be handled easily by initializing the PRM so it contains the set of goals G. To show probabilistic completeness in this application we rely on prior analysis of the PRM by Kavraki et al. [88].

Theorem 2 (Completeness and Convergence of the PRM-Based Solution of MPP)**.** *Constructing a PRM in $\mathcal{Q}$ until the set of goals G belongs to a single connected component is a probabilistically complete algorithm for the MPP. Additionally, the*

probability that a feasible solution has not been found after m samples is bounded such that

$$Pr[FAILURE] \leq \frac{|\mathcal{A}_{MPP}|}{e^{m\epsilon}} \quad . \tag{3.7}$$

Proof. The analysis of the PRM in [88] also applies to the use of a PRM for solution of the MPP. The key difference is that the standard PRM requires at least one sampled configuration to land in every non-goal set A_l in an attraction sequence $\mathcal{A}_{a,b}$, which is the attraction sequence for a single goal-to-goal path. The MPP requires at least one sampled configuration to land in every non-goal set A_l in the family of attraction sequences $\mathcal{A}_{MPP}$, and ϵ represents the smallest non-goal set in $\mathcal{A}_{MPP}$ rather than $\mathcal{A}_{a,b}$. This difference in the analyses changes the numerator in (3.7) and the factor ϵ in the denominator of (3.7). In all feasible instances of the MPP, these quantities are finite and nonzero, respectively, and so the result of [88] still applies. □

In the case of the redundant roadmap algorithm, a revised ordering of the goals in G is determined in each iteration of the MPP procedure, and the RRT is subsequently called to find feasible goal-to-goal paths for all goal pairings in this ordering. In the absolute worst case, RRTs are constructed for all $O(n^2)$ possible goal-to-goal queries. To analyze this solution of the MPP, we will build on the analysis of RRT probabilistic completeness from LaValle and Kuffner [105].

Theorem 3 (Completeness and Convergence of the RRT-Based Solution of MPP)**.** *Iteratively connecting the goals in G by a sequence of RRTs is a probabilistically complete algorithm for the MPP. Additionally, the probability that a feasible solution has not been found after m samples is bounded such that*

$$Pr[FAILURE] \leq \frac{e^{|\mathcal{A}_{MPP}|}}{e^{\,m\epsilon/2}} \quad . \tag{3.8}$$

Proof. The analysis of the RRT in [105] also applies to the use of RRTs for solution

of the MPP. The key difference is that the standard RRT requires $|\mathcal{A}_{a,b}|$ successes in a series of m Bernoulli trials, in which $\mathcal{A}_{a,b}$ is an attraction sequence for a single goal-to-goal path. The MPP requires $|\mathcal{A}_{MPP}|$ successes instead, and ϵ represents the smallest non-goal set in $\mathcal{A}_{MPP}$ rather than $\mathcal{A}_{a,b}$. This difference in the analyses changes the exponent in the numerator of (3.8) and the factor ϵ in the denominator of (3.8). In all feasible instances of the MPP, these quantities are finite and nonzero, respectively, and so the result of [105] still applies. □

We also note that in spite of the watchman route algorithm and redundant roadmap algorithm possessing probabilistic completeness with respect to both the CSP and MPP subproblems, there exists a family of coverage path planning problems for which a feasible 100%-coverage inspection tour may exist and both algorithms might fail. These are problems that contain a "prison cell" in $\mathcal{Q}_{free}$ from which a configuration can collect meaningful sensor information but there exists no feasible path from the cell to the rest of the configuration space. As long as prison cells are avoided, any feasible CSP solution will constitute a feasible MPP solution. A variety of measures can be taken to ensure this problem does not occur in practice; our specific solution is to ensure that all configurations sampled in the CSP can be connected via feasible path to a common origin in the configuration space.

3.5 Point Robot Test Case

We now compare the computational performance of the watchman route and redundant roadmap algorithms. Our aim is to examine the effect of the CSP solution method on the quality of the resulting inspection tour. To support this goal, we have modified the watchman route algorithm to allow the fairest-possible comparison. Once a full-coverage set of configurations is incrementally constructed using dual sampling, we apply the same high-performance combinatorial optimization procedure used in our implementation of the redundant roadmap algorithm. This modification, summarized in Figure 3-4, ensures that any performance gains due to the construction of a redundant roadmap are indeed due to our proposed CSP algorithm and not dif-

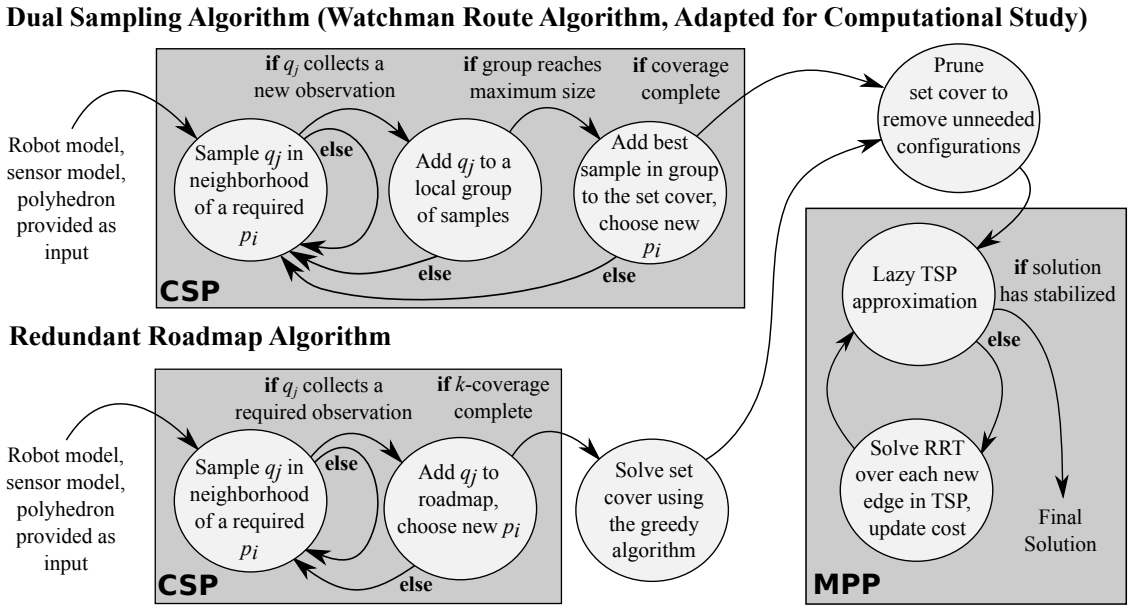


Figure 3-4: A stateflow diagram illustrating two algorithms for feasible sampling-based coverage path planning, highlighting the subroutines that solve the CSP and MPP subproblems. This diagram reflects the algorithms as implemented in software.

ferences in how a feasible tour is constructed from a set cover. For example, we have found in practice that pruning an approximate min-cardinality set cover eliminates a significant number of unneeded configurations and affords major improvements to both algorithms. As a result, we perform this step in both algorithm implementations, even if it is not part of the watchman route algorithm's original description. We will refer to the procedure at the top of Figure 3-4 as the *dual sampling algorithm*, since the dual sampling view planning strategy is the distinguishing feature of this adapted method.

First, we evaluate the performance of our inspection planning procedure on a point robot test case. This problem addresses algorithm performance as a function of the number of primitives, independent of collision and occlusion-checking. The unit cube is populated with a designated number of randomly sampled points, and the robot must plan a tour that observes them. Mimicking the HAUV inspection problem, the point robot has a four-dimensional state, comprised of three spatial coordinates, x, y, and z, and a yaw angle, θ. The sensor footprint is a cube, centered at the robot's location and designed to occupy about one percent of the workspace volume. The

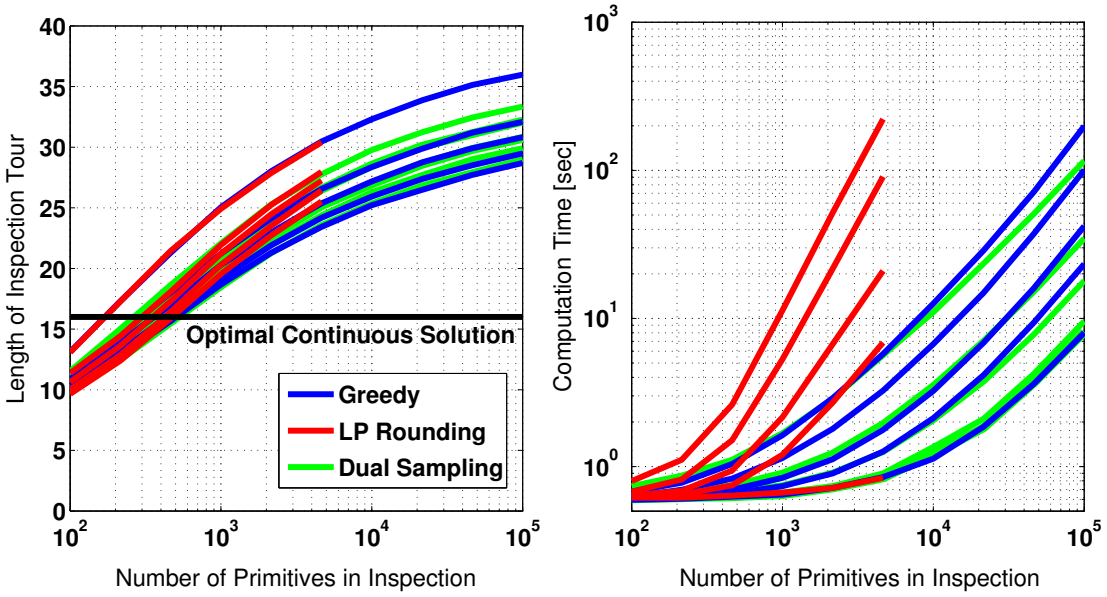


Figure 3-5: Inspection planning results from a point robot in an obstacle-free, unit-cube workspace, with a quarter-unit cube sensor. Inspection tour length and computation time are plotted as a function of the number of required primitives; each data point represents the mean over 100 problem instances. On left, LP rounding and greedy algorithm lines represent increasing roadmap redundancy [1,5,10,25,50] downward on the vertical axis. Dual sampling lines have increasing numbers of local samples, [10,25,100,250,1000] also moving downward. Data on right plot refer to computation times for the same trials, with redundancy and numbers of local samples increasing upward on the vertical axis. Due to prohibitively high computation time, larger quantities of primitives were not tested using the LP rounding algorithm, indicated by the end of the red lines.

sensor is a quarter unit in dimension to allow an integer number of sensor views to cover the workspace exactly. There are no obstacles in the point robot's workspace.

For several quantities of required primitives, ranging from one hundred to one hundred thousand, one hundred instances of the planning procedure were run for each of three solution methods: redundant roadmaps with a greedy set cover, redundant roadmaps with LP rounding, and the dual sampling method. For the redundant roadmap cases, five different redundancies were tested, ranging from one to fifty. For the dual sampling cases, five different numbers of local samples were tested, ranging from ten to one thousand. The chained Lin-Kernighan improvement heuristic was applied for 0.5 seconds after each computation of the Christofides TSP approximation. All trials were run on a Lenovo T400 laptop with a 2.53GHz Intel Centrino 2 processor

and 3GB of RAM. The planning procedure was implemented in C++ and run using several high-performance algorithm and data structure implementations; the sources of these are listed in Table B.1 in Appendix B.

To sample in the local neighborhood of a geometric primitive, a random configuration is constructed in a spherical coordinate system centered at the primitive. A range value is sampled uniformly at random between the minimum and maximum viewing range of the robot, and corresponding azimuth and elevation angles are randomly sampled as well. This places the robot at a position from which the primitive is in viewing range. Finally, the yaw angle is selected deterministically, such that a relative bearing of zero exists between the primitive and the robot. For a higher-dimensional vehicle state, a closed-form solution for angular orientation may not be available, and a Jacobian pseudoinverse method can be used to choose a robot orientation. This sampling procedure is used in both the point-robot and AUV test cases. It is followed by a series of geometric computations to catalog the full set of primitives that lie within the sensor footprint.

Figure 3-5 displays the results of this series of point-robot path planning computations. Increasing the redundancy of the coverage roadmap improved the quality of the greedy SCP solution and the LP rounding solution, but the relative quality of the LP rounding solution begins to worsen just short of a 1000-primitive inspection (for redundancies greater than one). In addition, the LP rounding algorithm, for large numbers of primitives, chooses much larger sets than the greedy algorithm. As a result, the pruning of sets became prohibitively expensive and LP set covers were not solved for large numbers of primitives. Due to lower-quality planned tours and exhaustive computation time required by the LP rounding algorithm, this optimization strategy is not pursued in the AUV inspection test case.

Increasing the number of local samples in a dual sampling scheme improved the quality of the solution, which was comparable with the results for redundant roadmaps solved by the greedy set cover algorithm. To provide further basis for comparison, the length of the optimal back-and-forth sweep path for covering the full continous workspace is plotted alongside the tour costs in Figure 3-5. For cases with a

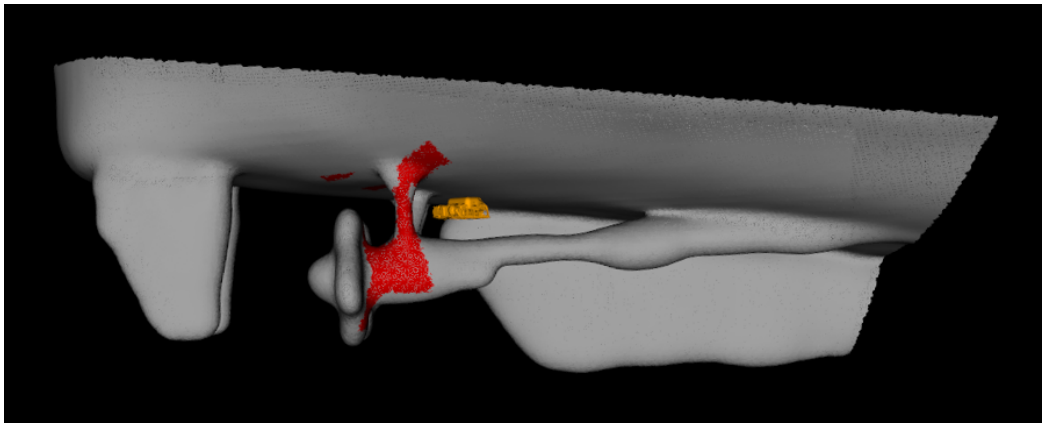


Figure 3-6: A polygonal mesh obtained from a safe-distance survey of the *USCGC Seneca* is depicted. The HAUV is illustrated in a configuration from which it observes a portion of a shaft and propeller strut. The red patch shows mesh points imaged at a desired sensor range between one and three meters, as the sonar sweeps through 180 degrees pitch. The ship mesh contains 131,657 points and 262,173 triangular faces. Each propeller is approximately 2.5 meters in diameter.

small number of primitives, in which the computational outcomes are shorter-than-optimal in length, the sensor did not have to cover the entire workspace volume in the sampling-based test cases. For cases with larger numbers of primitives, the discrete coverage problem is a better approximator for covering the full continuous workspace volume. All algorithms are sub-optimal by a growing margin in the number of discrete primitives, due to the use of random sampling and heuristics for the set cover and TSP.

3.6 AUV Inspection Test Case

Our planning procedure is next applied to a real-world problem, the inspection of the stern of a ship by the HAUV. Inspections are planned for the *USCGC Seneca*, an 82-meter Coast Guard Cutter, and the *SS Curtiss*, a 183-meter aviation logistics support ship. The complex structures are large; the *Seneca* has two shafts with propellers that are 2.5 meters in diameter, while the *Curtiss* has a single propeller seven meters in diameter and a shaft that is 1.5 meters in diameter.

We first surveyed the ships with back-and-forth rectangular sweep patterns at

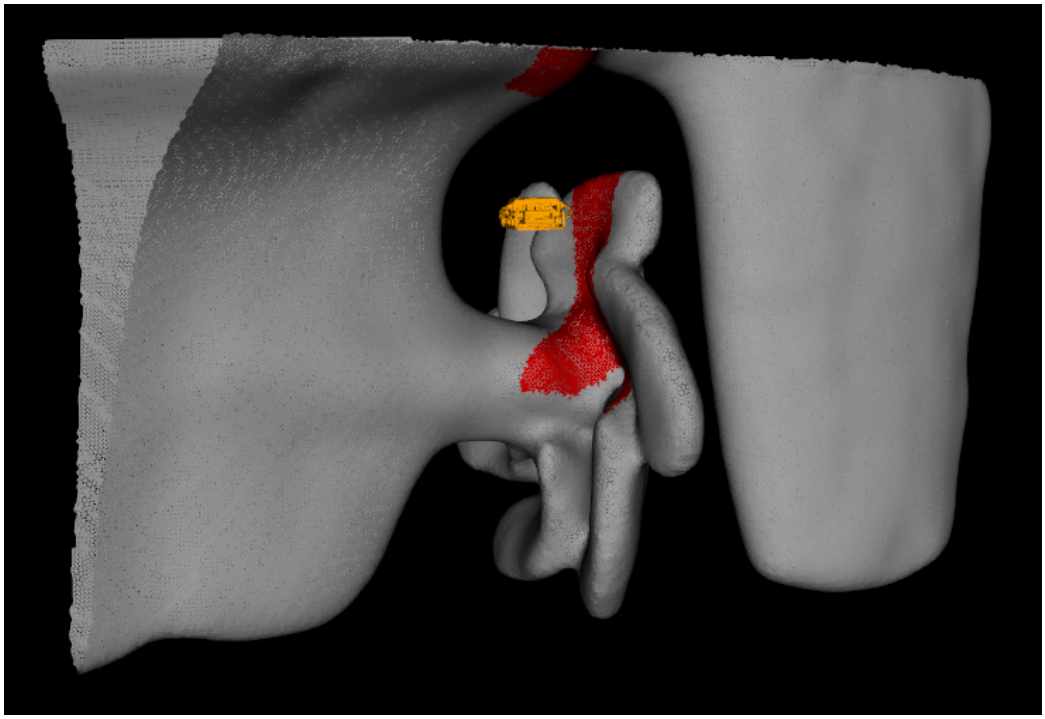


Figure 3-7: A polygonal mesh obtained from a safe-distance survey of the *SS Curtiss* is depicted. The HAUV is illustrated in a configuration from which it observes a portion of the ship's propeller. The red patch shows mesh points imaged at a desired sensor range between one and three meters, as the sonar sweeps through 180 degrees pitch. The ship mesh contains 107,712 points and 214,419 triangular faces. The propeller is approximately 7 meters in diameter.

safe distances of around eight meters. These preliminary surveys, although they did not achieve 100% coverage of all structures, were intended to build a polygonal mesh model of each ship's stern suitable for planning a detailed inspection. For this, the Poisson reconstruction algorithm [91], which is typically applied to laser point clouds, was used to build watertight 3D meshes from acoustic range data, pictured in Figures 3-6 and 3-7. The mesh generation process is described in full detail in Chapter 6. Each mesh shown has been discretized such that no triangle edge is longer than 0.1 meters, a resolution sufficient to identify a mine on the hull if all vertices are observed. Also in Figures 3-6 and 3-7, the sensor footprint represents the sonar field of view when the sonar nods up and down through a full 180-degree range of rotation. Although the sonar can only produce a single range scan at a time, we assume that in this planned inspection, the vehicle, at each configuration, will nod the sonar over its full

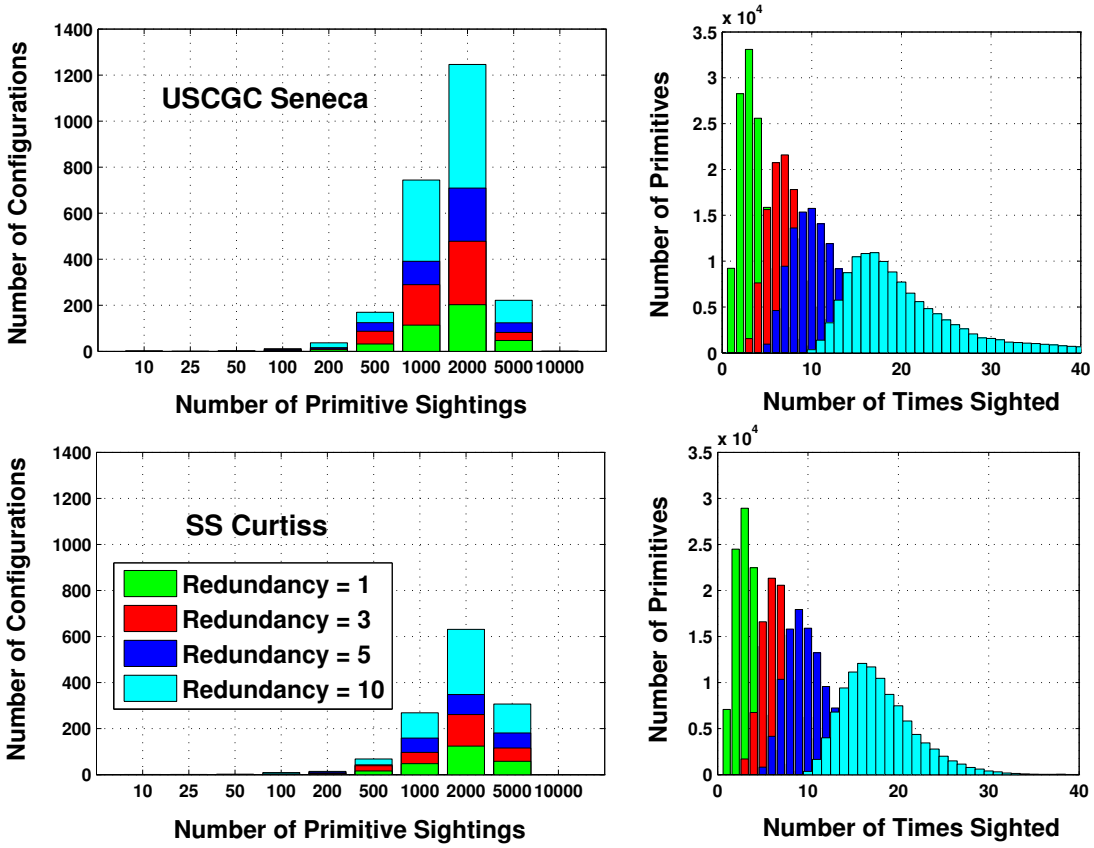


Figure 3-8: Histograms display the coverage topology of typical roadmaps for the *USCGC Seneca* and *SS Curtiss* ship hull inspection tasks, sensing with a 1-3 meter field of view. The quantities of geometric primitives observed by roadmap configurations are illustrated at left, and the quantities of shared sightings of geometric primitives are illustrated at right.

range of angular motion to obtain a larger field of view. Paths for the vehicle will be planned, as before, in x, y, z, and yaw angle θ.

Inspection tours planned using the redundant roadmap algorithm and dual sampling algorithm were computed using a Dell T3500 desktop with a 3.20GHz Intel Xeon processor and 24GB of RAM. Because a ship mesh comprises a large, non-convex obstacle, the inspection planning procedures described in Sections 3.2 and 3.3 were used in their entirety. This includes collision-checking of sampled configurations, use of the bi-directional RRT to perform lazy inquiries of point-to-point paths, and ray shooting to check whether primitives lying in the geometric sensor footprint are obscured from view by occlusions. The high-performance codes used for path planning, ray shooting,

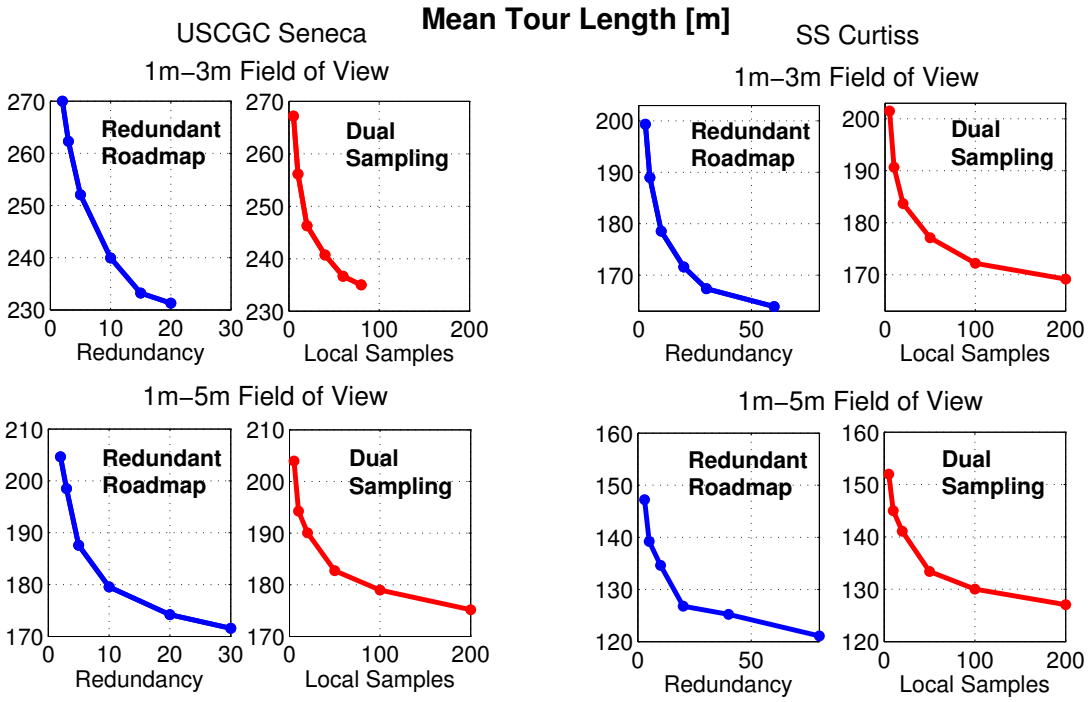


Figure 3-9: The mean tour length over fifty computed inspection tours is plotted for both the redundant roadmap and dual sampling strategies, over both ship models and both sensor range settings. The x-axis parameter of each plotted point was selected for comparision of algorithm computational performance (featured in Figures 3-11 and 3-12).

and geometric data structures are detailed in Table B.1 of Appendix B; these were integrated into the C++ software implementation used for the point robot test case. All combinatorial optimization tools were applied using the same settings as in the point robot case.

The coverage topology of a typical redundant roadmap, for both ship models over a number of different redundancy settings, is depicted in Figure 3-8. It is clear that increased redundancy both increases the size of the roadmap and increases the mean and variance of the number of times a primitive is sighted. For the purpose of comparison with dual sampling, inspection tours were planned using two DIDSON range settings, one that spans from one to three meters, and another that spans from one to five meters. For each sensing range and each ship model, six parameterizations were selected for comparison between algorithms, and fifty path-planning trials were run for each parameterization. We will compare the computational overhead required

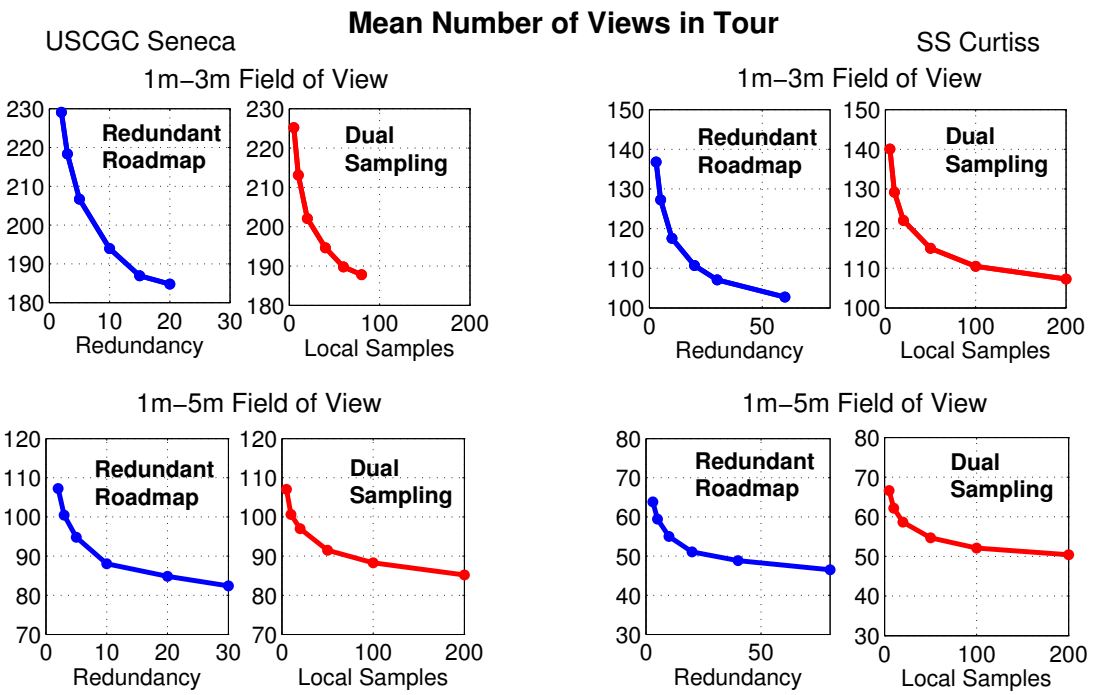


Figure 3-10: The mean number of planned views over fifty computed inspection tours is plotted for both the redundant roadmap and dual sampling strategies, over both ship models and both sensor range settings. The x-axis parameter of each plotted point was selected for comparison of algorithm computational performance (featured in Figures 3-11 and 3-12).

by each algorithm to produce solutions of similar quality. The mean lengths of the planned inspection tours are plotted in Figure 3-9, and the mean quantities of planned views are plotted in Figure 3-10, as functions of the algorithm tuning parameters.

As illustrated in Figures 3-9 and 3-10, increased roadmap redundancy leads to an improvement in both tour length and the number of planned views in the tour, but the size of the improvement evidently diminishes as the redundancy increases. This is also true for the dual sampling algorithm; raising the number of local samples enhances tour quality, with diminishing improvements as the parameter is increased. In Figures 3-11 and 3-12, we compare the mean computation time and the mean number of ray shooting calls required by the algorithms to produce solutions of a specific quality. It is evident, for both the mean length of a tour and the mean number of planned views in a tour, that the redundant roadmap algorithm requires less time and fewer ray shooting calls in the limit to produce solutions of quality on par with the dual

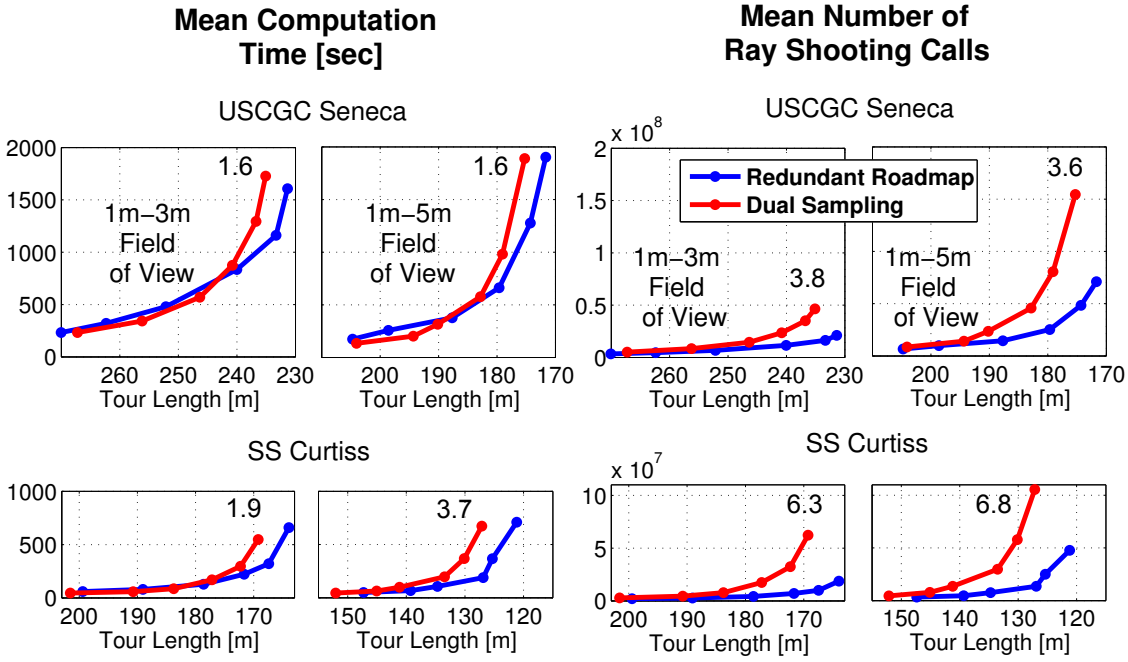


Figure 3-11: The mean computation time, at left, and ray shooting calls, at right, required to plan inspection tours of the mean lengths depicted on the x-axis. Each point represents fifty inspection tours, which were computed for both the redundant roadmap and dual sampling strategies, over both ship models and both sensor range settings. Included in each plot is the approximate ratio of dual sampling performance to redundant roadmap performance at the final dual sampling data point. The data displayed in these plots are the same data depicted in Figures 3-9 and 3-10.

sampling algorithm.

The mean number of ray shooting calls represents the geometric complexity of a planning problem. In planning a stern inspection, most sampled robot configurations require hundreds of ray shooting calls to ensure that primitives lying in the sensor's geometric field of view are not blocked by obstacles. For many of the problem instances examined, tens of millions of ray shooting calls were performed in total. It is evident in Figures 3-11 and 3-12 that the redundant roadmap algorithm exceeds the dual sampling algorithm in efficiency, in some instances by a factor of five or more, when judged by the mean number of ray shooting calls. We attribute this advantage to the fact that in each local sampling phase, the dual sampling algorithm draws and catalogs many benefical samples, but discards all but one configuration at the end

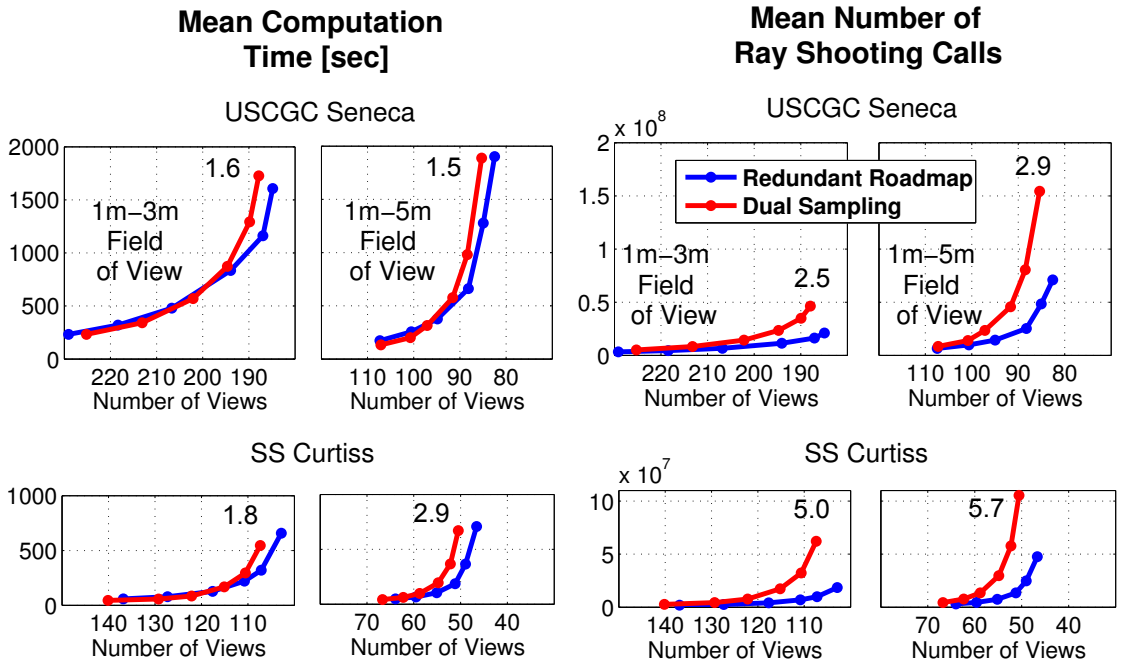


Figure 3-12: The mean computation time, at left, and ray shooting calls, at right, required to plan inspection tours with the mean number of planned views depicted on the x-axis. Each point represents fifty inspection tours, which were computed for both the redundant roadmap and dual sampling strategies, over both ship models and both sensor range settings. Included in each plot is the approximate ratio of dual sampling performance to redundant roadmap performance at the final dual sampling data point. The data displayed in these plots are the same data depicted in Figures 3-9 and 3-10.

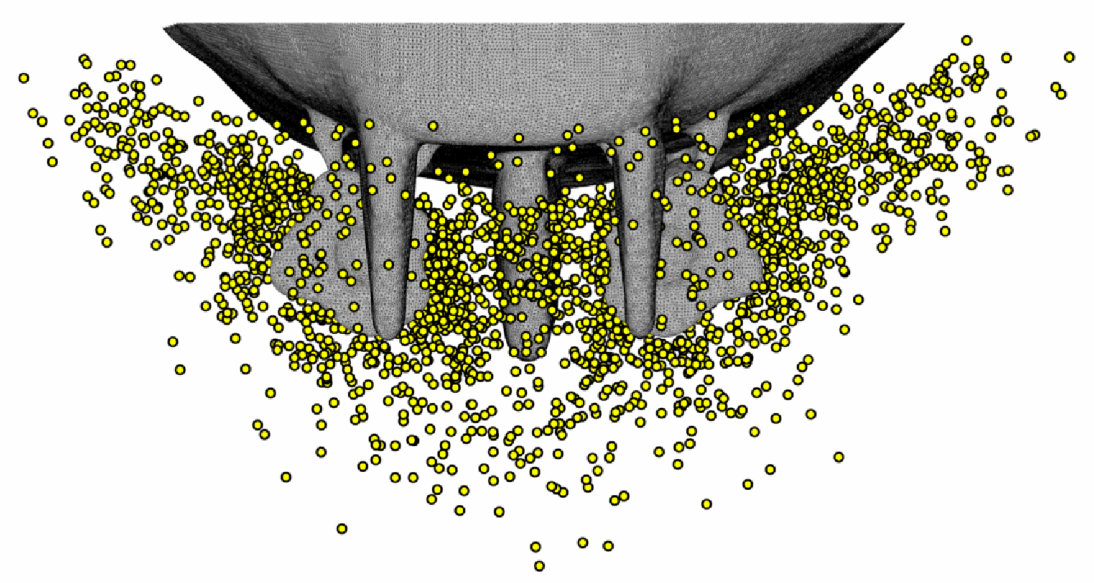
of the sampling phase. By storing all configurations that observe required primitives and delaying selection of the final set of views, the redundant roadmap algorithm requires less overall geometric computation. Representative coverage roadmaps and planned inspection tours, of roadmap redundancy ten, are illustrated for both ship models in Figures 3-13 and 3-14.

3.7 Summary

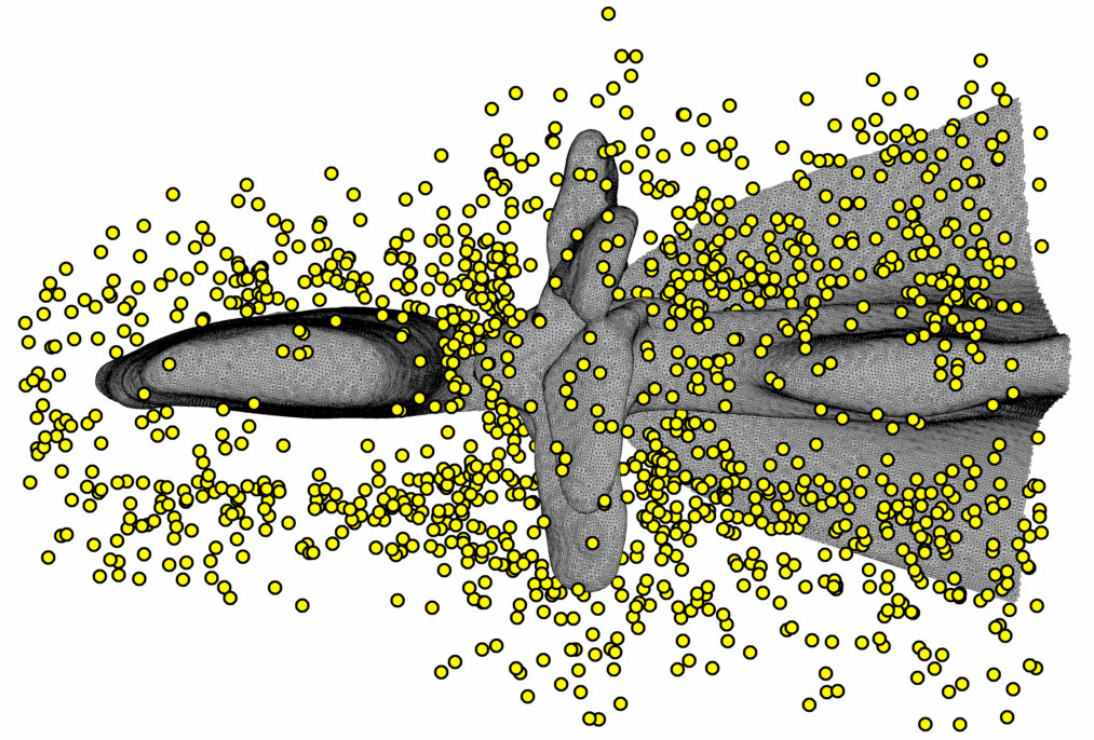
In this chapter we presented an algorithm that plans feasible robot inspection paths giving 100% sensor coverage of required geometric primitives. A key development is redundancy in a covering roadmap, which endows batch view planning with a

tunable parameter for improving solution quality in exchange for additional sampling. We have also developed a high-performance solution procedure for coverage planning over 3D structures comprised of several hundred thousand primitives (using highly developed data structures and algorithm implementations where possible). After a full-coverage roadmap is constructed, we sequentially apply practical set cover and traveling salesman algorithms, with lazy, point-to-point sampling-based planning. We have identified that the redundant roadmap method, in comparison to a dual sampling procedure, yields a consistent computational advantage in a large-scale, real-world coverage problem.

We have also developed a framework for the analysis of a sampling-based coverage path planning algorithm. We have used it to show that the sampling-based subroutines of the redundant roadmap algorithm are probabilistically complete, with appealing convergence bounded by sharply decreasing exponential functions. The analysis and development of this coverage planning algorithm form a foundation for further improvements, including the smoothing, shortening, and regularizing of planned inspection routes.

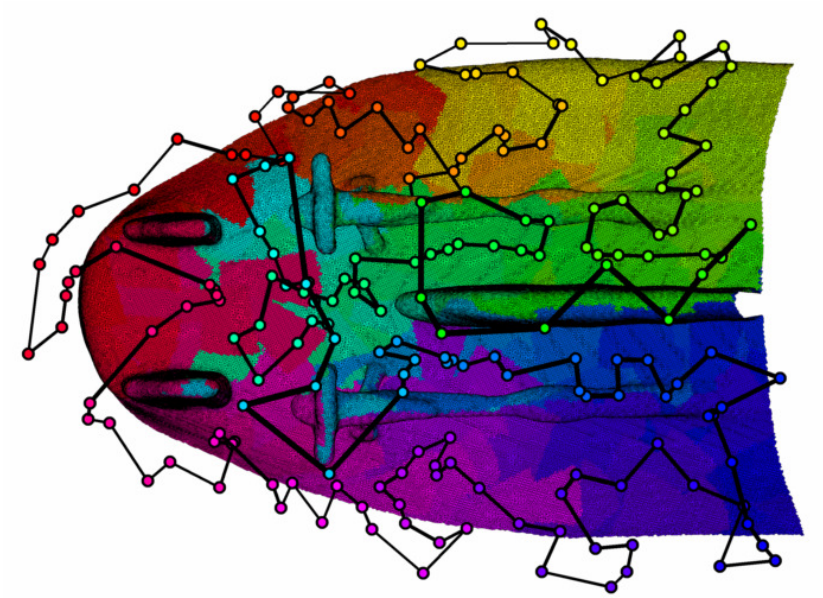


(a) *USCGC Seneca* redundancy-ten roadmap containing 2433 configurations.

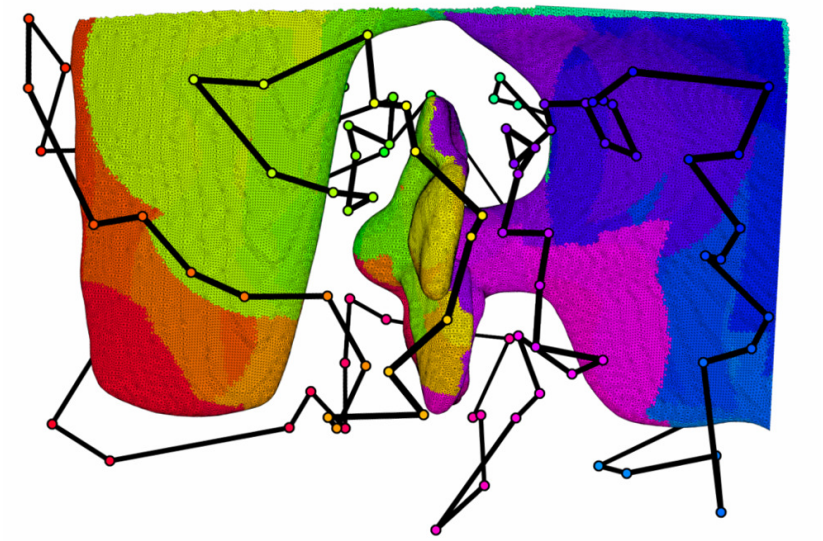


(b) *SS Curtiss* redundancy-ten roadmap containing 1300 configurations.

Figure 3-13: Redundancy-ten roadmaps constructed for full coverage of the *USCGC Seneca* and *SS Curtiss* at 1-3 meter viewing range. The coverage topology of these specific roadmaps is given in Figure 3-8.



(a) *USCGC Seneca* inspection tour is 244 meters long and contains 193 configurations.



(b) *SS Curtiss* inspection tour is 179 meters long and contains 118 configurations.

Figure 3-14: Full-coverage inspection tours planned using the roadmaps depicted in Figure 3-13. Each plotted point represents a position and orientation of the HAUV at which required information is collected. Robot configurations along each tour are color-coded and correspond to the colored patches of sensor information projected onto each ship model. The changes in color occur gradually and folllow the sequence of the inspection tour. The thickness of each line segment along the path corresponds to the relative distance of that segment from the viewer's perspective.

Chapter 4

Sampling-Based Improvement Procedure

4.1 Introduction

The algorithms presented in Chapter 3 plan *feasible* robot inspection routes; these routes avoid collision and obtain full sensor coverage of required structures. Although optimization procedures are used to reduce the duration of these feasible tours, they are built from randomly-sampled view configurations and are sub-optimal. A higher-quality solution is desirable both to permit a shorter-duration inspection and to improve the ease of implementation on a field robotic system. Unfortunately, even in simple cases the coverage path planning problem is equivalent to NP-hard variants of the watchman route problem [33],[165], so we do not seek a globally optimal solution. Instead, we pursue a compromise, and aim to develop a high-quality heuristic that offers smoother, shorter inspection paths than the algorithms of the previous chapter.

In this chapter we propose an iterative improvement procedure which, given a feasible, full-coverage inspection tour as input, gradually shortens the tour and reduces the number of view configurations, making gradual progress toward a locally optimal solution. A method of this type has not yet been applied to coverage path planning, but successful improvement algorithms have been developed for standard point-to-point path planning. The PRM* and RRT* algorithms, which add new sam-

ples to their roadmaps long after a feasible solution is obtained, iteratively augment sub-optimal paths to achieve optimality in the limit, a property known as *asymptotic optimality* [86].

We require the shortening of paths that are not only collision-free, but contain view configurations that collectively satisfy thousands of coverage constraints. This problem is addressed by repeatedly solving a sub-problem that is local in scope: an individual view configuration is replaced by a new view that satifies all unique coverage constraints and is closer to the two neighboring views in the inspection tour. Using the RRT* algorithm as a tool in this procedure, we show that this improvement algorithm is asymptotically optimal, with respect to this local sub-problem.

Revisiting the application of autonomous in-water ship hull inspection, we also propose a heuristic speed-up of the improvement algorithm for problems in which the set of planned sensor views is numerous, and the views lie in close proximity to one another. This speed-up, ideally suited to the inspection of a large contiguous structure by a robot with a small field of view, allows our local problem to be solved quickly, and in many instances to optimality.

Over an ensemble of computational trials, we use the redundant roadmap algorithm to design an initial, feasible inspection tour. We then apply the proposed sampling-based improvement procedure, which achieves significant reductions in tour length with reasonable computational effort. A description and analysis of the proposed algorithm is given in Section 4.2, brief computational results are given for the obstacle-free point robot test case in Section 4.3, and more extensive computational results are given for ship hull inspection by the HAUV in Section 4.4.

4.2 A Sampling-Based Improvement Procedure

In our presentation of the sampling-based improvement procedure, we rely on the same set system taxonomy used in Chapter 3. We refer to the geometric primitives of the structure under inspection as $p_i \in P$, and the sampled robot view configurations are denoted $q_j \in \mathcal{Q}$. $S_i \in \mathcal{S}$ refers to the set of all feasible configurations in $\mathcal{Q}$ that

Algorithm 3 $W'_G = ShortenInspection(G, W_G, P, Obstacles)$

$W'_G \leftarrow W_G$;
while $TimeRemaining > 0$ **do**
  $q_j \leftarrow ChooseGoal(G)$;
  $P_j \leftarrow UniquelyObservedPrimitives(q_j, G)$;
  $(q'_j, W'_{q_{j-1},\, q_{j+1}}) \leftarrow RRT^*_{||}(q_{j-1}, q_{j+1}, P_j, Obstacles)$;
  if $Cost(W'_{q_{j-1},\, q_{j+1}}) < Cost(W_{q_{j-1},\, q_{j+1}})$ **then**
    $W'_G \leftarrow W'_G \setminus W_{q_{j-1},\, q_{j+1}}$;
    $W'_G \leftarrow W'_G \cup W'_{q_{j-1},\, q_{j+1}}$;
    $G \leftarrow G \setminus q_j$;
    $G \leftarrow G \cup q'_j$;
    $UpdateCoverageTopology(G)$;
  end if
end while
return W'_G

map to sightings of the primitive $p_i \in P$.

4.2.1 An Asymptotically Optimal Subroutine

We assume that a feasible inspection tour is provided as input to the improvement procedure. The inspection is described by the closed walk W_G of the set of goals G; G contains view configurations only. W_G contains the precise sequence of nodes and edges that are traversed in the inspection, which begins and ends at the same goal. W_G may include intermediate nodes that obtain no sensor information, but are required to manueuver safely around obstacles. The improvement procedure is summarized in Algorithm 3. As time for improvement allows, the algorithm iteratively selects a goal configuration $q_j \in G$ in a round-robin fashion and tries to find a lower-cost configuration q'_j that observes all primitives in P which are uniquely observed by q_j. This is achieved by the subroutine $\text{RRT}^*_{||}$, an implementation of the RRT^* algorithm [86] in which two problems are solved in parallel: an optimal collision-free path from q_{j-1} to q'_j, and an optimal collision-free path from q'_j to q_{j+1}. Solving these problems in parallel gives $W_{q_{j-1},\, q_{j+1}}$, the portion of the walk W_G that travels between goals q_{j-1} and q_{j+1} and includes the intermediate goal q'_j. We term this subproblem the *local coverage planning problem.*

Definition 5 (Local Coverage Planning Problem)**.** *Let $W_{q_{j-1},\, q_{j+1}}$ be a feasible path on the inspection tour W_G in which a robot travels from goal configuration q_{j-1} to goal configuration q_j to goal configuration q_{j+1}. Let $\mathcal{S}_{i \in q_j}$ be the intersection of all ranges $S_i \in \mathcal{S}$ corresponding to the primitives $p_i \in P$ that are uniquely observed by goal configuration q_j. Find a replacement configuration q'_j that lies in $\mathcal{S}_{i \in q_j}$ and a feasible path $W'_{q_{j-1},\, q_{j+1}}$ such that the path is of minimum length over all possible choices of q'_j.*

The RRT* algorithm of Karaman and Frazzoli plays an important role in the solution of this problem. This algorithm contains the same set of tree nodes as a standard RRT; these are generated by sampling $\mathcal{Q}$ at random and "growing" a distance η in the direction of each sample q_{rand} from the nearest node in the tree, q_{near}. A new node, q_{new}, is defined at this location, and the standard RRT connects q_{new} and q_{near} if there are no interfering obstacles. RRT*, on the other hand, searches a local neighborhood of q_{new} and connects it to the tree node giving the minimum accumulated cost from the root of the tree. In addition, all nodes in this local neighborhood with higher accumulated costs are re-routed through q_{new} if this lowers their cost. This procedure yields a path from a start configuration in $\mathcal{Q}$ to a goal region in $\mathcal{Q}$ that approaches global optimality in the limit. The local neighborhood must be maintained at an appropriate size, which is allowed to shrink over time at a rate equivalent to the dispersion of the uniform random sequence. Because the set of tree nodes is propagated identically in RRT and RRT*, this algorithm retains the probabilistic completeness guarantees of the standard RRT algorithm. We now define the properties of probabilistic completeness and asymptotic optimality as they apply to the local coverage planning problem.

Definition 6 (Probabilistic Completeness of a Local Coverage Planning Algorithm)**.** *Let LCA be a proposed algorithm for the local coverage planning problem. If, when both a feasible path $W'_{q_{j-1},q_{j+1}}$ exists such that $\mu(\mathcal{S}_{i \in q_j})/\mu(\mathcal{Q}_{free}) \geq \delta > 0$ and there is non-degenerate clearance from obstacles along the full length of the path, the probability that such a path is found by LCA approaches one as the number of samples drawn*

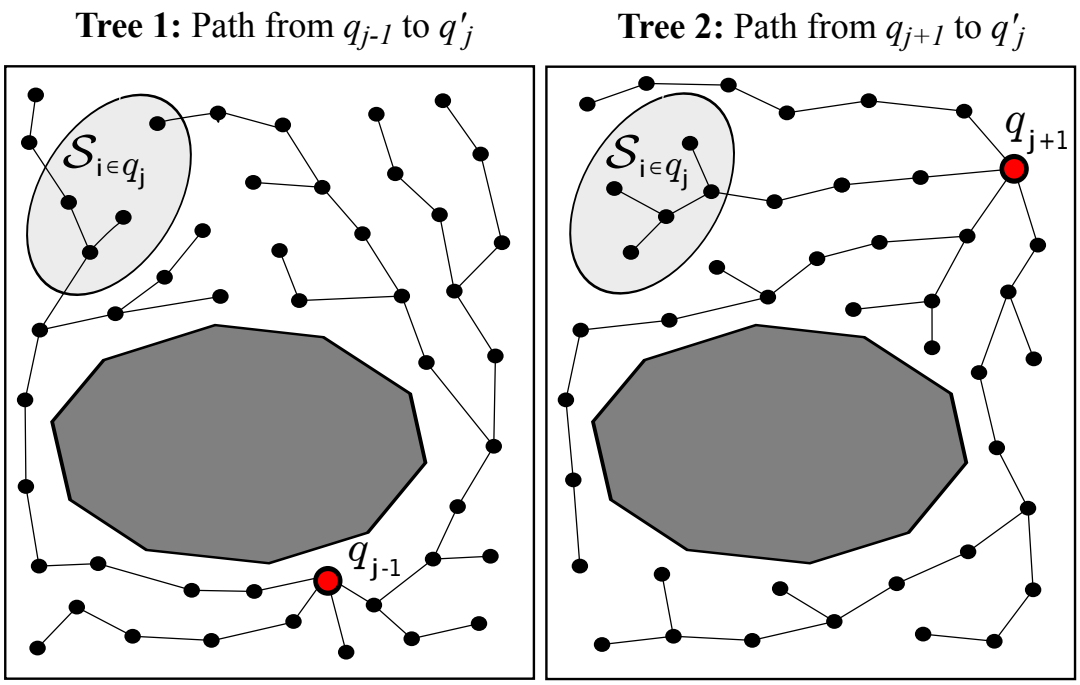


Figure 4-1: An illustration of the $\text{RRT}^*_{||}$ subroutine of the sampling-based improvement procedure.

from $\mathcal{Q}$ approaches infinity, then LCA is probabilistically complete.

Definition 7 (Asymptotic Optimality of a Local Coverage Planning Algorithm)**.** *Let LCA be a probabilistically complete algorithm for the local coverage planning problem. If, when an optimal path $W^*_{q_{j-1},\, q_{j+1}}$ exists with non-degenerate clearance from obstacles along the full length of the path, the length of the shortest path obtained by LCA approaches the optimal length $|W^*_{q_{j-1},\, q_{j+1}}|$ as the number of samples drawn from $\mathcal{Q}$ approaches infinity, then LCA is an asymptotically optimal algorithm.*

We intend to show that the $\text{RRT}^*_{||}$ subroutine, which builds a pair of RRT^* trees concurrently, possesses both probabilistic completeness and asymptotic optimality. Figure 4-1 shows q_{j-1}, q_{j+1}, and $\mathcal{S}_{i \in q_j}$ in the context of $\text{RRT}^*_{||}$. Tree 1 is rooted at q_{j-1} and Tree 2 is rooted at q_{j+1}. Both of these trees share $\mathcal{S}_{i \in q_j}$ as a goal region. The two trees, unlike two completely separate instances of RRT^*, share the same sampling process. Every randomly sampled configuration must be introduced into the tree rooted at q_{j-1} and the tree rooted at q_{j+1}. When this occurs, the nearest

Algorithm 4 $(q'_j, W'_{q_{j-1},\, q_{j+1}}) \leftarrow RRT^*_{||}(q_{j-1}, q_{j+1}, P_j, Obstacles)$

$Tree_1 \leftarrow InitializeTree(q_{j-1})$; $Tree_2 \leftarrow InitializeTree(q_{j+1})$;
$GoalReached \leftarrow False$;
$GoalCandidates \leftarrow \emptyset$;
while $(TimeRemainingRRT^* > 0) \vee (GoalReached = False)$ **do**
  $q_{rand} \leftarrow DrawSample()$;
  for $i \in \{1, 2\}$ **do**
    $q_{near_i} \leftarrow Nearest(Tree_i, q_{rand})$;
    $q_{new_i} \leftarrow Extend(q_{near_i}, q_{rand})$;
    $IsFeasible_i \leftarrow ObstacleFree(q_{near_i}, q_{new_i}, Obstacles)$;
    if $IsFeasible_i$ **then**
      $Tree_i \leftarrow AddToRRT^*(Tree_i, q_{new_i})$;
    end if
  end for
  if $(IsFeasible_1) \wedge (IsFeasible_2) \wedge (q_{new_1} = q_{new_2})$ **then**
    if $ConstraintsSatisfied(q_{new_1}, P_j)$ **then**
      $GoalReached \leftarrow True$;
      $GoalCandidates \leftarrow GoalCandidates \cup q_{new_1}$
    end if
  end if
end while
$(q'_j, W'_{q_{j-1},\, q_{j+1}}) \leftarrow ShortestPathsToGoal(Tree_1, Tree_2, GoalCandidates)$
return $(q'_j, W'_{q_{j-1},\, q_{j+1}})$

node in each tree will "grow" toward the sample, or the tree will directly connect to the sample if this connection is collision-free and spans a distance less than the designated growth distance η. The $\text{RRT}^*_{||}$ algorithm is summarized in Algorithm 4.

We now state the probabilistic completeness and optimality properties of $\text{RRT}^*_{||}$. Our first statement relies once again on the concept of an attraction sequence. With respect to the local sub-problem solved by $\text{RRT}^*_{||}$, we consider the connection of $\{q_{j-1}, S_{i \in q_j}\} \in G$ using the attraction sequence $\mathcal{A}_{j-1,\, j} = \{A_0, A_1, ..., A_n\} \subseteq \mathcal{Q}$, where $A_0 = q_{j-1}$ and $A_n = S_{i \in q_j}$. We must also achieve the connection of $\{q_{j+1}, S_{i \in q_j}\} \in G$ using the attraction sequence $\mathcal{B}_{j+1,\, j} = \{B_0, B_1, ..., B_p\} \subseteq \mathcal{Q}$, where $B_0 = q_{j+1}$ and $B_p = S_{i \in q_j}$. The sampling process must generate new tree nodes in $n + p - 1$ total sets for a feasible solution to be obtained. This excludes the singleton sets A_0 and B_0, which are embedded in the solution from the beginning, and counts the shared goal region $S_{i \in q_j}$ only once. Using these preliminaries, we can now bound the failure

probability of RRT$^*_{||}$.

Theorem 4 (Convergence Rate of RRT$^*_{||}$)**.** *If a feasible path $W'_{q_{j-1},q_{j+1}}$ exists for a local coverage planning problem such that $\mu(\mathcal{S}_{i\in q_j})/\mu(\mathcal{Q}_{free}) \geq \delta > 0$, and there is non-degenerate clearance from obstacles along the full length of the path, the probability that such a path has not been found by RRT$^*_{||}$ after m samples is bounded such that*

$$Pr[FAILURE] \leq \frac{e^{n+p-1}}{e^{\,m\epsilon/2}} \quad , \tag{4.1}$$

where n and p are the lengths of the attraction sequences $\mathcal{A}_{j-1,\,j}$ and $\mathcal{B}_{j+1,\,j}$ needed to reach $S_{i\in q_j}$ from q_{j-1} and q_{j+1}, and ϵ is the volume fraction of the smallest set in $\mathcal{A}_{j-1,\,j} \cup \mathcal{B}_{j+1,\,j}$, excluding tree roots q_{j-1} and q_{j+1}.

Proof. The exponential bound on the convergence of failure probably for the RRT* algorithm has been established to be identical to that of the original RRT algorithm [86]. There is only a minor difference in (4.1) from the bound on the original RRT algorithm [105], which requires at least n successes in a series of m Bernoulli trials, in which $\mathcal{A}_{a,b} = \{A_0, A_1, ..., A_n\}$ is an attraction sequence for a single goal-to-goal path. RRT$^*_{||}$, which builds two concurrent trees that share a goal region, instead requires $n + p - 1$ successes, and ϵ represents the volume fraction of the smallest set in $\mathcal{A}_{j-1,\,j} \cup \mathcal{B}_{j+1,\,j}$ rather than $\mathcal{A}_{a,b}$. This difference in analyses changes the exponent in the numerator of (4.1) and the factor ϵ in the denominator of (4.1). These quantities are finite and nonzero, respectively, in the non-degenerate instances of the local coverage planning problem described in the statement of the theorem, and so the result of [105] still applies. □

By taking the limit of (4.1), we can deduce that RRT$^*_{||}$ is probabilistically complete. The use of attraction sequences, however, disguises the key challenge of RRT$^*_{||}$: generating tree nodes in $\mathcal{S}_{i\in q_j}$ that are common to both Tree 1 and Tree 2. Because attraction sequences are difficult to compute in practice, we offer a supplemental theorem and proof on probabilistic completeness, stated in simpler terms, that provides insight into the workings of the RRT$^*_{||}$ algorithm.

Theorem 5 (Probabilistic Completeness of RRT$^*_{||}$)**.** *RRT$^*_{||}$ is a probabilistically complete algorithm for the local coverage planning problem.*

Proof. Due to the probabilistic completeness of RRT*, we know that Tree 1 and Tree 2 will reach their respective goal regions in probability. We must also show, however, that they will have some identical nodes in their goal regions so that a feasible path $W'_{q_{j-1},\, q_{j+1}}$ will be produced. Due to the condition on $\mathcal{S}_{i \in q_j}$ in Definition 6, there is a nonzero probability that random samples will land in $\mathcal{S}_{i \in q_j}$. The samples that land in $\mathcal{S}_{i \in q_j}$ will be added as nodes to both Tree 1 and Tree 2 if they land within a distance η of existing nodes in both trees. We know this does occur because:

- The samples in an RRT* tree converge to the uniform distribution over $\mathcal{Q}_{free}$ [98, 86]
- The dispersion of the uniform distribution, which varies as $O((log(m)/m)^{1/d})$ in the number of samples of a d-dimensional space [121], will eventually reach η as the number of samples increases

After enough samples are drawn, all new samples will lie within a distance η of multiple tree nodes, and samples landing in $\mathcal{S}_{i \in q_j}$ will be directly connected to both trees. □

This result is important because it demonstrates the key factors that will allow a feasible solution $W'_{q_{j-1},q_{j+1}}$ to be obtained in finite time: the ease with which Trees 1 and 2 reach $\mathcal{S}_{i \in q_j}$, and time required for the sampling sequence to achieve a dispersion of η. We now give the result on asymptotic optimality:

Theorem 6 (Asymptotic Optimality of RRT$^*_{||}$)**.** *RRT$^*_{||}$ is an asymptotically optimal algorithm for the local coverage planning problem.*

Proof. $\mathcal{S}_{i \in q_j}$ is the goal region of each tree in RRT$^*_{||}$, and in the limit, we will obtain the set of asymptotically optimal paths from q_{j-1} and q_{j+1} to the goal region, by the properties of RRT*. By choosing the node $q'_j \in \mathcal{S}_{i \in q_j}$ that minimizes the sum of distances to q_{j-1} in Tree 1 and q_{j+1} in Tree 2, we obtain the optimal path $W^*_{q_{j-1},q_{j+1}}$. □

Our improvement procedure is designed to extract the maximal benefit from recent results on asymptotically optimal sampling-based planning while avoiding non-trivial combinatorial optimization. If we added just one additional degree of freedom and tried to design $W_{q_{j-1},\, q_{j+2}}$ optimally, which requires hitting the two sets $\mathcal{S}_{i \in q_j}$ and $\mathcal{S}_{i \in q_{j+1}}$ and connecting them with an optimal path from q_{j-1} to q_{j+2}, we could not do so by building trees. The much denser PRM* would be required to find an optimal path between the infinite-set goal regions $\mathcal{S}_{i \in q_j}$ and $\mathcal{S}_{i \in q_{j+1}}$, and choosing optimal states q'_j and q'_{j+1} and the order in which to visit them would amount to solving the NP-hard generalized traveling salesman problem [60] over the PRM* roadmap.

4.2.2 Updating the Coverage Topology

An update of coverage topology occurs every time a new configuration is added to the inspection tour. As goal configurations q_j are replaced by new goals q'_j that shrink the length of a tour, the coverage topology among the goals changes and occasionally a goal in G becomes obsolete, contributing no unique sensor observations to the inspection. When this occurs, the obsolete goal is removed from the tour, and an attempt is made to connect q_{j-1} and q_{j+1} using a shorter path than the path through obsolete q_j. Sometimes, the two goals can be bridged by a single straight-line path, and other times intermediate nodes are needed, which are found using the RRT-Connect algorithm [98]. Occasionally, a path shorter than the route through q_j cannot be found, and q_j remains in the tour as an intermediate node, but is no longer a member of the goal set G.

4.2.3 Modifications for Autonomous Ship Hull Inspection

We now discuss the application of the improvement procedure to the problem of autonomous ship hull inspection. This is a unique challenge for coverage path planning in which the structure to be inspected is comprised of one large, contiguous piece, and the robot's sensor footprint is small relative to the size of the structure. In turn, the set of goals required for 100% coverage is numerous, and every goal will be in

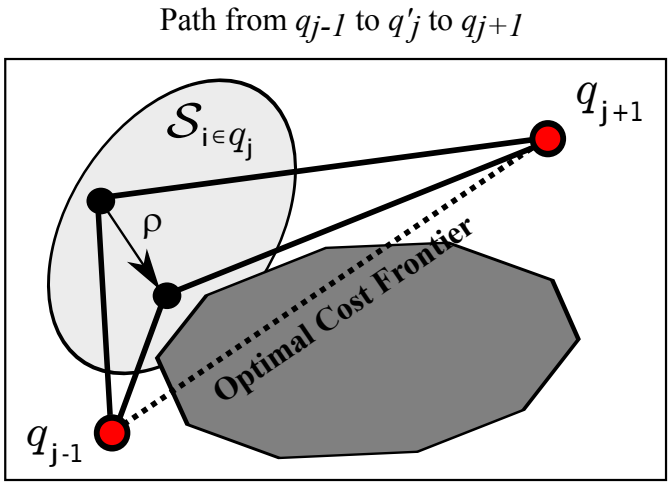


Figure 4-2: An illustration of the proposed heuristic speed-up, used in the sampling-based improvement procedure when a large, contiguous structure is inspected using a small field-of-view sensor.

close proximity to several others.

As a result, intermediate configurations are rarely needed between goal configurations in the inspection tour. This allows for a simplification of the improvement procedure, and the $\text{RRT}^*_{||}$ algorithm does not need to be used in its entirety. Instead, the algorithm will be used as a selection mechanism for goal-to-goal paths that have no intermediate nodes between q_{j-1}, q'_j, and q_{j+1}. Sampling will occur only in $\mathcal{S}_{i \in q_j}$ (specifically, in a larger region of $\mathcal{Q}$ known to contain $\mathcal{S}_{i \in q_j}$), and if a single graph edge cannot be built from each tree root to the sample q'_j, sampling of new q'_j continues until either this task is achieved or the maximum number of samples is reached and we move to a different goal in the inspection.

A benefit of this simplification is that we need not wait until samples land near the optimal location in $\mathcal{S}_{i \in q_j}$; we can project samples toward this location. Because we are looking for solutions in which the goal q'_j is connected directly to q_{j-1} and q_{j+1} by straight-line paths in $\mathcal{Q}_{free}$, we can move the individual samples from their random locations in $\mathcal{S}_{i \in q_j}$ to locations of improved cost, knowing that the path $W'_{q_{j-1},\, q_{j+1}}$ also improves in cost. We do this using a growth distance ρ, by which we incrementally push a sample toward the optimal-cost frontier (a straight-line path connecting q_{j-1} and q_{j+1}) until a collision is detected or we cross the boundary of $\mathcal{S}_{i \in q_j}$. Many fewer

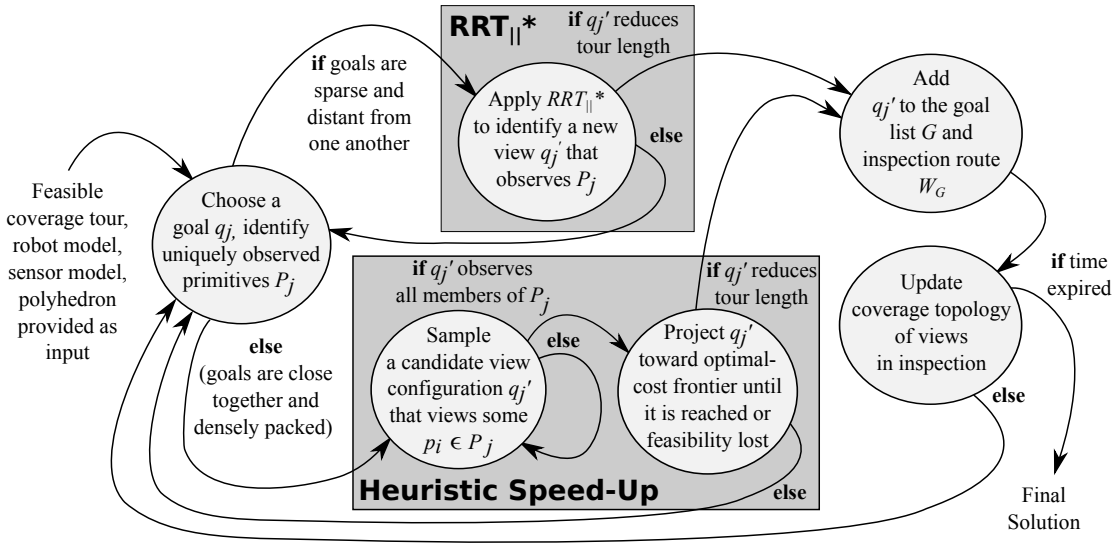


Figure 4-3: A stateflow diagram illustrating the iterative improvement procedure implemented in practice, in which either $\text{RRT}^*_{||}$ or a heuristic speed-up is used depending on the density of goal configurations.

samples need to be drawn to propagate new goals toward optimal-cost locations. This procedure is illustrated in Figure 4-2.

When the opposite situation occurs, and a structure to be inspected is comprised of separate pieces which may be far from one another, the benefits of $\text{RRT}^*_{||}$ can be fully realized and the algorithm will be needed in its entirety to connect goal configurations with high-quality feasible paths. A stateflow diagram illustrating the full improvement procedure, including the choice between $\text{RRT}^*_{||}$ and our simplified algorithm, is given in Figure 4-3. When the former is required, a number of heuristic improvements can be employed to reduce RRT* computation time, without simplification to the full extent of our straight-line-path algorithm. RRT* has been sped up in high-dimensional C-Space by biasing the sampling process to favor the local neighborhood near the working feasible path, and rejecting samples prior to collision checking if they have no potential to decrease the length of the path [6]. An algorithm developed for manipulation problems sparsifies the tree used to construct an initial feasible path, and also reduces the amount of collision-checking through a memoization process that relies on known collision-checking outcomes to deduce the outcomes of new

queries [125]. If the resources are available, computation time can also be sped up substantially by taking advantage of new techniques for parallelization of the RRT* algorithm [20]. Finally, recent work by Jaillet and Porta [83] extends the RRT* algorithm to planning on reduced-dimensional manifolds defined by kinematic and contact equality constraints. If an inspection robot is bound by contact constraints, then this algorithm may be useful, but it does not apply to the coverage constraints themselves, which are defined by inequalities in the robot's state variables.

4.3 Point Robot Test Case

We now re-examine the point robot coverage problem introduced in Chapter 3 to evaluate the proposed improvement procedure. The workspace is devoid of obstacles, and the coverage constraints alone will shape the resulting inspection route. Of the various parameterizations examined in Section 3.5, we explore the case of one hundred thousand primitives only, since this was the best approximator for requiring full coverage of the continuous internal volume of the unit cube workspace. One hour was alloted in each problem instance for the computation of a feasible tour, using a redundancy-ten roadmap, followed by implementation of the improvement procedure with the heuristic described in Section 4.2.3. One hundred problem instances, each with a different set of randomly sampled geometric primitives, were solved using a Dell T3500 desktop with a 3.20GHz Intel Xeon processor and 24GB of RAM. The improvement procedure was implemented in C++ and, like the redundant roadmap algorithm, relies on supporting software detailed in Table B.1 of Appendix B.

The initial feasible inspection tours were an average of 30.9 units in length, with an average of 219.3 view configurations. The smoothed tours were an average of 19.1 units in length, with an average of 183.3 view configurations. The mean length of the smoothed tours fell within twenty percent of the optimal tour length of 16 units, but the number of view configurations was nearly three times greater than the optimum number of 64 views. Many of the view configurations in a typical tour observed more than one thousand geometric primitives each. A view may only

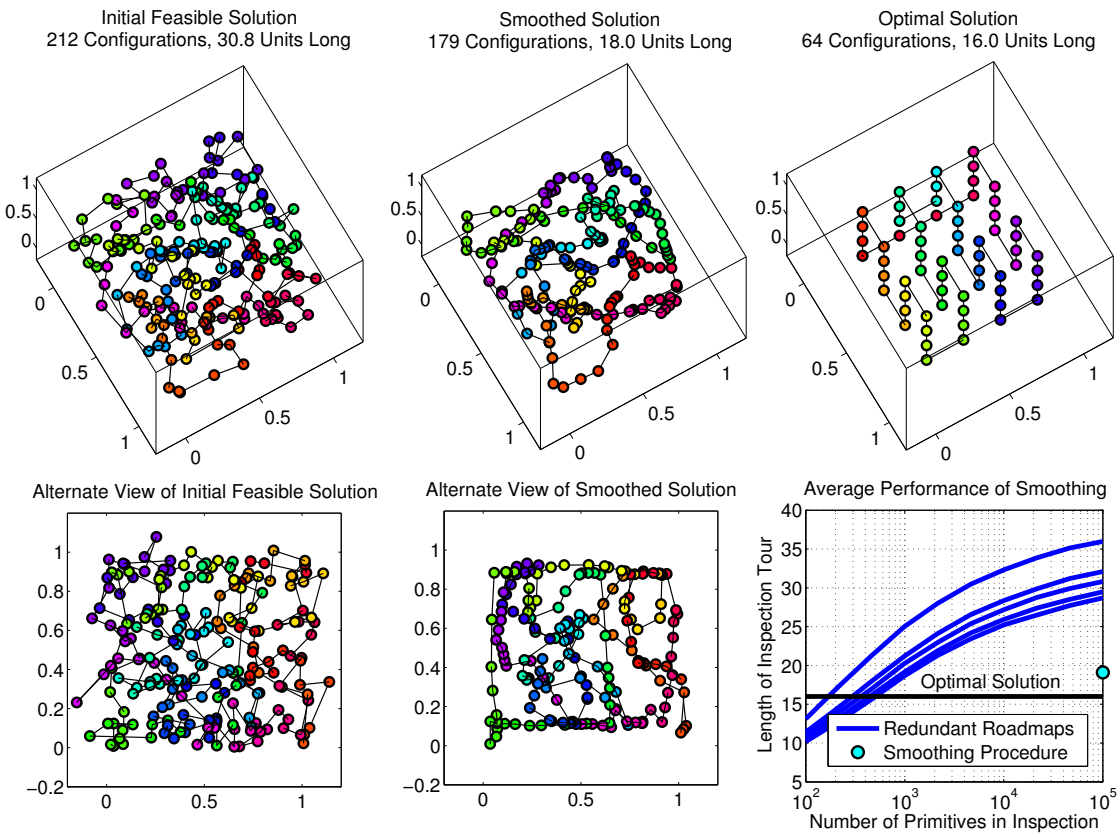


Figure 4-4: Full-coverage inspection tours for a point robot covering one hundred thousand randomly sampled primitives in the unit cube, with a cube-shaped sensor a quarter-unit in dimension. At left, an initial feasible tour constructed using a roadmap of redundancy ten. At center, the same tour after applying the improvement procedure over an hour of total computation time. At top right, an optimal tour for full coverage of the continuous volume of the unit cube workspace. The robot configurations along each tour are color-coded; the changes in color occur gradually and indicate the sequence of the inspection. At bottom right, a portion of Figure 3-5 is reproduced that contains average tour lengths for roadmaps of redundancies [1,5,10,25,50] downward on the vertical axis, computed using the greedy set cover approximation scheme. Added to the plot is the mean length when a tour from a redundancy-ten roadmap is smoothed over an hour of total computation time.

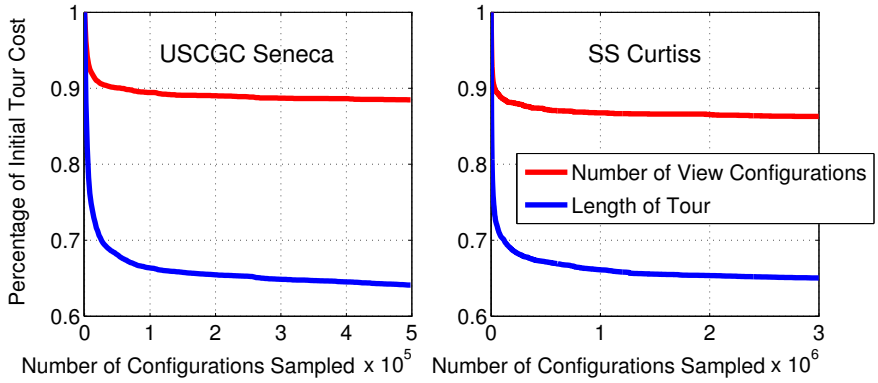
be pruned from the tour when none of these primitives are observed uniquely, and achieving this for a large number of views proves challenging with such a densely packed field of primitives. The primitives in this test case are denser, and of course, more uniformly distributed throughout the workspace, than most structure boundary coverage problems. A representative inspection tour from the computational results is depicted, along with the optimal solution, in Figure 4-4.

4.4 AUV Inspection Test Case

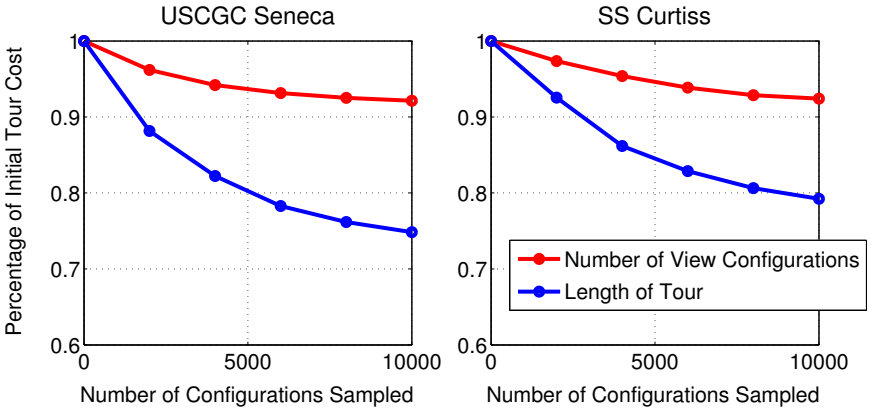
An ensemble of ship hull inspection tours were computed using the redundant roadmap algorithm and iteratively shortened using our proposed improvement procedure. We assume the DIDSON sonar is operated at a sensor viewing range of 1-3m, and the tours are once again planned for inspection of the *USCGC Seneca* and *SS Curtiss* using the mesh models introduced in Chapter 3. Initial, feasible routes for 100% coverage of the meshes are computed using roadmaps of redundancy ten, giving inspection tours whose one to two hundred nodes are chosen from a one to two thousand node instance of the set cover. Two hours were alloted in each problem instance for the computation of a feasible path and implementation of the improvement procedure. Twenty-five two-hour trials were run for each mesh using the same computer and software described in Section 4.3.

Computation of the initial feasible path required no more than nineteen minutes in any problem instance. This initial step was solved faster for the *Curtiss*, which required a maximum of four minutes in any problem instance. Figure 4-5 illustrates the average shortening of the inspection tours and the average reduction in the number of views as a function of the number of samples drawn by the improvement procedure. We show the total number of samples drawn, which includes samples found to be in collision with the mesh. The *Seneca* test cases each achieved at least a half-million samples in the alloted time, while the *Curtiss* cases achieved at least three million samples in the alotted time. The *Seneca* mesh contains more confined and occluded areas, especially between the shafts and the hull. More time is required to construct a full-coverage roadmap for this structure, and this roadmap, which is propagated through the improvement algorithm and updated as new configurations are added to the tour, is about twice as large for the *Seneca* as it is for the *Curtiss*.

Diminishing returns can be observed in Figure 4-5 as cost improvements are made. The first ten thousand samples drawn, emphasized in Figure 4-5(b), are especially productive, and responsible for the majority of improvement during the two-hour computational trials. These samples were drawn for the *Curtiss* in about five minutes of



(a) Each plot above features, on the x-axis, the largest number of samples common to all twenty-five trials, rounded to the nearest hundred thousand at left and to the nearest million at right.



(b) The plots above zoom in on the first ten thousand samples drawn for each ship.

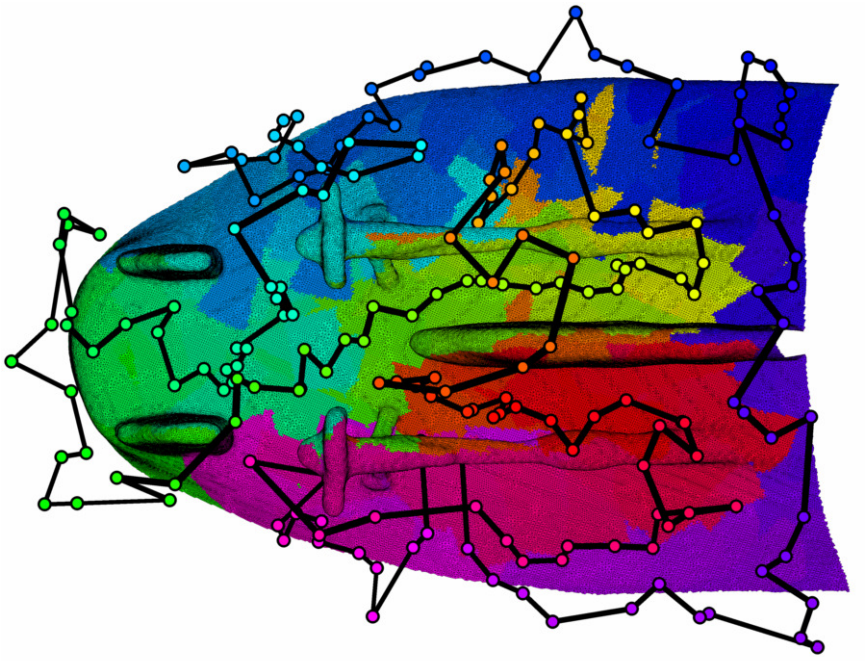
Figure 4-5: Each plot above summarizes the results of twenty-five two-hour trials in which inspection tours were planned for both the *USCGC Seneca* and the *SS Curtiss*. The mean percentage reduction in both the number of view configurations and the tour length is plotted as a function of the number of configurations sampled during the improvement procedure. A data point is plotted for every two thousand samples drawn.

computation time, and for the *Seneca* in about twenty minutes of computation time. The representative inspection tours plotted in Figures 4-6 and 4-7 show that significant simplification and shortening has occured in the time alloted for improvement. In particular, the heuristic speed-up introduced in Section 4.2.3 is responsible for bringing view configurations in certain local sections of a tour into perfect alignment. This could not be achieved in finite time using solely the $\text{RRT}^*_{||}$ algorithm.

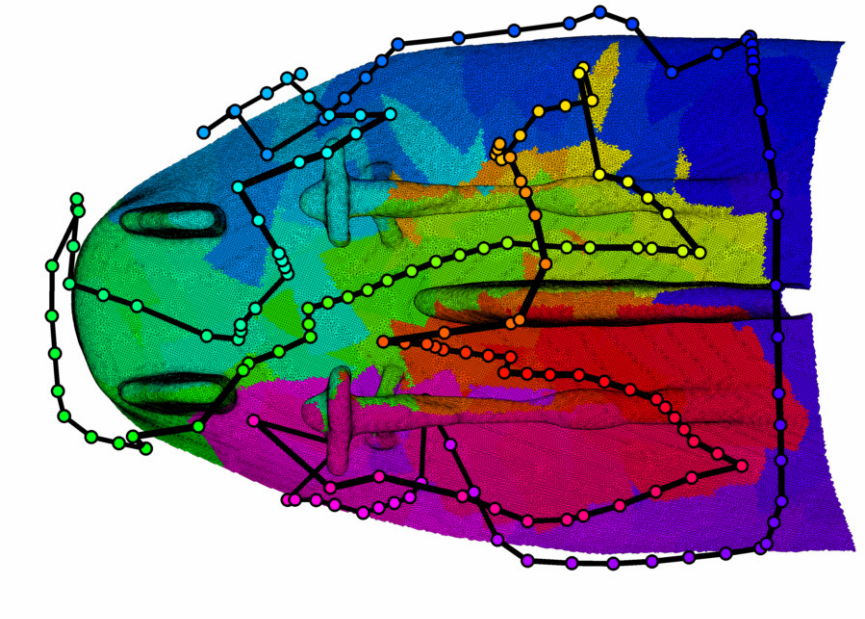
4.5 Summary

We have proposed an iterative procedure for shortening feasible coverage paths over complex structures. This method makes asymptotically optimal local improvements to an inspection, the best possible without invoking an NP-hard combinatorial optimization problem. As is generally the case in the iterative improvement of paths in obstacle-filled environments, a larger investment is required to achieve a near-optimal solution than to simply construct a feasible solution. This investment is characterized by a diminishing returns relationship, but it is worth pursuing when significant mission time can be saved as a result.

We have extended the work on this subject from traditional path planning to coverage path planning, in which not only is obstacle avoidance required, but also the observation of thousands of geometric primitives by the robot sensor. This is a challenging task for which sampling-based planning tools continue to be well-suited.

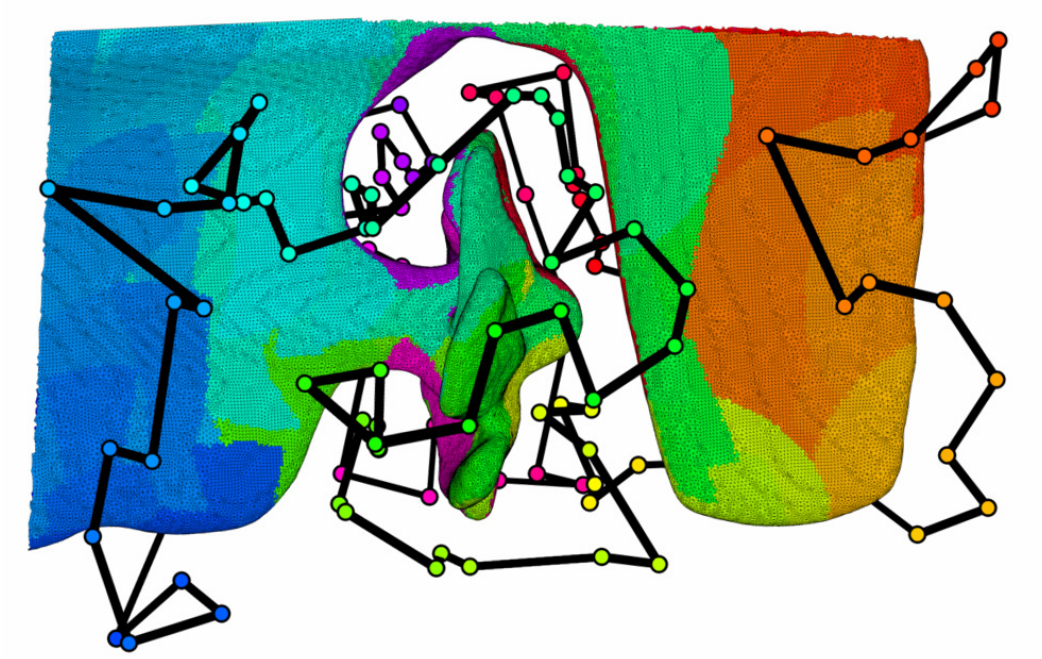


(a) Feasible tour for full coverage of *USCGC Seneca* running gear. The tour is 246 m in length and contains 192 configurations.

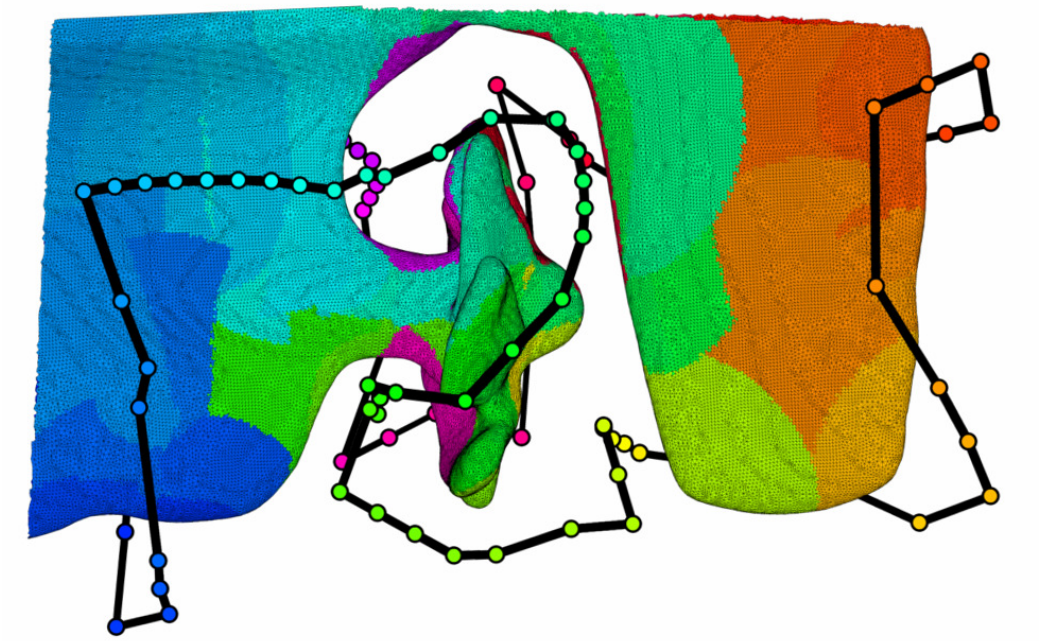


(b) Shortening the tour of (a) using improvement procedure. The shortened tour is 157 m in length and contains 169 configurations.

Figure 4-6: Representative full-coverage *USCGC Seneca* inspection paths before (top) and after (bottom) the improvement procedure.



(a) Feasible tour for full coverage of *SS Curtiss* running gear. The tour is 176 m in length and contains 121 configurations.



(b) Shortening the tour of (a) using improvement procedure. The shortened tour is 102 m in length and contains 97 configurations.

Figure 4-7: Representative full-coverage *SS Curtiss* inspection paths before (top) and after (bottom) the improvement procedure.

Chapter 5

Sampling-Based Sweep Paths

5.1 Introduction

The planning algorithms presented in this thesis are designed to cover 3D structures with confined and occluded regions, but sometimes such structures also possess areas that are open and easily accessible. Not every square inch of such a structure needs a specially designed view to be observed; it may be possible to cover large sections using a highly regular path like the back-and-forth sweep patterns and cross-sectional looping patterns used in cell decomposition methods [3]. Paths that contain uniform spacing between tracklines and accumulate data slice-by-slice along a single spatial dimension of the workspace will improve the clarity and continuity of an inspection's final data product, simplifying the tasks of analysis and processing for a human operator. We are concerned with situations in which easy reading and interpretation of the robot's data is a desirable objective, but the structure as a whole is too complex and occluded to be covered in its entirety by a back-and-forth sweeping pattern.

To address this task we have developed a two-phase path planning strategy that takes advantage of the simplicity and efficiency of modular and sweep-based planning methods while considering the collision and occlusion hazards in the most confined areas of a ship's stern. First, *a priori* triangle mesh models of structures are segmented to isolate planar areas using a hierarchical face-clustering algorithm [13], and a planar, sweep-based path is designed for each segment. The paths are generated using a

sampling-based algorithm that checks all sweep paths against the entire mesh model for collisions, not just the segment being covered. This procedure comes with no guarantee of full coverage; it is simply intended to exploit the open, planar regions of a complex structure using simple and intutive paths.

Then, after designing sweep trajectories for all segments, the redundant roadmap algorithm of Chapter 3 is used to fill in the gaps in coverage with individual robot configurations that observe the remaining areas of the structure. An inspection tour specifying the order of traversal among sweep paths and gap-filling configurations is computed by reduction to the traveling salesman problem, which is solved using the chained Lin-Kernighan heuristic [8].

In Section 5.2 we introduce our hybrid sweeping-and-sampling procedure used to obtain 100% structure coverage. We define the property of probabilistic completeness in the context of sweep-based path planning and analyze our algorithm's completeness and convergence to a feasible solution. In Section 5.3 the combinatorial optimization steps are presented that build a full-coverage inspection tour from our hybrid components, and in Section 5.4 we present computational results of the algorithm.

5.2 Obtaining 100% Coverage of the Structure

We obtain full coverage of the structure through a combination of back-and-forth sweep paths and individual configurations, which fill in the gaps in coverage left by the sweep paths. Unlike most sweep-based coverage planning algorithms, which assume continuous sensing by the end effector along a back-and-forth trajectory, the paths we construct are comprised of discrete, static waypoints arranged in a grid. This strategy, much like the randomized algorithms of Chapters 3 and 4, allows the HAUV to accurately stabilize at each waypoint for the collection of data.

The complete algorithm for generating a sweep-based 100%-coverage inspection tour is illustrated in Figure 5-1. In this section we present our solution of the coverage sampling problem (CSP), the problem of sampling a set of feasible configurations that achieves 100% coverage of a structure boundary. In Phase I of the CSP, a waypoint

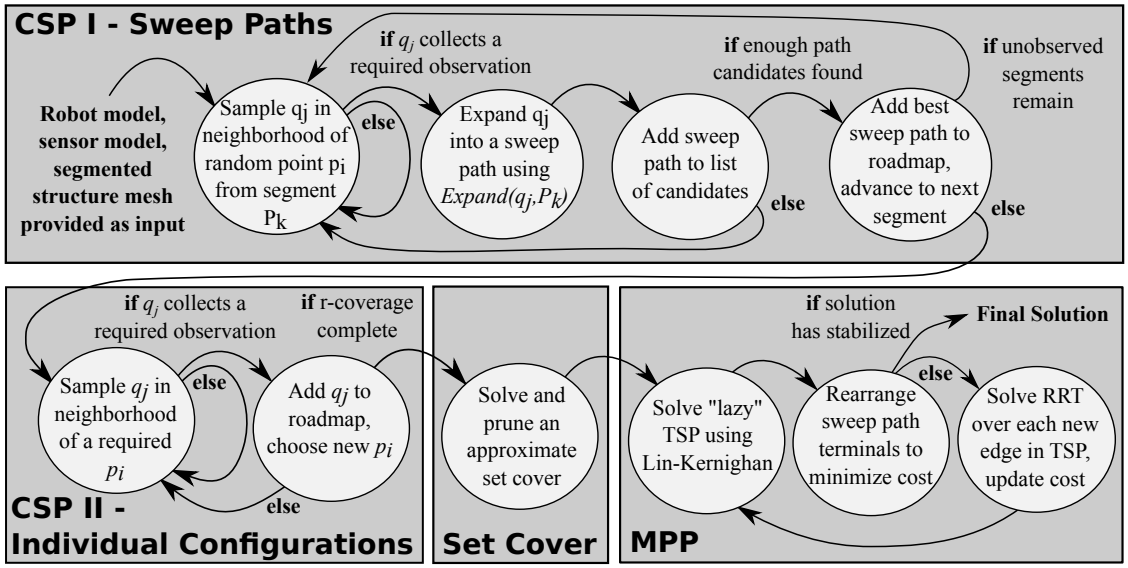


Figure 5-1: A stateflow diagram illustrating the complete algorithm for sampling-based coverage path planning, comprised of a coverage sampling phase to generate sweep paths, a coverage sampling phase to fill in the remaining gaps in coverage, a set cover phase, and a multigoal planning phase.

grid is generated for each surface in the mesh segmentation. In Phase II, individual configurations are sampled to cover the unobserved remainder of the structure mesh. An example of the waypoints designed in each phase of the CSP is given in Figure 5-2. Once a full-coverage set of configurations is obtained, a set cover is solved over the configurations. The final step is solution of the multi-goal planning problem (MPP), in which the grids and other sensing configurations are connected by feasible paths, and an inspection tour is constructed by iterative solution of the TSP.

5.2.1 Sampling-Based Sweep Paths

As mentioned above, a sweep path is not required to cover 100% of the surface segment it is inspecting; the goal is instead to exploit the open, planar areas of the structure wherever possible using a simple trajectory. Using a sampling-based method to achieve this goal reduces the amount of geometric computation required. We can avoid the explicit construction of the robot configuration space, which, for the HAUV, is comprised of four degrees of freedom, x,y,z, and yaw, and is populated with mesh models comprised of hundreds of thousands of geometric primitives. In addition, as

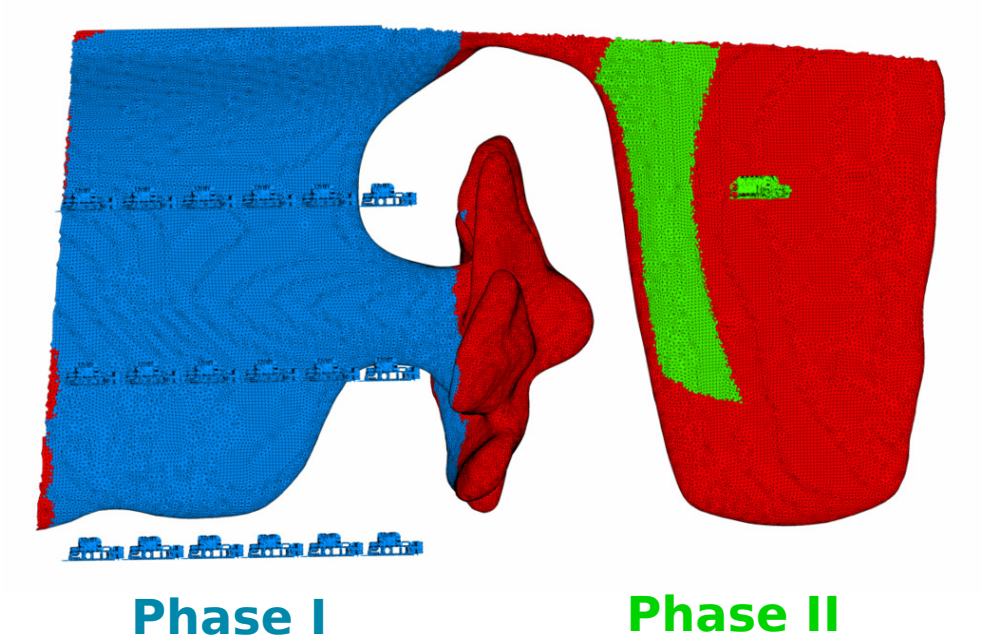


Figure 5-2: A triangle mesh model of the *SS Curtiss* constructed from an HAUV survey, along with waypoints designed to cover the mesh. Illustrating Phase I of the CSP, a waypoint grid and the surface area observed by its waypoints are plotted in blue. The grid is designed to cover a large, planar segment of the mesh. Illustrating Phase II of the CSP, an individual waypoint and its observed surface area are plotted in green.

we demonstrate below, a cell decomposition is not required to fit a long, efficient sweep path in the obstacle-free areas of configuration space; this is achieved instead using random sampling.

Set System Preliminaries

The set system nomenclature used in Chapters 3 and 4 is adapted for the treatment of a polyhedral structure that has been segmented into K non-overlapping pieces. For a mesh segment k, P_k is the set of primitives contained in the segment, $\mathcal{Q}_k$ is the user-defined region of configuration space that is sampled to achieve views of P_k, and $\mathcal{S}_k$ is the set of all configurations that observe any primitive $p_i \in P_k$. For a simple structure with three segments, these parameters are illustrated in Figure 5-3.

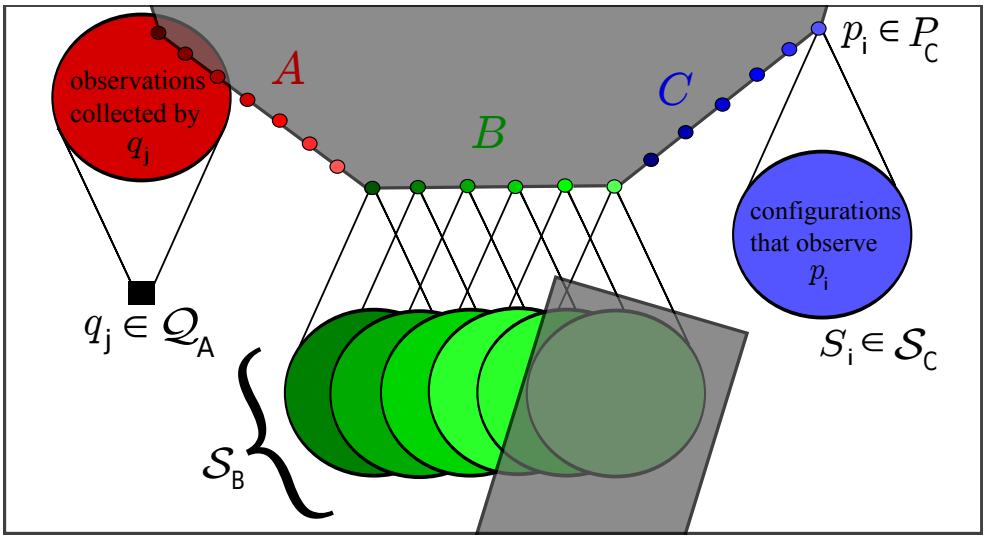


Figure 5-3: An illustration of the set systems involved in the coverage sampling problem, for a robot with a circular sensor footprint capable of translational motion in $\Re^2$. In this example, the structure to be inspected is discretized and segmented into three pieces. One of the primitives in the green partition cannot be observed due to the presence of an obstacle.

Sweep Path Construction Algorithm

As illustrated at the top of Figure 5-1, after choosing a specific mesh segment k to cover, a point from this segment, p_i, is selected at random and configurations q_j are randomly sampled in a local neighborhood of p_i, such that p_i lies within the field of view of the sensor. This procedure gives us the *seed configuration* from which a sweep path is produced. If q_j collects observations of segment k, then the subroutine $Expand(q_j, P_k)$ is called to expand q_j into a grid of waypoints.

Each waypoint grid is constructed in a 2D plane with a single yaw angle common to all waypoints, selected to capture mesh segment k in the sensor field of view. The plane is oriented using the distribution of points in mesh segment k, with the eigenvectors of the segment's statistical covariance matrix comprising the axes for alignment. The waypoints in each grid are either depth-varying or fixed in depth depending on the orientation of the normal vectors in mesh segment k.

A user-specified spacing is enforced between waypoints when $Expand(q_j, P_k)$ expands a seed configuration into a waypoint grid. $Expand(q_j, P_k)$ is given in Algorithm 5; each subroutine attempts to add one extra row or column to the grid, separated

Algorithm 5 $\mathcal{Q}_k = Expand(q_j, P_k)$

$\mathcal{Q}_k \leftarrow q_j$;
$ExpansionComplete = false$;
$SweepPlane = GeneratePlane(q_j, P_k)$;
while $!ExpansionComplete$ **do**
  $\mathcal{Q}_k^{right} \leftarrow ExpandRight(\mathcal{Q}_k, P_k, SweepPlane)$;
  $\mathcal{Q}_k \leftarrow \mathcal{Q}_k \cup \mathcal{Q}_k^{right}$;
  $\mathcal{Q}_k^{up} \leftarrow ExpandUp(\mathcal{Q}_k, P_k, SweepPlane)$;
  $\mathcal{Q}_k \leftarrow \mathcal{Q}_k \cup \mathcal{Q}_k^{up}$;
  $\mathcal{Q}_k^{left} \leftarrow ExpandLeft(\mathcal{Q}_k, P_k, SweepPlane)$;
  $\mathcal{Q}_k \leftarrow \mathcal{Q}_k \cup \mathcal{Q}_k^{left}$;
  $\mathcal{Q}_k^{down} \leftarrow ExpandDown(\mathcal{Q}_k, P_k, SweepPlane)$;
  $\mathcal{Q}_k \leftarrow \mathcal{Q}_k \cup \mathcal{Q}_k^{down}$;
  if $|\mathcal{Q}_k^{right} \cup \mathcal{Q}_k^{up} \cup \mathcal{Q}_k^{left} \cup \mathcal{Q}_k^{down}| = \emptyset$ **then**
    $ExpansionComplete = true$;
  end if
end while
return $\mathcal{Q}_k$

by the designated spacing. Due to this systematic expansion procedure, the seed configuration q_j determines the layout of the entire grid.

Because it may not be possible for a grid to observe all primitives in segment k, we wish to identify the seed configurations whose grids, after expansion, observe the maximum-possible number of primitives in segment k subject to the presence of obstacles, occlusions, and the spacing enforced between waypoints. We are not concerned with growing the shortest-possible sweep path from $\mathcal{S}_k$, simply a feasible path. We use the notation $\mathcal{S}_k^*$ to describe the special subset of $\mathcal{S}_k$ from which a sampled configuration will generate a grid that satisfies the maximum-possible number of coverage constraints. $\mathcal{S}_k^*$ is depicted in Figure 5-4 for the coverage of segment B. It is evident that the rightmost mesh point in segment B is obscured from view by the presence of an obstacle, but any seed configuration in $\mathcal{S}_B^*$ will yield a single-row grid that observes the other five mesh primitives. The spacing of the gray regions of $\mathcal{S}_B^*$ is determined by the user-selected waypoint spacing for this particular example problem.

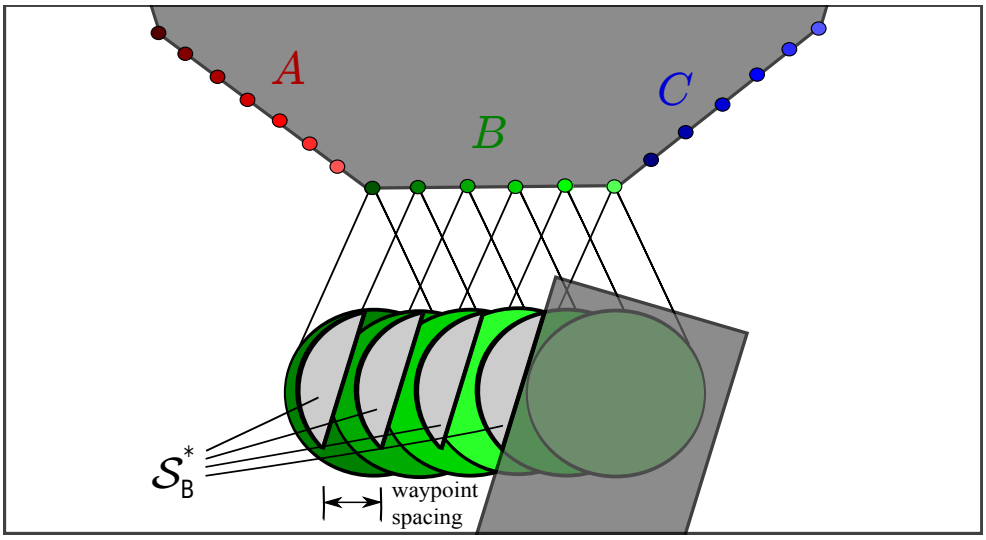


Figure 5-4: An illustration of additional set system nomenclature for a robot with a circular sensor footprint capable of translational motion in $\Re^2$. The set of configurations that map to a maximally informative sweep path is depicted for segment B. One of the primitives in the green partition cannot be observed due to the presence of an obstacle.

Probabilistic Completeness

Random sampling proves to be a powerful tool for finding a maximal-coverage feasible sweep path, and it motivates our definition of *probabilistic completeness* in the context of sweep paths. We analyze probabilistic completeness with respect to a local set system, $(\mathcal{Q}_k, \mathcal{S}_k)$, that applies to a specific segment k. We define the property of probabilistic completeness for a CSP algorithm as follows.

Definition 8 (Probabilistic Completeness, CSP I)**.** *Let CSA be a proposed sweep-based coverage sampling algorithm for Phase I of the CSP. Let* $\delta = \min_k \mu(\mathcal{S}_k^*)/\mu(\mathcal{Q})$ *be the smallest maximal-coverage volume fraction of all segments* k*, where the measure* μ *represents the volume of the specified region of configuration space. If, when* $\delta > 0$*, the probability that at least one sample has landed in every* $\mathcal{S}_k^*$ *approaches one as the number of samples of* $\mathcal{Q}$ *drawn by CSA approaches infinity, then CSA is probabilistically complete.*

This definition implies that a probabilistically complete CSP algorithm will, in the limit, find sweep paths that satisfy as many coverage constraints as possible while

avoiding collision and obeying the rules of sweep path construction. This definition is intended to eliminate degenerate scenarios from consideration in which $\mathcal{S}_k^*$ is a manifold of lower dimension than $\mathcal{Q}$. By relaxing additional coverage constraints, it is possible that a degenerate $\mathcal{S}_k^*$ can be replaced with a new set that achieves a nonzero volume fraction of $\mathcal{Q}$. We can further analyze probabilistic completeness by studying the simple event of whether a randomly sampled configuration q_j lands in a particular set $\mathcal{S}_k^*$.

Theorem 7 (Completeness & Convergence, CSP I)**.** *Any algorithm for Phase I of the CSP that samples uniformly at random from all $\mathcal{Q}_k$ such that $\mu(\mathcal{S}_k^*)/\mu(\mathcal{Q}_k) \geq \epsilon > 0 \; \forall k$ is probabilistically complete. Additionally, the probability that a solution has not been found after m samples of each $\mathcal{Q}_k$ is bounded such that*

$$Pr[FAILURE] \leq \frac{K}{e^{\,m\epsilon}} \quad , \tag{5.1}$$

where K is the number of partitions into which P is segmented.

Proof. The probability of m samples of each $\mathcal{Q}_k$ producing a maximal-coverage CSP solution is equivalent to the probability that at least one random sample has landed in every set $\mathcal{S}_k^*$. This fails to occur if there is at least one $\mathcal{S}_k^*$ in which no samples have landed. To model this event, we define the binomial random variable $X_k = X_{k_1} + X_{k_2} + ... + X_{k_m}$, which gives the number of samples that have successfully landed in $\mathcal{S}_k^*$ out of m total trials. We express the probability of CSP algorithm failure as follows:

$$\begin{aligned} Pr[FAILURE] &\leq Pr\left[\bigcup_{k=1}^{K} X_k = 0\right] \\ &\leq \sum_{k=1}^{K} Pr[X_k = 0] \\ &\leq K \cdot Pr[X_{k^*} = 0] \end{aligned} \tag{5.2}$$

Using the union bound, the probability that $X_k = 0$ for at least one $\mathcal{S}_k^*$ is bounded above by the sum of the probabilities of this event for all $\mathcal{S}_k^*$. This is further sim-

plified by taking $Pr[X_{k^*} = 0]$ as an upper bound on the failures of all X_k, where X_{k^*} is the binomial random variable corresponding to the segment k that minimizes $\mu(\mathcal{S}_k^*)/\mu(\mathcal{Q}_k)$.

Since we are sampling uniformly at random, $Pr[X_{k^*} = 0]$ can be expressed and bounded in the following way:

$$Pr[X_{k^*} = 0] = (1 - \epsilon)^m \leq e^{-m\epsilon}, \quad 0 \leq \epsilon \leq 1 \tag{5.3}$$

Combining the result of (5.3) with (5.2), we obtain the desired relationship between m and the probability of failure:

$$Pr[FAILURE] \leq \frac{K}{e^{\,m\epsilon}}, \quad \lim_{m\to\infty} \frac{K}{e^{\,m\epsilon}} = 0 \tag{5.4}$$

Since $\mu(\mathcal{S}_k^*)/\mu(\mathcal{Q}_k) > 0 \; \forall k$, $\epsilon > 0$ and the limit behaves as indicated in (5.4). □

As a direct consequence of Theorem 7, our algorithm for Phase I of the CSP illustrated in Figure 5-1 is probabilistically complete if the $\mathcal{Q}_k$ are selected to allow $\epsilon > 0$ whenever $\delta > 0$. By iteratively choosing a random $p_i \in P_k$ and sampling from the region of $\mathcal{Q}$ in which p_i lies in the sensor's geometric footprint, we are sampling from a subset of $\mathcal{Q}$ which fully includes $\mathcal{S}_k^*$ and the condition on ϵ and δ is always satisfied. The bounding methods used in this analysis were used previously in the proof of completeness of the probabilistic roadmap [88] to analyze the failure of m samples of a common configuration space to construct a collision-free path between two configurations. We have applied the same bounds here to analyze the failure of m samples of each $\mathcal{Q}_k$ to land at least once in every set $\mathcal{S}_k^*$.

5.2.2 Filling in the Gaps

To fill in the remaining gaps in coverage left by the sweep paths, we rely on individual robot configurations rather than waypoint grids. This sub-problem comprises Phase II of the CSP as illustrated in Figure 5-1. To solve this problem, we utilize the sampling method of the redundant roadmap algorithm presented in Chapter 3, which

samples robot configurations until a set is obtained that views each required geometric primitive from r distinct configurations, termed r-coverage in Figure 5-1. From these configurations, a subset is selected for traversal by approximation of the minimum-cardinality set cover. In Phase II of the CSP, we apply the sampling routine of the redundant roadmap algorithm only to primitives left unobserved by the sweep paths designed in Phase I.

This sampling routine is also probabilistically complete. If a feasible, 100%-coverage set of configurations exists for the remaining primitives, then the redundant roadmap algorithm will find such a solution with probability that tends to one as the number of samples tends to infinity, as demonstrated in Chapter 3. Using this result, we state the convergence of algorithm failure probability as a function of the number of samples m, the volume fraction ϵ of the configuration space that is sampled, and $|P|_{gaps}$, the number of primitives comprising the gaps in coverage remaining at the beginning of Phase II.

$$Pr[FAILURE] < |P|_{gaps} \cdot \frac{e^r}{e^{\,m\epsilon/2}} \tag{5.5}$$

The coefficients in (5.5) differ slightly from (5.1) because the Phase II sampling process must achieve r-coverage, as opposed to single-coverage. Despite the minor differences between (5.1) and (5.5), both phases of the coverage sampling problem are solved by algorithms for which the probability of failure plunges toward zero exponentially fast in the number of robot configurations sampled.

5.3 Computing A Hybrid Inspection Tour

Phases I and II of the CSP yield a set of feasible robot configurations that observe 100% of the structure boundary. Part of this set is comprised of waypoint grids, which form the basis for back-and-forth sweep paths. The remainder of the set is comprised of individual robot configurations that fill in the gaps in coverage left by the waypoint grids. Before constructing an inspection route among these configurations, we apply a

set cover approximation to both the sweep-path subset and gap-filling subset, followed by iterative pruning of each set cover solution. After the set cover step, an order of traversal among waypoints is computed by reduction to a symmetric instance of the TSP.

5.3.1 Set Cover Sub-Problem

The set cover problem is solved twice; once over the K sweep paths and once over the individual configurations that fill the remaining gaps in coverage. In the former case, each sweep path is treated as an individual set, and in the latter case, each robot configuration is treated as an individual set. Each set cover is posed over the specific group of geometric primitives required in the respective phase of the CSP. In both cases, the greedy algorithm is used to give a polynomial-time approximation to the minimum-cardinality set cover. The greedy algorithm returns a feasible solution, but this solution may contain sets that can be eliminated completely while preserving feasibility. A pruning algorithm is implemented to remove sets which cover no elements uniquely, as is performed in the original redundant roadmap algorithm. For the sweep paths, however, the pruning procedure is also applied to the individual rows and columns of each waypoint grid, and in each iteration the obsolete row or column with the largest number of waypoints is eliminated. This allows redundant waypoints to be eliminated from the sweep paths while preserving their rectangular structure.

5.3.2 Traveling Salesman Sub-Problem

Our aim is to solve the TSP over a graph containing sweep paths without re-computing the order of traversal within the sweep paths themselves. After choosing an entry and exit point, the order of traversal within a sweep path is trivial, as depicted in Figure 5-5. Consequently, we reduce each sweep path in the set cover to a single pair of graph nodes in the TSP, representing the points of entry and exit. To ensure that this pair of sweep path *terminals* appear adjacent to one another in the final TSP

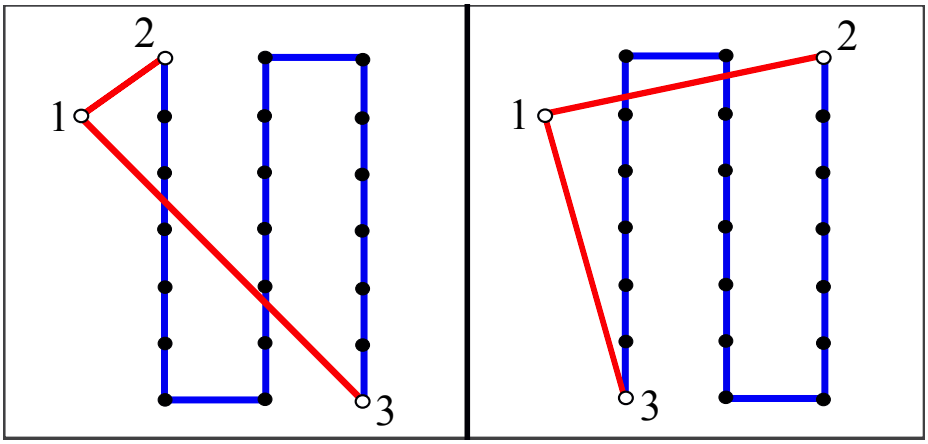


Figure 5-5: A diagram illustrating the integration of back-and-forth sweep paths into the TSP. At left, one possible choice of sweep path is depicted, and at right, the alternate choice is depicted. Each choice results in a different set of terminals being used to connect the sweep path to the rest of the inspection tour. For each choice of terminals, the red lines and numbered points represent edges and nodes that are introduced into the TSP. The blue lines represent the sweep path, which is omitted from the TSP and represented by a zero-cost edge between the two terminals.

tour, the costs of travel between other configurations are augmented. A cost of zero is assigned to every edge connecting a pair of terminals, and all other node-to-node costs are initialized using the Euclidean distances between robot configurations. A large number is then added to the costs of all Euclidean-distance edges. This large number, selected to be larger than any true path length that will be returned as a solution to the TSP, will ensure that pairs of terminals remain adjacent in the final TSP tour. We are not aware of prior work on the topic of forcing a pair of TSP nodes to be adjacent. After this initialization, the TSP is solved using the chained Lin-Kernghan heuristic [8].

Even though a pair of terminals is selected for each sweep path in advance of solving the TSP, it is possible that the alternate pair of entry and exit terminals will yield a shorter inspection tour, as demonstrated in Figure 5-5. To address this possibility, we consider alternate choices of sweep path terminals and switch them after solution of the TSP if the alternate terminals lower the cost of the tour. We iterate through the sweep paths in round robin fashion, and stop once a single pair of terminals is adjusted. This adjustment requires an update to the node-to-node

distances in the adjacency matrix used for TSP computations. After this update is performed, any node-to-node pairings that have not been collision-checked are solved according to the multi-goal planning problem (MPP) iterative procedure illustrated in Figure 5-1; a similar lazy procedure is used in the redundant roadmap algorithm presented in Chapter 3.

Our terminal-switching heuristic is intended to avoid the complexity of examining, in every iteration of the MPP procedure, all 2^K combinations of terminals over K sweep paths. Despite our simplification of the problem, even the proposed heuristic risks the worst-case scenario of an MPP procedure that marches through every one of these 2^K combinations while approaching a stable solution. However, this would only occur in the unlikely scenario that every combination makes incremental progress toward a single optimal solution. We have found the heuristic MPP procedure to converge quickly in practice; the entire sweep-based planning algorithm has required no more than ten minutes of computation time in any of the problem instances tested. If the switching procedure were to result in excessive computation, then a time limit, a ceiling on the allowed number of MPP iterations, or a stopping criterion based on the cost improvement of the MPP procedure could always be imposed.

5.4 Computational Results

We now give computational results of the sampling-based sweep path algorithm as applied to the HAUV. Once again, we will assume that the HAUV will inspect the *USCGC Seneca* and *SS Curtiss* using a DIDSON viewing range of 1-3 meters. This is a small sensing volume relative to the size of the structures being inspected, and conservative waypoint spacing must be used to prevent the occurence of gaps in the data collected while sweeping over open and planar areas.

In addition to the need for heuristic design of waypoint spacing, we must decide how many partitions are appropriate in the segmentation of both structures. To explore the effect of this parameter, we have computed planned inspection paths over both ships for a variety of segmentations, from the trivial case of a fully randomized

inspection (an order-zero segmentation), to a segmentation of order twenty. This was performed using EfPiSoft, an implementation of a hierarchical face clustering algorithm [13] that we have used to select segments based on their quality of fit to a plane. It is also possible to select segments based on their quality of fit to spherical and cylindrical primitives, but we have found spherical and cylindrical sweep paths to be less suitable for generalized inspection of the open areas of man-made structures. Given a mesh segmentation as input, our sweep path algorithm carries out random sampling until ten feasible candidate sweep paths are achieved for each segment, and the paths offering the most comprehensive coverage of their respective segments are used in the inspection. We proceed this way in practice since we do not know exectly when the maximal-coverage set $\mathcal{S}_k^*$ is reached.

After the sweep paths are generated, the remaining gaps in coverage are filled using the CSP procedure of the redundant roadmap algorithm. The gaps in coverage are filled using redundancy-three roadmaps, which must observe all required geometric primitives from three distinct sampled configurations. In each iteration of the MPP, a TSP tour is initialized using the nearest-neighbor heuristic and the chained Lin-Kernighan algorithm is applied for one second, although sometimes only milliseconds are needed to make significant improvements to the TSP tour. All computations were performed on a Dell T3500 desktop with a 3.20GHz Intel Xeon processor and 24GB of RAM, and no single instance of the full planning algorithm required more than ten minutes of computation time for the structures tested. The planning procedure was implemented in C++, and the high performance software tools listed in Table B.1 of Appendix B were once again used where applicable.

We note that this algorithm is fully compatible with the sampling-based improvement procedure described in Chapter 4. Assuming that the improvement procedure is only applied to randomized configurations in the inspection tour, to avoid disturbing the regularity of the sweep paths, the impact of the procedure will vary depending on the parameterization of the sweep-based planning algorithm. When a great majority of the structure can be covered using back-and-forth sweep paths, the ideal outcome of our hybrid algorithm, the impact of the improvement procedure will be limited.

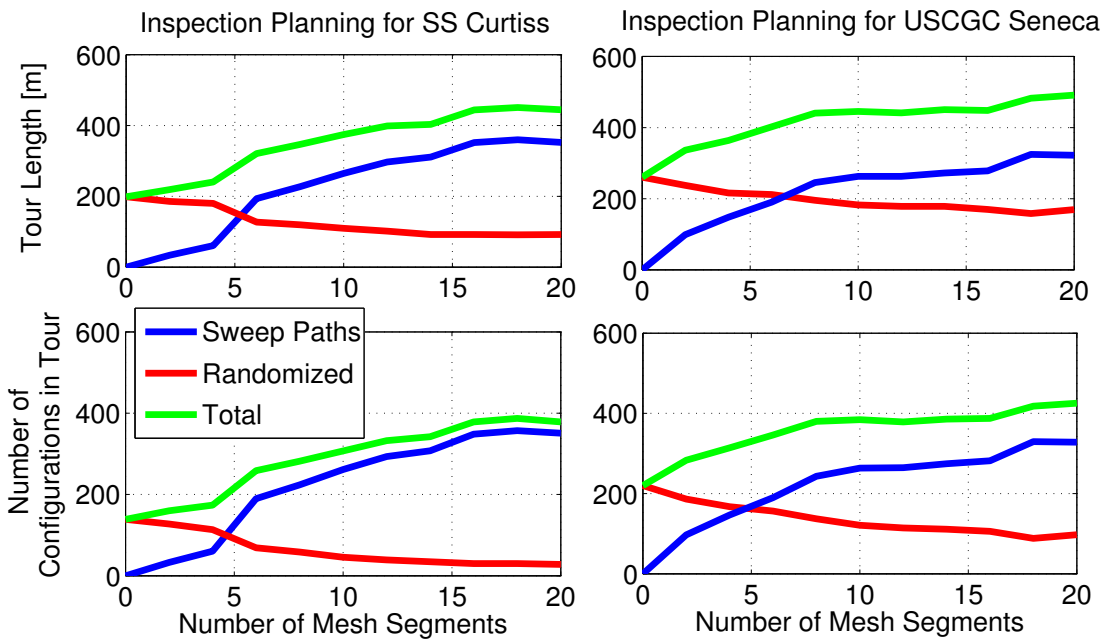


Figure 5-6: Results of sweep-based inspection planning on two vessels, the *SS Curtiss* and *USCGC Seneca*, over different segmentation cases. These range from the trivial case of a fully randomized inspection (zero segments) to the case in which one sweep path is formed for the entire structure (one segment) to nontrivial cases with up to twenty segments. The results give the mean inspection tour length over 25 trials and the mean number of configurations (waypoints) in the inspection for each test case. In blue, we plot the length of the tour contributed internally by all sweep paths. Blue also represents the number of sweep path configurations. In red, we plot the length of the tour required for interconnections among separate sweep paths and single configurations. Red also represents the number of single configurations. The sum total of these quantities is plotted in green.

For this reason, we omit post-optimization smoothing from the computational results of this chapter.

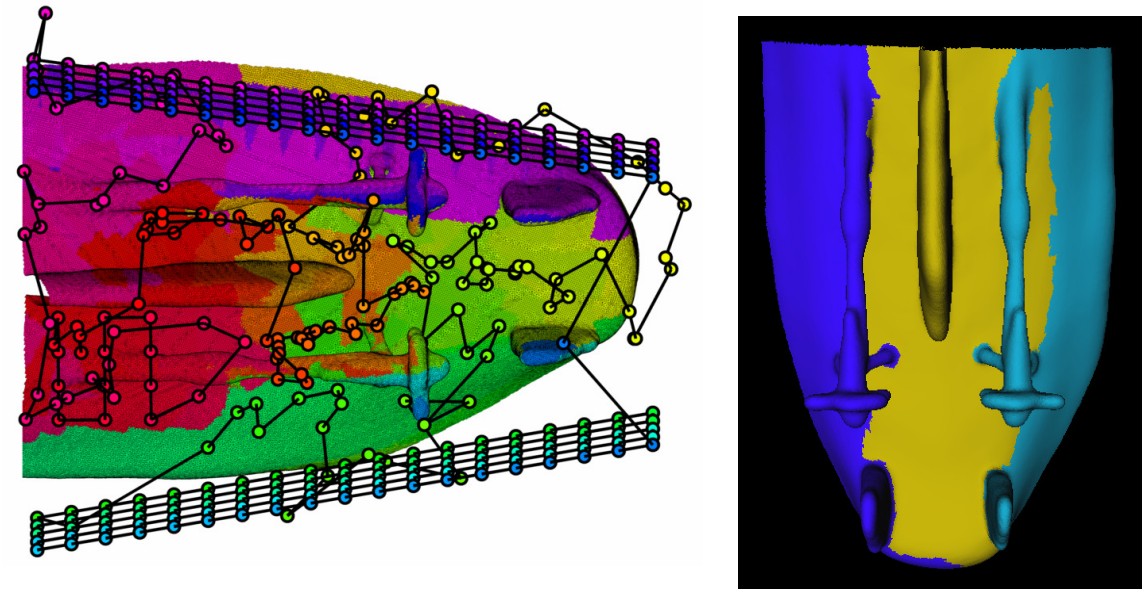
Due to the uniform spacing and fixed orientation of sweep path waypoints, HAUV trajectories that utilize sweep paths will suffer a loss in efficiency to exchange randomized inspection routes, which accomodate every unique twist and turn in the structure, for highly regularized inspection routes. This loss in efficiency impacts both the number of configurations used in the inspection and the distance traveled by the vehicle along the inspection tour. By planning for HAUV coverage of a large trivially-segmented cube, the loss of efficiency was determined to be a factor of two

for inspection tour length and a factor of 2.5 for the number of waypoints in an idealized inspection route for which nearly 100% of wayponts lie in sweep paths. These losses were matched and exceeded in some cases by the planned coverage paths for the *Curtiss* and *Seneca*, which were planned over a wide range of mesh segmentations. Figure 5-6 demonstrates these results, which illustrate the proportion of each planned inspection comprised of regularized and randomized configurations. As the quantity of segments increases, larger proportions of the tour are solved by sweep paths. This is accompanied by a net increase in length of the tours and the number of total configurations, with a decrease in the number of randomized configurations. The effect of higher-order segmentations on the decrease in randomized configurations is observed to diminish as the number of segments increases.

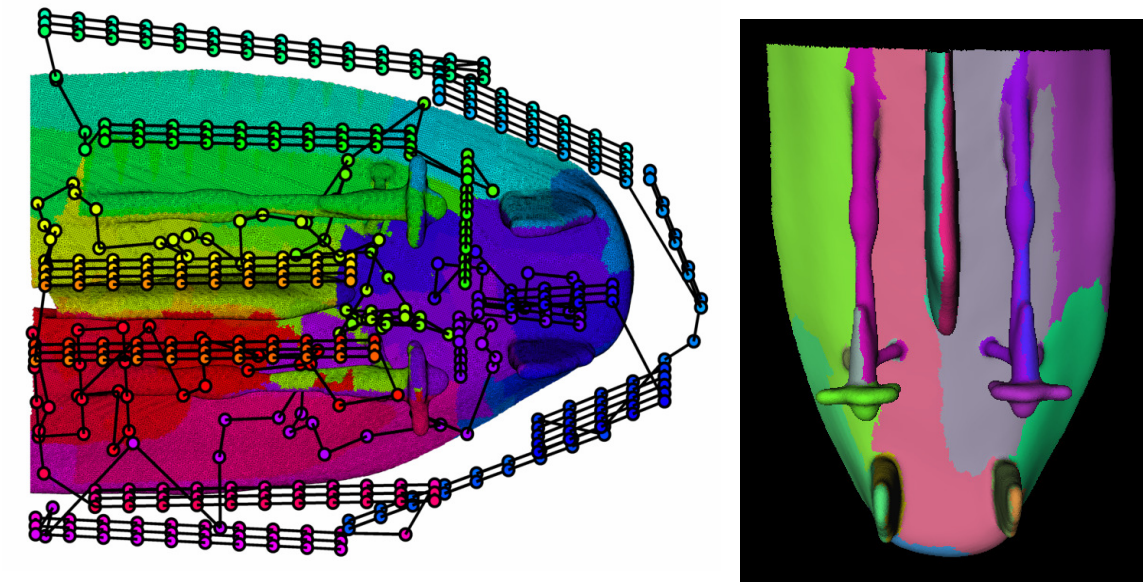
This diminishing-returns effect is especially evident for the *Curtiss*, which is covered almost entirely by sweep paths using an order-ten segmentation, pictured in 5-8. As illustrated in 5-6, increasing the order of the segmentation beyond ten has only a minor effect on the number of randomized configurations, while it increases the total length of the tour significantly. The *Seneca*, on the other hand, still requires a significant number of randomized configurations for an order-twenty segmentation. The *Seneca* has a larger number of protruding component structures, and many additional planes are needed to observe these structures from all sides. The coverage path planned for the order-twenty segmentation pictured in 5-7 uses planar sweep paths to observe both sides of the keel, both sides of each rudder, one side of each shaft, and the faces of the propellers. Although there is no single, correct choice of an optimal segmentation, it is clear that different structures will require subdivisions of differing complexity to approach full coverage with regularized sweep paths.

5.5 Summary

We have presented an algorithm which, to our knowledge, is the first coverage planning algorithm that utilizes both randomized and regularized component paths to achieve coverage of complex 3D structures. The component paths are joined seamlessly into



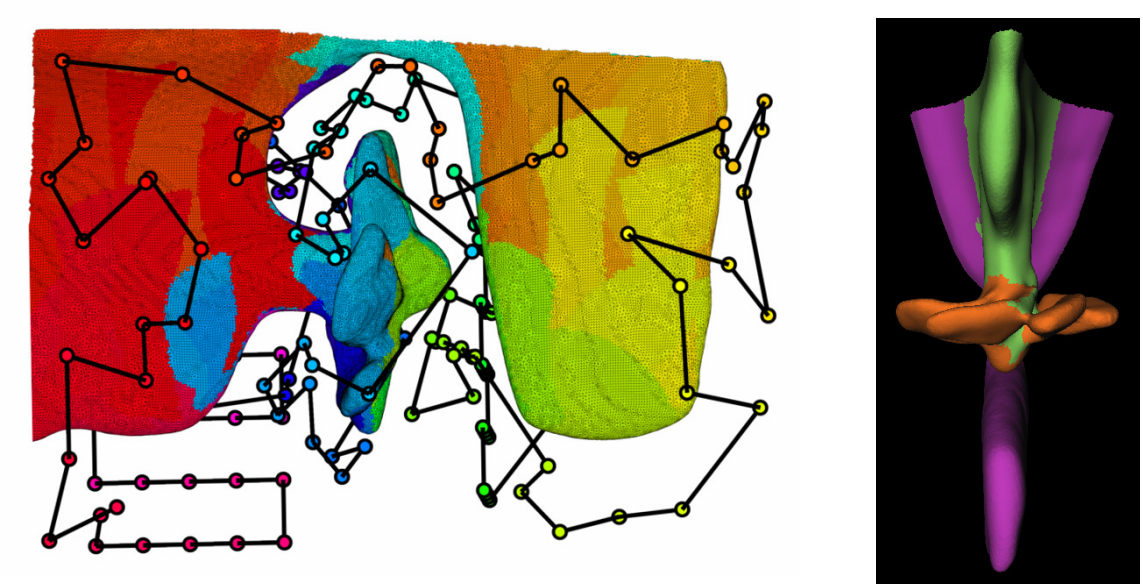
(a) This three-segment tour is 402 m in length and contains 145 randomized configurations and 202 sweep configurations.



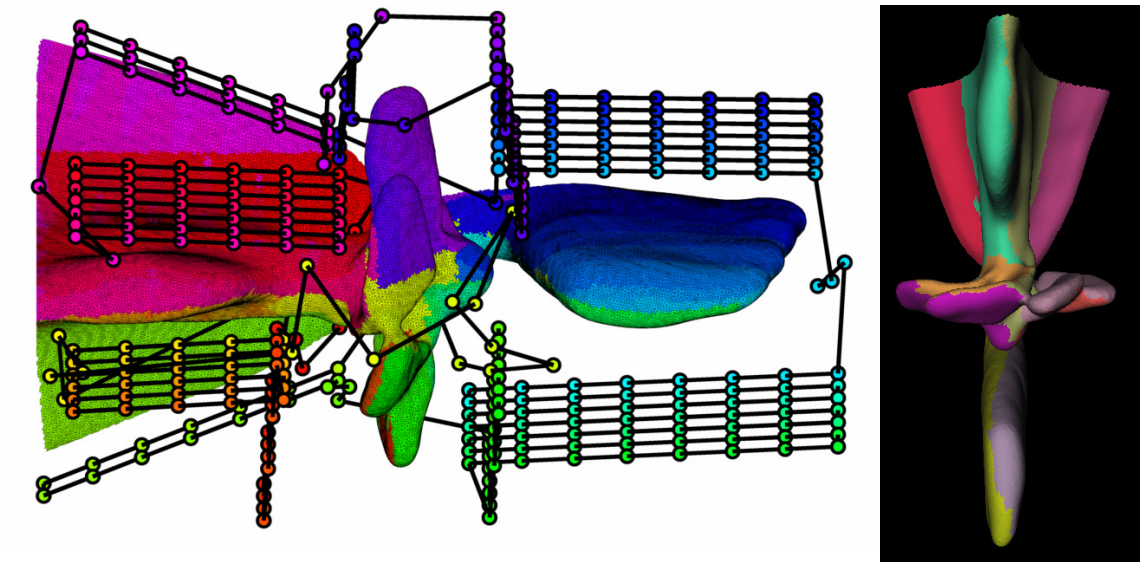
(b) This twenty-segment tour is 526 m in length and contains 84 randomized configurations and 377 sweep configurations.

Figure 5-7: Examples of planned inspection tours for the *USCGC Seneca*, for three-segment and twenty-segment test cases. The images at right illustrate the segmentation only, and the images at left illustrate the full-coverage inspection tour.

a single contiguous inspection tour. Given a segmented structure as input, a back-and-forth sweep path is designed for coverage of each segment. A probabilistically complete sampling procedure is used to establish the origin of each sweep path. This



(a) This three-segment tour is 241 m in length and contains 112 randomized configurations and 57 sweep configurations.



(b) This ten-segment tour is 383 m in length and contains 36 randomized configurations and 282 sweep configurations.

Figure 5-8: Examples of planned inspection tours for the *SS Curtiss*, for three-segment and ten-segment test cases. The images at right illustrate the segmentation only, and the images at left illustrate the full-coverage inspection tour.

procedure is designed to cover the open, easily accessible areas of a structure using simple paths that yield easy-to-interpret sensor data. Randomized paths are used to inspect the confined, highly-occluded areas of a structure that elude the sweep paths.

To minimize the number of random configurations used in a planned inspection, a loss in efficiency must be accepted in the substitution of uniformly spaced waypoint grids for individually designed single waypoints. This tradeoff is often desirable when the ability to monitor, interpret, and intervene in an inspection-in-progress is a key requirement, and our algorithm offers the flexibility to "trade" for increased regularity as needed.

Chapter 6

Experimental Outcomes

6.1 Introduction

Several field and laboratory experiments have been performed to support the development and testing of the path planning algorithms presented in this thesis. Despite the availability of computer-aided design (CAD) models for many ships and other man-made structures, we have been unable to obtain such models for the specific ships used in our hull inspection field experiments. Consequently, the ability to discover and map a vessel's stern arrangements, without the aid of a CAD or other model, has been valuable in our work with naval vessels. Many vessels are older, and poorly documented, or have been modified to an extent that the available description is simply incorrect. Thus, our methodology is intended to proceed from having no knowledge of the structure, to a survey made at large range and poor resolution, to another survey made at short range and high resolution. The coarse survey enables the fine survey "up and into" the gear, which is planned using the algorithms of the preceding chapters.

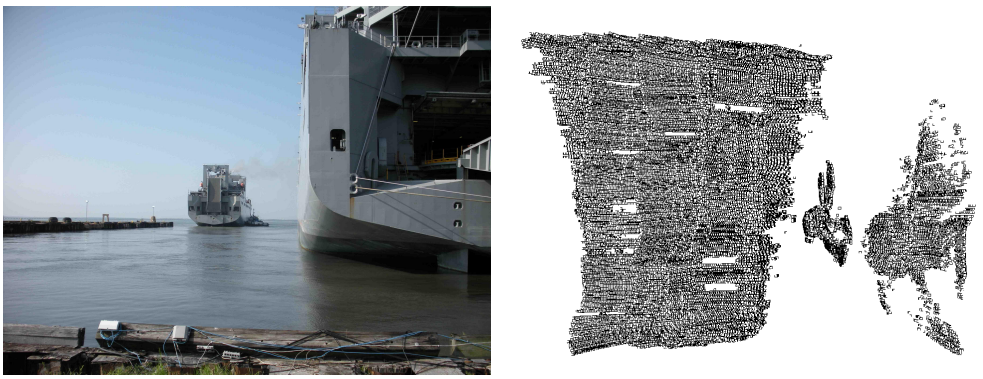
At a safe distance from the stern (typically seven to ten meters), we first execute the low-resolution *identification survey*; this is intended to identify the major ship structures, and enable the construction of a 3D model or at least allow the HAUV to reference itself to a prior model. Due to the challenges associated with filtering profiling-mode DIDSON data—including the removal of noise and second returns—

we use manual processing to construct the mesh. Using this coarse 3D model, a path is then planned for a subsequent high-resolution *inspection survey* to support the recognition of mines. Once this path is executed, the mesh modeling techniques of the identification survey can be applied again at higher resolution. Seven HAUV field experiments, performed at six different vessels over the course of three years, have provided an opportunity to refine the implementation of the identification and inspection surveys. Data products from these experiments are illustrated in Figures 6-1 and 6-2, and a more detailed summary is provided in Table B.2 in Appendix B.

In Section 6.2 we describe the procedure used to survey a vessel from a safe distance to support the generation of a structure-resolution triangle mesh model. In the subsequent sections we give results from the execution of planned inspection surveys, intended to achieve mine-resolution coverage of the structures inspected. A planned path implemented by the HAUV at the stern of the *USCGC Seneca* is presented in Section 6.3, and a laboratory experiment performed using a laser rangefinder in air is discussed in Section 6.4. This experiment, an approximate one-tenth-scale mockup of a ship hull inspection, was designed to achieve both sensing and positioning outcomes of higher precision than is attainable using the current HAUV. Our laboratory experiment is intended to shed some light on future HAUV capabilities, and how our algorithms can support them, as underwater navigation and sensing are improved.

6.2 Mesh Construction Procedure

Within the community of laser-based range sensing, specialized algorithms have been designed to generate watertight, 3D mesh models from high-resolution point clouds [74], [45]. Laser-based range sensing, ubiquitous in ground, air, and space applications, however, yields substantially higher-resolution point clouds than does underwater acoustic range sensing: typically sub-millimeter versus sub-decimeter resolution. This is evident in several studies that have pursued mapping of 3D structures using underwater acoustic range data [29], [57], [94], [25]. Fortunately, a number of derivative tools have been developed for processing point clouds containing gaps, noise, and

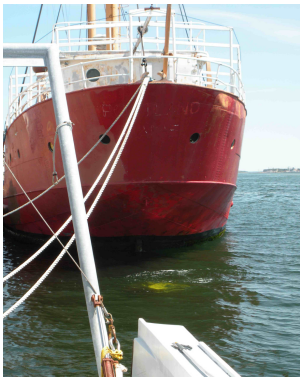
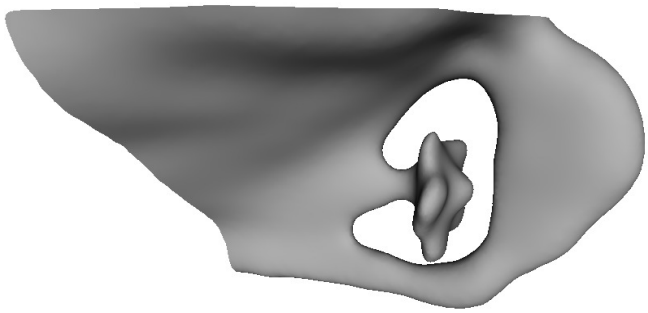


(a) Photo and point cloud from the *USNS Red Cloud*. The photo shows the *Red Cloud* at right, with a ship of the same class departing at left. The point cloud shows a rudder and portions of both propellers.

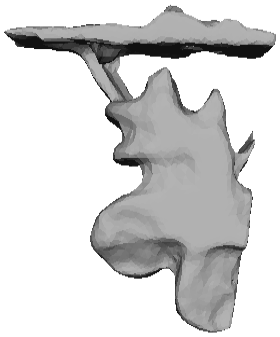
(b) Photo and point clouds from the *USCGC Venturous*. The point clouds show the *Venturous* from the starboard side and the stern, respectively. Photo credit: US Coast Guard, `http://www.uscg.mil/lantarea/cgcventurous/`

(c) Photo and mesh from the *SS Curtiss*. This mesh is based on high-quality, comprehensive point cloud data and was used as one of the primary tools in planning algorithm development.

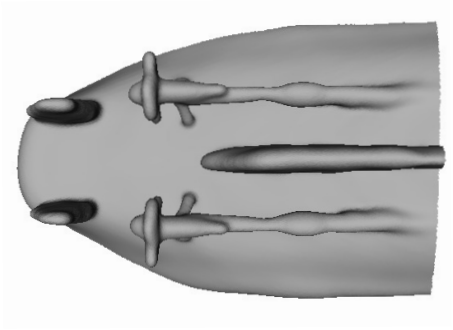
Figure 6-1: A summary of HAUV field experiments performed in support of coverage algorithm development and planned path execution, part one.

(a) Photo and mesh from the *Nantucket Lightship*. This small ship was used for practicing the execution of a planned inspection route.

(b) Photo and mesh from the *M/V Terry Bordelon*. The mesh depicted focuses on a propeller and its supporting structures. This small ship was used for testing the production of an improved-resolution mesh after executing a planned inspection. Photo Credit: Bordelon Marine, `http://www.bordelonmarine.com/terry.html`

(c) Photo and mesh from the *USCGC Seneca*. This is the only vessel that was visited for a second field test. The mesh, developed from the first test, was used to plan a coverage path that was executed during the second test.

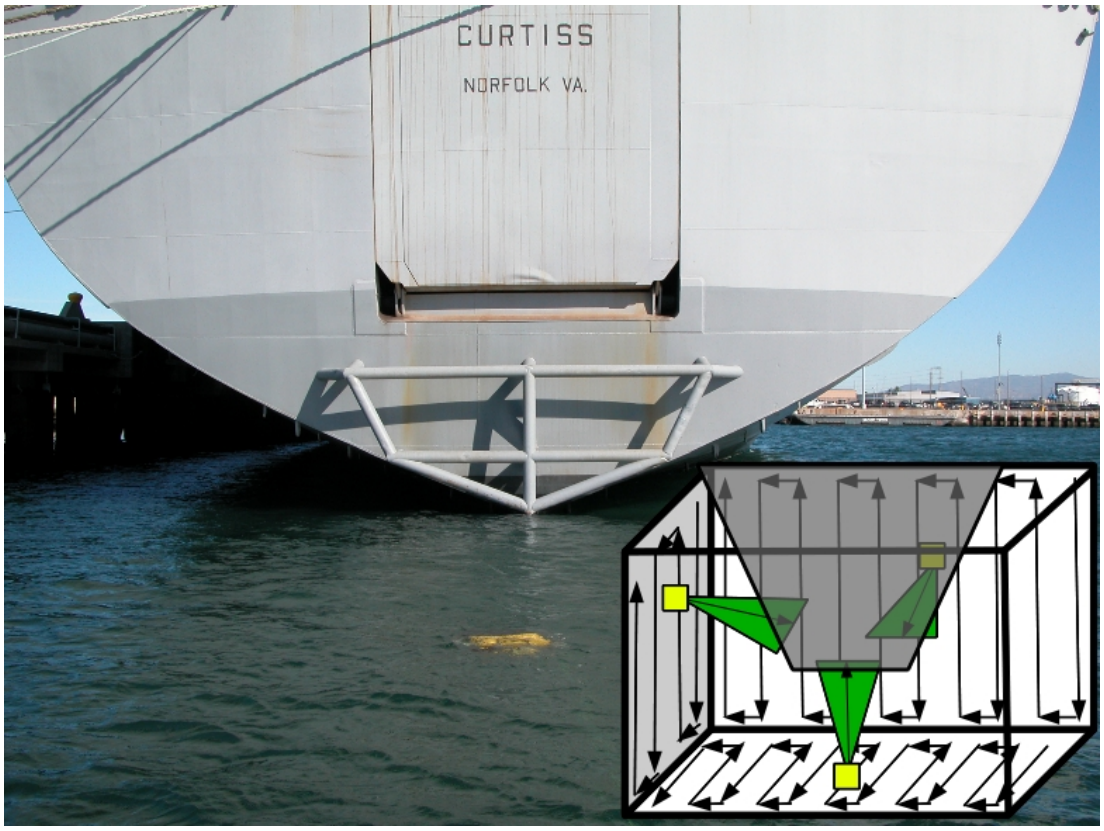
Figure 6-2: A summary of HAUV field experiments performed in support of coverage algorithm development and planned path execution, part two.

outliers [166], [78], and these provide a direct avenue for us to pursue our identification survey mesh model.

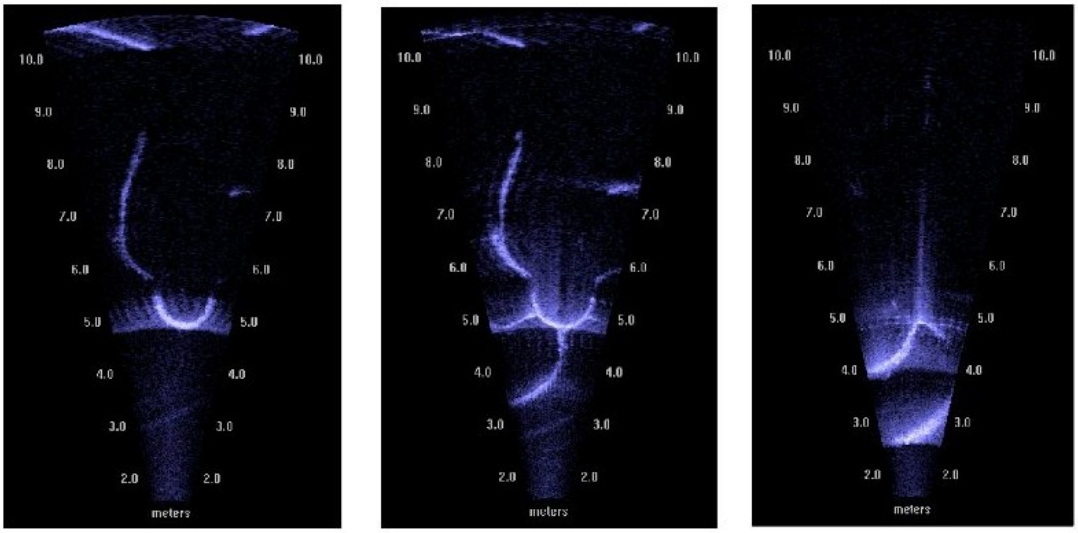
Figures 6-3 and 6-4 illustrate the execution and processing of an identification survey from start to finish. First, the HAUV traces out the walls of a safe bounding box that observes the stern from a distance known to be collision-free; thousands of DIDSON frames are collected along with navigation estimates. Evident in the sonar frames shown is the range noise which makes this modeling task difficult in comparison to laser-based modeling.

To transform a set of dense, raw-data point cloud slices into a 3D mesh reconstruction, we first apply a simple outlier filter to the individual sonar frames collected. All points of intensity greater than a specified threshold are introduced into a slice, and then each is referenced using the HAUV's seafloor-relative navigation. These steps are performed using software from SeeByte Ltd., and all remaining steps are performed using Meshlab [41]. These and other software tools used for processing and acquisition of data are described in Table B.3 in Appendix B. After assembling the sonar frames into a single point cloud, areas containing obvious noise and second returns are cropped out manually. The raw points are then sub-sampled using Poisson disk sampling [42], which draws random samples from the point cloud, separated by a specified minimum distance. The point cloud is typically reduced to about 10% of its original density, and it is then partitioned into separate component point clouds.

Partitions are selected based on the likelihood that they will yield individually well-formed surface reconstructions. Objects such as rudders, shafts, and propellers are thin structures that may not be captured in the final model without separate processing from the hull. Normal vectors are computed over the component point clouds, and some flat surfaces, for which only one of two sides was captured in the data, are duplicated. A point's normal vector is computed by applying principal component analysis to the point's k nearest neighbors, and the normal's direction is selected to locally maximize the consistency of vector orientation [74]. Both sub-sampling and estimation of normals are key steps in the processing sequence, found in practice to significantly impact the accuracy of the mesh [78]. Sub-sampling generates a low-

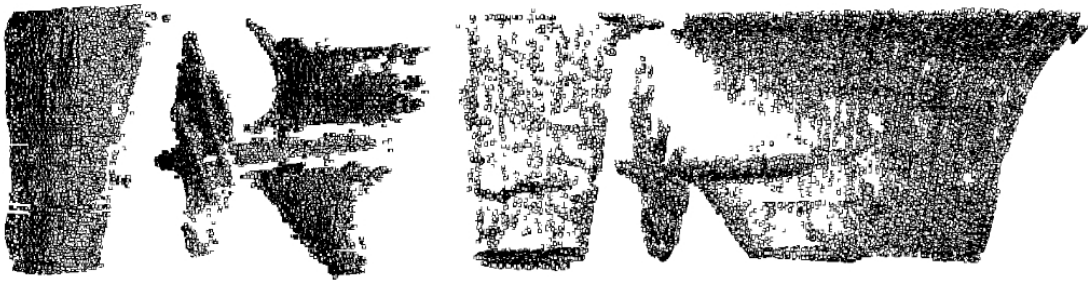


(a) Survey in progress at the *SS Curtiss*, with a diagram of the identification survey procedure.

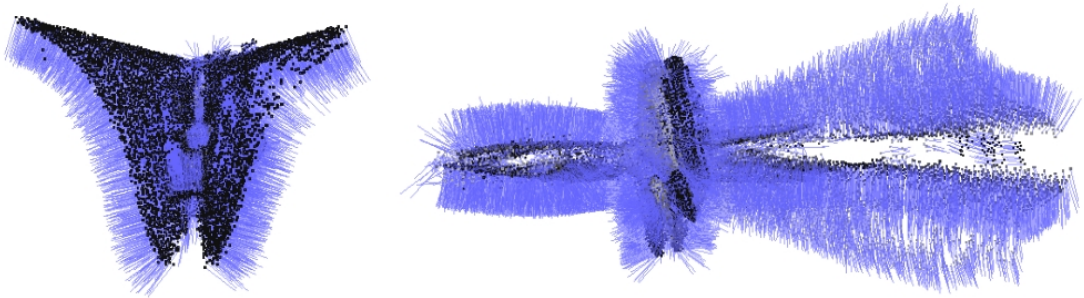


(b) Representative sonar frames from survey of *SS Curtiss* running gear, looking up at the shaft and propeller. Ranges are given in meters.

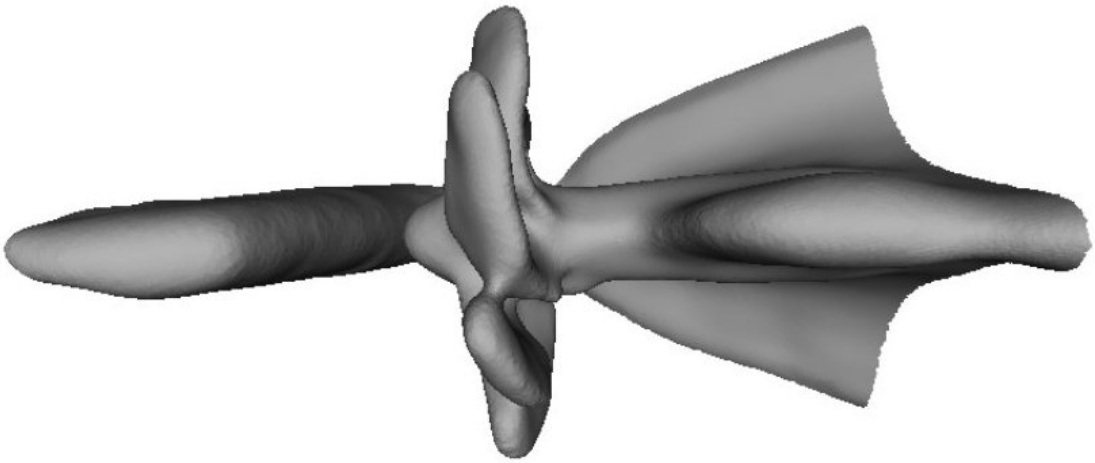
Figure 6-3: An overview of the identification survey procedure and the data obtained from it, part one.



(a) Raw-data point clouds obtained from the starboard-side wall and bottom wall of the identification survey, respectively.



(b) Merged, subsampled data is displayed with a vertex normal pointing outward from each individual point.



(c) A mesh model of *SS Curtiss* generated by applying the Poisson reconstruction algorithm to the point cloud of (b).

Figure 6-4: An overview of the identification survey procedure and the data obtained from it, part two.

density, evenly-distributed set of points, and normals aid in defining the curvature of the surface.

The Poisson surface reconstruction algorithm [91] is next applied to the oriented point clouds. Octree depth is selected to capture the detail of the ship structures without including excess roughness or curvature due to noise in the data. The component surfaces are merged back together, and a final Poisson surface reconstruction is computed over the components. If the mesh is used as a basis for high-resolution inspection planning, then it may be further subdivided to ensure the triangulation suits the granularity of the inspection task. We iteratively apply the Loop subdivision algorithm [112] for this purpose, which divides each triangle larger than a specified size into four subtriangles.

6.3 Execution of Planned Path at USCGC Seneca

An inspection survey was planned and executed at the *USCGC Seneca* using the HAUV, version HULS3. For coverage path planning, we used a triangle mesh model produced from a previous identification-survey field experiment at the *Seneca*; this model is shown in the computational results of Chapters 3-5 and is also pictured in Figure 6-2(c). Due to the time constraints of the field experiment, a section of the model representing one half of the ship's stern was used for planning, and the inspection was designed for sonar viewing at 1-4m range. Because version HULS3 of the HAUV was used, the DIDSON sonar could only be pitched from 0 to 90 degrees rather than the full range of ± 90 degrees assumed in the preceding computational results. The 100%-coverage inspection route is pictured in Figure 6-5. The tour was planned over two hours of computation on a Lenovo T400 laptop with a 2.53GHz Intel Centrino 2 processor and 3GB of RAM. After construction of an initial feasible solution using a redundancy-ten roadmap, which required approximately six minutes, the sampling-based improvement procedure was run for the remaining time. Seven view configurations were pruned during the improvement procedure, and the tour was shortened by twenty-seven meters.

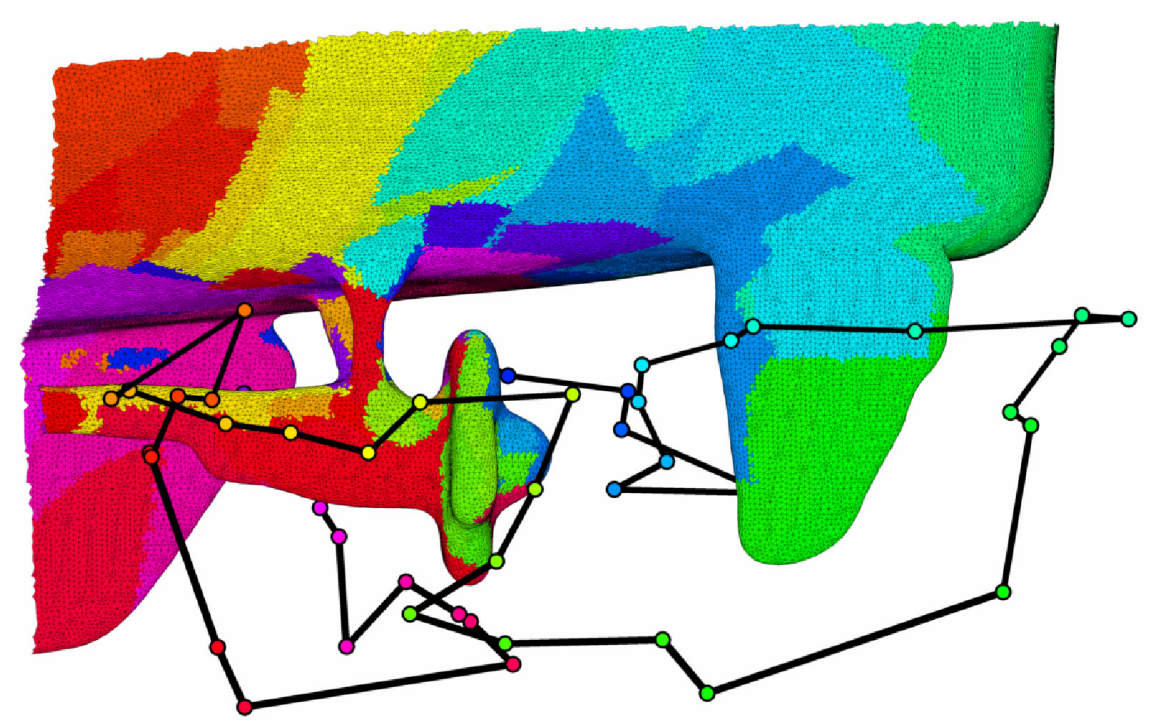
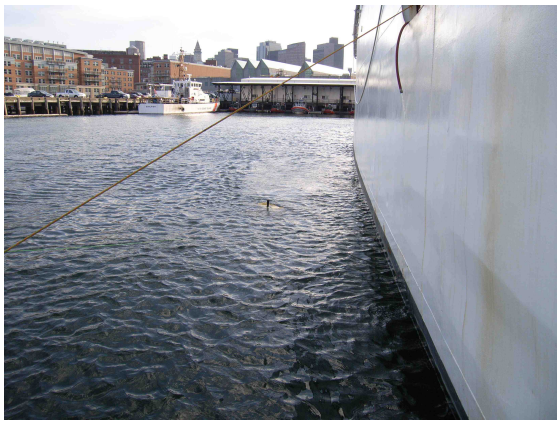


Figure 6-5: Planned route for inspection of the stern of the *USCGC Seneca*. The inspection is planned for sensing at 1-4 meter range, with a DIDSON pitch axis limited to motion between 0 and 90 degrees for use with HAUV version HULS3. The route is 54 meters in length and contains 53 planned views.

To allow in-water execution of the inspection, the waypoints were transformed into the seafloor-relative coordinate frame of each HAUV dive. This was achieved by performing a short, depth-varying identification survey along the side of the ship about seven meters away from the centerline, sufficient to obtain views of the running gear in the DIDSON data. While holding station after completion of this survey, the sonar data was registered to the *a priori* model, and the model, along with the planned waypoints, were transformed into the HAUV coordinate frame using the iterative cloest point (ICP) algorithm [18] with manual alignment as an initialization. Due to occasional mission aborts from the HAUV, the planned views were collected gradually, with some views repeated, over the course of five dives. The data obtained at the end of each dive was manually aligned with the *a priori* mesh model. Due to inaccuracies associated with the quality of the ICP registration, the HAUV's inability to hold station at a waypoint with decimeter precision (sometimes drifting upwards

(a) The HAUV, version HULS3, prior to deployment at the stern of the *Seneca*.



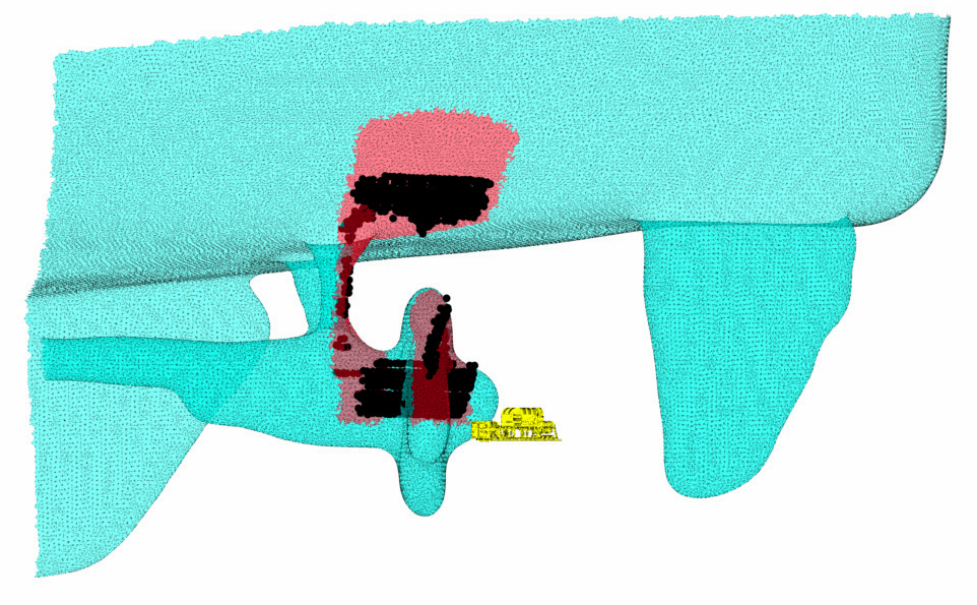
(b) The HAUV in Boston Harbor at the start of a *Seneca* survey.

Figure 6-6: Photos of operations at the *USCGC Seneca* during the February 2012 field experiment.

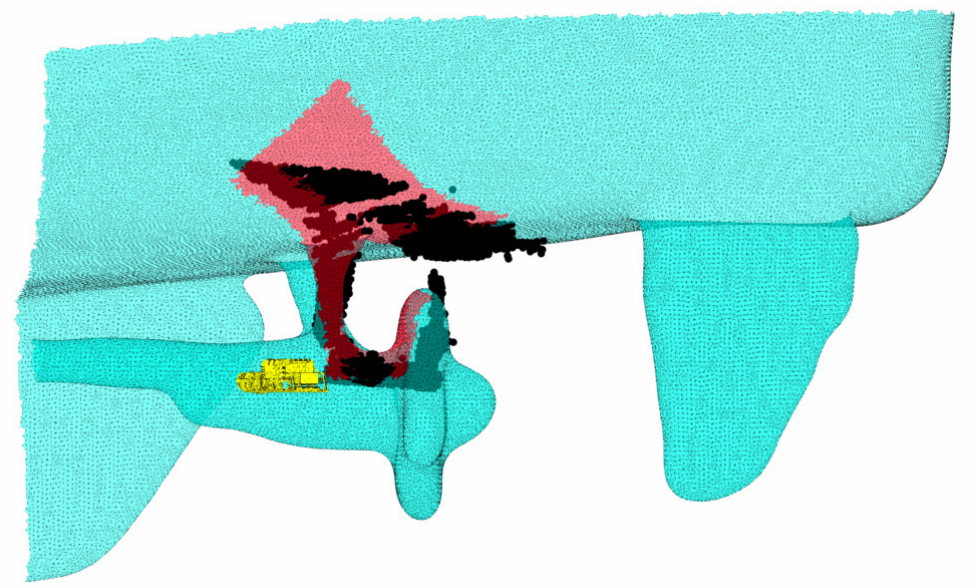
of a quarter meter while attempting to hold station), and accumulation of DVL drift during longer dives, not all planned views were well-matched with the data collected. Despite this, the collected range data was compared to the planned views of the mesh model and in many instances structural inaccuracies in the model could be deduced from the appearance of the data. Photos of the HAUV during the field exercises at the *USCGC Seneca* are given in Figure 6-6.

Due to a limited number of settings for DIDSON viewing window length, range data was recorded from 1.13-10.13 meters during the identification survey, and from 1.13-5.63 meters during the inspection survey. This latter setting allowed the planned sensor observations to be collected with some additional overlap to spare. The next available sensor setting, which reduces the maximum viewing range to 3.38 meters, was inadequate for obtaining the four-meter planned views. Because the DIDSON pitch axis is located forward of the point where acoustic scans originate, each recorded scan actually begins behind the origin of the composite "sensing volume" used for planning the views of an inspection. Sampled scans that exceed the range of the planned views aid in compensating for the offset of the DIDSON pitch axis.

Individual views obtained from the inspection tour are given in Figures 6-7 and 6-8. The points plotted reflect range data that has been processed using an outlier filter,

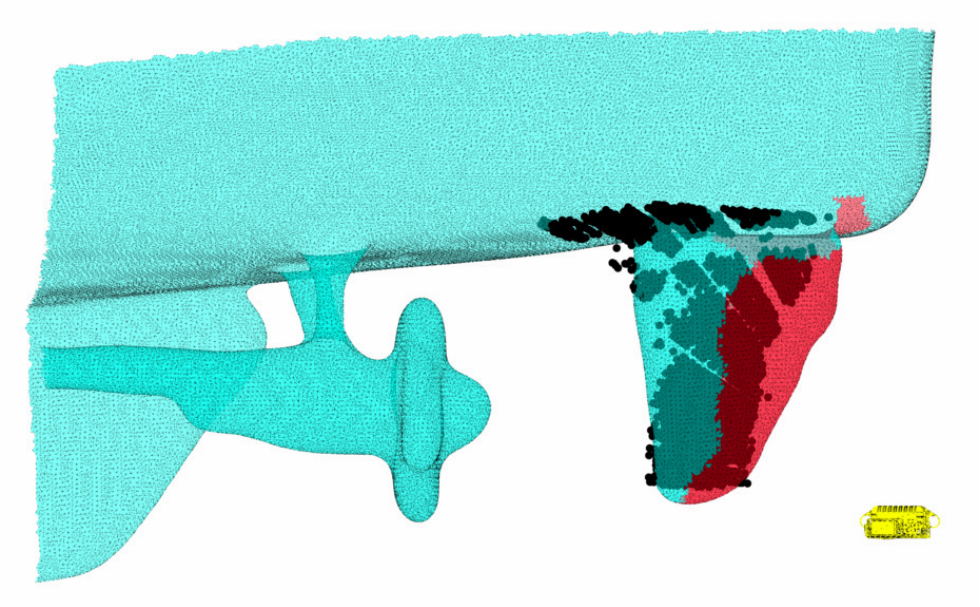


(a) View of propeller, showing differences in blade thickness and hull curvature.

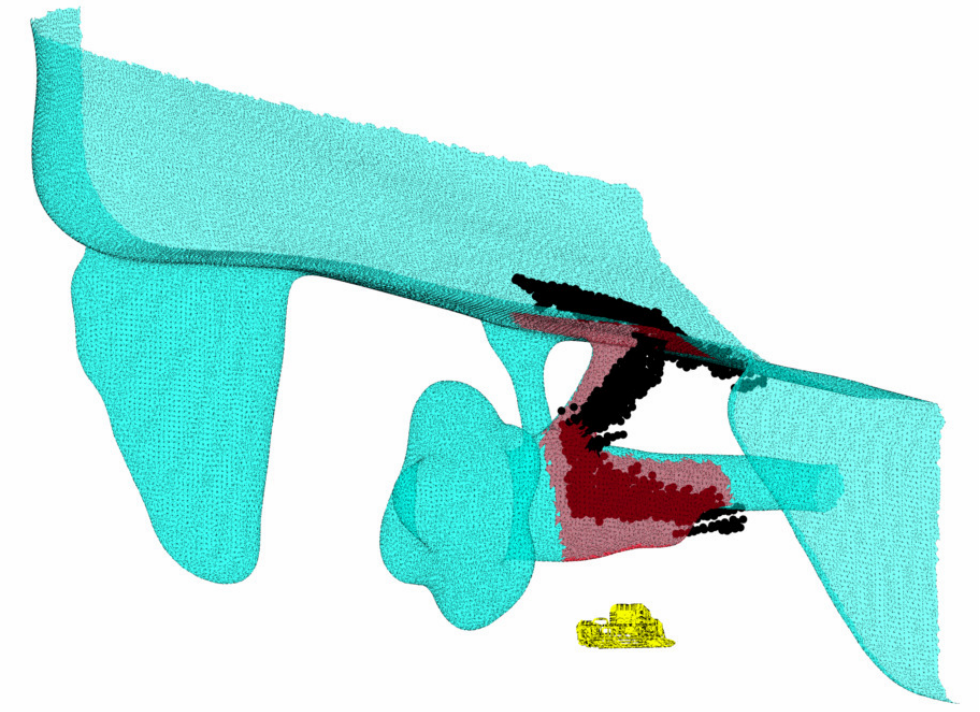


(b) View of propeller, showing differences in blade thickness and hull curvature.

Figure 6-7: Selected waypoints from the planned inspection of the *Seneca* are illustrated, comparing the planned view (red) with the obtained view (black), part one. The mesh has been rendered with some transparency to grant visibility of black sensor data lying within its boundaries.



(a) View showing differences in rudder geometry.



(b) View showing differences in geometry of the inboard propeller strut.

Figure 6-8: Selected waypoints from the planned inspection of the *Seneca* are illustrated, comparing the planned view (red) with the obtained view (black), part two. The mesh has been rendered with some transparency to grant visibility of black sensor data lying within its boundaries.

Figure 6-9: Composite point cloud showing the data collected from the planned inspection of the *USCGC Seneca* outboard stern. The alignment of each point's outward-facing normal vector with the camera viewpoint depicted in this image is reflected by the shading of each point. Points with normals directed toward the camera are light gray in color, and points with normals directed away from the camera are dark gray in color.

subsampled to about 10% of the original density of points, and manually filtered to eliminate obvious noise and second returns. These views reveal aspects of the ship's true structure that are absent from the *a priori* model used for path planning. It is clear from the data in Figure 6-7 that the mesh model's propeller blades are thicker than those of the true vessel, and the data in Figure 6-8 reveals inaccuracies in the rudder geometry and the angular orientation of the inboard propeller strut.

A composite point cloud showing all views collected over the course of the inspection is given in Figure 6-9. A normal vector was computed for each point using the same method described in Section 6.2 for the generation of a watertight mesh. It is evident from the shading of the points, which depicts the orientation of the normal vectors, that some gaps in coverage exist. Some of the propeller blades were observed from a single side only, and the shaft is not fully covered below the outboard strut. Other gaps, made evident by the presence of white spaces between points, exist due to the inability of the outlier filter to capture all of the range returns from specific sonar frames.

Despite these gaps in coverage, the data obtained from the close-range survey of the *Seneca* running gear permitted an improved triangle mesh model to be con-

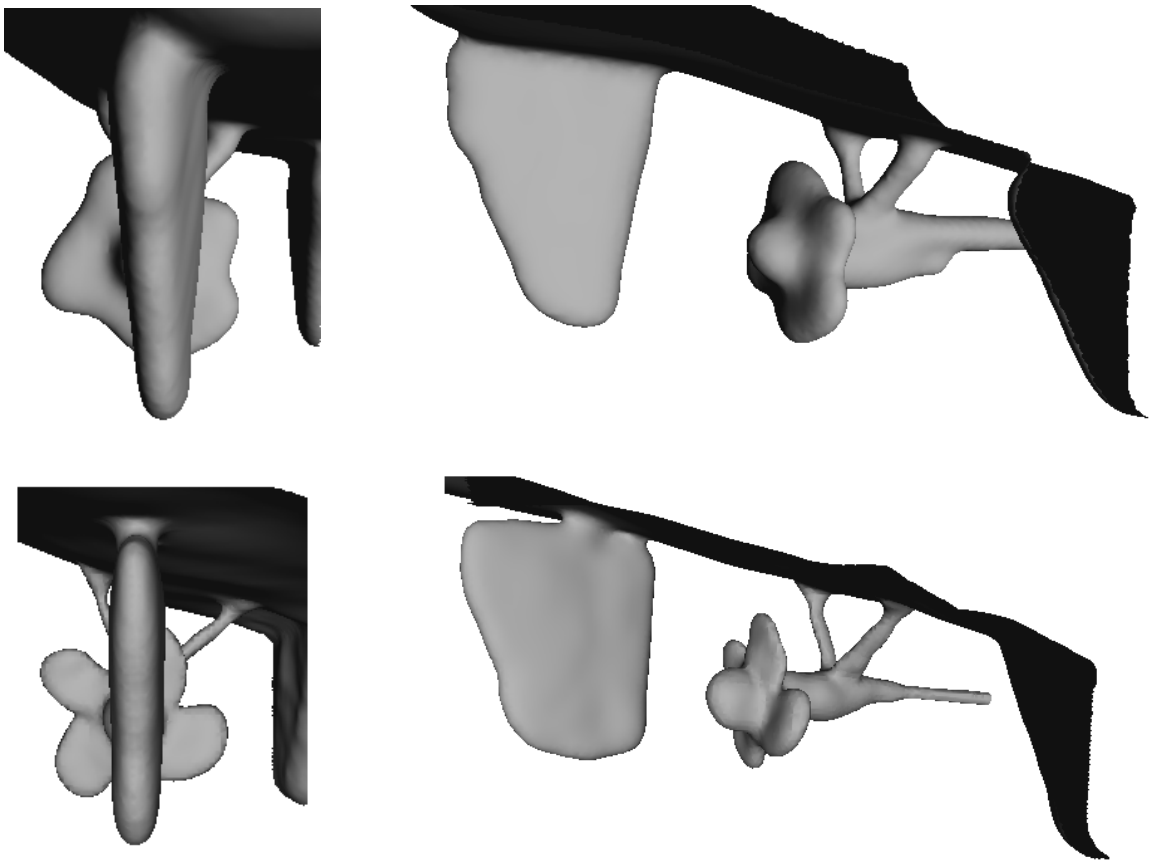


Figure 6-10: Comparison of the original, low-resolution *Seneca* mesh, at top, with a mesh obtained from the planned high-resolution inspection route, at bottom.

structed using the manual processing method of Section 6.2, with interpolation used where necessary to fill gaps in coverage. This model is compared with the original, low-resolution *a priori* mesh model in Figure 6-10. The range data obtained from the planned inspection was sufficient to resolve each individual propeller blade, the actuated post attaching the rudder to the hull, and the orientation of the propeller struts, features that were unresolved or incorrect in the *a priori* mesh model.

6.4 Results from Laser-Equipped Gantry System

Despite of the achievements of the *USCGC Seneca* field tests, full sensor coverage was not obtained, many aspects of the data processing were performed manually, and the navigation and ranging precision of the HAUV and DIDSON didn't quite

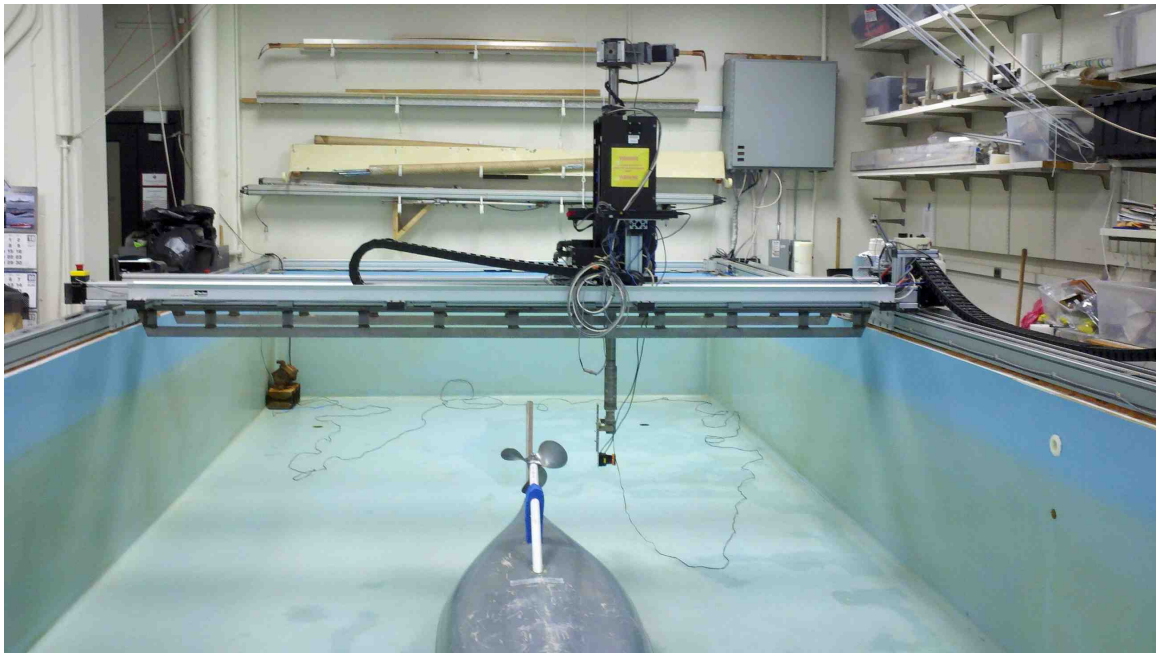


Figure 6-11: Photo of experimental apparatus used for coverage path planning laboratory experiment. The four degree-of-freedom robotic gantry is pictured, with a Hokuyo UTM-30LX laser mounted at the tip of the end effector.

match the decimeter resolution desired for mine detection. The precision of acoustic range sensors and the maneuvering and control capabilities of the HAUV have been and will continue to be improved over time, but for the purposes of algorithm validation, we also plan and execute a path to support laser-based range sensing in air. The experiment presented in this section uses a robotic gantry system capable of centimeter-precision translation along three axes and degree-precision rotation about a single yaw axis. The gantry has been used previously in underwater sonar navigation experiments [108], but we operate it in a dry tank, mounting a Hokuyo UTM-30LX laser rangefinder at the tip of the end effector. The testing tank in which the gantry operates is ten meters long, three meters wide, and one meter deep.

A kayak 3.6 meters in length and 0.7 meters wide was outfitted with an artificial set of running gear to serve as an approximate one-tenth-scale mockup of the *USCGC Seneca*. A thirty-centimeter-wide rudder and thirty-centimeter-diameter propeller are nearly an order of magnitude smaller than the 2.5-meter-wide rudder and the 2.5-meter-diameter propeller of the *Seneca*. The mockup inspection experiment, including

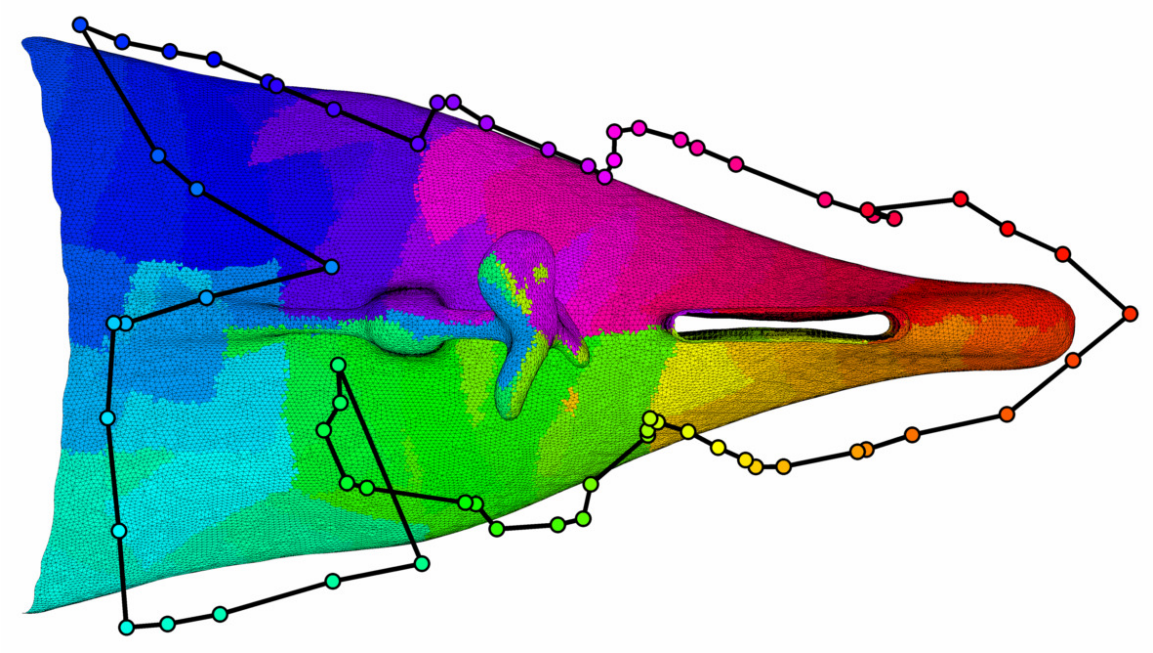


Figure 6-12: Planned route for inspection of the modified kayak hull. The inspection is planned for sensing at 0.1-0.3 m range, with the equivalent of DIDSON pitch between 0 and 180 degrees and robot-relative heading of ±15 degrees. The route is 6.8 meters in length and contains 66 planned views.

the gantry, laser, and modified kayak, is pictured in Figure 6-11. To achieve a tenth-scale equivalent of DIDSON sensing with the laser, the sensor is turned on its side and rotated through a thirty-degree span of heading angles at each planned sensor view. Each individual scan spans 180 degrees in pitch, equivalent to the available range of sonar pitch angles on the HAUV, model 1B. This gives a volumetric sensor footprint equivalent to that of the DIDSON when the sonar, with thirty-degree-wide individual range scans, is pitched through 180 degrees. To give appropriate scaled-down ranging, laser views are planned for observation at 0.1-0.3 meters only. This is a tenth-scale equivalent of the ranges assumed in the computuational studies of Chapters 3-5.

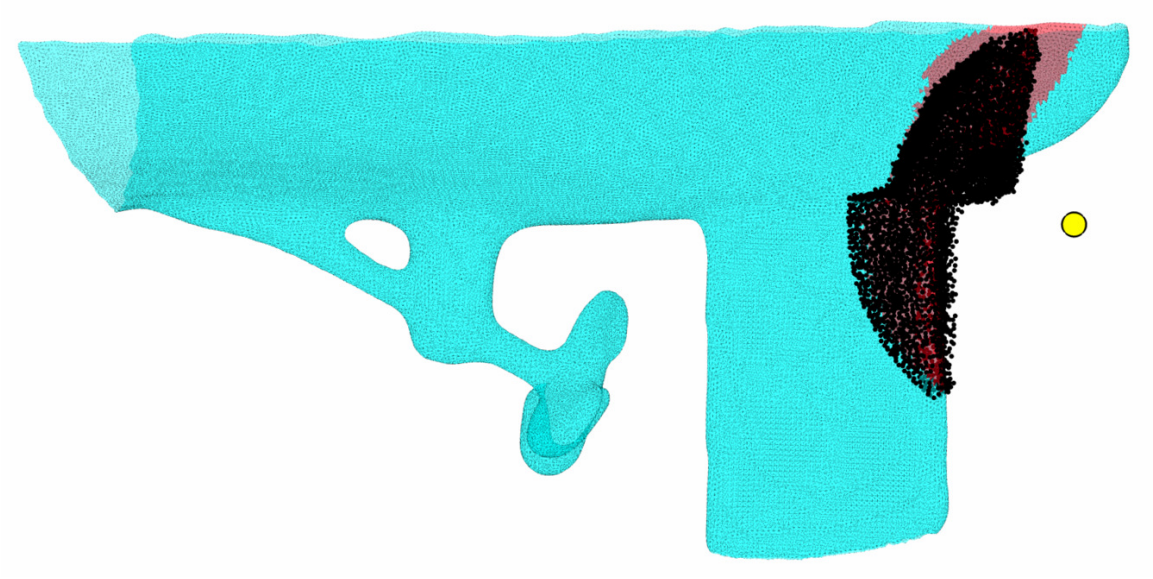
An identification survey was performed with the gantry to generate an *a priori* mesh model of the kayak. Views were collected at thirty-two individual waypoints set back one half-meter from the kayak centerline to generate this model. The positioning of the gantry is accurate enough that manual alignment of the sensor data was not required, and vertex normals for the point cloud were determined by the orientation

of the laser beam corresponding each individual data point. The point cloud did require subdivision into a few separate components, however, to yield a well-formed watertight mesh. Due to the restricted range of motion in the gantry's depth-wise direction, limited to 0.3 meters, the top surface of the kayak rudder was cropped from the model so infeasible coverage would not be required. A cylindrical CAD model was used to represent the gantry end effector for the purposes of collision-free path planning.

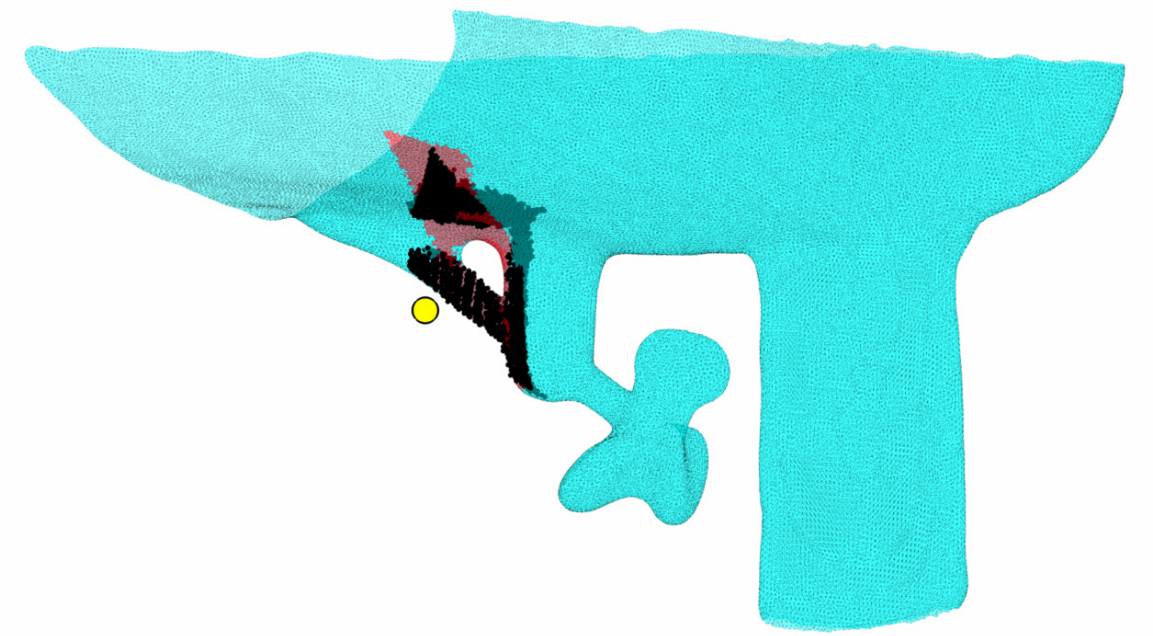
The inspection survey planned for 100% high-resolution coverage of the kayak stern is pictured in Figure 6-12. The tour was planned over two hours of computation on a Lenovo T400 laptop with a 2.53GHz Intel Centrino 2 processor and 3GB of RAM. After construction of an initial feasible tour using a redundancy-ten roadmap, which required approximately three minutes, the sampling-based improvement procedure was run for the remaining time. Thirteen view configurations were pruned during the improvement procedure, and the tour was shortened by 4.6 meters.

The entire set of procedures run for the identification and inspection surveys remained referenced in the same coordinate frame throughout, and no manual alignment of data was required. A small margin of safety was used in implementing the planned views; range data was sampled that overlapped, by two centimeters each, the maximum and minimum planned viewing ranges. Five additional degrees of heading were added to each end of the sampled volume to account for any heading angle biases remaining after calibration. Data collected during the planned close-range inspection was manually filtered to remove noise and any returns from the testing tank, the gantry, and other surrounding structures.

A selection of sensor views from the inspection is depicted in Figures 6-13 and 6-14. The points plotted reflect range data that has been manually filtered and subsampled to about 10% of the original density of points. These views reveal structures very similar to those in the plan, with a few small discrepancies between the *a priori* mesh model and the actual structure. It is clear from the data that the curvature of the true kayak hullform is slightly different from that of the model, and that the areas where the shaft and the shaft bearing meet the hull are slightly distorted in the model.

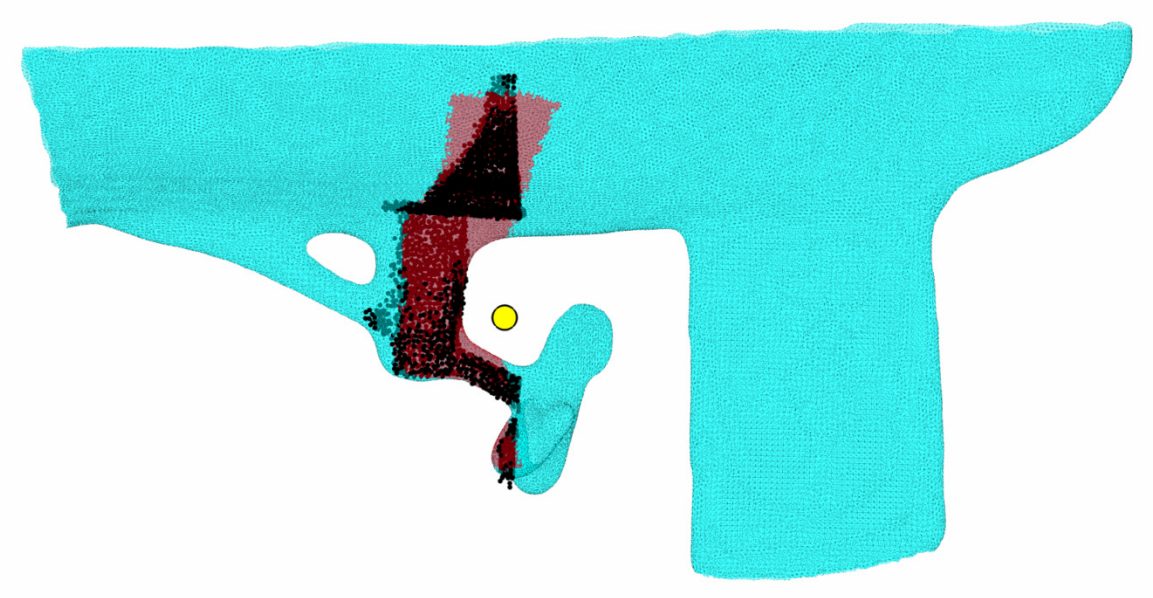


(a) View of rudder, showing some differences in hull curvature.

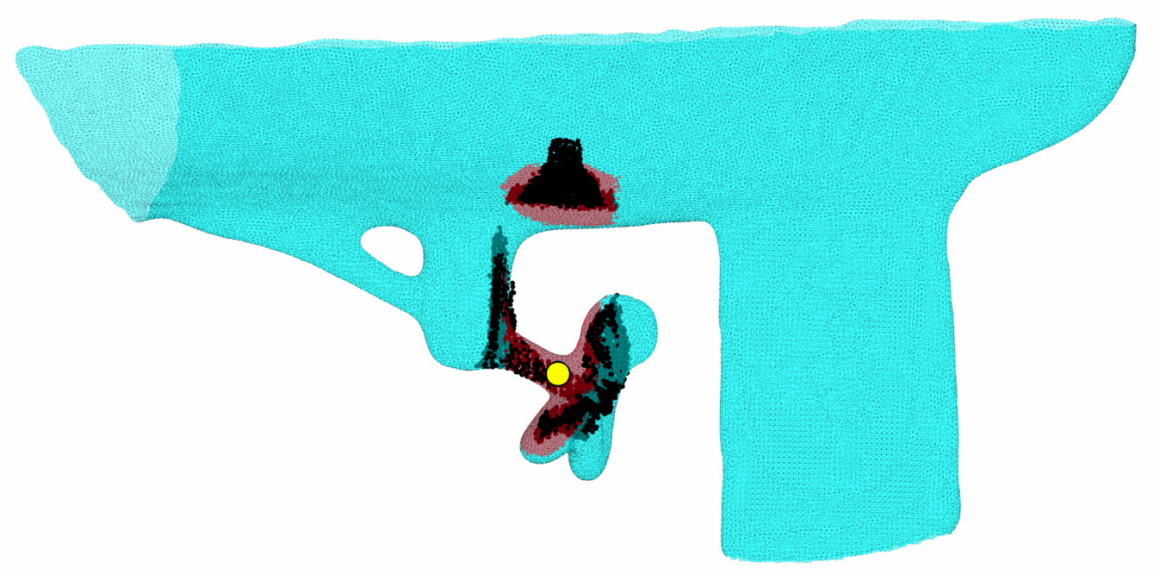


(b) View of shaft, showing a larger gap between shaft and hull than modeled.

Figure 6-13: Selected waypoints from the planned inspection of the modified kayak are illustrated, comparing the planned view with the obtained view, part one. The mesh has been rendered with some transparency to grant visibility of black sensor data lying within its boundaries.



(a) View of shaft, bearing, and hull, showing differences in hull curvature.



(b) View propeller, shaft, and hull, showing differences in propeller geometry.

Figure 6-14: Selected waypoints from the planned inspection of the modified kayak are illustrated, comparing the planned view with the obtained view, part two. The mesh has been rendered with some transparency to grant visibility of black sensor data lying within its boundaries.

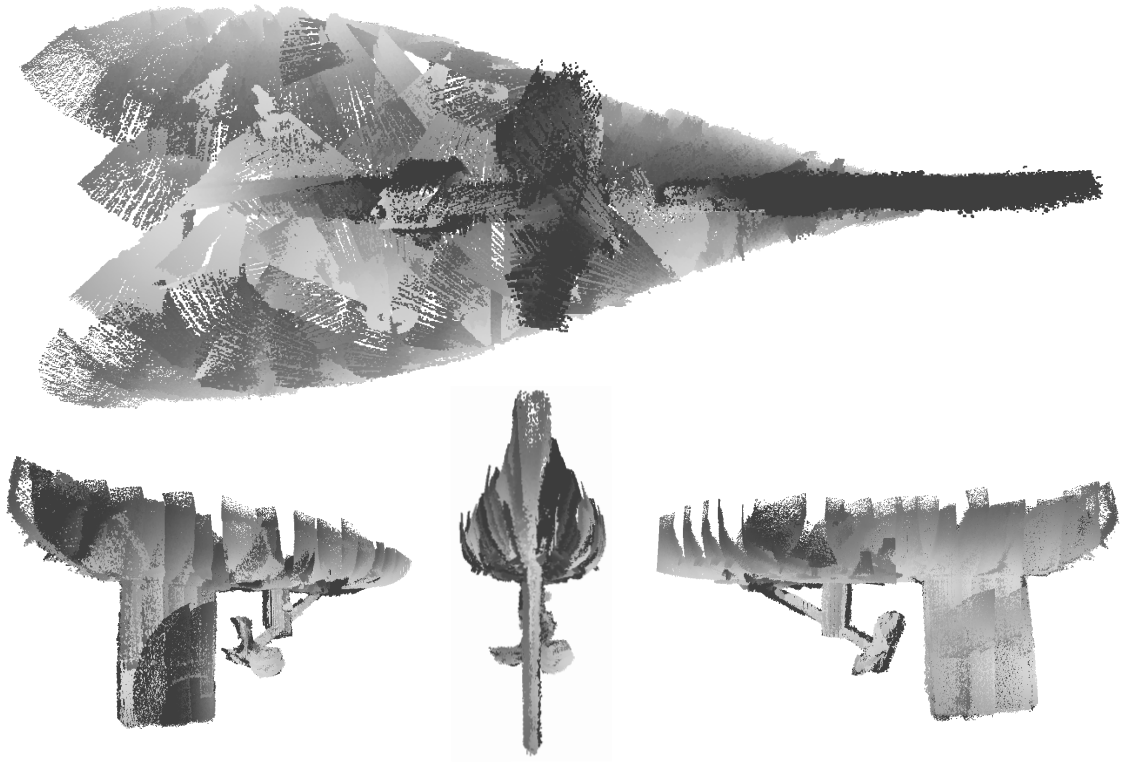


Figure 6-15: Composite point cloud showing the data collected from the planned inspection of the modified kayak. Noise has been manually filtered from the point cloud. Unlike the data displayed in Figure 6-9, this data is derived entirely from the gantry coordinate frame and has not been rotated or translated. The alignment of each point's outward-facing normal vector with the camera viewpoint of each image is reflected by the shading used. Points with normals directed toward the camera are light gray in color, and points with normals directed away from the camera are dark gray in color.

A composite point cloud showing all views collected over the course of the inspection is given in Figure 6-15. It is evident that a complex patchwork of views was required to derive a full-coverage point cloud at such limited viewing range; this is highlighted by the vertex normals specific to each view. A normal vector was computed for each point using the orientation of its corresponding laser beam, as used in the production of the *a priori* kayak mesh. Some small gaps in coverage do exist, evidenced by the white spaces between scans, that result from discrepancies in kayak hull curvature between the *a priori* model and the true structure.

The sub-centimeter resolution of the laser and precision positioning of the gantry

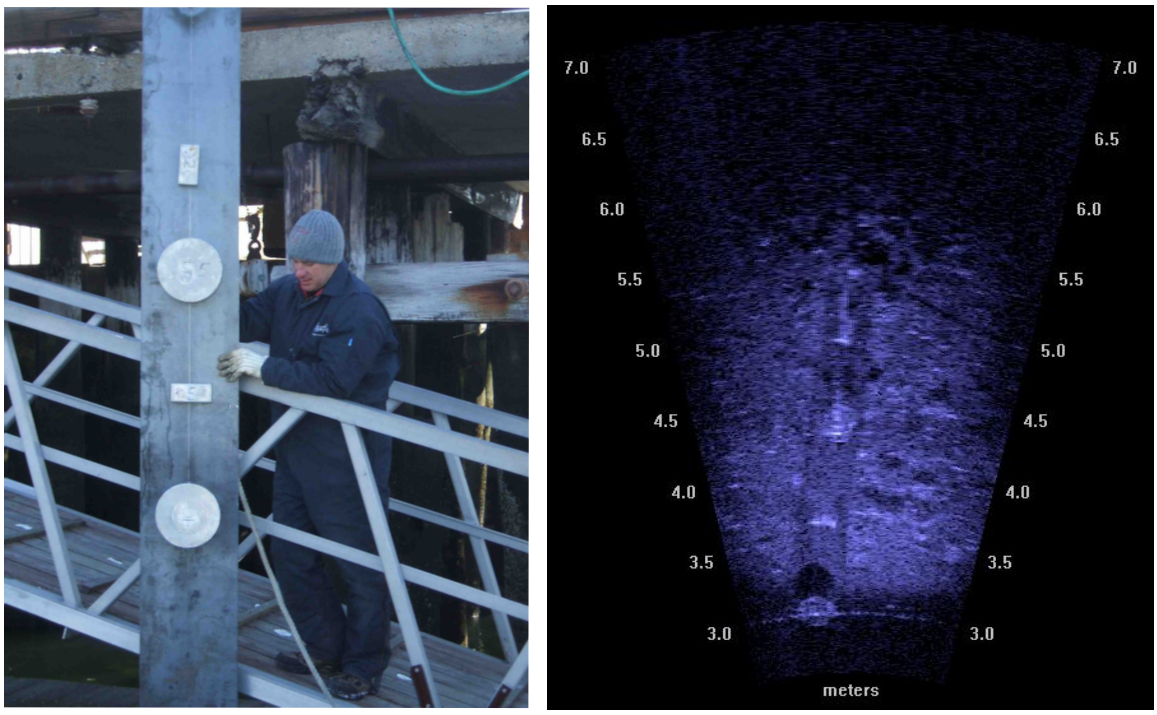


Figure 6-16: Aluminum targets used for HAUV mine detection exercises, including their appearance on the seafloor as viewed by the DIDSON in 2D imaging mode.

can be exploited further to detect mine-like objects on the surface of the kayak. During some of our ship hull inspection field exercises, decimeter-scale targets have been placed on the hulls to foster work on mine detection and classification. These training targets, pictured in Figure 6-16, have been detected consistently using the DIDSON in 2D imaging mode, in which the distinctive outlines of the targets can be used to automate classification. Detection of these targets in profiling-mode range scans, however, has proven a difficult task. The extent to which these targets protrude from the ship hull's surface rivals the resolution of the range scan and the precision of vehicle maneuvering. This is not the case for the gantry system, and we test this capability by planting two scaled-down mine-like objects on the kayak: a bolt protruding from the propeller, and a cylindrical cap protruding from the shaft bearing, both of which are pictured in Figure 6-17 planted on the kayak.

The bolt is 1.3 cm in diameter, and protrudes 2 cm from the surface of the propeller. The cap is 2.5 cm in diameter, and protrudes 2 cm from the surface of the shaft bearing. The length of the brick-shaped target in Figure 6-16 is ten times the

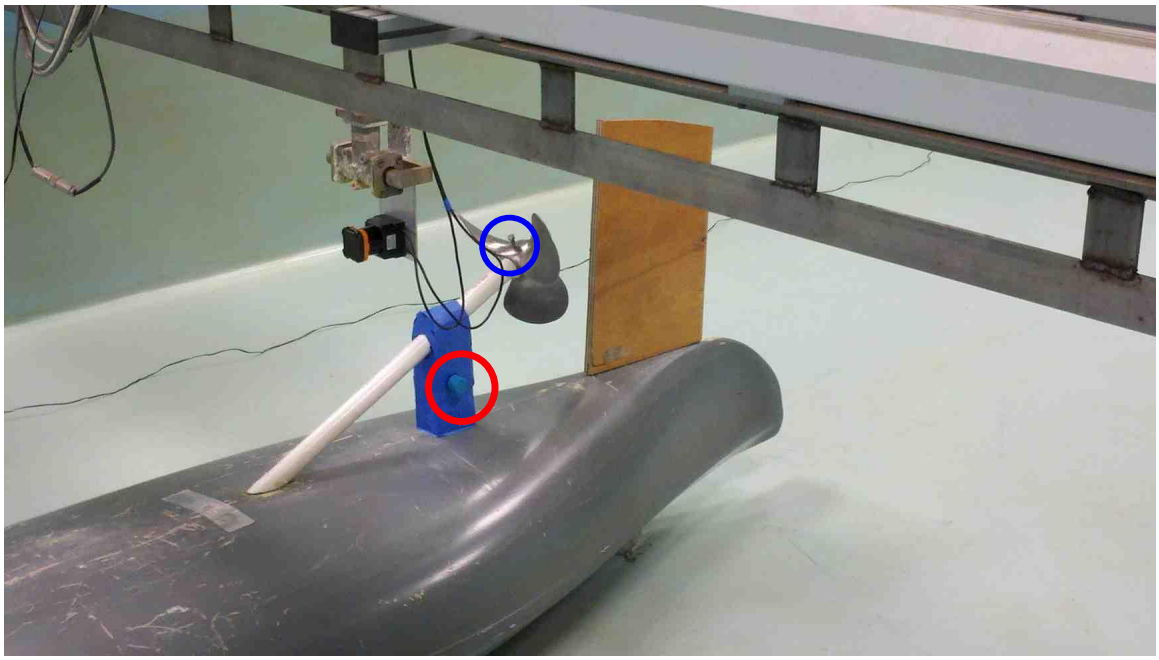


Figure 6-17: Small-scale targets placed on the modified kayak. A plastic cylindrical cap, circled in red, is attached to the shaft bearing. A steel bolt, circled in blue, protrudes from the propeller.

diameter of the bolt, and the diameter of the cake-shaped target in Figure 6-16 is ten times the diameter of the cap. The height of the "cake" is more than five times the height of the bolt and cap. Both of these targets emerged successfully in the range scans of the planned coverage inspection. They are difficult to detect in the composite point cloud of Figure 6-15, but they are evident in individual sensor views. Views that contained compelling imagery of the bolt and cap targets are displayed in Figure 6-18.

6.5 Summary

This chapter detailed the experimental outcomes achieved in developing and testing the coverage path planning algorithms of this thesis. Quality *a priori* mesh models of complex structures were essential in the development and refinement of the coverage algorithms; these were produced over a series of field tests performed on six different ocean vessels. Tests were also needed to implement the high-resolution inspection

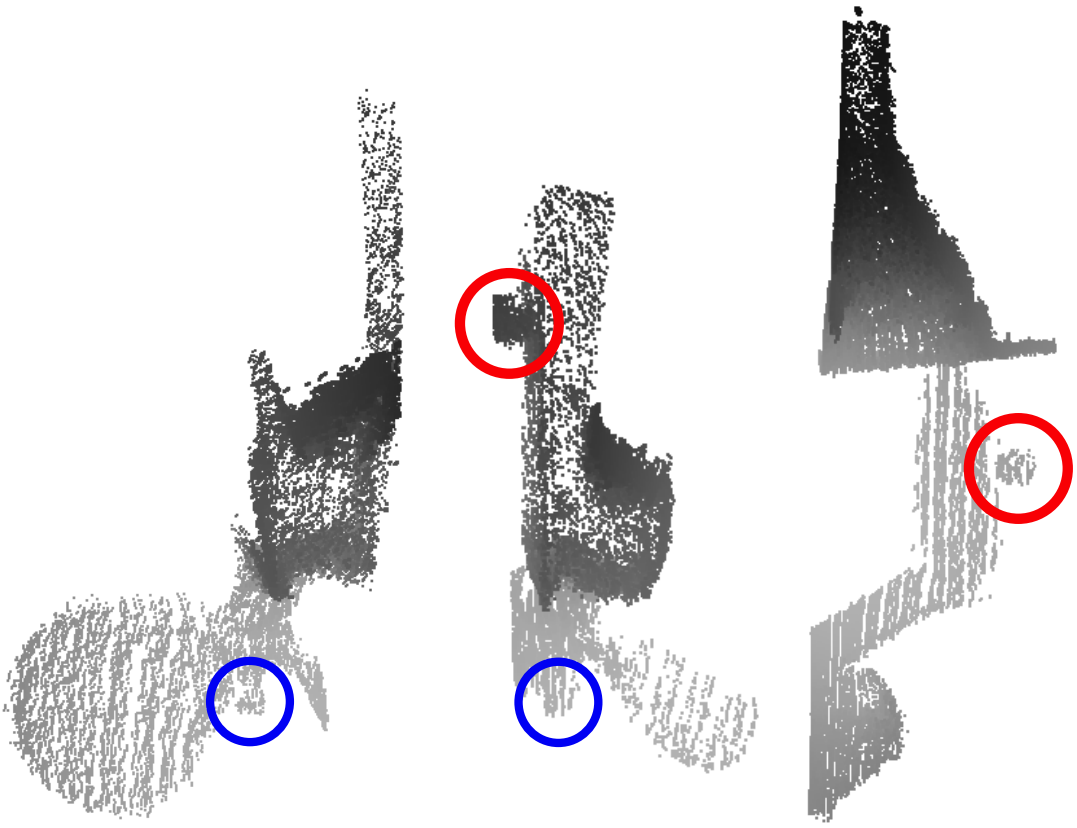


Figure 6-18: Views obtained during the planned inspection of the kayak that contain imagery of the targets. The colors marking the targets correspond to the colors in Figure 6-17.

survey paths planned using these models; this was accomplished at the *USCGC Seneca* and in a laboratory experiment using a robotic gantry and a mock ship hull. The *Seneca* experiment used a planned inspection route to improve the quality of the mesh model, and the gantry experiment validated the use of the algorithms for mine detection applications. When the precision of a robot's navigation and sensor suit the granularity of the inspection task, successful detection can be achieved. The final data product from the kayak inspection, the composite patchwork point cloud of Figure 6-15, embodies the uniqueness and the benefit of sampling-based coverage path planning: it allows an expansive structure to be covered by a near-sighted sensor, piece by piece, even when many pieces are required.

Chapter 7

Conclusions

In this concluding chapter, we summarize the contributions of this thesis, and offer a few comments on compelling areas for future work. We emphasize the need to adapt the algorithms of this thesis into real-time, reactive capabilities that can bring the HAUV closer to full, robust autonomy.

7.1 Review of Contributions

Three algorithms were presented in this thesis that advance the state of the art in path planning under coverage constraints. The redundant roadmap algorithm is proposed for planning feasible coverage paths. Building on the foundational sampling-based method of Danner and Kavraki used to cover simple 3D polyhedra [47], our algorithm implementation uses modern data structures and combinatorial optimization techniques to plan over 3D triangle mesh models comprised of hundreds of thousands of geometric primitives. Our algorithm also builds on the work of Gonzalez-Baños and Latombe in the area of sampling-based view planning [66], [67]. Two view planning strategies are proposed in their work: a method that constructs an incremental set cover and a method that solves a set cover in batch. They developed the former method into a tunable-quality view planning algorithm, and we endow the latter method with a similar practical, tunable functionality that it previously lacked. We have also shown that such a method, when used to plan views over complex 3D struc-

tures, is more computationally efficient, especially when high-quality solutions are desired.

Our sampling-based improvement algorithm can be used to simplify and shorten an initial, feasible coverage tour. This algorithm relies on a subroutine with a simple purpose: reducing the distance between a view configuration and its two immediate neighbors in the tour. Iteratively solving this problem for different views allows subtantial improvements to be made to an inspection route while avoiding the repeated invocation of an NP-hard reordering problem. The $\text{RRT}^*_{||}$ algorithm, a simple extension of RRT* [86], fits nicely in this role, and endows this subroutine with the properties of probabilistic completeness and asymptotic optimality. A heuristic speed-up for problems with dense sensor views allows many configurations to be brought into perfect alignment over local portions of the tour. This improvement procedure is an entirely new contribution to planning under coverage constraints, adapting the established tools for iterative smoothing of standard, point-to-point paths.

Our sampling-based sweep-path algorithm can be used to create an inspection route with regularized, rectangular structure. When the clarity and continuity of data is highly prioritized, or if the inspection route is to be monitored and analyzed by a human opearator, regularity may be a desirable property, and worth a sacrifice in the overall duration of the inspection. Our procedure, which employs structure segmentation and plans a sweep path for every segment, allows the simple components of a structure to be covered using simple paths. Unlike prior methods that plan back-and-forth sweep paths [3], [12], however, our algorithm does not demand 100% coverage from the regularized paths. Instead, the occluded areas of a structure that elude the sweep paths are covered using randomized, targeted view configurations. This algorithm, rather than constructing an explicit cellular decomposition, pioneers a unique sampling procedure that aims to "seed" each new sweep path in a location of maximal coverage, and does so with appealing convergence properties.

In general, we have established throughout this work a methodology for analyzing the probabilistic completeness of a sampling-based coverage algorithm. By unifying techniques from the analyses of the PRM [88] and RRT [105] with the language of

set systems [80], to express the coverage topology of the samples, we have established sharply decreasing exponential bounds governing the convergence of the sampling-based subroutines presented in this work.

Finally, the algorithms of this thesis were applied in support of autonomous, in-water ship hull inspection. *A priori* mesh models were generated over a series of field tests on six vessels, and these models have been instrumental in algorithm development. Close-range coverage routes planned using the redundant roadmap algorithm, coupled with the improvement algorithm, have been deployed on the *UCGCG Seneca* using the HAUV and on a mock ship hull using a laser-equipped gantry robot. The former experiment achieved an improved-resolution mesh model for the *Seneca*, and the latter experiment a high-quality composite point cloud of resolution sufficient to detect mine-like objects.

7.2 Reflections on Work Completed

This thesis has emphasized autonomy at a high level: path planning under task-specific constraints. The design and analysis of algorithms to satisfy these constraints has relied on many assumptions about the state of the robot and the workspace in which it operates. Although these assumptions hold under ideal operating conditions, it is rare that all conditions are ideal simultaneously. Learning when, why, and how these assumptions break in the physical world has been a humbling experience, and has instilled in the author a profound respect for the complexity of fielding of an autonomous system.

A well-designed path alone cannot overcome an ocean current that exceeds the robot's thrust capability, the unexpected movement of the structure being inspected, nor the structure's sudden high-flow-rate suction and discharge of water into the surrounding area. A fixed-orientation geometric path is less meaningful when dramatic roll and pitch result from an imperfect pairing of flotation with the salinity of the environment. A pre-planned path in general may be of little use when it is misaligned with the robot's true coordinate frame, the navigation sensors drift, or the robot can-

not stabilize at its intended destination, either due to exogenous disturbances, an improperly tuned feedback controller, or a poor thruster-to-body-force mapping. At a more fundamental level, a fiber-optic tether snare, a disrupted DVL beam return, a saturated communication network, a failed circuit, an overheated sensor, or a ground fault can prohibit even the lowest level of control from functioning properly. These non-ideal conditions are not purely hypothetical, they have all occurred in the course of operating the HAUV.

Although this thesis has not formally addressed vehicle design, hardware development, state estimation, or low-level feedback control, the path planner developed in this thesis interacted with a multitude of other software and hardware modules, some of which are described above. Understanding the complexity of these interactions, and the robot's interactions with its physical environment, were crucial to the execution of a planned path in the real world.

7.3 Compelling Areas for Future Work

An important area of ongoing work is the integration of our planning algorithms with tools that enhance autonomy during HAUV operations. Over the relatively flat, forward areas of a ship, where the HAUV navigates relative to the hull itself, real-time localization uses imaging sonar and camera-based registration to achieve high accuracy navigation over extended periods of time [76]. In the complex areas, however, where the HAUV must navigate relative to the seafloor, vision-based navigation correction has not yet been developed. A planned path that is run open-loop will drift in accuracy over time, and localization using sonar range scans could mitigate this problem.

In addition to localization, real-time mapping is needed to deduce the location of surrounding obstacles and ship structures while the HAUV operates at the stern. This would allow reactive measures that deviate from an existing plan, or re-plan on the fly. Such a framework may be used to keep track of the areas of the ship that have been inspected, adding views to the plan adaptively to close any unplanned gaps in

coverage. The algorithms of this thesis could be employed as "anytime" algorithms to design and improve paths in real-time for coverage of the remaining structures. The type of high-performance integrated localization, mapping, and planning required for such tasks has been implemented in 2D workspaces using road vehicles [107] and in 3D workspaces with micro aerial vehicles [144].

This level of performance, however, cannot be obtained as easily underwater. Cameras offer limited range due to turbidity, and the DIDSON sonar in profiling mode has a limited field of view, lower-precision range sensing than the lasers used in aerial and ground deployments, and troublesome second returns. A rigorous study of filtering techniques for processing acoustic range scans could open the gateway to high-level improvements in autonomy. Techniques such as the curvelet transform [147], which is widely successful in denoising and recovery of edges and curvilinear image features, may offer possibilities for improved performance.

Finally, for the inspection of colossal structures, such as container ships and aircraft carriers, it is likely that a team of vehicles will be needed to complete a full-coverage mission. Divers work in teams to cover these structures, and if comparable mission duration is to be achieved using robots, a multi-vehicle deployment is necessary. A number of algorithms have been proposed for multi-robot coverage planning using 2D and 2.5D methodologies [99], [131], [49], [32], [58]. Strategies used by these algorithms to partition a coverage task among a team of robots may be suitable for adaptation to the complex 3D structures explored in this thesis.

7.4 Concluding Remarks

A fully autonomous inspection of a known, expansive structure will benefit from principled, model-based path planning, combined with active perception [14] to modify the plan when unexpected events occur. We have advanced several steps closer toward this capability by contributing to the former of these two areas, developing sampling-based methods that construct high-quality routes for covering large structures with near-sighted sensors. These algorithms are rooted in new insights on how to

efficiently search a C-Space that is embedded not only with obstacles, but with a coverage topology that maps robot configurations to sensor observations, and *vice versa*. Our proposed randomized planning techniques, implemented using high-performance data structures and optimization methods, can be used to plan an inspection in whole or in part, over as long or as short a horizon as time allows. They will hopefully serve as an effective module in the capabilities of a fully autonomous underwater vehicle.

Appendix A

Observation of a Continuous Structure Boundary

In Section 3.4, we presented a probabilistic completeness analysis applicable to any algorithm that employs random sampling of robot configurations to cover a finite set of discrete geometric primitives. As mentioned in that section, the discrete analysis requires only two scalar parameters to describe the difficulty of a coverage problem: the total number of geometric primitives, and a ratio comparing the volumes of the C-Space region being sampled and the smallest subset of views with a single primitive in common. To guarantee coverage of the full continuous boundary of a structure, however, more problem-specific details are required in the analysis: the robot sensor's field of view, the dimensionality of the workspace, and the degrees of freedom available for positioning the sensor in the workspace.

Here we review the concepts that play a role in a continuous analysis of the coverage sampling problem (CSP), as defined in Definition 1 of Chapter 3. The key parameter is the *Vapnik-Červonenkis dimension* (VC-dimension) [161], a quantity that captures the "hardness" of a problem's geometry using a single scalar value. The derivation of this quantity for a specific robot, sensor, and workspace comprises the main challenge of a continuous analysis. We will introduce the tools that can be used, in combination with the VC-dimension, to establish quantitative bounds on algorithm convergence.

We rely once again on the set system taxonomy introduced in Chapter 3, with some modifications for continuous coverage. We refer to points on the continuous surface of the structure under inspection as $p_i \in P$, and the sampled robot view configurations as $q_j \in \mathcal{Q}$. $S_i \in \mathcal{S}$ refers to the set of all feasible configurations in $\mathcal{Q}$ that map to sightings of the point $p_i \in P$. We once again invoke the primal set system $(P, \mathcal{Q})$ and the dual set system $(\mathcal{Q}, \mathcal{S})$ to aid in the analysis.

A.1 Infinite P Preliminaries: VC-dimension and ϵ-nets

If P is an infinite set, the limit in (3.5) no longer holds and a different approach is required to show probabilistic completeness of a coverage sampling algorithm. Even if the number of sets $S_i \in \mathcal{S}$ is infinite, we can still establish a bound on the number of samples needed to guarantee k-coverage of P, required by the redundant roadmap algorithm, with a specific probability of failure.

We first introduce the concept of *shattering* a set. Consider a finite subset of points $B \subseteq P$. If the intersection of B with the members $q_j \in \mathcal{Q}$ yields every single one of the $2^{|B|}$ combinatorially distinct subsets of B, then B is *shattered* by $\mathcal{Q}$. Consequently, there must be at least $2^{|B|}$ distinct sets in $\mathcal{Q}$ for B to be shattered. An important property related to shattering is the VC-dimension, which we define below.

Definition 9 (Vapnik-Červonenkis (VC) Dimension)**.** *The VC-dimension of a set system $(P, \mathcal{Q})$ is the cardinality of the largest subset of P that can be shattered by the family of ranges $\mathcal{Q}$.*

The VC-dimension figures critically in several theorems on set systems. It dictates the approximation factor of a polynomial-time hitting set approximation algorithm [24], which has been used in planning and sensor placement problems [67], [81], [62] to achieve a better worst-case approximation than the classical set cover approximation algorithms. The VC-dimension also appears in theorems on the sampling of random points from a set $B \subseteq P$ of a set system $(P, \mathcal{Q})$ [70]. In particular, the VC-dimension

governs the maximum number of samples required to achieve an *ϵ-net* with high probability. An ϵ-net intersects all ranges whose intersection with B is greater than $\epsilon|B|$ in size.

Definition 10 (ϵ-net)**.** *Let $(P, \mathcal{Q})$ be a set system, let B be a subset of P, let $\epsilon \in [0, 1]$ be a real number, and let $N \subset B$ be a set of samples drawn randomly from B. The subset N is an ϵ-net for B if every range $q_j \in \mathcal{Q}$ of size $|q_j \cap B| > \epsilon|B|$ contains at least one point from N.*

In the dual set system $(\mathcal{Q}, \mathcal{S})$, $\mathcal{Q}$ is an infinite set of robot configurations, and $\mathcal{S}$ is a family of infinite subsets of $\mathcal{Q}$, each of which maps to a view of a specific point $p_i \in P$. We can construct ϵ-nets for the dual system by sampling configurations from an infinite, continuous $A \subseteq \mathcal{Q}$. The fact that A is infinite does not change the role of an ϵ-net; although most commonly presented over finite sets [70], [7], prior analyses have considered ϵ-nets comprised of infinite sets as well, particularly with application to robotics and sensor placement [80], [81]. The sizes of sets A and $S_i \cap A$ can still be compared using a fraction ϵ, but the measure $\mu(A)$, which returns the volume of a set A in robot configuration space, will replace the cardinality $|B|$, and uniform random sampling of continuous A will replace the drawing of samples from finite B.

A.2 Probabilistic Completeness of the Continuous Coverage Sampling Problem

We now present a theorem on the number of samples required to generate a k-covering ϵ-net. This is a recent result from Fusco and Gupta [62] that extends Haussler and Welzl's seminal theorem on sampling ϵ-nets [70] from single-coverage to k-coverage. For the dual set system $(\mathcal{Q}, \mathcal{S})$ and an infinite subset $A \subseteq \mathcal{Q}$, a k-covering ϵ-net is a finite set of points in A that intersects, at least k times each, all ranges whose intersection with A is greater than $\epsilon\mu(A)$ in volume. We now state the theorem for our infinite dual set system $(\mathcal{Q}, \mathcal{S})$.

Theorem 8 (Sampling a k-covering ϵ-net [62])**.** *Let $(\mathcal{Q}, \mathcal{S})$ be a set system, let A be an infinite subset of $\mathcal{Q}$, and let N be a subset of points of size m picked randomly from A. Then, for a number of samples*

$$m \geq max\left(\frac{2}{\epsilon}\log_2\left(\frac{2}{\delta}\right), \frac{C}{\epsilon}\log_2\left(\frac{C}{\epsilon}\right)\right) \tag{A.1}$$

the subset N is a k-covering ϵ-net for A with probability at least 1-δ, where $C = 4(d_{VC} + 2k - 2)$, and d_{VC} is the VC-dimension of the set system.

Theorem 8 gives the minimum number of samples required to guarantee a k-covering ϵ-net with worst-case failure probability δ. To apply this theorem in a useful way, we must obtain an ϵ-net for $\epsilon = \min_{S_i \in \mathcal{S}} \mu(S_i \cap A)/\mu(A)$, and we must also ensure that the VC-dimension is well-behaved in the parameters of the coverage problem. Prior analyses have established the worst-case VC-dimension for a variety of sensor coverage problems, which are summarized below.

- Floor coverage of a 2D workspace by limited-range sensors with a circle-shaped field-of-view [80]: 3
- Floor coverage of a 2D workspace by limited-range sensors with a triangle-shaped field-of-view [80]: 5
- Boundary coverage of a 2D polygon by infinite field-of-view cameras, placed anywhere in the 2D workspace [159]: 23
- Boundary coverage of a 2D polygon by infinite field-of-view cameras positioned along a circular track from which every point in P is visible [81]: 2
- Boundary coverage of a 2D polygon by infinite field-of-view cameras positioned anywhere in the 2D worksapce outside the convex hull of P [81]: 5
- Boundary coverage of a 3D polyhedron with v vertices, by infinite field-of-view cameras, placed on an enclosing sphere from which every point in P is visible [81]: $\Theta(log(v))$

Let us assume that a polygon boundary inspection problem in a 2D workspace is governed by some of the same parameters as the ship hull inspection example given in Section 3.4: $\epsilon = 10^{-3}$ and $k = 10$. Let us assume the robot has an infinite field-of-view sensor and can be positioned at any collision-free configuration in the workspace, so its worst-case VC-dimension is 23. Unlike (3.1) of our prior analysis in Section 3.4, (A.1) contains two arguments on the right-hand side, the larger of which gives the number samples required to guarantee a failure probability of δ. To guarantee a failure probability of one percent, the left argument requires about fifteen thousand samples; for one tenth of one percent, it requires about twenty-two thousand samples; for one hundredth of one percent, it requires about twenty-nine thousand samples. If the left argument were the largest argument, it would offer an appealing exponential decay of failure probability, similar to that established in Theorem 1. Unfortunately, the right argument of (A.1), irrespective of δ, evaluates to nearly three million samples. This means that a threshold of three million samples dominates the convergence guarantee of Theorem 8 until the left-hand term surpasses the large value of the right-hand term, which occurs at a trivially small value of δ. The presence of a sampling threshold is an issue unique to a continuous analysis; it emerges from the original theorem on sampling an ϵ-net for $k = 1$ [70].

Although a constant VC-dimension allows for simplification of Theorem 8, the final result in the above list, $d_{VC} = \Theta(log(v))$, is of the greatest relevance to our application of interest. For the coverage of 3D structures, even when the sensor is restricted to positioning on a 2D manifold, the VC-dimension is no longer a constant. The quantity instead depends on the number of vertices v of the polyhedron being covered. If an upper bound on v is known for the coverage problem of interest, as well as the problem-specific dependence of the VC-dimension on v, then it is possible to apply Theorem 8 in the way we have done for the 2D case. We have not pursued this for HAUV hull inspection, however, in the interest of broad applicability of the analysis. The HAUV is compatible with a variety of sensor payloads characterized by different geometries. The discrete analysis of Chapter 3 can be applied to sensor payloads with arbitrary geometry governing the field of view.

Appendix B

Tables

Table B.1: Resources Used for Coverage Path Planning Software Implementation

Software	Use	Link
OpenSceneGraph (OSG)	KD-Tree Data Structure for Triangle Mesh, Ray Shooting	http://www.openscenegraph.org
Fast Library for Approximate Nearest Neighbors (FLANN)	KD-Tree Data Structure for Nearest-Neighbor Queries	http://www.cs.ubc.ca/~mariusm/index.php/FLANN/FLANN
Open Motion Planning Library (OMPL)	RRT Implementation	http://ompl.kavrakilab.org/index.html
Proximity Query Package (PQP)	Collision Checking	http://gamma.cs.unc.edu/SSV
Boost Graph Library (BGL)	Minimum Spanning Tree	http://www.boost.org/libs/graph
Blossom IV	Min-Cost Perfect Matching	http://www2.isye.gatech.edu/~wcook/blossom4
Concorde	Lin-Kernighan TSP Heuristic	http://www.tsp.gatech.edu/concorde.html
IBM ILOG CPLEX Optimization Studio	Linear Programming Solution of Set Cover	http://www-01.ibm.com/software/integration/optimization/cplex-optimization-studio/
EfPiSoft	Mesh Segmentation	http://efpisoft.sourceforge.net
Point Cloud Library	Interface to FLANN and Rendering of Waypoints and Meshes	http://pointclouds.org
MATLAB	Data Plots and Multi-Colored Rendering of Waypoints and Meshes	http://www.mathworks.com
Myaa	Anti-Aliasing Script for Multi-Colored Renderings	http://www.mathworks.com/matlabcentral/fileexchange/20979

Table B.2: HAUV Field Experiments

Ship	Length [m]	Beam [m]	Test Location	Date	Tasks Performed
USNS Red Cloud (T-AKR-313)	290	32	Newport News, VA	June 2010	Preliminary Testing and Development of Identification Survey, Acquisition and Processing of Sonar-Derived Point Clouds
USCGC Venturous (WMEC-625)	64	10	St. Petersburg. FL	October 2010	Testing and Development of Identification Survey, Acquisition and Processing of Sonar-Derived Point Clouds
SS Curtiss (T-AVB-4)	183	27	San Diego, CA	February 2011	Successful Identification Survey, Generation of Mesh from Point Cloud
USCGC Seneca (WMEC-906)	82	12	Boston, MA	April 2011	Successful Identification Survey, Generation of Mesh from Point Cloud, Preliminary Testing and Development of Inspection Survey
Nantucket Lightship (LV-112)	45	10	Boston, MA	June 2011	Successful Identification Survey, Generation of Mesh from Point Cloud, Testing and Development of Inspection Survey
M/V Terry Bordelon	46	11	Panama City, FL	June 2011	Successful Identification Survey, Generation of Mesh from Point Cloud, Testing and Development of Inspection Survey
USCGC Seneca (WMEC-906)	82	12	Boston, MA	February 2012	Successful Inspection Survey, Planned and Executed using Prior Model from April 2011 Field Experiment
USCGC Seneca (WMEC-906)	82	12	Boston, MA	July 2012	Preliminary Testing of Close-Range Camera Inspection Surveys

Table B.3: Software Resources Used In Field and Laboratory Experiments

Software	Use	Link
Bluefin Robotics Standard Payload Interface	HAUV Control and Data Acquisition	http://www.bluefinrobotics.com/technology/autonomy-and-behaviors/
Lightweight Communications and Marshalling (LCM)	Message-Passing for HAUV Control and Data Acquisition	http://code.google.com/p/lcm/
SeeByte 3D Reconstruction Software	Filtering of DIDSON Data, Production of DIDSON-Derived Point Clouds	http://www.seebyte.com
Meshlab	Processing and Meshing of Acoustic Data	http://meshlab.sourceforge.net
Robotics Operating System (ROS)	Drivers for Hokuyo Laser Rangefinder	http://www.ros.org/wiki/

Bibliography

[1] E.U. Acar and H. Choset. Sensor-based coverage of unknown environments: Incremental construction of morse decompositions. *International Journal of Robotics Research*, 21(4):345–366, 2002.

[2] E.U. Acar, H. Choset, and J.Y. Lee. Sensor-based coverage with extended range detectors. *IEEE Transactions on Robotics*, 22(1):189–198, 2006.

[3] E.U. Acar, H. Choset, A.A. Rizzi, P.N. Atkar, and D. Hull. Morse decompositions for coverage tasks. *International Journal of Robotics Research*, 21(4):331–344, 2002.

[4] E.U. Acar, H. Choset, Y. Zhang, and M. Schervish. Path planning for robotic demining: Robust sensor-based coverage of unstructured environments and probabilistic methods. *International Journal of Robotics Research*, 22(7–8):441–466, 2003.

[5] P.K. Agarwal, E. Ezra, and S.K. Ganjugunte. Efficient sensor placement for surveillance problems. In *Proceedings of the International Conference on Distributed Computing in Sensor Systems, Lecture Notes in Computer Science*, volume 5516, pages 301–314, Berlin, Germany, 2009. Springer-Verlag.

[6] B. Akgun and M. Stilman. Sampling heuristics for optimal motion planning in high dimensions. In *Proceedings of the IEEE/RSJ International Conference on Intelligent Robots and Systems*, pages 2640–2645, San Francisco, CA, 2011.

[7] N. Alon and J.H. Spencer. ϵ-nets and VC-dimensions of range spaces. In *The Probabilistic Method*, pages 243–248. Wiley, Hoboken, NJ, 2008.

[8] D. Applegate, W. Cook, and A. Rohe. Chained Lin-Kernighan for large traveling salesman problems. *INFORMS Journal on Computing*, 15(1):82–92, 2003.

[9] D.L. Applegate, R.E. Bixby, V. Chvatal, and W.J. Cook. *The Traveling Salesman Problem: A Computational Study*. Princeton University Press, Princeton, NJ, 2006.

[10] P.N. Atkar, H. Choset, A.A. Rizzi, and E.U. Acar. Exact cellular decomposition of closed orientable surfaces embedded in $\Re^3$. In *Proceedings of the IEEE International Conference on Robotics and Automation*, volume 1, pages 699–704, Seoul, South Korea, 2001.

[11] P.N. Atkar, A. Greenfield, D.C. Conner, H. Choset, and A.A. Rizzi. Hierarchical segmentation of surfaces embedded in $\Re^3$ for auto-body painting. In *Proceedings of the IEEE International Conference on Robotics and Automation*, pages 572–577, Barcelona, Spain, 2005.

[12] P.N. Atkar, A. Greenfield, D.C. Conner, H. Choset, and A.A. Rizzi. Uniform coverage of automotive surface patches. *International Journal of Robotics Research*, 24(11):883–898, 2005.

[13] M. Attene, B. Falcidieno, and M. Spagnuolo. Hierarchical mesh segmentation based on fitting primitives. *The Visual Computer*, 22(3):181–193, 2006.

[14] R. Bajcsy. Active perception. *Proceedings of the IEEE*, 76(8):996–1005, 1988.

[15] J.E. Banta, L.M. Wong, C. Dumont, and M.A. Abidi. A next-best-view system for autonomous 3-D object reconstruction. *IEEE Transactions on Systems, Man, and Cybernetics – Part A: Systems and Humans*, 30(5):589–598, 2000.

[16] E. Belcher, W. Hanot, and J. Burch. DIDSON: Dual frequency identification sonar. In *Proceedings of the International Symposium on Underwater Technology*, pages 187–192, Tokyo, Japan, 2002.

[17] D. Berenson, S. Srinivasa, and J. Kuffner. Task space regions: A framework for pose-constrained manipulation planning. *International Journal of Robotics Research*, 30(12):1435–1460, 2011.

[18] P.J. Besl and N.D. McKay. A method for registration of 3-D shapes. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 14(2):239–256, 1992.

[19] J.T. Betts. Survey of numerical methods for trajectory optimization. *AIAA Journal of Guidance, Control, and Dynamics*, 21(2):192–207, 1998.

[20] J. Bialkowski, S. Karaman, and E. Frazzoli. Massively parallelizing the RRT and the RRT*. In *Proceedings of the IEEE/RSJ International Conference on Intelligent Robots and Systems*, pages 3513–3518, San Francisco, CA, 2011.

[21] P.S. Blaer and P.K. Allen. View planning and automated data acquisition for three-dimensional modeling of complex sites. *Journal of Field Robotics*, 26(11–12):865–891, 2009.

[22] A. Bottino and A. Laurentini. Optimal positioning of sensors in 3D. In *Proceedings of the Iberoamerican Conference on Pattern Recognition, Lecture Notes in Computer Science*, volume 3773, pages 804–812, Heidelberg, Germany, 2005. Springer.

[23] K.W. Bowyer and C.R. Dyer. Apsect graphs: An introduction and survey of recent results. *International Journal of Imaging Systems and Technology*, 2(4):315–328, 1990.

[24] H. Brönnimann and M.T. Goodrich. Almost optimal set covers in finite VC-dimension. *Discrete and Computational Geometry*, 14(1):463–479, 1995.

[25] H. Bülow and A. Birk. Spectral registration of noisy sonar data for underwater 3D mapping. *Autonomous Robots*, 30(3):307–331, 2011.

[26] B. Burns and O. Brock. Sampling-based motion planning with sensing uncertainty. In *Proceedings of the IEEE International Conference on Robotics and Automation*, pages 3313–3318, Rome, Italy, 2007.

[27] J.F. Canny and J.H. Reif. New lower bound techniques for robot motion planning problems. In *Proccedings of the 28th Annual IEEE Symposium on the Foundations of Computer Science*, pages 49–60, Los Angeles, CA, 1987.

[28] A.A. Carvalho, L.V.S. Sagrilo, I.C. Silva, J.M.A. Rebello, and R.O. Carneval. On the reliability of an automated ultrasonic system for hull inspection in ship-based oil production units. *Applied Ocean Research*, 25:235–241, 2003.

[29] U. Castellani, A. Fusiello, and V. Murino. Registration of multiple acoustic range views for underwater scene reconstruction. *Computer Vision and Image Understanding*, 87(1–3), 2002.

[30] S.Y. Chen and Y.F. Li. Automatic sensor placement for model-based robot vision. *IEEE Transactions on Systems, Man, and Cybernetics – Part B: Cybernetics*, 34(1):393–408, 2004.

[31] S.Y. Chen and Y.F. Li. Vision sensor planning for 3D model acquisition. *IEEE Transactions on Systems, Man, and Cybernetics – Part B: Cybernetics*, 35(5):894–904, 2005.

[32] P. Cheng, J. Keller, and V. Kumar. Time-optimal UAV trajectory planning for 3D urban structure coverage. In *Proceedings of the IEEE/RSJ International Conference on Intelligent Robots and Systems*, pages 2750–2757, Nice, France, 2008.

[33] W. Chin and S. Ntafos. Optimum watchman routes. *Information Processing Letters*, 28(1):39–44, 1988.

[34] W. Chin and S. Ntafos. Shortest watchman routes in simple polygons. *Discrete and Computational Geometry*, 6(1):9–31, 1991.

[35] H. Choset. Coverage for robotics - a survey of recent results. *Annals of Mathematics and Artificial Intelligence*, 31:113–126, 2001.

[36] H. Choset and J. Burdick. Sensor-based exploration: The hierarchical generalized voronoi graph. *International Journal of Robotics Research*, 19(2):96–125, 2000.

[37] H. Choset, K.M. Lynch, S. Hutchinson, G. Kantor, W. Burgard, L.E. Kavraki, and S. Thrun. Sampling-based algorithms. In *Principles of robot motion: theory, algorithms, and applications*, pages 197–267. MIT Press, Cambridge, MA, 2005.

[38] H. Choset and P. Pignon. Coverage path planning: the boustrophedon decomposition. In *Proceedings of the International Conference on Field and Service Robotics*, Canberra, Australia, 1997.

[39] N. Christofides. Worst-case analysis of a new heuristic for the traveling salesman problem. Technical Report CS-93-13, Carnegie Mellon University, 1976.

[40] V. Chvatal. A greedy heuristic for the set-covering problem. *Mathematics of Operations Research*, 4(3):233–235, 1979.

[41] P. Cignoni, M. Corsini, and G. Ranzuglia. Meshlab: An open-source 3D mesh processing system. *ERCIM News*, 73:45–46, 2008.

[42] D. Cline, S. Jeschke, K. White, A. Razdan, and P. Wonka. Dart throwing on surfaces. *Computer Graphics Forum*, 28(4):1217–1226, 2009.

[43] G.E. Collins. Quantifier elimination for real closed fields in cyclindrical algebraic decomposition. In *Proceedings of the 2nd GI Conference on Automata Theory and Formal Languages, Lecture Notes in Computer Science*, volume 33, pages 134–183, Berlin, Germany, 1975. Springer-Verlag.

[44] C.I. Connolly. The determination of next best views. In *Proceedings of the IEEE International Conference on Robotics and Automation*, pages 432–435, St. Louis, MO, 1985.

[45] B. Curless and M. Levoy. A volumetric method for building complex models from range images. In *Proc. 23rd Annual Conference on Computer Graphics and Interactive Techniques*, pages 303–312, New Orleans, LA, 1996.

[46] J.R. Current and D.A. Schilling. The covering salesman problem. *Transportation Science*, 23(3):208–213, 1989.

[47] T. Danner and L.E. Kavraki. Randomized planning for short inspection paths. In *Proceedings of the IEEE International Conference on Robotics and Automation*, volume 2, pages 971–976, San Francisco, CA, 2000.

[48] M. Dorigo. Ant colony system: A cooperative learning approach to the traveling salesman problem. *IEEE Transactions on Evolutionary Computation*, 1(1):53–66, 1997.

[49] K. Easton and J. Burdick. A coverage algorithm for multi-robot boundary inspection. In *Proceedings of the IEEE International Conference on Robotics and Automation*, pages 727–734, Barcelona, Spain, 2005.

[50] B. Englot and F. Hover. Inspection planning for sensor coverage of 3D marine structures. In *Proceedings of the IEEE/RSJ International Conference on Intelligent Robots and Systems*, pages 4412–4417, Taipei, Taiwan, 2010.

[51] B. Englot and F. Hover. Multi-goal feasible path planning using ant colony optimization. In *Proceedings of the IEEE International Conference on Robotics and Automation*, pages 2255–2260, Shanghai, China, 2011.

[52] B. Englot and F. Hover. Planning complex inspection tasks using redundant roadmaps. In *Proceedings of the International Symposium on Robotics Research*, Flagstaff, AZ, 2011.

[53] B. Englot and F.S. Hover. Sampling-based coverage path planning for inspection of complex structures. In *Proceedings of the International Conference on Automated Planning and Scheduling*, pages 29–37, Sao Paulo, Brazil, 2012.

[54] B. Englot and F.S. Hover. Sampling-based sweep planning to exploit local planarity in the inspection of complex 3D structures. In *Proceedings of the IEEE/RSJ International Conference on Intelligent Robots and Systems*, Vilamoura, Portugal, 2012. Accepted, To Appear.

[55] U.M. Erdem and S. Sclaroff. Automated camera layout to satisfy task-specific and floor plan-specific coverage requirements. *Computer Vision and Image Understanding*, 103(3):156–169, 2006.

[56] J. Faigl, M. Kulich, and L. Preucil. A sensor placement algorithm for a mobile robot inspection planning. *Journal of Intelligent and Robotic Systems*, 62(3–4):329–353, 2011.

[57] N. Fairfield, G. Kantor, and D. Wettergreen. Real-time SLAM with octree evidence grids for exploration in underwater tunnels. *Journal of Field Robotics*, 24(1–2):3–21, 2007.

[58] P. Fazli, A. Davoodi, P. Pasquier, and A.K. Mackworth. Complete and robust cooperative robot area coverage with limited range. In *Proceedings of the IEEE/RSJ International Conference on Intelligent Robots and Systems*, pages 5577–5582, Taipei, Taiwan, 2010.

[59] D. Ferguson and A. Stentz. Using interpolation to improve path planning: The field D* algorithm. *Journal of Field Robotics*, 23(2):79–101, 2006.

[60] M. Fischetti, J.J.S. Gonzalez, and P. Toth. A branch-and-cut algorithm for the symmetric generalized traveling salesman problem. *Operations Research*, 45(3):378–394, 1997.

[61] E. Frazzoli, M.A. Dahleh, and E. Feron. Real-time motion planning for agile autonomous vehicles. *AIAA Journal of Guidance, Control, and Dynamics*, 25(1):116–129, 2002.

[62] G. Fusco and H. Gupta. ϵ-net approach to sensor k-coverage. *EURASIP Journal on Wireless Communications and Networking*, 2010:12, 2010.

[63] G. Fusco and H. Gupta. Placement and orientation of rotating directional sensors. In *Proceedings of the 7th Annual IEEE Communications Society Conference on Sensor Mesh and Ad Hoc Communications and Networks*, pages 1–9, Boston, MA, 2010.

[64] Y. Gabriely and E. Rimon. Spanning-tree based coverage of continuous areas by a mobile robot. *Annals of Mathematics and Artificial Intelligence*, 31(1–4):77–98, 2001.

[65] R. Geraerts and M.H. Overmars. A comparative study of probabilistic roadmap planners. In *Proceedings of the Fifth Workshop on the Algorithmic Foundations of Robotics*, pages 15–17, Nice, France, 2002.

[66] H. Gonzalez-Baños and J.-C. Latombe. Planning robot motions for range-image acquisition and automatic 3D model construction. In *Proceedings of the AAAI Fall Symposium Series, Integrated Planning for Autonomous Agent Architectures*, pages 23–25, Orlando, FL, 1998.

[67] H. Gonzalez-Baños and J.-C. Latombe. A randomized art gallery algorithm for sensor placement. In *Proceedings of the 17th Annual ACM Symposium on Computational Geometry*, pages 232–240, Medford, MA, 2001.

[68] S. Gottschalk, M.C. Lin, and D. Manocha. OBBtree: A hierarchical structure for rapid interference detection. In *Proceedings of the 23rd Annual Conference on Computer Graphics and Interactive Techniques*, volume 30, pages 171–180, New Orleans, LA, 1996.

[69] S. Harris and E. Slate. Lamp ray: Ship hull assessment for value, safety, and readiness. In *Proceedings of the IEEE/MTS OCEANS Conference and Exhibition*, volume 1, pages 493–500, Seattle, WA, 1999.

[70] D. Haussler and E. Welzl. Epsilon-nets and simplex range queries. *Discrete and Computational Geometry*, 2(1):127–151, 1987.

[71] S. Hert, S. Tiwari, and V. Lumelsky. A terrain-covering algorithm for an AUV. *Autonomous Robots*, 3(2–3):91–119, 1996.

[72] D.S. Hochbaum. Approximation algorithms for the set covering and vertex cover problems. *SIAM Journal on Computing*, 11(3):555–556, 1982.

[73] G.A. Hollinger, B. Englot, F. Hover, U. Mitra, and G.S. Sukhatme. Uncertainty-driven view planning for underwater inspection. In *Proceedings of the IEEE International Conference on Robotics and Automation*, pages 4884–4891, St. Paul, MN, 2012.

[74] H. Hoppe, T. DeRose, T. DuChamp, J. McDonald, and W. Stuetzle. Surface reconstruction from unorganized points. In *Proc. 19th Annual Conference on Computer Graphics and Interactive Techniques*, pages 71–78, Chicago, IL, 1992.

[75] E. Hörster and R. Lienhart. On the optimal placement of multiple visual sensors. In *ACM International Workshop on Video Surveillance and Sensor Networks*, pages 111–120, Santa Barbara, CA, 2006.

[76] F.S. Hover, R.M. Eustice, A. Kim, B. Englot, H. Johannsson, M. Kaess, and J.J. Leonard. Advanced perception, navigation, and planning for autonomous in-water ship hull inspection. *International Journal of Robotics Research*, 2012. Accepted, To Appear.

[77] F.S. Hover, J. Vaganay, M. Elkins, S. Willcox, V. Polidoro, J. Morash, R. Damus, and S. Desset. A vehicle system for autonomous relative survey of in-water ships. *Marine Technology Society Journal*, 41(2):44–55, 2007.

[78] H. Huang, D. Li, H. Zhang, U. Ascher, and D. Cohen-Or. Consolidation of unorganized point clouds for surface reconstruction. *ACM Transactions on Graphics*, 28(5), 2009.

[79] W.H. Huang. Optimal line-sweep-based decompositions for coverage algorithms. In *Proceedings of the IEEE International Conference on Robotics and Automation*, pages 27–32, Seoul, South Korea, 2001.

[80] V. Isler, S. Kannan, and K. Daniilidis. Sampling based sensor-network deployment. In *Proceedings of the IEEE/RSJ International Conference on Intelligent Robots and Systems*, pages 1780–1785, Sendai, Japan, 2004.

[81] V. Isler, S. Kannan, K. Daniilidis, and P. Valtr. VC-dimension of exterior visibility. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 26(5):667–671, 2004.

[82] D. Latimer IV, S. Srinivasa, V. Lee-Shue, S. Sonne, H. Choset, and A. Hurst. Towards sensor based coverage with robot teams. In *Proceedings of the IEEE International Conference on Robotics and Automation*, pages 961–967, Washington, D.C., 2002.

[83] L. Jaillet and J. Porta. Asymptotically-optimal path planning on manifolds. In *Proceedings of the Robotics: Science & Systems Conference*, Sydney, Australia, 2012.

[84] H. Johannsson, M. Kaess, B. Englot, F. Hover, and J. Leonard. Imaging sonar-aided navigation for autonomous underwater harbor surveillance. In *Proceedings of the IEEE/RSJ International Conference on Intelligent Robots and Systems*, pages 4396–4403, Taipei, Taiwan, 2010.

[85] D.S. Johnson. Approximation algorithms for combinatorial problems. *Journal of Computer and System Sciences*, 9:256–278, 1974.

[86] S. Karaman and E. Frazzoli. Sampling-based algorithms for optimal motion planning. *International Journal of Robotics Research*, 30(7):846–894, 2011.

[87] S. Karaman, M.R. Walter, A. Perez, E. Frazzoli, and S. Teller. Anytime motion planning using the RRT*. In *Proceedings of the IEEE International Conference on Robotics and Automation*, pages 1478–1483, Shanghai, China, 2011.

[88] L.E. Kavraki, M.N. Kolountzakis, and J.-C. Latombe. Analysis of probabilistic roadmaps for path planning. *IEEE Transactions on Robotics and Automation*, 14(1):166–171, 1998.

[89] L.E. Kavraki, P. Svestka, J.-C. Latombe, and M.H. Overmars. Probabilistic roadmaps for path planning in high-dimensional configuration spaces. *IEEE Transcations on Robotics and Automation*, 12(4):566–580, 1996.

[90] G.D. Kazazakis and A.A. Argyros. Fast positioning of limited-visibility guards for the inspection of 2d workspaces. In *Proceedings of the IEEE/RSJ International Conference on Intelligent Robots and Systems*, pages 2843–2848, Lausanne, Switzerland, 2002.

[91] M. Kazhdan, M. Bolitho, and H. Hoppe. Poisson surface reconstruction. In *Proc. Fourth Eurographics Symposium on Geometry*, Cagliari, Italy, 2006.

[92] A. Kim and R. Eustice. Pose-graph visual SLAM with geometric model selection for autonomous underwater ship hull inspection. In *Proceedings of the IEEE/RSJ International Conference on Intelligent Robots and Systems*, pages 1559–1565, St. Louis, MO, 2009.

[93] S. Kirkpatrick, C.D. Gelatt Jr., and M.P. Vecchi. Optimization by simulated annealing. *Science*, 220(4598):671–680, 1983.

[94] D.M. Kocak, F.R. Dalgleish, F.M. Caimi, and Y.Y. Schechner. A focus on recent developments and trends in underwater imaging. *Marine Technology Society Journal*, 42(1):52–67, 2008.

[95] M. Krainin, B. Curless, and D. Fox. Autonomous generation of complete 3D object models using next best view manipulation planning. In *Proceedings of the IEEE International Conference on Robotics and Automation*, pages 5031–5037, Shanghai, China, 2011.

[96] S. Kriegel, T. Bodenmüller, M. Suppa, and G. Hirzinger. A surface-based next-best-view approach for automated 3D model completion of unknown objects. In *Proceedings of the IEEE International Conference on Robotics and Automation*, pages 4869–4874, Shanghai, China, 2011.

[97] J.J. Kuffner. Effective sampling and distance metrics for 3D rigid body path planning. In *Proceedings of the IEEE International Conference on Robotics and Automation*, volume 4, pages 3993–3998, New Orleans, LA, 2004.

[98] J.J. Kuffner and S.M. LaValle. RRT-connect: An efficient approach to single-query path planning. In *Proceedings of the IEEE International Conference on Robotics and Automation*, volume 2, pages 995–1001, San Francisco, CA, 2000.

[99] D. Kurabayashi, J. Ota, T. Arai, and E. Yoshida. Cooperative sweeping by multiple mobile robots. In *Proceedings of the IEEE International Conference on Robotics and Automation*, pages 1744–1749, Minneapolis, MN, 1996.

[100] F. Lamiraux and J.-P. Laumond. On the expected complexity of random path planning. In *Proceedings of the IEEE International Conference on Robotics and Automation*, volume 4, pages 3014–3019, Minneapolis, MN, 1996.

[101] J.-C. Latombe. *Robot Motion Planning*. Kluwer Academic Publishers, Boston, MA, 1991.

[102] S.M. LaValle. Rapdily-exploring random trees: A new tool for path planning. Technical Report TR 98-11, Computer Science Department, Iowa State University, 1998.

[103] S.M. LaValle. Sampling-based motion planning. In *Planning Algorithms*, pages 185–248. Cambridge University Press, Cambridge, UK, 2006.

[104] S.M. LaValle, M.S. Branicky, and S.R. Lindemann. On the relationship between classical grid search and probabilistic roadmaps. *International Journal of Robotics Research*, 23(7–8):673–692, 2004.

[105] S.M. LaValle and J.J. Kuffner. Randomized kinodynamic planning. *International Journal of Robotics Research*, 20(5):378–400, 2001.

[106] E.L. Lawler, J.K. Lenstra, A.H.G. Rinnooy Kan, and D.B. Shmoys, editors. *The Traveling Salesman Problem: A Guided Tour of Combinatorial Optimization*. John Wiley & Sons, Hoboken, NJ, 1985.

[107] J. Leonard, J. How, S. Teller, M. Berger, S. Campbell, G. Fiore, L. Fletcher, E. Frazzoli, A. Huang, S. Karaman, O. Koch, Y. Kuwata, D. Moore, E. Olson, S. Peters, J. Teo, R. Truax, M. Walter, D. Barrett, A. Epstein, K. Maheloni, K. Moyer, T. Jones, R. Buckley, M. Antone, R. Galejs, S. Krishnamurthy, and J. Williams. A perception-driven autonomous urban vehicle. *Journal of Field Robotics*, 25(10):727–774, 2008.

[108] J.J. Leonard, R.J. Rikoski, P.M. Newman, and M. Bosse. Mapping partially observable features from multiple uncertain vantage points. *International Journal of Robotics Research*, 21(10–11):943–975, 2002.

[109] D. Leven and M. Sharir. Planning a purely translational object for a convex object in two-dimensional space using generalized voronoi diagrams. *Discrete and Computational Geometry*, 2(1):9–31, 1987.

[110] S. Lin and B.W. Kernighan. An effective heuristic for the traveling-salesman problem. *Operations Research*, 21(2):498–516, 1973.

[111] T. Lolla, M.P. Ueckermann, K. Yigit, P.J. Haley Jr., and P.F.J. Lermusiaux. Path planning in time dependent flow fields using level set methods. In *Proceedings of the IEEE International Conference on Robotics and Automation*, St. Paul, MN, 2012.

[112] C.T. Loop. Smooth subdivision surfaces based on triangles. Master's thesis, University of Utah, 1987.

[113] L. Lovasz. On the ratio of optimal integral and fractional covers. *Discrete Mathematics*, 13(4):383–390, 1975.

[114] T. Lozano-Perez. Spatial planning: A configuration space approach. *IEEE Transactions on Computers*, C-32(2):108–120, 1983.

[115] T. Lozano-Perez and M.A. Wesley. An algorithm for planning collision-free paths among polyhedral obstacles. *Communications of the ACM*, 22(10):560–570, 1979.

[116] R. Mannadiar and I. Rekleitis. Optimal coverage of known arbitrary environments. In *Proceedings of the IEEE International Conference on Robotics and Automation*, pages 5525–5530, Anchorage, Alaska, 2010.

[117] N.A. Massios and R.B. Fisher. A best next view selection algorithm incorporating a quality criterion. In *Proceedings of the British Machine Vision Conference*, pages 780–789, Southampton, UK, 1999.

[118] J. Maver and R. Bajcsy. Occlusions as a guide for planning the next view. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 15(5):417–433, 1993.

[119] L.L. Menegaldo, M. Santos, G.A.N. Ferreira, R.G. Siqueira, and L. Moscato. SIRUS: A mobile robot for floating production storage and offloading (FPSO) ship hull inspection. In *Proceedings of the International Workshop on Advanced Motion Control*, pages 27–32, Trento, Italy, 2008.

[120] P.E. Missiuro and N. Roy. Adapting probabilistic roadmaps to handle uncertain maps. In *Proceedings of the IEEE International Conference on Robotics and Automation*, pages 1261–1267, Orlando, FL, 2006.

[121] H. Niederreiter. *Random Number Generation and Quasi-Monte Carlo Methods*. Society for Industrial and Applied Mathematics, Philadelphia, PA, 1992.

[122] S. Ntafos. Watchman routes under limited visibility. *Computational Geometry: Theory and Applications*, 1(3):149–170, 1992.

[123] G. Olague and R. Mohr. Optimal camera placement for accurate reconstruction. *Pattern Recognition*, 35(4):927–944, 2002.

[124] J. O'Rourke. *Art Gallery Theorems and Algorithms*. Oxford University Press, Oxford, UK, 1987.

[125] A. Perez, S. Karaman, A. Shkolnik, E. Frazzoli, S. Teller, and M.R. Walter. Asymptotically-optimal path planning for manipulation using incremental sampling-based algorithms. In *Proceedings of the IEEE/RSJ International Conference on Intelligent Robots and Systems*, pages 4307–4313, San Francisco, CA, 2011.

[126] R. Pito. A solution to the next best view problem for automated surface acquisition. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 21(10):1016–1030, 1999.

[127] H. Plantinga and C.R. Dyer. Visibility, occlusion, and the aspect graph. *International Journal of Computer Vision*, 5(2):137–160, 1990.

[128] S. Prentice and N. Roy. The belief roadmap: Efficient planning in belief space by factoring the covariance. *International Journal of Robotics Research*, 28(11–12):1448–1465, 2009.

[129] F. Prieto, T. Redarce, P. Boulanger, and R. Lepage. CAD-based range sensor placement for optimum 3D data acquisition. In *Proceedings of the Second International Conference on 3-D Digital Imaging and Modeling*, pages 128–137, Ottawa, Canada, 1999.

[130] M.K. Reed and P.K. Allen. Constraint-based sensor planning for scene modeling. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 22(12):1460–1467, 2000.

[131] I. Rekleitis, V. Lee-Shue, A.P. New, and H. Choset. Limited-communication, multi-robot team based coverage. In *Proceedings of the IEEE International Conference on Robotics and Automation*, pages 3462–3468, New Orleans, LA, 2004.

[132] E. Rimon and D.E. Koditschek. Exact robot navigation using artificial potential functions. *IEEE Transactions on Robotics and Automation*, 8(5):501–518, 1992.

[133] S. Russell and P. Norvig. *Artificial Intelligence: A Modern Approach*. Prentice Hall, Upper Saddle River, NJ, 2010.

[134] M. Saha, G. Sanchez-Ante, and J.-C. Latombe. Planning multi-goal tours for robot arms. In *Proceedings of the IEEE International Conference on Robotics and Automation*, pages 3797–3803, Taipei, Taiwan, 2003.

[135] M. Saha, G. Sanchez-Ante, T. Roughgarden, and J.-C. Latombe. Planning tours of robotic arms among partitioned goals. *International Journal of Robotics Research*, 25(3):207–223, 2006.

[136] S. Sakane, T. Sato, and M. Kakikura. Model-based planning of visual sensors using a hand-eye action simulator: HEAVEN. In *Proceedings of the Third International Conference on Advanced Robotics*, Versailles, France, 1987.

[137] G. Sanchez and J.-C. Latombe. On delaying collision checking in PRM planning: Application to multi-robot coordination. *International Journal of Robotics Research*, 21(1):5–26, 2002.

[138] G. Sanchez and J.-C. Latombe. Using a PRM planner to compare centralized and decoupled planning for multi-robot systems. In *Proceedings of the IEEE International Conference on Robotics and Automation*, pages 2112–2118, Washington, D.C., 2002.

[139] K. Savla, F. Bullo, and E. Frazzoli. Traveling salesperson problems for a double integrator. *IEEE Transactions on Automatic Control*, 54(4):788–793, 2009.

[140] K. Savla, E. Frazzoli, and F. Bullo. Traveling salesperson problems for the Dubins vehicle. *IEEE Transactions on Automatic Control*, 53(6):1378–1391, 2008.

[141] T. Schouwenaars, B. DeMoor, E. Feron, and J. How. Mixed integer programming for multi-vehicle path planning. In *Proceedings of the European Control Conference*, pages 2603–2608, Porto, Portugal, 2001.

[142] W.R. Scott. Model-based view planning. *Machine Vision and Applications*, 20:47–69, 2009.

[143] W.R. Scott, G. Roth, and J.-F. Rivest. View planning for automated three-dimensional object reconstruction and inspection. *ACM Computing Surveys*, 35(1):64–96, 2003.

[144] S. Shen, N. Michael, and V. Kumar. Autonomous multi-floor indoor navigation with a computationally constrained MAV. In *Proceedings of the IEEE International Conference on Robotics and Automation*, pages 20–25, Shanghai, China, 2011.

[145] T.C. Shermer. Recent results in art galleries. *Proceedings of the IEEE*, 80(9):1384–1399, 1992.

[146] S.N. Spitz and A.A.G. Requicha. Multiple-goals path planning for coordinate measuring machines. In *Proceedings of the IEEE International Conference on Robotics and Automation*, pages 2322–2327, San Francisco, CA, 2000.

[147] J.-L. Starck, E.J. Candes, and D.L. Donoho. The curvelet transform for image denoising. *IEEE Transactions on Image Processing*, 11(6):670–684, 2002.

[148] M. Stilman. Task constrained robot motion planning in robot joint space. In *Proceedings of the IEEE/RSJ International Conference on Intelligent Robots and Systems*, pages 3074–3081, San Diego, CA, 2007.

[149] P. Svestka and M.H. Overmars. Coordinated path planning for multiple robots. *Robotics and Autonomous Systems*, 23(3):125–152, 1998.

[150] K.A. Tarabanis, P.K. Allen, and R.Y. Tsai. A survey of sensor planning in computer vision. *IEEE Transactions on Robotics and Automation*, 11(1):86–104, 1995.

[151] K.A. Tarabanis, R.Y. Tsai, and P.K. Allen. The MVP sensor planning system for robotic vision tasks. *IEEE Transactions on Robotics and Automation*, 11(1):72–, 1995.

[152] G. Tarbox and S. Gottschlich. Planning for complete sensor coverage in inspection. *Computer Vision and Image Understanding*, 61(1):84–111, 1995.

[153] R. Tedrake. LQR-trees: Feedback motion planning on sparse randomized trees. In *Proceedings of the Robotics: Science & Systems Conference*, Zurich, Switzerland, 2009.

[154] L. Torabi and K. Gupta. An autonomous six-DOF eye-in-hand system for in situ 3D object modeling. *International Journal of Robotics Research*, 31(1):82–100, 2012.

[155] G.M. Trimble and E.O. Belcher. Ship berthing and hull inspection using the cetusII AUV and MIRIS high-resolution sonar. In *Proceedings of the IEEE/MTS OCEANS Conference and Exhibition*, volume 2, pages 1172–1175, Biloxi, MS, 2002.

[156] E. Trucco, M. Umasuthan, A.M. Wallace, and V. Roberto. Model-based planning of optimal sensor placements for inspection. *IEEE Transactions on Robotics and Automation*, 13(2):182–194, 1997.

[157] United States Department of the Navy. *The Navy Unmanned Undersea Vehicle (UUV) Master Plan*, 2004.

[158] J. Vaganay, M.L. Elkins, S. Wilcox, F.S. Hover, R.S. Damus, S. Desset, J.P. Morash, and V.C. Polidoro. Ship hull inspection by hull-relative navigation and control. In *Proceedings of the IEEE/MTS OCEANS Conference and Exhibition*, volume 1, pages 761–766, Washington, D.C., 2005.

[159] P. Valtr. Guarding galleries where no point sees a small area. *Israel Journal of Mathematics*, 104:1–16, 1998.

[160] J. van den Berg, P. Abbeel, and K. Goldberg. LQG-MP: Optimized path planning for robots with motion uncertainty and imperfect state information. *International Journal of Robotics Research*, 30(7):895–913, 2011.

[161] V.N. Vapnik and A.Y. Chervonenkis. On the uniform convergence of relative frequencies of events to their probabilities. *Theory of Probability and its Applications*, 16(2):264–279, 1971.

[162] V.V. Vazirani. *Approximation Algorithms*. Springer-Verlag, Berlin, Germany, 2001.

[163] M. Walter, F. Hover, and J. Leonard. SLAM for ship hull inspection using exactly sparse extended information filters. In *Proceedings of the IEEE International Conference on Robotics and Automation*, pages 1463–1470, Pasadena, CA, 2008.

[164] P. Wang, R. Krishnamurti, and K. Gupta. View planning problem with combined view and traveling cost. In *Proceedings of the IEEE International Conference on Robotics and Automation*, pages 711–716, Rome, Italy, 2007.

[165] P. Wang, R. Krishnamurti, and K. Gupta. Generalized watchman route problem with discrete view cost. *International Journal of Computational Geometry and Applications*, 20(2):119–146, 2010.

[166] T. Weyrich, M. Pauly, R. Keiser, S. Heinzle, S. Scandella, and M. Gross. Post-processing of scanned 3D surface data. In *Proc. IEEE Eurographics Symposium on Point-Based Graphics*, pages 85–94, Zurich, Switzerland, 2004.

[167] P. Whaite and F.P. Ferrie. Autonomous exploration: Driven by uncertainty. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 19(3):193–205, 1997.

[168] C. Wurll, D. Henrich, and H. Wörn. Multi-goal path planning for industrial robots. In *Proceedings of the International Conference on Robotics and Applications*, Santa Barbara, CA, 1999.

[169] Y. Yao, C.-H. Chen, B. Abidi, D. Page, A. Koschan, and M. Abidi. Can you see me now? Sensor positioning for automated and persistent surveillance. *IEEE Transactions on Systems, Man, and Cybernetics – Part B: Cybernetics*, 40(1):101–115, 2010.

[170] A. Zelinsky, R.A. Jarvis, J.C. Byrne, and S. Yuta. Planning paths of complete coverage of an unstructured environment by a mobile robot. In *Proceedings of the International Conference on Advanced Robotics*, pages 533–538, Tokyo, Japan, 1993.