University of California

Los Angeles

Visibility of Point Clouds and Exploratory Path Planning in Unknown Environments

A dissertation submitted in partial satisfaction

of the requirements for the degree

Doctor of Philosophy in Mathematics

by

Yanina Landa

2008

The dissertation of Yanina Landa is approved.

Stefano Soatto

Luminita A. Vese

Andrea L. Bertozzi

Stanley J. Osher, Committee Chair

University of California, Los Angeles

2008

To my wonderful family and,
in particular, to my newborn Lea,
with hope to inspire her curiosity.

Table of Contents

List of Figures

List of Algorithms

Acknowledgments

First of all, I would like to express my gratitude to my advisor, Professor Stanley Osher, for his very generous support and guidance, and the invaluable knowledge gained during his lectures and the Level Set Collective meetings.

I would also like to thank my other mentor and collaborator, Professor Richard Tsai, whose innovative ideas always presented the problem in a new light and sometimes lead to unexpected yet exciting turns in the research. He has also been a great host, when I visited the University of Texas in Austin.

I am grateful to my committee members: Professor Andrea Bertozzi, Professor Stefano Soatto and Professor Luminita Vese for their encouragement and valuable suggestions in my research and studies.

The energy and enthusiasm of Professor Bertozzi has always been a motivation for me to keep pushing forward towards achieving my academic and career goals. Our collaboration during the summer of 2006 has been very enjoyable and productive.

Professor Luminta Vese has played the key role in my decision to continue study Mathematics at the graduate level. She introduced me to Applied Mathematics research during the Research in Industrial Projects for Students (RIPS) program held by the Institute of Pure and Applied Mathematics (IPAM) during the summer of 2002.

Additionally, I would like to thank all the people, who helped me in my studies and supported me while at UCLA: Professor Russell Caflisch, Professor Christian Ratsch, Professor Chris Anderson, Professor Inwon Kim, Professor Mark Green, Professor Allon Percus, Maggie Albert, Babette Dalton, Rocie Carrillo, Igor Yanovsky, Pradeep Thiyanaratnam, and many others.

Chapters 2 and 3 of the dissertation are partially based on versions of [LTC06] and [LGH07], and on unfinished manuscripts by myself and Richard Tsai, titled "Visibility of point clouds and exploratory path planning in unknown environments" and "Visibility of three dimensional point clouds and ENO surface reconstruction".

This work was supported by ONR MURI subcontract from Stanford University and ARO MURI subcontract from University of South Carolina.

The experimental results described in Section 3.2, were obtained during the RIPS program at the IPAM, and funded in part by NSF grant DMS-0439872 and NSA grant H98230-06-1-0057. I would like to thank Doctor Matthew Sottile who was a RIPS industrial mentor from Los Alamos National Laboratory, Professor Andrea Bertozzi for her aid in conducting the experiments in the Applied Mathematics Laboratory, and the students, who participated in the project: David Galkowski, Yuan R. Huang, Abhijeet Joshi, Christine Lee, Kevin K. Leung, Gitendra Malla, Jennifer Treanor, and Vlad Voroninski.

Most importantly, my deepest gratitude goes to my parents, Asya and Michael Landa, for their continuous encouragement and support, and to my husband, Yevgeniy Segal, for his care, patience, and assistance. Finally, I am endlessly grateful to my American family: Wendy Japhet and Jared Seide for making it all possible.

Vita

1981	Born, Dnepropetrovsk, Ukraine
2003	B.S. Applied Mathematics with Specialization in Computing, B.A. Design and Media Arts, University of California, Los Angeles
2003-2008	Research and Teaching Assistant, Department of Mathematics, University of California, Los Angeles

Publications and Presentations

Y. Landa, D. Galkowski, Y. R. Huang, A. Joshi, C. Lee, K. K. Leung, G. Malla, J. Treanor, V. Voroninski, A. L. Bertozzi, and Y.-H. R. Tsai, "Robotic path planning and visibility with limited sensor data", *American Control Conference, 2007. ACC '07*, pp. 5425–5430, 2007.

Y. Landa, R. Tsai, and L.-T. Cheng, "Visibility of point clouds and mapping of unknown environments", *Advanced Concepts for Intelligent Vision Systems, 2006. ACIVS '06*, pp. 1014–1025, 2006.

C. Ratsch, Y. Landa, and R. Vardavas, "The asymptotic scaling limit of point island models for epitaxial growth", *Surface Science*, **578**:196–202, 2005.

Abstract of the Dissertation

Visibility of Point Clouds and Exploratory Path Planning in Unknown Environments

by

Yanina Landa

Doctor of Philosophy in Mathematics

University of California, Los Angeles, 2008

Professor Stanley J. Osher, Chair

The problem of visibility involves the determination of regions in space that are visible to a given observer when obstacles to sight are present. In this thesis we investigate the problem of visibility of point clouds. The problem is defined as follows: given a point cloud sampled from opaque objects in two- or three-dimensional space, the regions in space that are visible to a given observer must be determined and the visible portions of the occluding surfaces must be reconstructed. In this dissertation, we present an algorithm that projects point clouds onto a sphere centered at the observing position and performs essentially non-oscillatory (ENO) [HEO87] interpolation of the projected data. Furthermore, it is demonstrated how our visibility formulation can be incorporated into novel algorithms for mapping unknown environments with a single or multiple observers. Experimental results are presented as a validation of the proposed algorithm. Moreover, theoretical estimates of the algorithm's convergence are discussed. Also, postprocessing optimization techniques are considered to obtain a more uniform exposure of the explored region along the path.

CHAPTER 1

Introduction

The problem of visibility can be formulated as follows: given a collection of hypersurfaces representing the boundaries of objects, called the *occluders*, in two- or three-dimensional space, establish the regions of space or on the surfaces that are visible to a given observer. When the observer is replaced by a light source in the simplified geometrical optics setting with perfectly absorbing boundary condition at the obstacles, the problem translates to that of finding illuminated regions. In this regard, the visibility problem is highly related to the high frequency wave propagation problems [LOT06] and is needed in many computational high frequency wave approaches.

Below is a brief survey of the research related to visibility. Section 1.1 summarizes main applications of visibility in different fields of science. It also formulates general problems that will be addressed in the thesis. In Section 1.2, different representations of visibility are considered with the emphasis on combinatorial and variational approaches. A robotic path planning with visibility considerations is described in Section 1.3. The chapter terminates with an overview of the main contributions and organization of this dissertation.

1.1 Visibility Overview

The visibility problem arises as an essential part of various applications a number of scientific fields, *e.g.* computer graphics and visualization [FDF90, Rog97, Dur99], robotic motion planning [Can88, Lat91, Lau98, LaV06], tool paths generation [DM97], computational geometry [GO04], etching [SA97], modeling of melting ice [Bet01], and inverse problems [TCO04], to name a few. In the field of computer graphics and visualization, for example, visibility information can be used to improve the efficiency of a complicated rendering process by skipping over an occlusion. In robotics mission planning, achieving certain visibility objectives may be a part of the mission. One such example is a video camera surveillance design. In modeling problems, such as etching and ice melting, visibility is used as a physical condition to advance the surface given a radiating source. There are also variational problems that minimize the corresponding energy functionals over the visible regions of the ambient space, see *e.g.* [CT05]. Visibility problems have also been studied by geometers. For example, Wente asked if connectedness of the surface shadow is sufficient to imply convexity of the occluding surface [Gho02].

In general, one may consider the following categories of visibility problems:

Category 1: Given occluders, construct shadow volume and its boundary.

Category 2: Given a projection of visible regions, construct the occluders.

Category 3: Find vantage location(s) that maximize visibility using certain predefined metric.

In many visualization applications, the problems in category 1 are solved by projecting triangles. The question studied in [Gho02] can be viewed as in category 2.

Problems related to surveillance fit in category 3. This thesis addresses problems that fall under all these categories. Below we summarize the main problems under consideration.

Problem 1.1. *A vantage point and a set of points (a point cloud) that are evenly distributed over solids are given. The surfaces of the solids are piecewise smooth. Construct a high order accurate representation of the portions of the solid surfaces that are visible from the vantage point. Also, generate the corresponding occlusion volume.*

Problem 1.2. *A bounded domain with unknown solid obstacles and a vantage point are given. Assume an evenly distributed set of points can be sampled from the portions of the unknown solids that are visible to a given vantage location. Construct a piecewise linear path so that (a) any point on the solid surfaces is seen by at least one vertex of the path; and (b) an accurate representation of the solids is constructed from the point clouds that are collected at the vertices of the path.*

In practice, the point cloud can be obtained from sensors such as LIDAR or even from triangulated surfaces (here the point cloud would be the set of vertices).

1.2 Representations of Visibility

Today, computational geometry and combinatorics are the primary tools to solve the visibility-based problems [Urr00], [GO04]. These techniques are mainly concerned with defining visibility on polygons and more general planar environments with special structure. An alternative approach to represent visibility is considered in [TCO04]. The authors construct an implicit framework, where obstacles to sight are represented by a level set function [OS88]. Then, the visibility prob-

lem is formally stated as a boundary value problem (BVP) of a first order partial differential equation.

The above techniques vary greatly in their treatment of the visibility problem, each offering a number of advantages in particular settings. For example, the combinatorial approach leads to fast and elegant solutions in simplified planar polygonal environments. However, this approach becomes increasingly complex in more realistic settings, especially in three dimensions. On the other hand, the implicit level set framework allows great flexibility in the structure of the environment. Still, the variational approach may be too computationally expensive for some applications. The visibility representation introduced in this thesis is tied to both techniques, utilizing their advantages while working around the aforementioned shortcomings. Therefore, below we present a brief survey of the main results in both areas.

1.2.1 Computational Geometry and Combinatorial Approach

A class of visibility problems in computational geometry was originated by Klee in 1973. He asked: *How many guards are necessary, and how many are sufficient to patrol the paintings and works of art in an art gallery with n walls?* Equivalently, one can imagine light bulbs instead of guards and require full direct-light illumination. The most general results obtained to date are summarized in the book by O'Rourke [OR87] as well as surveys [She92, Urr00].

Numerous variations of the art gallery problems have been studied in the last two decades, including mobile guards, guards with limited visibility and/or mobility [TML07], illumination of families of convex sets on the plane [Gho02], guarding of rectilinear polygons [LSS02], and others. The classical art gallery theorems revolve around polygons [Chv75, CN91]. Another cluster of problems

requiring guarding (or illumination) of the boundaries of planar regions with special geometries, such as convex sets, circles, triangles, congruent squares, and line segments have all been investigated [Urr00, GO04]. The art gallery problem is related to many questions raised in vision and robotics as presented in Section 1.3, and, recently, in computer graphics, where the acquisition of models from photographs requires the choice of good viewpoints.

Whereas the art gallery theorems seek to encapsulate environment's visibility into a single function of n (where n is the number of walls in a gallery), *visibility graphs* allow to consider a more detailed structure of visibility. Visibility graph is defined as a graph with nodes for each object and arcs between objects that can see each other. Given any abstract graph, one may ask: *is it the visibility graph of any scene?*

In [PV96], the visibility complex of a finite collection of pairwise disjoint convex sets in plane is considered. The search-domain is decomposed into cells, such that all the points inside a cell see the same set of objects in the environment. This two-dimensional complex may be thought of as a generalization of the tangent visibility graph of the region with obstacles.

More generally, a space may be decomposed into equivalence classes of similar visibility of an object. All the elements inside a class have a similar qualitative view, or an *aspect*, of the object. In this context, an aspect is the set of views of the object that share the same combinatorial structure. This leads to the study of aspect graphs [BD90].

In [GMR97], a planar polygonal environment is decomposed into cells with equivalent visibility properties. The combinatorial structure of the region is defined in terms of *spurious* and *non-spurious* edges. A spurious edge is an edge that does not exist in the environment's boundary, but is formed by the occluding

surface. The sequence of spurious and non-spurious edges defines the *visibility skeleton*. The environment is then decomposed into cells which share the same visibility skeleton. Such a decomposition is called the *visibility cell decomposition*.

In the aforementioned decompositions, there is no significant change in information if the observer moves inside a cell. However, once the observer crosses the cell boundary, there are drastic changes in the visibility structure. Such sudden changes are referred to as *visual events* [Dur99].

The classical approaches often lack reliability when applied in practice due to problems such as mapping uncertainty, registration, segmentation, localization errors, and unpredictable control errors. Furthermore, previous algorithmic efforts have often assumed the availability of perfect geometric models. An alternative approach that minimizes information requirements has been developed by LaValle *et al.* [RL01, TGL05, LaV06]. In order to avoid traditional problems such as complete map building and exact localization, the authors introduce a minimal visibility representation based on detecting discontinuities in depth information (called *gaps*) and their topological changes in time, (referred to as *gap critical events*). Note that this formulation does not require any geometric measurements.

In LaValle's approach, the observer is modeled as a moving point in a connected open set R in the plane. Let $O = \{o_1, o_2, ..., o_n\}$ be the possibly empty set of pairwise disjoint obstacles, in which $o_i \subset R$ for $i = 1, 2, \ldots, n$ is a closed set. Let ∂o_i be the boundary of $o_i \in O$. Then $F = R \setminus \cup_{i=0}^{n} o_i$ is the free space. The observer is free to move in F. The boundary ∂F of F consists of piecewise smooth closed curves.

Assume the observer is equipped with a sensor that is capable of producing representation as shown in Figure 1.1. A sample region R with obstacles O is

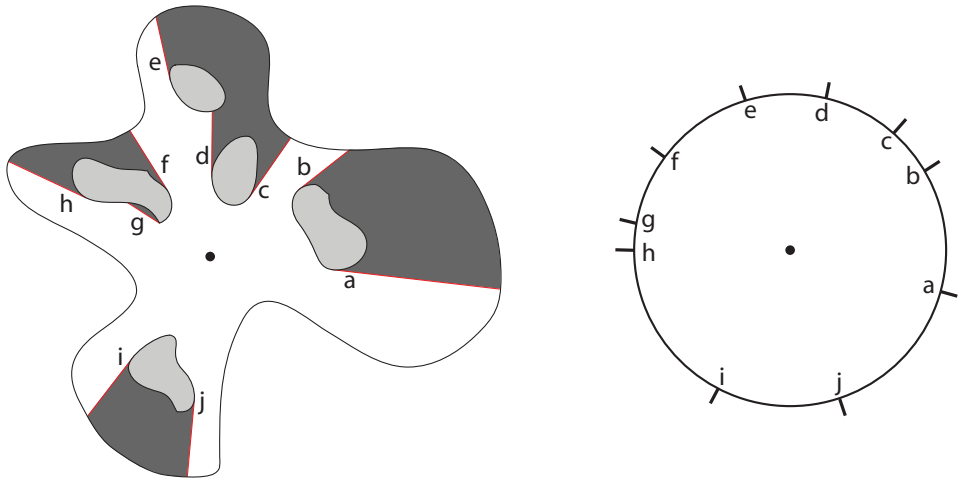
displayed in Figure 1.1 (a). The observer's position is marked by a black point in the center of the region. The shaded regions are invisible to the observer from its current position. In a sense, Figure 1.1 (b) indicates how the world appears to the observer at all times. All the gaps are marked on a disk centered at the observer's position, relative to observer's heading. Note that each gap corresponds to a connected portion of R that is not visible to the observer. The precise distances or angular directions of the discontinuities are unknown.

The simplified representation of the environment is the major limitation of the combinatorial approach. All the results are based on an underlying assumption of straight lines of sight. Furthermore, the extension of these algorithms to three dimensional problems may be extremely complicated. Examples of visibility construction algorithms in three dimensions can be found in [AS96, CT97, Dur99, DDP02]. Such algorithms often combine special data structures and related algorithms for the efficient decomposition and the information retrieval of the configuration space.

1.2.2 Level Set Visibility

While explicit surfaces, *e.g.* triangulated surfaces, are used in the majority of computer graphics and vision applications, implicit surface representation becomes increasingly popular. This is partly due to the fact that in many applications the data, *i.e.* surfaces, are obtained and stored in an implicit form. Thus, it is natural to work directly with the implicit data rather then convert to an explicit representation. Another reason is the increasing popularity of the level set methods, first introduced by S. Osher and J. Sethian in [OS88].

In the level set method an interface is represented as the zero level set of a continuous real valued function, called the *level set function*. Denote this



(a) (b)

Figure 1.1: The observer's view of environment according to [TGL05]. The position of observer is marked by a black disk. (a) The environment and respective labeling of the detected gaps. The free space F is white. (b) Relative angular position of gaps in the visibility space.

function by ϕ. Then the interface Γ is embedded as the zero level set of ϕ: $\Gamma = \{x \in \mathbb{R}^n | \phi(x) = 0\}$. Such a representation retains geometric information of the interface. Furthermore, the level set function can be used to capture a given dynamics of the interface using a time dependent partial differential equation.

Note that the level set function is not unique. For example, $\alpha\phi$, where $\alpha \neq 0$ is also a level set function corresponding to the same interface. The most important form of the level set function is the one that results in small errors when numerically solving the time dependent PDE for the dynamics of the interface or when extracting the interface location [CT07].

The idea behind most visibility algorithms with implicit surfaces is to send rays out of the vantage point to the point of interest (or the reverse) and test for

intersections of the ray with the obstacle's surface, based on information arising from implicit formulation. Alternatively, in order to determine visibility of a given point one may compare geodesic and Euclidean distances between the observer and this point [Set99]. Here, the *geodesic distance* is the distance between two points in space in the presence of obstacles.

The algorithm proposed by Y.-H. Tsai *et al.* in [TCO04] is based on the *causality relation of visibility*: if a point is occluded, then all other points farther away from the vantage point along the same ray are also occluded. The key distinction of this formulation from all the previous approaches is that the rays are sent out from the vantage point in an *implicit* manner so as to propagate the causality relation of visibility.

The algorithm from [TCO04] works with an implicit representation of obstacle surfaces and generates an implicit description of the visibility information with a PDE-based method. Using the notation from [TCO04], let D be the occluding objects in a bounded domain Ω. Let x_0 be the location of the observer. Denote by ψ the level set function representing obstacles, such that ψ is negative in the interior of the objects. The level set visibility function ϕ is generated in an implicit manner, proceeding along the rays emanating from x_0. Analytically,

$$\phi(x) = \min_{z \in \mathcal{L}(x,x_0)} \psi(z), \tag{1.1}$$

where $\mathcal{L}(x, x_0)$ is the integral curve of the vector field $r(x)$ of not necessarily straight rays connecting x and x_0. Thus $\phi(x)$ is negative when x is occluded. The visibility problem can then be formulated as a boundary value problem of a first order partial differential equation [TCO04].

The advantage of this formulation is in the substantial amount of information that can be extracted. Simple operations allow to obtain accurate location

of shadow boundaries, their normal vectors, curvatures, and surface areas. One may also easily compute the distance from any point in space to a shadow boundary. The visible and invisible regions can be accurately classified using the level set representation. Moreover, the formulation retains nearly all the benefits of a level set method, including painless Boolean operations on sets, incorporation of the geometric information, and handling of various surface topologies afforded. The dynamics of the visibility with respect to a moving vantage point or dynamic surfaces can be derived and tracked implicitly within the same framework. Importantly, this formulation allows simple solutions to a class of visibility optimization problems [CT05].

The disadvantage of the method is in its inefficiency in typical computer graphics applications. For example, in rendering only the visibility of the two-dimensional object surfaces is of interest, as opposed to the entire three-dimensional space used in the level set computations. In many cases, the existing hardware allows for fast polygon rendering, but not implicit surface operations. Moreover, the representation of open surfaces is problematic using the level set approach. In addition, the level set visibility may not be used in some applications, *e.g.* navigation in unknown environments, where no map of the region is available *a priori*. In such cases, an online sensing is used to determine visibility [TML07].

1.3 Robotic Path Planning with Visibility Considerations

The motion planning is a fundamental problem in robotics. It has been an active area of research since 1980's. A comprehensive presentation of motion planning techniques can be found in the book by Latombe [Lat91]. In [HKL99, HKL04], the authors provide a broad review of problems specific to the research in robotics. In the most general form, motion planning consists of finding a robot's path

from a start position to a goal position, while avoiding obstacles and satisfying some constraints [Lat91]. One may also consider multiple observers and moving obstacles [WS03].

This thesis is primarily concerned with visibility-based navigation. Those robotics tasks, for which sensor information can be modeled as a *visibility region* are of particular interest. The visibility region is the set of points that can be joined with a line segment to the observer's position, without intersecting the obstacles' boundaries. As the observer moves in space, its visibility region changes, thus modifying information available to the observer about the space or progress towards the goal.

Similarly to the previous section, the navigation algorithms discussed below are split into two groups according to the underlying visibility formulations: the graph-based and the level set-based. After a brief general survey of typical goals of robotic path planning and the corresponding strategies, we concentrate on the "gap-chasing" algorithm by S. LaValle, B. Tovar *et al.* [TML07]. Their navigation strategy has inspired the navigation algorithm introduced in this dissertation, and thus requires a detailed account.

As with most combinatorial techniques, the algorithm from [TML07] becomes more complex in general types of environments. For example, several modifications to the original algorithm are required in multiply-connected domains. Moreover, in mapping applications, the graph-based representation of the environment does not provide any physical description of the explored region, which may be very important in applications. Due to lack of the geometric representation, optimality of the navigational path may be difficult to achieve in some cases, *e.g.* in multiply-connected regions.

In contrast, the implicit visibility formulation [TCO04] provides simple tools

to construct optimal paths for the observers in general types of environments. Unfortunately, the level set method is inapplicable to the problem of mapping of an unknown environment, where the level set map of the region may not be constructed in advance.

The mapping algorithm proposed in Chapter 3 combines the implicit geometric representation of the explored environment with the underlying idea of the "gap-chasing" algorithm. Moreover, the optimization techniques from [CT05] are used to postprocess the resulting path.

1.3.1 Tasks and Strategies for Robotic Navigation

Visibility tasks include but are not limited to robot localization, target-finding, pursuit-evasion, and environment exploration. In this review we are going to concentrate on robot localization and exploratory map-building strategies. Both problems require an autonomous observer to navigate in a bounded region to accomplish a certain task.

The task of *localization* refers to the technique through which a robot can determine or update its location in a known environment through analysis of sensor data [Wan90]. That is, we are dealing with a robot at an *unknown* location in an environment for which it *does* have a map. The problem of localization arises in settings that range from the digital analysis of aerial photographs [YD92] to the design of autonomous Mars rovers [SN89, MAW90]. Another application comes from robots that follow a planned path through a scene: as the robot navigates along the planned path, its guiding control system gradually accumulates errors due to mechanical drift. Thus it is desirable to use localization from time to time to verify the actual position of the robot in the map, and apply corrections as necessary to return it to the planned path [Wan90].

A diverse collection of works have studied the localization problem on theoretical and practical levels. An overview of the existing approaches can be found in [Wan90, BEF96, CT00]. In [Kle94], the author utilizes an on-line algorithm to address the following question: *given a map of a polygonal environment, how should a robot equipped with a vision system move around in the environment so as to determine its location while traveling as little as possible?* The subject of [GMR97] is localization in two-dimensional polygonal environments using a range finder and a compass. The algorithm is based on the visibility cell decomposition of the environment. In [OL07], a robot, whose configuration is composed of its position and orientation, moves in a fully-known, simply connected polygonal environment. A sequence of visual events (compass) helps uniquely identify the robot's position up to a rigid body transformation of the environment.

In contrast to localization problems, where the environment is known *a priori*, the *exploratory motion planning* deals with navigation problems in which the environment is not known before the observer starts motion. It is assumed that the observer is equipped with a sensory device such as a range sensor or a camera. As the observer is placed in an unknown environment, it is asked to construct a map, which can be used for subsequent navigation. The decision as to *where to go next?* is formed only by the data contained in the partially complete map.

The probabilistic road-mapping methods provide a heuristic approach to generate a road map through the space, then search to find the lowest cost path [KSL96]. In particular, Yamauchi [Yam97] introduced the frontier based approach, where the robot equipped with a laser-limited sonar is repeatedly directed towards the nearest frontier between the open explored space and the unexplored space. In contrast, possible control sequences of [TBF05] are evaluated by combining the entropy in the pose posterior with expected entropy of the

map averaged over all paths, and selecting the control that minimizes the resulting entropy (or uncertainty). The heuristic nature of the path generation leads to difficulty in characterizing the algorithms in terms of performance, robustness, complexity, and reliability [KKL98].

Feature-based methods use landmarks extracted from the environment to guide the exploration. In [NBL03], each feature within the map is responsible for determining nearby unexplored areas that, if visited, are likely to constitute exploration. The location of the features is uncertain and represented by a set of probability distribution functions. A utility function is used in [MWB02] to trade off between information gain, the cost of moving to the next sensing location, and the utility of localization based on the covariance matrix in selecting the next position. The weakness of the feature-based methods lies in their reliance on the use of uniquely identifiable landmarks in the environment during the exploration. Such landmarks may not always be available.

A recurring theme in robotics research has been the notion of *minimal sensing*, that is, completing a given task with minimum information necessary [Don95]. Moreover, an *abstract sensor* may be designed that suits a particular task. For example, in [RL01], a gap tracking sensor is introduced to track discontinuities in depth information, as described in previous section. In [Rim97], the minimal sensing philosophy is applied towards the classical path-planning problem. A *critical-point detector* and a *passage-point detector* are the abstract sensors which provide a complete characterization of what the robot should look for in its configuration space to achieve a globally convergent navigation. Minimal sensing techniques are used in [CN01] to localize robot on a partially constructed map. The generalized Voronoi graph is used to encode the topological map of the environment. Then, a graph-matching process leads to localization of the robot.

Although providing a compact representation, the lack of metric information makes localization extremely difficult. Hence, some topological approaches are hybrids that also incorporate geometric maps [VR04].

1.3.2 Gap Navigation Tree

Another topological strategy for navigating unknown environments has been introduced by S. LaValle, B. Tovar *et al.* [TGL05, LaV06, TML07]. It relies on the construction of a Gap Navigation Tree (GNT), which is described below in detail, as it motivated the exploration algorithm introduced in this thesis.

The algorithm of [TML07] constructs a topological representation of the environment in the form of a tree with the aid of an abstract gap sensor described in Subsection 1.2.1. The key idea behind GNT is to avoid traditional problems, such as complete map building and exact localization, by constructing a minimal representation, based entirely on critical events in online sensor measurements made by the robot. In [TML07], the authors propose the use of GNT for optimal robot navigation in simply-connected environments, locally optimal navigation in multiply-connected environments, pursuit-evasion, and robot localization.

A typical environment representation obtained through a gap sensor is depicted in Figure 1.1. Each gap hides a connected region of the environment that is occluded to the robot from its current position. It is assumed that the robot may track gaps at all times and record any topological changes. The robot is equipped with a single motion primitive. It enables the robot to rotate itself towards the location of a gap and approach the gap at a constant speed. This is referred to as *chasing the gap*. The chase gap operation can only terminate when the gap disappears from the gap sensor.

Construction of the GNT begins with the addition of nodes as children to the

root of a tree. Each node corresponds to a gap detected by the gap sensor at the starting position of the robot. As the robot moves through the environment, changes to the tree are triggered by the *gap critical events*. These events occur when the robot crosses either an inflection ray or a bitangent ray of the environment boundary. As illustrated in Figure 1.2, gaps appear and disappear when the robot crosses the line of inflection, while crossing a bitangent line will trigger a split or merge of two gaps, depending on the direction of crossing. Each event requires updating of the GNT:

Event of gap appearance: A node is added as a child of the root node.

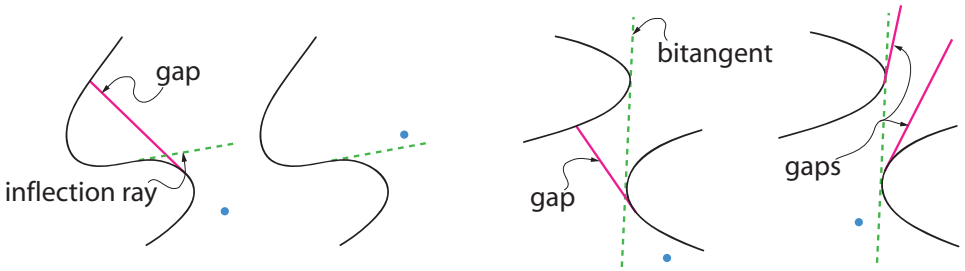
Event of gap disappearance: The node is removed from the tree.

Event of gaps merging: When two gaps g_1 and g_2 merge into a single gap g, g is added to the tree as a child of the root node. The existing nodes g_1 and g_2 become children of g.

Event of gaps splitting: If a gap splits, the corresponding child of the root will be replaced with two children.

According to [TML07], these four operations are sufficient to represent all feasible changes to the environment.

Furthermore, nodes in a GNT fall into two categories. A *non-primitive* node corresponds to an unexplored occluded region. It arises as a result of gap appearance or from the splitting of one of the nodes or its non-primitive children. Non-primitive nodes are used to motivate exploration. *Primitive* nodes are added to the tree as a result of a previously visible area becoming occluded. Hence, chasing a primitive gap will only result in its disappearance and the retracing of previously covered territory.



(a) (b)

Figure 1.2: Gap critical events. (a) Appearance and disappearance of gaps when crossing the inflection ray. (b) Splitting and merging when crossing the bitangent complement rays.

The robot is said to be *exploring*, if it is building the GNT of an environment. When the robot is placed in an unknown environment, all the nodes, or leaves, of the GNT are marked as non-primitive, since the robot has not seen what is behind each corresponding gap. The robot proceeds in the environment by approaching any one of the gaps. The navigation triggers one of the gap critical events. The construction of the tree terminates when all of the leaves are marked as primitive. This condition indicates that the robot has explored the entire environment.

Several complications come up when the robot is placed in a multiply-connected environment [TLM03a]. A problem arises when the environment boundary has no inflections or bitangents, *e.g.* there is a single convex obstacle. In this case, if the robot chases any of the gaps, it will go around the obstacle forever, since the gap will never disappear. In terms of the GNT, no gap-chasing movement modifies the current information state of the robot. In order to resolve this complication, one needs to introduce a marker, which the robot would pick up when it returns to the same position again.

Another drawback of the algorithm is its inability to discriminate one ob-

stacle from another with only gap sensing information in a multiply-connected environment. If assume that the obstacles are uniquely identifiable, *e.g.* have different colors, a second sensor can indicate which gap has already been explored. In this case, a simple strategy to construct a GNT is to follow the wall of each obstacle [TLM03a]. The exploration is complete when the robot has surrounded every obstacle in the environment. We remark that this strategy requires a wall following algorithm [LS87] in addition to gap-chasing, which further complicates its practical implementation.

Optimality of the motion planning is important in practical applications. Below, we provide a few remarks on the optimality of the GNT-based algorithm. It is demonstrated in [TML07] that the path encoded in a GNT between the root of the GNT and any point in the interior of a simply-connected region is optimal. Thus, the GNT is equivalent to the shortest-path tree in simply-connected environments. This observation is applied towards target-finding problem in [TLM03b]. Using the GNT, the location of the target could be associated with the neighboring gap. Therefore, the observer is able to find the shortest path to the target in a known simply-connected environment. This technique also has applications in maze-searching problems.

Unfortunately, global optimality is not guaranteed in multiply-connected regions, since no distance information is encoded in the GNT. In case multiple paths to same location exist, the algorithm chooses the one with fewest gaps. That is, the robot would proceed along the least cluttered but not necessarily the shortest path.

The main limitation of the GNT-based algorithms is that no geometric representation of the explored region is produced as a result of exploration. Also, as with most combinatorial techniques, the extension of the algorithm to three

dimensional environments may be extremely complicated.

1.3.3 Level Set-based Motion Planning

A wide range of problems is related to optimal motion planning, which has a great importance in practical applications. Different notions of optimality result in different algorithmic considerations. The central and most basic question concerning optimal visibility posed in [CT05] is: *What are the optimal positions* $\{x_i\}$ *for a collection of observers, so that jointly the volume of the visible region in* Ω *is maximized?* This question is chosen as the central one due to numerous extensions to a continuous path, weighted regions of importance in space, human visual detail, illumination of the region along a path, shortest path, and accumulation of visibility over time, to name a few. Applications of the above problems include geometric optics [OCK02], scattering [BR94], path planning [CM04, CM06, AS07], surface reconstruction [KS01, JYT03], photolithography [SA97], and differential games [GLL99].

The above question may be addressed under the framework of [TCO04] described in Subsection 1.2.2. One strategy involves producing a visible volume function whose local maxima are the preferred locations for the observers to maximize visibility. Visibility level set functions and gradient flows are applied to obtain the local maxima of this function. The numerical algorithms for a variety of optimization visibility problems considered in [CT05], rely on the continuity of the visibility representation provided by the level set function. For example, the strategy for a more uniform view of space along a path involves the construction of an energy whose minimum achieves the desired effect.

The authors of [CM04] propose a variational approach for finding a search path through a two-dimensional domain, where some information about the location of

the targets and obstacles is available. The level set framework of [TCO04] is used to define an energy integral along the path. Then, the gradient flow is applied to evolve the entire path until a locally optimal steady state is reached. Moreover, the paths are allowed to have positive varying widths, which, in particular, makes the algorithm suitable for computing tool paths trajectories. Other applications include obstacle avoidance, multiple-intersecting paths, paths through multiple prescribed points, and self-intersecting paths. The technique may also be applied towards segmentation of thin objects. In [CM06], the algorithm is extended to three-dimensional problems.

In [KS01], path planning for robots is studied using level sets in a two-dimensional domain with obstacles. Instead of a point robot, a two-dimensional rectangle with given width and length is used to model the observer. The method of solution is to construct a weighted distance function over the entire domain. Then, from a final position, back-propagate the solution orthogonally to the level sets of the distance function, resulting in an optimally shortest path.

While direct optimization techniques provide high performance and robustness, they tend to be computationally intensive. Combining the optimization techniques with more pragmatic heuristic approaches can lead to some guarantees on performance and robustness while reducing the complexity and computation time. For example, in [AS07] the authors consider the problem of finding optimal paths to navigate a terrain with various speeds and obstacles, while reaching a given number of fixed stations. The solution technique combines the fast marching algorithm [Tsi94, Tsi95] with the classical algorithms for the discrete traveling salesman problem to obtain an $O(MN \log N)$ optimal solution, where M is the number of stations and N is the computational grid size. Remark that the task of path optimization while navigating an *unknown* environment is much more

difficult then when the environment is known. The level set based algorithms may not be applied any more, since the map of the region is not provided.

1.4 Contributions and Dissertation Organization

In current chapter we have provided a brief survey of the visibility problem, the representation of visibility, and the visibility-based motion planning. In particular, we have concentrated on two classes of algorithms: the computational geometry and combinatorial-based algorithms and the implicit level set-based algorithms. A special attention has been paid to the visibility formulation and subsequent navigation algorithm of S. LaValle, B. Tovar *et al.* [TML07] and the level set methodology of R. Tsai, L.-T. Cheng *et al.* [TCO04]. The above strategies motivate the visibility representation and the environment-mapping algorithm introduced in this thesis. Further dissertation organization is outlined below.

Chapter 2 presents a proposed algorithm for interpolating the visible portions of a point cloud that are sampled from opaque objects in space. A piecewise high-order reconstruction of the visible obstacle boundary is obtained via Essentially Non-oscillatory (ENO) interpolation. Furthermore, the extension of the algorithm to curved light rays is considered in Subsection 2.2.4. Error analysis of the resulting visibility reconstruction is performed in Subsection 2.2.6, whereas in Subsection 2.2.7 the dynamics of the visibility function with respect to observer's motion is derived. While the algorithm is straightforward in a two-dimensional setting, its extension to three dimensions is much more complicated as demonstrated in Section 2.3.

In Chapter 3, we consider application of visibility to problem of exploration

of unknown bounded regions, which may contain obstacles. An algorithm for a single and multiple observers is introduced. The robustness of the algorithm is discussed in Section 3.2. We also present the experiment of mapping an unknown environment using multiple mobile inexpensive sensors where noise is an issue. In Subsection 3.1.2, we present the statistical control experiments demonstrating the superiority of our proposed strategy for environment exploration, comparing to the random walk strategy. In Section 3.3 postprocessing of the exploration path via optimization with respect to a more uniform illumination of the region of interest is considered. Finally, theoretical complexity estimates of our single-observer exploration algorithm in two- and three-dimensional environments with special structure are provided in Section 3.4.

Chapter 4, summarizes the main contributions of the dissertation and states the problems for future investigation.

CHAPTER 2

Visibility of Point Clouds and Surface Reconstruction

During the past decade a point-based representation (*a point cloud*) has gained increasing popularity [RL00, ZOF01, ZPK02, ABC03, PKK03, KB04]. A point cloud is a set of points in two or three dimensions, which are "uniformly" sampled from a continuous surface of an object. In practice, this data could be obtained from sensors such as LIDAR [LPC00] or even from triangulated surfaces (here the point cloud is formed by the set of vertices). Apart from coordinates, point clouds may possibly be associated with additional information, such as colors and normals. This representation is extremely simple and flexible. Moreover, it offers additional advantage of avoiding connectivity information and topological consistency.

This chapter investigates visibility of point clouds. One possible technique to determine visibility is to reconstruct the surface [ZOM00, ABC03, CBM03]. Once a surface is reconstructed, it is certainly possible to determine which of the points are visible. This implies that a point cloud inherently contains the visibility information of the points. The challenge is to avoid the full reconstruction, which is a difficult problem, both theoretically and implementation-wise. It often requires additional information, such as normals and sufficiently dense input.

In current chapter we discuss the algorithms for extraction of visibility directly

from the point cloud. The visibility information could then be used to reconstruct and visualize visible surfaces. Consider, for example, the algorithm described in [KTB07]. The proposed hidden point removal operator (HPR) extracts the points residing on the convex hull of the transformed point cloud. This procedure amounts to filtering out the visible points. The algorithm works on both sparse and dense point clouds. However, it requires the point cloud to be sampled from a *continuous* surface. If the scene contains several disjoint objects, each object has to be considered individually. This restriction may not be appropriate when dealing with point clouds sampled in an environment with multiple objects.

Below we propose an approximation scheme to determine the visibility of a point cloud sampled from a given occluding surface, possibly with many disconnected components. Given a vantage point, our algorithm would produce a subset of visible data points and a piecewise polynomial interpolation of the visible portions of the surface.

The first step of our algorithm, in some sense, can be viewed as the reverse action of ray-tracing, where discrete rays are sent out from the origin to sample given surfaces. However, instead of assuming a complete explicit or implicit representation of the surfaces, we assume that a set of points is "uniformly" sampled from the surfaces of occluding objects. Unlike the level set representation [TCO04], our algorithm can handle open surfaces and does not require *a priori* knowledge of occluding surfaces to construct visibility. Our scheme can be regarded as a surface reconstruction scheme for the portions of surfaces that are visible to the given vantage point.

The algorithm consists of the following steps:

Step 1: Begin with the point cloud P sampled from the occluding surfaces.

Step 2: Project P onto a unit sphere centered at the vantage point x_0.

Step 3: Filter out portions of P visible to the observer at x_0. [1]

Step 4: Interpolate visible data to obtain a piecewise smooth reconstruction.

The details of the suggested approach are presented in the following sections.

2.1 Projection and Filtering of Data Points

Our approach is based on the observation that visibility along each ray emanating from the vantage point satisfies a *causality condition*: if a point is occluded, then all other points farther away from the vantage point along the same ray are also occluded.

Let $\mathcal{S}^{d-1}$ be the unit sphere in $\mathbb{R}^d$, centered at the origin. We set up a spherical coordinate system centered at the vantage point x_0 by $y = x_0 + r\nu$, where $\nu \in \mathcal{S}^{d-1}$ and $r = |y - x_0|$.

Define the projection operator $\pi_{x_0} : \mathbb{R}^d \mapsto S^{d-1}$, mapping a point onto the unit sphere centered at x_0, by $\pi_{x_0}(x_0 + r\nu) = \nu$. Let Ω be a subset of $\mathbb{R}^d$. Define $\rho_{x_0} : \mathcal{S}^{d-1} \mapsto [0, M)$ by

$$\rho_{x_0}(\nu) := \min_{x_0 + r\nu \in \bar{\Omega} \bigcup \partial B(x_0, M)} r. \tag{2.1}$$

where $B(y, M) = \{y' \in \mathbb{R}^d : |y - y'| < M\}$ is the unit disc with radius M centered at y. The filtering procedure is illustrated in Figure 2.1.

The points $\tilde{y} = \tilde{y}(\tilde{r}, \nu) = x_0 + \tilde{r}\nu \in \Omega$ are classified as *occluded* for all $\tilde{r} > \rho_{x_0}(v)$. A point $y(r, \nu) = x_0 + r\nu \in \partial\Omega$ is called *a horizon point* if and only if $\nu \cdot n(y) = 0$, where $n(y)$ is the outer normal of $\partial\Omega$ at y.

[1] Note that this step is optional if P has only been sampled from x_0.

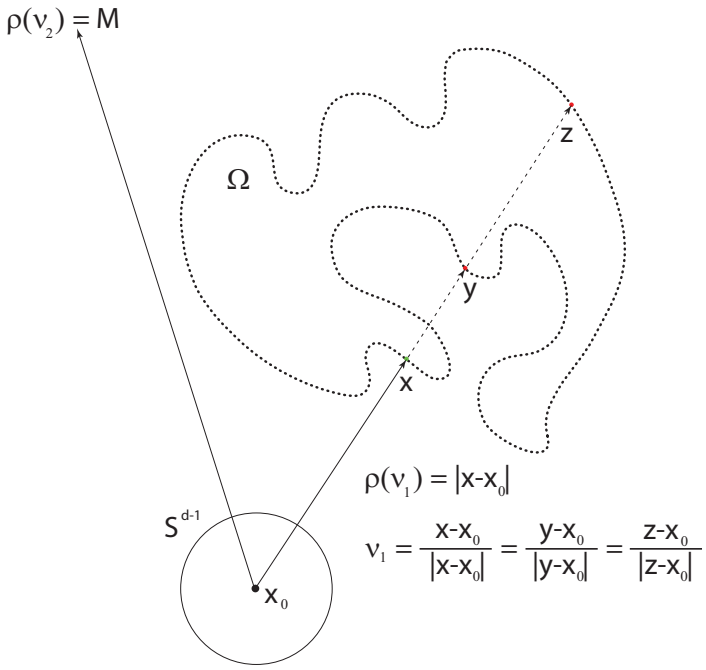


Figure 2.1: Projection of the point cloud onto $\mathcal{S}^{d-1}$ centered at x_0. Filtering of the visible data: x is visible, y and z are invisible from x_0. Values of the visibility function $\rho_{x_0}(\nu_1) = \frac{x-x_0}{|x-x_0|}, \rho_{x_0}(\nu_2) = M$, where $\nu_1, \nu_2 \in \mathcal{S}^{d-1}$.

In practice, points cannot occlude one another (unless they accidentally fall along the same ray from the viewpoint), and therefore no point is actually hidden. Given the set of data points $\{y_j\}$, we start with a partition of the unit sphere $\mathcal{S}^{d-1} = \cup_{i=0}^{N} \bar{K}_i$, where K_i are open regions with diameter ϵ. Similar to the step of performing cell averaging in the Godunov method for conservation laws [LeV92], we define a piecewise constant approximation of ρ_{x_0} by

$$\tilde{\rho}_{x_0}(y) = \min_{y_j} |x_0 - y_j|, \text{for every } y,\ \pi_{x_0} y \in K_i, i = 0, \cdots, N. \tag{2.2}$$

Figure 2.2 illustrates the construction of $\tilde{\rho}_{x_0}$.

Consequently, we classify y as occluded if $\tilde{\rho}_{x_0}\big(\pi_{x_0}(y)\big) < |y - x_0|$. Thus we may define the *visibility indicator*

$$\Xi(y) := \rho_{x_0}\big(\pi_{x_0}(y)\big) - |y - x_0|, \tag{2.3}$$

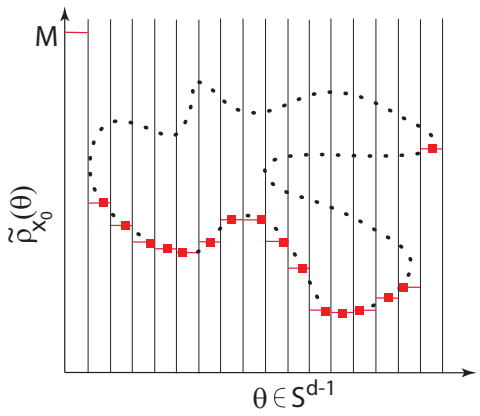


Figure 2.2: Piecewise constant approximation of ρ_{x_0} by $\tilde{\rho}_{x_0}$ using formula (2.2). Squares are the filtered out visible points serving as "originators" of constant values of $\tilde{\rho}_{x_0}$.

such that $\{\Xi \geq 0\}$ is the set of visible regions and $\{\Xi < 0\}$ is the set of regions invisible from x_0.

In case the surface normals are available for each data point, we can use an ellipse instead of a ball in the above construction. A similar approach is also used by QSplat in rendering of the digitized data of Michelangelo's statues [RL00].

2.2 Smoother Reconstruction by ENO Interpolation

Analytically the visibility function ρ_{x_0} is piecewise continuous with jumps corresponding to the locations of horizons. Smoothness of ρ_{x_0} in each of its continuous pieces relates to the smoothness of the corresponding visible part of $\partial\Omega$. In the previous section we obtained a piecewise constant approximation $\tilde{\rho}_{x_0}$ to ρ_{x_0} using formula (2.2). Along the way, we also extracted a subset of visible data points $\tilde{P} \subseteq P$ serving as "originators" of each constant value of $\tilde{\rho}_{x_0}$.

We will use $\tilde{\rho}_{x_0}$ to construct a piecewise polynomial approximation $\rho_{x_0}^{\text{int}}$ to the visibility function which would preserve jump discontinuities. Essentially non-

oscillatory (ENO) interpolation is used to compute such $\rho_{x_0}^{\text{int}}$. ENO interpolation, first introduced by E. Harten *et al.*, is a nonlinear polynomial interpolation that has been widely and successfully used in shock problems of computational fluid dynamics [HEO87, SO88, SO89].

2.2.1 Overview of ENO Polynomial Interpolation

Since its introduction in [HEO87], ENO schemes have been influential in numerical solutions of nonlinear hyperbolic equations. This pioneering work was built upon an adaptive algorithm to choose a local stencil among several possible candidates so that the resulting polynomial interpolant yields high order accuracy whenever the function is smooth but avoids Gibbs phenomena at discontinuities.

In this subsection we briefly review the procedure of constructing one-dimensional ENO polynomial interpolants. For our purposes we will always assume that we have a point x_0 where we will evaluate the interpolant $P(x)$, and that x_0 is between the two values $x_{-1} < x_0 < x_1$ that are on the stencil of points used to construct P. Begin with the stencil $\{x_{-1}, x_1\}$. Define $L = -1$ and $R = 1$ as left and right endpoints of the stencil. Denote by $f_i \equiv f(x_i)$ the function values used in the interpolation. Let r be a maximum polynomial degree. Then the corresponding set of candidate stencil points has the size $2r$: $\{x_{-r} < x_{-r+1} < \ldots < x_{r-1} < x_r\}$. The procedure to construct P is described in Algorithm 2.1.

An example comparing the results of ENO interpolation to a standard Newton divided differences algorithm is depicted in Figure 2.3. The effects of Gibbs phenomena can be clearly seen in Newton's interpolation near the discontinuities at $x = -0.3$ and $x = 0.3$, while the adaptive stencil of ENO interpolation preserves the smoothness of the function up until the discontinuity.

ENO interpolation can be easily extended to multiple dimensions on stan-

Algorithm 2.1 One-dimensional ENO polynomial interpolation

1: **if** $R - L < r + 1$ **then**
2: continue to step 7
3: **else**
4: evaluate $P(x_0)$ using the interpolation stencil $\{x_L, \ldots, x_R\}$
5: **return**
6: **end if**
7: form divided differences $a = f[x_{L-1}, \ldots, x_R]$, $b = f[x_L, \ldots, x_{R+1}]$
8: **if** $|a| < |b|$ **then**
9: $L \leftarrow L - 1$
10: **else**
11: $R \leftarrow R + 1$
12: **end if**
13: go to step 1

dard rectangular grids. The idea is to interpolate the multidimensional data dimension-by-dimension, repeatedly applying Algorithm 2.1.

2.2.2 The Two-dimensional Case

Below we describe the interpolation procedure on $\mathcal{S}^1$ for two dimensional problems. Possible extension of the strategies to the three dimensional problems will be described in Section 2.3.

Denote the extracted visible data points by $p_i \in \tilde{P}$. Since $\mathcal{S}^1$ can be parameterized by angles $\theta \in [-\pi, \pi)$, we can sort the points in $\tilde{P}$ in the increasing order of the angle they form with respect to our chosen spherical coordinate system; that is, points in $\tilde{P}$ are sorted in the increasing order of $\tilde{\rho}_{x_0}^{-1}(p_i) = \arg(p_i - x_0)$.

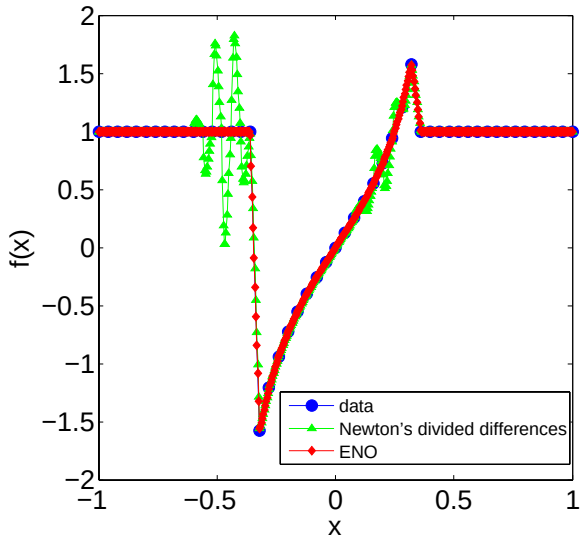


Figure 2.3: Comparison of standard Newton's divided differences interpolation to ENO interpolation. The order of interpolation is 8.

The edges, or discontinuities in the visibility function ρ typically occur near the locations of horizons. A standard in image processing choice for the edge-detection function $g : \mathbb{R} \mapsto (0, 1]$, is $g(s) = 1/\left(1 + s'^2\right)$ [AK02]. If the value of $g(s)$ is below some threshold value, we get an edge. The threshold value depends on the sampling of s.

We implement a modified version of the edge-detection function. Using the piecewise constant values of the visibility function $\tilde{\rho}_{x_0}$, we substitute a finite difference approximation for the derivative of ρ. The resulting edge-detection function $g : \mathcal{S}^1 \mapsto [0, 1)$ maps θ onto

$$g\left(\tilde{\rho}_{x_0}(\theta)\right) = 1/\left(1 + \left(\frac{\tilde{\rho}_{x_0}(\theta_{i+1}) - \tilde{\rho}_{x_0}(\theta_i)}{\theta_{i+1} - \theta_i}\right)^2\right), \theta_i \leq \theta < \theta_{i+1}. \tag{2.4}$$

Periodic boundary conditions are used in this formulation. A natural choice of the threshold value is the polar grid size $\delta\theta$.

We can then construct $\rho_{x_0}^{(1)}$ by linearly interpolating between each successive

pair of p_i and p_{i+1} if $\tilde{\rho}_{x_0}(\theta) \neq M$ for $\theta \in \left[\tilde{\rho}_{x_0}^{-1}(p_i), \tilde{\rho}_{x_0}^{-1}(p_{i+1})\right)$. Instead of linear interpolation, we use ENO interpolation Algorithm 2.2.1 to construct $\rho_{x_0}^{(p)}$, a piecewise p-th order approximation of ρ_{x_0}.

We use the piecewise p-th order approximation $\rho_{x_0}^{(p)}$ to compute derivatives on the occluding surfaces (away from the edges) and to extract various geometric quantities. For example, the curvature of the occluding surface away from the discontinuities can be computed via

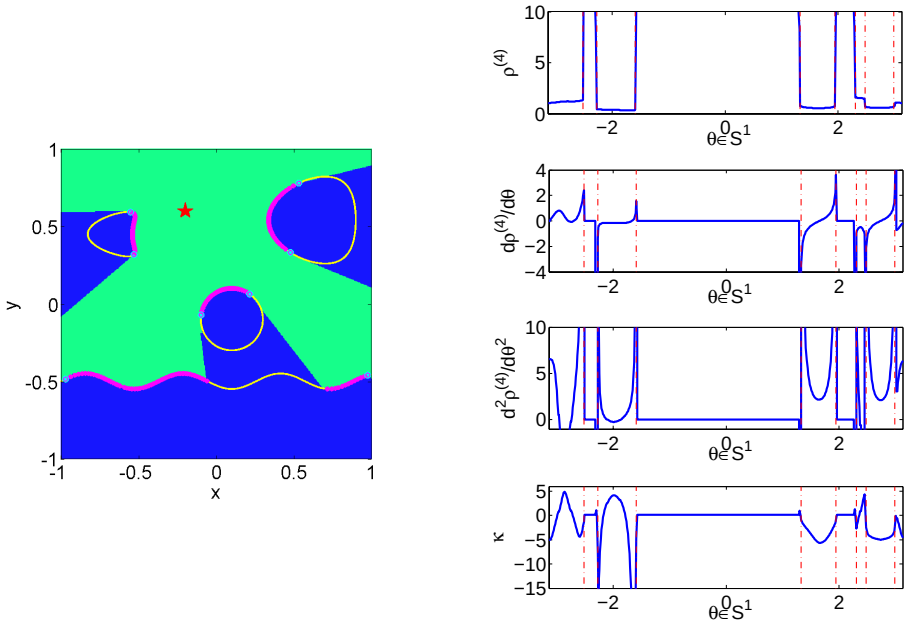
$$\kappa = \frac{\rho^2 + 2\rho_\theta^2 - \rho\rho_{\theta\theta}}{\left(\rho^2 + \rho_\theta^2\right)^{3/2}}. \tag{2.5}$$

In Figure 2.4 (a) we illustrate visibility from the vantage point at $(-0.2, 0.6)$. A corresponding visibility function $\rho^{(4)}(\theta)$, its derivatives, and the curvature κ are displayed in Figure 2.4 (b). We obtain a high order approximation of the derivatives and, subsequently, the curvature along the visible occluding boundaries away from the discontinuities corresponding to horizons.

2.2.3 Processing and Denoising

In real-life applications we frequently deal with noisy data. There are different sources of noise. For example, noise may be introduced by the measuring device as in [LGH07] and [ZOL04]. As one can see from [ZOL04], even a high accuracy sensor produces significant error in curvature computations. Filtering is used in [ZOL04] to clean up the sensor data. In addition to sensor error, noise in the data can be introduced from an uneven terrain and/or presence of foliage, cars, and people in the scene, as in [WHS05].

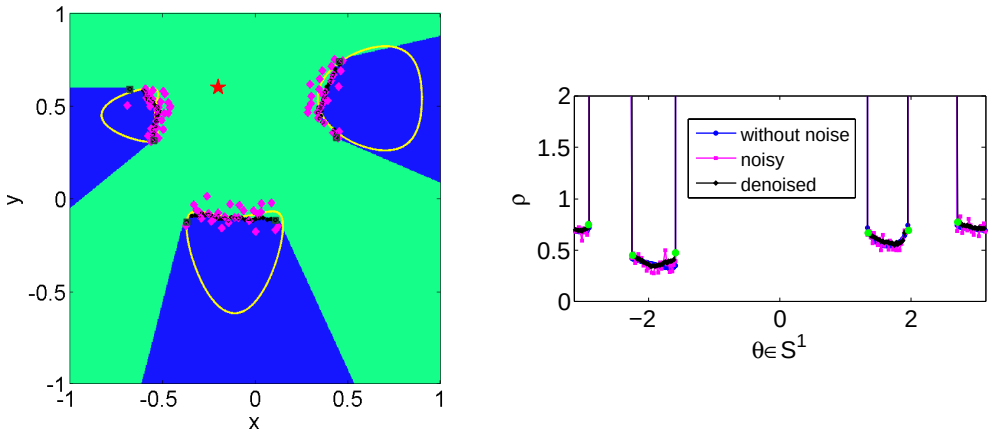
We propose the use of a simple edge-preserving total variation based noise removal algorithm [ROF92], which can be applied to the interpolated data to reduce the effect of noise in the scene. In Figure 2.5 (a), we plot visibility based on



(a) (b)

Figure 2.4: (a) Visibility map generated from artificial data: dark regions - invisible, light regions – visible. Also marked are the vantage point $(-0.2, 0.6)$, actual obstacles' boundaries, visible obstacles' boundaries, and horizon points. (b) Forth order interpolation of the visibility function ρ corresponding to (a), computation of $\frac{d\rho}{d\theta}$, $\frac{d^2\rho}{d\theta^2}$, and the curvature κ via formula (2.5) away from the discontinuities (dashed vertical lines).

the denoised visibility function ρ depicted with black diamonds in Figure 2.5 (b). Here, an artificial noise of variance $\sigma = 0.05$ is added to the projected point cloud. The obtained data is then filtered and interpolated. Afterwards we apply the denoising algorithm from [ROF92].



(a) (b)

Figure 2.5: (a) Visibility map generated from noisy data: dark regions – invisible with respect to the denoised visibility function, light regions – visible. Also marked are the vantage point $(-0.2, 0.6)$, actual obstacles' boundaries (light outline), noisy visible boundaries (diamonds), denoised visible boundaries (dark circles), and horizon points (dark squares). (b) Visibility function ρ corresponding to (a), edges/horizon points are marked by circles.

2.2.4 Curved Lines of Sight

To demonstrate the flexibility of our formulation, consider the case when the integral curves of $\nu(x)$ are more complicated than straight lines. In this case, we may no longer use the relation $\nu(x) = (x - x_0)/|x - x_0|$ in the definition of the visibility 2.1. As in [TCO04, LTC06], we consider instead the flow lines connecting x_0 to x. The construction of the visibility function is done as follows. We begin by constructing the distance function φ to the vantage point x_0 on the entire domain D by solving the eikonal equation

$$|\nabla\varphi(x)| = r(x), \quad \text{in } D,\ \varphi(x_0) = 0, \tag{2.6}$$

where $r(x) > 0$ is the variable index of refraction. We employ the fast sweeping technique from [TCO03] to solve (2.6).

To determine the polar coordinates $(\theta, \rho(\theta))$ corresponding to the point p on the occluding surface we need to trace p back to the vantage point x_0 along the line of sight connecting them:

$$\begin{aligned} \dot{x} &= -\nabla\varphi(x), \\ x|_{t=0} &= p. \end{aligned} \tag{2.7}$$

Then θ is the angle made by $\nabla\varphi$ at x_0, and the distance from x_0 to p is simply $\varphi(p)$. The visibility function ρ can then be constructed using the causality condition with respect to φ.

An example of visibility computation under refraction is depicted in Figure 2.6. The index of refraction r is set to be 1 for $x \in [-1, 0)$ and 2 for $x \in [0, 1]$. The contour lines of φ are represented in Figure 2.6 (a). The observer is positioned at $(-0.2, -0.4)$. The resulting visibility function ρ is shown in Figure 2.6 (b).

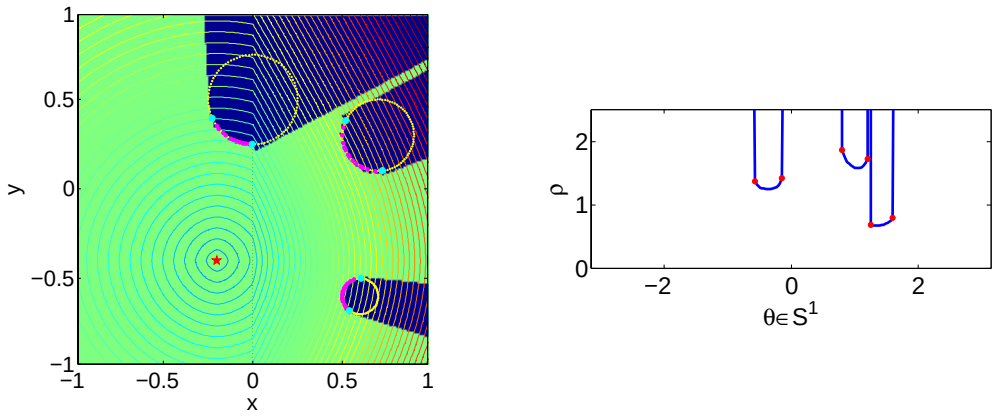
Such computations may be useful when determining visibility in regions with variable refraction such as water or fog, or in an anisotropic medium.

2.2.5 Conversion to Cartesian Level Set Formulation

The piecewise polynomial reconstruction of the visibility function ρ may be used to obtain a smooth level set visibility function ϕ defined on a Cartesian coordinate system. The following construction yields a level set visibility function that is smooth across the discontinuities. Begin by defining a set

$$G := \{(\theta, r) : r < \rho(\theta)\} \tag{2.8}$$

containing the visible points on polar coordinates. We proceed to construct a smooth signed distance function ϕ to the shadow boundary ∂G using redistancing



(a) (b)

Figure 2.6: Visibility under a bending ray field. The refraction index is 1 in the left half-plane and 2 in the right half-plane. (a) Contours of the ray field, observer's position $(-0.2, -0.4)$, obstacles' boundaries, visible boundaries, and horizon locations. Dark regions are invisible, light are invisible from the vantage point. (b) Corresponding visibility function ρ. Discontinuities are marked by circles.

[CT07]:

$$\begin{cases} \phi(\theta, r) > 0, & \text{if } (\theta, r) \in G, \\ \phi(\theta, r) < 0, & \text{if } (\theta, r) \in G^C, \\ \phi(\theta, r) = 0, & \text{if } (\theta, r) \in \partial G. \end{cases} \tag{2.9}$$

The resulting signed distance function may then be easily converted from polar to Cartesian coordinates $\phi(x, y) = \phi(x(\theta, r), y(\theta, r))$ via

$$\begin{aligned} x(\theta, r) &= r\cos(\theta) + x_{0_1}, \\ y(\theta, r) &= r\sin(\theta) + x_{0_2}. \end{aligned} \tag{2.10}$$

On the grid level, this is done by interpolation. Thus obtained level set visibility representation is consistent with the one obtained in [TCO04]. In Figure 2.7 we present the smooth level set visibility function corresponding to the vantage point marked by the red star.

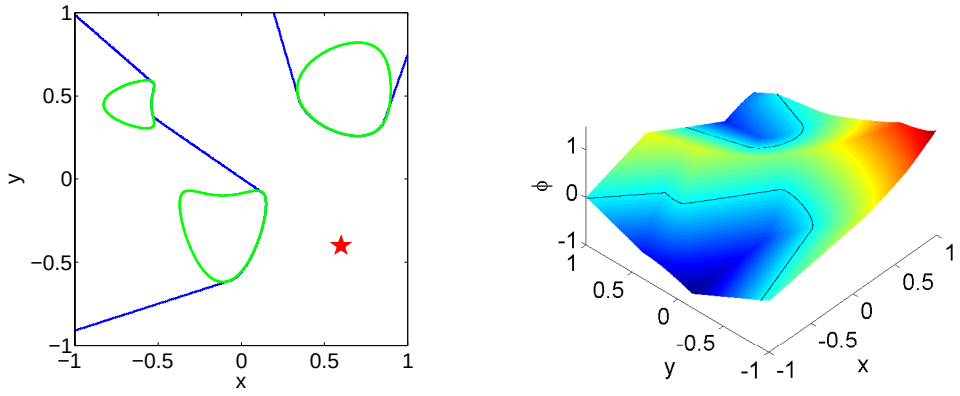
Level set formulation on a fixed Cartesian coordinate system allows for easy boolean operations on visibility of different vantage locations. For example, let $x_0, x_1, \ldots, x_m$ denote the locations of $m+1$ separate observers. For each $i = 0, 1, \ldots, m$ we can construct the visibility level set function $\phi(\cdot, x_i)$ associated with x_i. Visibility information of all the observers can be determined from the visibility information of individual ones, using the definition that a point is visible with respect to multiple observers if it is visible by at least one observer.

In the level set framework, there is an analogy to unions and intersections. For two level set functions ϕ_1 and ϕ_2, the union of their negative regions is implicitly captured as the negative region of $\min\{\phi_1, \phi_2\}$. Correspondingly, the positive region of this function is the intersection of positive regions of ϕ_1 and ϕ_2. Similarly, the intersection of the negative regions of ϕ_1 and ϕ_2 is the negative region of $\max\{\phi_1, \phi_2\}$, and its positive region is the union of positive regions of ϕ_1 and ϕ_2.

From this, we can construct a visibility level set function for multiple observers, $\phi(\cdot; x_0, x_1, \ldots, x_m)$, by taking the maximum value of the visibility level set function for individual observers:

$$\phi(y; x_0, x_1, \ldots, x_m) = \max_{i=0,1,\ldots,m} \phi(y; x_i). \tag{2.11}$$

Joint visibility from three vantage points is depicted in Figure 2.8.



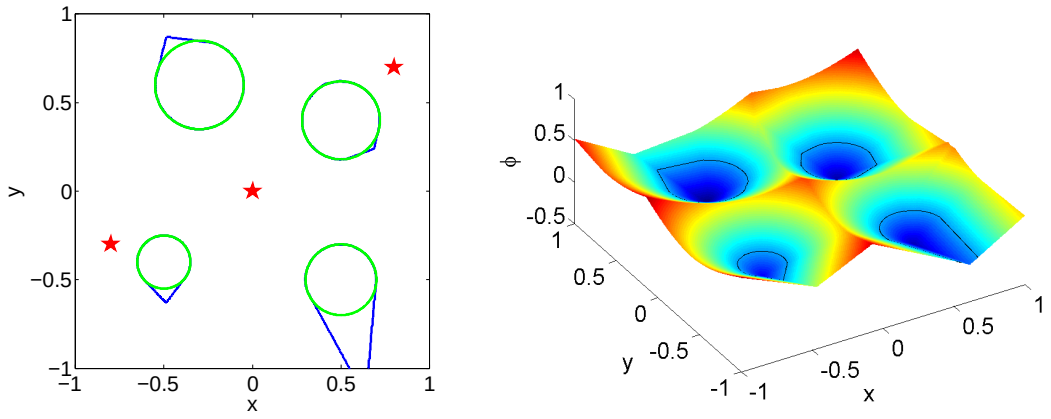
(a) (b)

Figure 2.7: (a) Environment with obstacles, observer at $(0.6, -0.4)$, and shadow boundary. (b) Visibility level set function ϕ corresponding to setup (a).

2.2.6 Error Analysis

In this section we discuss the accuracy of the visibility function resulting from the projection method that uses ENO interpolation. For simplicity, we consider sample environments containing a finite number of disjoint strictly convex objects. The observer is positioned outside the obstacles. We demonstrate how the error relates to the distance from the observer, the view direction, and the size of the fan $\delta\theta$. In particular, we demonstrate how the quality of interpolant deteriorates as the view direction becomes orthogonal to the outer surface normal near the horizon locations.

Without loss of generality, we may assume that there is no partially occluded object. The analysis for the case of partially occluded objects is a straightforward generalization. Due to the convexity assumption, there are exactly two horizons corresponding to each object in the scene, as illustrated in Figure 2.9. Let the horizon locations correspond to θ_L and θ_R. Suppose $\Theta := \{\theta_i \in [\theta_L, \theta_R], i =$



(a) (b)

Figure 2.8: (a) Joint visibility from three observers located at $(0.8, 0.7)$, $(0, 0)$, and $(-0.8, -0.3)$ (stars). Also depicted are the obstacles' boundaries and the shadow boundary. (b) Visibility level set function $\phi(\cdot; x_0, x_1, x_2) = \max_{i=0,1,2} \phi(\cdot; x_i)$.

$0, \cdots, \nu\} \subset [-\pi, \pi)$ are distinct angles sorted in the increasing order. The angles in Θ correspond to view directions from x_0 to points $p_i \in \tilde{P}$ on a visible region of occluding surface that is bounded by two horizons. These angles and points are obtained using the projection described in Section 2.1.

Assume the occluding surface between θ_L and θ_R is smooth such that the visibility function $\rho(\theta) \in C^{n+1}(\theta_L, \theta_R)$. Then for each $\theta \in [\theta_L, \theta_R]$, we have the standard error estimate

$$\rho(\theta) = \rho^{\text{ENO } n}(\theta) + \frac{\rho^{(n+1)}(\xi(\theta))}{(n+1)!} \Pi_{i=0}^{n} (\theta - \theta_i), \text{ for some mean value } \xi(\theta) \in (\theta_L, \theta_R), \tag{2.12}$$

where $\rho^{\text{ENO } n}(\theta)$ is the n-th order ENO polynomial approximation and

$$E^n := \frac{\rho^{(n+1)}(\xi(\theta))}{(n+1)!} \Pi_{i=0}^{n} (\theta - \theta_i) \tag{2.13}$$

is the error term. Since ρ is smooth in $[\theta_L, \theta_R]$, depending on the order of approximation, the error term E^n is bounded according to the regularity of ρ. For

example, with the third order ENO, for any $\theta \in [\theta_3, \theta_{\nu-4}]$, we have

$$\left|E^3\right| \leq \max_{\xi \in (\theta_3, \theta_{\nu-4})} \frac{\left|\rho^{(4)}(\xi)\right|}{4!} (2\delta\theta)^4 .$$

Note that in order for the above bound to hold, we have to assume that ENO interpolation would not choose a stencil that goes across the discontinuities of ρ. Otherwise, if ENO stencil includes the jump location, the remainder term (2.13) can be very big. In order to avoid this problem we introduce the following assumption on the size of the fan used in filtering:

Assumption 2.1. *$\delta\theta$ is small enough, so that $g(\rho(\theta)) < \delta\theta$ implies there is a discontinuity in the visibility function ρ at θ.*

In the above, g is the edge-detector function defined in (2.4). Such $\delta\theta$ can always be found in the asymptotic limit. However, it may not always exist in practical applications, as will be demonstrated in the next chapter.

Furthermore, we require that $\delta\theta$ is small enough, so that the arc connecting p_i and p_{i+1} may be approximated by a straight line segment. This translates into the following assumption:

Assumption 2.2. *$\delta\theta^2|\kappa| < \epsilon$ if $\kappa \neq 0$. Here $0 < \epsilon \ll 1$ is a small constant.*

The above assumption can be easily derived using the Taylor's expansion: for any $\theta \in [\theta_i, \theta_{i+1}]$ we can write $\rho(\theta) = \rho(\theta_i) + (\theta - \theta_i)\rho'(\theta_i) + \frac{1}{2}(\theta - \theta_i)^2\rho''(\xi)$ for some ξ between θ and θ_i. A linear approximation of ρ is obtained by setting the second order term in Taylor's approximation to 0. Precisely, we have $\delta\theta^2\kappa = 0$, where κ is the curvature of the occluding surface. Then, Assumption 2.2 follows.

Thus we have derived the two conditions on the size of the fan $\delta\theta$ which guarantee a bounded error term in the estimate (2.12).

Note that the accuracy of ENO polynomial approximation of ρ is not uniform along the occluding surface. It depends on the view direction and the proximity to the observer. We would like to find an upper bound on the derivatives of ρ, and thus obtain an expression for the remainder term E^n, which relies on the properties of the projection and filtering method.

Without loss of generality, assume $|p_{i+1} - x_0| \geq |p_i - x_0|$. Denote the outer normal to the surface at p_{i+1} by $\vec{n}$ and let the angle between $\vec{n}$ and the view direction θ_{i+1} be $\pi - \phi$. Let $M = \max_{p_i \in \tilde{P}} |x_0 - p_i|$. Using simple trigonometry, we obtain the following bounds

$$|p_i - p_{i+1}| = \frac{\sin(\theta_{i+1} - \theta_i)\,|x_0 - p_i|}{\cos\phi} \leq \frac{M \sin(2\delta\theta)}{\cos\phi}, \tag{2.14}$$

$$|\rho(\theta_{i+1}) - \rho(\theta_i)| = \frac{\sin\left(\phi - \frac{\theta_{i+1}-\theta_i}{2}\right)|p_{i+1} - p_i|}{\cos\left(\frac{\theta_{i+1}-\theta_i}{2}\right)} \leq \frac{2M\tan\phi}{\cos\delta\theta} := K. \tag{2.15}$$

From the above estimates we see that the shortest distance between the two neighboring sample points p_i and p_{i+1} as well as the smallest difference in their corresponding visibility values is obtained when $\phi = 0$, *i.e.* when the view direction is parallel to the outer normal $\vec{n}$. As ϕ approaches $\pi/2$, which happens near the horizon locations, both $|p_{i+1} - p_i|$ and $|\rho(\theta_{i+1}) - \rho(\theta_i)|$ tend to infinity, see Figure 2.9. Also, as $\delta\theta$ decreases to 0, the difference $|\rho(\theta_{i+1}) - \rho(\theta_i)|$ decreases to $2M\tan\phi$.

Denote the minimum distance to a point in $\tilde{P}$ by m. From the relation (2.14),

$$|p_i - p_{i+1}| \geq \frac{m \sin(2\delta\theta)}{\cos\phi}$$

for any points p_i and p_{i+1} in $\tilde{P}$. Then

$$\theta_{i+1} - \theta_i \geq \psi := 2\sin^{-1}\left(\frac{m\sin(2\delta\theta)}{M\cos\phi}\right)$$

for any angles $\theta_i, \theta_{i+1} \in \Theta$. Using the estimate on divided differences

$$\rho[\theta_0, \theta_1, ..., \theta_n] = \frac{\rho^{(n)}(\xi)}{n!}. \tag{2.16}$$

and the relation (2.15), we obtain the following bounds on derivatives of the visibility function

$$\begin{aligned}
|\rho'(\xi)| &= \frac{|\rho(\theta_{i+1}) - \rho(\theta_i)|}{\theta_{i+1} - \theta_i} \leq \frac{1}{\psi} K, \\
\frac{|\rho''(\xi)|}{2!} &= \min\{|\rho[\theta_i, \theta_{i+1}, \theta_{i+2}]|, |\rho[\theta_{i-1}, \theta_i, \theta_{i+1}]|\} \leq \frac{2}{2!\psi^2} K, \\
\frac{\left|\rho^{(3)}(\xi)\right|}{3!} &\leq \frac{2^2}{3!\psi^3} K, \\
&\vdots \\
\frac{\left|\rho^{(n)}(\xi)\right|}{n!} &\leq \frac{2^{n-1}}{n!\psi^n} K.
\end{aligned} \tag{2.17}$$

Then the error term (2.13) can be bounded by

$$\begin{aligned}
|E^n| &= \left| \frac{\rho^{(n+1)}(\xi(\theta))}{(n+1)!} \Pi_{i=0}^n (\theta - \theta_i) \right| \\
&\leq \frac{2^n K \Pi_{i=0}^n |\theta - \theta_i|}{(n+1)!\psi^{n+1}} \\
&\leq \frac{2^n K (\theta_R - \theta_L)^{n+1}}{(n+1)!\psi^{n+1}}.
\end{aligned} \tag{2.18}$$

Furthermore, in a bounded domain $\Omega = B_R(x_0)$, we can estimate the error in the shadow boundary location near the horizon (as θ approaches $\pi/2$), *i.e.* the area of the grey regions in Figure 2.9. The maximum angle of the resulting fan is $\delta\theta$. Then the error in the shadow boundary corresponding to a given horizon is only linear

$$A_{\text{horizon}} \leq \frac{\delta\theta}{2}(R^2 - m^2). \tag{2.19}$$

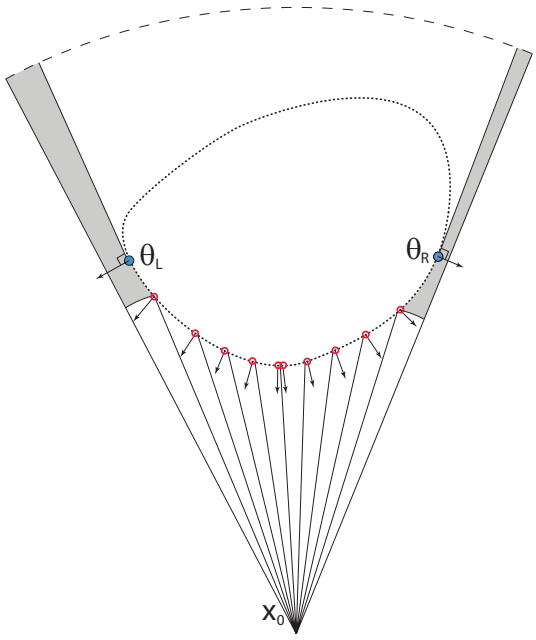


Figure 2.9: Filtered out visible data $p_i \in \tilde{P}$ along with surface normals. Error in the approximation of horizon locations.

2.2.7 Dynamics

Below we derive the dynamics equations of the visibility function and horizon points with respect to the moving vantage point. In two dimensions let us consider a coordinate system centered at x_0 with the visible portions of the occluding surfaces parameterized by polar coordinates. A point z on the occluder is visible from x_0. Assume the observer moves with the velocity $v = (v_1, v_2)$. The value of the visibility function is $\rho_{x_0}(\theta) = |z - x_0|$. Suppose during the period of time Δt the observer has moved to a new location $x_0 + v\Delta t$. The corresponding value of the visibility function is $\tilde{\rho}_{x_0+v\Delta t}(\tilde{\theta}) = |z - (x_0 + v\Delta t)|$. The angle between the velocity vector v and the x-axis is $\varphi = \tan^{-1}\frac{v_2}{v_1}$. The angle between $z - x_0$ and the velocity vector v is ψ. Then, the angle between $z - x_0$ and the x-axis is

$\theta = \varphi + \psi$, see Figure 2.10(a). Then we can compute

$$\begin{aligned}
\frac{d}{dt}\left(\rho^2\right) &= \lim_{\Delta t \to 0} \frac{\tilde{\rho}^2 - \rho^2}{\Delta t} \\
&= \lim_{\Delta t \to 0} \frac{|z - (x_0 + v\Delta t)|^2 - |z - x_0|^2}{\Delta t} \\
&= -2v \cdot (z - x_0) \\
&= -2\rho v \cdot \begin{pmatrix} \cos\theta \\ \sin\theta \end{pmatrix}. \qquad (2.20)
\end{aligned}$$

On the other hand,

$$\frac{d}{dt}\left(\rho^2\right) = 2\rho\frac{d\rho}{dt} = 2\rho\left(\dot{\rho} + \rho_\theta\dot{\theta}\right). \qquad (2.21)$$

Therefore,

$$\dot{\rho} + \rho_\theta\dot{\theta} = -v \cdot \begin{pmatrix} \cos\theta \\ \sin\theta \end{pmatrix}. \qquad (2.22)$$

To find an expression for $\dot{\theta}$, note from Figure 2.10 (a) that

$$\rho \sin\psi = \tilde{\rho}\sin\tilde{\psi} = L. \qquad (2.23)$$

Since L is the distance from z to $x_0 + vt$, it is independent of the motion of x_0 once the direction v is fixed. Therefore,

$$\frac{dL}{dt} = \frac{d\rho}{dt}\sin\psi + \dot{\psi}\rho\cos\psi = 0. \qquad (2.24)$$

Then

$$\dot{\theta} = \dot{\psi} = -\frac{\frac{d\rho}{dt}\sin\psi}{\rho\cos\psi} = \frac{v}{\rho} \cdot \begin{pmatrix} \sin\theta \\ -\cos\theta \end{pmatrix}. \qquad (2.25)$$

Combining (2.25) with (2.22) we finally obtain

$$\dot{\rho} = -\frac{\rho_\theta}{\rho}v \cdot \begin{pmatrix} \sin\theta + \cos\theta \\ \sin\theta - \cos\theta \end{pmatrix}. \qquad (2.26)$$

The above equation (2.26) describes the change of the visible portion of the occluding surface, *i.e.* between the horizons. In order to have a complete description of the visibility we must derive the motion of horizons e_1 and e_2 on Figure 2.10 (b) with respect to the observer.

Note that $(e_i - x_0) \cdot n_{e_i} = 0$, where n_{e_i} is the outer unit normal to the occluding surface at the point e_i for $i = 1, 2$. That is, the vector $e_i - x_0$ is tangent to the occluding surface at the horizon point. Without loss of generality, in all the computations below we will consider just e_1.

In the coordinate system defined as above, $\theta = \varphi + \psi$ is the angle between $e_1 - x_0$ and the x-axis. The value of the visibility function is $\rho_{x_0}(\theta) = |e_1 - x_0|$. Now suppose the observer moves to a new position $x_0 + v\Delta t$, moving with the velocity $v = (v_1, v_2)$. For this new location, the position of the edge has changed to $\tilde{e}_1$ and the corresponding value of the visibility function is $\tilde{\rho}_{x_0+v\Delta t}(\tilde{\theta}) = |\tilde{e}_1 - (x_0 + v\Delta t)|$. Here $\tilde{\theta} = \varphi + \tilde{\psi}$ is the angle between $\tilde{e}_1 - (x_0 + v\Delta t)$ and the x-axis in the coordinate system centered at $x_0 + v\Delta t$. Our goal is to find the change in the position of horizon, *i.e.* $\dot{e_1}$.

First, note that the curvature of the occluding surface at the point $(\theta, \rho(\theta))$ is given by formula (2.5). Also, since $e_1 - x_0$ is tangent to the occluder at e_1, we obtain

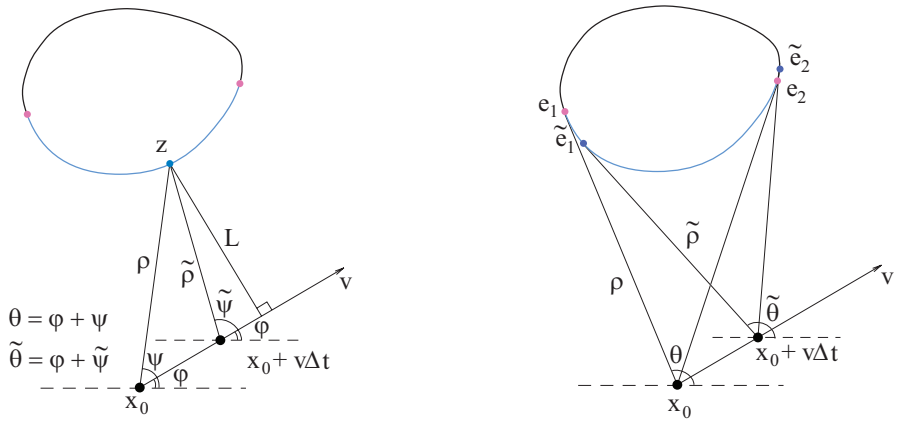
$$\begin{aligned} n^{\perp}(e_1) &= \frac{e_1 - x_0}{|e_1 - x_0|} \\ n(e_1) &= \left(n^{\perp}(e_1)\right)^{\perp} = \left(\frac{e_1 - x_0}{|e_1 - x_0|}\right)^{\perp}. \end{aligned} \tag{2.27}$$

Now we can plug in the above into the formula for horizon dynamics from [TCO04] to get

$$\dot{e_1} = \frac{v \cdot n(e_1)}{\kappa\rho} n^{\perp}(e_1). \tag{2.28}$$

Therefore, from the equations (2.26) and (2.28) we obtain a full description

of the change in the visible portion of the occluder with respect to the observer's motion. The corresponding expressions can also be derived in three dimensions, see [TCO04] for details.



(a) (b)

Figure 2.10: Derivation of the dynamics equations for the visibility function (a) and the horizons (edges) (b).

2.3 Smoother Reconstruction in Three Dimensions

Below we discuss the possible extensions to three dimensions of the algorithm for interpolating visible portions of the point clouds that are sampled from opaque objects in the environment. Section 2.1 describes the algorithm for extracting the subset of visible data points $\tilde{P}$ from a point cloud P and a piecewise constant visibility function $\tilde{\rho}_{x_0}$ via the projection of the point cloud onto a unit sphere $\mathcal{S}^{d-1} \in \mathbb{R}^d$, centered at the vantage point x_0. This simple procedure can be applied in both two- and three-dimensional environments.

Consider an example of a point cloud of the Stanford Bunny, one of the most commonly used test models in computer graphics [TL94]. The point cloud

consists of 35947 points. Figure 2.11 displays the filtered visible points on the surface of the bunny from the vantage point at $(-0.5, 1, 0.7)$. The number of visible points is 2678. More examples will be discussed in Subsection 2.3.3.

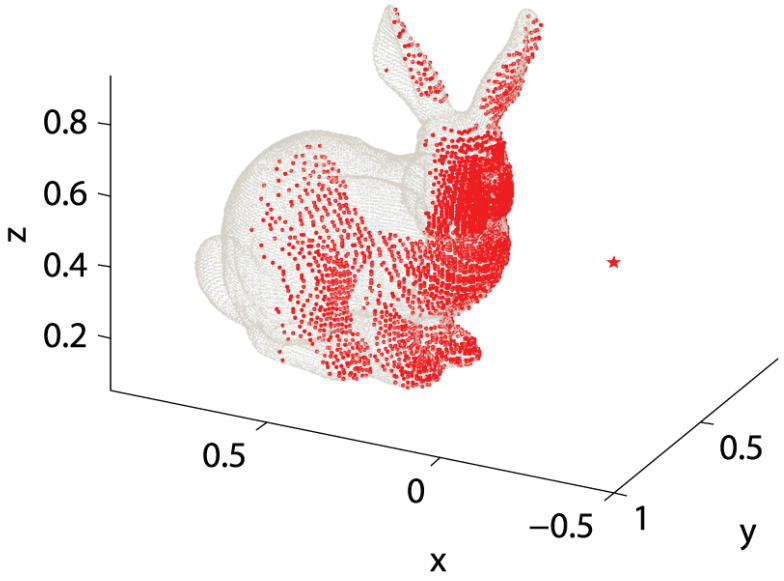


Figure 2.11: Points visible from the vantage point at $(-0.5, 1, 0.7)$. Point cloud size is 35947 points, the number of visible points is 2678.

The last step of the visibility reconstruction algorithm, presented at the beginning of the Chapter, involves a piecewise smooth reconstruction of the visible occluding surfaces through interpolation of the extracted visible data. The details of interpolation on $\mathcal{S}^1$ in two-dimensional setting can be found in Section 2.2.2. ENO interpolation allows to reconstruct a piecewise high order polynomial representation of the visible boundaries of the obstacles.

The problem arises when we try to interpolate the projected data on $\mathcal{S}^2$. The

standard dimension-by-dimension approach may not be applied in this case since the projected data is generally not on a grid. Below we are going to describe the two procedures to obtain a better approximation of the visibility function and the corresponding reconstruction of the visible occluding surfaces. The next two subsections include the detailed description of our selected approach, while the Subsection 2.3.3 illustrates the proposed technique with numerical examples.

The first approach is to use the projected data points as vertices for a triangulation on $\mathcal{S}^2$. Classical Delaunay technique [Del34] can be used to construct the triangulation of unstructured data points. For locally high order reconstructions, we can use the ENO interpolation approaches described in, *e.g.* [LO96, HS99, ZS03], to construct $\rho_{x_0}^{\mathrm{ENO}}$. Thus obtained high order interpolant can be easily mapped back to Cartesian coordinates to obtain a reconstruction of the visible surface in $\mathbb{R}^3$.

The proposed technique highly depends on the triangular mesh generation of the projected data points. Mesh generation is a complicated problem, partly because of the difficulty of forcing a mesh to conform to objects' sharp creases and corners without sacrificing the quality of the resulting triangles. While Delaunay triangulation aims to maximize the minimum angle of all the angles of the triangles in the triangulation, in practice it is difficult to avoid the so called "sliver" triangles near the mesh boundary. In Figure 2.12, we demonstrate the Delaunay triangulation based on 50 randomly generated points on a plane. One can see the narrow and long triangles along the right and bottom boundaries. Such irregular triangles introduce difficulties in computation of the derivatives as well as high order interpolation techniques near the edges of the mesh, which correspond to the locations of the horizons in our visibility formulation.

The second approach is to continue working on a rectangular grid that dis-

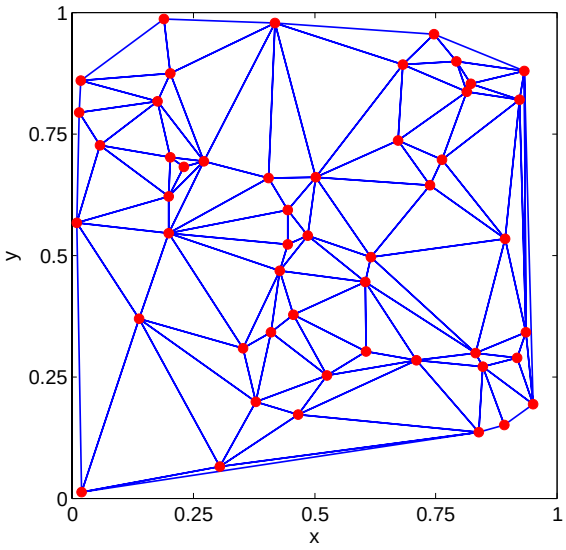


Figure 2.12: The Delaunay triangulation for 50 randomly generated points.

cretizes $\mathcal{S}^2$. We propose to use an ENO-based preprocessing step to map the filtered data to a coarse rectangular grid in $\mathcal{S}^2$. Then a straight forward dimension-by-dimension ENO interpolation can be applied on the refined rectangular grid to obtain a high order approximation, which then can be mapped to $\mathbb{R}^3$. Below are the main steps of the algorithm to construct a high order interpolation of the visible data:

1. Construct a Delaunay triangulation on $\mathcal{S}^2$ using as vertices the filtered data points $\pi(\tilde{P})$.

2. Set up a coarse rectangular grid based on the triangulation.

3. Compute a preliminary interpolant on a coarse rectangular grid.

4. Refine rectangular grid and ENO interpolate the data dimension-by-dimension.

The following subsections contain a detailed description of each step of the algo-

rithm and provide the evaluation of its performance on sample data sets.

2.3.1 Rectangular Grid Construction and Interpolation

Assume we have a triangulation $T = \{K_1, \ldots, K_m\}$ of m non-overlapping triangles K_i, such that no vertex of one triangle lies on the edge of another triangle (as in Figure 2.12). The vertices of the triangles are the projections of the filtered visible points $p \in \tilde{P}$ onto $\mathcal{S}^2$. Figure 2.13 represents the triangulation on $\mathcal{S}^2$, based on the filtered out points on the visible surface of the bunny depicted in Figure 2.11. One can see the irregular triangles corresponding to the horizon locations.

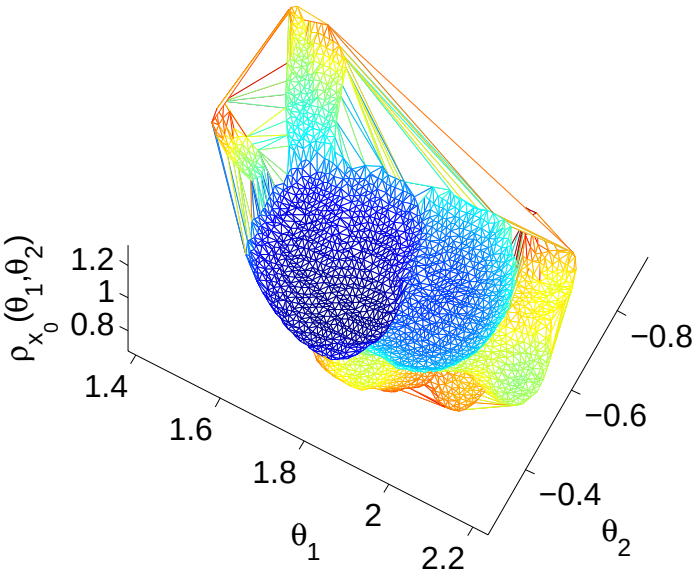


Figure 2.13: Triangulation of $\mathcal{S}^2$ based on filtered visible points. The number of triangles is 5329.

We would like to set up a rectangular grid based on the triangulation T.

Denote the grid nodes of the rectangular grid by $X_{i,j}$. We may interpolate through the values $\tilde{\rho}_{x_0}$, which are given at the vertices of the triangles, to obtain the values of the visibility function on the rectangular grid nodes $X_{i,j}$. We use the following procedure to construct a preliminary second order interpolant on the rectangular grid $X_{i,j}$.

Let $K_{i,j} \subset T$ be the triangle enclosing the grid node $X_{i,j}$. In order to be able to construct a quadratic interpolant, we require the rectangular grid to satisfy the following conditions:

$$\begin{aligned} K_{i,j} \cap K_{i+1,j+1} &= \{\emptyset\}, \\ K_{i,j} \cap K_{i+1,j-1} &= \{\emptyset\}, \\ K_{i,j} \cap K_{i-1,j+1} &= \{\emptyset\}, \\ K_{i,j} \cap K_{i-1,j-1} &= \{\emptyset\}. \end{aligned} \tag{2.29}$$

Figure 2.14 illustrates the requirements (2.29) for the rectangular grid $\{X_{i,j}\}$.

Since the triangles in T are not uniform, it would not be possible to satisfy the criteria (2.29) everywhere on $\mathcal{S}^2$, unless the rectangular mesh is very sparse. Therefore, we choose to satisfy the requirements only partially, that is only on triangles $K \subset T$, which satisfy

$$\operatorname{diam}(K) < \operatorname{diam}(K_{\max}). \tag{2.30}$$

Here, $\operatorname{diam}(K)$ is the diameter of K, *i.e.*, the length of the longest side of the triangle K, and $\operatorname{diam}(K_{\max})$ denotes the threshold value for the "big" triangles. As noted above, the elongated triangles correspond to the locations of horizons. Thus, the thresholding (2.30) allows us to filter the *interior* triangles that are away from the discontinuities. In order to partially satisfy the requirements (2.29) in the interior set, in all our experiments we set the rectangular mesh size

$$h = C \operatorname{mode}_{K \subset T}\{\operatorname{diam}(K) |\ \operatorname{diam}(K) < \operatorname{diam}(K_{\max})\}. \tag{2.31}$$

In the above, C is a constant parameter, which is usually set to 1. Thus, h is the most frequently occurring side length of the interior triangles.

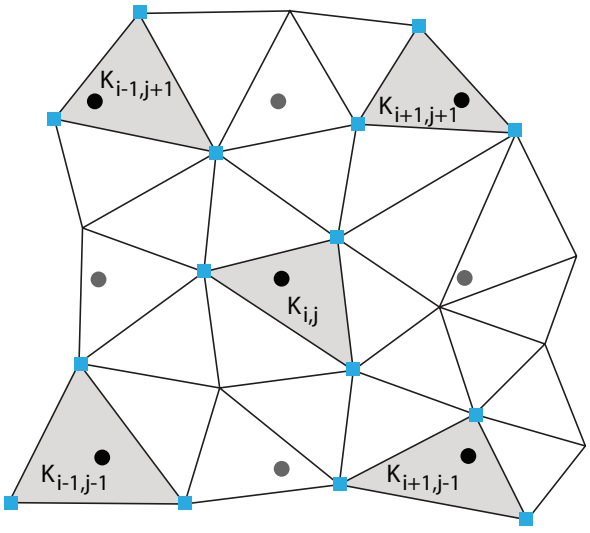
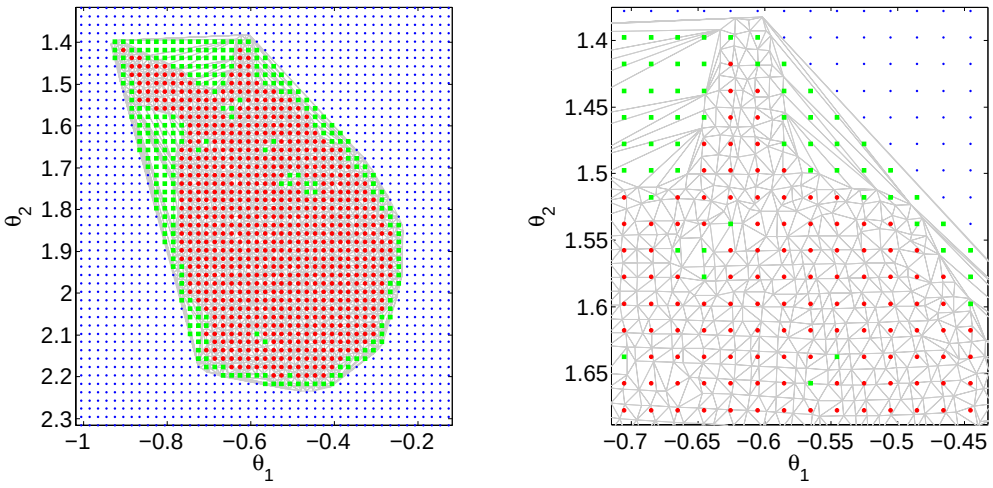


Figure 2.14: Rectangular grid construction: circles – rectangular grid vertices $X_{i,j}$, squares – triangular mesh vertices corresponding to diagonal neighbors of $K_{i,j}$.

Figure 2.15 illustrates the rectangular grid construction for the bunny data set. The grid is based on the triangulation depicted in Figure 2.13. The red circles correspond to the interior set. That is, each node marked by the red circle satisfies the conditions (2.29). The green squares are inside the elongated triangles or do not have four neighbors to satisfy (2.29), and the blue points are entirely outside of the triangulation. Note that most of the triangles along the edges have irregular shapes. Thus we can automatically classify the grid points near the horizons as the points inside the elongated triangles (marked by the green squares in our example).

When mapping the data onto the rectangular grid, we wish to construct a high order interpolation across the discontinuities along curves. Therefore, we use the



(a) (b)

Figure 2.15: Rectangular grid construction and interior set detection based on triangulation of visible points on the surface of the bunny. Circles are in the interior set and have four neighbors to satisfy the criteria (2.29), squares are inside the elongated triangles or do not have a complete set of neighbors to satisfy the criteria (2.29), and points are outside the triangulation. (a) Portion of the grid covering the triangular mesh. (b) Close-up detail of the rectangular grid. The resulting grid size is 158×315.

data clusterings at different directions of the point being interpolated. For each interior grid point $X_{i,j}$ (marked by a red circle in Figure 2.15), we construct four quadratic interpolants:

$$\begin{aligned} f_{++} &\quad \text{using the vertices of} \quad \triangle K_{i,j}, \triangle K_{i+1,j+1} \\ f_{+-} &\quad \text{using the vertices of} \quad \triangle K_{i,j}, \triangle K_{i+1,j-1} \\ f_{-+} &\quad \text{using the vertices of} \quad \triangle K_{i,j}, \triangle K_{i-1,j+1} \\ f_{--} &\quad \text{using the vertices of} \quad \triangle K_{i,j}, \triangle K_{i-1,j-1}. \end{aligned} \tag{2.32}$$

The 6×6 linear systems for the interpolation coefficients are invertible, since the triangles are well separated due to the condition (2.29). Note that the choice of the diagonal neighbors in (2.29) instead of the nearest neighbors allows for a smaller rectangular mesh size h, and therefore, a denser rectangular grid.

Following the ENO philosophy, we choose the interpolating polynomial f_{int} to be the "smoothest" out of the four quadratic polynomials constructed in (2.32). The smoothness of a quadratic polynomial is determined by analyzing its second derivative. In two dimensions, the Laplacian $\Delta f = f_{xx} + f_{yy}$ is a poor choice, since Δf can be small at saddle points, where positive and negative partial derivatives cancel. Instead, we set f_{int} to be one of the four polynomials in (2.32) with the smallest

$$|f_{xx}| + |f_{xy}| + |f_{yx}| + |f_{yy}|.$$

at $X_{i,j}$. Finally, we evaluate $f_{\text{int}}(i,j) = f_{\text{int}}(X_{i,j})$.

A linear interpolation is used to obtain the values on the grid points inside the degenerate triangles with the diameter exceeding $\text{diam}(K_{\max})$. These grid points are marked by the green squares in Figure 2.15. The values at the grid points outside of triangulation (blue points) are set to M, which is a large constant corresponding to infinity or finite sensor range, as in the definition of the visibility function (2.1).

The interpolation procedure can be extended similarly to a standard ENO algorithm to obtain a higher order interpolant instead of quadratic. For example, in order to construct a third order interpolant at $X_{i,j}$, one needs to compare all the possible cubic polynomials whose stencil includes the triangle enclosing the grid node $X_{i,j}$. Through numerical experiments, it has been determined that a quadratic polynomial is sufficient for our purposes.

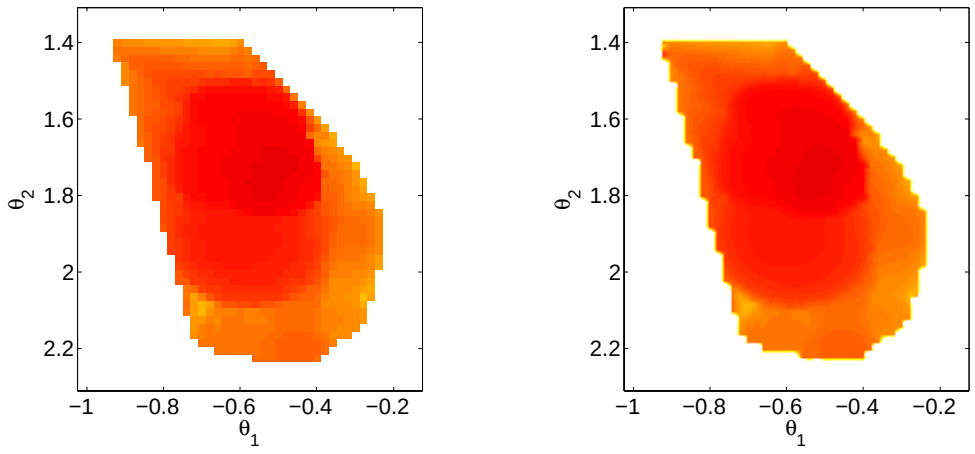
During the next stage of the algorithm, we refine the rectangular grid $\{X_{i,j}\}$ to

obtain a denser grid $\{Z_{\mu,\nu}\} \supset \{X_{i,j}\}$. ENO interpolation is performed dimension-by-dimension using the values of f_{int} on $\{X_{i,j}\}$ to obtain a higher order reconstruction on $\{Z_{\mu,\nu}\}$. The dimension-by-dimension procedure to obtain a p-q ENO interpolant $\rho^{\text{ENO } p,q}$ is the following:

1. For every ν, construct $\rho^{\text{ENO } p,*}(\cdot, \theta_{2_\nu})$ by performing a p-th order ENO interpolation on the selected points from the values of $f_{\text{int}}(\cdot, \theta_2)$.

2. For every μ, construct $\rho^{\text{ENO } p,q}(\theta_{1_\mu}, \cdot)$ by performing a q-th order ENO interpolation on the selected points from the values of $\rho^{\text{ENO } p,*}(\theta_{1_\mu}, \cdot)$.

Figure 2.16 displays the results of the initial quadratic coarse grid interpolation (a), and a fine grid fifth order ENO reconstruction (b). The initial coarse grid size is 131×261, and the refined grid size is 629×1257. Thus we have obtained a piecewise high-order representation of the visibility function on $\mathcal{S}^2$.



(a) (b)

Figure 2.16: (a) Initial coarse grid interpolation. (b) Final fine grid ENO interpolation. Grid is refined by the factor of 4. The order of ENO interpolation is 5.

2.3.2 Level Set Representation

A piecewise polynomial representation of the visibility function can be used to obtain a high-order reconstruction of the visible portion of the occluding surfaces. We use the implicit level set function to represent the final reconstruction on a Cartesian coordinate system. Similarly to (2.8), we define the visibility set

$$G := \{(\theta_1, \theta_2, r) : r < \rho(\theta_1, \theta_2)\}. \tag{2.33}$$

A smooth signed distance function ϕ to the shadow boundary ∂G can be constructed using redistancing [CT07] in three dimensions:

$$\begin{cases} \phi(\theta_1, \theta_2, r) > 0, & \text{if } (\theta_1, \theta_2, r) \in G, \\ \phi(\theta_1, \theta_2, r) < 0, & \text{if } (\theta_1, \theta_2, r) \in G^C, \\ \phi(\theta_1, \theta_2, r) = 0, & \text{if } (\theta_1, \theta_2, r) \in \partial G. \end{cases} \tag{2.34}$$

Finally, we map ϕ to Cartesian coordinates via

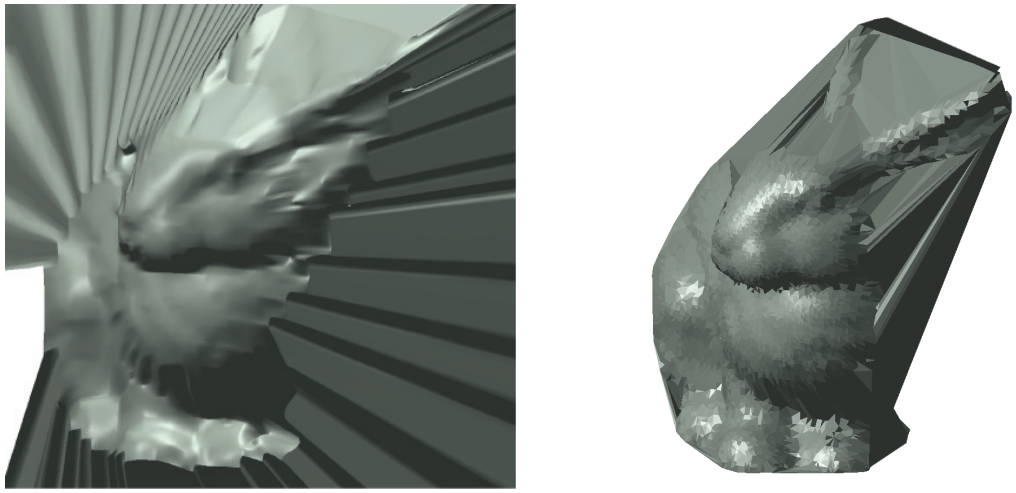
$$\begin{aligned} x(\theta_1, \theta_2, r) &= \phi(\theta_1, \theta_2, r)\cos(\theta_1)\sin(\theta_2) + x_{0_1}, \\ y(\theta_1, \theta_2, r) &= \phi(\theta_1, \theta_2, r)\sin(\theta_1)\sin(\theta_2) + x_{0_2}, \\ z(\theta_1, \theta_2, r) &= \phi(\theta_1, \theta_2, r)\cos(\theta_2) + x_{0_3}. \end{aligned} \tag{2.35}$$

A linear interpolation is used on a grid. Figure 2.17 (a) represents the zero level set of the signed distance function to the shadow boundary of the bunny, corresponding to a point cloud depicted in Figure 2.11. In contrast, Figure 2.17 (b) displays the triangulated surface obtained via mapping to Cartesian coordinates of the initial triangulation from Figure 2.13. One can see that the initial quality of detail is preserved in the level set representation. In addition, a high-order reconstruction of the visible surface is available through ENO interpolation.

Furthermore, the level set formulation allows for easy reconstruction of the joint visibility from multiple viewpoints. The level set surface representations

can be "stitched" together using the boolean operation (2.11). In contrast, it can be very tricky to combine together the triangulated surface patches.

In the next section, we are going to discuss more numerical examples that illustrate the performance of our algorithm.



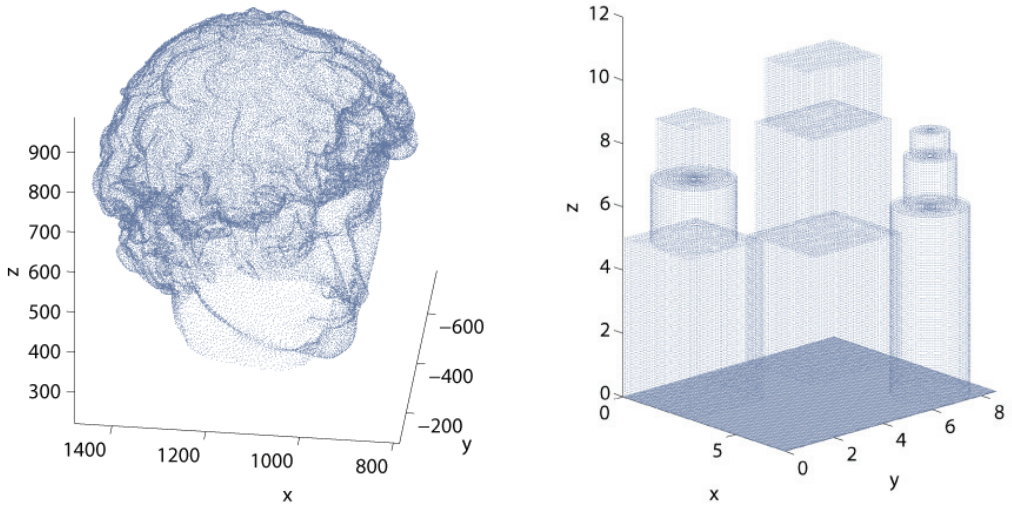
(a) (b)

Figure 2.17: (a) Signed distance function to the shadow boundary. (b) Triangulation of the visible data points.

2.3.3 Numerical Examples

In this section we are going to present two examples of visibility construction out of the three-dimensional point clouds. The first sample data set is a portion of the scan of the Michelangelo's David statue [LPC00]. To better illustrate the fine details of the reconstruction, we only consider the point cloud corresponding to the David's head. It contains 78,958 points. The second point cloud is a simulated city block consisting of 190,704 points. Both sample data sets are displayed in Figures 2.18 (a) and (b). Below we illustrate the steps leading to the

reconstruction of the visible surfaces based on visibility interpolation techniques described in the previous section.

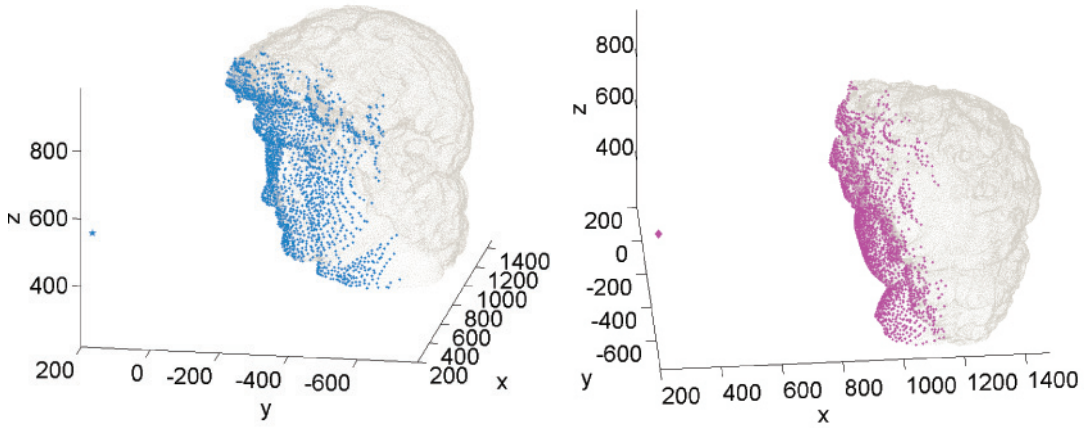


(a) (b)

Figure 2.18: (a) Point cloud of David's head, 78,958 points. (b) Point cloud of urban environment, 190,704 points.

On Figure 2.19 we present the data visible from two distinct vantage points: $(400, 200, 500)$ and $(200, -700, 700)$. Figures 2.20 (a) and (b) display the corresponding initial coarse level interpolations of the visibility functions, while Figures 2.20 (c) and (d) correspond to a fine level fifth order ENO interpolation. The respective fine grid sizes are 521×1041 (a) and 477×953 (b). The grid size has been refined by a factor of four from the initial coarse level interpolation.

The interpolated visibility function $\rho^{\mathrm{ENO}}(\theta_1, \theta_2)$ is used to construct an implicit level set representation of the visible occluding surfaces displayed in Figures 2.21 (a) and (b). In contrast, Figures 2.21 (c) and (d) depict the triangulation of the visible surfaces with vertices at the extracted visible points. One can note



(a) (b)

Figure 2.19: Data visible from the two vantage points: (a) $(400, 200, 500)$ and (b) $(200, -700, 700)$.

that the reconstructions from the top and bottom rows of Figure 2.21 have the same amount of detail. However, the reconstructions in the first row are piecewise smooth, which allows us to compute derivatives on the surfaces away from the discontinuities.

Figure 2.22 (b) depicts the surface obtained by stitching together the two level set reconstructions corresponding to two distinct vantage points. The corresponding visible point cloud is depicted in Figure 2.22 (a.) The joint visibility formula (2.11) is used to obtain the union of the two reconstructions. Note that it would be extremely difficult to combine the triangulations from Figures 2.21 (c) and (d) into a single piecewise smooth surface.

In the next example we are going to consider the point cloud representing an urban environment. Figures 2.23 (a), (b), and (c) display the data visible from three distinct vantage points: $(7.2, 0, 12)$, $(4, 10, 13)$, and $(-2, 4, 13)$. Once the visible points have been filtered out, use them to construct a triangulation

on $\mathcal{S}^2$. The next step is a preliminary coarse level interpolation depicted in Figures 2.24 (a), (b), and (c). The corresponding coarse grid sizes are 95×190, 134×267, and 179×358. Finally, the rectangular mesh is refined to perform the fifth order ENO interpolation. The mesh refinement factor is 4. Fine level interpolants are displayed in Figures 2.24 (d), (e), and (f).

The resulting level set surface reconstructions are depicted in Figures 2.25 (a), (b), and (c). The triangulations based on filtered visible data are presented in Figures 2.25 (d), (e), and (f). Note the sharp corners and flat surfaces of the buildings are preserved by the high order interpolation. Linear interpolation is used to reconstruct the data near the horizon locations.

The surface resulting from combining the three visibility reconstructions is depicted in Figure 2.26 (b). The underlying point cloud is shown in Figure 2.26 (a). The three vantage points turn out to be sufficient to reconstruct all the surfaces of the buildings. Some irregularity of the walls and, especially, the ground is due to expression (2.11) for combining the level set surfaces corresponding to different vantage points.

Sometimes a less accurate region near the horizon with respect to one vantage point replaces a more accurate representation corresponding to a different vantage point, thus diminishing the overall quality of the resulting joint reconstruction. This problem can be avoided by checking the reliability of a given reconstruction, *i.e.* the proximity to a horizon, prior to applying the formula (2.11).

The examples David's head and the city block reconstruction lead to the following questions: How many vantage points are sufficient to "see" the entire environment, and how to choose the vantage points to accomplish the desired visibility? These questions can be addressed as a part of environment exploration problem and will be discussed in the following chapter.

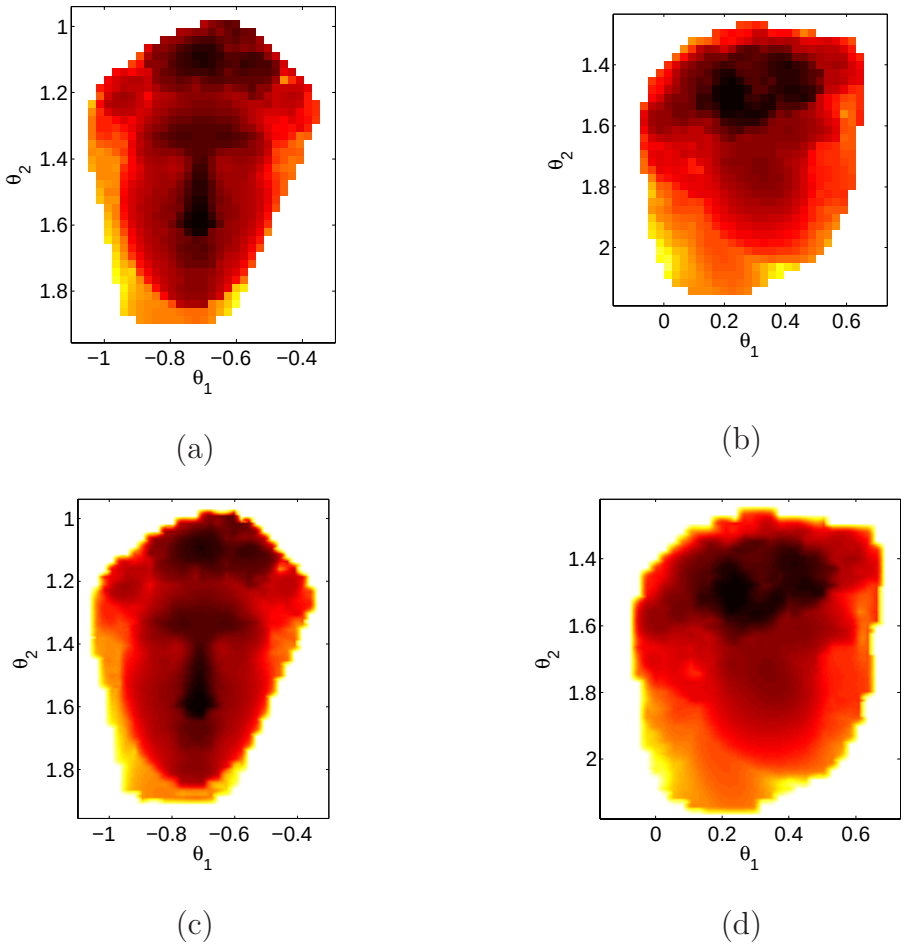
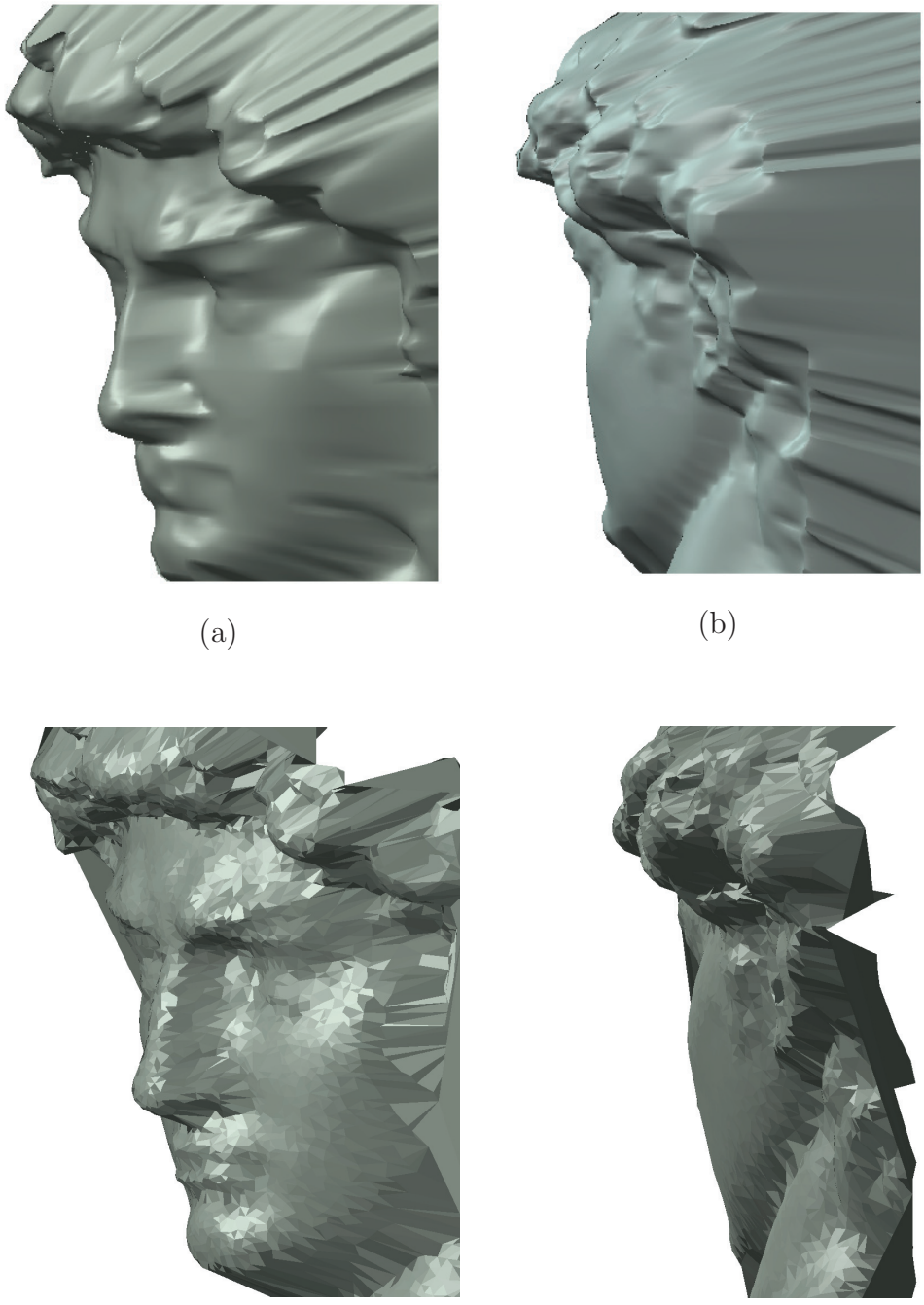


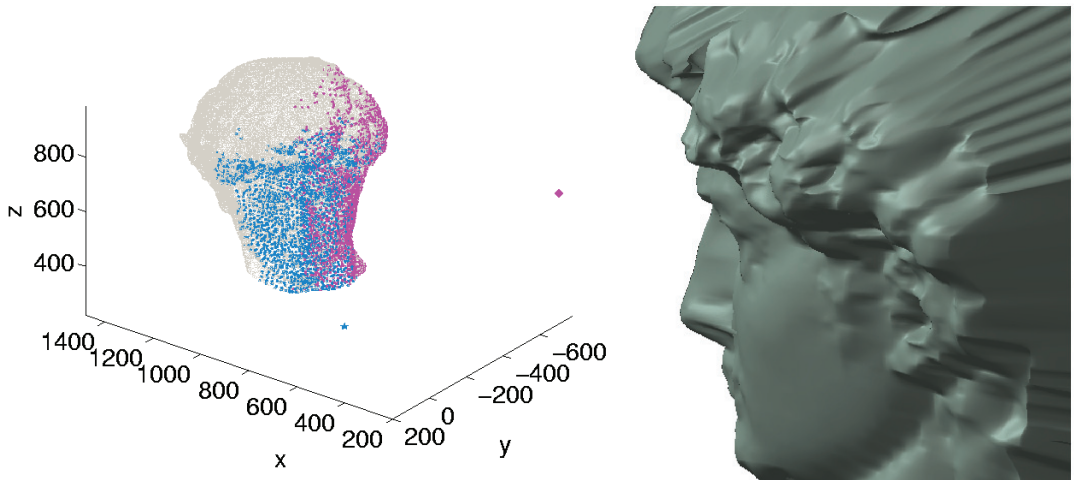
Figure 2.20: Top row: coarse level visibility interpolation corresponding to the vantage point (a) $(400, 200, 500)$ and (b) $(200, -700, 700)$. The grid sizes are 131×261 and 120×240. Bottom row: fine level fifth order ENO interpolation of the visibility function. A mesh refinement factor is 4.



(a) (b)

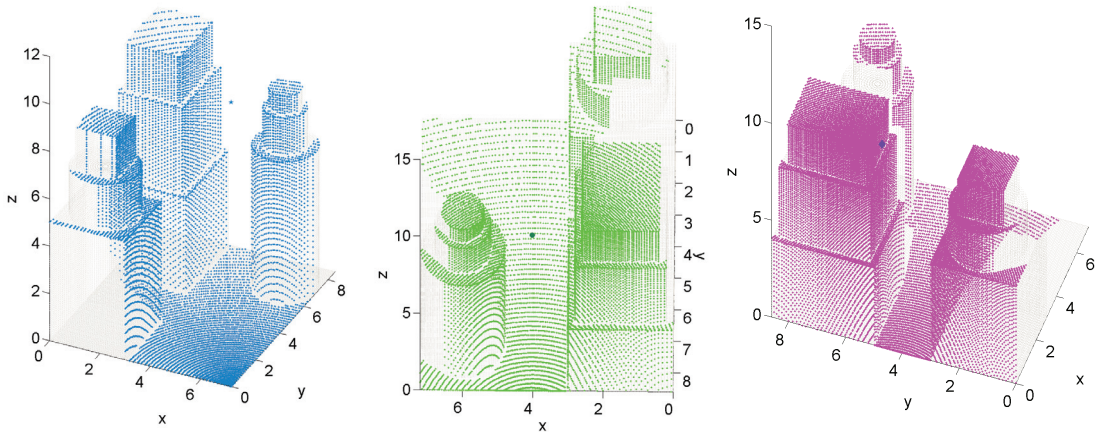
(c) (d)

Figure 2.21: Top row: level set reconstruction of the visible occluding surfaces corresponding to the vantage points (a) $(400, 200, 500)$ and (b) $(200, -700, 700)$. Bottom row: triangulation of the visible data points. The number of triangles used in reconstruction is (a) 2482 and (b) 2154.



(a) (b)

Figure 2.22: (a) Data visible from the two vantage points: $(400, 200, 500)$ (star) and $(200, -700, 700)$ (diamond). (b) Level set representation of joint visibility corresponding to two distinct vantage points.



(a) (b) (c)

Figure 2.23: Data visible from the three vantage points: (a) $(7.2, 0, 12)$, (b) $(4, 10, 13)$, and (c) $(-2, 4, 13)$.

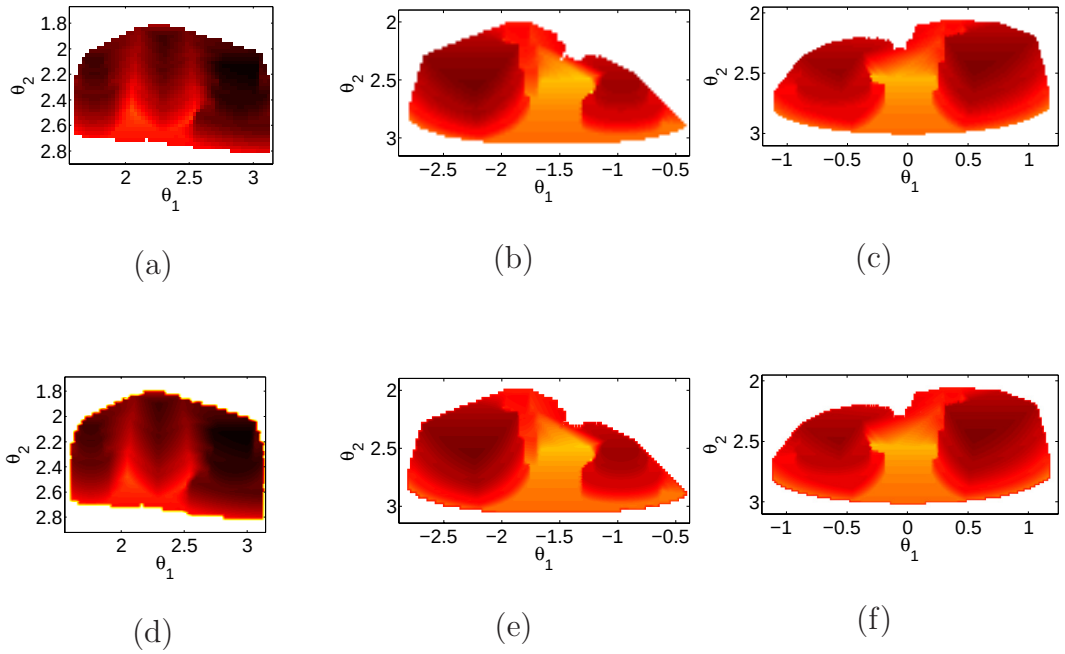


Figure 2.24: Top row: coarse level visibility interpolation corresponding to the vantage points (a) $(7.2, 0, 12)$, (b) $(4, 10, 13)$, and (c) $(-2, 4, 13)$. The grid sizes are 95×190, 134×267, and 179×358. Bottom row: fine level fifth order ENO interpolation of the visibility function. A mesh refinement factor is 4.

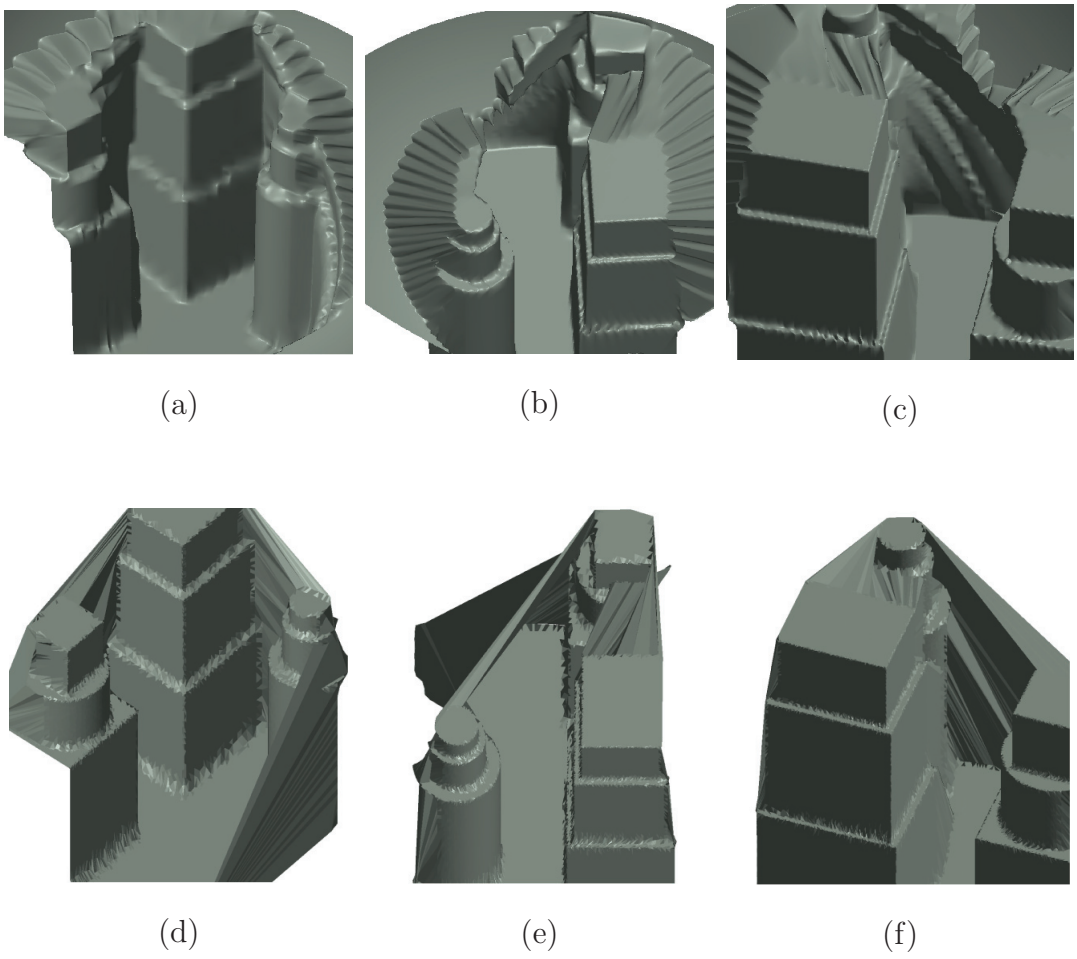
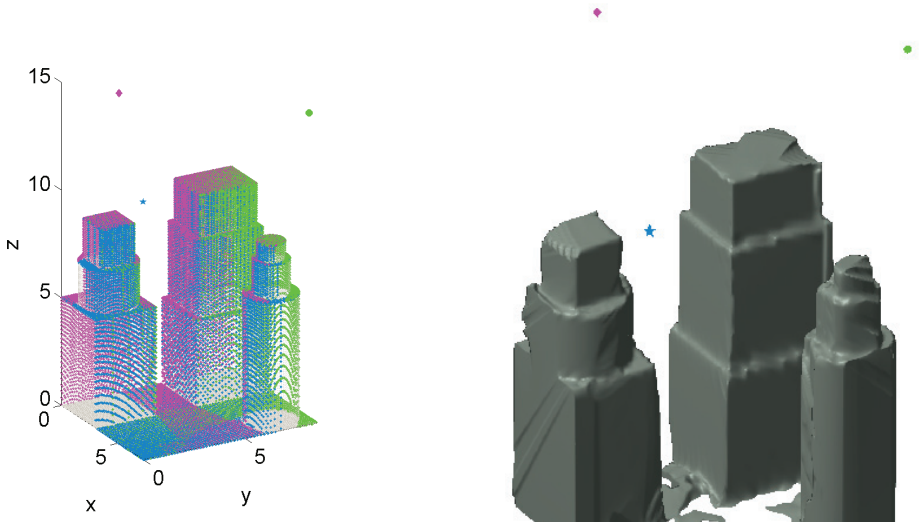


Figure 2.25: Top row: level set reconstruction of the visible occluding surfaces corresponding to the vantage points (a) $(7.2, 0, 12)$, (b) $(4, 10, 13)$, and (c) $(-2, 4, 13)$. Bottom row: triangulation of the visible data points. The number of triangles used in reconstruction is (d) 24807, (e) 27857, and (f) 47444.



(a) (b)

Figure 2.26: (a) Data visible from the three vantage points: $(7.2, 0, 12)$ (star), $(4, 10, 13)$ (circle), and $(-2, 4, 13)$ (diamond). (b) Level set representation of joint visibility corresponding to two distinct vantage points.

CHAPTER 3

Mapping of Unknown Environments

In this chapter we discuss the navigation algorithms based on visibility. In particular, we address the solution to Problem 1.2 defined in Section 1.1, *i.e.* the problem of exploration of unknown bounded environments, which may contain obstacles. Our goal is to obtain an algorithm that would utilize visibility information to allow an autonomous observer(s) equipped with a range sensor to fully explore the region and to map the obstacles' boundaries. The latter refers to the construction of an accurate mathematical representation of the obstacles.

The following constraints on the observer's path ensure that the algorithm is practical in applications:

Constraint 1: The path is continuous and consists of discrete steps.

Constraint 2: The number of steps is finite.

Constraint 3: The total distance traveled is finite.

The motivation for our algorithm comes from the work of S. LaValle, B. Tovar *et al.* [TGL05, LaV06, TML07] described in detail in the introductory Sections 1.2.1 and 1.3.2. At each step of the navigation algorithm, the observer randomly chooses to approach one of the gaps, *i.e.* depth discontinuities projected onto $\mathcal{S}^1$. The visibility map, represented by the dynamic data structure, a Gap Navigation Tree, is then updated as a result of gap critical events. The

process is repeated until the entire region has been explored, that is, there are no more unexplored gaps left. As a result of exploration, the region is characterized by the number of gaps and their relative positions. No distance or angular information is accumulated.

Quite the opposite, the algorithm presented in this chapter maps the obstacles in Cartesian coordinates as the observer proceeds through the environment and utilizes the recovered information for further path planning. At the termination of the path all the obstacles' boundaries are reconstructed. Thus, a complete representation of the environment is obtained.

In contrast to our simple discrete approach, a practical implementation of the algorithm from [TML07] requires a constant gap tracking. Furthermore, a wall-following procedure needs to be implemented to navigate the robot in the environment. Additional modifications of the algorithm are required when dealing with multiply connected environments, *i.e.* markings of the visited gaps. Meanwhile, our algorithm does not require any special treatment of certain types of environments.

In addition, the strategy proposed in this thesis is easily scalable to allow for multiple observers. In [LGH07], the algorithm has been validated on a group of autonomous micro-cars. The vehicles, equipped with range sensors, have been used to explore an unknown bounded region and construct the map of the explored environment.

The organization of the rest of the chapter is as follows. Section 3.1 provides the description of the navigation algorithm for a single and multiple observers in unknown planar environments. Section 3.2 discusses the experiments with multiple autonomous vehicles. In Section 3.3, the optimization techniques are applied to the constructed paths, to obtain a more uniform illumination of the

explored region. Finally, Section 3.4 introduces the complexity estimates of the paths in two and three dimensions.

3.1 Horizon-chasing

Consider a bounded region which may contain an unknown number of arbitrary positioned obstacles of unknown general shapes. Our objective is to construct a path for an observer, so that at the termination of the path the observer has seen the entire domain. In addition, a map of the explored region representing the boundaries of obstacles would be constructed. The navigation algorithm is designed with the consideration of handling general geometries.

The intuition behind our algorithm is the following. Assume some portions of the obstacles boundaries are visible to the observer from a given vantage point. Each continuous portion of the visible boundary terminates with the horizon points, or edges on the visibility map, as in Figure 2.4. These horizon points are similar to an edge of the door that is ajar. One must proceed beyond the edge of the door to see more. Similarly, an observer must proceed beyond the horizon point to gain new information about the environment.

At this point, we need to decide how far the observer should march beyond the chosen horizon point. Our strategy relies on the geometry of the obstacle near the horizons. Briefly, if the obstacle is a simple circle, then naturally the march distance should depend on the radius of the circle; this corresponds directly to the curvature at horizon points. These simple insights allow us to construct a path consisting of discrete steps.

In the following subsections we are going to describe our horizon-chacing algorithm for a single observer and its extension to the case of multiple observers.

We also provide the results of navigation simulations in sample environments. In Subsection 3.1.2, we provide the statistics of our algorithm for some general types of environments.

3.1.1 Single Observer

Below we discuss the navigation algorithm for a single observer operating in two dimensions. The details are provided in Algorithm 3.1 below. The key idea behind our algorithm is to proceed in the environment Ω by approaching one of the currently visible horizons. When a new horizon appears, it is stored in a list L. Once the horizon has been explored, it is removed from the list. The observer must explore every horizon in the list before the algorithm terminates. The observer is allowed to return and inspect previously skipped horizons if no more new horizons are available, as done in step 10 of the algorithm. The exploration is complete once there are no more horizons left to approach. The following discussion in this subsection applies to Algorithm 3.1.

First, we are going to address the choice of the next horizon to approach in step 7. In all our numerical experiments, the observer always approaches the nearest previously unexplored horizon, unless stated otherwise. Intuitively, this choice would minimize the total length of the path. The convergence proof in Section 3.4 is based on the nearest edge approach.

However, other choices may be more applicable under different circumstances. For example, one may choose to approach a random horizon as was done in [TGL05], or a horizon with the largest curvature κ (so that the overshoot step size r_2 is the smallest). In [LGH07], the choice of the next horizon is dictated by the specifics of sensor design: to minimize the errors produced by the sensors in the experiments, it is always preferable to navigate around the objects in

Algorithm 3.1 Navigation in planar environment by single observer

$k = 0$, L: list of unexplored edges, initially empty

repeat

  ρ_{x_k}: visibility function corresponding to vantage point x_k

  update the map Ξ of the explored region {Ξ was defined in (2.3)}

  find all the edges/horizons on the $(\theta, \rho_{x_k}(\theta))$ map

  if edge is found **then**

    choose θ_e – the edge to approach {choice depends on the problem}

    store the rest of the edges in a list L

    remove those edges from L that are currently visible

  else {no edges found}

    pick an edge θ_e from the list of unexplored edges L

    backtrack x_k to one of the previous positions corresponding to θ_e

  end if

  if $\rho_{x_k}(\theta_e) < \rho_{x_k}(\theta_e + \delta)$ **then**

    choose the direction $\Theta = \theta_e + \delta$

  else

    choose the direction $\Theta = \theta_e - \delta$

  end if {$\delta > 0$ is chosen to allow a buffer between the observer and obstacles}

  $x_{k+\frac{1}{2}}$ is obtained by moving from x_k in the direction Θ by amount

$$r_1 = \min\{\rho_{x_k}(\Theta), \rho_{x_k}(\theta_e) - \tan\left(\frac{\pi}{3}\right)\frac{1}{\kappa_e}\}$$

  {κ_e: curvature near the edge θ_e (if $\kappa_e \equiv 0$, shift x_k by small amount)}

  $\rho_{x_{k+\frac{1}{2}}}$: visibility function corresponding to $x_{k+\frac{1}{2}}$

  update the map Ξ of the explored region

  remove those edges from L that are currently visible

  x_{k+1} is obtained by overshooting from $x_{k+\frac{1}{2}}$ by

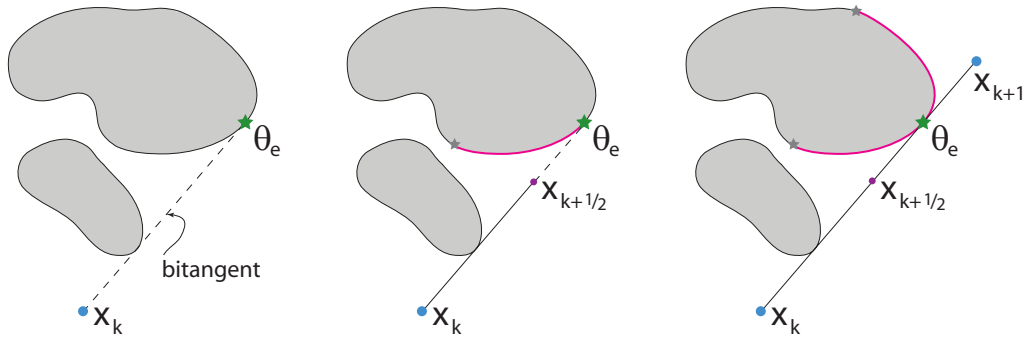
$$r_2 = \min\{\rho_{x_k}(\Theta) - r_1, 2\tan\left(\frac{\pi}{3}\right)\frac{1}{\kappa_e}\}$$

until $L \neq \{\emptyset\}$

the counterclockwise fashion. Thus, the observer always prefers to approach the right-most edge of the obstacle. The details of the experiment will be discussed in Section 3.2.

In step 19 of Algorithm 3.1, we define an intermediate position $x_{k+\frac{1}{2}}$. The main motivation for this additional step is to have a homogeneous coverage of the obstacles' boundaries. By keeping the observer a uniform distance away from the boundary, we are able to obtain the same level of detail everywhere in the region. Additionally, the step $x_{k+\frac{1}{2}}$ is motivated by the convergence proof in Section 3.4.

An alternative algorithm would be to introduce an intermediate position $x_{k+\frac{1}{2}}$ *only* when approaching a horizon through a bitangent, as depicted in Figure 3.1. Note that if the step $x_{k+\frac{1}{2}}$ is omitted, then the entire portion of the obstacle's boundary revealed in Figure 3.1 (b) would remain occluded. We use this modified version of Algorithm 3.1 to obtain all the results in current chapter.



(a) (b) (c)

Figure 3.1: Approaching a horizon through a bitangent: (a) horizon θ_e visible from x_k, (b) intermediate step $x_{k+\frac{1}{2}}$ to reveal a previously occluded portion of the obstacle's boundary, (c) complete the move at x_{k+1}.

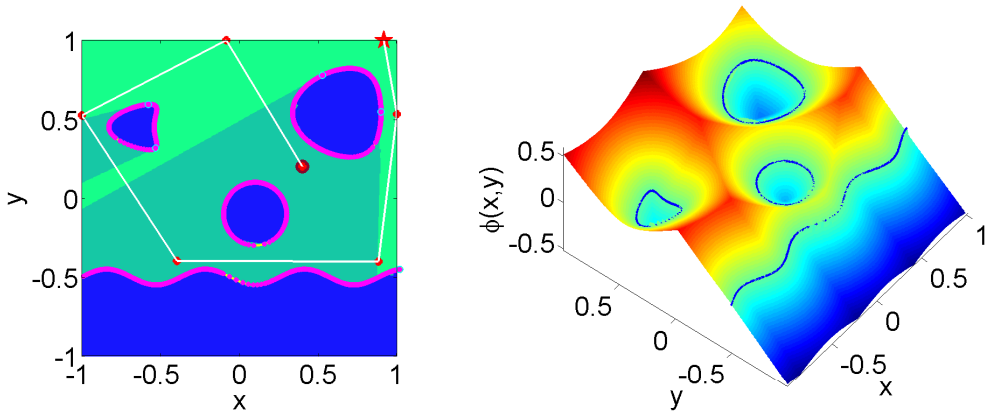
The next position x_{k+1} of the observer, is obtained in step 23 of the navigation algorithm by proceeding beyond the horizon point in the chosen direction Θ. We

choose the overshoot step size to be inversely proportional to the curvature κ_e of the obstacle's boundary near the horizon. On a simple example of a disk-shaped obstacle, the size of the overshoot step is the half the length of an equilateral triangle circumscribing the disk. This particular choice would allow the observer to explore the entire disk's boundary in just three steps. The overshoot step size will be further examined in Section 3.4. High order ENO interpolation allows to compute the curvature at the obstacle's boundary with desired accuracy.

Finally, note that the chosen direction Θ in steps 15 and 17 is not exactly the direction of the horizon θ_e. A small buffer of size $\delta > 0$ is added to provide extra space between the observer and the obstacle's boundary. The buffer size may depend on the observer's size and mobility. If application allows, δ may change in the process of exploration. For example, a smaller δ would allow the observer to explore narrow regions of high curvature. When the curvature of the occluding surface is large, a larger buffer would be more suitable.

As a result of the navigation algorithm, we obtain a complete map of the environment, *i.e.* polynomial interpolated boundaries of the obstacles along with the visibility indicator function Ξ which marks the interior and exterior of the obstacles. In addition, one may easily construct the level set representation of the reconstructed environment map via (2.11). The use of the level set maps in postprocessing algorithms will be presented in Section 3.3. Further applications of the level set representation of visibility are described in [TCO04] and [CT05].

In Figures 3.2, 3.3, and 3.4 we demonstrate the paths generated using Algorithm 3.1 in different environments. The left sub-figures (a) depict observer's paths in various environments along with the visibility map at the final step of the exploration. The right sub-figures (b) represent the signed distance to the obstacles' boundaries.



(a) (b)

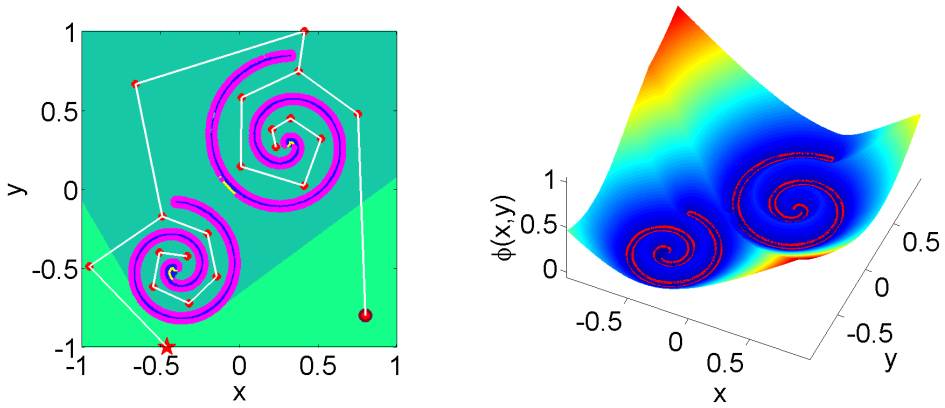
Figure 3.2: Three non-overlapping shapes and a sine wave. (a) Exploration path and visibility map at the final step: dark circle – initial position, star – final position, white line with circles – observer's path steps. (b) Signed distance to occluding boundaries.

In Figure 3.2 (a), the test environment consists of three disjoint obstacles, one of which is not convex, and a sine wave. The observer is able to explore the environment in seven discrete steps without having to backtrack and clear previously unexplored edges. The starting position is $(0.4, 0.2)$ and the final position is $(0.9, 1)$. The intermediate steps are marked by circles.

In contrast, in Figures 3.3 and 3.4, the observer must backtrack to one of its previous locations to clear previously unexplored edges. In the case of two spirals in Figure 3.3 (a), the observer starting at $(0.8, -0.8)$, first explores the interior of one spiral, then returns to proceed inside the second spiral. The path terminates at $(-0.46, -1)$.

The environment in Figure 3.4 (a) is generated by taking a level set of a portion of the Grand Canyon elevation data[1]. The path runs begins at $(0.8, -0.95)$ and

[1]The terrain data were obtained from
`ftp://ftp.research.microsoft.com/users/hhoppe/data/gcanyon/`

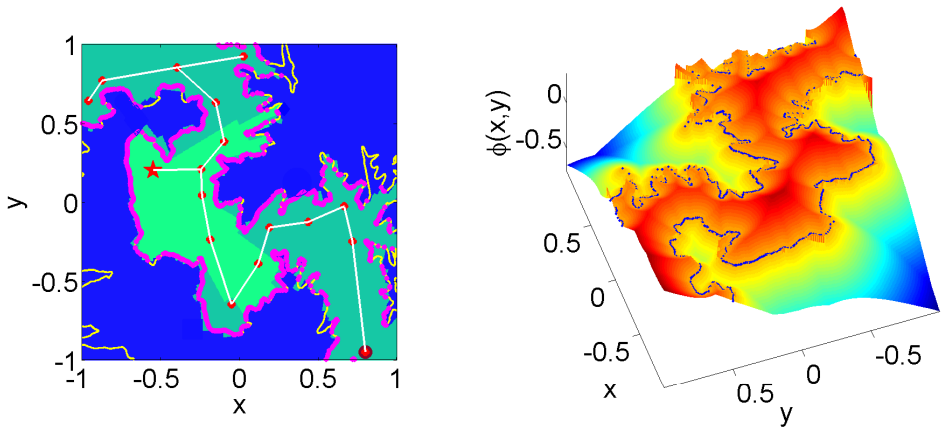


(a) (b)

Figure 3.3: Two spirals. (a) Exploration path and visibility map at the final step: dark circle – initial position, star – final position, white line with circles – observer's path steps. (b) Signed distance to occluding boundaries.

terminates at $(-0.55, 0.2)$. The non-uniform change of curvature of the visible boundary makes it difficult to explore the fractal-like portions of the boundary. A constant parameter δ (introduced in step 14 of Algorithm 3.1), which controls how close an observer may approach an obstacle, does not allow for a more detailed exploration of the narrow regions. As this parameter may depend on the physical size of the observer, the result is an illustration of a realistic exploration outcome. If the size and mobility of the observer allows for it, an algorithm with δ varying as a function of curvature may be implemented. Another approach would be to utilize optimization techniques to construct a new path based on the results of the initial "rough" exploration as in Section 3.3.

Furthermore, the sparsity of the data in the initial point cloud does not allow us to choose the scanning fan size $\delta\theta$ small enough to satisfy the Assumption 2.1 in Subsection 2.2.6. As a result, some of the horizons are not distinguished by the edge-detection function (2.4).



(a) (b)

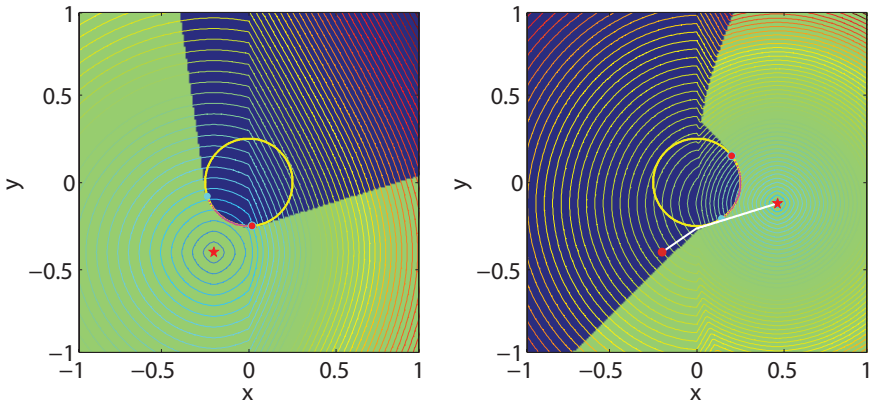
Figure 3.4: Grand Canyon terrain. (a) Exploration path and visibility map at the final step: dark circle – initial position, star – final position, white line with circles – observer's path steps. (b) Signed distance to occluding boundaries.

The above examples illustrate the performance of the exploration Algorithm 3.1 in simulated environments of increasing complexity. Application of the algorithm in the "real world" environment will be described in Section 3.2.

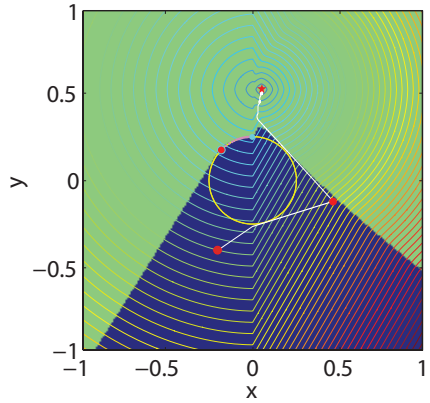
We would like to remark that Algorithm 3.1 can also be applied towards environment exploration in case of curved lines of sight, with the following modifications. Since the curvature of the occluding surfaces cannot be recovered from the visibility function ρ constructed in Subsection 2.2.4, the overshoot step-size must be defined by the user in steps 19 and 23. To proceed further from the edge, we follow the line of sight passing through this edge by solving

$$\begin{aligned} \dot{x} &= \nabla\varphi(x), \\ x|_{t=0} &= x_e. \end{aligned} \tag{3.1}$$

where x_e is the position of the edge and $\varphi(x)$ is the distance to the observer resulting from solution of the eikonal equation (2.6). Consider Figure 3.5 for a sample step-by-step path. The index of refraction r is set in Subsection 2.2.4.



Stage 1 Stage 2



Stage 3

Figure 3.5: Stages of exploration under a bending ray field.

A similar approach would be effective in three dimensions. At each vantage point, after the reconstruction of the visible surface, we carve out the region, which can be considered as a reliable reconstruction. One way to classify the reliable regions is to analyze the triangulation, used in the initial step of the reconstruction process, see Section 2.3. The irregular elongated triangles correspond to the regions near the horizon locations. Linear interpolation is used to reconstruct the surface near the horizons and, therefore, may not be as reliable as the high order reconstruction away from horizons. In Figure 2.14, the rect-

angular grid nodes corresponding to the interior set (red circles) represent the reliable regions.

When the preliminary rectangular mesh is refined, we obtain a tube of points in the neighborhood of horizons. The points inside this tube would then be stored in a three-dimensional edge map. Once the direction of the of the next vantage point is chosen, the observer would fly pass this direction, while preserving some buffer space around the obstacle. The exploration is considered to be complete when the edge map is empty.

The choice of the next direction may depend on the proximity of an edge point to the observer, as in our two-dimensional algorithm. Alternatively, the observer may choose to fly pass the direction with the largest normal curvature.

Finally, we would like to remark that surface curvatures may not be accurately computed near the horizons, and thus may not be used to estimate the overshoot step size as was done in two dimensions. A parameter must be chosen to determine the overshoot step size in three-dimensional exploration algorithm.

In Subsection 3.4.2, we provide an estimate by L.-T. Cheng on the number of steps required to explore a single convex shape in three dimensions.

3.1.2 Statistics

Below, we compare the histograms of paths lengths and the number of steps required to explore a sample environment using two different exploration algorithms: Algorithm 3.1 and the random walk exploration. In the random strategy, the next position of the observer is randomly chosen in the currently visible region. New horizons are being detected and stored similarly to Algorithm 3.1. The exploration terminates when there are no more unexplored horizons left.

Figure 3.7 depicts the statistics of environment exploration simulations using Algorithm 3.1 and that using a random walk strategy, which serve as control experiments. The statistics is collected from 1000 independent runs. A sample environment is generated by an arbitrary placement of twelve disjoint non-convex obstacles in the 2×2 bounded region. One such environment configuration is depicted in Figure 3.6. The initial position of the observer is chosen randomly, however, is constrained to be outside of the obstacles.

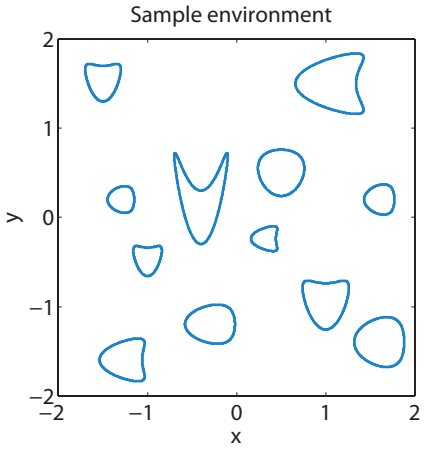


Figure 3.6: Sample environment for statistics experiment.

Histograms of the total number of steps required to explore the environment are presented in Figures 3.7 (a) and (b). One can see that Algorithm 3.1 requires no more than 26 steps to explore the environment. Furthermore, the exploration most frequently terminates in 19 or 20 steps. The minimum number of steps required to explore this type of environment is 11.

In the simulations using the random walk strategy mentioned above, a limit of 400 steps is imposed, regardless of whether the environment has been entirely explored or not. The red portions of the histograms in Figures 3.7 (c) and (e) correspond to the experiments terminated at 400 steps. One can see that about a quarter of experiments terminate before the entire region has been explored.

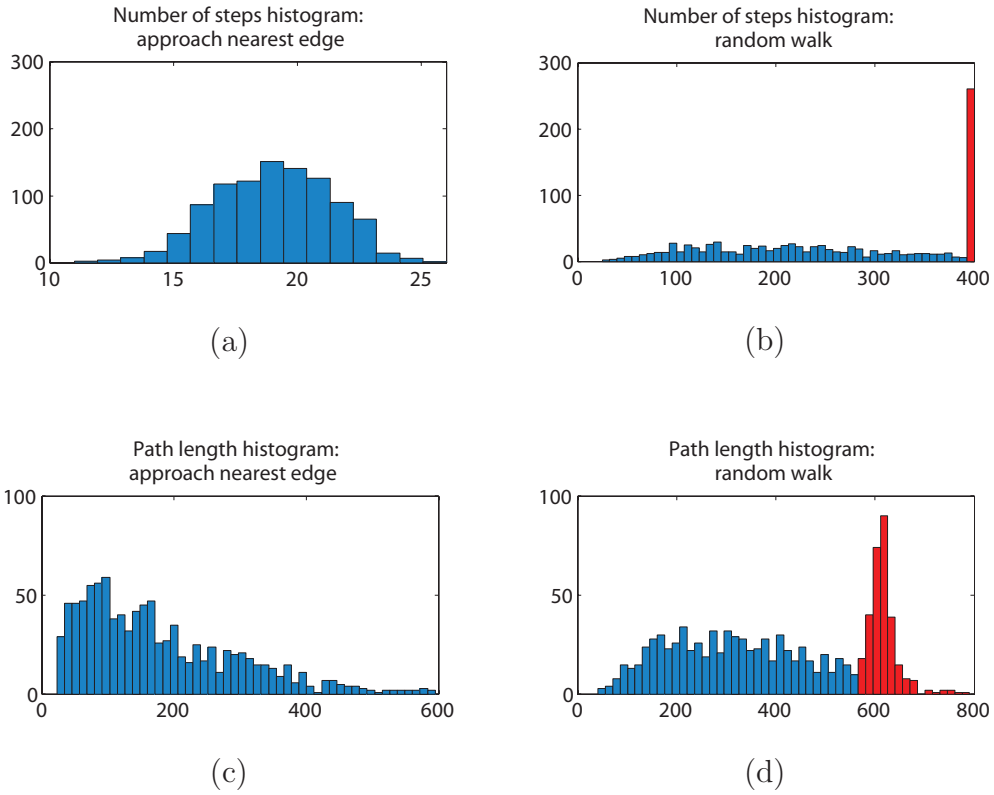


Figure 3.7: Statistics experiment: (a) number of steps histogram for Algorithm 3.1; (b) number of steps histogram for the random walk; (c) path length histogram for Algorithm 3.1; (d) path length histogram for the random walk. The simulation of random walk is stopped if the step count is greater than 400.

Path lengths histograms are depicted in Figures 3.7 (c) and (d). The path lengths corresponding to Algorithm 3.1 range between 23 and 600, with most path lengths between 100 and 200. The path lengths corresponding to the random walk approach range between 42 and 800, peaking near 600. This peak corresponds to the experiments which terminated after 400 steps before the environment has been entirely explored.

Note that if the curvature changes its sign $2m$ times along the boundary of

a single star-shaped obstacle, the sufficient number of steps required to explore the entire boundary is $3 + m$. Here, 3 steps are needed to see the convex hull of the object and a maximum of m steps would be enough to explore each concave part. The complexity estimates for our algorithm are discussed in detail in Subsection 3.4.1. For now, we would like to remark that if each object in the region is treated independently, the sufficient number of steps required to explore a region with n disjoint star-shaped obstacles is $n(3 + m)$. For our particular experiment setup, this amounts to a maximum of 48 steps. Note that the Algorithm 3.1 allows to explore the entire environment in half as many steps.

Clearly, Algorithm 3.1 provides a superior strategy for environment exploration comparing to the random walk approach. The statistics also provides an estimate on path length and the number of steps required to explore the region with twelve non-convex obstacles. In Subsection 3.4.1 we prove that our proposed algorithm would always terminate in finite number of steps in any bounded region containing an arbitrary finite number of disjoint convex obstacles.

3.1.3 Multiple Observers

The extension of the navigation algorithm for multiple observers is straightforward. Let $\{x_j\}_{j=1}^n$ be a set of observing locations. Similarly to (2.3), define the visibility indicator $\Xi_j(y) := \rho_{x_j}\big(\nu(y)\big) - |y - x_j|$, such that $\{\Xi_j \geq 0\}$ is the set of visible regions and $\{\Xi_j < 0\}$ is the set of invisible regions with respect to x_j. In addition, let $\Theta_j = \{\theta_{j,1}, \ldots, \theta_{j,k}\}$ be the set of edges visible from the vantage point x_j. The Algorithm 3.2 below describes the strategy for environment exploration with multiple observers.

Note that in step 6 of Algorithm 3.2 we exclude those edges corresponding to x_j, which are visible by another observer x_i and thus do not need to be further ex-

Algorithm 3.2 Navigation in planar environment by multiple observers (based on Algorithm 3.1)

1: N: number of observers
2: x_j: vantage points outside the occluding objects, $j = 1, ..., N$
3: ρ_{x_j}: visibility function corresponding to x_j
4: compute $\Xi = \max_j\{\Xi_j\}$
5: find all the edges/horizons Θ_j corresponding to each observer x_j
6: exclude those $\theta_{j,k}$ for which $\Xi \geq 0$
7: **if** found an edge **then**
8: proceed as in Algorithm 3.1 for each individual observer
9: **else** {no edges found}
10: move observer at x_j in the direction orthogonal to the direction of the nearest x_i to see new edges;
11: **end if**
12: proceed as in Algorithm 3.1 until no more new edges

plored. This step is illustrated in Figure 3.8. At the current stage of exploration,

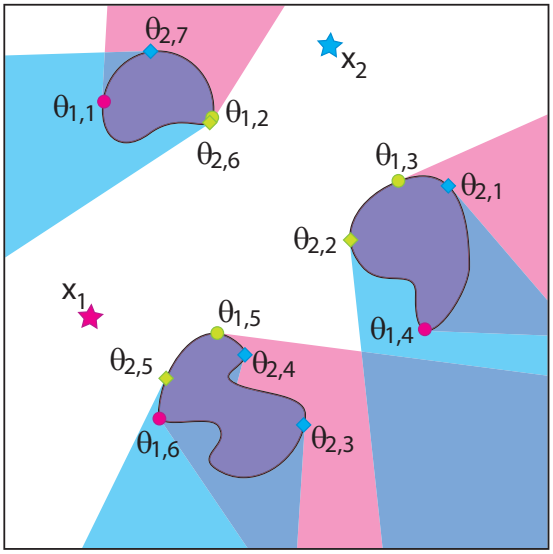


Figure 3.8: Joint visibility of two observers. Visible horizons, $\theta_{1,2}$, $\theta_{1,3}$, $\theta_{1,5}$, $\theta_{2,2}$, $\theta_{2,5}$, and $\theta_{2,6}$, are removed from the list of unexplored horizons.

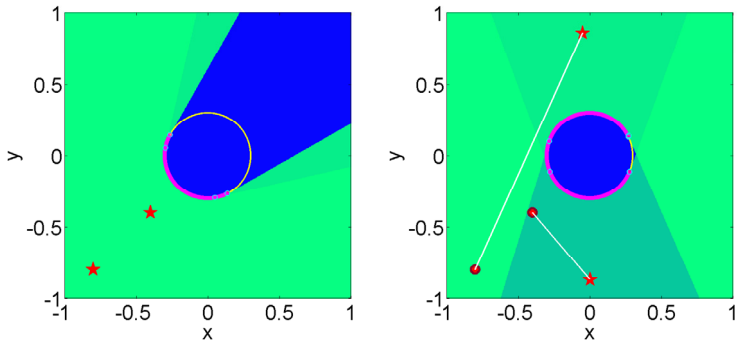
there are six horizons corresponding to the observer at x_1 and seven horizons corresponding to the observer at x_2. However, not all of them need to be further explored. For example, the edges $\theta_{1,2}$, $\theta_{1,3}$, and $\theta_{1,5}$, detected by x_1 are visible from x_2 and hence, are removed from the list of unexplored edges. Similarly, we remove the edges $\theta_{2,2}$, $\theta_{2,5}$, and $\theta_{2,6}$, corresponding to x_2, since they are visible from x_1.

Before we proceed to discuss the examples, we would would like to remark on different modes of execution of Algorithm 3.2. In *concurrent mode* all observers process sensor data simultaneously. This way, the next vantage point of each observer depends only on their previous positions. In *sequential mode* the observers are ordered as a sequence, and only one may move at a time. In this situation, the position of the next observer depends on new positions of the previous observers. The ordering may change according to the decision to optimize joint visibility. In

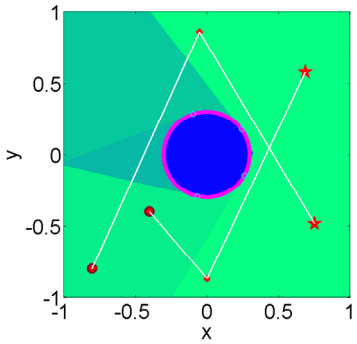
some applications the concurrent mode would be more desirable, since this mode allows for more autonomous maneuver for each observer. Of course, in practice, the usage of one mode or switching from one to the other depends on the data communication model as well as the routing algorithm. In the experiments in [LGH07] as well as the numerical simulations below, the concurrent mode has been implemented.

The orthogonal move in step 10 is chosen to maximize the chance of "seeing" more new area. For example, the two observers in Figure 3.9 are initially positioned so that the first observer at $(-0.4, -0.4)$, which is closer to the obstacle, does not see any new horizons that are invisible to the other observer at $(-0.8, -0.8)$, which is farther away. In this situation, the first observer makes a move in the direction orthogonal to the second observer, according to step 4 of Algorithm 3.2. This step is illustrated in Stage 2 of Figure 3.9. It takes three steps by each of the two observers to explore the region with a single obstacle. Such a result could still be achieved by a single observer. Thus, in a simple environment depicted in Figure 3.9, there is no advantage in the use of multiple observers as opposed to a single observer. However, multiple observers prove to be much more efficient in more complex environments, as can be seen from the following examples.

In Figure 3.10 we have two observers in a more complex environment consisting of three shapes with non-convex boundaries. Two observers explore such an environment in four steps. Finally, in Figure 3.11, we have three observers in the environment with four circles. This time it takes only three steps to complete the exploration.



Stage 1 Stage 2



Stage 3

Figure 3.9: Stages of environment exploration with two observers, obstacle: a circle. Dark circles – initial position's, stars – final positions, white lines with circles – observers' path steps.

3.2 Experimental Results: Robotic Path Planning with Limited Sensor Data

In this section we present an implementation of the path planning Algorithm 3.2, which allows a group of autonomous vehicles equipped with range-sensors (observers) to explore an unknown bounded region and construct the map of the explored environment. The results of this experiment were initially introduced in [LGH07]. To test the robustness of our algorithm, we consider the problem

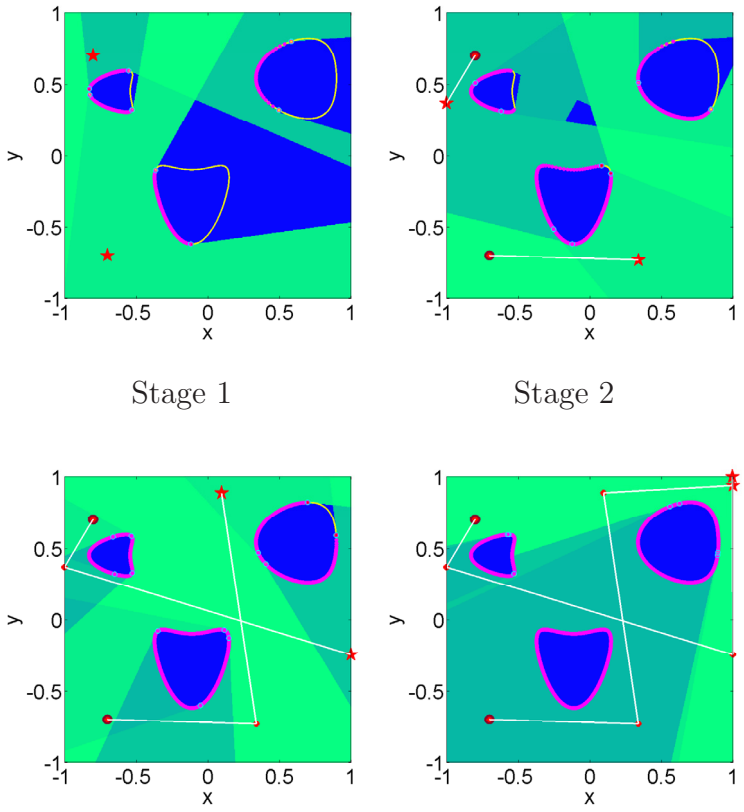
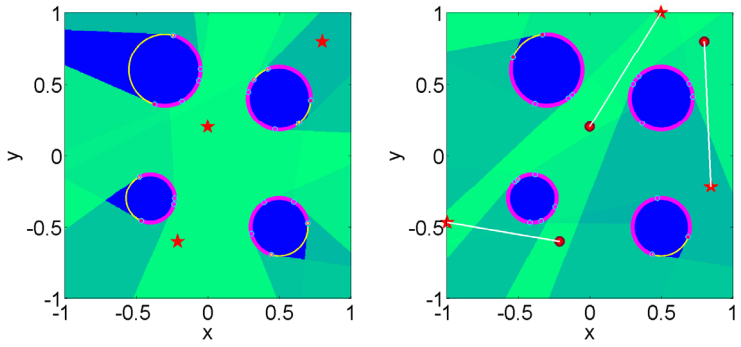


Figure 3.10: Stages of environment exploration with two observers, obstacles: three shapes. Dark circles – initial position's, stars – final positions, white lines with circles – observers' path steps.

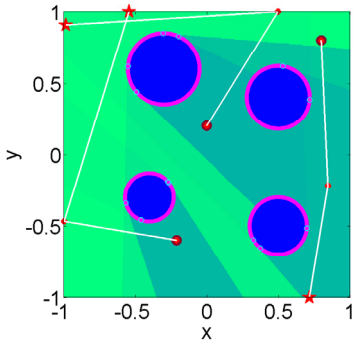
of mapping an unknown environment using multiple mobile inexpensive sensors where noise is an issue.

In [TLM03b], the Gap Navigation Tree based algorithm [TML07] was tested on a Pioneer 2-DX platform equipped with two SICK laser range sensors, which provide an omnidirectional view. The gap sensor implementation combined the data of these two sensors. Simple test environments were chosen to be within the sensor range. Wall-following capabilities were implemented to avoid collisions.

Another wall-following control algorithm is discussed in [ZOL04]. In this



Stage 1 Stage 2



Stage 3

Figure 3.11: Stages of environment exploration with three observers, obstacles: four circles. Dark circles – initial position's, stars – final positions, white lines with circles – observers' path steps.

work, curvature-based control algorithms from [ZJK03] are tested using real range sensors. Curvature is computed from the range data obtained by SICK LMS-200 laser range sensors. Unlike range sensors used in our experiment, the range-finder in [ZOL04] has a range of 10 m and relative error less then 0.8%.

Even with such a high precision, curvature estimates have significant inaccuracies in the absence of filtering. The noise in curvature computations is related to the computation of derivatives of the range data which are prone to noise. To deal with this problem, ENO interpolation is used in Algorithm 3.2, to obtain

a high order representation of the range data, so that derivatives can be easily estimated away from discontinuities.

In another work [GCB06], a multiple vehicle cooperative control algorithm is described. The model problem is extended from the classical Art Gallery Problem [Urr00]. Here, each robot must find a location in a non-convex polygonal environment, so that each point of the environment is visible to at least one robot. In [GCB06], the visibility-based deployment problem is solved under the assumption that all the vehicles are initially collocated.

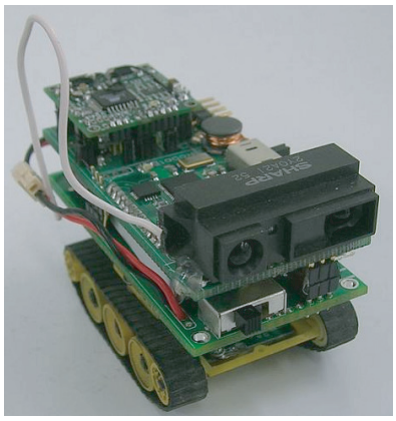
3.2.1 Test-bed and Range Sensors

The results in this experiment were obtained using the second generation [LHH07] of an economical micro car test-bed developed in [HCH06]. The purpose of the test-bed is to design a cost effective platform to study cooperative control strategies. The dimensions of the test-bed floor are 200×160 cm. The second generation vehicles communicate at 30 Hz and possess onboard processing and onboard range sensing. Tank-based vehicles with caterpillar-style drive are used to allow for a zero turning radius. The tank has dimensions $7 \times 3.8 \times 4.6$ cm and weighs 65 g with batteries. Such a tank is depicted in Figure 3.12 (a).

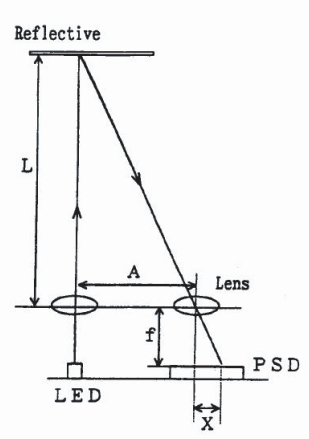
The position of the vehicles is tracked by overhead cameras. An off-board computer is used for communication with the overhead cameras and for processing sensor data from the vehicles. All the basic motion maneuver, sensor acquisition, and communication routine is processed onboard by a 16 MHz Atmel (Atmega 8) microprocessor. The tank drives two belts independently, resulting in turns of arbitrary radius, while moving forward and backward. One can obtain more details about the test-bed and the vehicles in [LHH07].

Now we shall describe the range sensors used in our experiments. We work

with sensors manufactured by Sharp (model 2YOAO2 F58) of range 20–150 cm. The sensors are equipped with a PSD onto which the light is focused. IR EM radiation is emitted via LED at the front of the sensor. The wavelength range in use is 850 ± 70 nm. The half-intensity angle of the device is 1.5°. See Figure 3.12 (b) for schematic sensor layout.



(a)



(b)

Figure 3.12: (a) Tank with the attached sensor. (b) Schematic sensor layout and ray patterns.

The sensor must be mounted so that the line between LED output and receiver is parallel to the ground (see Figure 3.12 (a)) to minimize the effect of sensor sensitivities to the boundaries, *i.e.* sharp differences in texture or color of the object. For simplicity, we have idealized our environment by covering the occluding objects with white paper for uniformity in color, texture, and reflected ambient light.

Here we describe the process of sensor data calibration. The sensor takes readings at a rate 25 Hz. Sensor readings are produced by Analog Digital Converter (ADC), which outputs values proportional to voltage output ($V \times 204.8$). In Figure 3.13 we plot values at given distances from the object measured along

the normal to the surface of an object.

In Figure 3.14 we show several range curves constructed from different angles to the surface of the object. As one can see from Figure 3.14, the range calibration curves are shifted with respect to one another for different viewing angles (upward, when the object is viewed from the right, downward, when object is viewed from the left). This results in the same sensor output value for two different sensor positions. For example, sensor output at a distance of 90 cm from the object at an angle $-85°$ to the normal to the surface is the same as the sensor output at a distance of 45 cm at an angle $+75°$ and yet the same as the output at a distance of 60 cm along the normal to the surface.

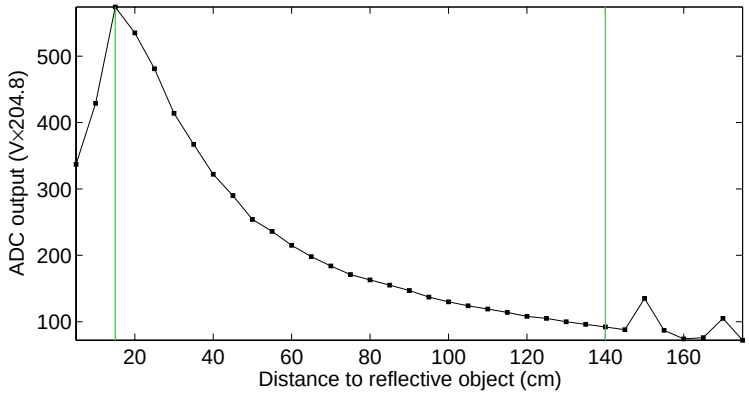


Figure 3.13: Sensor ADC output corresponding to distance to reflective object measured along the normal to the surface; green vertical lines mark working sensor range

If we take as a reference the range curve measured along the normal to the surface of reflective object, we obtain inaccuracies when looking at an object from a different angle. For example, one can see from Figure 3.15 the tilt in the measured surface position with respect to the actual one.

The results may be improved by taking several measurements along a given direction. This way we can find a matching range curve from which we can deduce

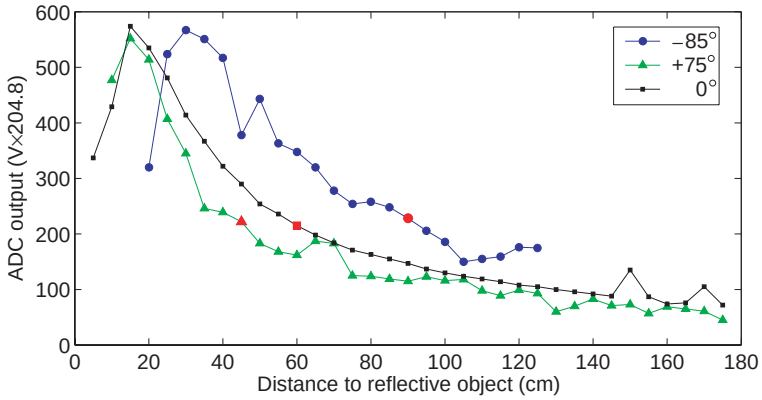


Figure 3.14: Sensor ADC output corresponding to distance to reflective object measured along different angles to the normal to the surface; red marks correspond to points on the range curves with similar sensor output.

the distance to the object and the incident angle. However, this solution is too expensive and thus we did not implement it.

In addition, we note that the shift is only significant when looking at an object from the right. Thus, a path-planning algorithm is modified with a bias towards moving in a counter-clockwise manner. See Figure 3.15 for an example.

3.2.2 Results

We use two boxes as our sample obstacles for easy representation. The positions, shapes, and quantities of obstacles are unknowns. Two tank-based vehicles equipped with the range sensors are initially positioned on the test-bed floor outside the obstacles. Each tank makes a 360° sweep to gather range data from its surrounding environment. About 80 samples are taken in one sweep. Each sweep takes less then a minute to complete. Then, a visibility map and next position of each vehicle is computed off-board based on sensor output. The next observers

position is transmitted to the robots and they proceed to collect data from new vantage point. This process is repeated until the whole region has been explored as in Algorithm 3.2 above. In the example, exploration took two steps by each observer.

The obtained range data is fit to the range calibration curve in Figure 3.13 via cubic interpolation. Then the data is processed in the following way. Whenever we get a hit which is outside of the range of the sensor or its x, y position is outside the test-bed floor, we assign the value of "infinity", which is set to be at 120 cm.

Joint visibility maps after each step are depicted in Figure 3.15. Actual obstacle boundaries are represented by yellow lines on each figure. Red stars represent positions of the robots after each step. The red lines mark the path of each vehicle up until its current location. Dark regions are invisible at current step and lighter regions are visible. Magenta circles represent shadow boundary obtained via ENO high order interpolation of the obtained range data. Black circles represent horizon points which will be approached in the next step.

The complete visibility map is depicted in Figure 3.16. It is constructed by taking the union of visibility maps of all observers at all steps. From this map, one can estimate the quantity, size, and locations of the obstacles. However, the boundaries are not accurately represented due to low sensor accuracy and small number of samples.

As was mentioned above, the results may be improved by correcting for the angle of incidence of the IR beam. Overall, the quality of the results is satisfactory taking into account hardware limitations.

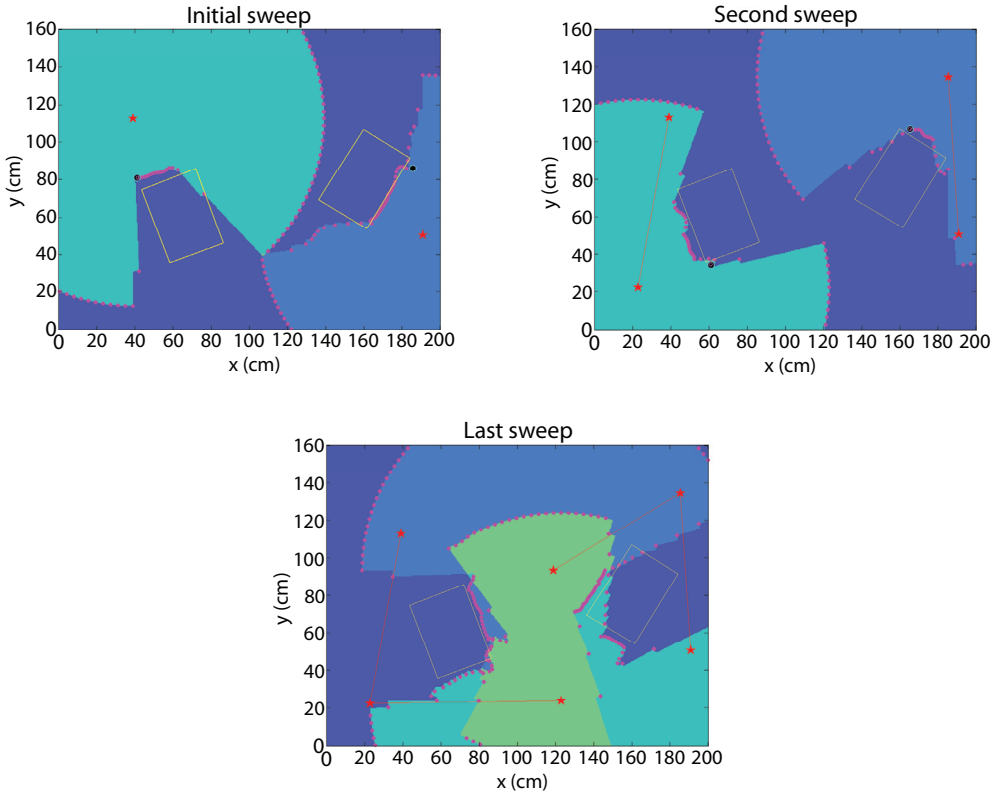


Figure 3.15: Exploration of environment with two observers. Stars are the observers' positions; small circles are the sensor output converted to range data; big dark circles are the next edges to be approached; boxes are the actual obstacles' outlines; dark regions are currently invisible; light regions are currently visible.

3.3 Postprocessing of the Path: Exposure Optimization

Once we have constructed a route to explore an unknown environment via Algorithm 3.1, we can apply optimization techniques to post-process the obtained path in order to obtain a more uniform illumination/exposure of the explored region. In what follows, let Ω be the region of exploration and D be the obstacles. Then the region discovered along the path is $\Omega \backslash D$.

Let $\gamma = \{z_0, z_2, ..., z_N\}$ be the positions returned by the exploration algorithm.

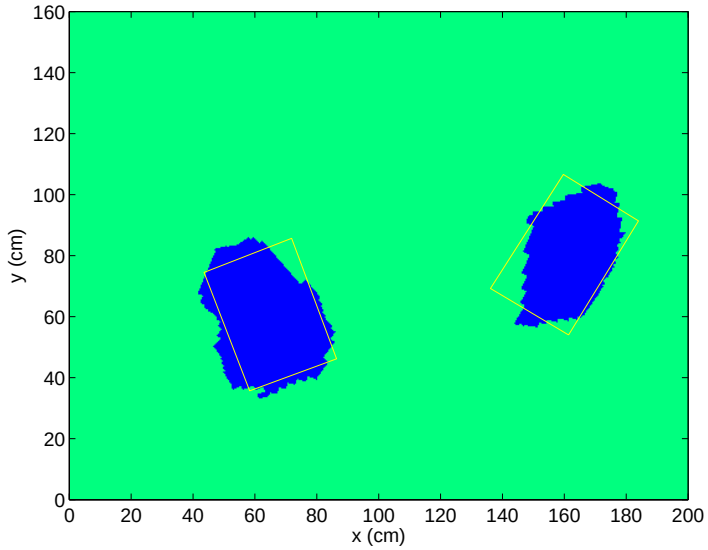


Figure 3.16: Map resulting from the environment exploration. Dark regions are invisible and light regions are visible. Boxes are the actual outlines of obstacles.

Let $\phi(\cdot; z_k)$ be the visibility level set function corresponding to observer at z_k:

$$\begin{cases} \phi(x; z_k) > 0, & \text{if } x \text{ is visible from } z_k, \\ \phi(x; z_k) < 0, & \text{if } x \text{ is invisible from } z_k, \\ \phi(x; z_k) = 0, & \text{if } x \text{ is on the shadow boundary with respect to } z_k. \end{cases} \tag{3.2}$$

Define the total illumination of a point $x \in \Omega \backslash D$ due to γ by

$$I(x; \gamma) := \sum_{k=0}^{N} H(\phi(x; z_k)), \tag{3.3}$$

where H is the one-dimensional Heaviside function. Note that I records the number of observers that can view x, and so $0 \leq I(x; \gamma) \leq N$.

From a given set of observing locations $\{z_k\}$, we seek a set of nearby $\{z_k^*\}$, so that in average points outside obstacles are viewed in a more uniform manner – the deviation of illumination from a prescribed illumination level is small.

In certain applications, a higher priority may be placed on viewing a specific region in space, while lower priority is placed on other regions. In our formulation

we simulate this effect through the use of weights. Let $w : \Omega \to \mathbb{R}^+$ be a positive real-valued function defined over Ω. We relate the magnitude of w to the importance of a given point $x \in \Omega$ to be visible, with larger magnitude associated with greater importance. By including w in the measure used in spatial integration, we attach importance weights to the visibility of space.

More precisely, we formulate the variational problem as follows:

Problem 3.1. *Given a positive constant C and a weight function $w(x)$. Find $\gamma \in \mathbb{R}^{2N}$ that minimizes*

$$\begin{aligned} E(\gamma; C) &= \frac{1}{2}\sum_{k=0}^{N-1} |z_{k+1} - z_k|^2 \\ &+ \frac{\lambda}{2}\int_{\Omega\setminus D} \left(I\left(x;\gamma\right) - C\right)^2 w(x)dx \\ &+ \mu\sum_{k=0}^{N-1} \left(|z_{k+1} - z_k| - \rho_{z_k}\left(\theta\right)\right). \end{aligned} \tag{3.4}$$

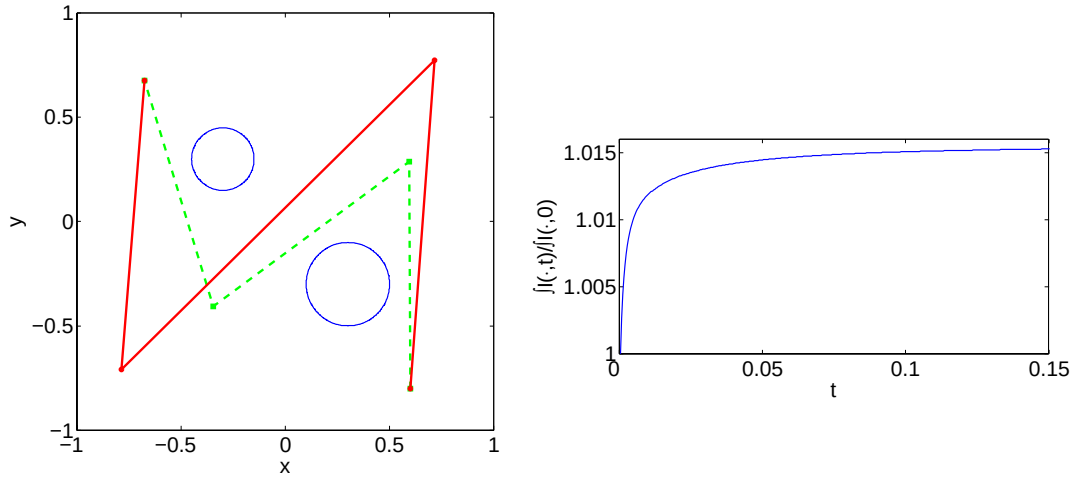
The first term in the above functional seeks to stabilize the problem, by penalizing against fractal or space filling paths. In the meantime, the last term in (3.4) prevents the continuous path connecting the discrete locations in γ from accidentally crossing obstacles' boundary. If θ is the direction of z_{k+1} when looking out of z_k, boundary non-crossing condition is equivalent to keeping $|z_{k+1} - z_k| < \rho(\theta)$ for all k. Coefficients λ and μ serve as parameters for the penalty terms.

Using summation by parts and fixing z_0 and z_N, we arrive at the following Euler-Lagrange equation:

$$\begin{aligned} \dot{z_k} &= (z_{k+1} - 2z_k + z_{k-1}) \\ &- \lambda\int_{\Omega\setminus D} \left(\sum_{j=0}^{N} H\left(\phi(x; z_j)\right) - C\right) \delta\left(\phi(x; z_k)\right) \nabla_{z_k}\phi(x; z_k) w(x)dx \\ &- \mu\left[\left(\frac{z_{k+1} - z_k}{|z_{k+1} - z_k|} - \frac{z_k - z_{k-1}}{|z_k - z_{k-1}|}\right) - \nabla_{z_k}\rho(\theta)\right],\ 1 \le k \le N-1. \end{aligned} \tag{3.5}$$

Using the path γ_0 constructed via Algorithm 3.1 as an initial guess, the equation (3.5) can be solved by simple integration techniques.

Unless stated otherwise, in the following discussion, we take $w \equiv 1$. In Figure 3.17 (a) we have two circular objects. The dashed green line segments joint the four vantage points forming initial exploration path γ_0 obtained via Algorithm 3.1. This path is then deformed using the flow (3.5), resulting in a new path γ represented in solid red. To maximize the total illumination of the region we set constant $C = 15$, so that the desired exposure is always greater then maximum possible $C = 4$. The flow eventually reaches a steady state. In Figure 3.17 (b) we plot the ratio $\int_{\Omega\setminus D} I(x;\gamma;t)dx / \int_{\Omega\setminus D} I(x;\gamma;0)dx$ of total exposure at time t to the initial total exposure. One can see that the total increase in the exposure for this simple geometry is only roughly 1%.

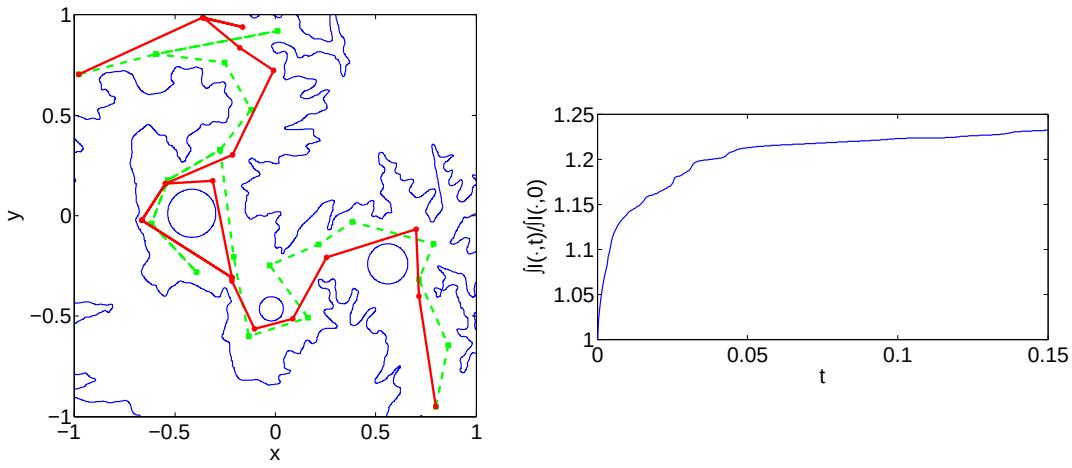


(a) (b)

Figure 3.17: Path postprocessing corresponding to Problem 3.1, obstacles – two circles: (a) initial path (dashed) and optimized path (solid); (b) $\int_{\Omega\setminus D} I(x;\gamma;t)dx / \int_{\Omega\setminus D} I(x;\gamma;0)dx$. Here $C = 15, \lambda = 0.1, \mu = 1$.

In contrast to the previous example, the gain in exposure in our second exper-

iment is around 23%, see Figure 3.18 (b). Here, the region is constructed from a slice of Grand Canyon elevation data, which has a much more complex geometrical structure comparing to the example with two circles. We further increased the complexity of the Grand Canyon terrain by adding disk-shaped holes to the interior of the explored region. The initial and optimized paths are depicted in Figure 3.18 (a). The original exploration path (dashed green) branches out to explore the regions occluded by the circles. Note that the optimized path (solid red) is shorter than the original and has fewer kinks. Here, we again choose the desired exposure constant $C = 20$.

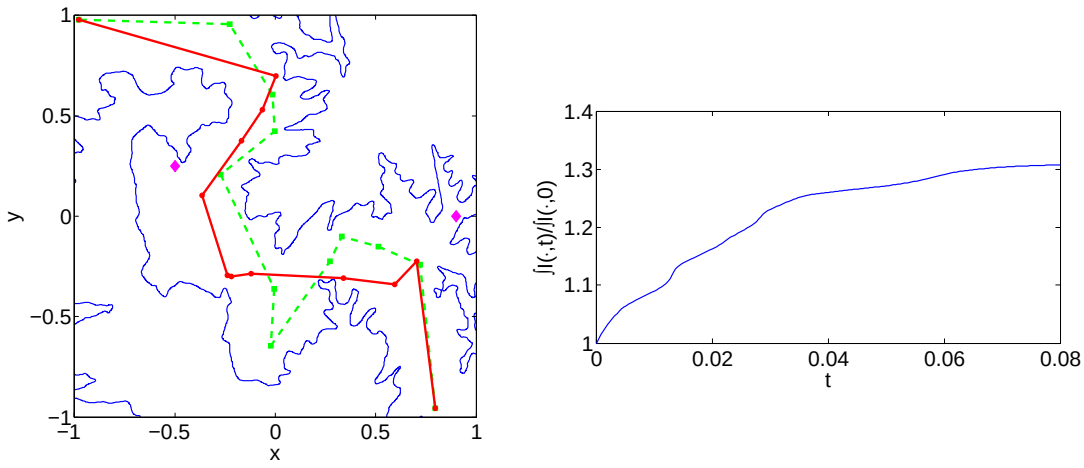


(a) (b)

Figure 3.18: Path postprocessing corresponding to Problem 3.1, obstacles – Grand Canyon terrain: (a) initial path (dashed) and optimized path (solid); (b) $\int_{\Omega\setminus D} I(x;\gamma;t)dx / \int_{\Omega\setminus D} I(x;\gamma;0)dx$. Here $C = 20, \lambda = 0.1, \mu = 1$.

Figure 3.19 shows the evolution of the path along the Grand Canyon terrain and the resulting exposure of the region where Gaussian importance weights are centered at $(0.9, 0)$ and $(-0.5, 0.25)$. We see an increase of about 30% in total illumination of the region in Figure 3.19 (b). The resulting optimized path (solid

red) in Figure 3.19 (a) is shorter than the original path (dashed green), with the observers' positions concentrated near the regions of increased importance (magenta diamonds).



(a) (b)

Figure 3.19: Path postprocessing corresponding to Problem 3.1, obstacles – Grand Canyon terrain, the weights are centered at $(0.9, 0)$ and $(-0.5, 0.25)$ (diamonds); (a) initial path (dashed) and optimized path (solid); (b) $\int_{\Omega\setminus D} I(x;\gamma;t)dx / \int_{\Omega\setminus D} I(x;\gamma;0)dx$. Here $C = 20, \lambda = 0.1, \mu = 1$.

It may be desirable in some applications to increase the number of observing locations along the path. Our goal is to arrange the new observers in an optimal way with respect to the total illumination of the region. Starting with an initial path γ, obtained by our algorithm, we can insert points along the line segments connecting z_k and z_{k+1} for each k to obtain a new set $\tilde{\gamma} = \{Z_1, Z_2, ..., Z_M\}$, such that $\gamma \subset \tilde{\gamma}$. Given a parameter $ds > 0$, we optimize the positions of thus obtained vantage points with an additional constraint $|Z_{k+1} - Z_k| = ds$. Precise variational formulation of the problem is provided below:

Problem 3.2. *Given* $\{z_k\}_{k=1}^N$, *a constant* $C > 0$, *and a parameter* $ds > 0$, *find*

$\tilde{\gamma} \in \mathbb{R}^{2M}$ *that minimizes*

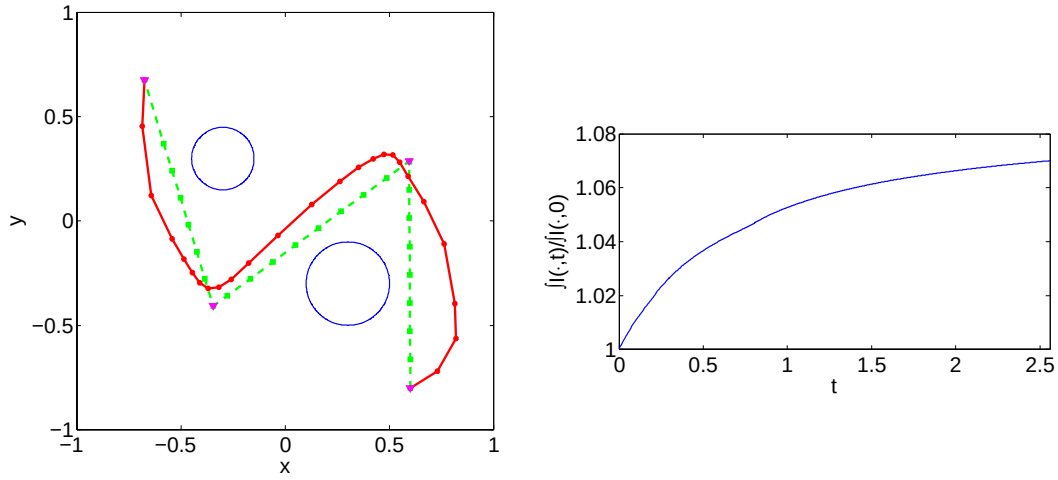
$$\begin{aligned} E(\tilde{\gamma}; C) &= \frac{1}{2} \sum_{k=1}^{M-1} \left(|Z_{k+1} - Z_k| - ds \right)^2 \\ &+ \frac{\lambda}{2} \int_{\Omega \setminus D} \left(I\left(x; \tilde{\gamma}\right) - C \right)^2 dx \\ &+ \mu \sum_{k=1}^{M-1} \left(|Z_{k+1} - Z_k| - \rho_{Z_k}\left(\theta\right) \right). \end{aligned} \tag{3.6}$$

Similar to Problem 3.1, the first term in the above functional acts as a regularizer of the path. By setting ds to be the actual arc length along the path, we additionally enforce the uniform distribution of the observing locations along the path. The Euler-Lagrange equation corresponding to Problem 3.2 is

$$\begin{aligned} \dot{Z}_k &= \left(Z_{k+1} - 2Z_k + Z_{k-1} + ds \left[\frac{Z_k - Z_{k-1}}{|Z_k - Z_{k-1}|} - \frac{Z_{k+1} - Z_k}{|Z_{k+1} - Z_k|} \right] \right) \\ &- \lambda \int_{\Omega \setminus D} \left(\sum_{j=1}^{M} H\left(\phi\left(x; Z_j\right)\right) - C \right) \delta\left(\phi\left(x; Z_k\right)\right) \nabla_{Z_k} \phi\left(x; Z_k\right) dx \\ &- \mu \left[\left(\frac{Z_{k+1} - Z_k}{|Z_{k+1} - Z_k|} - \frac{Z_k - Z_{k-1}}{|Z_k - Z_{k-1}|} \right) - \nabla_{Z_k} \rho(\theta) \right], \ 2 \leq k \leq M-1. \end{aligned} \tag{3.7}$$

In Figures 3.20 and 3.21 we present the results of exploration according to Problem 3.2. In the case of two circles, the path expands away from the obstacles' boundaries, which provides better illumination of the region. The total gain in illumination as a result of the flow (3.7) is about 7%. The path, in the case of Grand Canyon, clearly smoothes out and contracts as a result of postprocessing, as can be seen from Figure 3.21 (a). Note that in order to have a continuous path, the observer has to backtrack in places where the path branches out. The exposure of the region keeps increasing with the total gain slightly under 15%. Remark that more complex environments *e.g.* Grand Canyon, allow for greater improvement in illumination through postprocessing of the path as opposed to simpler environments like the one with two circles.

Other types of visibility optimization problems are discussed in [CT05].

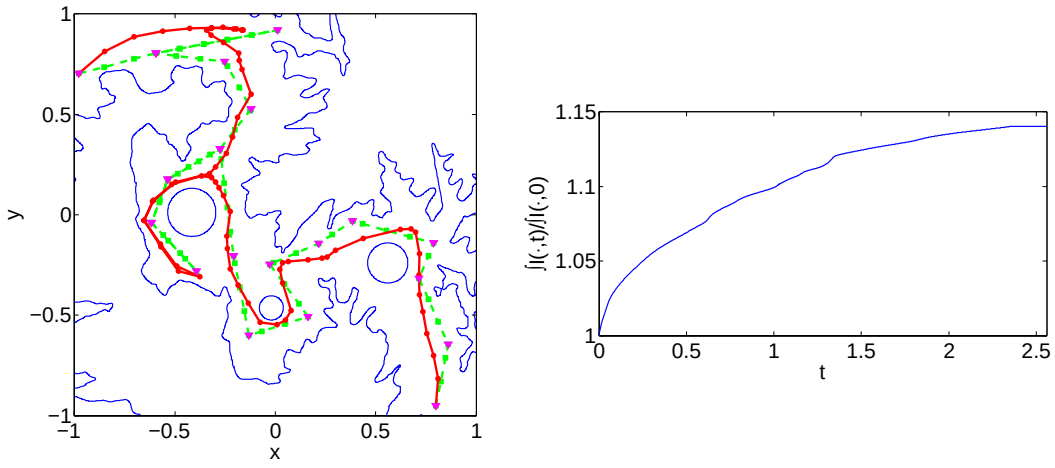


(a) (b)

Figure 3.20: Path postprocessing corresponding to Problem 3.2, obstacles – two circles: (a) initial path (dashed), four original observers' locations (triangles), and optimized path (solid); (b) $\int_{\Omega\setminus D} I(x;\gamma;t)dx / \int_{\Omega\setminus D} I(x;\gamma;0)dx$. Here $C = 100, \lambda = 0.001, \mu = 1, ds = 0.01$, total number of steps along the path is 26.

3.4 Complexity Estimates

Below, we provide the convergence analysis of the exploration Algorithm 3.1 in some canonical types of environments. We begin the Subsection 3.4.1 with the discussion of the complexity of the algorithm in simple environments with a single strictly convex or star-shaped obstacle. We proceed to demonstrate the convergence of the Algorithm 3.1 in an bounded region containing a finite number of non-overlapping convex obstacles. Subsection 3.4.2 contains the result by L.-T. Cheng, showing that four steps would be enough to fully explore the surface of a single convex object in three dimensions.



(a) (b)

Figure 3.21: Path postprocessing corresponding to Problem 3.2, obstacles – Grand Canyon terrain: (a) initial path (dashed), original observers' locations (triangles), and optimized path (solid); (b) $\int_{\Omega\setminus D} I(x;\gamma;t)dx / \int_{\Omega\setminus D} I(x;\gamma;0)dx$. Here $C = 150, \lambda = 0.001, \mu = 1, ds = 0.01$, total number of steps along the path is 71.

3.4.1 Two-dimensional Case

Let us begin by considering a single obstacle Ω bounded by a simple C^2 regular convex curve Γ.

Proposition 3.3. *The Gauss map $S : \Gamma \mapsto \mathcal{S}^1$ is monotone.*

Proof. Note that a single closed C^2 regular curve is convex if and only if its signed curvature κ does not change sign, in particular, if it is never zero. But the curvature κ equals the magnitude of the derivative of the tangent vector parameterized by the arc length of a given curve. Since the normal vector to a curve is just a tangent vector rotated by $\pi/2$, we require the angle of the normal vector to be monotone along the curve, otherwise κ would change its sign. Thus,

the Gauss map S is monotone. □

Next, we are going to use the monotonicity of the Gauss Map to construct a path for the observer to explore the boundary Γ of the obstacle Ω.

Claim 3.4. *At least three steps are required to explore a simply connected region containing a single convex obstacle.*

Proof. Let O be the center of mass of Ω. Since the Gauss map $S : \Gamma \mapsto \mathcal{S}^1$ is monotone, "seeing" Γ is equivalent to seeing the boundary of a disk C centered at O that encloses Ω. Let z_0, z_1, and z_2 be the vertices of a triangle that encloses C and is tangent to C at e_0, e_1, and e_2 (Figure 3.22). The observer placed at z_0 is able to see the portion of the boundary of C between e_0 and e_1. Similarly, from z_1 the observer is able to see an arc between e_1 and e_2, and from z_2, the remaining portion of the boundary of C, and, correspondingly, Γ. Since the observer's path must be finite, we exclude the case of exploring the entire boundary in just two steps. □

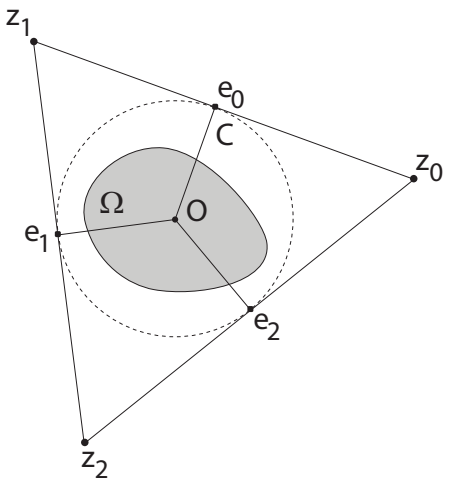


Figure 3.22: Constructing a three-step path around a single convex obstacle.

If we further assume that the observer may not approach the obstacle nearer than $\lambda > 0$, and may not depart from the obstacle further than η, the number of steps required to explore Γ is

$$\left\lfloor \frac{\pi}{\cos^{-1} \frac{r+\lambda}{r+\eta}} \right\rfloor .$$

Here, r is the radius of the smallest disk enclosing Ω, centered at the center of mass of Ω as in Figure 3.23.

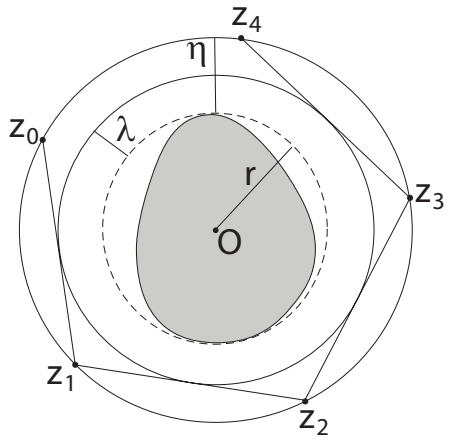


Figure 3.23: Constructing a path around a convex obstacle under restrictions.

Using the above results, we can also estimate the number of steps required to explore the boundary of a single star-shaped obstacle.

Claim 3.5. *Let Ω be a star-shaped obstacle bounded by a simple curve Γ. Assume the curvature κ of Γ changes its sign $2m$ times. Then $3 + m$ steps would be sufficient to fully explore the environment containing a single obstacle Ω.*

Proof. Let the obstacle Ω be bounded by $\Gamma = \Gamma_0 \cup \Gamma_1 \cup \Gamma_2 \cup \Gamma_3$, as depicted in Figure 3.24. To begin with, assume the signed curvature κ of Γ changes its sign twice at points Q_1 and Q_2.

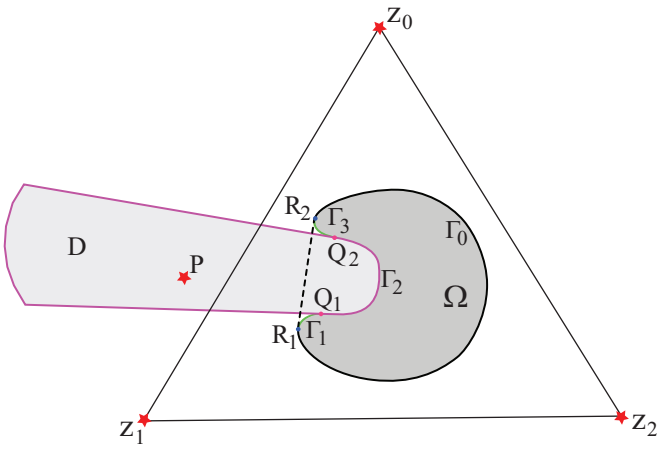


Figure 3.24: Constructing a path around a star-shaped obstacle.

Let H be the convex hull of Γ with $\partial H = \Gamma_0 \cup R_1R_2$. Then, according to Claim 3.4, three points z_0, z_1, and z_2 are enough to see the entire boundary ∂H. Next, construct a convex region D by extending the rays from Q_1 and Q_2 tangent to Γ_2, until hit the boundary of the region (see Figure 3.24). With this construction $\bar{D} \cap \bar{\Omega} = \Gamma_2$. Furthermore, the concave portion of Γ is visible from any point P in D. Thus at most four points z_0, z_1, z_2 and P are sufficient to explore the entire Γ. Note that if one of the points $\{z_i\}, i = 0, 1, 2$ is in D, then only three points would be sufficient.

If the curvature changes its sign $2m$ times along Γ, three steps are still required to explore the convex hull of Γ, and at most m additional steps are required to explore each concave portion of Γ. Thus the total number of steps required to explore the boundary of Ω is $3 + m$. □

Next, consider the scenario where the environment contains a finite number of disjoint, closed, strictly convex sets as in Figure 3.25. Denote the convex components by $\{\Omega_j\}_{j=1}^{M}$. Let C_j be the smallest disk enclosing Ω_j. Note that according to Claim 3.4, seeing the boundary of Ω_j is equivalent to seeing the

boundary of C_j.

Furthermore, let C'_j be the smallest disk encircling an equilateral triangle that contains C_j and is tangent to C_j at three points. If r_j is the radius of C_j and r'_j is the radius of C'_j, then the length of a side of the triangle is $L_j = 2\sqrt{3}r_j = \sqrt{r'_j}$. Assume that $C'_j \cap C'_k = \emptyset$ for all j and k.

Finally, let $R = \max_{j,k} \text{dist}\{C'_j, C'_k\}$ be the largest distance between any two disks in $\{C_j\}_{j=1}^{M}$. Since the number of obstacles is finite, all the disks must be contained in some bounded domain B_R.

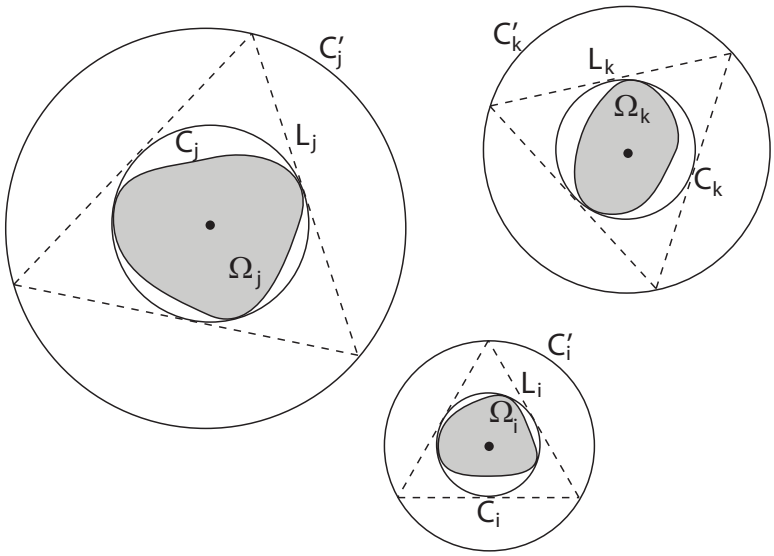


Figure 3.25: Sample environment with closed, convex, disjoint obstacles.

The following result is essential in proving convergence of the exploration Algorithm 3.1 in a sample environment described above.

Proposition 3.6. *Start at z_0 on some $C'_1 \supset C_1$ and overshoot the horizon e_1 by $L/2$ to arrive at z_1. Then will explore the entire C_1, i.e. the remaining arc between the horizons e_0 and e_2 on Figure 3.26 in finite number of steps.*

Proof. From its current position z_1 the observer may

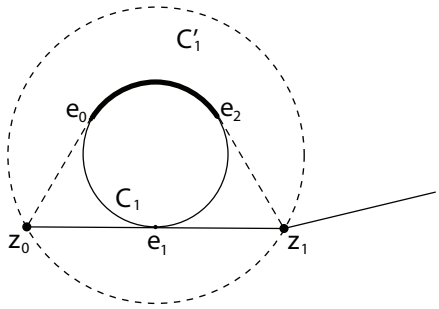


Figure 3.26: Setup for Proposition 3.6. A bold arc is the unexplored portion of C_1.

1. Proceed to see e_2, so that $z_2 \in C_1'$ and all of C_1 has been explored in just three steps: z_0, z_1, and z_2.

2. Proceed to some $e_3 \in C_2$, such that $\|z_1 - e_3\| \leq \|z_1 - e_2\|$. In this case z_2 is on some $C_2' \neq C_1'$.

In case 2 the following is true:

Claim 3.7. *If there exists $k \geq 1$, such that $z_{k+1} \in C_1'$ then C_1 is entirely seen from z_0, z_1, $z_{k+1/2}$, and z_{k+1}.*

Proof. If $z_{k+1} \in C_1'$, then there is a horizon $e_k \in (e_0, e_2)$ on C_1, which is the nearest to $z_k \in C_j'$ for some j. Let $z_{k+1/2} \in C_1'$ be the point of intersection of the ray (z_k, z_{k+1}) and C_1'. Thus, the arc $e_k e_2$ is completely visible from z_{k+1}. Hence, the entire arc of C_1 between e_0 and e_2 has been explored. The proof of the Claim 3.7 is now complete. □

Suppose towards a contradiction that the observer does not return to C_1 at all. Since the collection $\{C_k\}$ is finite, the observer must be then trapped in a loop, *i.e.* there exist C_j and C_k such that C_j gets approached from C_k infinitely many times. But this is impossible according to the following claim.

Claim 3.8. *A disk C_j may be approached from another disk C_k at most twice.*

Proof. Note that given any two disks, there exist four bitangents: ℓ_1, ℓ_2 and symmetric ℓ_1' and ℓ_2' as in Figure 3.27. We will only consider ℓ_1 and ℓ_2 below

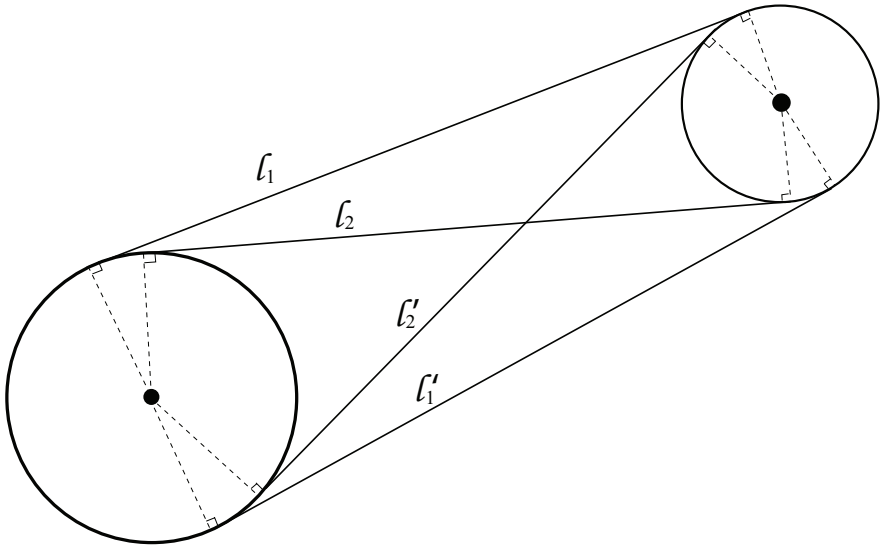


Figure 3.27: Four bitangents to two disks.

unless we indicate otherwise. The two disks C_j and C_k are *neighbors*, if there is no other obstacle in the region bounded by arcs of C_j' and C_k' between the outermost bitangents to C_j' and C_k' (ℓ_1 and ℓ_1' in Figure 3.27).

If there exists $C_k' \neq C_j'$ such that C_j and C_k are neighbors, then there is a ray from the center of C_k' tangent to C_j and orthogonal to some θ, as in Figure 3.28. Then C_k' tangents the rays orthogonal to $\theta \pm \delta\theta$. By construction, it is obvious that all possible return angles on C_j from $z_m \in C_k'$ are within $\theta \pm \delta\theta$, *i.e.* the observer may only approach horizons on C_j that lie on an arc of size $2\delta\theta$. Refer to Figure 3.28 for an illustration.

Now, let $z_{m+1/2}$ be the point of intersection of the ray $z_m z_{m+1}$ and the disk C_j'. Then let θ_1 be the angle visible from $z_{m+1/2} \in C_j'$ and θ_2 be the angle visible

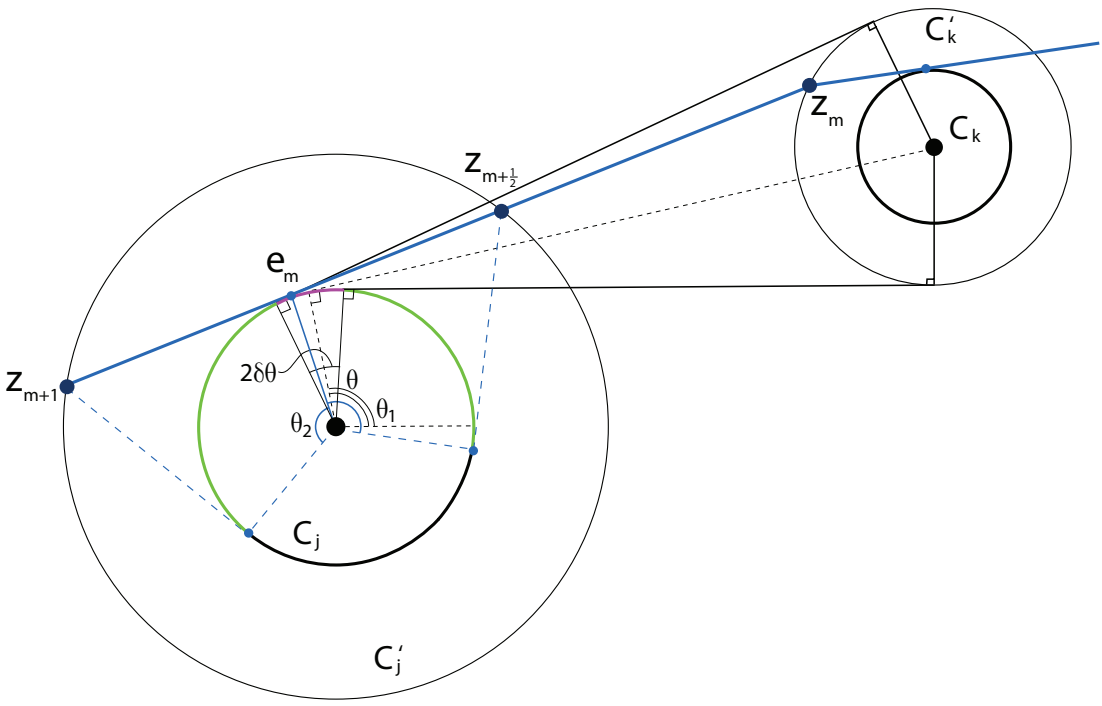


Figure 3.28: Portions of C_j visible from $z_{m+\frac{1}{2}} \in C'_j$ and $z_{m+1} \in C'_j$.

from $z_{m+1} \in C'_j$ depicted in Figure 3.28. Note that $\theta_1 = \theta_2 = 2\pi/3$. Hence, the arc corresponding to the angle of $2\delta\theta$ is entirely visible from $z_{m+1/2}$ and z_{m+1}.

Thus an observer is able to see the entire arc of C_j, containing all possible horizons that could be detected from C'_k, in a single approach. The symmetrical case with another pair of bitangents provides the possibility for the second approach from C'_k. Hence, an observer may only approach C_j from C'_k at most twice. This completes the proof of Claim 3.8. □

We have shown that it is not possible for an observer to approach a single disk infinitely many times. Therefore, the observer must return to the initial disk C_1 in finite number of steps. Finally, Claim 3.7 implies the result of Proposition 3.6. □

In the final step of the convergence proof, we need to show that the observer

has indeed explored the entire environment at the termination of the path.

Proposition 3.9. *The entire environment B_R has been explored at the termination of the exploration algorithm. In other words, the observer has seen the boundary of every obstacle at the termination.*

Proof. The proof follows from the two claims below.

Claim 3.10. *During the exploration of C_j, the observer must detect at least one horizon/edge on every neighbor of C_j.*

Proof. By the definition of neighbors, there are no other objects obstructing the neighbors from each other. Assume, without loss of generality, that the observer visits vantage points z_0 and z_1 on C_j' during the exploration of C_j. Consider the rays ℓ_1, ℓ_2, and ℓ_3, which are tangent to C_j at the horizon points e_0, e_1, and e_2, as in Figure 3.29. Then every neighbor of C_j must lie in one of the half-

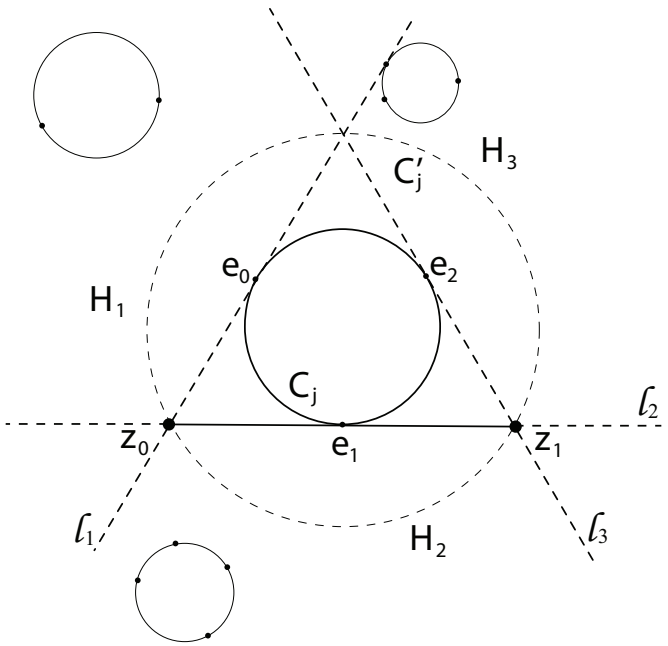


Figure 3.29: Detecting horizons on the neighbors of C_j.

planes H_1, H_2, H_3, or their intersections. The observer at z_0 is able to see all of $H_1 \cup H_2$, whereas $H_2 \cup H_3$ is visible from z_1. Therefore, the observer is able to see at least one horizon/edge on each neighbor from just z_0 and z_1. This proves Claim 3.10. □

Once a horizon has been detected on a disk C_j, the entire C_j will be explored entirely later on, according to Proposition 3.6. Now we shell prove that every disk in the domain would have a horizon detected on it during the exploration.

Claim 3.11. *Every disk in the given configuration will have at least one horizon labeled on it at some stage of the exploration.*

Proof. The following proof is by induction. Suppose we start at some disk C_1. Then all the neighbors $\{C_{1_j}\}_{j=1}^{M}$ of C_1 will have at least one edge marked on them according to Claim 3.10. Suppose at some stage of the exploration, all the disks have at least one edge labeled on them but C_k. In the given configuration, C_k has at least one neighbor C_{k_1}. By the induction assumption, C_{k_1} also has some horizons labeled on it. Then, at some stage of the algorithm, the observer must come to explore C_{k_1}. At that time it will label an edge on C_k, since C_k and C_{k_1} are neighbors. Thus we have proven Claim 3.11. □

It follows from the Claims 3.10 and 3.11 that the entire environment will be explored at the termination of the algorithm. This completes the proof of Proposition 3.9. □

Propositions 3.6 and 3.9 imply that the Algorithm 3.1 will terminate in finitely many steps. At the termination of the path, the entire environment, consisting of a finite number non-overlapping, closed, strictly convex sets, will be mapped.

Although the exact number of steps may not be estimated for a given configuration, a crude upper bound may be obtained by considering each obstacle in the environment individually and applying the results of Claim 3.4, for a single convex obstacle and Claim 3.5 for a single star-shaped obstacle. As has been statistically demonstrated in Subsection 3.1.2, the actual number of steps required to explore an environment with closed disjoint obstacles is much smaller than thus obtained upper bound.

3.4.2 Three-dimensional Case

Consider a closed, compact, convex, smooth hypersurface M with interior Ω. Below we are going to state the proof by L.-T. Cheng that four steps are enough to explore the region in three dimensions that contains an obstacle bounded by M.

Notation 3.12. *We say that $x \in \mathbb{R}^3$ is* visible *to $x_0 \in \mathbb{R}^3$ if $tx + (1-t)x_0 \notin \Omega$ for any $t \in [0,1]$.*

Let $x_1, x_2, x_3, x_4 \in \mathbb{R}^3$ such that M is contained in the convex hull H of $\{x_1, x_2, x_3, x_4\}$.

Proposition 3.13. *Given $x \in M$, x is visible to x_i for some $i \in \{1,2,3,4\}$.*

The proof of Proposition 3.13 is postponed until the end of current subsection.

Claim 3.14. *Let $\Omega_0 \subseteq \mathbb{R}^3$ be a convex set and P_0 a hyperplane. Then there exists $A_0, B_0 \subset \Omega_0$, disjoint and convex, such that $\Omega_0 \backslash P_0 = A_0 \cup B_0$.*

Proof of Claim 3.14. There exists $n \in \mathbb{R}^3$, such that $P_0 = \{x \in \mathbb{R}^3 | n \cdot x = 0\}$. Let $A_0 = \{x \in \Omega_0 | n \cdot x > 0\}$ and $B_0 = \{x \in \Omega_0 | n \cdot x < 0\}$. Trivially, $A_0 \cap B_0 = \emptyset$ and $\Omega_0 \backslash P_0 = A_0 \cup B_0$.

If $A_0 = \emptyset$, then A_0 is convex. Otherwise, let $x, y \in A_0$ and $f : [0,1] \mapsto \mathbb{R}$ with $f(t) = n \cdot (ty + (1-t)x)$. Then

$$\begin{aligned} f(t) &= n \cdot (ty + (1-t)x) \\ &= (n \cdot y - n \cdot x)\, t + n \cdot x, \end{aligned}$$

so f is monotone. Thus $f(0) > 0$ and $f(1) > 0$ implies $f(t) > 0$, or $n \cdot (ty + (1-t)x) > 0$ for $t \in [0,1]$. Since $x, y \in A_0 \subseteq \Omega_0$ and Ω_0 is convex, we also know $ty + (1-t)x \in \Omega_0$ for all $t \in [0,1]$. Thus $ty + (1-t)x \in A_0$ for all $t \in [0,1]$, which implies A_0 is convex.

Similarly, we show B_0 is convex. Therefore, $\Omega_0 \backslash P_0 = A_0 \cup B_0$ where A_0, B_0 are disjoint and convex. This concludes the proof of Claim 3.14. □

Henceforth, we write $P = \{x \in \mathbb{R}^3 | n \cdot x = 0\}$ and let $A = \{x \in \mathbb{R}^3 | n \cdot x > 0\}$ and $B = \{x \in \mathbb{R}^3 | n \cdot x < 0\}$. So we know $A, B \subseteq \Omega$ disjoint and convex, such that $\Omega \backslash P = A \cup B$. Furthermore, let $C = H \cap A$ and $D = H \cap B$. So we know that $C, D \subseteq H$ are disjoint and convex, such that $H \backslash P = C \cup D$.

Claim 3.15. *$\Omega \subseteq A$ or $\Omega \subseteq B$.*

Proof of Claim 3.15. This follows from the convexity of Ω. □

Claim 3.16. *The sets $A \cup P$, $B \cup P$, $C \cup (P \cap H)$, and $D \cup (P \cap H)$ are all convex.*

Proof of Claim 3.16. Given $x, y \in \mathbb{R}^3$, consider $f : [0,1] \mapsto \mathbb{R}$ with $f(t) = n \cdot (ty + (1-t)x)$. Note that f is monotone.

If $x, y \in A \cup P$, then $f(0), f(1) \geq 0$, so $f(t) \geq 0$ for all $t \in [0,1]$. Thus $ty + (1-t)x \in A \cup P$ for all $t \in [0,1]$.

If $x, y \in B \cup P$, then $f(0), f(1) \leq 0$, so $f(t) \leq 0$ for all $t \in [0,1]$. Thus $ty + (1-t)x \in B \cup P$ for all $t \in [0,1]$.

If $x, y \in C \cup (P \cap H)$, then $f(0), f(1) \geq 0$, so $f(t) \geq 0$ for all $t \in [0,1]$. By convexity of H, $ty+(1-t)x \in H$ for all $t \in [0,1]$. Thus $ty+(1-t)x \in C \cup (P \cap H)$ for all $t \in [0,1]$.

Finally, if $x, y \in D \cup (P \cap H)$, then $f(0), f(1) \leq 0$, so $f(t) \leq 0$ for all $t \in [0,1]$. By convexity of H, $ty + (1-t)x \in H$ for all $t \in [0,1]$. Thus $ty + (1-t)x \in D \cup (P \cap H)$ for all $t \in [0,1]$.

Therefore, $A \cup P$, $B \cup P$, $C \cup (P \cap H)$, and $D \cup (P \cap H)$ are all convex, and the proof of Claim 3.16 is complete. □

Claim 3.17. *If $\Omega \subseteq A$, implying $x_0 \notin A$ and if $\Omega \subseteq B$, implying $x_0 \notin B$, then x is visible to x_0.*

Proof of Claim 3.17. If $\Omega \subseteq A$, then by the hypothesis, $x_0 \notin A$. This implies $x_0 \in B \cup P$. The convexity of $B \cup P$ implies $tx + (1-t)x_0 \in B \cup P$ for all $t \in [0,1]$, so $tx + (1-t)x_0 \notin \Omega$ for all $t \in [0,1]$. Therefore, x is visible to x_0.

Similarly, if $\Omega \subseteq B$, then by the hypothesis, $x_0 \notin B$ and x is visible to x_0. Thus, when the hypothesis is satisfied, x is visible to x_0. The proof of Claim 3.17 is complete. □

Claim 3.18. *If there exist $j, k \in \{1,2,3,4\}$, such that $x_j \notin A$ and $x_k \notin B$ then x is visible to x_j or x_k.*

Proof of Claim 3.18. Suppose there exist $j, k \in \{1,2,3,4\}$, such that $x_j \notin A$ and $x_k \notin B$. Then $\Omega \subseteq A$ implies x is visible to x_k. Similarly, $\Omega \subseteq B$ implies x is visible to x_j. Thus, in all cases, x is visible to either x_j or x_k. □

Claim 3.19. *There exist $j, k \in \{1,2,3,4\}$, such that $x_j \notin C$ and $x_k \notin D$.*

Proof of Claim 3.19. Suppose $x_j \in C$ for all $j \in \{1, 2, 3, 4\}$. Since $C \subseteq H$, then $C = H$, since H is the convex hull of $\{x_1, x_2, x_3, x_4\}$. But this contradicts the fact that $P \cap H \neq \emptyset$. Similar argument works for set D. Thus, there exist $j, k \in \{1, 2, 3, 4\}$, such that $x_j \notin C$ and $x_k \notin D$. □

Finally, we use the above claims to prove the Proposition 3.13.

Proof of Proposition 3.13. There exist $j, k \in \{1, 2, 3, 4\}$, such that $x_j \notin C$ and $x_k \notin D$. This implies $x_j \notin A$ and $x_k \notin B$. Thus, given $x \in M$, x is visible to either x_j or x_k. □

We have demonstrated that four points are sufficient to explore the entire surface of a convex obstacle Ω in a three-dimensional space.

CHAPTER 4

Summary and Future Work

In Chapter 1 we introduce the problem of visibility and its applications. We present an extensive survey of existing techniques for visibility representation based on various fields of application ranging from computer graphics and vision to robotics and computational geometry. In particular, the combinatorial and computational geometry approach has been emphasized for its great value and influence on most modern visibility-based research. Another technique that has recently gained popularity is the implicit level set representation of [TCO04], which, among other benefits, offers a framework for solution of problems related to optimal navigation planning [CT05].

Chapter 2 of the thesis deals with visibility of point clouds and visible surface reconstruction – a problem that has gained increasing popularity during the past decade. We propose a novel algorithm for reconstructing visibility from point clouds, which is based on the causality condition of visibility: *if a point is occluded, then all other points farther away from the vantage point along the same line of sight are also occluded*. The main steps of the proposed algorithm are projection of the point cloud onto a unit sphere centered at the vantage point, filtering out the visible data points, and interpolation of the visible data. We utilize essentially non-oscillatory (ENO) interpolation [HEO87] to obtain a piecewise high-order reconstruction of the occluding surfaces. Such a representation allows computation of derivatives and other useful geometric quantities on the occlud-

ing surfaces. Image-processing techniques may be incorporated to process noisy point clouds. Moreover, the formulation can be easily extended to the case of bending ray-fields. Combining reconstructions from several vantage points into a single piecewise-smooth surface is made possible via a straightforward conversion to the implicit level set visibility representation. Additionally, we provide the error analysis of the resulting interpolant. Finally, the differential equations for the dynamics of the visibility region is derived.

While the proposed algorithm is straight forward in two spacial dimensions, its extension to three-dimensional environments is complicated. The problem arises on the last step of the algorithm. The projected data points need to be interpolated to obtain a visible surface reconstruction. Since the data does not lie on any grid, a standard dimension-by-dimension rectangular grid interpolation technique may not be applied right away. One possible strategy would be to work on triangular grid. Instead, we propose an ENO-based preprocessing step. It allows us to map the filtered data onto a coarse rectangular grid on $\mathcal{S}^2$. Then, a dimension-by-dimension high-order ENO interpolation can be applied to obtain a piecewise smooth representation of the occluding surface in three dimensions. Again, conversion to a level set formulation allows a simple representation of the union of surfaces visible from multiple viewpoints.

Chapter 3 addresses the application of the visibility of point clouds to mapping of unknown environments. We propose a navigation algorithm that allows autonomous observer(s), equipped with a range-finding device, to explore an unknown bounded region with obstacles. To ensure that the algorithm could be utilized in real-life applications we require the observer's path to be continuous and consist of a finite number of discrete steps. The work of S. LaValle, B. Tovar, *et al.* [TML07] served as an inspiration for our algorithm. Similarly to

[TML07, TGL05], the key idea behind our approach is to navigate towards one of the unexplored horizons, *i.e.* terminating points on the occluding surfaces that signal a connected region of the environment that is occluded from the observer at its current position. Unlike the strategy of [TML07], our algorithm utilizes the visibility map of previously uncovered environment in the decision-making process. The performance of our algorithm has been evaluated in [LGH07] on an economical cooperative control tank-based platform, using multiple mobile inexpensive sensors where noise is an issue. Once a path through the environment has been constructed, we apply post-processing optimization techniques to obtain a more uniform exposure of the explored region along the path. More examples of visibility optimization are provided in [CT05].

We use statistical analysis to estimate the number of steps required to explore a bounded region with a finite number of disjoint obstacles. A convergence proof of our algorithm in sample bounded regions with a finite number of disjoint convex obstacles is provided. In particular, we show that the algorithm would always terminate in finite time, and all the obstacles' boundaries would be seen at its termination. Additionally, we prove simple estimates on the number of steps required to explore a region with a single convex or star-shaped obstacle. These bounds may be used to obtain an upper bound on the number of steps required to explore more complex environments with multiple non-overlapping obstacles.

The exploration algorithm could be extended to three dimensions as well. However, no satisfactory computational results have been obtained at the moment. Current thesis contains a proof by L.-T. Cheng, demonstrating that a single convex obstacle in a three-dimensional space can be explored in at least four steps. However, an thorough theoretical analysis of the algorithm in three dimensions presents a challenging problem for further study.

Another difficult task, not addressed by the dissertation, is that of finding a hyperbolic or diffusive point source in an unknown environment. Consider a bounded domain $D \subset \mathbb{R}^2$ and an obstacle $\Omega \subsetneq D$. Let s_0 and x_0 denote respectively the coordinates of the point-source signal and the observer. Assume the signal is emitted in all directions, and the observer is capable to survey all directions from its present location. Let ϕ_{x_0} denote the visibility function from the observing location x_0, such that $\{\phi_{x_0} > 0\}$ is the set of points visible from x_0.

For the wave source, let $w(x) = w(x; s_0, \Omega)$ denote the wave field of the signal that is emitted from s_0. Assume it satisfies the Helmholtz equation

$$-c^2(x)\Delta w = k^2 w, w(s_0) = w_0 \text{ and } x \in D\backslash\Omega, \tag{4.1}$$

with the wave speed $c(x) = s_0\chi_{D\backslash\Omega}$ (meaning the signal cannot penetrate Ω) and $0 < k \ll 1$.

For the diffusive source:

$$-\Delta w = \delta_{x_0}, w|_{\partial\Omega} = 0 \text{ and } x \in D\backslash\Omega. \tag{4.2}$$

We further assume that the observer is capable of sensing both the strength of the signal w and the directional derivatives $\nabla w \cdot \theta$, for all $\theta \in \mathcal{S}^1$. We would like to solve the general type of problems, which may resemble the standard imaging problems:

Problem 4.1. *Find the shortest path $x_0(t) \in D\backslash\Omega$ from $x_0(0)$ to see s_0 for the following cases with Helmholtz source for $k \ll 1$ or the diffusive source:*

1. *Ω is known and the location of s_0 is known.* (This problem is solved in [CT05], where Ω is given by a level set function.)

2. *Ω is sampled by a point cloud from the observing location and s_0 is known.*

3. *Ω is known, s_0 is unknown, but $w(x_0)$ and $w(x_0) \cdot \theta$ are available.*

4. *Ω is sampled by a point cloud from the observing location, s_0 is unknown, but $w(x_0)$ and $w(x_0) \cdot \theta$ are available.*

Remark that in the first two subproblems we do not need to use the information we get from w, since the location of the source s_0 is known.

References

[ABC03] M. Alexa, J. Behr, D. Cohen-Or, S. Fleishman, D. Levin, and C. T. Silva. "Computing and rendering point set surfaces." *IEEE Trans. Vis.& Graph.*, **9**(1):3–15, 2003.

[AK02] G. Aubert and P. Kornprobst. *Mathematical problems in image processing. Partial differential equations and the calculus of variations.* Springer, 2002.

[AS96] P. K. Agarwal and M. Sharir. "Ray shooting amidst convex polyhedra and polyhedral terrains in three dimensions." *SIAM J. Comp.*, **25**(1):100–116, 1996.

[AS07] J. Andrews and J. A. Sethian. "Fast marching methods for the continuous traveling salesman problem." In *Proc. of the National Academy of Sciences of the United States of America (PNAS)*, volume 104, pp. 1118–1123, 2007.

[BD90] K. W. Bowyer and C. R. Dyer. "Aspect graphs: An introduction and survey of recent results." *Int. J. of Imaging Systems and Technology*, **2**(4):315–328, 1990.

[BEF96] J. Borenstein, B. Everett, and L. Feng. *Navigating mobile robots: Systems and techniques.* A. K. Peters Ltd., Wellesley, MA, 1996.

[Bet01] M. D. Betterton. "Theory of structure formation in snowfields motivated by penitentes, suncups, and dirt cones." *Phys. Rev. E*, **63**(5):056129, 2001.

[BR94] O. P. Bruno and F. Reitich. "A new approach to the solution of problems of scattering by bounded obstacles." In *SPIE*, pp. 2192–2028, 1994.

[Can88] J. F. Canny. *The complexity of robot motion planning.* MIT Press, 1988.

[CBM03] J. C. Carr, R. K. Beatson, B. C. McCallum, W. R. Fright, T. J. McLennan, and T. J. Mitchell. "Smooth surface reconstruction from noisy range data." In *GRAPHITE '03: Proc. of the 1st intl. conf. on comp. graph. and interactive techniques in Australasia and South East Asia*, pp. 119–ff, 2003.

[Chv75] V. Chvátal. "A combinatorial theorem in plane geometry." *J. Comb. Theory Ser. B*, **18**:39–41, 1975.

[CM04] T. Cecil and D. Marthaler. "A variational approach to search and path planning using level set methods." UCLA CAM report, 04-61, 2004.

[CM06] T. Cecil and D. Marthaler. "A variational approach to path planning in three dimensions using level set methods." *J. Comp. Phys.*, **211**(1):179–197, 2006.

[CN91] W.-P. Chin and S. Ntafos. "Shortest watchman routes in simple polygons." *Desc. Comp. Geom.*, **6**:9–31, 1991.

[CN01] H. Choset and K. Nagatani. "Topological simultaneous localization and mapping (SLAM): Toward exact localization without explicit localization." *IEEE Trans. Robotics and Automation*, **17**(2):125–137, 2001.

[CT97] S. Coorg and S. Teller. "Real-time occlusion culling for models with large occluders." In *SI3D '97: Proceedings of the 1997 symposium on interactive 3D graphics*, 1997.

[CT00] J. A. Castellanos and J. D. Tardós. *Mobile robot localization and map building: A multisensor fusion approach.* Springer, 2000.

[CT05] L.-T. Cheng and Y.-H. R. Tsai. "Visibility optimization using variational approaches." *Comm. Math. Sci.*, **3**:425–451, 2005.

[CT07] L.-T. Cheng and Y.-H. Tsai. "Redistancing by flow of time dependent eikonal equation." UCLA CAM report, 07-16, 2007.

[DDP02] F. Durand, G. Drettakis, and C. Puech. "The 3D visibility complex." *ACM Trans. Graph.*, **21**(2):176–206, 2002.

[Del34] B. Delaunay. "Sur la sphère vide." *Izvestia Akademii Nauk SSSR, Otdelenie Matematicheskikh i Estestvennykh Nauk*, **7**:793–800, 1934.

[DM97] D. Dragomatz and S. Mann. "A classified bibliography of literature on NC milling path generation." *Comp.-Aided Design*, **29**(3):239247, 1997.

[Don95] B. R. Donald. "On information invariants in robotics." *Artif. Intell.*, **72**:217–304, 1995.

[Dur99] F. Durand. *3d visibility: analysis study and applications*. PhD thesis, Univeristy J. Fourier, Grenoble, France, 1999.

[FDF90] J. D. Foley, A. van Dam, S. K. Feiner, and J. F. Hughes. *Computer graphics: Principles and practice*. Addison-Wesley Publishing Co., Reading, MA, second edition, 1990.

[GCB06] A. Ganguli, J. Cortes, and F. Bullo. "Distributed deployment of asynchronous guards in art galleries." In *Proc. of the 2006 American Control Conference, ACC'06*, pp. 14–16, 2006.

[Gho02] M. Ghomi. "Shadows and convexity of surfaces." *Annals of Mathematics*, **155**:281–293, 2002.

[GLL99] L. J. Guibas, J. C. Latombe, S. M. LaValle, D. Lin, and R. Motwani. "A visibility-based pursuit-evasion problem." *Intl. J. Comp. Geom. Appl.*, **9**:471–494, 1999.

[GMR97] L. J. Guibas, R. Motwani, and P. Raghavan. "The robot localization problem." *SIAM J. Comput.*, **26**(4):1120–1138, 1997.

[GO04] J. E. Goodman and J. O'Rourke, editors. *Handbook of discrete and computational geometry*. CRC Press LLC, Boca Raton, FL, second edition, 2004.

[HCH06] C. H. Hsieh, Y.-L. Chuang, Y. Huang, K. K. Leung, A. L. Bertozzi, and E. Frazzoli. "An economical micro-car testbed for validation of cooperative control strategies." In *Proc. of the 2006 American Control Conference, ACC'06*, 2006.

[HEO87] A. Harten, B. Engquist, S. Osher, and S. R. Chakravarthy. "Uniformly high order accurate essentially nonoscillatory schemes, III." *J. Comput. Phys.*, **71**:231–303, 1987.

[HKL99] D. Halperin, L. E. Kavraki, and J. C. Latombe. "Robot algorithms." In M. Atallah, editor, *Handbook of Algorithms and Theory of Computation*. CRC Press, Boca Raton, FL, 1999.

[HKL04] D. Halperin, L. E. Kavraki, and J. C. Latombe. "Robotics." In J. E. Goodman and J. O'Rourke, editors, *Handbook of Discrete and Computational Geometry*. CRC Press, Boca Raton, FL, 2004.

[HS99] C. Hu and C.-W. Shu. "Weighted essentially non-oscillatory schemes on triangular meshes." *J. Comp. Phys.*, **150**(1):97–127, 1999.

[JYT03] H. Jin, A. J. Yezzi, Y.-H. Tsai, L.-T. Cheng, and S. Soatto. "Estimation of 3D surface shape and smooth radiance from 2D images: A level set approach." *J. Sci. Comp.*, **19**(1-3):267–292, 2003.

[KB04] L. Kobbelt and M. Botsch. "A survey of point-based techniques in computer graphics." *Computers & Graphics*, **28**(6):801–814, 2004.

[KKL98] L. Kavraki, M. Kolountzakis, and J. Latombe. "Analysis of probabilistic roadmaps for path planning." *IEEE Trans. Robot. Automat.*, **14**(1):166–171, 1998.

[Kle94] J. M. Kleinberg. "The localization problem for mobile robots." In *IEEE Symp. on Found. of Comp. Sci.*, pp. 521–531, 1994.

[KS01] R. Kimmel and J. A. Sethian. "Optimal algorithm for shape from shading and path planning." *J. Math. Imag. and Vis.*, **14**(3):237–244, 2001.

[KSL96] L. Kavraki, P. Svestka, J. Latombe, and M. Overmars. "Probabilistic roadmaps for path planning in high dimensional configuration spaces." *IEEE Trans. Robot. Automat.*, **12**(4):566–580, 1996.

[KTB07] S. Katz, A. Tal, and R. Basri. "Direct visibility of point sets." In *SIGGRAPH '07: ACM SIGGRAPH 2007 papers*, 2007.

[Lat91] J.-C. Latombe. *Robot motion planning*. Kluwer Academic Publishers, 1991.

[Lau98] J.-P. Laumond, editor. *Robot motion planning and control*, volume 229 of *Lecture Notes in Control and Information Sciences*. Springer-Verlag London Ltd., London, 1998.

[LaV06] S. M. LaValle. *Planning Algorithms*. Cambridge University Press, 2006.

[LeV92] R. J. LeVeque. *Numerical methods for conservation laws*. Birkhaüser, Verlag, second edition, 1992.

[LGH07] Y. Landa, D. Galkowski, Y. R. Huang, A. Joshi, C. Lee, K. K. Leung, G. Malla, J. Treanor, V. Voroninski, A. L. Bertozzi, and Y.-H. R. Tsai. "Robotic path planning and visibility with limited sensor data." In *Proc. American Control Conference, ACC 2007*, pp. 5425–5430, 2007.

[LHH07] K. K. Leung, C. H. Hsieh, Y. R. Huang, A. Joshi, V. Voroninski, and A. L. Bertozzi. "A second generation micro-vehicle testbed for cooperative control and sensing strategies." In *Proc. of the 2007 American Control Conference, ACC'07*, pp. 1900–1907, 2007.

[LO96] F. Lafon and S. Osher. "High order two dimensional non-oscillatory methods for solving Hamilton-Jacobi scalar equations." *J. Comp. Phys.*, **2**:235–253, 1996.

[LOT06] H. Liu, S. Osher, and R. Tsai. "Multi-valued solution and level set methods in computational high frequency wave propagation." *Commun. Comput. Phys.*, **1**(5):765–804, 2006.

[LPC00] M. Levoy, K. Pulli, B. Curless, S. Rusinkiewicz, D. Koller, L. Pereira, M. Ginzton, S. Anderson, J. Davis, J. Ginsberg, J. Shade, and D. Fulk. "The digital Michelangelo project: 3D scanning of large statues." In *SIGGRAPH '00: Proc. of the 27th annual conf. on comp. grap. and interactive techniques*, pp. 131–144, 2000.

[LS87] V. J. Lumelsky and A. A. Stepanov. "Path-planning strategies for a point mobile automaton moving amidst unknown obstacles of arbitrary shape." *Algorithmica*, **2**:403–430, 1987.

[LSS02] S. M. LaValle, B. Simov, and G. Slutzki. "An algorithm for searching a polygonal region with a flashlight." *Int. J. Comp. Geom. Appl.*, **12**(1-2):87–113, 2002.

[LTC06] Y. Landa, R. Tsai, and L.-T. Cheng. "Visibility of point clouds and mapping of unknown environments." In *Advanced Concepts for Intelligent Vision Systems, 2006. ACIVS'06*, pp. 1014–1025, 2006.

[MAW90] D. Miller, D. Atkinson, B. Wilcox, and A. Mishkin. "Autonomous navigation and control of a Mars rover." *Automatic Control in Aerospace*, pp. 111–114, 1990.

[MWB02] A. Makarenko, S. Williams, F. Bourgault, and H. Durrant-Whyte. "An experiment in integrated exploration." In *Proc. of the IEEE/RSJ Intl. Conf. on Intelligent Robots and Systems*, volume 1, pp. 534–539, 2002.

[NBL03] P. Newman, M. Bosse, and J. Leonard. "Autonomous feature-based exploration." In *Proc. Robotics and Automation*, volume 1, pp. 1234–1240, 2003.

[OCK02] S. Osher, L.-T. Cheng, M. Kang, H. Shim, and Y.-H. Tsai. "Geometric optics in a phase space based level set and eulerian framework." *J. Comp. Phys.*, **179**(2):622–648, 2002.

[OL07] J. M. O'Kane and S. M. LaValle. "Localization with limited sensing." *IEEE Transactions on Robotics*, **23**(4):704–716, 2007.

[OR87] J. O'Rourke. *Art gallery theorems and algorithms*. Oxford University Press, New York, NY, 1987.

[OS88] S. Osher and J. A. Sethian. "Fronts propagating with curvature-dependent speed: algorithms based on HamiltonJacobi formulations." *J. Comp. Phys.*, **79**(1):12–49, 1988.

[PKK03] M. Pauly, R. Keiser, L. P. Kobbelt, and M. Gross. "Shape modeling with point-sampled geometry." In *SIGGRAPH '03: ACM SIGGRAPH 2003*, pp. 641–650, 2003.

[PV96] M. Pocchiola and G. Vegter. "The visibility complex." *Int. J. Comp. Geom. & Appl.*, **6**(3):279–308, 1996.

[Rim97] E. Rimon. "Construction of C-Space roadmaps from local sensory data. What should the sensors look for?" *Algorithmica*, **17**(4):357–379, 1997.

[RL00] S. Rusinkiewicz and M. Levoy. "QSplat: a multiresolution point rendering system for large meshes." In *SIGGRAPH '00: Proc. of the 27th annual conference on Computer graphics and interactive techniques*, pp. 343–352, 2000.

[RL01] S. Rajko and S. M. LaValle. "A pursuit-evasion bug algorithm." In *Proc. IEEE Int. Conf. on Robotics and Automation*, volume 2, p. 19541960, 2001.

[ROF92] L. I. Rudin, S. J. Osher, and E. Fatemi. "Nonlinear total variation based noise removal algorithms." *Physica D*, **60**:259–268, 1992.

[Rog97] D. F. Rogers. *Procedural elements for computer graphics*. Mc Graw-Hill, second edition, 1997.

[SA97] J. A. Sethian and D. Adalsteinsson. "An overview of level set methods for etching, deposition, and lithography development." *IEEE Trans. Semiconductor Manufacturing*, **10**(1):167–184, 1997.

[Set99] J. A. Sethian. *Level Set Methods and Fast Marching Methods*. Cambridge Univ. Press, Cambridge, UK, second edition, 1999.

[She92] T. C. Shermer. "Recent results in art galleries." *Proc. IEEE*, **80**(9):1384–1399, 1992.

[SN89] C. Shen and G. Nagy. "Autonomous navigation to provide long-distance surface traverses for Mars rover sample return mission." In *Proc. IEEE International Symposium on Intelligent Control*, pp. 362–367, 1989.

[SO88] C.-W. Shu and S. Osher. "Efficient implementation of essentially non-oscillatory shock capturing schemes." *J. Comput. Phys.*, **77**(2):439–471, 1988.

[SO89] C.-W. Shu and S. Osher. "Efficient implementation of essentially non-oscillatory shock capturing schemes, II." *J. Comput. Phys.*, **83**:32–78, 1989.

[TBF05] S. Thrun, W. Burgard, and D. Fox. *Probabilistic Robotics*. MIT Press, Cambridge, MA, 2005.

[TCO03] Y.-H. R. Tsai, L.-T. Cheng, S. Osher, and H.-K. Zhao. "Fast sweeping methods for a class of Hamilton-Jacobi equations." *SIAM J. Numer. Anal.*, **41**(2):673–694, 2003.

[TCO04] Y.-H. R. Tsai, L.-T. Cheng, S. J. Osher, P. Burchard, and G. Sapiro. "Visibility and its dynamics in a PDE based implicit framework." *J. Comp. Phys.*, **199**:260–290, 2004.

[TGL05] B. Tovar, L. Guilamo, and S. M. LaValle. "Gap navigation trees: A minimal representation for visibility-based tasks." In M. Erdmann, D. Hsu, M. Overmars, and A. F. van der Stappen, editors, *Algorithmic Foundations of Robotics, VI*. Springer-Verlag, Berlin, 2005.

[TL94] G. Turk and M. Levoy. "Zippered polygon meshes from range images." In *Proc. SIGGRAPH '94*, pp. 377–318, 1994.

[TLM03a] B. Tovar, S. M. LaValle, and R. Murrieta. "Locally-optimal navigation in multiply-connected environments without geometric maps." In *Proc. of the IEEE/RSJ Intl. Conf. on Intelligent Robots and Systems*, pp. 3491–3497, 2003.

[TLM03b] B. Tovar, S. M. LaValle, and R. Murrieta. "Optimal navigation and object finding without geometric maps or localization." In *Proc. Robotics and Automation*, pp. 14–19, 2003.

[TML07] B. Tovar, R. Murrieta-Cid, and S. M. LaValle. "Distance-optimal navigation in an unknown environment without sensing distances." *IEEE Trans. Robotics*, **23**(3):506–518, 2007.

[Tsi94] J. Tsitsiklis. "Efficient algorithms for globally optimal trajectories." In *Proc. of the 33rd Conf. on Decision and Control*, pp. 1368–1373, 1994.

[Tsi95] J. Tsitsiklis. "Efficient algorithms for globally optimal trajectories." *Trans. on Automatic Control*, **40**:1528–1538, 1995.

[Urr00] J. Urrutia. "Art gallery and illumination problems." In J. R. Sack and J. Urrutia, editors, *Handbook of Computational Geometry*. Elsevier Science Publishers, 2000.

[VR04] A. Victorino and P. Rives. "An hybrid representation well-adapted to the exploration of large scale indoors environment." In *Proc. Robotics and Automation*, pp. 2930–2935, 2004.

[Wan90] C. M. Wang. "Location estimation and uncertainty analysis for mobile robots." In I. J. Cox and G. T. Wilfong, editors, *Autonomous robot vehicles*. Springer-Verlag, Berlin, 1990.

[WHS05] F. Wolf, A. Howard, and G. S. Sukhatme. "Towards geometric 3D mapping of outdoor environments using mobile robots." In *Proc. IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pp. 1507–1512, 2005.

[WS03] D. F. Wolf and G. S. Sukhatme. "Towards mapping dynamic environments." In *Proc. Int. Conf. Adv. Rob. (ICAR)*, pp. 594–600, 2003.

[Yam97] B. Yamauchi. "A frontier-based approach for autonomous exploration." In *Proc. of the 1997 IEEE Int. Symp. on Computational Intelligence in Robotics and Automation, CIRA '97*, p. 146, 1997.

[YD92] Y. Yacoob and L. Davis. "Computational ground and airborne localization over rough terrain." In *Proc. CVPR '92*, pp. 781–783, 1992.

[ZJK03] F. Zhang, E. Justh, and P. S. Krishnaprasad. "Steering control, curvature and Lyapunov navigation." preprint, 2003.

[ZOF01] H.-K. Zhao, S. J. Osher, and R. Fedkiw. "Fast surface reconstruction using the level set method." *IEEE Workshop on Variational and Level Set Methods (VLSM'01)*, pp. 194–ff, 2001.

[ZOL04] F. Zhang, A. O'Connor, D. Luebke, and P. S. Krishnaprasad. "Experimental study of curvature-based control laws for obstacle avoidance." In *Proc. Robotics and Automation, 2004, ICRA '04*, volume 4, pp. 3849–3854, 2004.

[ZOM00] H.-K. Zhao, S. Osher, B. Merriman, and M. Kang. "Implicit and non-parametric shape reconstruction from unorganized data using variational level set method." *Computer Vision and Image Understanding*, **80**(3):295–314, 2000.

[ZPK02] M. Zwicker, M. Pauly, O. Knoll, and M. Gross. "Pointshop 3D: An interactive system for point-based surface editing." In *SIGGRAPH '02: Proc. of the 29th annual conference on Computer graphics and interactive techniques*, pp. 322–329, 2002.

[ZS03] Y.-T. Zhang and C.-W. Shu. "High order WENO schemes for Hamilton-Jacobi equations on triangular meshes." *SIAM J. Sci. Comput.*, **24**:1005–1030, 2003.