Howie Choset
Sean Walker
Kunnayut Eiamsa-Ard

Carnegie Mellon University
Scaife Hall
Pittsburgh, Pennsylvania 15213 USA
choset@cs.cmu.edu

Joel Burdick

California Institute of Technology
Mail Code 104-44
Pasadena, California 91125 USA

Sensor-Based Exploration: Incremental Construction of the Hierarchical Generalized Voronoi Graph

Abstract

This paper prescribes an incremental procedure to construct roadmaps of unknown environments. Recall that a roadmap is a geometric structure that a robot uses to plan a path between two points in an environment. If the robot knows the roadmap, then it knows the environment. Likewise, if the robot constructs the roadmap, then it has effectively explored the environment. This paper focuses on the hierarchical generalized Voronoi graph (HGVG), detailed in the companion paper in this issue. The incremental construction procedure of the HGVG requires only local distance sensor measurements, and therefore the method can be used as a basis for sensor-based planning algorithms. Simulations and experiments using a mobile robot with ultrasonic sensors verify this approach.

1. Introduction

This paper describes a numerically well-posed and complete algorithm for sensor-based robot mapping of unknown environments. This algorithm produces a *roadmap*, a network of one-dimensional curves that concisely represents the salient geometry of a robot's environment. Once the robot constructs the roadmap, it can use the roadmap to plan paths in the environment, and hence, when the robot constructs a roadmap, it effectively explores an unknown environment.

The incremental construction procedure is general to many roadmaps, but this paper applies the construction algorithm to a roadmap termed the *hierarchical generalized Voronoi graph* (HGVG) which is described in Choset and Burdick (2000). The incremental construction algorithm is applied to the Opportunistic Path Planner (OPP) (Canny and Lin 1993; Rimon and Canny 1994) in Choset (1998).

To map an unknown environment, the robot must systematically move about and sense the environment because most environments do not contain one vantage point from which a robot can "see" the entire world. In other words, the robot cannot turn itself on, "read" in the world, process it, and then construct the HGVG (or any other structure) from one single vantage point. The robot must use an incremental algorithm.

A lot of work in sensor-based planning deals with interleving sensing with motion. Here, we use an incremental construction procedure that automatically determines when to sense and to move. This algorithm uses distance information to numerically construct the HGVG edges. Since sensors provide distance measurements, the numerical procedure readily uses raw sensor data to generate a small portion of an HGVG edge. The robot then moves along this portion, and the procedure is repeated to generate the next segment. This incremental construction technique, therefore, automatically interleves sensing with motion.

The robot traces an edge until it reaches a node in the HGVG, at which point it branches to explore all edges emanating from that node. When all nodes have no unexplored directions (and all cycles have been traversed), the algorithm finishes. This termination property differentiates the HGVG sensor-based construction procedure from other mobile robot techniques: it is complete. In other words, using the HGVG procedure, the robot can conclude it has explored the environment.

Finally, since the HGVG is defined in multidimensional configuration spaces and work spaces, the incremental

The International Journal of Robotics Research
Vol. 19, No. 2, February 2000, pp. 126-148,

construction allows exploration of multidimensional environments. While sensor-based planning motivates this work, the HGVG has many other applications when full knowledge of the world is available. In fact, prior methods for construction Voronoi diagrams (the HGVG in the plane) are limited to point sets, simple polygons, or convex obstacles with known curvature. The incremental construction procedure here does not place any of these restructions on the obstacles; in fact, the obstacle can also be a sampled set of points, which is what sensors really provide.

1.1. Relation to Prior Work

There is a vast literature in sensor-based planning, especially for mobile robots. However, most of this work is not complete and limited to the plane. One class of heuristic algorithms employs a behavioral-based approach in which the robot is armed with a simple set of behaviors (e.g., following a wall) (Brooks 1986). Another heuristic approach involves discretizing a planar world into pixels of some resolution. Typically, this approach handles errors in sonar sensing readings quite well by assigning each pixel a value indicating the likelihood that it overlaps an obstacle (Borenstein and Koren 1990). Strong experimental results indicate the utility of these approaches, and thus these algorithms may provide a future basis for complete sensor-based planners. Unfortunately, these approaches neither afford proofs of correctness that guarantee a path can be found nor offer well-established thresholds for when these heuristic algorithms fail. Finally, these approaches do not typically generalize into higher dimensions.

There are many nonheuristic sensor-based algorithms for which provably correct solutions exist in the plane (Rao et al. 1993). Our approach is to adapt the structure of a provably correct classical motion-planning scheme to sensor-based implementation. One such approach is based on a roadmap (Canny 1988).

Our roadmap approach was motivated by Rimon's work, which adapted Canny and Lin's OPP (Canny and Lin 1993) method to sensor-based use. Originally, the OPP constructs part of its roadmap (the freeways) for a multidimensional work space using local information and is therefore partially incremental. However, the construction of "bridge curves," which guarantee the roadmap's connectivity, requires the identification of "interesting critical points." Complete prior knowledge of the world's geometry is needed to identify the critical points. This is a major limitation of their algorithm for sensor-based implementation. Rimon and Canny (1994) suggested a way to "sensorize" the OPP algorithm. They introduce the notion of a "critical point sensor" and a "minimum clearance" sensor, though the implementation of such sensors is not well detailed. Furthermore, they do not provide a detailed method to construct the freeway segments from sensor data.

In contrast, this paper formulates a method for the construction of roadmap segments from sensor data. Although the construction procedure generates the HGVG (Choset and Burdick 2000), it can be readily adapted to construct other roadmaps, such as the OPP (Choset 1998).

It is worth noting that in the planar case, there have been other Voronoi diagram–based approaches. In Rao, Stolzfus, and Iyengar (1991), an incremental approach to create a Voronoi diagram–like structure was introduced. Also, in Kuipers and Byan (1991), the robot essentially traces double equidistance until a sensor threshold is met, at which point the robot follows the obstacle boundaries. The nodes in this graph are termed distinct places, which are local maxima of the distance to nearby obstacles.

1.2. Overview

The incremental construction algorithms borrow techniques from the numerical curve tracing literature. Initially, the techniques were developed to generate GVG edges (Section 2), but then they were generalized to trace GVG^2 edges (Section 3). The incremental linking procedure is described in Section 5. Next, combining the GVG-based results with some basic nonsmooth analysis, we describe the numerical procedure to effect incremental accessibility (Section 6). The accessibility section may seem out of order because the tracability and accessibility sections use some common results that are easier to visualize in the tracability context. The incremental departability procedure is described in Section 7. The entire algorithm is verified by simulations and experiments that are reviewed in Section 8 and Section 9, respectively.

Recall from Choset and Burdick (2000), that the HGVG is designed to operate in a bounded environment of the world. That is,

ASSUMPTION 1. Boundedness Assumption: The robot operates in a bounded, connected subset of the free space $\mathcal{FS}$. This subset is bounded by obstacles.

Also, the obstacles are positioned in a generic fashion giving rise to equidistant surfaces that transversally intersect. In other words,

ASSUMPTION 2. The Equidistant Surface Transversality Assumption: If equidistant surjective surfaces are manifolds, then they intersect transversally. That is, $\mathcal{SS}_{i_1 \ldots i_k j_1} \overline{\cap} \, \mathcal{SS}_{i_1 \ldots i_k j_2}$ with respect to $\mathcal{SS}_{i_1 \ldots i_k}$ if $j_1 \neq j_2$.

2. Tracability of the GVG

In an incremental context, the property of connectivity is interpreted as *tracability*. More specifically, tracability implies that using only local data, the robot can "trace" the GVG (or HGVG) edges and determine when to terminate the tracing procedure. The robot concludes the edge-tracing process

when it encounters (1) a meet point, a point where GVG edges intersect; (2) a boundary point, a point where a GVG edge intersects an obstacle; or (3) a floating boundary point, a point where two gradient vectors converge on each other (Choset and Burdick 2000) (i.e., become colinear). At each node, the robot begins tracing the appropriate edges that emanate from the node, or returns to a node with unexplored edges emanating from it. In this section, we present and analyze a method for tracing a connected component of the GVG. Now, we will demonstrate how to trace a GVG edge and when to terminate the tracing procedure.

2.1. Properties for Tracing

Our approach borrows some basic ideas and techniques from numerical continuation methods (Keller 1987). Continuation methods are used to trace the roots of the expression $G_1(y, \lambda) = 0$ as the parameter λ is varied. In a sense, incremental construction techniques are an instantiation of the implicit function theorem.

THEOREM 1. Implicit Function Theorem: Let $G : Y \times \mathbb{R} \to Y$ such that

- Y is a Banach space,
- $G(y^*, \lambda^*) = 0$ for some y^* and λ^*
- $\nabla_Y G(y^*, \lambda^*)$ is nonsingular with bounded inverse, i.e., $\|(\nabla_Y G(y^*, \lambda^*))^{-1}\| \leq M$ for some M
- $G(y^*, \lambda^*)$ and $\nabla_Y G(y^*, \lambda^*)$ are continuous in a neighborhood of (y^*, λ^*) denoned $\text{nbhd}(y^*) \times \text{nbhd}(\lambda^*)$

then for all $\lambda \in \text{nbhd}(\lambda^*)$, there exists a $y : \mathbb{R} \to \mathbb{R}^{m-1}$ such that for $y(\lambda) \in \text{nbhd}(y^*)$ such that

- $y(\lambda^*) = y^*$
- $G(y(\lambda), \lambda) = 0$ (existance),
- for all $\lambda \in \text{nbhd}(\lambda^*) \bigcap \mathbb{R}$, there is no solution of $G(y, \lambda) = 0$ in $\text{nbhd}(y^*) \bigcap \mathbb{R}^{m-1}$ other than $y(\lambda)$ (uniqueness),
- $y(\lambda)$ is continuous.

The incremental construction of a GVG edge can be implemented as follows. Let x be a point on the GVG. Choose local coordinates at x so that the first coordinate, z_1, lies in the direction of the tangent to the graph at x (see Fig. 1). At x, let the hyperplane spanned by coordinates $z_2, \ldots, z_m$ be termed the "normal plane." We can thus decompose the local coordinates into $z = (y, \lambda)$, where $\lambda = z_1$ is termed the "sweep" coordinate and $y = (z_2, \ldots, z_m)$ are the "slice" coordinates. Now, let $Y = \mathbb{R}^{m-1}$ and define the function $G_1 : Y \times \mathbb{R} \to Y$ as follows:

$$G_1(y, \lambda) = \begin{bmatrix} (d_1 - d_2)(y, \lambda) \\ (d_1 - d_3)(y, \lambda) \\ \vdots \\ (d_1 - d_m)(y, \lambda) \end{bmatrix}. \quad (1)$$

The function $G_1(y, \lambda)$ assumes a zero value only on the GVG. Let $\nabla_Y G_1$ be the matrix[1] formed by taking the derivative of eq. 1 with respect to the Y coordinates. It takes the form

$$\nabla_Y G_1(y, \lambda) = \begin{bmatrix} (\nabla_Y d_1(y, \lambda) - \nabla_Y d_2(y, \lambda))^T \\ (\nabla_Y d_1(y, \lambda) - \nabla_Y d_3(y, \lambda))^T \\ \vdots \\ (\nabla_Y d_1(y, \lambda) - \nabla_Y d_m(y, \lambda))^T \end{bmatrix}, \quad (2)$$

where ∇_Y denotes the gradient with respect to the y-coordinates. If $\nabla_Y G_1(y, \lambda)$ is surjective at $x = (\lambda, y)^T$, then the implicit function theorem implies that the roots of $G_1(y, \lambda)$ locally define a GVG edge as λ is varied. By numerically tracing the roots of this function, we can locally construct an edge.

While there are a number of curve-tracing techniques (Keller 1987), we use an adaptation of a common predictor-corrector scheme, as illustrated in Figure 1. Assume that the robot is located at a point x on the GVG. The robot takes a "small" step, $\Delta\lambda$, in the z_1-direction (i.e., the tangent to the local GVG edge). In general, this *prediction step* takes the robot off the GVG. Next, a *correction method* is used to bring the robot back onto the GVG. If $\Delta\lambda$ is small, then the graph will intersect a *correcting plane,* which is a plane orthogonal to the z_1-direction at a distance $\Delta\lambda$ away from the origin. The correction step finds the location where the GVG intersects the correcting plane and is an application of the Newton convergence theorem (Keller 1987).

THEOREM 2. Newton-Raphson Convergence Theorem: Let $G : Y \to Y$ such that Y is a Banach space and $G(y^*) = 0$. For some $\rho > 0$, let G satisfy

- $\nabla G(y^*)$ is nonsingular with bounded inverse, i.e., $\|(\nabla G(y^*))^{-1}\| \leq \beta$
- $\|\nabla G(x) - \nabla G(y)\| \leq \gamma \|x - y\|$ for all $x, y \in B_\rho(y^*)$.
- $\rho\beta\gamma \leq \frac{2}{3}$

Then for every $y^0 \in B_\rho(y^*)$ (ball of radius ρ), the iterates,

$$y^{h+1} = y^h - (\nabla G(y^h))^{-1} G(y^h),$$

satisfy

1. Here, we are abusing notation. $D_Y G_1$ is more conventional, but we use D as the multiobject distance function and therefore want to avoid using D here.

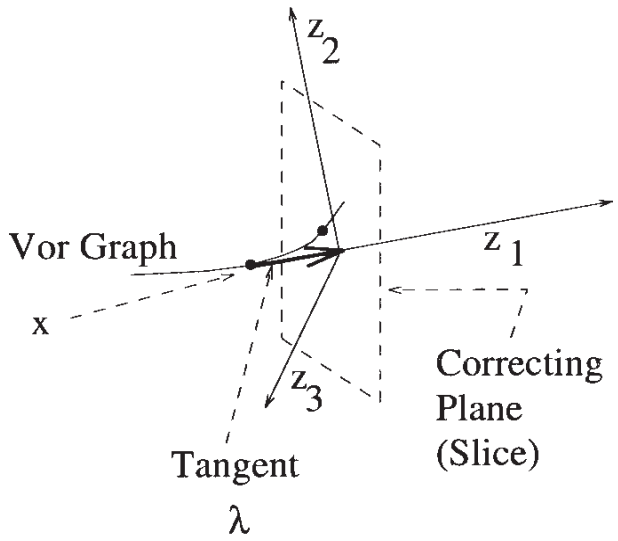

Fig. 1. Continuation method.

- $y^h \in B_\rho(y^*)$,
- $\{y^h\}$ quadratically converge onto y^*, i.e.,

$$\|y^{h+1} - y^*\| \leq a\|y^h - y^*\|^2,$$

where $a = \frac{\beta\gamma}{2(1-\rho\beta\gamma)} < \frac{1}{\rho}$.

We will show that $\nabla_Y G_1(y, \lambda)$ is full rank at every (y, λ) in a neighborhood of the GVG, and so it is possible to use an iterative Newton's method to implement the corrector step. If y^h and λ^h are the hth estimates of y and λ, the $h+1$st iteration is defined as

$$y^{h+1} = y^h - (\nabla_Y G_1)^{-1} G_1(y^h, \lambda^h), \tag{3}$$

where $\nabla_Y G_1$ is evaluated at (y^h, λ^h). After taking the prediction step, the goal of the correction step is to find where the GVG locally intersects the "correcting plane."

There are several things worth noting about this method. First, to evaluate $G_1(y, \lambda)$ and $\nabla_Y G_1(y, \lambda)$, one only needs to know the distance and direction to the m objects that are closest to the robot's current location—information that is easily obtained from local distance sensor data. Second, Newton methods are quadratic in their convergence, and thus they would be substantially faster than the naive gradient ascent techniques. Third, $\nabla_Y G_1(y, \lambda)$ is an $(m-1) \times (m-1)$ matrix, and is thus typically quite small in size (e.g., a scalar for two-dimensional environments, or a 2×2 matrix for three-dimensional environments), and the method is not computationally burdensome. In fact, we symbolically invert the matrix once and use the result in the actual programs that generate the HGVG. Finally, there is the issue for how big a step-size $\Delta\lambda$ should be. This step-size λ should keep the robot in the ball $B_\rho(y^*)$ for Newton's method to work; currently, determining a rigorous upper bound for the size of the ball is a topic of research, but in our experiments, we use a step-size of one robot diameter.

The following two subsections demonstrate that equation (3) is well posed because $(\nabla_Y G_1(y, \lambda))^{-1}$ is defined, and that we can always compute (using local sensor data) a vector that is tangent to the GVG. In proving these assertions, several new and useful properties of the generalized Voronoi graph are presented.

2.1.1. Computing the Tangent to the Graph.

We first tackle the question of how to determine the tangent to a GVG edge from sensor data. Recall that the Voronoi graph (Avis and Bhattacharya 1983) was defined for point sites. To better distinguish it from the GVG, let the *regular Voronoi graph* (RVG) be the Voronoi graph for the case in which the obstacles are points. Furthermore, let the *regular* k*-equidistant face*, $\mathcal{R}_{i_1 \ldots i_k}$, be a k-equidistant face whose k closest obstacles are points. In m dimensions, a regular m-equidistant face is an RVG edge and it is equidistant to m closest point objects.

The following proposition produces the tangent to the GVG by exploiting the coincidence of a GVG edge and an RVG edge at a point x where the GVG edge is defined by $\{C_i : i = 1, \ldots, m\}$ and the RVG edge is defined by $\{c_i : i = 1, \ldots, m\}$, the m closest points on the m closest obstacles.

PROPOSITION 1. The tangent to a GVG edge at x is defined by the vector orthogonal to the hyperplane, which contains the m closest points, $c_1, \ldots, c_m$, of the m closest objects, $C_1, \ldots, C_m$.

Proof. This proposition is a simple consequence of the following two lemmas when $k = m$. The proofs of these lemmas appear in the appendix.

LEMMA 1. Let $c_1, \ldots, c_k$ be the k closest obstacle points to a point x. Let $\mathcal{R}_{i_1 \ldots i_k}$ be the regular Voronoi graph face defined by these points where $k \leq m$. When Assumption 1.2 is upheld, any vector in the tangent space $T_x \mathcal{R}_{i_1 \ldots i_k}$ is orthogonal to the $(k-1)$-dimensional affine hull of $c_1, \ldots, c_k$. The tangent space $T_x \mathcal{R}_{i_1 \ldots i_k}$ is also orthogonal to the $(k-1)$-dimensional affine hull of the heads of the gradient vectors based at x.

LEMMA 2. Let $c_1, \ldots, c_k$ be the closest points in the k nearest obstacles to $x \in \mathcal{F}_{i_1 \ldots i_k}$. At a point x in the k-equidistant face, the tangent space $T_x \mathcal{F}_{i_1 \ldots i_k}$ is the same as the tangent space $T_x \mathcal{R}_{i_1 \ldots i_k}$, where $\mathcal{R}_{i_1 \ldots i_k}$ is the regular k-equidistant face defined by $c_1, \ldots, c_k$.

Let x be a point on a GVG edge defined by the obstacles $C_1, \ldots, C_m$. The m closest points $c_1, \ldots, c_m$ of the m closest obstacles define an RVG edge. When $k = m$, Lemma 1 asserts that the tangent space of the RVG edge at x is a

one-dimensional vector space whose basis vector is orthogonal to the hyperplane that contains the m closest points $c_1, \ldots, c_m$.

Lemma 2 contends that the tangent space at x of the RVG edge, defined by $c_1, \ldots, c_m$, is the same as the tangent space at x of the GVG edge defined by $C_1, \ldots, C_m$. Thus, by knowing the distance and direction to the m nearest points, the tangent to a generalized Voronoi graph edge is easily computed. □

EXAMPLE 1. Two-Dimensional World: Let C_1 and C_2 be the two closest obstacles to a point x on $\mathcal{F}_{12}$. Let c_1 and c_2 be the two closest points on the two closest obstacles. Pass a line through c_1 and c_2; parallel shift this line so it passes through x. The displaced line is the normal plane and the line orthogonal to the normal plane is the tangent space. See Figure 2.

EXAMPLE 2. Three-Dimensional World: Let C_1, C_2, and C_3 be the three closest obstacles to a point x on $\mathcal{F}_{123}$. Let c_1, c_2, and c_3 be the three closest points on the three closest obstacles. The tangent to the GVG at x is a vector that is normal to the plane defined by c_1, c_2, and c_3. Let $\bar{c}_{12}$ be the vector formed by subtracting c_2 from c_1. Let $\bar{c}_{13}$ be defined in a likewise manner. The normal to the plane that contains c_1, c_2, and c_3 is collinear with the vector $\bar{c}_{12} \times \bar{c}_{13}$.

2.1.2. The Matrix $\nabla_Y G_1$ is Invertible

Now we can take a step along the tangent direction of the GVG. If this tangent step takes the robot off of the GVG, then the robot must invoke a correction procedure on a hyperplane orthogonal to the tangent. This correction procedure is described in eq. 3. The following proposition illustrates that the numerical procedure defined by eq. 3 is well posed for $\Delta\lambda$ sufficiently small.

PROPOSITION 2. Equidistant Surface Full Rank Property: $\nabla_y G_1(y, \lambda)$ has full rank (i.e., has rank $(m-1)$) on the correcting plane in a neighborhood of the GVG.

Proof. The following two lemmas are necessary in showing $\nabla_y G_1(x)$ is full rank. These lemmas furnish a general result for the function $G : \mathbb{R}^m \to \mathbb{R}^q$, which is defined as

$$G(x) = \begin{bmatrix} (d_{i_1} - d_{j_1})(x) \\ \vdots \\ (d_{i_q} - d_{j_q})(x) \end{bmatrix}. \tag{4}$$

If for all $r_1, r_2 \in \{1, \ldots, n\}$,

$$\{i_{r_1}, j_{r_1}\} \neq \{i_{r_2}, j_{r_2}\}, \tag{5}$$

then $G^{-1}(0)$ represents the intersection of q *distinct* two-equidistant surjective surfaces, i.e., $G^{-1}(0) = \mathcal{SS}_{i_1 j_1} \bigcap \cdots \bigcap \mathcal{SS}_{i_q j_q}$. When the condition in eq. 5 is met and

$$\begin{aligned} q &= m-1, \\ i_r &= 1, \qquad \text{for } r = 1, \ldots, m-1, \\ j_r &= r+1, \qquad \text{for } r = 1, \ldots, m-1, \end{aligned} \tag{6}$$

$G^{-1}(0)$ is the intersection of $m-1$ two-equidistant surjective surfaces, which gives rise to a GVG edge. In other words, $G(x) = G_1(x)$. From here, this proof is now a simple consequence of the following (whose proofs appear in the appendix):

LEMMA 3. Consider the mapping $G : \mathbb{R}^m \to \mathbb{R}^q$ defined as

$$G(x) = \begin{bmatrix} (d_{i_1} - d_{j_1})(x) \\ \vdots \\ (d_{i_q} - d_{j_q})(x) \end{bmatrix}. \tag{7}$$

The rank of $\nabla G(x)$ is q for all $x \in \mathcal{SS}_{i_1 j_1} \bigcap \cdots \bigcap \mathcal{SS}_{i_q j_q}$, when

$$\mathcal{SS}_{i_1 j_1} \bigcap \cdots \bigcap \mathcal{SS}_{i_q j_q} \neq \emptyset$$

and for all r_1, r_2, $\{i_{r_1}, j_{r_1}\} \neq \{i_{r_2}, j_{r_2}\}$. That is, each pair $\{i_r, j_r\}$ is unique.

LEMMA 4. Consider the mapping $G : \mathbb{R}^m \to \mathbb{R}^q$ defined as

$$G(x) = \begin{bmatrix} (d_{i_1} - d_{j_1})(x) \\ \vdots \\ (d_{i_q} - d_{j_q})(x) \end{bmatrix}. \tag{8}$$

On the normal slice plane (and all planes parallel to it) $\text{rank}(\nabla_y G) = \text{rank}\ (\nabla G)$ for $x \in \mathcal{SS}_{i_1 j_1} \bigcap \cdots \bigcap \mathcal{SS}_{i_q j_q}$, when $\mathcal{SS}_{i_1 j_1} \bigcap \cdots \bigcap \mathcal{SS}_{i_q j_q} \neq \emptyset$ and each pair $\{i_r, j_r\}$ is unique. That is, for all r_1, r_2, $\{i_{r_1}, j_{r_1}\} \neq \{i_{r_2}, j_{r_2}\}$.

The matrix ∇G_1 is an $m-1$ by $m-1$ matrix, and thus by Lemma 3, the rank of ∇G_1 is $m-1$. Lemma 4 asserts that $\text{rank}(\nabla_y G_1)$ is $m-1$ for all $x \in \mathcal{F}_{i_l \ldots i_m}$ and therefore must be invertible at x.

Since the rank operation is a continuous function, $\nabla_y G$ must be invertible in an open neighborhood around $x = (y, \lambda) \in \mathcal{F}^m$. This open neighborhood will intersect the correcting plane for $\Delta\lambda$ sufficiently small, and thus $\nabla_y G$ is invertible on the correcting plane as well. □

In practice, the neighborhood of invertibility is quite large with this method. Practically speaking, this result states that the numerical procedure defined by eq. 3 will be robust for reasonable errors in robot position, sensor errors, and numerical round-off errors.

Naively, one could trace an edge by repeated application of the accessibility method. That is, the robot would move a small distance along a given direction—either a fixed direction or perhaps the tangent direction to the current edge. Gradient ascent would then be used to move back onto the local edge. The OPP (Canny and Lin 1993) method and its sensor-based adaptation (Rimon and Canny 1994) use this strategy

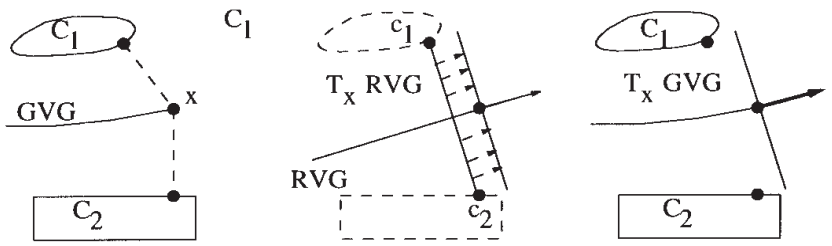

Fig. 2. The tangent space is the line orthogonal to the line that connects the two closest points on the two closest obstacles.

with a fixed stepping direction. However, gradient ascent can be a computationally expensive procedure because of its slow convergence. Also, the constant step direction leads to undesirable roadmap artifacts (Choset and Burdick 1994).

2.2. Terminating Conditions

So far, we have shown that the robot can trace a GVG edge, but when does a tracing procedure stop? Due to the boundedness of the robot's environment, the GVG edges must terminate, as stated in the following proposition.

PROPOSITION 3. Given the Equidistant Surface Transversality Assumption, in a bounded environment, if a generalized Voronoi edge is not a cycle (a GVG edge C^2 diffeomorphic to a circle), it must terminate (1) at a generalized Voronoi vertex (a meet point), (2) on the boundary of the environment, or (3) at a point where two gradients of single object distance functions become collinear.

Proof. This proof is a consequence of the following proposition from Choset and Burdick (2000) when $k = m$.

PROPOSITION 4. If a $(k+1)$-equidistant face $\mathcal{F}_{i_1 \ldots i_{k+1}}$ is nonempty, then the k-equidistant face $\mathcal{F}_{i_1 \ldots i_k}$ must also be nonempty; however, the converse is not necessarily true. Furthermore,

$$\partial \mathcal{F}_{i_1 \ldots i_k} = \mathcal{F}_{i_1 \ldots i_k i_{k+1}} \bigcup C_{i_1 \ldots i_k} \bigcup FC_{i_1 \ldots i_k}.$$

□

When the GVG edge is cycle, the edge tracing procedure terminates when the robot circumnavigates the cycle. This procedure requires that the robot possess an accurate dead reckoning system.

Incremental construction of the GVG is akin to a graph search method where the generalized Voronoi edges are the "edges" and the meet points and boundary points are the "nodes." Once the robot has accessed a point on the GVG, it begins tracing an edge. If the robot encounters a meet point, it marks the direction from where it came as explored, and then explores one of the other m edges that emanate from the meet point. It also marks that direction as being explored. If the robot encounters another unvisited meet point, the above procedure is recursively repeated. When the robot reaches a boundary point, it simply turns around and retraces its path to some previous meet point with unexplored directions. The robot terminates exploration of the GVG component (i.e., there may be other disconnected GVG components) when there are no more unexplored directions at any meet point. If the robot is looking for a particular destination whose coordinates are known, then the robot can invoke graph-searching techniques such as the A-star algorithm or depth first search algorithm.

2.2.1. Meet Point Detection

Finding the meet points is essential to proper construction of the graph. While meet points occur when the robot is equidistant to $m+1$ objects, it is unreasonable to expect that a robot can exactly detect such points. For example, while tracing an edge, it is unlikely that the robot will pass exactly through an $m+1$ equidistant point. Furthermore, sensor error may make such detection difficult. Nevertheless, as shown in Figure 3, meet points can be robustly detected by watching for an abrupt change in the direction of the (negated) gradients to the m closest obstacles. Such a change will occur in the vicinity of a meet point.

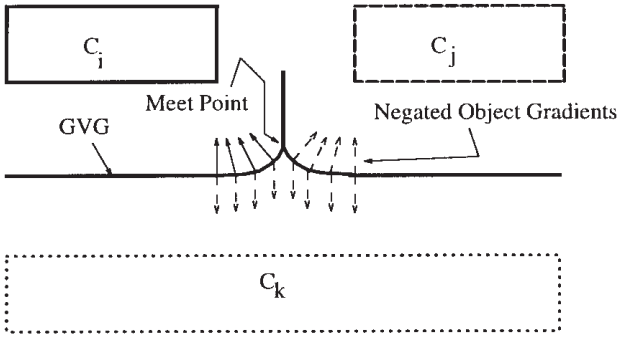

Fig. 3. Meet point detection.

2.2.2. Departing a Meet Point

Recall that the robot is equidistant to $m+1$ objects at a meet point. It must be able to identify and explore the $m+1$ GVG edges that emanate from each meet point to completely construct the GVG. Note that each emanating edge corresponds to an m-wise combination of the $m+1$ closest objects. Assume that we wish to explore and trace the edge corresponding to objects $C_1, \ldots, C_m$, the distances to which are $d_1(x) = d_2(x) = \cdots = d_m(x)$, respectively. Proposition 1 yields the one-dimensional tangent space to the generalized Voronoi edge corresponding to these m objects. If v is a basis vector of this GVG edge's tangent space, the robot must determine if it should depart the meet point in the $+v$ or $-v$ direction. We want the robot to move *away* from the $m+1$st obstacle, the distance to which is $d_{m+1}(x)$. If $\langle \nabla d_{m+1}, v \rangle > \langle \nabla d_i, v \rangle$, where $i \in \{1, \ldots, m\}$, then the robot should move in direction $+v$, otherwise $-v$. This effects motion away from C_{m+1}.

3. Constructing the Second-Order GVG

The GVG serves as the backbone for the HGVG roadmap. This section applies the curve-tracing methods of the previous section to trace the edges of the HGVG, which connect the GVG when the GVG is disconnected. In this section, however, we only consider edges that can be represented by the roots of a function G, i.e., GVG^2 equidistant edges, boundary edges, and floating boundary edges. Occluding edges are considered separately because they are points where functions become discontinuous.

3.1. Second-Order Generalized Voronoi Edges

The second- (and higher) order GVG can be incrementally constructed in an analogous fashion. The key is to define a function whose roots define GVG^2 edges. The roots of the function

$$G_2(y, \lambda) = \begin{bmatrix} (d_1 - d_2)(y, \lambda) \\ (d_3 - d_4)(y, \lambda) \\ \vdots \\ (d_3 - d_m)(y, \lambda) \end{bmatrix} \tag{9}$$

locally trace out a GVG^2 equidistant edge. The first row of G_2 enforces equidistance between the closest objects C_1 and C_2. The remaining rows enforce equidistance between the second closest objects. Again, a predictor-corrector algorithm is used.

3.1.1. Computing the Tangent

The tangent to the GVG^2 edges is the null space of

$$\nabla_y G_2(y, \lambda) = \begin{bmatrix} (\nabla_y (d_1 - d_2)(y, \lambda))^T \\ (\nabla_y (d_3 - d_4)(y, \lambda))^T \\ \vdots \\ (\nabla_y (d_3 - d_m)(y, \lambda))^T \end{bmatrix}. \tag{10}$$

In $\mathbb{R}^3$, this can be easily computed with local sensor information. In $\mathbb{R}^3$,

$$\nabla_y G_2(y, \lambda) = \begin{bmatrix} (\nabla_y (d_1 - d_2)(y, \lambda))^T \\ (\nabla_y (d_3 - d_4)(y, \lambda))^T \end{bmatrix}. \tag{11}$$

In $\mathbb{R}^3$, the null space of $\nabla_y G_2(y, \lambda)$ is the set of vectors, v, for which

$$\begin{aligned} \langle \nabla_y (d_1 - d_2)(y, \lambda), v \rangle = 0 \ \text{ and} \\ \langle \nabla_y (d_3 - d_4)(y, \lambda), v \rangle = 0. \end{aligned} \tag{12}$$

That is, the tangent to a GVG^2 equidistant edge is the intersection of the tangent spaces $\mathcal{SS}_{12}$ and $\mathcal{SS}_{34}$. The Equidistant Surface Transversality Assumption guarantees that these tangent spaces transversally intersect and thus their intersection is one-dimensional. A basis vector for this tangent space is $\nabla_y (d_1 - d_2)(y, \lambda) \times \nabla_y (d_1 - d_3)(y, \lambda)$. Since the tangent space is computed from the cross product of gradient vectors, the tangent space can be readily computed from sensor information.

3.1.2. The Matrix ∇G_2 Is Invertible

When

$$\begin{aligned} q &= m - 1, \\ i_1 &= 1, \\ j_1 &= 2, \\ i_r &= 3, \qquad \text{for } r = 2, \ldots, m-1, \\ j_r &= r + 2, \qquad \text{for } r = 2, \ldots, m-1, \end{aligned} \tag{13}$$

Lemmas 3 and 4 guarantee the matrix $\nabla_y G_2(y, \lambda)$ has full rank in a neighborhood of the GVG^2 on the correcting plane.

3.2. Boundary Edges

The incremental tracing of boundary edges requires that the robot move along the perimeter of the environment where m obstacles meet in m dimensions. This can be done by tracing

the roots of the following function, G_b, defined as

$$G_b(y,\lambda) = \begin{bmatrix} d_1(y,\lambda) - \epsilon \\ d_3(y,\lambda) - \epsilon \\ \vdots \\ d_{m-1}(y,\lambda) - \epsilon \end{bmatrix}, \tag{14}$$

where in this case ϵ is a small "safety" distance away from the environment. Continuity of the single object distance function guarantees there exists a small enough $\epsilon > 0$ such that the topology of the traced edges reflects that of the actual boundary edges. In future work, we will see that G_b will be used in tracing edges of the *saturated generalized Voronoi graph*, which is a roadmap used when sensors function over a finite range, less than ϵ.

3.3. Floating Boundary Edges

Floating boundary edges are straight-line segments and thus do not require complicated numerical curve-tracing techniques. Proposition 4 asserts that a floating boundary edge terminates at either a GVG edge end point or a boundary edge (C_{ij}) end point, which is a point where two convex obstacles (C_i and C_j) merge into one convex obstacle.

When a GVG edge terminates at a floating boundary edge point, the basis vector of the floating boundary edge must be determined. Let x^* be the point where the two gradient vectors converge and let v be the limiting vector of $\nabla d_i(x)$ (or $\nabla d_j(x)$) as x approaches x^*. The vector v is the basis vector of the floating boundary edge point, and after encountering the floating boundary edge point, the robot moves in the $-v$ direction. When a boundary edge terminates at a floating boundary edge, the robot moves in a direction v, as described above. (Note that when obstacles are polyhedra, the floating boundary edge is a straight-line extension and a boundary edge is also a straight line.)

3.4. Terminating Conditions

In summary, the GVG^2 has the similar terminating conditions as the GVG: a *second-order meet point*, *second-order boundary point*, *second-order floating boundary point*, and *second-order cycle*. The second-order meet points are detected in a fashion analogous to the (first-order) meet points—the robot looks for a change in the gradients to the second nearest object while maintaining equidistance to the two nearest objects. At a second- order boundary point, the robot does not necessarily turn around and retrace its steps to the previous second-order meet point with unexplored directions. Instead, it traces both of the directions of the boundary edge it intersects (second-order equidistant edges intersect boundary edges only in the interior).

4. Occluding Edges

So far, we have described the tracing procedure for HGVG edges that can be represented by the roots of a differentiable function. Since occluding edges cannot be described this way, they present a bigger challenge to edge tracing. In this section, we describe a method for tracing discontinuities. The underlying method of generating occluding edges traces the boundaries of the adjacent second-order generalized Voronoi regions who share the common occluding edge, as opposed to explicitly tracing the occluding edge.

4.1. Accessing the Occluding Edge

The robot arrives at an occluding edge $V_{kl}|_{\mathcal{F}_{ij}}$, either as a result of a linking procedure or tracing an HGVG edge. In the latter case, an HGVG edge either terminated at a floating boundary edge, a second-order floating boundary edge, or at some point in the interior of a boundary edge.

Let x^0_{kl} be the point of arrival on the occluding edge $V_{kl}|_{\mathcal{F}_{ij}}$. Recall that an occluding edge lies on the shared boundary of two second-order generalized Voronoi regions where the distance to the second closest obstacle changes discontinuously. In actuality, the robot stores $x^0_k \in \mathcal{F}_k|_{\mathcal{F}_{ij}}$ and $x^0_l \in \mathcal{F}_l|_{\mathcal{F}_{ij}}$, which are the points on the boundary of their respective second-order generalized Voronoi region near their boundaries. Either x^0_k or x^0_l are approximations to x^0_{kl}.

At this point, the robot draws a small circle around x^0_k (or x^0_l) contained in $\mathcal{F}_{ij}$ and determines all points on this circle that are in occluding edges also. Again, it looks for pairs of points where the distance to the second closest obstacles drastically changes. Assume for the sake of discussion that $\mathcal{F}_{ij}$ is locally flat. If there are only two additional occluding points, say x^1_{kl} and $x^{1'}_{kl}$, and they are collinear, then x^0_{kl} is a point in the interior of the occluding edge $V_{kl}|_{\mathcal{F}_{ij}}$ (between x^1_{kl} and $x^{1'}_{kl}$). In fact, the line defined by x^1_{kl} and $x^{1'}_{kl}$ is the "tangent" of the occluding edge (see Fig. 4).

Now consider the scenario where x^1_{kl} and $x^{1'}_{kl}$ do not define a line or there are points in the circle from multiple occluding edges. Since we can let the circle be as small as we want, we are guaranteed that there is at least one point x^1_{kl} on the circle that belongs to the same occluding edge as x^0_{kl} and is in a straight line.

Again, a point on the occluding edge is represented by two points, one from each second-order generalized Voronoi region. So, when the robot searches for a point on the circle associated with the same occluding edge as x^0_{kl}, it is really looking for the appropriate pair of points. The robot performs this search by drawing two segments between the access' pair of points and the candidate pair on the circle. If both line segments are fully contained in their respective second-order generalized Voronoi regions, then the access point and the candidate point on the circle form a straight-line segment,

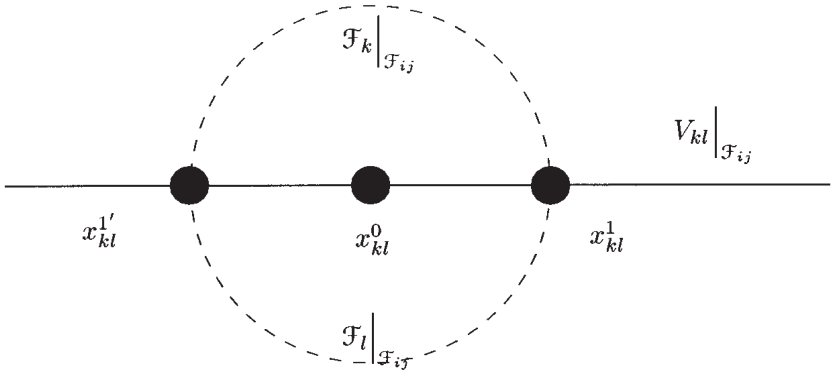

Fig. 4. Local neighborhood of occluding edge $V_{kl}|_{\mathcal{F}_{ij}}$.

albeit very small, of an occluding period. The initial tangent vector is based on the perimeter of the circle pointing toward the access point, i.e., $x_{kl}^1 - x_{kl}^0$.

If multiple points on the circle satisfy this no-cutting-the-corner condition, then the robot simply picks one direction to go. If all points on the circle satisfy this test, then the robot is at a node of an occluding period. Since we are only looking for a tangent direction, the robot simply picks one direction to initiate tracing.

4.2. Tracing the Occluding Edge

The robot traces the occluding edge by moving in the direction of the tangent. In actuality, both x_k^1 and x_l^1 are translated in the tangent direction, not x_{kl}^1. Since $\mathcal{F}_{ij}$ is nominally not flat, x_k^1 or x_l^1 may fall off the sheet, so we invoke the similar Newton iterative correction procedure to bring them back onto $\mathcal{F}_{ij}$. (Note that correction onto a sheet is described in Section 6.)

The robot then steps again in the tangent direction of the occluding edge. This tangent is approximated by $x_k^h - x_k^{h-1}$ (or $x_l^h - x_l^{h-1}$), where h is the hth step the robot has taken since it accessed the occluding edge and the subscript still indicates which region contains the point. Again, both x_k^h and x_l^h are translated in the tangent direction.

The occluding edge may also have curvature, in which case $x_k^h - x_k^{h-1}$ takes the robot off of the occluding edge, even after correcting back onto the two-equidistant sheet. Therefore, the robot moves along a line L, which is the projection of the hyperplane orthogonal to the current tangent.

The direction to move is determined as follows: let x_k, x_l be the current location of the robot after it corrected back onto the two-equidistant sheet, and assume without loss of generality $x_k, x_l \in \mathcal{F}_k|_{\mathcal{F}_{ij}}$ and L passes through x_k and x_l. The robot moves along L away from x_k until the robot reaches a point where there is a change in the second closest obstacle. This point is x_l^{h+1} and the previous point before the change is x_k^{h+1}. So, there are two types of correction: a correction step onto the two-equidistant sheet and one along the two-equidistant sheet to the occluding edge.

Figures 7 and 8 display only a box that is in the middle of a rectangular enclosure. The halolike structure surrounding the box is a GVG cycle. Emanating from the cycle is a link that terminates at an occluding edge. The occluding edge is the rectangular-like structure hovering over the box.

The outer halo surrounding the box is a GVG edge, which is three-way equidistant to the box, the floor, and the ceiling. The floor has been removed. The square period floating on top of the box is a collection of occluding edges that form a period. Points inside of the occluding period over the box cannot look straight down to the floor.

4.3. Terminating Conditions

The robot repeats this pairwise step and correct procedure until it encounters an occluding corner, an occluding meet point, a GVG floating boundary point, a $\text{GVG}^2|_{\mathcal{F}_{ij}}$ floating boundary point, or a boundary point. These nodes are essentially detected when the above described correction procedures fail to converge onto an occluding edge. In such a case, at least one of the pairs of points being traced has changed second-order generalized Voronoi regions. Now, we describe the methods to detect which node the robot has encountered.

Unlike the rest of the edges in the HGVG, an occluding edge is not guaranteed to be C^2-diffeomorphic to $\mathbb{R}^1$. The occluding edges can have nonsmooth kinks, which are termed occluding corners. The robot can detect occluding corners by looking at x_k^{h+1} and x_l^{h+1}. When both points are in the same second-order generalized Voronoi region (e.g., $x_k^{h+1}, x_l^{h+1} \in \mathcal{F}_k|_{\mathcal{F}_{ij}}$), then the robot has passed by a corner point. By taking successively small steps from the hth iteration, the robot can hone in on the true location of the corner point. The robot determines the new direction to trace using the same circle-based approach when accessing the occluding edge.

An occluding meet point is where two or more occluding edges meet. This can occur in a variety of ways:

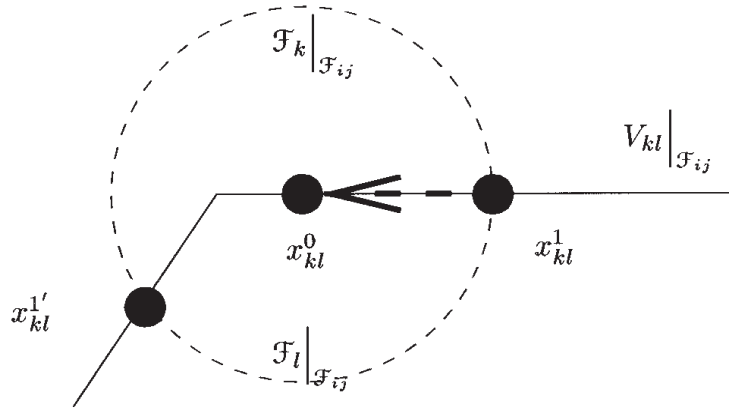

Fig. 5. Local neighborhood of occluding edge $V_{kl}|_{\mathcal{F}_{ij}}$.

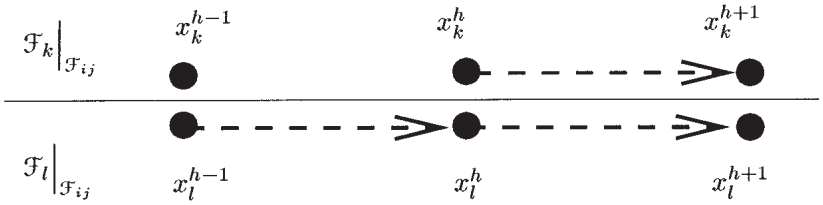

Fig. 6. Tracing an occluding edge $V_{kl}|_{\mathcal{F}_{ij}}$.

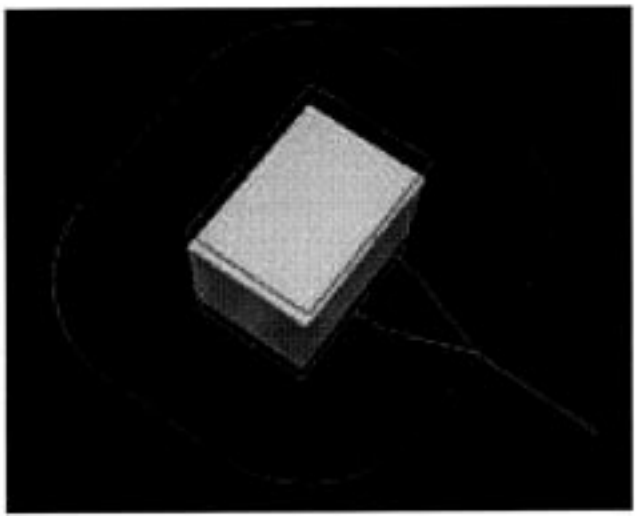

Fig. 7. The GVG is disconnected.

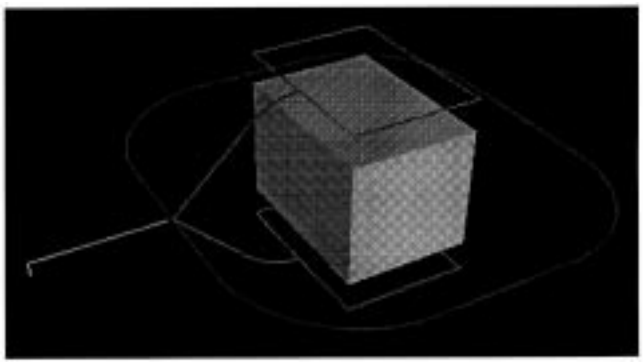

Fig. 8. The HGVG is connected.

T-intersection, Y-intersection, a cross-intersection, and so on. The robot detects occluding meet points in a similar fashion as it does meeting points and second-order meet points. The robot looks for a change in one of the points from the pair it is tracing; if either point enters a *new* second-order generalized Voronoi region ($\mathcal{F}_p|_{\mathcal{F}_{ij}}$ where $p \neq k$ and $p \neq l$), then it has passed by an occluding meet point. At this point, the robot takes a step back to the previous iteration and hones in on the occluding meeting point by taking successively smaller

steps toward it. The robot determines the new directions to trace using the same circle-based approach when accessing the occluding edge.

4.4. Departing the Occluding Edge

The robot encounters other HGVG nodes—floating boundary points, GVG^2 floating boundary points, and isolated points in the interior of the boundary edge—much in the same way it finds nodes in the occluding period. In fact, when the robot is tracing an occluding period, it finds the node first and then determines what type of node it is. Take, for example, a second-order floating boundary that has occluding edges and one GVG^2 equidistant edge emanating from it. When the robot initiates motion along the GVG^2 equidistant edge, it initially treats the edge as an occluding edge. In other words, it traces two points x_k and x_l along the shared boundary of the adjacent second-order generalized Voronoi regions. Nominally, if the distances to the second closest obstacles of each of these points are the same, then the robot can conclude it has started tracing a GVG^2 equidistant edge and invoke the appropriate procedure. In practice, these distances will never be identical, so one would naturally look for a threshold of difference of these distances. However, there is a better way. If C_l is within line of sight of x_k and C_k is within line of x_l, then the robot is tracing a GVG^2 equidistant edge because there is no occlusion.

Departing onto a boundary edge is quite simple: when the distance of the *first* pair of closest obstacles goes to zero, then the robot has encountered a boundary edge at which point it can branch into two directions along the boundary edge.

5. Incremental Linking

The companion paper (Choset and Burdick 2000) outlines four types of links to and from the following: an inner GVG^2 period, a boundary period, an occluding boundary period, and a GVG cycle.

Recall from Choset and Burdick (2000) that we introduced the notion of an inner and outer boundary for a second-order generalized Voronoi region. There are situations in which the robot can infer if it is on an outer or inner boundary component. For example, while traversing a boundary component of a second-order generalized Voronoi region, if the robot detects a boundary period, then the robot is on an outer boundary component (Choset and Burdick 2000) and the boundary period is an inner boundary component. In our current implementation, the robot exploits the fact that it is tracing a one-dimensional closed curve (the boundary of a second-order generalized Voronoi region) on a two-dimensional surface to determine the outer and inner boundaries.

5.1. Links to Cycles

Linking to (from) GVG cycles, if they exit, is achieved via gradient descent (ascent) of distance to the second closest obstacles in a second-order generalized Voronoi region, $\mathcal{F}_k|_{\mathcal{F}_{ij}}$. In other words, $\dot{c}(t) = -\pi_{T_{c(t)}\mathcal{F}_{ij}} \nabla d_k(c(t))$. The projected vector is

$$\pi_{T_x\mathcal{F}_{ij}} \nabla d_k(x) = \nabla d_k(x) - \frac{\langle \nabla d_i(x) - \nabla d_j(x), \nabla d_k(x) \rangle}{\langle \nabla d_i(x) - \nabla d_j(x), \nabla d_i(x) - \nabla d_j(x) \rangle} \nabla d_k(x).$$

Note that the correcting hyperplane is a line (see Fig. 9). When linking from the outer period to the inner cycle, the link can start from any point on the outer period, and likewise from the inner cycle to the outer period. All that needs to be determined is that the robot has traced a cycle or period. Detecting this can be computationally expensive, so in Section 8, we establish a computationally simple method that is sufficient for making links but comes at the cost of making redundant links.

For the next two linking procedures, the following lemma, whose proof appears in Choset and Burdick (2000), is useful.

LEMMA 5. If the objects $C_{i_1} \ldots, C_{i_k}$ intersect, then the associated k-equidistant surjective surface, $\mathcal{SS}_{i_1 \ldots i_k}$, is unbounded. In fact, if objects $C_{i_1}, \ldots, C_{i_k}$ intersect, then none of the gradients, $\nabla d_{i_1}(x), \ldots, \nabla d_{i_k}(x)$, is orthogonal to $T_x \mathcal{SS}_{i_1 \ldots i_k}$ for all $x \in \mathcal{SS}_{i_1 \ldots i_k}$. In other words, there are no extrema of D in the interior of $\mathcal{SS}_{i_1 \ldots i_k}$.

5.2. Inner Boundary Edge Period

The linking procedure to an inner boundary period is a two-step process: detection of the inner period and then the explicit construction of the link. By Lemma 5, this linking procedure amounts to following a path defined by gradient descent of D on the second-order generalized Voronoi region, which contains the boundary period. Linking from the inner boundary period is accomplished via gradient ascent of D, constrained to a two-equidistant face (Section 5).

To describe the detection scheme, we define the *raw distance function*, which provides the distance to all the points on the boundary of the environment that are *within line of sight* of the robot. For the following definition, recall that S^{m-1} is an $(m-1)$-dimensional sphere embedded in $\mathbb{R}^m$. Sometimes we treat $s \in S^{m-1}$ as a point on an $(m-1)$-dimensional sphere, and other times we treat it as a unit vector whose head is in the $(m-1)$-dimensional sphere.

DEFINITION 1. Raw Distance Function: The distance between a point, $x \in \mathbb{R}^m$, and a point on an object that is within line of sight of x, in a direction $s \in S^{m-1}$. This is the length of the line segment $x + \lambda s$ and where $\lambda = \min_{\Lambda \in [0,\infty)} D(x + \Lambda s) = 0$. That is,

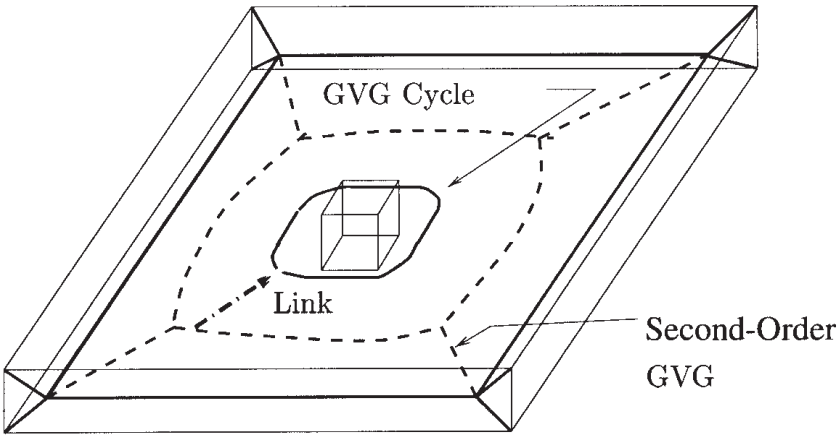

Fig. 9. Box in a room. Solid lines represent the GVG, and the dashed lines are the GVG^2. The thick dashed arrow represents the set of points equidistant to the floor and ceiling that decrease distance to the inner box, i.e., the path traced out by constrained gradient descent to the second closest obstacles.

$$\rho(x, s) = \|x + \lambda s\| \quad \text{where } \lambda = \min_{\Lambda \in [0,\infty)} D(x + \Lambda s) = 0, \tag{15}$$

where D is the multiobject distance function that measures distance to the nearest point on the nearest obstacle.

A key feature of the raw distance function (Fig. 10) is that it can be readily approximated by many realistic sensor configurations. The sensor measurement provides an approximate value of the distance function $\rho(x, s)$, and the direction to which the sensor is facing corresponds to the direction of measurement ($s \in S^{m-1}$). We term this function the "raw distance function" because raw sensor readings approximate this function. The raw distance function is a necessary component for the experimental implementation of the GVG.

To detect a boundary edge from an outer boundary component, we look at the values of $\rho(x, s)$ restricted to the normal plane (a hyperplane orthogonal to the tangent vector at x on a GVG edge). It can be seen from Figure 11 that for convex polyhedra, if there exists a local maxima of $\rho(x, s)$ with respect to s restricted to the normal plane, then there exists a point on a boundary edge. If for all points on the outer boundary component of a second-order generalized Voronoi region there exists a local maxima on each normal plane, then the outer boundary component surrounds an inner boundary period. However, we will demonstrate in Section 8 an alternative method to looking at maxima ρ, which is quicker but provides redundant links.

5.3. Occluding Period

The linking procedure to an occluding period is the same as it is for a boundary period. The detection scheme is similar; instead of looking for local maxima, the robot looks for discontinuities in the raw distance function restricted to a normal slice. If for all points on the outer boundary component of a second-order generalized Voronoi region there exists a discontinuity on each normal plane, then the outer boundary component surrounds an inner occluding period (see Fig. 7).

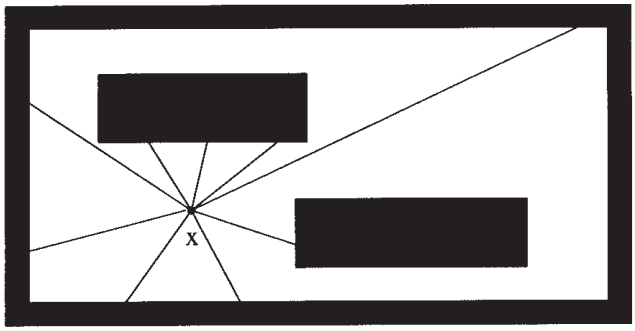

Fig. 10. The solid lines are values of the raw distance function, $\rho(x, s)$, for a fixed $x \in \mathbb{R}^2$ and nine samples of $s \in S$. The filled regions are obstacles.

5.4. Inner GVG^2 Period Link

While traversing an inner boundary component, $\partial_I \mathcal{F}_k|_{\mathcal{F}_{ij}}$, which contains GVG^2 equidistant edges and is disconnected from the outer boundary component, the robot builds a link outward from a meet point, $\mathcal{F}_{klp}|_{\mathcal{F}_{ij}}$, which is formed by the edges $\mathcal{F}_{kl}|_{\mathcal{F}_{ij}}$, $\mathcal{F}_{kp}|_{\mathcal{F}_{ij}}$, and $\mathcal{F}_{pl}|_{\mathcal{F}_{ij}}$. The link is the intersection of $\mathcal{SS}_{pl}|_{\mathcal{F}_{ij}} \backslash \mathcal{F}_{kl}|_{\mathcal{F}_{ij}}$. That is, instead of tracing the GVG^2 equidistant edge, the robot traces the points, x, where $d_l(x) = d_p(x) > d_k(x) > d_i(x) = d_j(x)$. This link brings the robot to an outer boundary component (Choset and Burdick 2000).

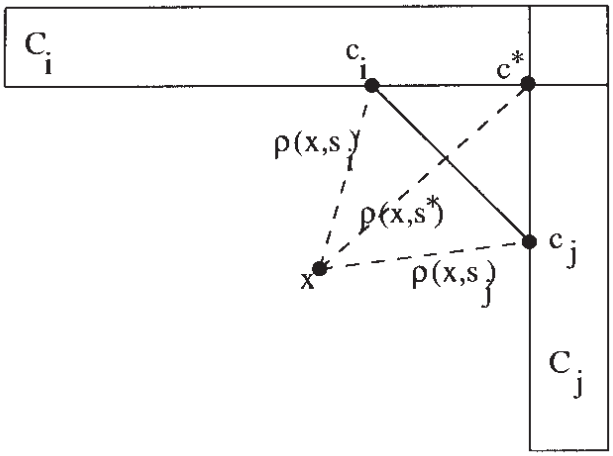

Fig. 11. A local maxima of $\rho(x, s)$ with respect to s corresponds to a boundary point. The dotted lines delineate two values of the raw distance function on opposite sides of a local maxima. The solid line connects two points from two different convex sets, C_i and C_j.

Unfortunately, the robot may not know initially if it is on an inner boundary component. Therefore, at all second-order meet points, the robot must perform this procedure, which results in redundant links. The robot terminates the link-tracing procedure when it encounters a GVG edge, a GVG^2 edge, a boundary edge, or a floating boundary edge. The terminating point becomes a node in the HGVG.

Similarly, if the robot is on an outer boundary component, it must look for an additional pair of equidistant obstacles. However, the robot need not know if it is on an outer boundary component, so it must always perform the following linking strategy, once again resulting in redundant links. On a GVG edge, $\mathcal{F}_{ijk}$, the robot starts constructing a link when it encounters a point where two additional obstacles, C_l and C_p, are equidistant with the following distance relationship: $d_l(x) = d_p(x) = d_k(x) > d_i(x) = d_j(x)$. The robot terminates the link-tracing process when it encounters a structure in the HGVG. If this structure is a second-order meet point, the link is formed. Otherwise, the robot may backtrack to continue tracing the outer boundary component or save the link as another redundant structure.

A similar procedure is followed when a GVG^2 equidistant edge is on the outer boundary component. While tracing $\mathcal{F}_{kl}|_{\mathcal{F}_{ij}}$, the robot's range sensor must look for equidistant between two obstacles, C_p and C_q. At this point, the robot traces a path where $d_p(x) = d_q(x) > d_k(x) > d_i(x) = d_j(x)$ on $\mathcal{F}_{ij}$ until it encounters an HGVG structure. Just like before, if this structure is a second-order meet point, the link is formed. Otherwise, the robot may backtrack to continue tracing the outer boundary component or save the link as another redundant structure. A similar procedure exists for boundary edges and floating boundary edges.

6. Incremental Accessibility

Incremental accessibility is the ability to access some point on the GVG via a collision-free path from any point in the free space, using only *local information*. It is obtained by gradient ascent of the multiobject distance function, D (Choset and Burdick 2000). Recall from Choset (1996, 1998), using nonsmooth analysis it can be shown that the *generalized gradient* of $D(x)$ is

$$\partial D(x) = \text{Co}\{\nabla d_i(x) \ : \ i \in I(x)\}, \tag{16}$$

where Co is the convex hull operation, and $I(x)$ is the set of indices such that $\forall i \ \in I(x)$, each C_i is the closest object to x (so there can be more than one "closest" object). Since $\partial D(x)$ is composed of single object distance gradients, it can be readily *computed from sensor data*.

In the planar case, gradient ascent of D is simply moving away from the nearest obstacle until the robot is two-way equidistant. In $\mathbb{R}^3$, the robot initially moves away from its nearest obstacle until it achieves two-way equidistance, and then while maintaining two-way equidistance, the robot performs gradient ascent until it reaches a point that is equidistant to three obstacles, a point on the GVG in $\mathbb{R}^3$. In $\mathbb{R}^m$, one can assume that gradient ascent of D reduces to a sequence of gradient ascent operations, constrained to equidistant faces where the robot travels via a collision-free path along a two-equidistant face, then a three-equidistant, and eventually to an m-equidistant face (Choset and Burdick 2000).[2]

EXAMPLE 3. Figure 12 is a cross section of a three-dimensional world (imagine the polygons are coming out of the page), which contains two examples of accessibility in three dimensions. Starting from (A), the robot follows gradient ascent of d_j until it reaches $\mathcal{F}_{jk}$. From there, it does gradient ascent of $D = d_j = d_k$ constrained to $\mathcal{F}_{jk}$ until it reaches $\mathcal{F}_{ijk}$, an edge of the GVG.

The procedure to trace a path on a k-equidistant face, using constrained gradient ascent of the multiobject distance function, borrows some basic ideas and techniques from numerical continuation methods (Keller 1987), in a fashion similar to the approach described in Section 2. Here, the roots of the expression $G_A(y, \lambda) = 0$ as the "parameter" λ is varied describe

2. In actuality, the description of gradient ascent of D, cascading through a sequence of increasing equidistant sheets, is not entirely correct in dimensions greater than three. The above procedure represents the tail end of a sequence of gradient ascent operations, each constrained to an equidistant face. It is possible that gradient ascent of D describes a path of the robot that traverses a two-equidistant face, then a three-equidistant face, then *another* two-equidistant face, then a three-equidistant face, and so on. That is, in the course of doing gradient ascent of D, the robot may drop down to double equidistance before undergoing the cascading sequence of constrained gradient ascent operations that bring the robot to a GVG edge. We have constructed generic four-dimensional examples where this occurs.

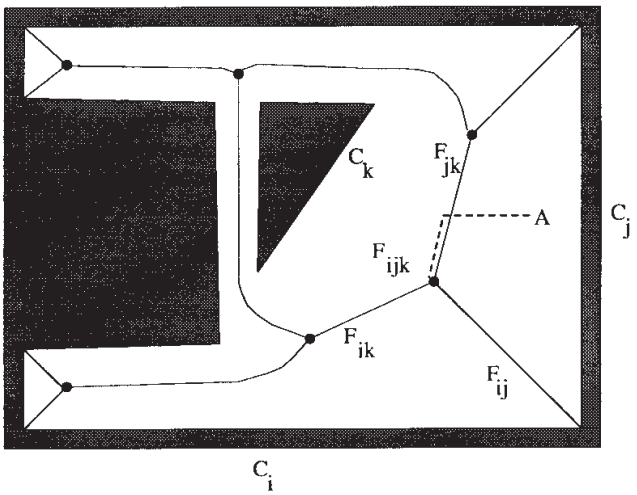

Fig. 12. Gradient ascent accessibility in $\mathbb{R}^3$.

a path on a k-equidistant face. Note, in this case λ is a vector, i.e., it is not a scalar as it was in the incremental tracability procedure.

Let x be a point on the k-equidistant face. Choose a local coordinate frame at x so that the first $m-k+1$ coordinates, $(z_1, \ldots, z_{m-k+1})$, are the coordinates that span the tangent space of the k-equidistant face at x, and the next $k-1$ coordinates, $(z_{m-k+2}, \ldots, z_m)$, span a plane termed the "normal slice plane." We can thus decompose the local coordinates into $z = (y, \lambda)$, where $\lambda = (z_1, \ldots, z_{m-k+1})$, the "sweep coordinates," and $y = (z_{m-k+2}, \ldots, z_m)$ are the "slice" coordinates. Note that there can be some confusion with this choice of coordinates: when $z = (y, \lambda)$, λ is the *first* $m-k+1$ coordinates and y is the *next* $k-1$ coordinates.

Now define the function $G_A : \mathbb{R}^{k-1} \times \mathbb{R}^{m-k+1} \rightarrow \mathbb{R}^{k-1}$ as follows:

$$G_A(y, \lambda) = \begin{bmatrix} (d_1 - d_2)(y, \lambda) \\ (d_1 - d_3)(y, \lambda) \\ \vdots \\ (d_1 - d_k)(y, \lambda) \end{bmatrix}. \tag{17}$$

The procedure for tracing a path on the k-equidistant face is similar to the approach described in Section 2. The robot starts at a point on the k-equidistant face. At this point, and all others on the k-equidistant face, G_A vanishes. The robot takes a "small" step, $\Delta\lambda$, in the tangent space of the k-equidistant face such that $D(x)$ increases the most. Typically, this step takes the robot off of the k-equidistant face. So, on a $(k-1)$-dimensional plane orthogonal to the tangent space, the robot moves back onto the k-equidistant face. This $(k-1)$-dimensional plane is called the "correcting plane." The correction step is the same as the one described in Section 2. If y^h and λ^h are the h^{th} estimates of y and λ, the $h+1$st iteration is defined as

$$y^{h+1} = y^h - \left(\nabla_y G_A\right)^{-1} G_A(y^h, \lambda^h), \tag{18}$$

where $\nabla_y G_A$ is evaluated at (y^h, λ^h). After taking the prediction step, the goal of the correction step is to find where the k-equidistant face locally intersects the correcting plane.

Again, it is important to note that to evaluate $G_A(y, \lambda)$ and $\nabla_y G_A(y, \lambda)$, one only needs to know the distance and direction to the k objects that are closest to the robot's current location—information that is easily obtained from local distance sensor data. The following propositions and lemmas demonstrate that this procedure is theoretically sound and can be implemented using local information.

Computing the Tangent Vector. The predictor step is a small step the direction in the tangent space of the k-equidistant face that maximally increases $D(x)$. This step is determined in two steps: first the tangent space of the k-equidistant face at x is computed, and then the generalized gradient of D is projected onto it. Lemmas 1 and 2 furnish the tangent space, and the following proposition shows how the generalized gradient is projected onto it. In fact, the following proposition states that the generalized gradient of D projects to a single vector on the tangent space.

PROPOSITION 5. The restriction of the multiobject distance function D to a k-equidistant face is smooth. That is, the generalized gradient of $D(x)$ projected onto $T_x\mathcal{F}_{i_1 \ldots i_k}$ is equal to $\pi_{T_x\mathcal{F}_{i_1 \ldots i_k}} \nabla d_i$ for all $i \in \{i_1 \ldots i_k\}$, where π is the orthogonal projection operator.

Let E be a plane in $T_x\mathbb{R}^m$. Let v_e be the unique minimum length vector in E (i.e., v_e is based at the origin of $T_x\mathbb{R}^m$ and its head is in E). Define P_E to be the subspace of $T_x\mathbb{R}^m$ parallel to E, i.e., $P_E = E - v_e$. Let $P_E^{\perp}$ be the orthogonal compliment of P_E. Therefore, $T_x\mathbb{R}^m = P_E \bigoplus P_E^{\perp}$, and thus for all vectors $u \in T_x\mathbb{R}^m$, u can be written as the sum $u_1 + u_2$, where $u_1 \in P_E$ and $u_2 \in P_E^{\perp}$. The orthogonal projection $\pi_{P_E}(u)$ is u_1. We can now define the orthogonal projection operator π_E to be π_{P_E}.

Proof. Note that $\partial D(x)$ is the affine hull of the heads of the k gradient vectors $\nabla d_{i_1}, \ldots, \nabla d_{i_k}$. So, $\partial D(x)$ can be viewed as a plane in $T_x\mathbb{R}^m$ and by Lemmas 1 and 2, the plane $\partial D(x)$ is orthogonal to $T_x\mathcal{SS}_{i_1 \ldots i_k}$. Transversality considerations imply that $\partial D(x)$ and $T_x\mathcal{SS}_{i_1 \ldots i_k}$ intersect at a point, and thus the generalized gradient of D constrained to $T_x\mathcal{SS}_{i_1 \ldots i_k}$ is always a point, not a vector. This point, which we denote by $v \in T_x\mathcal{SS}_{i_1 \ldots i_k} \bigcap \partial D(x)$ is the closest point in $\partial D(x)$ to $0 \in T_x\mathbb{R}^m$.

Define P to be a subspace of $T_x\mathbb{R}^m$ given by $P = \partial D(x) - v$ (again, $\partial D(x)$ is viewed as a plane). The orthogonal projection of $u \in \partial D(x)$ is given by

$$\pi_{T_x\mathcal{SS}_{i_1 \ldots i_k}} : \partial D(x) \rightarrow T_x\mathcal{SS}_{i_1 \ldots i_k}. \tag{19}$$

Since D and $\pi_{T_x \mathcal{SS}_{i_1 \ldots i_k}}$ are continuous, the restriction of the generalized gradient of D on $\mathcal{SS}_{i_1 \ldots i_k}$ is continuous. Therefore, the restriction of the multiobject distance function D to a k-equidistant face is smooth. □

Therefore, the robot takes the following step:

$$\begin{aligned} \pi_{\mathcal{F}_{i_1 \ldots i_k}} \partial D(x) &= \pi_{\mathcal{F}_{i_1, \ldots i_k}} \nabla d_{i_1}(x) \\ &= \nabla d_{i_1}(x) \\ &\quad - \sum_{j=2}^{j=k} \frac{\nabla d_{i_1}(x) - \nabla d_{i_j}(x)}{\| \nabla d_{i_1}(x) - \nabla d_{i_j}(x) \|} \| \nabla d_{i_1}(x) \|. \end{aligned}$$

Computing the Correction Step. The correction procedure is guaranteed by

PROPOSITION 6. The matrix $\nabla_y G_A(y, \lambda)$ has full rank (i.e., has rank $(k-1)$) in a neighborhood of a k-equidistant face on the correcting plane.

Proof. This is a simple consequence of Lemmas 3 and 4. Since $\nabla_y G_A$ is a $k-1$ by $k-1$ matrix, by these lemmas, it must have rank $(k-1)$ for $x \in \mathcal{F}^k$, and therefore be invertible at x. Since the rank operation is a continuous function, $\nabla_y G_A$ must be invertible in an open neighborhood around $x = (y, \lambda) \in \mathcal{F}^m$. This open neighborhood will intersect the correcting plane for $\| \Delta\lambda \|$ sufficiently small, and thus $\nabla_y G_A$ is invertible on the correcting plane as well. □

7. Incremental Departability

In sensor-based exploration, the robot may or may not know the coordinates of its goal location. If the robot does not know the goal coordinates, it is assumed that the goal is defined by a beacon or other feature that the robot can detect once it is within line of sight of it. We therefore would like to find a departing method in which the robot can access the goal in a straight line. Treating the goal as an object, create a "virtual" generalized Voronoi graph (Fig. 14). A star-shaped set, bounded by the virtual GVG, surrounds the goal, and thus there exists a straight-line path between any point on the boundary of this virtual star-shaped set and the goal. Generally, the virtual GVG is connected to the GVG and thus there is a point within line of sight of the goal on the GVG. However, as we know from previous sections, the virtual GVG may be disconnected. In this case, it is necessary to build a link to the disconnected component that surrounds the goal. The linking strategy is a special case of the strategy one would use to link GVG cycles to other second-order GVG edges.

8. Simulations

8.1. Planar Simulations

A planar simulator has validated this approach for a point or circularly symmetric robot operating in the plane. Figure 15 contains an example of a bounded environment in which our algorithm was tested. In Figure 16, the robot has accessed the GVG, traced one GVG edge, encountered a meet point, and continued tracing until a boundary point. The ticked solid lines represent the planar GVG (also the GVD); these are the locus of points equidistant to the two nearest obstacles. The ticks point to the nearest obstacles. Figures 17 and 18 display two more intermediate simulation results. Figure 19 shows the final simulation result.

Figures 20 and 21 contain other examples of planar GVGs.

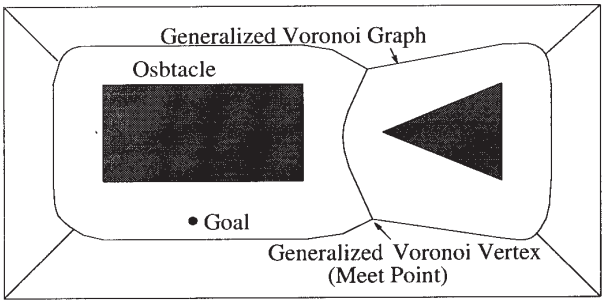

Fig. 13. Original GVG.

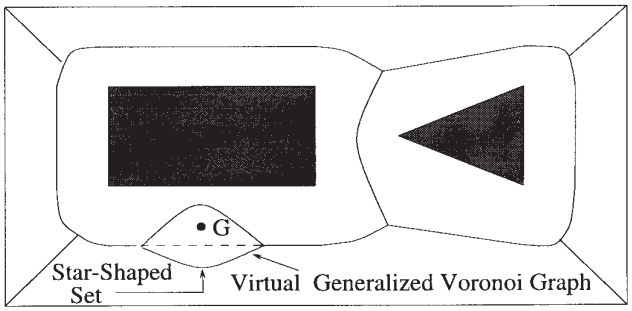

Fig. 14. Virtual GVG of Figure 13.

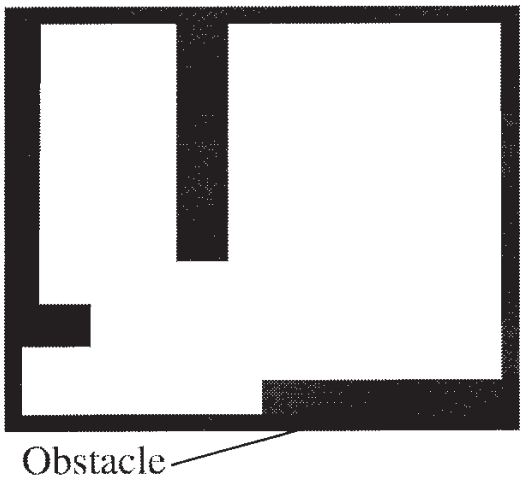

Fig. 15. Floor plan of bounded environment.

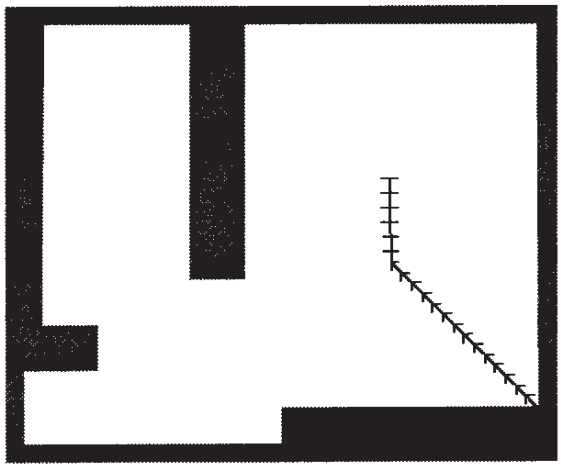

Fig. 16. Iteration 1.

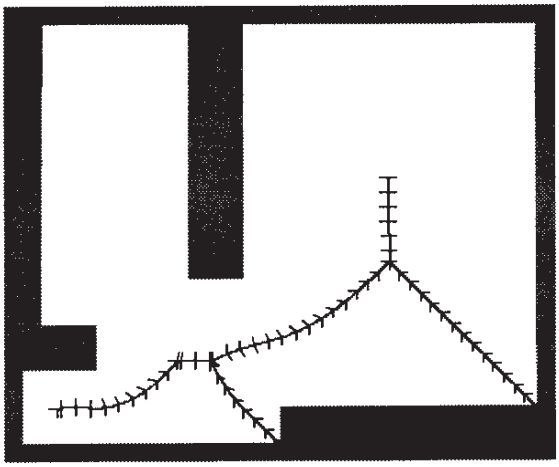

Fig. 17. Iteration 5.

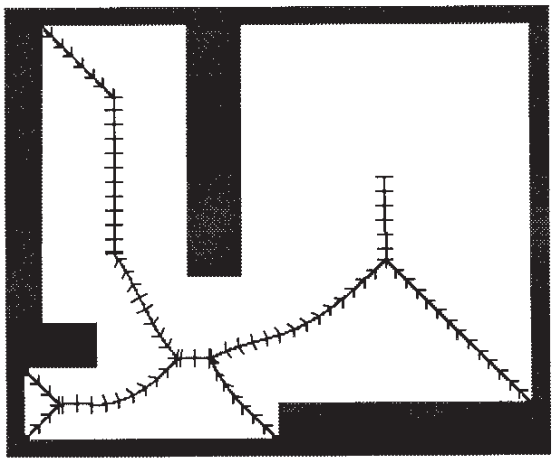

Fig. 18. Iteration 10.

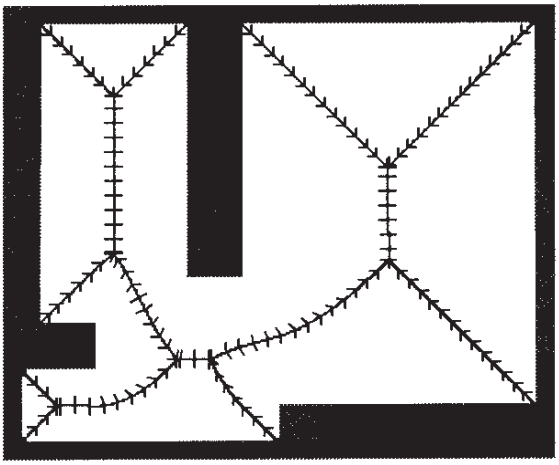

Fig. 19. Iteration 14.

8.2. Three-Dimensional Simulator

A major advantage that the HGVG has over other methods is that it is applicable in higher dimensional workspaces. To this end, we have implemented a three-dimensional simulator that traces GVG edges. The algorithm and data structure of the three-dimensional simulator is similar to that of the planar version. The distance function code, used in the three-dimensional simulator, was written by Brian Mirtich at Berkeley. Currently, the linking procedures are under development. Figure 22 contains a GVG for a three-dimensional environment, and Figure 23 contains the GVG and GVG^2 equidistant edges for the same environment.

Figure 24 contains the GVG for a rectangular enclosure with two boxes in its interior. In this example, the GVG is not connected and thus the robot cannot use the GVG to plan paths in this environment. Essentially, the robot-highway system has a big gap in it. However, our solution recursively defines GVG edges on the two-dimensional sheets; these edges are second-order GVG edges and together with the GVG form the HGVG, which is connected in Figure 25.

Figures 26 and 27 display only a box that is in the middle of a rectangular enclosure. In these figures, the box has a hole on its top, which can be a through-hole or entrance to an environment in the box's interior. Regardless, the hole has a GVG structure associated with it and there is a GVG cycle surrounding the box. Emanating from the cycle is a link that terminates at an occluding edge. In these figures, the second-order GVG, comprising occluding edges and GVG^2 equidistant edges, links the GVG connected components. Again, the HGVG is connected in this example.

Figure 28 contains an HGVG that is connected through links that were formed via constrained gradient descent to the second closest obstacle, as described in Section 5.1. Although one link is sufficient to connect the HGVG, the HGVG here has many redundant links. These redundant links are reason-

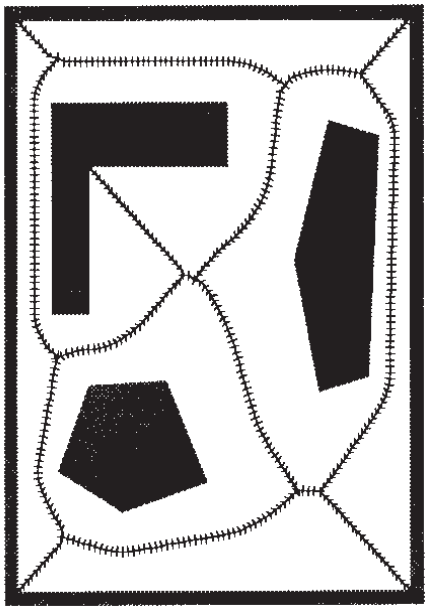

Fig. 20. Planar GVG.

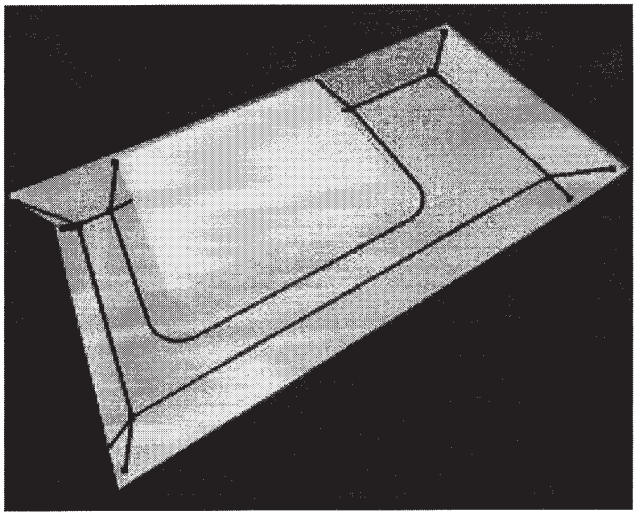

Fig. 22. Results of applying the simulator to a three-dimensional box with a long box that is located off-center in the interior. Note that some of the walls were removed so the GVG lines, depicted as thick solid lines, can be displayed.

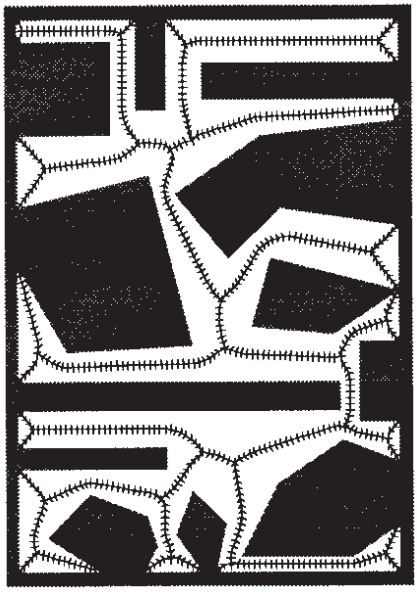

Fig. 21. Planar GVG.

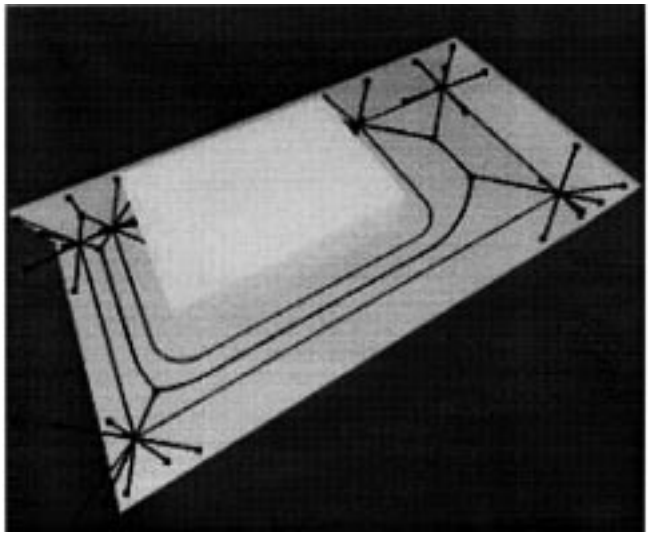

Fig. 23. HGVG for the same environment in Fig. 23.

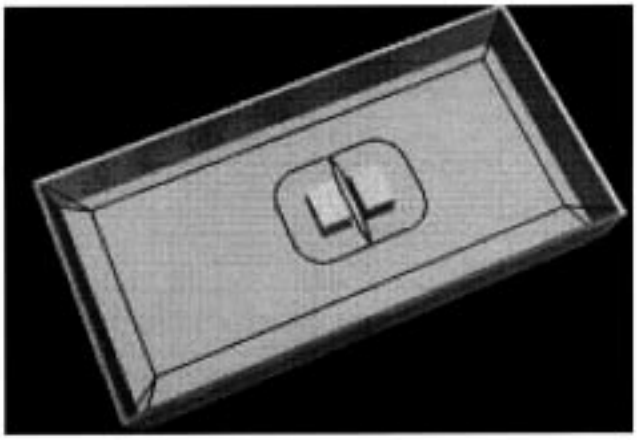

Fig. 24. The GVG is disconnected.

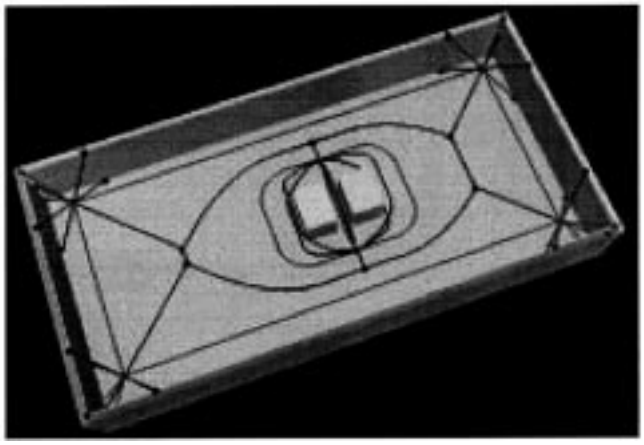

Fig. 25. The HGVG is connected.

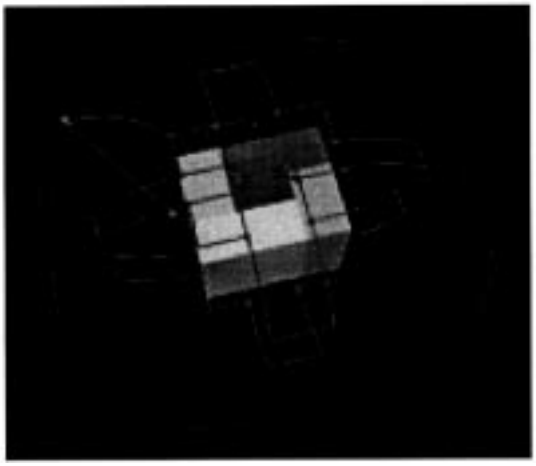

Fig. 26. The GVG is disconnected

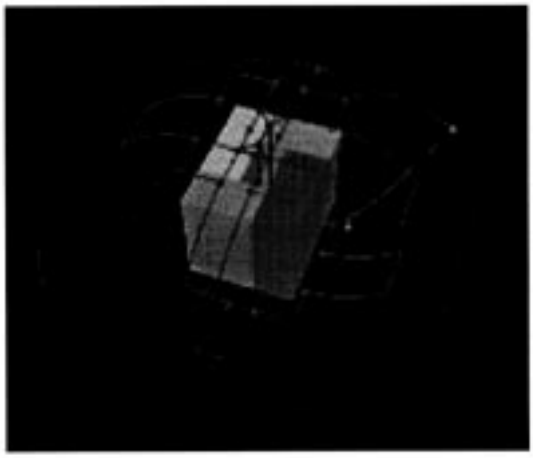

Fig. 27. The HGVG is connected.

able because they provide parallel pathways that may serve as shortcuts that direct the robot from the exterior to the interior of the environment. However, they were formed for a different reason. In the course of developing the simulator, searching for every possible GVG and GVG^2 cycle became computationally expensive. Instead, it proved to be quicker to bypass searching for cycles and just invoke the linking procedure at every node in the HGVG. Figure 28 also highlights these nodes. Note that additional nodes, termed critical points, are also used. These are points where D obtains an extremal value on the GVG edge; two nodes represent each degenerate extrema.

The main advantage of storing the nodes comes into play when multiple links terminate on the same GVG edge. In Figure 28, four links terminate on the GVG cycle in the interior of the room. In this example, the front-left second-order meet point was the origin of the first link to the cycle. Once the link was formed, the planner generated the inner GVG cycle. The revised algorithm requires linking from all the nodes, so the front-right second-order meet point also sources a link. When this link terminates on the GVG cycle, how does the planner know the cycle was already traced out? A naive method would be to search every point on every edge that has been generated, but that would prove to be computationally intensive. Instead, the planner starts to retrace the GVG cycle until it encounters a node. At this point, the planner can look up the coordinates of this point in a list of all nodes. Since all of the nodes for a set of measure zero on the HGVG, this does not constitute a major search. If a duplicate node was found, the link is inserted into the appropriate edge that was already generated.

9. Experiments

To verify the incremental construction procedure, we implemented this approach in the planar case (i.e., $m = 2$) on a circular mobile robot base. The mobile robot is the B12

Fig. 28. Two views of the HGVG with links connecting the outer GVG to the inner one.

Mobile Robot Base, produced by Real World Interface, Inc., and it is instrumented with a ring of 12 sonar sensors that provide local distance measurement information. While the sensors are quite accurate in distance measurement (on the order of 1 cm), their angular resolution is only accurate to 22°. In terms of our algorithm, $d_i(x)$ can be accurately measured using this robot, but $\nabla d_i(x)$ will be inaccurate.

The result of one experiment is shown in Figures 29 and 30, though many other experiments were successfully completed. In this trial, the room was "T-shaped," with the geometry of the room and the theoretical GVG shown in Figure 29. In Figure 30 the experimental GVG constructed by the robot is shown. The small squares denote the edge termination points, while the hatched squares represent meet points. For safety reasons, the robot does not trace the edge all the way to the wall's boundary. The octagon shown on the graph represents the scale size of the robot. The experimental GVG edges are jagged because the tangent is crudely approximated. This crude approximation results from the angular inaccuracy of the sonar distance sensors. However, the GVG is connected, and the edges are maximally far away from the workspace boundary. Note that the actual GVG construction is quite robust even with large errors in distance measurements.

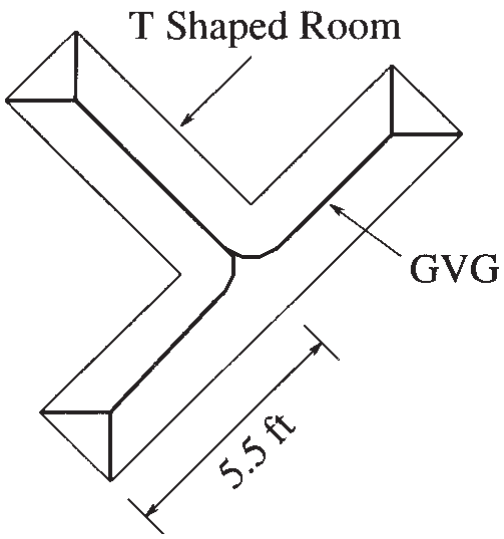

Fig. 29. Room with Actual GVG.

10. Conclusion

This paper introduced an incremental procedure to construct the GVG and the HGVG. This procedure requires only local sensor distance measurement data, and is therefore practically implementable, as demonstrated by our simulations and experiments. Hence, the generalized Voronoi graph and hierarchical generalized Voronoi graph introduced in this work appear to be useful means for implementing sensor-based motion-planning algorithms. We have shown in related work the numerical methods introduced are useful for "sensorizing" other (e.g., the OPP method) robot motion planners.

In addition to tracing the roots of a continuous function, we also developed a procedure to trace a one-dimensional set of points in the domain of a discontinuous function. This procedure has been implemented in simulation and in the future will be generalized to all discontinuous functions.

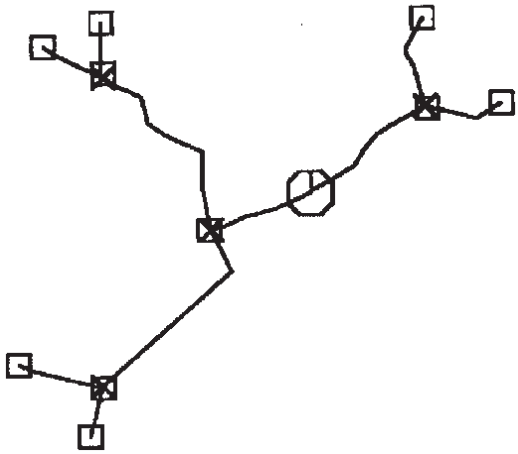

Fig. 30. Experimental GVG.

A critical component to sensor-based exploration is the robot's ability to ascertain its location in the partially explored map or to determine that it has entered new territory. Many conventional methods attempt to make this determination via a localization scheme that updates the (x, y) coordinates of the robot. Most robots update their location by integrating data from their wheel encoders that count the number of wheel rotations (or fractional rotations). If the robot slips, the wheels do not rotate and thus this motion cannot be integrated by the robot's encoder, thereby causing error. GPS systems may offer an alternative, but commercially available systems do not work inside buildings nor provide the necessary resolution. Finally, landmarks with known locations can be deployed in the environment, but the task described in this paper considers environments that are completely unknown a priori. Future work will exploit geometries of the HGVG to locate itself on the partially explored map or conclude the robot has entered new territory.

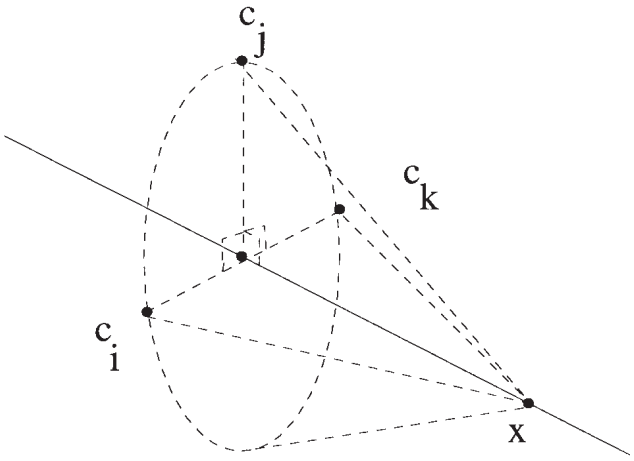

Fig. 31. Cone formed by points.

Appendix

Proof of Lemma 1

Proof. It can be shown that given Assumption 2, the *regular* k-equidistant faces (k-equidistant faces of the RVG) are C^2-diffeomorphic to $\mathbb{R}^{m-k+1}$ and thus regular k-equidistant faces are isometric to their tangent spaces. In other words, $T_x\mathcal{R}_{i_1\ldots i_k} \simeq \mathcal{R}_{i_1\ldots i_k}$. Assumption 2 guarantees that the affine hull of $\{c_1, \ldots, c_k\}$, $\mathrm{Af}\{c_1, \ldots, c_k\}$, is a $(k-1)$-dimensional plane. Furthermore, Assumption 2 guarantees that embedded in the base plane, there is a unique $(k-2)$-dimensional sphere, S, defined by $\{c_1, \ldots, c_k\}$. Define a coordinate frame whose origin is the center of this sphere. Let 0 denote the origin.

Pick a $w \in T_x\mathcal{R}_{i_1\ldots i_k}$ and translate this vector so it is based at 0, the origin of the above described coordinate system. Let $\bar{w}$ be the translated vector. Since $T_x\mathcal{R}_{i_1\ldots i_k} \simeq \mathcal{R}_{i_1\ldots i_k}$, $\bar{w}$ can be viewed as the difference of two points, $\bar{x} - 0$, where $\bar{x}$ is equidistant to $\{c_1, \ldots, c_k\}$. (This is the same thing as saying there is a natural identification between a Euclidean space and the tangent space of a point of a Euclidean space.)

Drop a perpendicular from $\bar{x}$ to $\mathrm{Af}\{c_1, \ldots, c_k\}$. Let p be the point where the perpendicular intersects the affine hull. Since $\|\bar{x}-c_i\| = \|\bar{x}-c_j\|$ for all i and j, $\|p-c_i\| = \|p-c_j\|$ for all i and j. Therefore, p is an element of $\mathcal{R}_{i_1\ldots i_k}$.

The next step is to show that p is 0, the origin of the coordinate system (the center of the sphere S). Transversality guarantees that

$$\dim(T_x\mathcal{R}_{i_1\ldots i_k} \bigcap \mathrm{Af}\{c_1, \ldots, c_k\}) = 0.$$

Furthermore, since the intersection of two convex sets is a convex set, $T_x\mathcal{R}_{i_1\ldots i_k} \bigcap \mathrm{Af}\{c_1, \ldots, c_k\}$ is also a convex set, which is connected. Since this intersection is connected and is zero dimensional, there can only be one point in $T_x\mathcal{R}_{i_1\ldots i_k} \bigcap \mathrm{Af}\{c_1, \ldots, c_k\}$. Therefore, $p = 0$.

Therefore, $\bar{w}$, which is equal to $\bar{x} - 0$, is orthogonal to all vectors in the affine hull of $\{c_1, \ldots, c_k\}$. Since w is a translate of $\bar{w}$, w is also perpendicular to $\mathrm{Af}\{c_1, \ldots, c_k\}$.

Now, let *gradient plane* be the $(k-1)$-dimensional plane that contains the heads of the k gradient vectors based at x. This is the affine hull of the heads of k gradient vectors. Note that $d_{i_j}(x)$ is the length of the line that connects x and c_j where $c_j \in \{c_1, \ldots, c_k\}$. Since, $\|x - c_{i_1}\| = \|x - c_{i_j}\|$ for all j, $x - c_{i_j} = \|x - c_{i_1}\|\nabla d_{i_j}(x)$. Therefore, the cone formed by the gradient vectors is "similar" to the one formed by the k closest points and thus the *gradient plane* is parallel to $\mathrm{Af}\{c_1, \ldots, c_k\}$. Therefore, the tangent space is perpendicular to the gradient plane. □

Proof of Lemma 2

Proof. A k-equidistant face can be defined by $G^{-1}(0)$, where

$$G(x) = \begin{bmatrix} (d_1 - d_2)(x) \\ \vdots \\ (d_1 - d_k)(x) \end{bmatrix}.$$

Let $\{c_i\}$ denote the closest points to x in the k closest obstacles. The *regular* k-equidistant face for the set of points $\{c_i\}$ is defined by $VG^{-1}(0)$, where

$$VG(x) = \begin{bmatrix} |x - c_1| - |x - c_2| \\ |x - c_1| - |x - c_3| \\ \vdots \\ |x - c_1| - |x - c_k| \end{bmatrix}.$$

Since the set of closest points, $\{c_i\}$, is the same for the k-equidistant face and the regular k-equidistant face at x,

$d_h(x) = |x - c_h|$ for all $h = 1, \ldots, k$. That is, $\nabla(d_{h_1} - d_{h_2})(x) = |x - c_{h_1}| - |x - c_{h_2}|$ for all $h_1, h_2 = 1, \ldots, k$. Therefore, $G(x) = VG(x)$ at x.

By the preimage theorem, the tangent space at a point $x \in G^{-1}(0)$ is the null space of $\nabla G(x)$, where

$$\nabla G(x) = \begin{bmatrix} \nabla(d_1 - d_2)(x) \\ \vdots \\ \nabla(d_1 - d_k)(x) \end{bmatrix}.$$

And the tangent space at a point $x \in VG^{-1}(0)$ is the null space of $\nabla VG(x)$, where

$$\nabla VG(x) = \begin{bmatrix} \frac{x-c_1}{|x-c_1|} - \frac{x-c_2}{|x-c_1|} \\ \frac{x-c_1}{|x-c_1|} - \frac{x-c_3}{|x-c_1|} \\ \vdots \\ \frac{x-c_1}{|x-c_1|} - \frac{x-c_k}{|x-c_1|} \end{bmatrix}.$$

Again, since the set of closest points, $\{c_i\}$, is the same for the k-equidistant face and the regular k-equidistant face at x, $\nabla d_h(x) = \frac{x-c_h}{|x-c_h|}$ for all $h = 1, \ldots, k$. That is, $\nabla d_{h_1}(x) - \nabla d_{h_2}(x) = \frac{x-c_{h_1}}{|x-c_{h_1}|} - \frac{x-c_{h_2}}{|x-c_{h_2}|}$ for all $h_1, h_2 = 1, \ldots, k$. Therefore, $\nabla G(x) = \nabla VG(x)$ at x. Hence, the k-equidistant face and the regular k-equidistant face have the same tangent space at x. □

Proof of Lemma 3

Proof. First, consider the case when $q = 2$. In this case, the robot is either equidistant to three obstacles (e.g., $i_1 = 1$, $j_1 = 2$, $i_1 = 1$, and $j_2 = 3$) or two sets of two obstacles (e.g., $i_1 = 1$, $j_1 = 2$, $i_2 = 3$, and $j_2 = 4$). The respective tangent spaces of $\mathcal{SS}_{i_1 j_1}$ and $\mathcal{SS}_{i_2 j_2}$ are

$$T_x\mathcal{SS}_{i_1 j_1} = \{v \in T_x\mathbb{R}^m : \langle \nabla(d_{i_1} - d_{j_1})(x), v\rangle = 0\},$$
$$T_x\mathcal{SS}_{i_2 j_2} = \{v \in T_x\mathbb{R}^m : \langle \nabla(d_{i_2} - d_{j_2})(x), v\rangle = 0\}.$$

By the Equidistant Surface Transversality Assumption from Choset and Burdick (2000), we know that $\mathcal{SS}_{i_1 j_1} \pitchfork -\mathcal{SS}_{i_2 j_2}$. Assume at some point x, $\nabla(d_{i_1} - d_{j_1})(x) = \kappa\nabla(d_{i_2} - d_{j_2})(x)$. By definition, for all $w \in T_x\mathcal{SS}_{ij}$, $\langle \nabla(d_{i_1} - d_{j_1})(x), w\rangle = 0$. Since $\nabla(d_{i_1} - d_{j_1})(x) = \kappa\nabla(d_{i_2} - d_{j_2})(x)$, for $w \in T_x\mathcal{SS}_{ij}$, $\langle \nabla(d_{i_2} - d_{j_2})(x), w\rangle = 0$. This implies that $T_x\mathcal{SS}_{i_1 j_1} = T_x\mathcal{SS}_{i_2 j_2}$, which violates the Equidistant Surface Transversality Assumption (Assumption 2). Therefore, $\nabla(d_{i_1} - d_{j_1})(x) \neq \kappa\nabla(d_{i_2} - d_{j_2})(x)$; i.e., they are linearly independent. It therefore follows that

$$\operatorname{rank}\begin{bmatrix} (\nabla(d_{i_1} - d_{j_1})(x))^T \\ (\nabla(d_{i_2} - d_{j_2})(x))^T \end{bmatrix} = 2.$$

Now, we consider the case where $q = 3$. Here, the robot may be equidistant to four obstacles, three sets of two obstacles, or three obstacles and an additional pair of obstacles. Consider the matrix,

$$\nabla G(x) = \begin{bmatrix} (\nabla(d_{i_1} - d_{j_1})(x))^T \\ (\nabla(d_{i_2} - d_{j_2})(x))^T \\ (\nabla(d_{i_3} - d_{j_3})(x))^T \end{bmatrix}.$$

The Equidistant Surface Transversality Assumption (Assumption 2) guarantees each row is pairwise linearly independent:

$$\begin{aligned} \nabla d_{i_1} - \nabla d_{j_1} &\neq \kappa_{12}\big(\nabla d_{i_2} - \nabla d_{j_2}\big), \\ \nabla d_{i_1} - \nabla d_{j_1} &\neq \kappa_{13}\big(\nabla d_{i_3} - \nabla d_{j_3}\big), \\ \nabla d_{i_2} - \nabla d_{j_2} &\neq \kappa_{23}\big(\nabla d_{i_3} - \nabla d_{j_3}\big). \end{aligned} \tag{20}$$

It remains to show that no one row is a linear combination of the other two. Again, we prove this by contradiction. Assume $\nabla(d_{i_1} - d_{j_1}) = \alpha(\nabla(d_{i_2} - d_{j_2})) + \beta(\nabla(d_{i_3} - d_{j_3}))$. By definition, for all $w \in T_x\mathcal{SS}_{i_1 j_1}$, $\langle \nabla(d_{i_1} - d_{j_1})(x), w\rangle = 0$. Thus,

$$\begin{aligned} &\nabla(d_{i_1} - d_{j_1}) = \alpha(\nabla(d_{i_2} - d_{j_2})) + \beta(\nabla(d_{i_3} - d_{j_3})), \\ &\Rightarrow \big\langle \big(\alpha\big(\nabla(d_{i_2} - d_{j_2})\big) + \beta\big(\nabla(d_{i_3} - d_{j_3})\big)\big), w\big\rangle = 0, \\ &\Rightarrow \big\langle \big(\big(\nabla(d_{i_2} - d_{j_2})\big) + \tfrac{\beta}{\alpha}\big(\nabla(d_{i_3} - d_{j_3})\big)\big), w\big\rangle = 0. \end{aligned}$$

Since by Equidistant Surface Transversality Assumption (Choset and Burdick 1995), for all $w \in T_x S_{i_1 j_1}$:

$$\begin{aligned} \big\langle \big(\nabla(d_{i_2} - d_{j_2})(x)\big), w\big\rangle &\neq 0 \\ \big\langle \big(\nabla(d_{i_3} - d_{j_3})(x)\big), w\big\rangle &\neq 0, \end{aligned}$$

we conclude that $\nabla(d_{i_2} - d_{j_2}) = \frac{\beta}{\alpha}\big(\nabla(d_{i_3} - d_{j_3})\big)$. However, this contradicts one of the three inequalities in eq. 20. Therefore, all the rows of ∇G are linearly independent of each other and $\operatorname{rank}(\nabla G) = 3$.

The lemma follows by induction. Assume the matrix

$$\tilde{G}(x) = \begin{bmatrix} (\nabla(d_{i_1} - d_{j_1})(x))^T \\ \vdots \\ (\nabla(d_{i_{q-1}} - d_{j_{q-1}})(x))^T \end{bmatrix}$$

has a rank of $q - 1$, and let $\mathcal{SS}_{i_q j_q}$ be a two-equidistant surjective surface defined by obstacles C_{i_q} and C_{j_q}. The remainder of this proof follows by contradiction. Assume that $\nabla(d_{i_q} - d_{j_q}) = \sum_{r=1}^{q-1} \alpha_r(\nabla(d_{i_r} - d_{j_r}))$. At a point $x \in \mathcal{SS}_{i_q j_q}$, for all $w \in T_x\mathcal{SS}_{i_q j_q}$, w is orthogonal to $\nabla(d_{i_q} - d_{j_q})(x)$. Therefore,

$$\sum_{r=1}^{q-1} \langle \alpha_r(\nabla(d_{i_r} - d_{j_r})(x)), w\rangle = 0$$
$$\Longrightarrow \sum_{r=1}^{q-1} \alpha_r(\nabla(d_{i_r} - d_{j_r})(x)) = 0. \tag{21}$$

It follows that

$$\nabla(d_{i_1} - d_{j_1})(x) = \sum_{r=2}^{q-1} \frac{\alpha_r}{\alpha_1}(\nabla(d_{i_r} - d_{j_r})(x)), \tag{22}$$

which leads to a contradiction because the rank of $\tilde{G}$ is $q-1$ (i.e., the rows of $\tilde{G}$ are linearly independent of each other). Therefore, $\nabla(d_{i_q} - d_{j_q}) \neq \sum_{r=1}^{q-1} \alpha_r (\nabla(d_{i_r} - d_{j_r}))$, and thus

$$\text{rank}(G(x)) = \text{rank} \begin{bmatrix} \nabla(d_{i_1} - \nabla d_{j_1})(x) \\ \vdots \\ \nabla(d_{i_q} - \nabla d_{j_q)(x)} \end{bmatrix} = q.$$

□

Proof of Lemma 4

Proof. As a consequence of the preimage theorem, each element (i.e., row) of G defines a two-equidistant surjective surface. Since $\mathcal{SS}_{i_1 j_1} \bigcap \cdots \bigcap \mathcal{SS}_{i_q j_q} \neq \emptyset$ and each pair $\{i_r, j_r\}$ is unique (i.e., for all r_1, r_2, $\{i_{r_1}, j_{r_1}\} \neq \{i_{r_2}, j_{r_2}\}$), no two components of G are the same. Therefore, when Assumption 1.2 is upheld, the preimage theorem asserts that $G^{-1}(0)$ is a manifold with codimension q whose tangent space at a point $x \in \mathcal{SS}_{i_1 j_1} \bigcap \cdots \bigcap \mathcal{SS}_{i_q j_q}$ is the null space of $\nabla G(x)$, which is equal to $T_x(\mathcal{SS}_{i_1 j_1} \bigcap \cdots \bigcap \mathcal{SS}_{i_q j_q})$. Finally, let the *normal slice* be the q-dimensional plane orthogonal to the tangent space of $G^{-1}(0)$ at x.

Pick $r \in \{1, \ldots, q\}$. Let c_{i_r} and c_{j_r} be the two closest points on obstacles C_{i_r} and C_{j_r}, respectively, to x. By Lemmas 1 and 2, $T_x \mathcal{SS}_{i_r j_r}$ can be viewed as codimension one plane that is the locus of points equidistant to c_{i_r} and c_{j_r}.

Let $n_1, \ldots, n_{m-1}$ be an orthonormal basis for $T_x \mathcal{SS}_{i_r j_r}$ whose origin is the midpoint of the segment that connects c_{i_r} and c_{j_r}. In this coordinate frame,

$$\begin{aligned} x &= (x^1, \ldots, x^{m-1}, 0)^T \\ c_{i_r} &= (0, \ldots, 0, \alpha)^T \\ c_{j_r} &= (0, \ldots, 0, -\alpha)^T, \end{aligned}$$

where $\alpha = \frac{\|c_{i_r} - c_{j_r}\|}{2}$.

Let the *slice line*, sl_r, be the line that is orthogonal to $T_x \mathcal{SS}_{i_r j_r}$ and passes through x. That is,

$$sl_r = x + \lambda v \quad \forall \lambda \in \mathbb{R},$$

where $v \in (T_x \mathcal{SS}_{ij})^{\perp}$. Let the *base line*, bl_r, be the line defined by c_{i_r} and c_{j_r}, i.e.,

$$\begin{aligned} bl_r &= \lambda(c_{j_r} - c_{i_r}) \qquad \forall \lambda \in \mathbb{R} \\ &= (0, \ldots, 0, \lambda)^T \quad \forall \lambda \in \mathbb{R}. \end{aligned}$$

By construction, bl_r is also orthogonal to $T_x \mathcal{SS}_{i_r j_r}$, i.e., for all $w \in T_x \mathcal{SS}_{ij}$,

$$\begin{aligned} \langle w, bl_r \rangle &= \langle (w^1, \ldots, w^{m-1}, 0)^T, (0, \ldots, 0, \lambda)^T \rangle \quad \forall \lambda \in \mathbb{R} \\ &= 0. \end{aligned}$$

Therefore, the slice line and the base line are parallel, and thus for all $\lambda \in \mathbb{R}$, $bl_r = \lambda v$ where $v \in (T_x \mathcal{SS}_{ij})^{\perp}$.

Let π_{bl_r} be the orthogonal projection onto the bl_r operator. By definition, $\pi_{bl_r}(c_{j_r} - c_{i_r}) = c_{j_r} - c_{i_r}$. From this, we can conclude that $(x - c_{i_r}) - (x - c_{j_r})$ is equal to the projection of itself onto bl_r. In other words, $(x - c_{i_r}) - (x - c_{j_r}) \in bl_r$, or

$$\begin{aligned} \pi_{bl_r}((x - c_{i_r}) - (x - c_{j_r})) &= \pi_{bl_r}(c_{j_r} - c_{i_r}) \\ &= c_{j_r} - c_{i_r} \\ &= (x - c_{i_r}) - (x - c_{j_r}). \end{aligned} \tag{23}$$

Note that $(x - c_{i_r}) = -d_{i_r}(x) \nabla d_{i_r}(x)$ and $(x - c_{j_r}) = -d_{i_r}(x) \nabla d_{j_r}(x)$ (recall that $d_{i_r}(x) = d_{j_r}(x)$). Substitute these relationships into eq. (19).

$$\pi_{bl_r}((x - c_{i_r}) - (x - c_{j_r})) = (x - c_{i_r}) - (x - c_{j_r})$$

$$\begin{aligned} &\pi_{bl_r}(-d_{i_r}(x) \nabla d_{i_r}(x) - (-d_{i_r}(x) \nabla d_{j_r}(x))) \\ &\qquad = -d_{i_r}(x) \nabla d_{i_r}(x) - (-d_{i_r}(x) \nabla d_{j_r}(x)) \end{aligned}$$

$$\begin{aligned} &d_{i_r}(x) \pi_{bl_r}(\nabla d_{i_r}(x) - \nabla d_{j_r}(x)) \\ &\qquad = d_{i_r}(x)(\nabla d_{i_r}(x) - \nabla d_{j_r}(x)) \end{aligned}$$

$$\pi_{bl_r}(\nabla d_{i_r}(x) - \nabla d_{j_r}(x)) = \nabla d_{i_r}(x) - \nabla d_{j_r}(x).$$

Since the slice line is parallel to the base line,

$$\begin{aligned} \pi_{sl_r}(\nabla d_{i_r}(x) - \nabla d_{j_r}(x)) &= \pi_{bl_r}(\nabla d_{i_r}(x) - \nabla d_{j_r}(x)) \\ &= \nabla d_{i_r}(x) - \nabla d_{j_r}(x). \end{aligned}$$

We can conclude that

$$\begin{bmatrix} \nabla(d_{i_1} - d_{j_1})(x) \\ \vdots \\ \nabla(d_{i_q} - d_{j_q})(x) \end{bmatrix} = \begin{bmatrix} \nabla_{sl_1}(d_{i_1} - d_{j_1})(x) \\ \vdots \\ \nabla_{sl_q}(d_{i_q} - d_{j_q})(x) \end{bmatrix},$$

and since slice plane Y is the span of $sl_1, \ldots, sl_q$,

$$\nabla G(x) = \nabla_Y G(x). \tag{24}$$

Therefore, $\text{rank}(\nabla G(x)) = \text{rank}(\nabla_Y G(x))$. □

Acknowledgments

The authors would like to thank Dr. Teresa McMullen at ONR, Grant 97PR06977, and Dr. Howard Moraff, Dr. Jing Xiao, Dr. Larry Reeker, and Dr. Ephraim P. Glinert at NSF, Grant IRI-9702768, for supporting this work. The authors would like to thank the fearless members of Carnegie Mellon's Sensor Based Planning Lab and Caltech Robotics group for the frequent conversations on the topics in this paper. In particular, special thanks goes out to Andrew Lewis, Jim Ostrowski, Luis Goncalves, Andrew Connely, Bill Messner, and Elon Rimon for their insights and contributions to this work. Finally, the authors would like to thank an anonymous reviewer for his or her careful and considerate comments toward this work.

References

Avis, D., and Bhattacharya, B. K. 1983. Algorithms for computing *D*-dimensional Voronoi diagrams and their duals. *Advances in Computing Research* 1:159–180.

Borenstein, J., and Koren, J. 1990 (May). Real-time obstacle avoidance for fast mobile robots in cluttered environments. *IEEE Conference of Robotics and Automation*, pp. 572–577.

Brooks, R. A. 1986. A robust layered control system for a mobile robot. *IEEE Journal on Robotics and Automation* RA-2(March).

Canny, J. F. 1988. *The Complexity of Robot Motion Planning*. Cambridge, MA: MIT Press.

Canny, J. F., and Lin, M. C. 1993. An opportunistic global path planner. *Algorithmica* 10:102–120.

Choset, H. 1996. *Sensor Based Motion Planning: The Hierarchical Generalized Voronoi Graph*. Ph.D. thesis, California Institute of Technology, Pasadena, CA.

Choset, H. 1998. Nonsmooth analysis, convex analysis, and their applications to motion planning. Special Issue of the *Int. J. of Comp. Geom. and Apps.*, to appear.

Choset, H., and Burdick, J. W. 1994. Sensor based planning and nonsmooth analysis. *Proc. IEEE Int. Conf. on Robotics and Automation*, pp. 3034–3041.

Choset, H., and Burdick, J. W. 1995. Sensor based planning, Part I: The generalized Voronoi graph. *Proc. IEEE Int. Conf. on Robotics and Automation*.

Choset, H., and Burdick, J. 2000. Sensor-based motion Exploration: The hierarchical generalized Voronoi graph. *International Journal of Robotics Research* 19:119–148.

Keller, H. B. 1987. *Lectures on Numerical Methods in Bifurcation Problems*. Bombay, India: Tata Institute of Fundamental Research.

Kuipers, B., and Byan, Y. T. 1991. A robot exploration and mapping strategy based on a semantic hierarchy of spatial representations. *Journal of Robotics and Autonomous Systems* 8:47–63.

Rao, N.S.V., Kareti, S., Shi, W., and Iyenagar, S. S. 1993. Robot navigation in unknown terrains: Introductory survey of non-heuristic algorithms. *Oak Ridge National Laboratory Technical Report* ORNL/TM-12410(July):1–58.

Rao, N.S.V., Stolzfus, N., and Iyengar, S. S. 1991. A retraction method for learned navigation in unknown terrains for a circular robot. *IEEE Transactions on Robotics and Automation* 7(October):699–707.

Rimon, E., and Canny, J. F. 1994. Construction of C-space roadmaps using local sensory data—What should the sensors look for? *Proc. IEEE Int. Conf. on Robotics and Automation*, pp. 117–124.