May 1990

# Exact robot navigation in geometrically complicated but topologically simple spaces

Elon Rimon
*Yale University*

Daniel E. Koditschek
*University of Pennsylvania*, kod@seas.upenn.edu

NOTE: At the time of publication, author Daniel Koditschek was affiliated with Yale University. Currently, he is a faculty member in the Department of Electrical and Systems Engineering at the University of Pennsylvania.

# Exact robot navigation in geometrically complicated but topologically simple spaces

**Abstract**
A *navigation function* is an artificial potential energy function on a robot configuration space (C-space) which encodes the task of moving to an arbitrary destination without hitting any obstacle. In particular, such a function possesses no spurious local minima. In this paper we construct navigation functions on *forests of stars*: geometrically complicated C-spaces that are topologically indistinguishable from a simple disc punctured by disjoint smaller discs, representing "model" obstacles. For reasons of mathematical tractability we approximate each C-space obstacle by a Boolean combination of linear and quadratic polynomial inequalities (with "sharp corners" allowed), and use a "calculus" of implicit representations to effectively represent such obstacles. We provide evidence of the effectiveness of this "technology" of implicit representations in the form of several simulation studies illustrated at the end of the paper.

**Comments**

Copyright 1990 IEEE. Reprinted from *Proceedings of the IEEE International Conference on Robotics and Automation*, Volume 3, 1990, pages 1937-1942.

NOTE: At the time of publication, author Daniel Koditschek was affiliated with Yale University. Currently, he is a faculty member in the Department of Electrical and Systems Engineering at the University of Pennsylvania.

# Exact Robot Navigation in Geometrically Complicated but Topologically Simple Spaces

Elon Rimon and Daniel E. Koditschek [1]

Center for Systems Science
Yale University, Department of Electrical Engineering

## Abstract

A *navigation function* is an artificial potential energy function on a robot configuration space (C-space) which encodes the task of moving to an arbitrary destination without hitting any obstacle. In particular, such a function possesses no spurious local minima. In this paper we construct navigation functions on *forests of stars*: geometrically complicated C-spaces that are topologically indistinguishable from a simple disc punctured by disjoint smaller discs, representing "model" obstacles. For reasons of mathematical tractability we approximate each C-space obstacle by a Boolean combination of linear and quadratic polynomial inequalities (with "sharp corners" allowed), and use a "calculus" of implicit representations to effectively represent such obstacles. We provide evidence of the effectiveness of this "technology" of implicit representations in the form of several simulation studies illustrated at the end of the paper.

## 1 Introduction

This paper presents the last in a series concerning exact (and provably correct) navigation in perfectly known environments possessed of a particularly simple topology — the "sphere worlds" In previous work, we have shown that only the topology matters [6]. Moreover, we have constructed an explicit "navigation-function" algorithm for the simplest geometric instance of the sphere world topology [6]. Subsequently, we have provided an algorithm for a much richer variety of geometric detail in that topological equivalence class — the "star worlds" [10].

This paper presents a methodical recipe for handling what we surmise (but have not formally proven) to be "almost every" possible member of this equivalence class. Our present results thus conclude the first stage of a larger research program, attempting to recast the robot navigation problem as the search for a control law that uses an effective parametrization of the robot's à priori knowledge concerning its environment to move in cluttered spaces.

The results of this paper offer persuasive evidence that parametrizations which respect the configuration-space topology are effective with respect to both controller design and geometric complexity. Our controller — a refinement of the well known artificial potential field design [4] — guarantees convergence of the closed-loop robotic system to the desired destination from "almost all" initial configurations (no smooth time invariant controller can do better than this [6]), and guarantees that no collisions with any obstacle will occur along the way. Our recipe for handling geometric details relies upon a "calculus" for implicit representations of semi-analytic sets worked out in the Russian literature [12]. [1] The assumption of perfect à priori information is obviously unrealistic and its relaxation comprises the second stage of our program. In this context, there arise problems of classification and adaptation whose proper formulation we are only beginning to understand.

In the next section (Sec. 2) we present the machinery of our algorithm: the class of forests of stars, and their *purging transformations*. For reasons to be made clear below, the availability of such a transformation solves the navigation problem as posed here. The computational complexity of this procedure turns out to be *polynomial* in the "geometrical complexity" and, in the worst case, *exponential* in the degrees of freedom (Sec. 4). Finally, some simulation studies suggest the practicability of this methodology (Sec. 5).

### 1.1 Recasting the Navigation Problem: The Longer Term Program of Research

We are interested in the following loosely defined problem. Find a parameterized representation of the robot's free C-space, $\mathcal{F}$, and desired destination. Find a map from the parameter space into a set of bounded-torque feedback controllers such that for each parameter value the closed loop robotic system (resulting from application of the controller) moves to the destination without hitting the parameter-determined C-space obstacles. Find a procedure for adjusting the parameter values in conjunction with new sensory data such that the controller brings the robot to the desired destination, or parametric adjustment ceases with the decision that the desired state is not reachable.

We have rigorously formulated and solved only a particular piece of the problem. Namely, we are presently able to map a parameterized class of configuration spaces to bounded-torque controllers. In order to suggest how this piece of work points toward the larger research program it seems worth pausing first to review the general robot motion-planning literature.

The purely geometric problem of constructing a path between two points in a space obstructed by sets with arbitrary polynomially described boundary given perfect information has

[1] This work was supported in part by the National Science Foundation under grants DMC-8505160, and a Presidential Young Investigator Award held by the second author.

[1] We would like to thank Vadim Shapiro and Herb Voelcker for alerting us to the existence of this literature.

CH2876-1/90/0000/1937$01.00 © 1990 IEEE

been completely solved by Schwartz and Sharir[11] and Canny[2]. Unfortunately, it is not presently possible to use these methods "incrementally". Given new information about the world, the computation of a new solution must, in principle, start from the beginning. Moreover, the generation of paths does not address the need for a controller.

The idea of using "potential functions" for the specification of robot tasks with a view of the control problems in mind was pioneered by Khatib[4] in the context of obstacle avoidance. Fundamental work of Hogan[3] in the context of force control further advanced the interest in this approach. A similar methodology has been developed independently by Arimoto in Japan [1], and by Soviet investigators as well [7]. This version of the navigation problem has the virtue of admitting parametric information about the underlying configuration space directly into the controller structure with the result that small changes in present information might result in small changes in the controller. Unfortunately, the use of potential functions has been (prior to our results) a poor heuristic with respect to the global path-planning problem. There are well known and frequent failures due to the appearance of spurious local minima that "trap" the robot away from its nominal destination. In addition, the heuristic approach has paid the price of unbounded inputs for the guarantee of obstacle avoidance.

### 1.1.1 Our Results

In a recent paper [6], we propose a specification the class of *navigation functions* which solves both the control problem and the path generation problem (up to the limits that the topology of the free C-space allows). We construct a navigation function on any Euclidean *sphere world* as a "model" of its topologically deformed class [6]. In a second paper [10], we construct navigation functions on the geometrically much richer class of *star worlds*: a compact star-shaped set in $E^n$ punctured by a finite number of smaller disjoint star-shaped sets. To achieve this, we "pull back" a navigation function on the simpler sphere world model, via a change of coordinates. We suppose that perfect information about these spaces has been furnished in the form of the implicit representation of the star obstacles' boundaries.

Unfortunately, our star world construction requires the obstacles to be disjoint analytic manifolds. In particular, obstacles with "sharp corners" are excluded. Moreover, the restriction to star shapes (while including, for instance, all convex sets) does not seem sufficiently expressive. Finally, the assumption that à priori information will take the form of a global implicit representation of each obstacle seems dubious.

We address all these problems here. First, we allow obstacles with sharp corners, and devise a means of making the gradient of the resulting navigation function smooth and globally bounded away from the sharp corners. Second, we dramatically increase the "geometric expressiveness" of the method by resort to *trees of stars*. These are obstacles built from intersecting star shapes, such that if one puts an edge between the centers of each pair of intersecting stars, the resulting graph is a set of disjoint trees. In the robotics context, C-space obstacles comprised of intersecting stars arise naturally in some (simple) one-link situations (for example, a disc robot moving in a room cluttered with star-shaped objects). Finally, we propose a methodology for generating the implicit representation of a very rich variety of shapes, generated by a catalog of "building blocks" — planar and quadratic polynomial inequalities in $E^n$ — via a "calculus" of implicit representations.

## 1.2 Problem Statement

### 1.2.1 Workspace and Obstacles

In the robotics context, even in the simplest (realistic) situations, the C-space obstacles have sharp corners. The following definition extends the notion of an obstacle to include this possibility. A set $\mathcal{O}_i$ is *semianalytic* if it can be expressed as a Boolean combination of polynomial inequalities [8]. Roughly speaking, two surfaces meet transversally if they are not "tangent" to each other at their intersection points. In the sequel, we denote the boundary of a set $\mathcal{S}$ by $\partial\mathcal{S}$.

**Definition 1** *An* **obstacle with corners,** *$\mathcal{O}_i$, is a semianalytic set in $E^n$ comprised of a finite union and/or intersection of semi-analytic sets, $\mathcal{O}_{ij}$ for $j = 1 \dots L_i$, such that*

1. *the resulting $\mathcal{O}_i$ is a connected and compact manifold;*
2. *the boundaries of any two intersecting subsets, $\partial\mathcal{O}_{ij}$ and $\partial\mathcal{O}_{ik}$, meet transversally.*

For each obstacle, $\mathcal{O}_i$, we require a knowledge of an "obstacle function": a continuous map $\beta_i : E^n \rightarrow I\!R$, representing $\mathcal{O}_i$ in the form

$$\mathcal{O}_i = \{q \in E^n : \beta_i(q) \leq 0\}. \qquad (1)$$

We require as well that each $\beta_i$ be analytic away from the sharp corners of $\mathcal{O}_i$. In general, $\beta_i$ can be automatically constructed from the knowledge of an "obstacle function" for each of its constituent shapes $\mathcal{O}_{ij}$, as we show in the Appendix. For example, any compact polyhedron is an obstacle with corners.

The robot *configuration space*, $\mathcal{W}$, is a connected and compact $n$-dimensional subset of $E^n$, which is an analytic manifold with boundary. For example, $\mathcal{W}$ for a one-link robot is the space of all translations and rotations of $E^3$. The C-space obstacles, $\mathcal{O}_1, \dots \mathcal{O}_M$, are placed inside $\mathcal{W}$ such that they are pairwise disjoint. The *free C-space*, $\mathcal{F}$, remains after subtracting the obstacles' interiors from the workspace. Thus, $\mathcal{F}$ is a compact and connected subset of $E^n$ punctured by $M$ disjoint $n$-dimensional "holes". We refer to the complement of $\mathcal{W}$ in $E^n$ as the zero'th obstacle. In the special case in which $\mathcal{W}$ and the obstacles removed from it are $n$-discs in $E^n$, the resulting free space, $\mathcal{M}$, is an $n$-dimensional *sphere world* with $M$ obstacles.

### 1.2.2 Navigation Functions and Their Invariants

A function $f$ is a homeomorphism if it is one-to-one and onto, and both $f$ and its inverse are continuous. If, in addition, both $f$ and its inverse are analytic, then $f$ is an analytic diffeomorphism.

**Problem statement:** Given a general $n$-dimensional free space, $\mathcal{F}$, each of whose obstacles is homeomorphic to the Euclidean $n$-disc, consider the pair $(\mathcal{F}, \mathcal{M})$, where $\mathcal{M}$ is an $n$-dimensional sphere world with the same (finite) number of obstacles. In each space there is an interior destination point, $q_d \in \overset{\circ}{\mathcal{F}}$ and $p_d \in \overset{\circ}{\mathcal{M}}$. We seek a transformation $f_\lambda$, defined on $\mathcal{F}$, satisfying

1. $f_\lambda$ is a homeomorphism onto $\mathcal{M}$;
2. $f_\lambda$ is an analytic diffeomorphism away from the obstacles' sharp corners;
3. $f_\lambda(q_d) = p_d$.

We call this $f_\lambda$ a **modeling transformation.**

The motivation for this problem is most simply provided by reference to the following definition of a navigation function and a method by which navigation functions can be induced on other spaces. A smooth real valued map $\varphi : \mathcal{F} \to [0,1]$ is a *Morse function* if at each of its critical points ($\nabla\varphi(q) = 0$) its second derivative (the Hessian matrix of $\varphi$) is non-singular. Also, $\mathcal{C} \subset \mathcal{F}$ denotes the obstacles' sharp corners (mentioned in Def. 1).

**Definition 2 ([6],Def. 1 Revised)** *Let $\mathcal{F}$ be an n-dimensional free space, which is connected and compact. A map $\varphi : \mathcal{F} \to [0,1]$, is a* **navigation function** *if $\varphi$*

1. *is continuous on $\mathcal{F}$ and analytic on $\mathcal{F} - \mathcal{C}$;*
2. *has a unique minimum on $\mathcal{F}$, at $q_d \in \overset{\circ}{\mathcal{F}}$;*
3. *is Morse on $\mathcal{F} - \mathcal{C}$;*
4. *attains its maximal value (uniformly) exactly on $\partial\mathcal{F}$.*
5. *has a bounded gradient on $\mathcal{F} - \mathcal{C}$.*

These conditions make $\varphi$ a suitable artificial potential function. It is shown in [5, 6] that the bounded torque control law resulting from a navigation function defines a closed loop robotic system whose trajectories approach the destination point with zero velocity without intersecting obstacles, starting at almost any initial position in the robot configuration space. In general, this is the "strongest" convergence behavior that the topology of the underlying free space allows, as we have shown in [6]. Moreover, we have shown as well that smooth navigation functions exist on any smooth manifold with boundary — hence it makes sense to attempt explicit constructions in specific cases. In particular, we have shown how to do so on any sphere world.

The following Proposition suggests that once a navigation function is constructed on a "model" space, $\mathcal{M}$, it induces navigation functions on many other space.

**Proposition 1.1 ([6],Prop. 2.6)** *Let $\hat{\varphi} : \mathcal{M} \to [0,1]$ be a navigation function on $\mathcal{M}$, and $h : \mathcal{F} \to \mathcal{M}$ be a modeling transformation. Then $\varphi \triangleq \hat{\varphi} \circ h$ is a* navigation function *on $\mathcal{F}$.*

We use the Proposition to construct navigation functions on simple forests of stars (Def. 3 below).

### 1.3 The Available Information

We are concerned with free C-spaces $\mathcal{F}$ whose tree-like obstacles, $\mathcal{O}_i$ for $i = 0 \ldots M$, are comprised of several intersecting stars (Def. 3 below). Denote the index set of all the stars in $\mathcal{F}$ by $\mathcal{I}$, and denote the various stars in $\mathcal{F}$ by $\mathcal{O}_{ij}$, meaning the $j^{th}$ star in the $i^{th}$ tree. Each star in $\mathcal{F}$ is equiped with its own "obstacle function", $\beta_{ij}$ satisfying (1) above. We assume *perfect information*, that is, for a given forest, $\mathcal{F}$, the intersection tree of the stars for each of the tree-like obstacles, the obstacle function for each of the stars, as well as the star center points, are known.

Moreover, for each star obstacle in $\mathcal{F}$ we assume the knowledge of constants, $\{E_{i1}, \ldots E_{i4}\}$ and $E_{p(i)}$ (discussed in [9, Sec. 2]) describing bounds on geometrical features required in the construction of the transformation, and later in its proof of correctness. For example, $E_{i1}$ is an upper bound on the $i^{th}$ obstacle function, $\beta_i$, guaranteeing that the " $2E_{i1}$-thickened " boundaries of the stars (except a "parent-son" pair) still do not intersect. In general, the extraction of these constants from the knowledge of the obstacle functions may prove to be computationally intensive. However, in this paper the obstacle functions are constructed from planar and quadratic polynomials, and explicit formulas giving bounds on the required features should be much easier to obtain.

In the robotics setting, the connectedness of the free space is not a realistic assumption. Certainly, the robot initial configuration determines a specific connected component of its free space, yet this might not include the destination point. At the present we rule out this possibility: our admissible spaces are homeomorphic to some sphere world, therefore connected.

## 2 Construction of the Transformation

In this section we define the forests of stars, and a family of candidate transformations of a forest onto its "purged" version. Composing several such transformations — the number of which is the depth of the deepest tree — yields a transformation of the original forest, $\mathcal{F}$, onto a particular star world. Since we have already shown how to construct a transformation from any star world to a suitable model sphere world [10], and since we have shown how to construct a navigation function on any sphere world [6], the problem is solved according to Proposition 1.1.

### 2.1 Forests of Stars

#### 2.1.1 Star-Shapes and Obstacles

A set $\mathcal{S} \subset E^n$ with non-empty interior is *star shaped* (at $q_0$) if there exists a point $q_0 \in \mathcal{S}$ such that for all $q \in \mathcal{S}$, the line segment joining $q_0$ and $q$ is contained in $\mathcal{S}$. Any star-shaped set is path-connected, and it can be shown that any open star-shaped set is homeomorphic to an open $n$-disc.

Consider an obstacle $\mathcal{O}_i$ with a point $q_i \in \mathcal{O}_i$. Given the implicit representation of its boundary in the form of the zero level set of some real-valued function, $\partial\mathcal{O}_i = \beta_i{}^{-1}(0)$, there is a simple inner-product test on $\nabla\beta_i$ which checks that $\partial\mathcal{O}_i$ is star-shaped with center at $q_i$ [10]. Our construction presumes that the various star-shaped obstacles satisfy this condition.

### 2.1.2 Trees of Stars

Consider now an obstacle $\mathcal{O}_i$ which is a union of several intersecting stars. The arrangement of the stars in $\mathcal{O}_i$ can be partially described by a graph, $(V, E)$, where $V$ are the vertices and $E$ the edges. A tree is a connected graph with no (reduced) edge loops, and a forest is a set of disjoint trees. Given a finite collection of star-shaped obstacles, their *connectivity graph* is the graph whose vertices are the star centers, and whose edges connect centers of stars having non-empty intersection.

**Definition 3** *A* **tree-of-stars** *obstacle is a finite union of intersecting star-shaped obstacles whose connectivity graph is a tree. A* **forest of stars** *is a freespace $\mathcal{F}$ whose obstacles are trees of stars.*

In the special case in which each tree consists of one star, the resulting space is a *star world* [10]. We denote the index set of all the leaves in the forest which belong to trees consisting of more than one star by $\mathcal{L}$, and the index set of all the stars in $\mathcal{F}$ by $\mathcal{I}$.

In this paper we construct navigation functions on a simplified subclass of forests of stars. First, we require that the constituent shapes of the various star obstacles (the shapes in the Boolean combination comprising the $\mathcal{O}_i$'s) be described by polynomial inequalities of degree at most 2. Thus, the boundary of each star obstacle is partitioned into planar and quadratic patches, with "sharp corners" allowed. We call such obstacles *quadratic*, and say that a free space $\mathcal{F}$ is quadratic when all its obstacles are quadratic. Clearly, the boundary of any star-shaped obstacle can be approximated with such patches. The second restriction requires that the trees of stars have the following three properties: the center of each star, $q_i$, is contained in its parent in the tree, each of the stars is connected to its parent via a unique patch, and this patch is "star shaped" with respect to $q_i$. We call a tree-like obstacle having these three properties *simple*, and say that a (quadratic) forest of stars is simple if all its tree-like obstacles are simple. A precise definition of these terms is given in [9, Sec. 2].

## 2.2 The Transformation

We begin with the complete recipe, and then discuss its constituent terms: the star-set deforming factors, $\nu_i$; and the switches, $\sigma_i$. As a notational aid, in certain expressions we write $\vec{x}$ to emphasize that $x$ is vector valued.

**Definition 4** *Let $\mathcal{F}$ be a simple forest of stars. The forest* **purging transformation,** *$f_\lambda$, is a member of the one-parameter family of continuous maps from $\mathcal{F} \subset E^n$ into $E^n$, defined by*

$$\vec{f}_\lambda(q) \triangleq \sum_{j\in\mathcal{L}} \sigma_j(q,\lambda)\vec{f}_j(q) + \sigma_d(q,\lambda)\vec{q}, \qquad (2)$$

*where the "ray scaling" maps $\vec{f}_j$ are defined by,*

$$\vec{f}_j(q) \triangleq \nu_j(q)\,(\vec{q}-\vec{q}_j) + \vec{q}_j,$$

*such that $\nu_j$ is the $j^{th}$ star-set deforming factor, $\sigma_j$ is the $j^{th}$ switch, and $\sigma_d \triangleq 1 - \sum_{j\in\mathcal{L}} \sigma_j$.*

*The forest* **modeling transformation,** *$h_{\lambda_1,\ldots\lambda_d}$, is defined by,*

$$h_{\lambda_d,\ldots\lambda_1}(q) \triangleq f_{\lambda_d} \circ \cdots f_{\lambda_1},$$

*where d is the depth of $\mathcal{F}$, and $f_{\lambda_j}$ are purging transformations.*

The star-set deforming factors, $\nu_i$, and the switches, $\sigma_i$, are defined in [9, Sec. 2]. Let us sketch their role in the transformation. The $i^{th}$ deforming factor, $\nu_i$, is the solution to the following problem. Given a point $q \in E^n$, find a scalar $\nu_i(q)$ such that the line $r_q(\nu_i) = \nu_i(q - q_i) + q_i$ intersects the ellipse $(p-p_i)^T Q_i (p-p_i) = 1$. This translates to a quadratic equation, one of whose solutions is $\nu_i(q)$. The $i^{th}$ switch, $\sigma_i$, maps the interior of $\mathcal{F}$ to $(0,1)$, attains a uniform value of 1 on $\partial\mathcal{O}_i$ (which is a boundary component of $\mathcal{F}$), and vanishes on any other obstacle's boundary except on a portion of its parent's boundary inside a $2E_i$-neighborhood about $\partial\mathcal{O}_i$ in $\mathcal{F}$, on which it varies smoothly between 0 and 1. In the transformation scheme, sufficiently close to the $i^{th}$ leaf's boundary, the "switches" provide a means by which the transformation problem is reduced to the simpler problem of mapping *one* star-shaped leaf onto a portion of its parent's boundary.

Consider a simple forest of stars $\mathcal{F}$. Its *purged version*, $\hat{\mathcal{F}}$, is defined to be $\mathcal{F}$ with the leaves in trees consisting of more than one star filled-in and "reattached" to $\mathcal{F}$ (by construction, $\mathcal{O}_i$ intersects only its parent, and $\mathcal{F}$ is Euclidean $n$-space "punctured" by the various obstacles). In the next section we prove that for suitable values of the parameter $\lambda$, the purging transformation maps the forest $\mathcal{F}$ onto $\hat{\mathcal{F}}$. By composing $d$ such transformations, $\mathcal{F}$ is mapped onto a "model" star world, on which we already know how to construct a navigation function. In [9, Sec. 2] we discuss the construction in detail.

# 3 Correctness of the Construction

The following Theorem is the central contribution of this paper. Given a simple forest of stars, its purged version, $\hat{\mathcal{F}}$, was defined as the space $\mathcal{F}$ with the leaves in trees consisting of more than one star filled-in and "re-attached" to $\mathcal{F}$. In particular, $\hat{\mathcal{F}}$ has the same number of boundary components as $\mathcal{F}$(but each component is a tree with fewer stars).

**Theorem 1** *For any simple forest of stars, $\mathcal{F}$, there exists a positive constant, $\Lambda$, such that if $\lambda \geq \Lambda$, then the* **purging transformation** (Def. 4), *$f_\lambda : \mathcal{F} \to E^n$, has the following properties*

1. *$f_\lambda$ is a* **homeomorphism** *from $\mathcal{F}$ onto $\hat{\mathcal{F}}$;*
2. *$f_\lambda$ is an* **analytic diffeomorphism** *away from $\mathcal{C} \subset \partial\mathcal{F}$, the obstacles' sharp corners;*
3. *the destination point, $q_d$, is a fixed point: $f_\lambda(q_d) = q_d$;*
4. *the Jacobian of $f_\lambda$ is bounded on $\mathcal{F} - \mathcal{C}$, provided that the gradient of the "obstacle functions", $\nabla\beta_i$ for $i \in \mathcal{I}$, is bounded on $\mathcal{F} - \mathcal{C}$.*

The proof is relegated to [9, Sec. 3]. It is important to note that during the proof we compute an explicit formula for $\Lambda$ in terms of the obstacle functions $\beta_i$ and the geometrical parameters $E_i$.

## 4 Counting the Floating Point Operations

The computation involved has two parts. First, when presented with the geometrical data (Sec. 1.3) describing the forest $\mathcal{F}$, we construct a navigation function in the form,

$$\varphi = \varphi_s \circ \left( f_{\lambda_d} \circ \cdots f_{\lambda_1} \right),$$

where $\varphi_s$ is a navigation function on the corresponding star world, $\mathcal{F}_s$, whose star obstacles are the roots of the various trees in $\mathcal{F}$, and $f_{\lambda_i}$ for $i \in \{1, \ldots d\}$ are successive purging transformations, mapping $\mathcal{F}$ onto $\mathcal{F}_s$. The construction of $\varphi$ involves the choice of $d+2$ parameters for the $d+1$ functions comprising $\varphi$ ($\varphi_s$ has two parameters). The computational cost of this part is shown in [9, Sec. 4] to be bounded by

$$10M^2 n + 15M^2 + 25d\,|\mathcal{I}| + |\mathcal{I}|\, n^3$$

(in addition to that of the data terms), where $M$ is the number of tree-like obstacles, $|\mathcal{I}|$ the total number of star obstacles, $d$ the depth of the forest, and $n$ the dimension of the ambient Euclidean space.

Second, the controller has to compute $\nabla\varphi$,

$$\nabla\varphi = \nabla(\varphi_s \circ f_{\lambda_d} \circ \ldots f_{\lambda_1}). \tag{3}$$

It is shown in [9] that this computation is bounded by

$$5d\,|\mathcal{I}| + 15\,|\mathcal{I}|\, n^2 + 5dn^3 + 5d \sum_{j \in \mathcal{I}} (\#\beta_j + \#\nabla\beta_j),$$

where $\#(\beta_i)$ and $\#(\nabla\beta_i)$ are "place holders" for the computational cost of $\beta_i$ and $\nabla\beta_i$.

The terms $\#\beta_i$ and $\#\nabla\beta_i$ are not completely unknown: each obstacle function is obtained from a Boolean combination of $m$ linear and quadratic inequalities. We show in the Appendix that these Boolean combinations can be effectively arranged in a recursive form, so that they involve the evaluation of square-root functions whose number is *linear* in $m$.

It is intuitively clear (but we have not established rigorously) that the number, $m$, of linear or quadratic patches required to represent even simple shapes, for instance an $n$-cube, grows exponentially with the dimension $n$. In general however, a shape is described in terms of polynomials of some high degree $k$. Clearly, the dependence of the number of linear and quadratic inequalities required to approximate such shapes on the dimension of the ambient space and on $k$ should be investigated. We are not aware of any work on this problem.

## 5 Simulation Studies

In Figure 1 we show a forest of stars resembling a building floor plan. There are three internal tree-like obstacles, and the depth of the deepest tree is $d = 4$. According to our method, the purging transformation, $f_{\lambda_i}$, is applied $d$ times, until a space whose obstacles are the roots of the original trees is obtained. This space is a star world, and we apply our previously constructed star-world to sphere-world transformation, $h_\lambda$ [10], to obtain the corresponding model sphere world, $\mathcal{M}$. Thus the sequence of transformations is

$$\mathcal{F} \xrightarrow{f_{\lambda_1}} \mathcal{F}_1 \xrightarrow{f_{\lambda_2}} \mathcal{F}_2 \xrightarrow{f_{\lambda_3}} \mathcal{F}_3 \xrightarrow{f_{\lambda_4}} \mathcal{F}_4 \xrightarrow{h_\lambda} \mathcal{M}.$$

We show each of the "intermediate" spaces, as well as the level lines of the navigation function as it is "pulled back" via these spaces. The destination point is chosen arbitrarily at the origin, and the level lines vary between zero (at the destination point), and one (on all the boundary components). The parameter in the sphere world is chosen sufficiently high to eliminate spurious local minima in $\mathcal{M}$, according to the results of [6]. The parameters in the transformations $h_\lambda$ and $f_{\lambda_i}$ for $i = 1 \ldots 4$ are chosen sufficiently high so that no new local minima will appear, according to [10] and this paper. Thus there is a unique minimum at the destination point, and one saddle point near each (internal) tree-like obstacle. As we have shown in [6], one cannot do better than this using smooth vector fields.

The simulations, while corroborating the theory, reveal certain numerical difficulties. The level lines clearly depict a unique minimum at the destination point. But when the various "valleys" approach the obstacles, they are so close to the boundary that they can be seen again only on the far side of each obstacle, as we portray in Figure 2. We have found from further numerical experimentation that such situations — when the valleys "hug" the boundaries too tightly — lead to gradient vector fields that vary too abruptly to be implemented in a practical setting. The phenomenon becomes even more acute when we increase the parameter values. Of course, the theory requires in general that these values be increased to avoid spurious local minima.

To understand this behavior, consider the purging transformation,

$$\vec{f}_\lambda(q) = \sum_{i \in \mathcal{L}} \sigma_i(q, \lambda) \vec{f}_i(q) + (1 - \sum_{i \in \mathcal{L}} \sigma_i(q, \lambda)) \vec{\mathbf{id}}(q).$$

The effect of increasing the parameter $\lambda$ is to make the switches $\sigma_i$ vanish more rapidly away from the obstacles' boundary, rendering the transformation essentially the identity mapping. Thus, all the "interesting" features are confined to small neighborhoods about the obstacles' boundaries.

In Figure 3, we have experimented with various intuitive numerical remedies for the problem of "disappearing" valleys. Additional parameters introduced in the switches $\sigma_i$ make their value close to unity in a neighborhood about the boundaries, and only then decrease to zero. Intuitively, this slows the transition of the purging transformation to the identity map, and as a consequence the valleys move away from the obstacles' boundary. We plot in this figure the enlarged image of some interesting regions in $\mathcal{F}$, demonstrating the "reappearance" of the valleys.

It becomes clear from these simulations that the practicability of our construction depends on the development of some control mechanism for the location of the valleys. This is the subject of research now in progress.

## Conclusion

We suspect that obstacles in $E^n$ comprised of intersecting star-shaped sets can approximate the entire class of topologically deformed sphere worlds. Moreover, it is plausible that a tree-like decomposition of any topologically deformed disc into intersecting star-shaped sets could be found. However, we currently do not know how "far away" the sphere-world equivalence class lies from the most general realistic problem — the class of free configuration spaces which arise when a general kinematic chain operates in a cluttered environment.

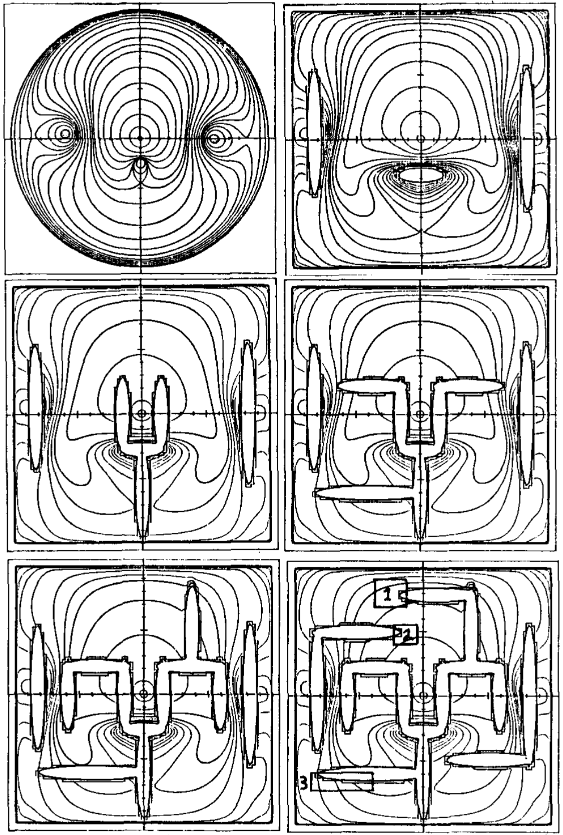

Figure 1: Planar forest of stars with three internal tree-like obstacles (bottom right), its "purged" versions, and its model sphere world (top left).

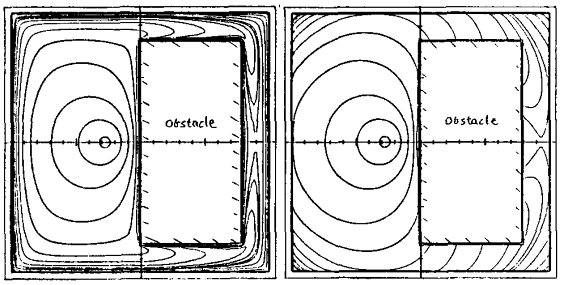

Figure 2: The phenomenon of "disappearing" valleys. The valleys are clearly depicted for low parameter values (left) but disappear as the parameter values increase (right).

## A A "Calculus" for Implicit Representations

In this section we present formulas, introduced by [12], for constructing the implicit representation of a set described by an arbitrary union and intersection of other sets, each of whose implicit representation is already known. If $m$ sets in $E^n$ are described by the inequalities $\beta_i(x_1,\dots x_n) \le 0$, then their intersection is given by $\psi_\cap(\beta_1,\dots\beta_m) \le 0$, and their union by $\psi_\cup(\beta_1,\dots\beta_m) \le 0$. The functions $\psi_\cap$ and $\psi_\cup$ for $m = 2$ are

$$\psi_\cap(\beta_1,\beta_2) \triangleq \beta_1+\beta_2+\sqrt{\beta_1^2+\beta_2^2} \quad \psi_\cup(\beta_1,\beta_2) \triangleq \beta_1+\beta_2-\sqrt{\beta_1^2+\beta_2^2}.$$

The general formulas are discussed in [9, App. A].

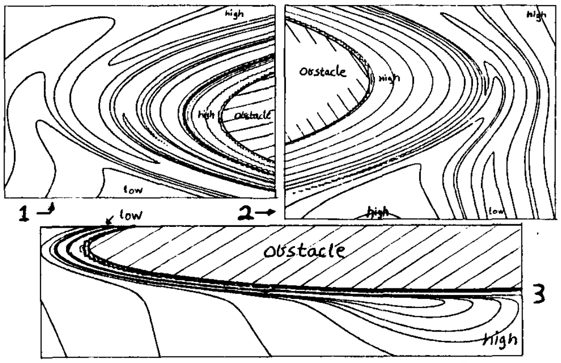

Figure 3: The enlarged image of some regions in Figure 1.

The computation of $\psi_\cup$ (or $\psi_\cap$) for $m$ sets can be effectively arranged in a binary tree. Each node computes a term of the form

$$\beta_i + \beta_j - \sqrt{\beta_i^2 + \beta_j^2}. \qquad (4)$$

The number of (internal) nodes in a balanced binary tree with $m$ leaves is $m - 1$. Thus $\psi_\cup$ "invokes" itself roughly $m$ times, and the computational cost of $\psi_\cup$ is *linear* in $m$; in addition to the computational cost of the $\beta_i$'s themselves.

## References

[1] S. Arimoto and F. Miyazaki. Asymptotic stability of feedback controls for robot manipulators. In *Proceedings 1st IFAC Symposium on robot Control*, Barcelona, Spain, 1985.

[2] J. F. Canny. The complexity of robot motion planning. Ph.d. dissertation, Dpt of Electrical Engineering and Computer Science M.I.T., May 1987.

[3] N. Hogan. Impedance control: An approach to manipulation. *ASME Journal of Dynamics Systems,Measurement, and Control*, 107:1–7, Mar 1985.

[4] O. Khatib. Real time obstacle avoidance for manipulators and mobile robots. *The International Journal of Robotics Research*, 5(1):90–99, Spring 1986.

[5] D. E. Koditschek. The application of total energy as a lyapunov function for mechanical control systems. In Marsden and Krishnaprasad, editors, *Control Theory and Multibody Systems, AMS Series in Contemporary Mathematics*, volume 97, pages 131–158. American Mathematical Society, 1988.

[6] D. E. Koditschek and E. Rimon. Robot navigation functions on manifolds with boundary. *Advances in Applied Mathematics*, (to appear).

[7] V. V. Pavlov and A. N. Voronin. The method of potential functions for coding constraints of the external space in an intelligent mobile robot. *Soviet Automatic Control*, (6), 1984.

[8] A. Requicha. Representations for rigid solids: Theory, methods, and systems. *ACM Computing Surveys*, 12:437–464, 1980.

[9] E. Rimon. The construction of analytic diffeomorphisms for exact robot navigation on forests of stars. Technical Report 8910, Center for Systems Science, Yale University, July 1989.

[10] E. Rimon and D. E. Koditschek. The construction of analytic diffeomorphisms for exact robot navigation on star worlds. *Transactions of the American Mathematical Society*, (to appear).

[11] J. T. Schwartz and M. Sharir. On the "piano movers" problem. ii. general techniques for computing topological properties of real algebraic manifolds. *Advances in Applied Mathematics*, 4(1):298–351, 1983.

[12] O. V. Zenkin. On the analytic description of geometric shapes (in russian). *Kibernetika*, (4):103–110, 1970.