Roadmaps using Gradient Extremal Paths

Ioannis Filippidis and Kostas J. Kyriakopoulos

Abstract— **This work proposes a motion planning method based on the construction of a roadmap connecting the critical points of a potential field or a distance function. It aims to overcome the limitation of potential field methods due to local minima caused by concave obstacles. The roadmap is incrementally constructed by a two-step procedure. Starting from a minimum, adjacent saddle-points are found using a local saddle-point search method. Then, the new saddle-points are connected to the minima by gradient descent. A numerical continuation algorithm from the computational chemistry literature is used to find saddle-points. It traces the valleys of the potential field, which are gradient extremal paths, defined as the points where the gradient is an eigenvector of the Hessian matrix. The definition of gradient bisectors is also discussed. The presentation conclude simulations in cluttered environments.**

I. INTRODUCTION

The problem of motion planning has attracted much attention over the last three decades [1], [2]. Representatives of different approaches are roadmaps [3], cell decompositions [4], sampling-based methods like RRT, RRT*, PRM* [5] and potential-field based methods.

Roadmap methods construct a graph whose edges are feasible paths. Given an initial and final point, they are first connected to the roadmap. This reduces the problem to a graph search within the roadmap to connect the entry and exit points. So they are global methods, with the associated computational overhead.

Potential field methods have been pioneered by Khatib [6] and concern the construction of an artificial potential field, such that the destination be a minimum. A robot following the potential's negated gradient is led to the desired destination. In order to avoid collision with obstacles, the potential is maximal on the obstacle boundaries.

However, if a local minimum other than the destination exists, and the robot starts somewhere within its basin of attraction, then it fails to reach its destination. This original limitation was overcome by Navigation Functions (NFs) introduced by Koditschek and Rimon [7], which are free of undesired local minima and exist for *any* (suff. smooth) Riemannian manifold with boundary. However a constructive procedure is available only for sphere worlds [8], [9]. Although also applicable to more complex geometries and topologies [10], [11], e.g., channel surfaces, a converse result holds for concave obstacles [11].

Ioannis Filippidis is currently with the Control and Dynamical Systems Dept., California Institute of Technology, Pasadena, CA, 91125, USA and was previously with CSL, NTUA. Kostas J. Kyriakopoulos is with the Control Systems Lab, Department of Mechanical Engineering, National Technical University of Athens, 9 Heroon Polytechniou Street, Zografou 15780, Greece. E-mail: `ifilippi@caltech.edu`, `kkyria@mail.ntua.gr`

The construction of a graph connecting critical points to obtain a roadmap is motivated by results from Morse theory and was first proposed by Canny [3], using slices of the C-space. Canny and Lin [12] proposed the construction of roadmaps connecting the critical points of the restriction of a distance function to slices of the world. Slices of fixed direction were used, so the critical points where those points where the distance gradient is normal to the slice. So the freeways of [12] comprise of those points which have a fixed gradient direction, the normal to the slices. These are similar to [3] but are constructed in the free space interior, not on the obstacle boundaries. Rimon and Canny [13], [14] extended this work towards using on-line distance measurements, without the need for an a priori available distance function. A difficulty in the above methods is the remote detection of split and joint points [12], [14] where the slice's topology changes. This issue is not encountered here, because the gradient direction is not fixed.

Barraquand et al. [15] proposed the construction of a graph connecting all critical points of a discrete potential field using various methods. One method is valley-guided motion, where valley points are minima of the potential within slices normal to the coordinate axes. This is a discretized version of the freeways in [12] and is equivalent to the Reduced Gradient Following (RGF) method [16], [17] reviewed later. The algorithm has similar structure to the retraction algorithm proposed in [18]. In contrast, the method used here follows the direction of least ascent, which can vary and defines exactly the valleys of the potential. It uses a modified version of the RGF, with an iterative correction of the gradient direction being searched [19]. During the review process, the authors obtained a copy of [20] where the locus of points where the gradient is a Hessian eigenvector was proposed for motion planning, although no algorithm for the continuous case was proposed there. Critical point roadmaps have also been studied for coverage tasks using slices of fixed direction in [21], where non-smooth boundaries are treated.

Path planning using a Lagrangian global optimization method under constraints has been proposed in [22], [23]. In [22] planning takes place in a space of dimension equal to the sum of the C-space dimension and the number of obstacles, resulting in a roadmap connecting the maxima and minima of the potential. The completeness of the algorithm relies on results from [24]. In [25] the construction of a graph connecting critical points of a distance function in contact space is discussed, with a discretized implementation.

In the present work we propose the construction of a roadmap comprised of gradient extremal paths (GE), which are valleys (and ridges) of the potential field. The GEs connect the minima to saddle-points and maxima, so they can overcome the basins of local minima, although the network is not always globally connected. To construct each GE segment (valley), a local saddle-point search method from [19] is used, which is based on a predictor-corrector numerical continuation algorithm [26]. Nesting valley construction within graph searching produces the roadmap.

Both Koditschek-Rimon (KR) potential functions (not necessarily NFs) and distance functions are used here as scalar fields. KR potentials reduce the search space, by navigating “around” geometric features with sufficient relative convexity [10], [11]. Concave obstacle points can cause local minima, “triggering” the graph search when needed.

The rest of this paper is organized as following: the problem is defined in § II, local saddle-point searching methods reviewed in § III, the composite navigation algorithm described in § IV and demonstrated though simulations in § V and conclusions are summarized in § VI.

II. PROBLEM DEFINITION

Let $\mathscr{O}_i \triangleq \{x \in M | \, \beta_i(x) < 0\}$ where $i \in I \triangleq \mathbb{N}^*_{\leq N}$ are $N \in \mathbb{N}^*$ obstacle sets on the C^2 n-dimensional Riemannian manifold M, where $\beta_i \in C^2(M \setminus \overline{\mathscr{O}_i}, \mathbb{R})$. When using potential fields, then β_i should be C^2 on a subset dense in $\partial\mathscr{O}_i$. If using distance functions and searching though obstacles, then $\beta_i \in C^2(M, \mathbb{R})$. Define the free space as $\mathscr{F} \triangleq M \setminus \bigcup_{i\in I} \mathscr{O}_i$ and assume it is bounded. By definition $\mathscr{F}$ is closed in M, so $\mathscr{F}$ is compact.

A. *Distance functions*

Distance functions are used to define obstacles and as one type of scalar field for roadmap construction. Being well-defined within obstacles, they enable searching through them, which can accelerate the search by taking ”short-cuts”. Their weak point is degeneracy (possibly uncountable critical set) even in simple cases, e.g. for a corridor with parallel sides. The roadmap constructed using a distance function can be used for multiple queries, by connecting each pair of initial condition and destination to the roadmap.

Constructive Solid Geometry can be performed using Rvachev functions [27], [28] to apply Boolean operations over the implicit obstacle functions β_i, e.g., Fig. 1. Two alternatives are $R^m(u,v) \triangleq (u + v \pm \sqrt{u^2+v^2})(u^2 + v^2)^{\frac{m}{2}}, R_p(u,v) \triangleq u + v \pm (u^p + v^p)^{\frac{1}{p}}$ where m, p even positive integers and $u, v \in \mathbb{R}$. The plus sign corresponds to disjunction and the minus to conjunction. While R_m functions are C^m everywhere (even at corners $u = v = 0$), they have zero gradient at corners [29]. They must be used if searching though obstacles using a distance field. R_p functions are not differentiable at corner points, but normalized [29]. In the case of R_p functions with $u = \beta_i(x)$ and $v = \beta_j(x)$, the gradient is $\nabla_x R_p = \nabla u + \nabla v \pm (u^p + v^p)^{\frac{1}{p}-1} \left(u^{p-1}\nabla u + v^{p-1}\nabla v\right)$ and the Hessian $D^2_x R_p = (1 \pm u^p w^{\frac{1}{p}-1})D^2 u + (1 \pm v^{p-1} w^{\frac{1}{p}-1})D^2 v \pm ((p-1)u^{p-2}v^p w^{\frac{1}{p}-1})\nabla u \nabla u^{\mathrm{T}} \pm ((p-1)u^p v^{p-2} w^{\frac{1}{p}-2})\nabla v \nabla v^{\mathrm{T}} \pm ((1-p)u^{p-1}v^{p-1}w^{\frac{1}{p}-2}) \cdot (\nabla u \nabla v^{\mathrm{T}} + \nabla v \nabla u^{\mathrm{T}})$, where $w \triangleq u^p + v^p$.

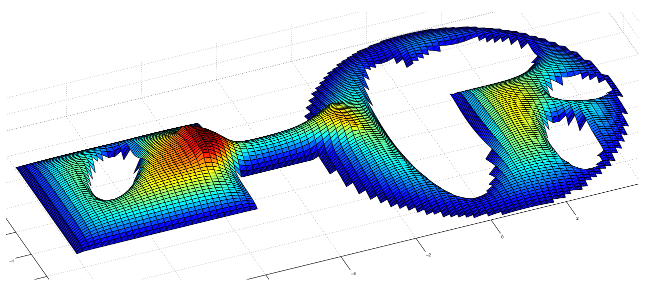
Fig. 1: Distance function for the 2-dimensional environment of Fig. 4.

B. *Potential fields*

In addition to distance functions, the potential field proposed by Koditschek and Rimon [7] is used here, defined as $\varphi \triangleq \frac{\gamma_d}{(\gamma_d^k + \beta)^{\frac{1}{k}}}$ where $k \geq 2$, $\gamma_d \triangleq \|x - x_d\|^2$ and β results using Rvachev operations over the geometric primitives β_i. The potential φ can be undesirably flat and is not always defined within obstacles, so the search should remain within the free space. For almost all destinations there exists some tuning for which φ is Morse [11]. By connecting multiple initial conditions and destinations to the roadmap using gradient descent, KRf roadmaps can be used for multiple queries. The gradient of the Koditschek-Rimon potential is $\nabla\phi = (\gamma_d^k + \beta)^{-\frac{1}{k}-1}(\beta\nabla\gamma_d - \frac{\gamma_d}{k}\nabla\beta)$ and the Hessian $D^2\varphi = (\beta(\gamma_d^k + \beta)^{-\frac{1}{k}-1})D^2\gamma_d + (-\frac{1}{k}\gamma_d(\gamma_d^k + \beta)^{-\frac{1}{k}-1})D^2\beta - (\gamma_d^k + \beta)^{-\frac{1}{k}-2}\Big(((k+1)\beta\gamma_d^{k-1})\nabla\gamma_d\nabla\gamma_d^{\mathrm{T}} + (-\frac{1}{k}\left(\frac{1}{k}+1)\gamma_d\right)\nabla\beta\nabla\beta^{\mathrm{T}} + (-\gamma_d^k + \frac{\beta}{k})(\nabla\gamma_d\nabla\beta^{\mathrm{T}} + \nabla\beta\nabla\gamma_d^{\mathrm{T}})\Big)$.

C. *Problem Statement*

The path planning problem comprises of finding a path in the free space $\mathscr{F}$ which connects the initial configuration $x(0)$ of the system to the desired configuration x_d. We are interested in an algorithm which solves the motion planning problem by constructing a roadmap connecting the minima and saddle-points of the scalar function used. The algorithm should also connect the initial configuration $x(0)$ to the roadmap and, depending on the type of scalar function used (β or φ), the final configuration x_d as well.

III. LOCAL SADDLE-POINT SEARCH METHODS

Finding saddle-points is inherently more difficult than minima and the gradient flow cannot be exploited for this purpose. It has been studied extensively in the literature, however most proposed methods are global (e.g. level-set methods, nudged elastic band) and many others are to a large extent heuristic. In motion planning a method using *local* information would be desired. A standard problem in computational chemistry is the calculation of reaction paths. These start from reactants and yield products, which are configurations associated with local minima of potential energy, through some saddle point. Local saddle-searching methods have been developed for this purpose.

A. Gradient Extremal Paths

Definition 1 (Gradient Extremal [30], [31]): Let f be a twice continuously differentiable function. The locus of points where the gradient is an eigenvector of the Hessian matrix, i.e., there exists a $\lambda \in \mathbb{R}$ such that $D^2 f \nabla f = \lambda \nabla f$ is called the *Gradient Extremal Locus*.

Geometrically, gradient extremal paths are the valleys, ridges, cirques and cliffs of a scalar function's graph [31]. Several equivalent definitions are possible. The gradient extremal is the locus of points x at which the gradient norm restricted to a level set $\|\nabla f\| \,|_{f=c}$ attains its minimum or maximum at x (more generally is stationary). This can be proved by showing that the defining equation is equivalent to the Lagrangian formulation of the constrained optimization problem of $\|\nabla f\|$ over the level set $f = c$ as $\nabla \left\{ \frac{1}{2} \|\nabla f\|^2 - \lambda f \right\} = 0$, which implies critical points belong to the gradient extremal locus. An equivalent definition is as the locus of points where the gradient flow is geodesic [32]. Moreover, all gradient extremal locus segments start and end at critical points along the direction of the eigenvectors of the Hessian matrix [31].

Gradient extremal paths have attracted much attention over the past 30 years in the theoretical and computational chemistry literature. One of the first methods of tracing these curves was introduced in [33]. However, it suffered from numerical difficulties at bifurcation and turning points.

Reduced Gradient Following (RGF) [16] is another saddle-searching method which is closely related to gradient extremal paths. It is equivalent to Branin's method [17] and implements a predictor-corrector numerical continuation algorithm [26] to follow *Newton Trajectories* [34], [35]. These are the loci of points where the gradient has some desired, fixed direction. The freeways in [12], ridge curves in [13] and valleys in [15] are all equivalent to RGF in a direction fixed and normal to the C-space slices.

The *TAngent Search Correction to the RGF* (TASC) is a modified version of Reduced Gradient Following proposed by Quapp et. all in [19]. While the predictor step of the RGF searches for a point with a priori fixed gradient direction, in the TASC the gradient direction searched is iteratively updated. In each iteration, the curve tangent of the previous step becomes the new desired direction for the gradient. The TASC algorithm converges to a valley curve, as proved in [36]. Therefore, it constitutes a more robust method for constructing those Gradient Extremal Paths which are valleys.

We can also define the locus of points where the gradient forms equal angles with each eigenvector of the Hessian (if on the plane, then it bisects the orthogonal angle formed by each pair of eigenvectors), which we will refer to as *gradient bisectors*. This new definition is also used here as an alternative to gradient extremal paths and compared to them in Fig. 2 and the simulations section.

B. Numerical Continuation and Adaptation

A predictor-corrector procedure [26] forms the core of the TASC algorithm, which traces the solutions of a different equation in each iteration, instead of a fixed equation (as in RGF). The predictor step in Algorithm 2 increases the integration step s if the proximity to the solution curve is smaller than the tolerance ε_{exact} and s is smaller than the upper threshold $s_{\max} = c_{\max} \varepsilon_{exact}$. Then, it moves along the current tangent $x'(t_i)$ by s. The corrector step in Algorithm 3 reduces the integration step s if s is larger than the lower threshold $s_{\min} = c_{\min} \varepsilon_{exact}$. Then, it solves the equation $P_r \nabla \varphi = 0$ (where $P_r = I - rr^{\mathrm{T}}$ is the matrix projecting on a basis of $r^{\perp}$) within $r^{\perp} = x'(t_i)^{\perp}$, using the corrector convergence tolerance $\varepsilon_{cor} = \frac{s}{10}$. During the solution it computes the Moore-Penrose inverse $B^{\dagger} = B^{\mathrm{T}}(BB^{\mathrm{T}})^{-1}$ of the projected Hessian matrix $P_r D^2 \varphi$.

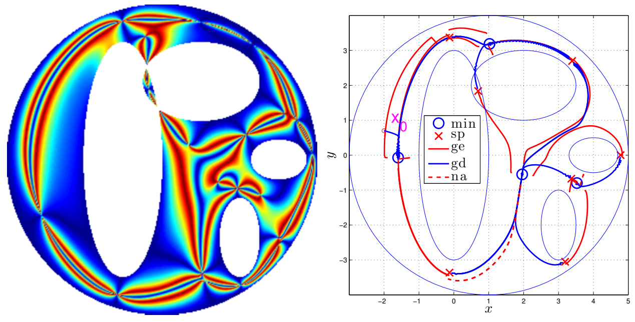

Fig. 2: Left: Minimal angle $\theta_{\min}$ of gradient with Hessian eigenvectors. Deep blue curves are *gradient extremal paths* ($\theta_{\min} = 0$) and red curves are *gradient bisectors* ($\theta_{\min} = \frac{\pi}{4}$). Observe that the network of gradient bisectors has better connectivity, as compared to gradient extremal paths (in free space). Right: Critical point roadmap using a distance function and gradient extremal paths for saddle-point searches.

In the TASC Algorithm 1 we define the predictor step s and the tolerance ε_{tol} (convergence to saddle-point), ε_{close} (close to the solution curve), ε_{exact} ("on" the solution curve). The search starts at a minimum $x_{\min}$ in the direction of the eigenvectors of the Hessian matrix, because gradient extremal paths are tangent to them at critical points. A small perturbation factor $c_p > \frac{\varepsilon_{tol}}{s}$ avoids false convergence to the initial point within the first iteration. Then, predictor-corrector steps are iteratively performed until convergence. Note that in each iteration the desired gradient direction r is updated to become equal with the tangent x' of the previous predictor step, ensuring convergence to a valley gradient extremal path [36].

For brevity, the algorithm listings do not include some additional convergence and other tests. The length of the Newton-Raphson step, the gradient norm and Hessian matrix index are used as convergence criteria. To avoid passing near a saddle-point without noticing because of too small ε_{tol}, a test is performed of switching from ascending to descending mode. If such a change is detected, then the algorithm stops and a Newton-Raphson iteration is locally performed to improve the saddle-point estimate. When using φ, which may not be well-defined in obstacles, it is tested whether $x \in \mathscr{F}$.

Algorithm 1 TASC Saddle-Searching Method by Gradient Extremal Following [19], [36]

```
1: procedure TASC($x_{\min}$, $\varphi$, $s$, $\varepsilon_{tol}$, $\varepsilon_{close}$, $\varepsilon_{exact}$, $c_{\min}$, $c_{\max}$, $c_p$)
2:     $r \leftarrow$ eigenvector($D^2\varphi$), $x(t_i) \leftarrow x_{\min} + c_p s \frac{r}{\|r\|}$
3:     $x'(t_i) \leftarrow r$, $stop = 0$
4:     while $stop = 0$ do
5:         $g \leftarrow \nabla\varphi(x(t_i))$, $H \leftarrow D^2\varphi(x(t_i))$
6:         $\Delta x_{NR} \leftarrow -H^{-1}g$          ▷ Newton-Raphson step
7:         if $\|\Delta x_{NR}\| < \varepsilon_{tol}$ then
8:             $stop \leftarrow 1$
9:         end if
10:        $r \leftarrow x'(t_i)$          ▷ Iteratively update gradient direction searched
11:        $P_r \leftarrow I - rr^{\mathrm{T}} \in \mathbb{R}^{(n-1)\times n}$          ▷ Projection on $r^{\perp}$
12:        $u \leftarrow P_r g \in \mathbb{R}^{(n-1)}$
13:        $B \leftarrow P_r H \in \mathbb{R}^{(n-1)\times n}$          ▷ Project Hessian
14:        $x'(t_{i+1}) \leftarrow \mathrm{null}(B)$          ▷ Kernel of $B$
15:        $\varepsilon \leftarrow \|u\|$          ▷ Equation satisfaction error
16:        if $\varepsilon < \varepsilon_{close}$ then
17:            $[x(t_{i+1}), s] \leftarrow$PREDICTOR($x(t_i)$, $x'(t_i)$, $s$, $\varepsilon_{exact}$, $c_{\max}$)
18:        else
19:            $[x(t_{i+1}), s] \leftarrow$CORRECTOR($s$, $\varepsilon_{tol}$, $c_{\min}$)
20:        end if
21:        $i \leftarrow i + 1$
22:    end while
23: end procedure
```

Algorithm 2 TASC Predictor

```
1: procedure PREDICTOR($x(t_i)$, $x'(t_i)$, $s$, $\varepsilon_{exact}$, $c_{\max}$)
2:     $s_{\max} \leftarrow c_{\max}\varepsilon_{exact}$
3:     if $\varepsilon < \varepsilon_{exact}$ and $s < s_{\max}$ then
4:         $s \leftarrow s\sqrt{2}$
5:     end if
6:     $x(t_{i+1}) \leftarrow x(t_i) + s\frac{x'(t_i)}{\|x'(t_i)\|}$
7:     return $x(t_{i+1}), s$
8: end procedure
```

Algorithm 3 TASC Corrector

```
1: procedure CORRECTOR($s$, $\varepsilon_{tol}$, $c_{\min}$)
2:     $s_{\min} \leftarrow c_{\min}\varepsilon_{tol}$
3:     if $s > s_{\min}$ then
4:         $s \leftarrow \frac{s}{\sqrt{2}}$
5:     end if
6:     $\varepsilon_{cor} \leftarrow \frac{s}{10}$ ,$stop \leftarrow 0$
7:     while $stop = 0$ do
8:         $g \leftarrow \nabla\varphi(x(t_i))$, $H \leftarrow D^2\varphi(x(t_i))$, $r \leftarrow x'(t_i)$
9:         $P_r \leftarrow I - rr^{\mathrm{T}} \in \mathbb{R}^{(n-1)\times n}$          ▷ Projection on $r^{\perp}$
10:        $u \leftarrow P_r g$, $B \leftarrow P_r H \in \mathbb{R}^{(n-1)\times n}$
11:        $\varepsilon \leftarrow \|u\|$
12:        if $\varepsilon < \varepsilon_{cor}$ then
13:            $stop \leftarrow 1$
14:        end if
15:        $B^{\dagger} \leftarrow B^{\mathrm{T}}(BB^{\mathrm{T}})^{-1}$          ▷ Moore-Penrose Inverse
16:        $x(t_i) \leftarrow x(t_i) - B^{\dagger}u$          ▷ Corrector step update
17:    end while
18:    $x(t_{i+1}) \leftarrow x(t_i)$
19:    return $x(t_{i+1}), s$
20: end procedure
```

Algorithm 4 Critical Point Graph Construction by Depth-First Search

```
1: procedure CRITICAL GRAPH($x_0$, $\varphi$, $depth$, $r$)
2:     $x(t_i) \leftarrow$GRADIENTDESCENT($x_0$)
3:     $Q \leftarrow \{x(t_i)\}$          ▷ Push to queue
4:     for $i = 1 : depth$ do
5:         $x \leftarrow pop(Q)$, $Q \leftarrow Q \setminus \{x\}$          ▷ Pop queue
6:         $\{x_{\min}, x_{sad}\} \leftarrow$ SEARCHNEXTSADDLES
7:         $x_{\min} \leftarrow$ REMOVEPROXIMATE($x_{\min}$, $r$)
8:         $x_{\min} \leftarrow$ REMOVEVISITED($x_{\min}$, $x_c$, $r$)
9:         $Q \leftarrow \{x_{\min}\} \cup Q$          ▷ Push new minima to queue
10:        $x_c \leftarrow x_c \cup \{x_{\min}, x_{sad}\}$
11:        $A \leftarrow$ UPDATEADJACENCY($A$, $x_c$, $r$)
12:        REMOVEPROXIMATE($x_c$, $A$, $r$)
13:        CONVERGENCETEST
14:    end for
15: end procedure
```

IV. NAVIGATION ALGORITHM

The path planning Algorithm 4 described starts by a gradient descent from $x(0)$ to find a local minimum. Then using TASC it follows the four gradient extremals emenating along the four Hessian eigenvectors at the minimum. This is repeated nested in a depth-first or breadth-first graph-searching algorithm. Duplicate saddle-points are avoided by considering x_1, x_2 as the same saddle-point if $\|x_1 - x_2\| < d$, some threshold. The RemoveVisited function works similarly and compares newly discovered critical points to already known ones. The UpdateAdjacency function adds the connectivity information between discovered minima and saddles to the roadmap graph's adjacency matrix maintained. The convergence test checks if the destination is connected to the roadmap (or the minimum associated with the destination, in case of a distance function).

The method's completeness depends on the global connectivity of the network of gradient extremal paths. Unfortunately, these are not always globally connected and the TASC method is attracted to valleys which run along the smallest eigenvalue of the Hessian matrix [19], so it is difficult to converge to steep and transversely shallow saddle-points, also called Don Quixote saddles [37]. Two possible improvements are constructing a ridge from a maximum to a saddle-point and the possibility of allowing the search to go through obstacles. Bidirectional search is another approach which can improve the completeness of the algorithm. Despite this limitation, the method proves its applicability in the simulations.

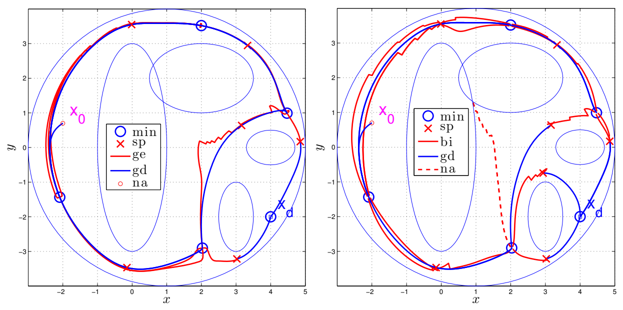

Fig. 3: Critical point roadmap on Koditschek-Rimon function. Critical point roadmap of a Koditschek-Rimon function constructed using the Gradient Bisector algorithm. Observe that the gradient bisectors achieve improved coverage of the free space.

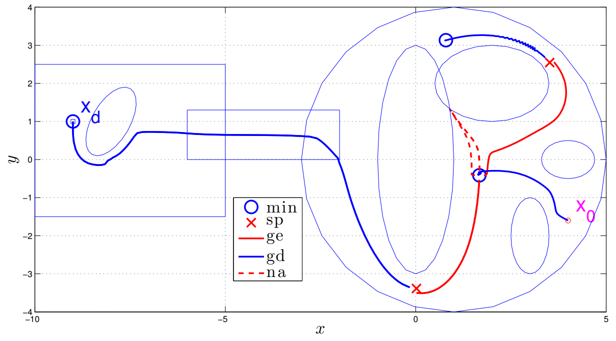

Fig. 4: Navigation in a complicated 2-dimensional environment using the Koditschek-Rimon potential field and gradient extremal paths for saddle-point searches.

V. SIMULATION RESULTS

This section includes a number of simulations demonstrating the proposed method. The figure abbreviations are: ge (gradient extremal), bi (gradient bisectors), gd (gradient descent), na (failed search). In Fig. 3-a a 2-dimensional world with ellipse obstacles is navigated using the method. Starting from the initial condition on the left, the search finds saddle-point passages and then discovers the adjacent basins and their associated minima. After a number of iterations, the method converges to the destination, connecting it to the initial configuration through the constructed roadmap, which is comprised of gradient extremal paths and gradient descent paths. In Fig. 3-b the same world is navigated by gradient bisectors and in Fig. 2-b using gradient extremals on the distance function. In Fig. 4 another 2-dimensional example is shown, where after escaping from the local minimum, the method converges directly to the destination, because the potential field reduces the number of minima. For more details please refer to [38].

In Fig. 5 the method has been applied to a 3-dimensional environment populated by ellipsoids and parallelepipeds. Each parallelepiped is defined as $\neg(w_1 \wedge w_2 \wedge \cdots \wedge w_6)$ by six planes w_i as geometric primitives using a custom CSG library for implicit functions written in MATLAB. The computation of the gradients and Hessians is exact, using $D^2\varphi$ and the gradients and Hessians of $\beta_{ellipsoid} = (q-q_c)^{\mathrm{T}}A(q-q_c) - 1$ for ellipsoids with center q_c and

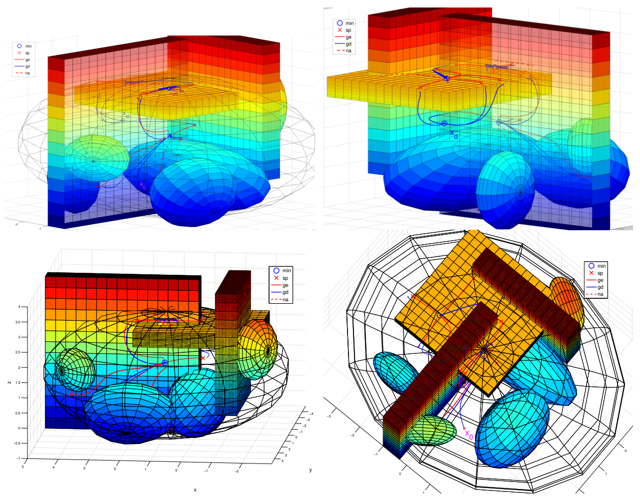

Fig. 5: Search yields 4 paths for this 3-dimensional world. The wire-frame sphere has free interior but obstacle exterior. Not all of gradient extremals followed lead to saddle-points, e.g., one ends on an obstacle boundary.

$\beta_{plane} = n^{\mathrm{T}}(q - q_p)$ for planes through q_p with unit normal n.

The computational complexity of the method depends on the number of critical points of the scalar function used, which are typically fewer for a potential compared to a distance field. Critical points of the potential arise due to obstructions of the desired flow, therefore the potential field has a critical point where "necessary" with respect to the desired destination, while distance functions have more critical points, representing all of the domain's skeleton, independently of the chosen destination. At every minimum, there are n gradient extremal paths with 2 alternative directions each, so $2n$ directions to follow. In addition, bifurcations of gradient extremals are possible and detectable by changes in certain determinants [26]. Although presently not handled, they are one aspect for future work.

In Fig. 6 a two-link manipulator is shown, with joint angles $\theta = [\theta_1, \theta_2]^{\mathrm{T}} \in [-\pi, \pi]^2$. A set of N_m sampling points $x_{mj}, j \in \mathbb{N}^*_{\leq N_m}$ is defined on the m^{th} link. Each x_{mj} is a distance $\lambda_j l_m$ from the m^{th} joint, where $\lambda_j \in (0,1)$ and l_m is the link length, with forward kinematics $x_{mj} = f(\theta) = \begin{bmatrix} l_1' \cos\theta_1 + l_2' \cos(\theta_1+\theta_2) \\ l_1' \sin\theta_1 + l_2' \sin(\theta_1+\theta_2) \end{bmatrix}$, where $l_m' = l_m$ for $m < j$, $l_m' = \lambda l_m$ if $j = m$ and $l_m' = 0$ if $j < m$. If the conjunction of the obstacle function values $b(x_{mj})$ ($\beta(x)$ previously) is taken over the manipulator, using the Rvachev function R_p, then we obtain an approximation to the implicit description of the C-obstacles. The resulting obstacle function can be expressed as $\beta(\theta) = R_p(b(x_{mj}(\theta)))$, where R_p here stands for recursive conjunction between the arguments $b(x_{mj}(\theta))$.

Its gradient $\nabla_\theta \beta(\theta)$ requires the gradients of each $b(x_{mj}(\theta))$. These are $\nabla_\theta b = \left(\frac{\partial x_{mj}}{\partial \theta}\right)^{\mathrm{T}} \nabla_x b$ and likewise computation of the Hessian $D^2_\theta \beta(\theta)$ requires computation of the Hessian matrix of each $b(x_{mj}(\theta))$. This is computed in the following way $\frac{\partial^2 b}{\partial \theta_i \partial \theta_j} = \left(\frac{\partial x}{\partial \theta_i}\right)^{\mathrm{T}} D^2_x b \frac{\partial x}{\partial \theta_j} + \nabla_x b^{\mathrm{T}} \frac{\partial^2 x}{\partial \theta_i \partial \theta_j}$,

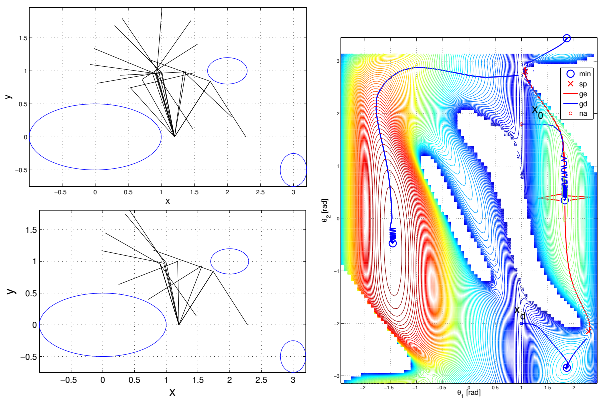
Fig. 6: Right: The C-space distance function for a 2-link robotic manipulator, constructed as an approximation of the C-space obstacles using a sampling of points on the manipulator and Rvachev conjunction. Two paths are found, comprised of gradient descents and a gradient extremal each. Left: The configuration changes through each path.

where $\frac{\partial x}{\partial \theta_i} = \begin{bmatrix} \frac{\partial x_1}{\partial \theta_i} & \frac{\partial x_2}{\partial \theta_i} \end{bmatrix}^{\mathrm{T}}$. The previous requires the forward kinematics Jacobian and Hessian, which are

$$\frac{\partial x}{\partial \theta} = \begin{bmatrix} -l_1 \sin\theta_1 - l_2 \sin(\theta_1 + \theta_2) & -l_2 \sin(\theta_1 + \theta_2) \\ l_1 \cos\theta_1 + l_2 \cos(\theta_1 + \theta_2) & l_2 \cos(\theta_1 + \theta_2) \end{bmatrix}$$

$$\frac{\partial^2 x}{\partial \theta_1^2} = \begin{bmatrix} -l_1 \cos\theta_1 - l_2 \cos(\theta_1 + \theta_2) \\ -l_1 \sin\theta_1 - l_2 \sin(\theta_1 + \theta_2) \end{bmatrix}$$

$$\frac{\partial^2 x}{\partial \theta_2 \partial \theta_1} = \frac{\partial^2 x}{\partial \theta_1 \partial \theta_2} = \begin{bmatrix} -l_2 \cos(\theta_1 + \theta_2) \\ -l_2 \sin(\theta_1 + \theta_2) \end{bmatrix} = \frac{\partial^2 x}{\partial \theta_2^2}.$$

For the kinematic Hessian of a general manipulator see [39]. More details can be found in [38].

VI. CONCLUSIONS

This paper proposed the combination of exact local valley following methods of a potential field or distance function with graph searching, in order to construct a roadmap connecting all the minima and saddle points. The method's potential is demonstrated though a number of case studies.

REFERENCES

[1] J.-C. Latombe, *Robot Motion Planning*. Kluwer, 1991.
[2] S. M. LaValle, *Planning Algorithms*. Cambridge Uni. Pr., 2006.
[3] J. Canny, *The complexity of robot motion planning*. MIT, 1988.
[4] J. Schwartz and M. Sharir, "On the "piano movers problem. II. general techniques for computing topological properties of real algebraic manifolds," *Adv in Appl Math*, vol. 4, no. 3, pp. 298 – 351, 1983.
[5] S. Karaman and E. Frazzoli, "Sampling-based algorithms for optimal motion planning," *Int J of Rob Res*, vol. 30, no. 7, pp. 846–894, 2011.
[6] O. Khatib, "Real-time obstacle avoidance for manipulators and mobile robots," *Int J of Rob Res*, vol. 5, no. 1, pp. 90–98, 1986.
[7] D. Koditschek and E. Rimon, "Robot navigation functions on manifolds with boundary," *Adv in Appl Math*, vol. 11, pp. 412–442, 1990.
[8] E. Rimon and D. E. Koditschek, "Exact robot navigation using artificial potential functions," *IEEE Trans on Rob & Autom*, vol. 8, no. 5, pp. 501–518, 1992.
[9] I. Filippidis and K. J. Kyriakopoulos, "Adjustable navigation functions for unknown sphere worlds," in *Proc 50th IEEE Conf on Dec & Contr*, 2011, pp. 4276–4281.
[10] I. F. Filippidis and K. J. Kyriakopoulos, "Navigation functions for everywhere partially sufficiently curved worlds," in *IEEE Int Conf on Rob & Autom*, 2012, pp. 2115–2120.
[11] I. Filippidis and K. Kyriakopoulos, "Navigation functions for focally admissible surfaces," in *Amer Contr Conf*, 2013, (to appear).
[12] J. F. Canny and M. C. Lin, "An opportunistic global path planner," in *Proc IEEE Int Conf on Rob & Aut*, 1990, pp. 1554–1559.
[13] E. Rimon and J. F. Canny, "Construction of C-space roadmaps from local sensory data-what should the sensors look for?" in *Proc IEEE Int Conf Rob & Aut*, 1994, pp. 117–123.
[14] E. Rimon, "Construction of C-space roadmaps from local sensory data. what should the sensors look for?" *Algorithmica*, vol. 17, pp. 357–379, 1997.
[15] J. Barraquand, B. Langlois, and J.-C. Latombe, "Numerical potential field techniques for robot path planning," *IEEE Trans on Systems, Man & Cybernetics*, vol. 22, no. 2, pp. 224 –241, 1992.
[16] W. Quapp, M. Hirsch, O. Imig, and D. Heidrich, "Searching for saddle points of potential energy surfaces by following a reduced gradient," *J of Comp Chemistry*, vol. 19, no. 9, pp. 1087–1100, 1998.
[17] F. H. Branin, "Widely convergent method for finding multiple solutions of simultaneous nonlinear equations," *IBM J of Research & Development*, vol. 16, no. 5, pp. 504 –522, 1972.
[18] C. ó'Dúnlaing, M. Sharir, and C. K. Yap, "Retraction: A new approach to motion-planning," in *Proc* 15th *annual ACM Symp on Theory of Computing*, 1983, pp. 207–220.
[19] W. Quapp, M. Hirsch, and D. Heidrich, "Following the streambed reaction on potential-energy surfaces: a new robust method," *Theoretical Chemistry Accounts*, vol. 105, pp. 145–155, 2000.
[20] J. Barraquand, B. Langlois, and J.-C. Latombe, "Robot motion planning with many degrees of freedom and dynamic constraints," in 5th *Int Symp on Robotics Research*, 1990, pp. 435–444.
[21] E. Acar, H. Choset, A. Rizzi, P. Atkar, and D. Hull, "Morse decompositions for coverage tasks," *Int J of Rob Research*, vol. 21, no. 4, pp. 331–344, 2002.
[22] H. Shashikala, N. Sancheti, and S. Keerthi, "Path planning: an approach based on connecting all the minimizers and maximizers of a potential function," in *Proc IEEE Int Conf on Rob & Aut*, 1992, pp. 2309–2314.
[23] ——, "A new approach to global optimization using ideas from nonlinear stability theory," in *Proc IEEE Int Symp on Circuits & Systems*, vol. 6, 1992, pp. 2817 –2820 vol.6.
[24] H.-D. Chiang, M. Hirsch, and F. Wu, "Stability regions of nonlinear autonomous dynamical systems," *IEEE Trans on Autom Contr*, vol. 33, no. 1, pp. 16 –27, 1988.
[25] T. Allen, J. Burdick, and E. Rimon, "Two-fingered caging of polygons via contact-space graph search," in *Proc IEEE Int Conf on Rob & Autom*, 2012, pp. 4183 –4189.
[26] E. L. Allgower and K. Georg, *Introduction to Numerical Continuation Methods*. SIAM, 2003.
[27] V. Rvachev, *Theory of R-functions and Some Applications*. Kiev: Naukova Dumka, 1982, (in Russian).
[28] V. Shapiro, "Theory of R-functions and Applications: A Primer," CS Dept, Cornell, Tech. Rep. TR91-1219, 1991.
[29] V. Shapiro and I. Tsukanov, "Implicit functions with guaranteed differential properties," in *Proc* 5th *ACM Symp on Solid Modeling & Appl*, 1999, pp. 258–269.
[30] J. Pancir, "Calculation of the least energy path on the energy hypersurface," *Collection of Czechoslovak Chemical Comm*, vol. 40, pp. 1112–1118, 1975.
[31] D. K. Hoffman, R. S. Nord, and K. Ruedenberg, "Gradient extremals," *Theoretical Chemistry Accounts*, vol. 69, pp. 265–279, 1986.
[32] D. J. Rowe and A. Ryman, "Valleys and fall lines on a Riemannian manifold," *J of Math Physics*, vol. 23, no. 5, pp. 732–735, 1982.
[33] P. Jörgensen, H. Jensen, and T. Helgaker, "A gradient extremal walking algorithm," *Theor Chemistry Accounts*, vol. 73, pp. 55–65, 1988.
[34] I. Diener, "Trajectory methods in global optimization," in *Handbook of Global Optimization*. Kluwer, 1995, pp. 649–668.
[35] M. Hirsch, "Zum reaktionswegcharakter von newtontrajektorien," Doctor Rerum Naturalium Dissertation, Universität Leipzig, 2004.
[36] W. Quapp, "A valley following method," *Optimization*, vol. 52, no. 3, pp. 317–331, 2003.
[37] P. G. Mezey, "Interrelation between energy-component hypersurfaces," *Chemical Physics Letters*, vol. 47, no. 1, pp. 70 – 75, 1977, definition of "Don Quixote" and "Sancho Panza" saddles in p.71.
[38] I. Filippidis and K. Kyriakopoulos, "Gradient extremal roadmaps," CSL, National Technical Uni Athens, Techical Report, 2012.
[39] A. Hourtash, "The kinematic hessian and higher derivatives," in *Proc IEEE Int Symp on Comp Intel in Rob & Autom*, 2005, pp. 169 – 174.