PDE-based robust robotic navigation

M. Sabry Hassouna *, Alaa E. Abdel-Hakim, Aly A. Farag

Computer Vision and Image Processing Laboratory (CVIP), Department of Electrical and Computer Engineering, University of Louisville, Louisville, KY 40292, USA

Received 30 November 2005; received in revised form 26 October 2006; accepted 5 March 2007

Abstract

In robotic navigation, path planning is aimed at getting the optimum collision-free path between a starting and target locations. The optimality criterion depends on the surrounding environment and the running conditions. In this paper, we propose a general, robust, and fast path planning framework for robotic navigation using level set methods. A level set speed function is proposed such that the minimum cost path between the starting and target locations in the environment, is the optimum planned path. The speed function is controlled by one parameter, which takes one of three possible values to generate either the safest, the shortest, or the hybrid planned path. The hybrid path is much safer than the shortest path, but less shorter than the safest one. The main idea of the proposed technique is to propagate a monotonic wave front with a particular speed function from a starting location until the target is reached and then extracts the optimum planned path between them by solving an ordinary differential equation (ODE) using an efficient numerical scheme. The framework supports both local and global planning for both 2D and 3D environments. The robustness of the proposed framework is demonstrated by correctly extracting planned paths of complex maps.

Keywords: Robotic navigation; Level set methods; Fast marching methods; Path planning; Optimum path; Skeletonization

1. Introduction

Path planning is a cornerstone in many fields of research such as robotic navigation, computer graphics, surgical simulation, and CAD applications. In robotic navigation, path planning is aimed at getting the optimum collision-free path between a starting and target locations [1,2]. The planned path is usually decomposed into line segments between ordered sub-goals or way points. In the navigation phase, the robot follows those line segments towards the target. The navigation environment is usually represented in a data structure called the “configuration space” [1]. Depending on the surrounding environment and running conditions, the optimality criterion for the path is determined. For example, in most of indoor navigation environments, the optimum path is the safest one, i.e. being as far as possible from the surrounding obstacles, whereas for outdoor navigation, the shortest path is more recommended. Several methods have been proposed for computing both the safest and the shortest paths, which can be classified as either local or global [3].

Global path planning takes into account all the information in the environment when finding the optimum path between the starting and target locations. Several methods have been proposed for global planning such as Voronoi planning [4–7], cell decomposition [8], and randomized planning [9–14]. The global approach is very time consuming in the pre-computation step, which can be accelerated either by (a) an auxiliary hardware [5], which means extra financial expenses, (b) by taking random samples from a probabilistic roadmap (PRM) in the robot’s configuration space [15–17], or (c) by using a randomized data structure such as the rapidly exploring random tree (RRT) [12–14]. Although the latter methods are fast, the key concern is how to generate sufficient samples of the environment to
capture the topology and connectivity of the configuration space.

Local planning algorithms are designed to avoid obstacles within a close vicinity of the robot. Therefore, only information about the nearby obstacles is used. The potential field-based methods such as [18–21] are the most known guidance methods, in which the configuration space is divided into a fine regular grid and then the optimum collision-free path is searched for. Different potentials are assigned to the cells of the grid. The attractive potentials are given to the cells that are close to the robot’s goal, while the repulsive potentials are assigned to the obstacles. The planned path is constructed along the most promising direction. Although the methods are fast, they can be trapped in local minima of the potential function. To avoid local minima and at the same time to get the best optimal path, an automatic planning approach has been presented in [22]. This approach depends on finding coarse clear paths and then refines them using an optimal control technique.

In this paper, we extend our recent work [23] and present a general, robust, and fast robotic path planning framework using level set methods. The framework can be applied to both planar and terrain environments, in a whole configuration space or a portion of it. The proposed framework propagates a monotonic wave front of a particular speed function from the starting location until the target is reached and then extracts the optimum planned path between them by solving an ODE using an efficient numerical scheme. The motion of the front is governed by a non-linear partial differential equation (PDE), which is computationally and efficiently solved using the fast marching method. The proposed speed function is controlled by one parameter to generate either the safest, the shortest, or the hybrid planned path.

2. Monotonically advancing fronts

Consider a closed interface $\partial \Gamma$ (that is, boundary) that separates one region from another. Assume that $\partial \Gamma$ moves in its normal direction with a known speed $F(x)$ that is either increasing or decreasing. The motion of the front is given by

$$|\nabla T(x)|F(x) = 1$$  \hspace{1cm} (1)

$$T(\partial \Gamma) = 0$$ \hspace{1cm} (2)

where, $T(x)$ is the arrival time of $\partial \Gamma$ as it crosses each point $x$. If the speed depends only on the position $x$, then the equation reduces to a non-linear first order PDE, known in geometrical optics as the Eikonal equation. The fast marching method (FMM) [24] solves that equation in one pass algorithm as follows. The numerical approximation of $|\nabla T|$ that selects the physically correct vanishing viscosity weak solution is given by Godunov [25],

$$\max(D_{ij}^b T, -D_{ij}^f T, 0)^2 + \max(D_{ij}^b T, -D_{ij}^f T, 0)^2 = \frac{1}{F_{ij}^2}$$ \hspace{1cm} (3)

where $D_{ij}^b$ and $D_{ij}^f$ are the standard backward and forward finite difference schemes, respectively, at location $(i,j)$. If $\nabla T$ is approximated by first order finite difference scheme, Eq. (3) can be rewritten as,

$$\sum_{i=1}^2 \max \left( \frac{T - T_i}{\Delta}, 0 \right)^2 = \frac{1}{F^2}$$ \hspace{1cm} (4)

where $A_1 = A_x$, $A_2 = A_y$, $T = T_{ij}$, $F = F_{ij}$, and

$$T_1 = \min(T_{i-1,j}, T_{i+1,j})$$ $$T_2 = \min(T_{i,j-1}, T_{i,j+1})$$ \hspace{1cm} (5)

The solution of Eq. (4) is given by,

- $T > \max(T_1, T_2)$: $T$ is the maximum solution of the following quadratic equation:

$$\sum_{i=1}^2 \left( \frac{T - T_i}{\Delta} \right)^2 = \frac{1}{F^2}$$ \hspace{1cm} (6)

- $T_2 > T > T_1$: $T = T_1 + \frac{A_x}{A_y}$

- $T_1 > T > T_2$: $T = T_2 + \frac{A_y}{A_x}$

The idea behind the FMM is to introduce an order in the selection of the grid points during computing their solutions (arrival times), in a way similar to the Dijkstra shortest path algorithm [26]. This order is based on the causality relationship, which states that the arrival time $T$ at any point depends only on the adjacent neighbors that have smaller values. During the evolution of the front, each grid point $x$ is assigned one of three possible tags.

1. *known*: the computed travel time at $x$ will not be changed later.

2. *narrow-band*: the computed travel time at $x$ may be changed later.

3. *far*: the travel time at $x$ is not yet computed.

The FMM algorithm can be summarized as follows: initially, all boundary points are tagged as *known*. Then, their nearest neighbors are tagged as *narrow-band* after computing their arrival time by solving Eq. (4).

1. **LOOP**: among all *narrow-band* points, extract the point with minimum arrival time and change its tag to *known*.

2. Find its nearest neighbors that are either *far* or *narrow-band*.

3. Update their arrival times by solving Eq. (4).

4. Go back to LOOP.

As a result of the update procedure (step 3), either a *far* point is tagged as a *narrow-band* or a *narrow-band* point gets assigned a new arrival time that is less than its previous value.
3. Methods

Consider the minimum cost path problem that finds the path \( C(s) : [0, \infty) \to \mathbb{R}^n \) that minimizes the cumulative travel cost from a starting point \( S \) to some destination \( E \) in \( \mathbb{R}^n \). If the cost \( U \) is only a function of the location \( x \) in the given domain, then the cost function is called isotropic, and the minimum cumulative cost at \( x \) is defined as

\[
T(x) = \min_{C_x} \int_0^L U(C(s)) \, ds
\]  

(7)

where \( C_{xx} \) is the set of the all paths linking \( S \) to \( x \). The path length is \( L \) and the starting and ending locations are \( C(0) = S \) and \( C(L) = x \), respectively. The path that gives minimum integral is the minimum cost path. In geometrical optics, it has been proven that the solution of Eq. (7) satisfies the Eikonal equation.

In this paper, we propose a new level set speed function Eq. (8) that is controlled by the parameter \( \alpha \) such that the minimum cost path between two points in the configuration space (i.e., map) is either the safest, the shortest, or the hybrid path.

\[
F(x) = \exp(\alpha \lambda(x)), \quad \alpha \geq 0
\]  

(8)

The medialness \( \lambda(x) \) is a function that assigns each safe path point \( x \) a higher weight than its non-medial neighbor points \( y \); \( \lambda(x) > \lambda(y) \). In this paper, we propose the following medialness:

\[
\lambda(x) = D(x) + \omega \left( \frac{1.0}{1.0 + |\nabla D(x)|} + \left| \min(0, \text{div} \nabla D(x)) \right| \right)
\]  

(9)

where \( D(x) \) is the Euclidean distance field, \( \omega \) is a weighting coefficient less than one, \( \alpha \) controls the path optimality, \( V \) is the gradient operator, and \( \text{div} \) is the divergence operator. In this section we will derive the value of \( \alpha \) that makes the planned path between \( S \) and \( E \) the safest path, while in the next section, we will see how \( \alpha \) can be modified to generate the other types of planned paths.

Consider the map of Fig. 1, where we need to find the safest path \( C \) between the starting location \( S \) and ending location \( E \). Assume that \( S \) is a point source that transmits a wave front, whose speed is given by Eq. (8). Our goal is to distinguish the safe path points by making them the locus of the front points of maximum positive curvatures. Assume that \( O \) belongs to \( C \) and let \( \delta_i \) be the 8-connected non-medial neighbors of \( O \); \( \lambda(O) > \lambda(b_j) \). Assume that the front reaches \( O \), \( E \), and \( b_j \) at the times \( t_0 \), \( t_0 + \Delta t_E \), and \( t_0 + \Delta t_j \), respectively. In order to make \( E \) the front point of maximum positive curvature, then the wave must reach \( E \) before reaching \( b_j \) even if \( d(O,E) > d(O,b_j) \), where \( d(., .) \) is the Euclidean distance between two points. Therefore

\[
t_O < t_{b_j}
\]  

(10)

\[
d(O,E) \frac{F(O)}{F(b_j)} < d(O,b_j)
\]  

(11)

In the worst case when \( C \) makes the largest slope; 45°, the wave front will travel the longest distance to reach \( E \) than reaching \( b_j \); \( d(O,E) = \sqrt{2} \) and \( d(O,b_j) = 1 \). Therefore,

\[
\frac{\sqrt{2}}{\exp(\alpha \lambda(E))} < \frac{1}{\exp(\alpha \lambda(b_j))}
\]  

(12)

Let \( \lambda(E) = h \) and \( \lambda(b_j) = h - \delta_i \), where \( \delta_i \) is the difference in medialness of two neighboring locations. Then, the value of \( \alpha \) that guarantees the safest path is given by,

\[
\alpha > \frac{\ln(\sqrt{2})}{\delta_i}
\]  

(13)

The parameter \( \delta_i \) can be computed in the worst case as

\[
\delta_i = \min_{y} \{ \lambda(x) - \lambda(y) \mid y \in \eta_x, \lambda(x) > \lambda(y) \}
\]  

(14)

where \( \eta_x \) is the 8-connected neighbors of \( x \).

4. Single optimum path extraction (SOPE)

4.1. Safest path extraction

For isotropic propagation using the FMM, where the cost is only a function of the position, the fastest traveling is always along the direction perpendicular to the wave

![Fig. 1. The front must be faster at safe path points.](image1)

![Fig. 2. The gradient of the distance map.](image2)
Since the propagating fronts are level sets, then the direction of the gradient at each point is normal to the front. Recall that the front is faster at the safest path points, then $C$ can be found by backtracking along the gradient of $T(x)$, which is the solution of the following ODE:

$$\frac{dC(s)}{ds} = -\frac{\nabla T}{|\nabla T|} \text{ given } C(0) = E$$

(15)

where $C(s)$ traces out the path. Let $S$ and $E$ be the starting and ending points of $C$, then backtracking continues from $C(0) = E$ until $S$ is found. The ODE has been solved using the second order Runge–Kutta method, where the total cumulative error is on the order of $O(h^3)$, where $h$ is the integration step, which is set to 0.5. The point $S$ is guaranteed to be found because the field is monotonically increasing from $S$ to $E$. Note that this technique extracts the planned path with sub-pixel accuracy even if the grid is discrete.

4.2. Shortest path extraction

For $\alpha = 0$, the propagating front is moving with a unit speed and hence resulting in an isotropic propagation (circular iso-contours). Therefore, following the gradient descent will result in the shortest path as explained in [24].

4.3. Hybrid path extraction

For $0 < \alpha < \alpha_c$, the extracted path is the hybrid one. It is much safer than the shortest path, but less shorter than the safest path. Therefore, the hybrid path strikes a balance in terms of length and safeness, when compared with other planned paths.

5. Medialness function $\lambda(x)$

$\lambda(x)$ is a scalar function that distinguishes medial points from others. It consists of three different medialness terms: the smoother $\lambda_1(x)$, the salient $\lambda_2(x)$, and the absolute average outward flux $\lambda_3(x)$.

5.1. Smoother medialness function $\lambda_1(x)$

This medialness assigns each point in the map its minimum distance from the boundary as given by Eq. (16). Although $\lambda_1(x)$ does not provide much distinction between medial and non-medial points, it provides a smooth transition among them because the distance is monotonically increasing from the boundary of the map towards its center as shown in Fig. 3(a).

$$\lambda_1(x) = D(x)$$

(16)

$D(x)$ is computed as follows. The map’s boundary is initialized at zero travel time; $T = 0$. Then, a unit speed front is...
propagated from the boundary towards the center of the map. The front motion is governed by Eq. (1), whose solution is the desired distance field.

5.2. Salient medialness function $\lambda_2(x)$

Since for a moving front with a unit speed, the solution of Eq. (1) is the Euclidean distance field, then $|D(x)| = 1.0$ except at local maximum points (e.g., medial points) where the gradient is theoretically zero. Since $D(x)$ is not differentiable at those points, it is convolved with a Gaussian kernel of a small variance to be differentiable everywhere. This medialness is called salient because it identifies only strong medial points with sufficiently small gradient as shown in Fig. 3(b).

$$\lambda_2(x) = \frac{1.0}{1.0 + |\nabla D(x)|}$$

5.3. Absolute average outward flux medialness function $\lambda_3(x)$

The gradient of a scalar function points in the direction where the function is increasing rapidly. Since the safest path consists of local maximum points, $\nabla D(x)$ is a vector field that points towards the center of the map. If we assume that this vector field represents a velocity field, then from fluid mechanics, the divergence of this field at a point measures the net outflow of the vectors in an infinitesimal area centered at the point. The points of the safest paths are sink points with strong negative divergence because the inflow is much higher than the outflow. On the other hand, the boundary points are source points with strong positive divergence because the outflow is much higher than the inflow. Finally, for all non-medial points, the inflow is nearly equal to the outflow and the divergence is nearly zero. This concept is illustrated in Fig. 2.

Unfortunately the divergence theorem does not apply at medial points because the vector field $\nabla D(x)$ is discontinuous. In [28], the divergence theorem has been modified to handle such limitation and to extract centerlines of 2D shapes. They defined the divergence at a point as the limiting behavior of the average outward flux through a neighborhood $R$ as it shrinks to a point, normalized by the area of $R$. In this paper, we propose the following medialness:

$$\lambda_3(x) = \min(0, \text{div}(|\nabla D(x)|))$$

Although $\lambda_3(x)$ identifies both salient and non-salient medial points, the final skeleton is thick as shown in Fig. 3(c).
By augmenting both $\lambda_2(x)$ and $\lambda_3(x)$, we get a strong distinction between medial and non-medial points, while preserving the non-salient ones. Since the composite medialness function is very strong, only a small fraction of it is enough to distinguish medial points from others. Therefore, $\omega$ is set to a small value (i.e., 0.1). If we propagate a front from a source point using that medialness, the resulting fronts will be very sharp at medial points because all non-medial points are moving with nearly a constant low speed, while medial ones are moving with very high speed. To solve this problem, we augment $\lambda_1(x)$ as a smoothing term to the weighted sum of both $\lambda_2(x)$ and $\lambda_3(x)$. In Fig. 3(d), we show that by augmenting the three medialness functions, we get an enhanced medialness that highly distinguishes medial points from others, when compared with that of Fig. 3(a). Each $\lambda_i(x)$ is normalized from 0.0 to 1.0, and the combined $\lambda(x)$ is also normalized from 0 to 1.0.

6. Global path planning for planar environment

In global planning, the environment is assumed to be known a priori. Let $S$ and $E$ be the starting and target locations, respectively. In order to extract the optimum
planned path between $S$ and $E$, we propose two methods. The first method is general and can be used to extract any optimum path. On the other hand, the second method is limited to the extraction of safest paths, but is quite fast when compared with the first method.

In the first method, we compute the proposed medialness function with respect to the entire map, then we propagate a wave from $S$ until $E$ is reached, and finally, we extract the planned path as described in Section 4.

The second method is optimal if the robot is performing a systematic job (for example, returning borrowed books to their locations in a library) that requires the extraction of different safest paths in the map. As a consequence, it is better to extract the entire safest paths of the map (skeleton) and store them for later use. The method works as follows: initially, we compute the entire safest paths of the map using our recent work [29]. Then, we map both $S$ and $E$ to the nearest locations on the skeleton (i.e., $S'$ and $E'$). Finally we determine the safest path that belongs to the skeleton and connects $S'$ to $E'$. The mapping is illustrated in Fig. 4, where we propagate two wave fronts from $S$ and $E$ until they intersect the skeleton at $S'$ and $E'$, respectively. Then, we find the line segments that connect $S$ with $S'$ and $E$ with $E'$ through the same backtracking mechanism. If a moving obstacle is detected by the robot sensors, the robot recomputes the medialness of the entire map, and then extracts a new optimum path from its current location to the same target location.

### 7. Local path planning for planar environment

In local planning, the robot only knows some information about the environment $\Omega$, normally in its close vicinity. Let the area of the map (i.e., environment) that is visible to the robot be denoted by $\Gamma$, while its boundary is denoted by $\partial \Gamma$. As the robot moves from $S$ to $E$, its visible area is updated and its next position $C$ is considered the new starting point. The proposed method assumes the following. If the target location $E$ is inside $\Gamma$, the next position $C = E$. Otherwise, $C \in \partial \Gamma$ such that,

$$
C = \arg \min_x \{ d(x, E) | x \in \partial \Gamma, \ x \notin \partial \Omega \}
$$  \hspace{1cm} (19)

where $\partial \Omega$ is the boundary of the environment. The optimum path segment between $S$ and $C$ at time $t$ is computed as described in Section 4. This concept is illustrated in Fig. 5(a), where the robot is allowed to move from point $S$ to $E$ looking for a particular optimum path. The robot is represented by a white square, while its visible area is represented by a closed contour. The Figs. 5(b and c) show the position of the robot in conjunction with $C$ at different times. In Fig. 6, we show the computed proposed medialness of $\Gamma$ at different positions. In Figs. 7–9, we show the computed shortest, hybrid, and safest segment paths within $\Gamma$. In Fig. 10, we show the final computed optimum paths from $S$ and $E$ by concatenating the optimum path segments at different robot positions. Although the difference between the shortest and either the hybrid or the safest paths

![Fig. 12. A map with different safest paths that begin from different starting locations but ends with the same target location.](image-url)
is apparent, there is no much difference between the safest and the hybrid paths, because in local planning, $C$ is small and both the safest and hybrid path segments are computed locally with respect to $C$.

8. Local path planning for terrain environment

The local path planning technique for planar environment can be applied as well for terrain environment. The idea is to replace the proposed medialness function for planar environment Eq. (9) with

$$\lambda(x) = \frac{1}{1 + d(x)}$$

(20)

where $d(x)$ is the height of each point on the terrain surface. The proposed approach is used to compute both the shortest ($\alpha = 0$) and minimum energy path $x > x_c$ between the starting and target locations assuming that the height map of the environment is known a priori.

9. Experimental results

We have tested the proposed framework using several maps of different complexity against local and global path planning for both planar and terrain environments. In Fig. 11, we show the shortest, hybrid, and safest paths by the proposed global planning technique for two different maps. It is clear that the hybrid path is much safer than the shortest path but less shorter than the safest path. In Fig. 12, we show a map with different safest paths that start from different locations but end with the same target location. In Fig. 13, we show the shortest and safest paths that are computed by the proposed local path planning method. In Fig. 14, we show the shortest and the minimum energy (safest) paths between two locations on a terrain surface. In Fig. 15, we show that the proposed framework can be extended easily to 3D planning. In Table 1, we compare the proposed framework with several well known path
Table 1
Comparison between the proposed method and some well known techniques

<table>
<thead>
<tr>
<th>Method</th>
<th>Local planning</th>
<th>Global planning</th>
<th>Safest path</th>
<th>Shortest path</th>
<th>Hybrid path</th>
<th>Local min. stuck</th>
</tr>
</thead>
<tbody>
<tr>
<td>Proposed model</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Virtual forces [19]</td>
<td>✓</td>
<td>✓</td>
<td></td>
<td>✓</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Heuristic search 4*[8,1]</td>
<td>✓</td>
<td>✓</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Voronoi planning [6]</td>
<td>✓</td>
<td>✓</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Optimal control [22]</td>
<td>✓</td>
<td>✓</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Skeletons [7]</td>
<td>–</td>
<td>–</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

planning techniques, where we can see that our contribution is more general.

10. Conclusion

In this paper, we have presented a general, robust, and fast path planning framework for robotic navigation using level set methods. We have proposed a new level set speed function such that the minimum cost path between the starting and target locations in the environment is the optimum planned path. The main idea of the framework is to propagate a monotonic wave front with a particular speed function from the starting location until the target is reached and then extract the optimum planned path between them. The framework can be applied to both 2D and 3D environments. It generates a collision-free optimum path for local and global planning. Finally, optimum planned paths can be controlled by a single parameter in order to follow the safest, the shortest, or the hybrid path.

Acknowledgement

This work has been supported by NASA Grant number NCC5-571.

References