Method of evolving junctions: A new approach to optimal path-planning in 2D environments with moving obstacles

Wuchen Li 1, Shui-Nee Chow 1, Magnus Egerstedt 2, Jun Lu 1 and Haomin Zhou 1

Abstract
We propose a novel algorithm to find the global optimal path in 2D environments with moving obstacles, where the optimality is understood relative to a general convex continuous running cost. By leveraging the geometric structures of optimal solutions and using gradient flows, we convert the path-planning problem into a system of finite dimensional ordinary differential equations (ODEs), whose dimensions change dynamically. Then a stochastic differential equation (SDE) based optimization method called intermittent diffusion is employed to obtain the global optimal solution. We demonstrate, via numerical examples, that the new algorithm can solve the problem efficiently.

Keywords
Path-planning, dynamic environment, optimal control, constraints, stochastic differential equations

Introduction
Finding optimal paths in dynamic environments has attracted significant attention in the robotics community (See Latombe [1990], LaValle [1999], Lu at al. [2015], Jur [2007] and the references therein). The problem is to navigate a robot from a starting point to a destination, avoiding collisions with moving obstacles while minimizing a cost functional, such as the energy consumption.

Even in static environments, optimal path planning is a challenging task. For example, finding the shortest path for a point robot in 3D with polyhedral obstacles is NP hard, see Canny and Reif [1987]. Recently, many path-centric algorithms have been introduced in two categories: (i) Grid based planners, such as A*, D* and D* Lite. See Koening and Likhachev [2002], Koening et al [2004], Koening and Likhachev [2005], Stentz [1994], Stentz [1995], Ferguson et al [2005], Likhachev et al [2008]. (ii) Sample based planners, such as PRM* (Probabilistic Road Maps) and RRT* (Rapidly exploring Random Tree). See Fiorini and Zvi [1998], Park et al. [2013], Karaman and Frazzoli [2011], Kavraki et al. [1996], LaValle [1999], Hsu [2002], Zacker et al. [2007] for more information. In this paper, we take a fundamentally different approach to design a path-centric algorithm to find the optimal solution for a point robot in dynamic environments.

In the existing literature, special cases, such as optimality being restricted to minimal arrival time and obstacles being limited to polygons or disks, have been studied when environmental dynamics are predictable. For example, if the dynamic environment only contains moving polygons or disks with constant speed, an algorithm has been proposed in Fujimura and Samet [1993] with complexity $O(n^2 \log n)$, where $n$ is the total number of vertices. Key to this method is that when the minimal arrival time is the objective, a robot must traverse along straight lines and go from one vertex to another. This observation reduces the minimal arrival time problem into a shortest path problem on a graph with finitely many nodes, so that well known algorithms can be applied directly. A similar idea was used in Jur and Mark [2008], which considers environments consisting of expanding or shrinking disks of constant speed. Other studies aiming to achieve minimal arrival time have been reported in Narayanan et al [2012], Phillips and Likhachev [2011], Nieuwenhuisen et al [2007].

In this paper, we consider a 2D dynamic environment in which the motions of obstacles are known a priori. The cost functional is quadratic in speed variable. The arrival time may or may not be part of the cost functional. The obstacles can have general shapes with their boundaries being characterized by unions of a finite number of convex or concave curves. In this case, the optimal path-planning problem can be posed in the framework of optimal control, see Bobrow [1988, 1985], Shin and Mckay [1985]. Hence it can be solved by three general numerical methods. (1) State space. One solves Hamilton-Jacobi-Bellman equations, see Yershov and Frazzoli [2015], which deals with nonlinear partial differential equations; (2) Indirect method. One employs the Pontryagin’s maximal principle, which leads to a system of boundary value ODEs Ghasemi et al [2011]; and (3) Direct method. One discretizes the state and control variables directly, and then finds the path by

---

1 School of Mathematics, Georgia institute of technology
2 School of Electrical and Computer Engineering, Georgia institute of technology.

The work is partially supported by NSF Award DMS 1419027 and ONR Award N000141310408.

Corresponding author:
Wuchen Li, 686 cherry street, Atlanta, Georgia, 30332, USA.
Email: wlli83@gatech.edu

*The path-centric algorithm finds the minimal cost path from the initial position to its destination. It contrasts with the policy-centric algorithm which computes the optimal cost-to-go function by solving Hamilton-Jacobi-Bellman equation, see Yershov and Frazzoli [2015].*
solving a large dimensional optimization, see \cite{Posa2014}. However, the dynamic environment introduces time dependent continuous constraints, and this is hard to treat numerically in general, see \cite{Lu2001}, especially when the number of moving obstacles is large.

Following the idea in \cite{Chow2016,Lu2014}, we adopt a recently developed algorithm, namely method of evolving junctions (MEJ). The method is motivated by the following facts: All local and global optimal paths share a similar geometric structure called “Separability”, meaning the path can be partitioned into a finite number of segments over which the constraints are either active (robot moving along the boundary of an obstacle, and/or at its maximum speed) or inactive (robot moving freely). We call the partition points junctions. Using those junctions, we can reduce the optimal control, which typically cast as an infinite dimensional problem in Banach space, to a finite dimensional optimization problem. Such a reduction allows us to find global optimal path(s) by initial value problems of SDEs. Compared to existing methods, the new algorithm has the following advantages: (i) it leverages the geometric structure of the dynamic environments, and transfers the optimal control into a finite dimensional optimization without compromising the accuracy of the path. (ii) the main computation is to solve initial value stochastic differential equations (SDEs) derived from the finite dimensional optimizations, which is easy to implement numerically. (iii) it can find the global optimal path with a high probability, and also obtain a series of local minimizers.

The paper is arranged as follows: In Section 2, we set up the mathematical description of the problem. In Section 3, we present the new algorithm. Several experiments are shown in Section 4.

Problem Description

We shall consider a predictable environment with \( N \) obstacles moving in \( \mathbb{R}^2 \). We assume that the workspace of the robot is the same as the configuration space. Initially, each obstacle is represented by a connected compact set \( P_k, k \in \{1, \cdots, N\} \), whose boundary \( \partial P_k \) is a union of a finite number of convex and concave curves, which satisfy Lipschitz condition with Lipschitz parameter bounded by a constant.

And we assume that each obstacle \( P_k \) moves at a constant velocity \( v_k \). Thus the dynamic obstacle can be expressed by a time-dependent set given by

\[
P_k(t) = \{ x + v_k t \mid x \in P_k \}, \quad k \in \{1, \cdots, N\}.
\]

We also assume that a robot is considered as a point, whose path is denoted by a curve \( \gamma(t) : [0, T] \to \mathbb{R}^2 \). Here \( T \) is the terminal time, which is unknown in general.

We call \( \gamma(t) \) a feasible path if

(i) the robot moves from a starting point \( X \) to a target point \( Y \),

\[
\gamma(0) = X, \quad \gamma(T) = Y;
\]

(ii) the robot avoids collisions with all moving obstacles during its course,

\[
\gamma(t) \in \mathbb{R}^2 \setminus \bigcup_k P_k(t),
\]

for any \( 0 \leq t \leq T \);

(iii) the robot travels with a speed restriction,

\[
al(\gamma(t_1), \gamma(t_2)) \leq v_m(t_2 - t_1),
\]

for any \( 0 \leq t_1 \leq t_2 \leq T \), where \( v_m \) is a positive constant indicating the maximal allowable speed for the robot, and \( al(\gamma(t_1), \gamma(t_2)) \) is the arc length of the path between the two points.

Denote the set of all feasible paths by

\[
A = \{ \gamma(t) \in AC[0, T] \mid (i), (ii), (iii) holds \},
\]

where \( AC[0, T] \) represents the set of absolutely continuous curves.

To model the energy consumption of the robot as well as the arrival time, we introduce the following cost functional,

\[
J(\gamma) = \int_0^T L(t, \gamma(t), \dot{\gamma}(t))dt,
\]

where \( L(t, \gamma, \dot{\gamma}) = \dot{\gamma}^2 + c, \ c > 0 \) is a given constant. The reason to use a quadratic function here is that the robot expends more energy for fast speeds while stalled (or slow) motion is also deemed inefficient.

The goal is finding a global optimal path \( \gamma_{opt} \) to attain

\[
\gamma_{opt} = \arg \min_{\gamma \in A} J(\gamma).
\]

For convenience, we adopt a level set expression to represent (ii). It is to define \( \phi_k \) as a signed distance function between a point \( y \) and the obstacle boundary set at time \( t \)

\[
\phi_k(t, y) = \begin{cases} 
\text{dist}(y, \partial P_k(t)), & \text{if } y \in P_k(t); \\
-\text{dist}(y, \partial P_k(t)), & \text{if } y \in \mathbb{R}^2 \setminus P_k(t), 
\end{cases}
\]

with \( \text{dist}(y, \partial P_k(t)) = \inf_{x \in \partial P_k(t)} \text{dist}(x, y) \). Then the second requirement-no collision with obstacles-can be rewritten as

\[
\phi_k(t, \gamma(t)) \leq 0, \quad t \in [0, T].
\]

As mentioned in the introduction, the above problem can be reformulated as an optimal control problem.

\[
\min_{\gamma,v} \int_0^T v^2 dt + cT
\]

where the state \( \gamma(t) \) and control \( v(t) \) are subject to

\[
\dot{\gamma} = v, \quad t \in [0, T]; \quad \gamma(0) = X, \quad \gamma(T) = Y;
\]

\[
\phi_k(t, \gamma(t)) \leq 0, \quad \|v(t)\| \leq v_m.
\]

The Algorithm

To solve \((1)\), we adopt a new computational framework, called method of evolving junctions (MEJ), which aims at rewriting \((1)\) as a finite dimensional optimization problem by leveraging the geometric structure of the optimal path given in the following definition.

\[\text{Comparing with } \text{Lu et al.} [2014], \text{where the authors only deal with the shortest path problem (time independent), we find a similar method for a broader class of optimal path planning problems, in which the time variable is built in.}\]
Lemma 3. The optimal path \( \gamma_i(t) \) connecting a junction pair \((\tilde{x}_i, \tilde{x}_{i+1})\) with active constraints is a line with constant speed \( v_m \).

(b) \( \gamma_i(t) \) is a geodesic on the moving obstacle with a relative constant speed.

(c) \( \gamma_i(t) \) is a geodesic on the moving obstacle with the maximal speed \( v_m \).

Proofs of the lemmas can be found in the Appendix.

From Lemmas 2 and 3, we know that \( \gamma_i \) can be rewritten as a function of a junction pair. So we can represent (1) by a finite dimensional optimization of junctions. More precisely, the cost functional of \( \gamma_i \) can be reformulated as,

\[
J(\tilde{x}) = \sum_{i \in I} J_i(\tilde{x}) + \sum_{i \in A} J_i(\tilde{x}).
\]

where \( \tilde{x} = (x_0, \ldots, x_{N+1}) \), \( J_i(\tilde{x}) \) is the cost of \( \gamma_i \) solving either (2) or (3), i.e.

\[
J_i(\tilde{x}) = \int_{t}^{t+1} L(t, \gamma_i(t), \dot{\gamma}_i(t)) dt.
\]

If \( \gamma_i \) solves (2), we say \( i \in I \), otherwise, \( i \in A \).

Moreover, \( \gamma_i \) must not violate the constraints of (1). We define

\[
V(\tilde{x}) = \max_{i \in I \cup A} \max_{t_i \leq t \leq t_{i+1}} \phi(t, \gamma_i(t)) = 0
\]

to ensure that \( \gamma_i(t) \) satisfies condition (ii), and

\[
S(\tilde{x}) = \max_{i \in I \cup A \cup \{0\}} \|\gamma_i(t)\| \leq v_m
\]

for condition (iii).

With the above reformulations, (1) is rewritten as

\[
\min_{\tilde{x}} J(\tilde{x}), \quad \text{s.t. } V(\tilde{x}) = 0, \quad S(\tilde{x}) \leq v_m.
\]

And this is the optimization we solve numerically. In this way, we transfer the optimal control problem (1) to a finite dimensional optimization (4), for which we gain a tremendous dimension reduction.

To compute (4), we adopt a global optimization technique, called Intermittent Diffusions (ID), see Chow et al. (2012).

The key idea of ID is to add white noise (diffusion) to the gradient flow (gradient descent method) of \( J(\tilde{x}) \) intermittently.

Namely, we solve the following SDEs on a constrained set

\[
d\tilde{x} = P_\alpha [-\nabla J(\tilde{x}) dt + \sigma(\theta) dW(\theta)],
\]

where \( \tilde{x} = (\tilde{x}_0, \tilde{x}_1, \ldots, \tilde{x}_N, \tilde{x}_{N+1}) \), \( \theta \) is an artificial time variable different from \( t \), \( W(\theta) \) is the standard Brownian motion, and \( P_\alpha \) is the orthogonal projection onto the tangent plane at \( \tilde{x} \) for the constraint set in (4).

If we denote the set of feasible directions at \( \tilde{x} \) as

\[
F(\tilde{x}) = \{ q | \nabla V(\tilde{x}, \tilde{x}_{i+1}) \cdot q = 0, \quad \nabla S(\tilde{x}, \tilde{x}_{i+1}) \cdot q \leq 0, \quad \|q\| = 1 \},
\]

then \( P_\alpha(\tilde{p}) \) is defined by

\[
- \frac{P_\alpha(\tilde{p})}{\|P_\alpha(\tilde{p})\|} = \arg \min_{q \in F(\tilde{x})} q \cdot \tilde{p}, \quad \|P_\alpha(\tilde{p})\| = \min_{q \in F(\tilde{x})} |q \cdot \tilde{p}|.
\]

The above two equations are the standard projection operator in constrained optimization, where the first and second
equation give direction and magnitude of the projected vector respectively. For convenience, we denote $\nabla c J(\tilde{x}) := P_z (\nabla J(\tilde{x}))$, which is the projected gradient vector.

The function $\sigma(\theta)$ is piecewise constant, controlling the amount of noise added intermittently. More precisely, $\sigma(\theta) = \sum_{j=1}^m \sigma_j \chi_{[S_j, T_j]}(\theta)$, where $\{[S_j, T_j]\}_{j=1}^m$ are disjoint intervals, and $\chi_{[S_j, T_j]}$ is the characteristic function on $[S_j, T_j]$. If $\sigma(\theta) = 0$, we obtain the projected gradient flow, whose solution converges to a minimizer. If $\sigma(\theta) \neq 0$, (6) is a SDE, whose solution has a positive probability to escape any attraction basins of minimizers. The theory of ID suggests that solutions of SDEs visit the global minimizers with probability arbitrarily close to 1, if $|T_j - S_j|$ is large enough. We illustrate how the ID algorithm works by the following theorem in [Chow et al. (2012)].

**Theorem 4.** Given any real number $\delta > 0$, there exists constants $\sigma > 0$, $\tau > 0$, and integer $m > 0$, such that if $T_i - S_i > \tau$, $\sigma_i < \sigma$ (for $i = 1, \ldots, m$), then equation (6) finds the global minimizer of $H$ with probability at least $1 - \delta$.

To solve SDE (5) numerically, we discretize it by Euler-Maruyama scheme:

$$\tilde{x}^{k+1} = \tilde{x}^k + \eta J(\tilde{x}^k) h + \sigma_k \sqrt{h} \xi_k,$$

where $h$ is the step size for the gradient descent method. What this means is that we sample a set of standard Gaussian random variables $\xi_k \sim N(0, 1)$, and add them to each projected gradient descent step. Coefficient $\sigma_k$ is chosen as a piecewise constant, which represents that noises are added intermittently. The turning parameter $m$ is the number of intervals that we turn on and off the noise. Heuristically, the larger the $m$ is, the larger the probability of finding the global minimizer. One can prove mathematically that the global optimal solution can be achieved with the probability 1 if $m$ tends to infinity. In practice, this is not the case. In fact, we find a set of local minimizers and pick the one with the smallest objective value as the best solution.

It is also worth mentioning that the number and locations of junctions may vary when solving (5). We propose a heuristic way to deal with appearing and disappearing junctions. We add new junctions when a straight line trajectory $\tilde{x}_1, \tilde{x}_2$ intersects with a moving obstacle. For example, insert the intersection points (including both position and time), denoted as $\tilde{y}_j$, into the sequence of junctions, as in Figure 2.

When two straight segments $\tilde{x}_1, \tilde{x}_2$ and $\tilde{x}_2, \tilde{x}_3$ share a common junction $\tilde{x}_2$, as depicted in Figure 3, a smaller cost functional can be obtained by connecting $\tilde{x}_1, \tilde{x}_3$ directly if the path $\tilde{x}_1, \tilde{x}_3$ does not intersect another moving obstacle during its course. We remove $\tilde{x}_2$ from the set of junctions.

We summarize the steps into the following algorithm.

**Algorithm 1.**

**Input:** Constraint $\phi_k, \psi_m$,
starting and ending points $X$ and $Y$,
running cost $L$,
number of intermittent diffusion intervals $m$.

**Output:** The optimal set: $\gamma_{\text{opt}}$ and junctions.

1. Initialization. Find an initial path $\gamma^{(0)}$; the initial junctions are intersections of $\gamma^{(0)}$ with the moving obstacles;
2. Select duration of diffusion $\Delta T_i, l \leq m$;
3. Select diffusion coefficients $\sigma_i, l \leq m$;
4. for $l = 1 : m$
5. $\gamma^{(l)} = \gamma^{(l-1)}$;
6. for $j = 1 : \Delta T_i$
7. Find $\nabla c J(\gamma^{(l)})$;
8. Update $\gamma^{(l)}$ according to (6) with $\sigma(\theta) = \sigma_l$;
9. Remove junctions from or add junctions to $\gamma^{(l)}$ when necessary;
10. end
11. while $\|\nabla c J(\gamma^{(l)})\| > \epsilon$
12. Update $\gamma^{(l)}$ according to (6) with $\sigma(\theta) = 0$;
13. end
14. end
15. Compare $J(\gamma^{(l)}), l \leq m$ and set $\gamma_{\text{opt}} = \arg\min_{\gamma^{(l)}} J(\gamma^{(l)})$;

**Remark 1.** Here $m$ represents the number of artificial time intervals, which is different from the number of junctions $N$. $m$ is used for adding or removing white noises in gradient descent steps, while $N$ is changed when adding or removing junctions is needed for keeping the path feasible. In some situations, $N$ can be fixed, see a simple case in the next section.

**Theorem 5.** Algorithm 1 solves problem (1) almost surely.

The proof directly follows from (4) and Theorem 4. However, there is a distinction between the theoretical guarantees and numerical implementation of the algorithm. In practice, we often choose a suitable upper bound for the number of time intervals in intermittent diffusion.

The proposed algorithm solves 2D problem efficiently. This is due to the fact the geodesic of 2D obstacles has an analytical solution. For problems with higher dimensions, the geodesic is usually not simple to calculate. This difficulty may prevent us from obtaining analytic or semi-analytic formulas of $\gamma_i(t)$ in (3) and consequently $J_i(\tilde{x})$.
for the corresponding constrained segment of the trajectory. That limits the application of the proposed algorithm in those challenging cases. On the other hand, one may use polygons to approximate (piecewise linearly) the boundaries of obstacles. In this way, the geodesic on the plane is still easy to calculate analytically in general. So the proposed method can be applied, while the accuracy of the optimality is limited by the approximation error. In addition, our algorithm find the global optimal solution in a probability sense. In practice, the algorithm returns a set of minimizers, and we pick the one with the smallest cost functional as the global solution. The complexity of the algorithm scales linearly with the number of obstacles \( n \), and it depends on the approximation error tolerance \( \epsilon \) and the probability tolerance \( \delta \). In fact, one can prove that the complexity is of order \( O(n \log(1/\epsilon) \log(1/\delta)) \) (see the proof in [Lu at (2014)].

**Numerical experiments**

In this section, we illustrate the performance of the algorithm by several numerical experiments.

**A simple case**

We use the following example to illustrate, step by step, how to implement our algorithm.

Consider an environment containing one obstacle \( P_1 \), which is a disk initially centered at \((0, 0)\) and with radius 1, moving at a constant velocity \( v_1 = (0, -0.1) \). The starting and ending points are \( X = (-2, 0), Y = (2, 0) \) respectively. The terminal time is fixed at \( T = 1 \). In addition, we assume that there is no speed constraint. Then the optimal control problem (1) becomes

\[
\min \left\{ \int_0^1 \gamma^2 dt : \gamma(0) = X, \gamma(1) = Y, \phi_1(t) \leq 0 \right\},
\]

where \( \phi_1(t, \gamma(t)) = 1 - \|\gamma(t) - v_1 t\| \).

**Proposition:** There are at most two junctions on \( P_1 \).

The proof of this proposition is given in the Appendix. To represent the two junctions, we denote \( \alpha(u) = (\cos u, \sin u) \) as the parametrization of \( \partial P_1 \), where \( u \) is an arc-length parameter. Hence a junction \( \tilde{u}_i = (\tilde{u}_1, \tilde{u}_2) \) represents a point on the moving obstacle, whose position is denoted as \( R(\tilde{u}_i) = \alpha(u_i) + v_1 t_i, i = 1, 2 \).

Two junctions \( \tilde{u}_1 \) and \( \tilde{u}_2 \), denoted as \( \tilde{u} = (\tilde{u}_1, \tilde{u}_2) \) for convenience, partition the trajectory into three segments

\[
\gamma(t) = \gamma_0(t) \cdot \gamma_1(t) \cdot \gamma_2(t),
\]

where \( \gamma_1(t) \) is the optimal path along \( \partial P_1 \), and \( \gamma_0(t), \gamma_2(t) \) are the optimal paths in the free space respectively. From Lemma 1, we know that \( \gamma_0, \gamma_2 \) are straight lines with constant speed,

\[
\gamma_0(t) = \frac{R(\tilde{u}_1) - X}{t_1} t + X, \quad t \in [0, t_1]
\]

and

\[
\gamma_2(t) = \frac{Y - R(\tilde{u}_2)}{T - t_2} (t - t_2) + R(\tilde{u}_2), \quad t \in [t_2, T]
\]

and the cost is

\[
J_0(\tilde{u}) = \frac{\|R(\tilde{u}_1) - X\|^2}{t_1}, \quad J_2(\tilde{u}) = \frac{\|R(\tilde{u}_2) - Y\|^2}{T - t_2}.
\]

From Lemma 2, \( \gamma_1(t) \) is a path along the boundary \( \partial P_1 \) with a constant speed relative to the moving obstacle, and it can be written as

\[
\gamma_1(t) = \alpha(u(t)) + v_1 t,
\]

where \( u(t) \) is the relative position on the boundary. There are two possibilities for \( u(t) \), the clockwise and counter-clockwise, and we denote them as \( u_+ \) and \( u_- \) respectively. The constant speed suggests that \( u_+ = u_1 + (u_2 - u_1)/(t_2 - t_1)(t - t_1) \), and \( u_- = 2\pi - u_+ \). Then \( u(t) \) takes the one with lower cost, i.e.

\[
u(t) = \arg \min_{\{u_+, u_-\}} \{ J_1(u_+), J_1(u_-) \},
\]

where

\[
J_1(u_+) = \frac{(u_2 - u_1)^2}{t_2 - t_1} + (\sin u_1 - \sin u_2) + v_1^2(t_2 - t_1),
\]

and

\[
J_1(u_-) = \frac{(2\pi - (u_2 - u_1))^2}{t_2 - t_1} + (\sin u_2 - \sin u_1)
\]

\[+ v_1^2(t_2 - t_1).\]

Using the junctions, we transfer the optimal control into a four dimensional optimization:

\[
\min_{\{(u_1, t_1, u_2, t_2)\}} \sum_{i=0}^2 J_i(\tilde{u})
\]

such that

\[
\max_{0 \leq t \leq t_1} \phi_1(t, \gamma_0(t)) = \max_{t_2 \geq t \leq 1} \phi_1(t, \gamma_2(t)) = 0. \tag{7}
\]

Notice that (7) actually gives constraints on \( \tilde{u}_1, \tilde{u}_2 \). For example, \( \max_{0 \leq t \leq t_1} \phi_1(t, \gamma_0(t)) = 0 \) implies that \( \gamma_0(t) - v_1 t \geq 1, \) for any \( t \in [0, t_1] \). If we denote \( \gamma_0(t) - v_1 t := a t + b \), where \( a = (R(\tilde{u}_1) - X)/t_1 - v_1 \) and \( b = \gamma_0(t) \). Then (7) is

\[
g(t) := a^2 t^2 + 2a \cdot b t + b^2 - 1 \geq 0, \quad \text{for any } t \in [0, t_1].
\]

Because \( g(t) \) is a quadric function of \( t \) and \( g(t_1) = 0 \) (\( \tilde{u}_1 \) lies on the moving obstacle), the above constraint is equivalent to

\[
g'(t)|_{t=t_1} = 2a^2 t_1 + 2a \cdot b \leq 0,
\]

which gives an explicit constraint on \( \tilde{u}_2 \). Similarly, we can get an explicit constraint on \( \tilde{u}_1 \) as well.

Through the intermittent diffusion process (6), using \( m = 40, \sigma_{2k} = 0.2, k \leq 20 \), we obtain two minimizers (there are only two minimizers for this example) satisfying stopping criterion \( \|\nabla \phi_1\| \leq 10^{-4} \). One is an optimal path determined by junctions \( \{(u_1, u_2, t_1, t_2) = (1.0493, 2.0732, 0.3885, 0.6179)\} \), and the other by junctions \( (2.1240, 5.2527, 0.3802, 0.6127) \). By comparing their costs, the former is the global minimizer with cost 19.9130, and the later is a local one with cost 20.8160. Among 20 ID intervals, the global optimal path is found 18 times, while the local minimizer is visited 2 times, indicating the proposed algorithm finds a global solution with a larger probability.
In the following examples, a maximal speed constraint $v_m = 20$ is imposed and the number of junctions are not known a priori.

**Multiple obstacles and fixed terminal time**

The environment consists of six disks initially centered at $(0,0)$, $(4.5,3)$, $(8,-3)$, $(10,4)$, $(12,-3)$, $(15,-4)$, with radii $1$, $1$, $1$, $1.2$, $1$, $1$, and moving at constant velocities $(3,5)$, $(-2,-5)$, $(-2,4.5)$, $(0,-5.5)$, $(1,5.5)$ and $(1,5.5)$ respectively. The starting and ending points are $X = (-2,0.5)$ and $Y = (20,0.5)$.

In this scenario, we set the terminal time $T = 1$. Since we don’t know the number of junctions and their locations, we start the algorithm by connecting $X$ and $Y$ with a straight path of constant speed. Its intersections with the moving obstacles are the initial values for the junctions. In fact, this is our general way to initialize the algorithm.

We run the proposed algorithm with $m = 6$ and find two minimizers. The first one, the global optimal path, intersects with four obstacles resulting in a total cost $510.353$, as shown in Figure 6 and a movie is available at [https://youtu.be/zq0GQQZGvE](https://youtu.be/zq0GQQZGvE). The other is a local minimizer with total cost $535.273$, whose trajectory encounters five obstacles, see a movie in [https://youtu.be/AO3Cy5J1-Rg](https://youtu.be/AO3Cy5J1-Rg).

We remark that these two optimal paths have different numbers of junctions corresponding to different dimensions in the optimization. The computation is efficient, the average time to obtain a solution is around 20 seconds, which is done by Matlab on a laptop with core i5, 1.5GHZ and 4GB RAM.

**Unknown terminal time**

The terminal time $T$ in this example is assumed unknown. The proposed method can handle it with a minor modification of the previous example, namely treating $(T,Y)$ as a new junction. This new junction does not move spatially, but its time is undetermined. With this consideration, we re-cast the finite dimensional optimization as

$$
\min_{\tilde{u}} \sum_i J_i(\tilde{u}), \quad \text{where } \tilde{u} = (\tilde{u}_1, \ldots, \tilde{u}_N, T),
$$

with the constraint set similar in (7).

To illustrate, we consider a moving environment with two disks of radii $1$ and $1$, initially centered at $(0,0)$, $(6.5,3)$ respectively. They move at constant velocities $(5,0)$, $(-5,0)$. All paths start at $X = (-2,0.5)$, and end at $Y = (10,0.5)$. The running cost is $L = \dot{\gamma}^2 + 200$.

**Conclusions**

In this work, we propose a new numerical algorithm for optimal path planning in 2D environments with moving obstacles. By leveraging the separable structure of optimal paths, we transfer the problem into a finite dimensional optimization problem which is solved numerically by initial value SDEs. We illustrate that our algorithm can efficiently find a series of minimizers, including the global one. The accuracy is high while the computation cost is relative low.

**Appendix**

In this appendix, we give proofs for several properties related to the Algorithm 1.

**Proof of Lemma**

A classical problem in calculus of variation. Since the optimal path satisfies the Euler-Lagrange equation

$$\nabla_{\dot{\gamma}} L(t, \gamma, \dot{\gamma}) - \frac{d}{dt} \nabla_{\dot{\gamma}} L(t, \gamma, \dot{\gamma}) = 0 \Rightarrow -2 \frac{d}{dt}(\dot{\gamma}) = 2\dot{\gamma} = 0,$$

we have

$$\dot{\gamma} = 0.$$

The above figure demonstrated the paths of two minimizers. The left one is the global minimizer, while the right one is the local minimizer.
which implies that the optimal trajectory is with zero acceleration. Hence
\[ \gamma_i(t) = \frac{x_{i+1} - x_i}{t_{i+1} - t_i} (t - t_i) + x_i \]
with
\[ J_i(\tilde{x}) = \frac{(x_{i+1} - x_i)^2}{t_{i+1} - t_i} + c(t_{i+1} - t_i). \]

**Proof of Lemma** Indeed, (8) contains three cases:

(a) The speed constraint is active while the path constraint is not.
\[ \|\gamma(t)\| = v_m, \quad t \in [t_i, t_{i+1}] ; \]
(b) The path constraint is active while the speed constraint is not. There exists an obstacle \( P_k \), such that
\[ \gamma(t) \in \partial P_k(t), \quad t \in [t_i, t_{i+1}] ; \]
(c) Both path and speed constraints are active. There exists an obstacle \( P_k \), such that
\[ \gamma(t) \in \partial P_k(t), \quad \|\gamma(t)\| = v_m, \quad t \in [t_i, t_{i+1}] . \]

The proofs for cases (a), (c) can be found in [1]. Here we only prove case (b). In this case, the control problem (3) becomes
\[ \min \int_{t_i}^{t_{i+1}} (\dot{\gamma}^2(t) + c) dt , \quad (8) \]
subject to
\[ \gamma(t_i) = x_i, \quad \gamma(t_{i+1}) = x_{i+1}, \quad \phi_k(t, \gamma(t)) = 0. \]

(8) can be solved explicitly. We parametrize the boundary of obstacle \( P_k \) by \( \alpha(u) \), where \( u \in [0, l_k] \) is an arclength parameter and \( l_k \) is the perimeter of \( \partial P_k \). Hence \( \gamma(t) \) is represented by its relative position \( u(t) \) on the obstacle
\[ \gamma(t) = \alpha(u(t)) + v_k \cdot t. \]

In this setting, (8) is equivalent to a new form
\[ \min \{ \int_{t_i}^{t_{i+1}} L_1(t, u, \dot{u}) dt \mid u(t_i) = u_i, \quad u(t_{i+1}) = u_{i+1} \} , \]
where
\[ L_1(t, u, \dot{u}) = \dot{u}(t)^2 + 2(\alpha_u(u(t)) \cdot v_k)\dot{u}(t) + v_k^2 + c , \quad (9) \]
and \( x_i = \alpha(u_i) + v_k \cdot t_i, \quad x_{i+1} = \alpha(u_{i+1}) + v_k \cdot t_{i+1} \).

Notice that
\[ \frac{\partial}{\partial u} L_1 = 2(\alpha_{uu} \cdot v_k)\dot{u}, \quad \frac{\partial}{\partial \dot{u}} L_1 = 2\ddot{u} + 2(\alpha_u \cdot v_k) , \]
and
\[ d \frac{\partial}{\partial \dot{u}} L_1 = 2\ddot{u} + 2(\alpha_{uu} \cdot v_k) \dot{u}. \]

From the Euler-Lagrange equation, we have
\[ \frac{\partial}{\partial u} L_1(t, u, \dot{u}) - d \frac{\partial}{\partial \dot{u}} L_1(t, u, \dot{u}) = 0 \Rightarrow \ddot{u} = 0 , \]
which implies that the optimal path has a relative constant speed. Finally, we prove that the optimal path in the sample example contains at most two junctions.

**Proof of Proposition** We only need to show that the optimal path \( \gamma^*(t) \) connecting \( (t_1, x_1) \), the first junction on \( P_1 \), and \( (t_2, x_2) \), the last junction on \( P_1 \), must lie on \( \partial P_1 \) entirely. Assume this is not true, \( \gamma^* \) does not lie on \( \partial P_1 \) entirely, we must have
\[ \max_{t \in [t_1, t_2]} \phi_1(t, \gamma^*(t)) > 0. \]

If we denote \( \theta_0 = \arg \max_{t_1 \leq t \leq t_2} \phi_1(t, \gamma^*(t)) \), then \( \tilde{\theta}_0 = (\theta_2, \gamma^*(\theta_0)) \) is at the outside of \( \partial P_1 \). Combine the fact that each optimal sub-arc is also optimal and Lemma 1, there must exist a line segment, which is part of \( \gamma^* \) pass though point \( \tilde{\theta}_0 \). In other words, there exists two points \( \tilde{\theta}_1, \tilde{\theta}_2 \) on \( \partial P_1 \), where
\[ \theta_1 = \sup \{ t \in [t_1, t_2] \mid \gamma^*(t) \gamma^*(\theta_0) \subset \gamma^* \} , \]
\[ \theta_2 = \inf \{ t \in [t_1, t_2] \mid \gamma^*(\theta_0) \gamma^*(t) \subset \gamma^* \} , \]
where \( \overline{ab} \) represents a line connecting points \( a, b \) with constant velocity.

For convenience, we denote \( \tilde{\theta}_1 = (\theta_1, \gamma^*(\theta_1)) \), \( \tilde{\theta}_2 = (\theta_2, \gamma^*(\theta_2)) \). The optimal path can be decomposed as
\[ \gamma^* = \gamma^*(t_1, \theta_1) \cdot \tilde{\theta}_1 \cdot \tilde{\theta}_2 \cdot \gamma^*(\theta_2, t_2) , \]
where \( \gamma^*(t_1, \theta_1), \gamma^*(\theta_2, t_2) \) refers trajectories of \( \gamma^* \) from time intervals \( (t_1, \theta_1), (t_2, \theta_2) \).

Since \( \overline{\theta_1 \theta_2} \) is in the free space and \( P_1(t) \) is a convex set in \( \mathbb{R}^3 \) \( (P_1(t) \) is a cylinder in time-space), then one can find two points \( \tilde{\theta}_3 = (\theta_3, \gamma^*(\theta_3)) \in \overline{\theta_1 \theta_0} \) and \( \tilde{\theta}_4 = (\theta_4, \gamma^*(\theta_4)) \in \overline{\theta_0 \theta_2} \), such that \( \overline{\theta_3 \theta_4} \) is also in the free space. Therefore we can construct another feasible path \( \gamma \),
\[ \gamma = \gamma^*(t_1, \theta_1) \cdot \tilde{\theta}_1 \cdot \tilde{\theta}_3 \cdot \tilde{\theta}_4 \cdot \tilde{\theta}_2 \cdot \gamma^*(\theta_2, t_2) . \]

Since the difference of two paths are the triangular which passes three points \( \tilde{\theta}_3, \tilde{\theta}_4 \), and \( \tilde{\theta}_1 \),
\[ J(\overline{\tilde{\theta}_1 \tilde{\theta}_3 \tilde{\theta}_4 \tilde{\theta}_2}) < J(\overline{\tilde{\theta}_1 \theta_0 \theta_2}) . \]

Hence
\[ J(\gamma) < J(\gamma^*) , \]
which contradicts the fact that \( \gamma^* \) is the optimal path.

**References**


Figure 6. Fixed terminal time. These are snapshots of the global optimal path in an environment with 6 moving obstacles (grey). The red trajectory represents the optimal path in the free space, while the blue part indicates that the path travels along the moving obstacle boundary.

Figure 7. Undetermined terminal time. These are snapshots of the local optimal path in an environment with 2 moving obstacles. The red trajectory represents the optimal path in the free space, while the blue part indicates that the path travels along the moving obstacle boundary.

Figure 8. Non-convex boundary obstacles. These are snapshots of the global optimal path. The red part represents the path in the free space, while the blue part indicates that the path travels along the moving obstacle boundary.


