



Method of evolving junction on optimal path planning in flows fields

Haoyan Zhai¹ · Mengxue Hou² · Fumin Zhang² · Haomin Zhou¹

Received: 17 December 2021 / Accepted: 15 August 2022

© The Author(s), under exclusive licence to Springer Science+Business Media, LLC, part of Springer Nature 2022

Abstract

We propose an algorithm using method of evolving junctions to solve the optimal path planning problems with piece-wise constant flow fields. In such flow fields, we prove that the optimal trajectories, with respect to a convex Lagrangian in the objective function, must be formed by piece-wise constant velocity motions. Taking advantage of this property, we transform the infinite dimensional optimal control problem into a finite dimensional optimization and use intermittent diffusion to solve the problems. The algorithm is proven to be complete. At last, we demonstrate the performance of the algorithm with various simulation examples.

Keywords Optimal path planning · Intermittent diffusion · Method of evolving junctions

1 Introduction

Autonomous Underwater Vehicles (AUVs) are a class of submerged marine robots capable of performing persistent missions in the ocean. Over the last few decades, AUVs have been widely applied to various applications, including ocean sampling (Leonard et al., 2010; Smith et al., 2010), surveillance and inspection (Ozog et al., 2016), and many more. Since most of the applications require the AUVs to execute long-term missions in unknown and dynamic oceanic environments with minimum human supervisions, their success is highly dependent on the level of autonomy of the AUVs.

For robots operating in complex and dynamic environments, path planning is one of the crucial and fundamental functions to achieve autonomy. In short, the task is finding a feasible or optimal path, under the influence of a dynamic flow field, for an AUV to reach a predefined target point.

Path planning has been studied extensively in robotics over the years. Several popular algorithms that have been applied to underwater vehicle navigation include graph based methods such as the A* method (Rhoads et al., 2012; Pereira et al., 2013; Kularatne et al., 2018, 2017) and the Sliding Wavefront Expansion method (SWE) (Soulignac, 2011), probability based methods like the Rapidly exploring Random Trees (RRTs) (LaValle, 1998; Kuffner and LaValle, 2000; Gammell et al., 2018; Chen et al., 2019; Shome et al., 2020), and methods that approximate the solution of HJ (Hamilton-Jacobi) equations, such as the Level Set Method (LSM) (Sethian, 1999; Lolla, 2016).

When the A* method is applied for an AUV, the continuous flow field is discretized into grid cells. At each step, it compares the cost of going from the current position to its neighboring cells so as to identify a path with the lowest cost. However, when the resolution, which is inverse proportional to the cell size, is not high enough, it may fail to find a feasible path even if there exists one. The SWE addresses the incompleteness issue of A* by introducing the sliders, which are points that slide on the boundary of cells, as the nodes in the graph search problem. As all feasible path connecting the starting and target position can be formed by a sequence of the sliders, the path planning problem is converted to a Mixed Integer Optimization problem (MIP) on cell sequence and slider positions. The advantage of SWE is that it can find a feasible solution if one exist. However, if the cost of traveling in each cell depends on slider positions, then the SWE method may not be able to find the optimal solution. RRT and

✉ Haomin Zhou
hmzhou@math.gatech.edu

Haoyan Zhai
haoyanzhai@gmail.com

Mengxue Hou
mhou30@gatech.edu

Fumin Zhang
fzhang37@gatech.edu

¹ Department of Mathematics, Georgia Institute of Technology, North Ave., Atlanta, GA 30332, USA

² Department of Electrical and Computer Engineering, Georgia Institute of Technology, North Ave., Atlanta, GA 30332, USA

RRT* explore the flow field by using random sampling, with a bias towards the unexplored area (Karaman and Frazzoli, 2011; Noreen et al., 2016). Like many other probabilistic based methods, RRTs provide, guaranteed in the asymptotic sense, a globally optimal solution only when the samples are dense enough. Methods to improve the convergence rate of RRT and RRT* have been introduced (Janson et al., 2015; Gammell et al., 2018). LSM computes a propagating front, incorporating both flow and vehicle speeds, to approximate the solution of the HJ equation. Then the optimal path is computed by back tracking along the normal direction of the wavefront from the destination position. LSM can plan a shortest time path over time-varying flow field, usually at the cost of longer computational time.

Using a regular grid to discretize the flow field can result in unnecessary large number of cells, which increases the computational burden of the planning methods. Since the flow speed in adjacent position is usually similar, the flow field can be partitioned into piece-wise constant subfields, within each the flow speed is a constant vector (Hou et al., 2019; Kularatne et al., 2017; Kaiser et al., 2014; Ser-Giacomi et al., 2015). Taking advantage of the flow field partitioning, we can parameterize a path by the index of cells crossed by the path, and the intersection points between the path and the boundaries of regions. In this way, the original path planning problem in the continuous flow field is reduced to a finite dimensional MIP, with the decision variables being the index of cells crossed by the path, and the intersection points between the path and the boundaries of regions. Because of the coupling between the integer and continuous variable, the optimization problem is high dimensional, and is computationally expensive to solve. Existing works following this approach solves the MIP in a bi-level approach (Kularatne et al., 2017; Soulignac, 2011). The lower-level solver computes the minimum-cost continuous variable in each partitioned cell using convex optimization approaches, and then the upper-level solver optimizes the integer variable by the A* algorithm, with the branch cost of the decision tree being the optimal solution derived from the lower-level controller. However, because of the decoupling between the cell sequence and the intersection points, such algorithms cannot guarantee optimality of the solution, without imposing more assumptions on the problem (Sinha et al., 2017).

In this paper, we present the interleaved branch-and-bound depth-first search with intermittent diffusion (iBnBDFS-ID) method for computing time and energy optimal paths for vehicles traveling in the partitioned flow field. Different from existing strategies that solves the MIP using a bi-level approach (Kularatne et al., 2018, 2017; Soulignac, 2011), we solve the MIP in an interleaved way, with guaranteed optimality. When a cell sequence connecting the start and the destination is found by the depth-first search (DFS), the intersection position between the optimal path and the cell

boundary curves is computed by the intermittent diffusion (ID) method (Li et al., 2017). For large search tree with high branching factor, DFS has shown to be computationally impractical (Forrest et al., 1974). To address this challenge, we propose a novel branch-and-bound (BnB) DFS method that reduces the number of nodes to be searched in the decision tree, and hence reduces the computational cost in solving the problem. The key insight is that we can find a lower bound of the the stage cost induced from traveling in one partitioned cell. Leveraging the lower bound, we compute a lower bound on the cost-of-arrival, and stop the DFS at the current node if lower bound on its cost-of-arrival exceeds the best solution found so far. Therefore, this work proposes a novel method to solve the AUV path planning problem, as an instance of MIP, without excessive demand on computing resources. Contributions of our work are as follows:

1. *Optimal solution structure* We prove that for vehicle traveling in partitioned flow regions, the structure of the global optimal path is a piece-wise constant velocity motion, if the objective function is given as the total traveling time or the total energy consumption (modeled by a quadratic function).
2. *Interleaved solver for MIP* We propose a novel computationally efficient method to solve the path planning method in the partitioned flow field, as an instance of the MIP, and proved completeness of the algorithm. Different from the existing works that decouples the process of optimizing the cell sequence assignment and the intersection position (Kularatne et al., 2017; Soulignac, 2011), our method optimizes the cell sequence assignment and the intersection position between the path and the cell boundaries at the same time. By incorporating a BnB technique in the DFS, the method avoids exhaustive search through the full decision tree. Hence it solves the planning problem with low computation cost.

We evaluate the performance of the proposed algorithm through simulation of AUV path planning in both simulated and realistic ocean flow fields, and compare its computational cost and solution quality with the LSM and A* method. The proposed method achieves comparable path cost, and has significantly lower computation cost compared with the LSM and A* method. Moreover, we provide time complexity analysis of the proposed algorithm, which gives insights on its computation cost with respect to different levels of partitioning.

In the next section, we present the formulation of the problem in the optimal control framework and the assumptions used in the paper. In Sect. 3, we show how to transform the original problem into the finite dimensional optimization, and provide the algorithm. In Sect. 4, proof of the complete-

ness of the algorithm is offered, accompanied with several numerical experiments in Sect. 5. At last, we end our paper with a brief conclusion.

2 Problem statement

We consider the vehicle moving in the space $\Omega \subset \mathbb{R}^d$ with dimension d , equipped with a dynamic $\dot{x} = u + v$, where u is the environment flow velocity, and v is the vehicle velocity or the control variable, satisfying $v \in \mathcal{U}$, providing that $\mathcal{U}$ is a compact space such that $\|v\| \leq V$, where V is the max speed of the vehicle. We further assume that by taking advantage of existing methods on ocean flow field partitioning (Hou et al., 2019, 2021), the flow field is divided into finite number of convex regions $\{R_\alpha\}_{\alpha \in I_R}$ by boundary curves (surfaces if is in $\mathbb{R}^d, d \geq 3$) $\{f_{\alpha\beta}\}_{(\alpha,\beta) \in I}$. Here

$$I = \{(\alpha, \beta) \in I_R \times I_R : \dim(\partial R_\alpha \cap \partial R_\beta) = d - 1\},$$

where $\dim(S)$ returns the dimension of the set S and ∂S is the boundary of S , and $f_{\alpha\beta}(x) = 0$ is the $(d - 1)$ -dimensional compact boundary of the region R_α and R_β (can be denoted as ∂R_α and ∂R_β respectively). Let x denote a point on the boundary, we can parameterize the cell boundary by defining a piece-wise diffeomorphism

$$x(\lambda) : D \subset \mathbb{R}^{d-1} \longrightarrow \{y : f_{\alpha\beta}(y) = 0\},$$

where D is a $(d - 1)$ -dimensional unit ball. Also within each region, we suppose that the flow velocity u is a constant vector. Hence we can denote the flow velocity in each region R_α separately by u_α . The vehicle needs to be controlled from an initial position x_0 , crossing different regions and finally reaching the target position x_f .

Since there could be infinitely many feasible paths linking x_0 and x_f , a cost function is introduced to measure the travel expense with respect to different potential trajectories. We denote the cost function to be

$$J(v, T) = \int_0^T L(v(t))dt, \quad (1)$$

letting $\gamma(t)$ be a continuous path with $\gamma(0) = x_0, \gamma(T) = x_f$ and $\dot{\gamma}(t) = u + v$. In this paper, we discuss the problem with the cost function specifically being the total travel time, that is $L(x, v) = 1$ and the kinetic energy combining a constant running cost, in which case $L(x, v) = \|v\|^2 + C$ where C and $\|v\|$ are not simultaneously 0 for all $x \in \Omega$ because of technical issues which will be discussed in Sect. 4.2. Our goal is to find the optimal control function $v(t)$ with minimum

cost, and can be expressed as the following problem:

$$\begin{aligned} \min_{v, T} \quad & \int_0^T L(v)dt \\ \text{s.t.} \quad & \dot{x} = v + u, \\ & x(0) = x_0, \\ & x(T) = x_f, \\ & \max_{t \in [0, T]} \|v\| \leq V \end{aligned} \quad (2)$$

There may not exist a feasible solution to (2), in the case where the flow speed is larger than the vehicle speed. To guarantee existence of a feasible solution, we assume that the flow speed is lower than the maximum vehicle speed V .

In the next section, we will concretely discuss our method, the idea of which is to seek a way to change the infinite dimensional optimal control problem into a finite dimensional optimization problem.

3 Our method

All possible trajectories will continuously pass through a sequence of regions. Since the flow velocity u is a constant vector in each region, we have the following theorem that will be proved in the Appendix:

Theorem 3.1 *In the optimal solution for*

$$\begin{aligned} \min_{v, T} \quad & \int_0^T L(v)dt \\ \text{s.t.} \quad & \dot{x} = f(v + u), \\ & x(0) = x_0, \\ & x(T) = x_f, \\ & \max_{t \in [0, T]} \|v\| \leq V. \end{aligned}$$

where u is a constant vector and f is invertible, the velocity v is a constant vector.

With Bellman's principle, it is the fact that the optimal path should be formed via a piece-wise constant velocity motion. Thus, we restrict the velocity of the vehicle in each region to be a constant. Further, the regions are convex, therefore, the straight line path always lies inside each of the cell crossed and the path is continuous. Now we can introduce the notation v_i to be the vehicle velocity in the i^{th} cell crossed.

Because of Theorem 3.1, we can parameterize solution to (2) by i) the cell sequence that the path goes through, and ii) the position where the path intersects with the cell boundaries. We assume that the initial and the terminal position in (2) are on the boundary of the partitioned cells. In general this can be achieved by incorporating a fixed rule to partition

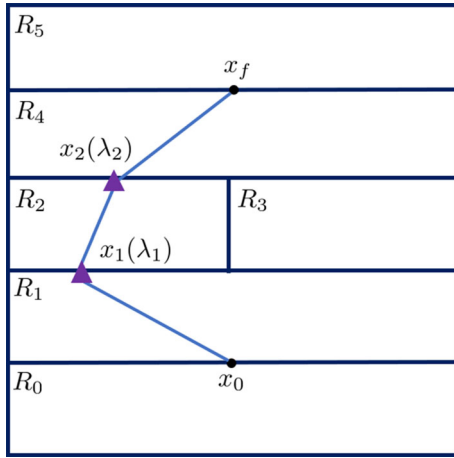


Fig. 1 Example of parameterizing a feasible path in the partitioned domain by the cell sequence and the junction position. The purple triangles are the junction position. The path goes through 3 cells. Index of 3 cells are $c_1 = 1, c_2 = 2, c_3 = 4$. On the boundary f_{12} and f_{24} there are two junctions, position of which is parameterized by λ_1 and λ_2 (Color figure online)

the cells containing x_0 and x_f into two separate cells, with the new generated boundary going through x_0 and x_f .

Let $\mathcal{C} = \{c_1, c_2, \dots, c_n\}$ denote a sequence of the index of cells that a path travels through, $c_i \in I_R, \forall i \in [1, n]$. We define the junction set $\{x_i\}_{i=0}^n$ to be the intersections between a trajectory and the cell boundaries. The junction x_i is on the boundary between the two regions indexed as c_i and c_{i+1} . The junction x_0 is the initial junction and the destination x_f is the last junction e.g. $x_n = x_f$. For each region R_i , the entrance junction is x_{i-1} and the exit junction is x_i . Fig. 1 shows illustration of parameterizing a feasible path using junctions and the cell sequence.

Given the above mentioned parameterization to feasible trajectories, to solve the planning problem, we need to i) determine the sequence of regions that the optimal path goes through, and ii) compute the optimal position of intersection position between the trajectory and the cell boundaries. In this section, we propose a novel method to solve the planning problem. The key idea of the method is based on two insights. First, we only need to determine the cell sequence when the vehicle reaches cell boundary. Each assignment of visited cell sequence will produce a branch of a decision tree, with each branch in the tree representing a cell that the vehicle crosses. The decision tree contains a finite number of branches, and a feasible solution must be a path from the root of the tree to the target position. Second, the optimal junction position can be easily computed if we fix the assignment of cell sequence.

3.1 Minimize total travel time

When minimizing total travel time is the goal, we further let the vehicle move in maximum speed V . Let u_{c_i} denote the flow speed in the cell indexed by c_i . Then the cost function

can be converted in the format below:

$$J(v, T, \mathcal{C}) = \int_0^T ds = T = \sum_{i=1}^n g_i^t(x_i, x_{i-1}, c_i),$$

where g_i^t is the travel time in region R_{c_i} and is expressed as:

$$g_i^t = \frac{\|x_i - x_{i-1}\|}{\|v_i + u_{c_i}\|}.$$

Since flow and vehicle velocities are constant in each cell, the motion must be in straight line, meaning that $\frac{x_i - x_{i-1}}{\|x_i - x_{i-1}\|} = \frac{v_i + u_{c_i}}{\|v_i + u_{c_i}\|}$, and it leads to:

$$v_i + u_{c_i} = \frac{x_i - x_{i-1}}{\|x_i - x_{i-1}\|} \|v_i + u_{c_i}\|. \quad (3)$$

We start from the following equality:

$$(v_i + u_{c_i})^T [(u_{c_i} + v_i) - 2u_{c_i} + (u_{c_i} - v_i)] = 0,$$

and inserting (3) into it yields:

$$\|v_i + u_{c_i}\|^2 - \frac{2(x_i - x_{i-1})^T u_{c_i}}{\|x_i - x_{i-1}\|} \|v_i + u_{c_i}\| + \|u_{c_i}\|^2 - V^2 = 0.$$

This leads to the following expression:

$$\|v_i + u_{c_i}\| = \frac{(x_i - x_{i-1})^T u_{c_i}}{\|x_i - x_{i-1}\|} \pm \left(\left(\frac{(x_i - x_{i-1})^T u_{c_i}}{\|x_i - x_{i-1}\|} \right)^2 + V^2 - \|u_{c_i}\|^2 \right)^{\frac{1}{2}}.$$

To minimize the travel time, we take the plus sign so that

$$\|v_i + u_{c_i}\| = \frac{(x_i - x_{i-1})^T u_{c_i}}{\|x_i - x_{i-1}\|} + \left(\left(\frac{(x_i - x_{i-1})^T u_{c_i}}{\|x_i - x_{i-1}\|} \right)^2 + V^2 - \|u_{c_i}\|^2 \right)^{\frac{1}{2}} \quad (4)$$

and have the travel time in region R_{c_i} as

$$g_i^t = \frac{1}{\|u_{c_i}\|^2 - V^2} \left((x_i - x_{i-1})^T u_{c_i} - \sqrt{((x_i - x_{i-1})^T u_{c_i})^2 + \|x_i - x_{i-1}\|^2 (V^2 - \|u_{c_i}\|^2)} \right).$$

At last, since x_i is on the boundary of the regions indexed by c_i and c_{i+1} , we have the smooth parameterization of each x_i

as $x_i = x_i(\lambda_i)$ where $\lambda_i \in D \subset \mathbb{R}^{d-1}$, transforming finally the cost function to be

$$J(\lambda_1, \dots, \lambda_{n-1}, C) = g_1^t(x_1(\lambda_1), x_0, c_1) + g_n^t(x_{n-1}(\lambda_{n-1}), x_f, c_n) + \sum_{i=2}^{n-1} g_i^t(x_i(\lambda_i), x_{i-1}(\lambda_{i-1}), c_i).$$

and the problem is changed to a finite dimensional optimization problem

$$\min_{\lambda_i \in D, c_i \in I_R, i=1, \dots, n-1} J(\lambda_1, \dots, \lambda_{n-1}, C). \quad (5)$$

3.2 Minimize energy

If the cost function is the kinetic energy with a constant running cost $C \geq 0$

$$J(v, T) = \int_0^T \|v\|^2 + C dt,$$

we first consider the constant velocity motion in each region R_i within time t_i . Fixing the entrance and exit junctions x_{i-1} and x_i , we can derive vehicle speed as

$$v_i = \frac{x_i - x_{i-1}}{t_i} - u_{c_i},$$

and

$$\|v_i\|^2 = \frac{\|x_i - x_{i-1}\|^2}{t_i^2} + \|u_{c_i}\|^2 - \frac{2(x_i - x_{i-1})^T u_{c_i}}{t_i}. \quad (6)$$

The cost function in the specific region R_{c_i} can be rewritten as

$$\begin{aligned} J(t_i) &= \int_0^{t_i} \|v_i\|^2 + C ds \\ &= \frac{\|x_i - x_{i-1}\|^2}{t_i} + (\|u_{c_i}\|^2 + C)t_i \\ &\quad - 2(x_i - x_{i-1})^T u_{c_i} \\ &\geq 2\sqrt{\|u_{c_i}\|^2 + C}\|x_i - x_{i-1}\| \\ &\quad - 2(x_i - x_{i-1})^T u_{c_i} \end{aligned} \quad (7)$$

with equality holds at

$$t_{i,1}^* = \frac{\|x_i - x_{i-1}\|}{\sqrt{\|u_{c_i}\|^2 + C}}. \quad (8)$$

Hence, if the maximum vehicle speed is large enough, that is,

$$V^2 \geq \frac{\|x_i - x_{i-1}\|^2}{t_{i,1}^{*2}} + \|u_{c_i}\|^2 - \frac{2(x_i - x_{i-1})^T u_{c_i}}{t_{i,1}^*}, \quad (9)$$

we can find the optimal vehicle forward speed by replacing t_i in (6) with the optimal solution in (8):

$$\|v_i\|^2 = 2\|u_{c_i}\|^2 + C - \frac{2(x_i - x_{i-1})^T u_{c_i}}{\|x_i - x_{i-1}\|} \sqrt{\|u_{c_i}\|^2 + C}. \quad (10)$$

Hence, if (9) holds, we let the vehicle move in the speed of $\|v_i\|$ in (10).

However, if the (9) does not hold, we set the vehicle speed to be the maximum speed V and the cost function is reduced to be $J = (V^2 + C)t_{i,2}^*$ where

$$\begin{aligned} t_{i,2}^* &= \frac{1}{\|u_{c_i}\|^2 - V^2} \left((x_i - x_{i-1})^T u_{c_i} - \left(\left((x_i - x_{i-1})^T u_{c_i} \right)^2 + \|x_i - x_{i-1}\|^2 (V^2 - \|u_{c_i}\|^2) \right)^{\frac{1}{2}} \right), \end{aligned} \quad (11)$$

which is a root of

$$(\|u_{c_i}\|^2 - V^2)t_i^2 - 2(x_i - x_{i-1})^T u_{c_i}t_i + \|x_i - x_{i-1}\|^2 = 0.$$

Thus the energy spent in each region is

$$g_i^e(x_i, x_{i-1}) = \begin{cases} 2\sqrt{\|u_{c_i}\|^2 + C}\|x_i - x_{i-1}\| & \text{if (9) holds} \\ -2(x_i - x_{i-1})^T u_{c_i} & \\ (V^2 + C)t_{i,2}^* & \text{otherwise} \end{cases}$$

By using the same parametrization as in the time-optimal planning, we finally have the problem to be a finite dimensional optimization formulated as

$$\min_{\lambda_i \in D, c_i \in I_R, i=1, \dots, n-1} J(\lambda_1, \dots, \lambda_{n-1}, C) \quad (12)$$

where

$$\begin{aligned} J(\lambda_1, \dots, \lambda_{n-1}, C) &= g_1^e(x_1(\lambda_1), x_0, c_1) \\ &\quad + g_n^e(x_{n-1}(\lambda_{n-1}), x_f, c_n) \\ &\quad + \sum_{i=2}^{n-1} g_i^e(x_i(\lambda_i), x_{i-1}(\lambda_{i-1}), c_i). \end{aligned}$$

Furthermore, g_i^t and g_i^e has the following properties:

Proposition 3.1 *If there exists a feasible trajectory from x_{i-1} to x_i in R_i and $V \neq \|u_i\|$, then $g_i^t(x_i, x_{i-1}, c_i)$ and $g_i^e(x_i, x_{i-1}, c_i)$ are differentiable.*

We give the proof of this property in the Appendix. With this proposition, we can take the derivative of the objective function, which is pivotal for applying the Intermittent Diffusion method described in Sect. 3.3.2.

3.3 Construction of decision tree

We introduce a tree structured graph to model how the sequence of cell that the path crosses affects the total cost. In the tree, each node represents a boundary curve that contains a junction point. To construct the decision tree, we first define two boundary curves f_{c_1, c_2} , f_{c_3, c_4} as adjacent if

$$\{c_1, c_2\} \cap \{c_3, c_4\} \neq \emptyset, \quad (13)$$

indicating that the two curves are two boundaries of the same cell.

3.3.1 Decision tree traversal

Starting from the boundary curve containing the initial position, which is the root node, a directed decision tree can be formed by connecting the adjacent boundary curves in the domain. The branch generation stops when the node is the boundary curve containing the destination position. Each node in the decision tree represents a boundary curve that a feasible path will go through. A connected path in the decision tree, starting from the root node to the target node represent a sequence of cells that a feasible path will cross. Fig. 2 shows the decision tree constructed from a partitioned workspace.

The construction of the entire decision tree is not necessary, and is time-consuming. However, for the purpose of clearly presenting the concept of the planning method, we will discuss how the tree can be fully constructed, and then present the branch pruning technique. We construct the decision tree iteratively using the depth-first search method. Let n_c represent the current node. At each step, we search for all the boundary curves adjacent to the current node, and call the current node as the predecessor of the new node. Since the optimal path will not visit one boundary curve more than one time, the optimal cell sequence should not include loops. Hence, when searching for the adjacent nodes to be expanded next, we will not expand an adjacent node of n_c if it is already visited. This node generation process terminates when the target node is visited.

By constructing the decision tree we can find all the cell sequences connecting the root and the terminal node. For one cell sequence, the MIP (5) and (12) reduces to a finite

dimensional non-convex optimization problem over the junction positions. Next we show how this optimization problem can be solved using the Intermittent Diffusion method.

3.3.2 Intermittent Diffusion

The objective functions in (5) and (12) are both differentiable. Hence we use the Intermittent Diffusion (ID) to get the global minimizer (Chow et al., 2013), the key idea of which is adding white noise to the gradient flow intermittently. Namely, we solve the following stochastic differential equation (SDE) on the configuration space

$$d\lambda = -\nabla J(\lambda)d\theta + \sigma(\theta)dW(\theta) \quad (14)$$

where $\lambda = (\lambda_1, \dots, \lambda_{n-1}) \in D^{n-1}$ and $W(\theta)$ is the standard Brownian motion. The diffusion is a piece-wise constant function defined by

$$\sigma(\theta) = \sum_{i=1}^N \sigma_i I_{[S_i, T_i]}(\theta)$$

where σ_i are constant and $I_{[S_i, T_i]}(\theta)$ is the characteristic function on interval $[S_i, T_i]$ with $0 \leq S_1 < T_1 < S_2 < T_2 < \dots < S_N < T_N < S_{N+1} = T$.

Thus, if $\sigma(\theta) = 0$, we obtain the gradient flow back while when $\sigma(\theta) \neq 0$, the solution of (14) has positive probability to escape the current local minimizer. The theory of ID indicates that the solution of (14) visits the global minimizer of J with probability arbitrarily close to 1 if $\min_i |T_i - S_i|$ is large enough, which is guaranteed by Theorem 6.1 in Appendix.

We use the forward Euler discretization to discretize the above SDE and get

$$\lambda^{k+1} = \lambda^k - h \nabla J(\lambda^k) + \sigma_k \xi^k \sqrt{h}. \quad (15)$$

The constant h is the step size, σ_k is the coefficient chosen to add the intermittent perturbation and $\xi^k \sim \mathcal{N}(0, 1)$ is a Gaussian random variable. In practice, the global minimizer can be reached by tuning the white noise strength σ_k as well as setting the total evolution round N long enough.

We summarize the ID algorithm in Algorithm 2 with the objective function being (5) or (12).

3.3.3 Branch cost lower bound

We leverage a BnB technique to prune the branches in the decision tree, in order to save the computation cost in both node generation and solving for the optimal junction position. We approach this problem by leveraging the lower bound of the decision tree branch cost. In the decision tree, each branch represents a path segment connecting the junction on

one boundary to the junction on one of its adjacent boundary. Hence, the branch cost of the decision tree represents the stage cost generated from going from one junction to another. However, since the optimal junction position is unknown when we construct the decision tree, the exact optimal branch cost cannot be calculated. Hence we introduce the lower bound of the branch cost, and use the DFSBnB technique (Poole and Mackworth, 2010) to prune some of the branches in the tree. For the time-optimal and energy-optimal planning, we find the upper bound of the branch cost as follows. Let us define the maximum and minimum distance between two junctions x_{i-1} and x_i ,

$$\begin{aligned} d_{\max} &= \max_{x_i, x_{i-1}} \|x_i - x_{i-1}\|, \\ \text{s.t. } f_{c_i, c_{i+1}}(x_i) &= 0, f_{c_{i-1}, c_i}(x_{i-1}) = 0, \\ d_{\min} &= \min_{x_i, x_{i-1}} \|x_i - x_{i-1}\|, \\ \text{s.t. } f_{c_i, c_{i+1}}(x_i) &= 0, f_{c_{i-1}, c_i}(x_{i-1}) = 0. \end{aligned}$$

We can find a lower bound on the minimum time spent on the path segment,

$$\begin{aligned} g_i^t &\geq \frac{1}{V^2 - \|u_{c_i}\|^2} (\|x_i - x_{i-1}\| V \\ &\quad - \sqrt{\|x_i - x_{i-1}\|^2 \|u_{c_i}\|^2 - ((x_i - x_{i-1})^T u_{c_i})^2}) \\ &\quad - (x_i - x_{i-1})^T u_{c_i} \\ &\geq \frac{1}{V^2 - \|u_{c_i}\|^2} (d_{\min} V - \|(x_i - x_{i-1})\| \|u_{c_i}\| \\ &\quad + \|(x_i - x_{i-1})^T u_{c_i}\| - (x_i - x_{i-1})^T u_{c_i}) \\ &\geq \frac{d_{\min}}{V + \|u_{c_i}\|} \triangleq g_{i,lb}^t. \end{aligned} \quad (16)$$

This lower bound is the travel time when the vehicle travels in the largest possible total speed $V + \|u_{c_i}\|$, which is the situation that u_{c_i} is in the same direction as the shortest path segment from x_{i-1} to x_i .

Similarly, given (7) we find a lower bound, denoted as $g_{i,lb}^e$ on the energy spent in one cell,

$$g_{i,lb}^e = \max\{2d_{\min}\sqrt{\|u_{c_i}^2 + C\|} - 2d_{\max}\|u_{c_i}\|, 0\}. \quad (17)$$

3.3.4 Branch-and-Bound method

We use the DFS algorithm to iteratively generate the nodes, starting from the root and terminates when it reaches the destination node. After the first cell sequence connecting the root with the destination node is found, we use ID algorithm to compute the optimal junctions that result in the minimum total travel cost (Line 12). To avoid traversing all feasible cell sequences, the algorithm maintains the lowest-cost path

to the target found so far, and its cost. At each step of node generation, for each of the adjacent cell m_i of the current node n_c , we compute a lower bound of the total cost of arrival, from the root to v_i ,

$$f_g(m_i) = f_g(n_c) + g_{i,lb}, \quad (18)$$

where for the time-optimal planning, $g_{i,lb} = g_{i,lb}^t$ is computed by (16), and for the energy-optimal planning, $g_{i,lb} = g_{i,lb}^e$ is computed by (17). If $f_g(m_i)$ is larger than the lowest-cost path found so far, then all the path that goes through the cell boundary represented by m_i cannot be the optimal solution, since its total cost will be larger than the lowest-cost path found so far. Thus we stop the DFS from m_i to its child nodes, and go to the next adjacent cell of n_c to continue the search (Line 24).

Algorithm 1: Main algorithm

Data: initial position x_0 , final position x_f , partitioned cell

$\{R_\alpha\}_{\alpha \in I_R}$

Output: optimal junction position $x(\lambda^{opt})$

```

1 visited ← {FALSE};
2 C ← {};
3 CostArrival = 0;
4 TotalCost_ub = 0;
5 Start node  $s = [c_0, c_1]$ , goal node  $d = [c_n, c_{n+1}]$ ;
6  $\lambda^{opt} = \text{findAllCellSeq}(s, d, \text{visited}, C, \text{adjacency}, f_g(s), \text{TotalCost\_ub})$ ;
7
8 Function findAllCellSeq ( $n_c, d, \text{visited}, C, f_g(n_c), \text{TotalCost\_ub}$ ):
9   visited( $n_c$ ) = TRUE;
10  C.append( $n_c$ );
11  if  $n_c = d$  then
12     $\lambda = \text{Intermittent\_Diffusion}(C)$ ;
13    if  $J(\lambda) < \text{TotalCost\_ub}$  or  $\text{TotalCost\_ub} = 0$  then
14      TotalCost_ub ←  $J(\lambda)$ ;
15       $\lambda^{opt} = \lambda$ ;
16    end
17  end
18  else
19    for all adjacent node  $\{m_j\}_{j=1}^M$  of  $n_c$  do
20      if visited( $m_j$ ) = FALSE then
21        Compute  $g_{j,lb}$  using (16) or (17);
22        Compute  $f_g(m_j)$  using (18);
23        if  $f_g(m_j) > \text{TotalCost\_ub}$  and  $\text{TotalCost\_ub} \neq 0$  then
24          continue;
25        end
26        findAllCellSeq( $m_j, d, \text{visited}, C, f_g(m_j) + g_{j,lb}, \text{TotalCost\_ub}$ );
27      end
28    end
29    C.pop;
30    visited( $n_c$ ) = FALSE;
31  end

```

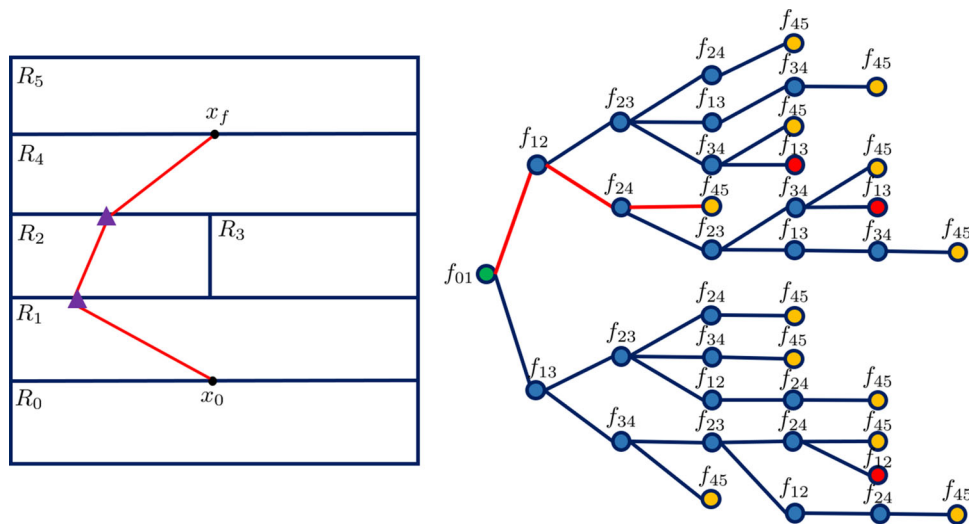


Fig. 2 (Left): Example of a partitioned space. The domain is partitioned into 6 cells, the red line represents one feasible path from the start to the destination position, with the junctions represented by the purple triangles. (Right): The entire decision tree for this partitioned space. The root node shown by the green circle is the boundary curve containing the starting position, and the terminal node (yellow circle) represents

the boundary curve containing the destination node. For each node, it is connected to its child node if it is an adjacent boundary to its child. The red nodes represent the nodes with no unvisited neighboring nodes. The red path in the graph corresponds to the cell sequence crossed by the feasible path shown on the left figure (Color figure online)

Algorithm 2: Intermittent Diffusion

Data: cell sequence $\mathcal{C}$
Output: the optimal junction position λ^* given the fixed cell sequence $\mathcal{C}$

- 1 Initialize $\lambda^0 = 0$;
- 2 Set evolution step number N ;
- 3 Choose threshold ϵ ;
- 4 **for** $i = 1, \dots, N$ **do**
- 5 Choose perturbation duration T_i ;
- 6 Choose perturbation intensity σ_i ;
- 7 **for** $j = 1, \dots, T_i$ **do**
- 8 Update λ^j using (15);
- 9 **end**
- 10 Set $\sigma_i = 0$;
- 11 **while not converges do**
- 12 Update λ^i using (15);
- 13 **end**
- 14 **end**
- 15 Set $\lambda^* = \arg \min_{i \leq N} J(\lambda^i)$;

3.4 Complexity analysis

In this subsection, we conduct the time complexity analysis of the proposed algorithm, and compare it with other graph search algorithms, such as the A* method. Let us assume that the flow field forecast is available on $N \times N$ grid points in the domain, and the OPEN set of A* algorithm is implemented using a heap data structure. Then each iteration of A* node expansion has a worst case running time of $\mathcal{O}(\log m N^2)$, where m is the number of neighbors to be expanded for each node, and there are $N^2 m$ nodes in the OPEN set in the worst case. Hence the worst case running time of A* is $\mathcal{O}(m N^2 \log(m N^2))$. For the proposed algorithm, we assume

the domain is partitioned into K number of cells in total, and each boundary curve has the number of adjacent boundaries no more than B . We first derive the worst case running time. If the computation of Algorithm 2 can be performed in a constant time, and the depth of optimal solution is D . Then the worst case running time of Algorithm 1 can be found by considering the scenario that each branch of the decision tree has to be searched at each iteration of decision tree expansion. This leads to the worst case time complexity $\mathcal{O}(B^D)$. To make the worst case running time of iBnBDFS comparable to that of A*, we let $B^D \leq m N^2 \log m N^2$. As the depth of the decision tree D of the proposed algorithm must be smaller than the total number of partitions K in the domain, the above inequality leads to

$$K \leq 2 \frac{\log N}{\log B} + \frac{\log m + \log \log m N^2}{\log B}.$$

Although the second term on the right hand side depends on N , it is much smaller than $\frac{\log N}{\log B}$. Hence the worst-case computation time of the iBnBDFS algorithm can be lower or comparable to that of A* if $K \leq 2 \frac{\log N}{\log B}$. Since the worst case iBnBDFS running time is the same as the worst case of the brute force DFS method, this is a conservative bound.

4 Completeness

In this section, we demonstrate that Algorithm 1 is complete if $L(v)$ is a convex function of v .

Theorem 4.1 *If the flow field is piece-wise constant and*

$$\max_{\alpha \in I_R} \|u_\alpha\| < V, \quad (19)$$

let Q be the set of global minimizers, U be a neighborhood of Q . Then for any $\epsilon > 0$, there exists T_0, N_0, σ_0 such that if $T_i > T_0, \sigma_i < \sigma_0$ (for $i = 1, 2, \dots, N$) and $N > N_0$ where T_i, σ_i, N are parameters in Algorithm 2, $\mathbb{P}(\lambda^{opt} \in U) \geq 1 - \epsilon$, where λ^{opt} is the optimal solution found by Algorithm 1. Thus, Algorithm 1 is complete.

The idea is that by Bellman principle, optimal trajectory admits an optimal sub-structure property, that is, any piece of the optimal trajectory is also optimal for the sub-problem. By applying this principle, we consider the path segment in each single region, and try to construct a solution ψ with two types of objective function described in Sect. 2, for the Hamilton-Jacobi-Bellman equation (HJB)

$$\psi_t(x, t) + H(x, \nabla \psi(x, t)) = 0 \quad (20)$$

where

$$H(x, p) = \max_{\|v\| \leq V} \{p^T(v + u) - L(x, v)\}.$$

is the Hamiltonian and

$$\psi(x, t) = \min_v \left\{ \int_0^t L(\gamma, v) ds : \dot{\gamma} = v + u, \gamma(0) = x_0, \right. \\ \left. \gamma(t) = x, \max_{s \in [0, t]} v(s) \leq V \right\}$$

is the value function. Since the original problem takes the minimum over all possible time, we take $\min_t \psi(x, t)$ to get the optimizer in the given region and claim that the corresponding motion gives a global optimal for the sub-problem in the single region. Hence in the following we give optimality proof of solution of the sub-problem in each constant flow region. Given the optimal solution of the sub-problem in each constant flow region, proof of Theorem 3.1 is provided in the Appendix.

4.1 Total travel time

$L(x, v) = 1$ for total travel time minimization. To construct the value function at the point (x, t) , we introduce the maximum speed constant velocity motion in the region with flow velocity u , that is, in this region, the vehicle moves in straight line from x_0 to x with velocity $v + u$ and $\|v\| = V, \|v + u\|$ is given by (4). We claim that

Lemma 4.2 *In a constant flow field, the maximum speed straight line motion is optimal if we minimize the total travel*

time

$$\min_{v, T} \int_0^T dt \\ s.t. \quad \dot{x} = v + u, \\ x(0) = x_0, \\ x(T) = x_f, \\ \max_{t \in [0, T]} \|v\| \leq V.$$

4.2 Quadratic energy with a constant running cost

Lemma 4.3 *In a constant flow field, the minimizer of energy optimal problem*

$$\min_{v, T} \int_0^T \|v\|^2 + C dt \\ s.t. \quad \dot{x} = v + u, \\ x(0) = x_0, \\ x(T) = x_f, \\ \max_{t \in [0, T]} \|v\| \leq V$$

is the constant velocity motion in the speed of $\|v\|$, where

$$\|v\| = \begin{cases} \left(C + 2\|u\|^2 \right. \\ \left. - \sqrt{C + \|u\|^2} \frac{2(x - x_0)^T u}{\|x - x_0\|} \right)^{1/2} & \text{if (9) holds} \\ V & \text{otherwise} \end{cases}.$$

Remark 4.1 When (9) does not hold, the minimizer of the energy optimal problem is the same as the minimizer of travel time optimal one. Hence, if the constant running cost C is large enough, solving the energy optimal problem is equivalent to solving the travel time optimal problem.

Based on the proof of Lemmas 4.2 and 4.3, we can have the following theorem, which tells the optimal path structure within each constant flow region, given entrance and exit locations.

Theorem 4.4 *In each constant flow region, given the entrance and exit locations, the vehicle motion defined in Lemmas 4.2 and 4.3 solves the HJB equation*

$$\psi_t(x, t) + \max_v \{ \nabla \psi(x, t)^T (u + v) - L(x, v) \} = 0$$

for $L = 1$ and $L = \|v\|^2 + C$ respectively. Moreover, among all the solutions of the above HJB, motion in these two lemmas gives the path with shortest time/minimum energy. Thus, we have the optimal solution of the sub-problem in each region.

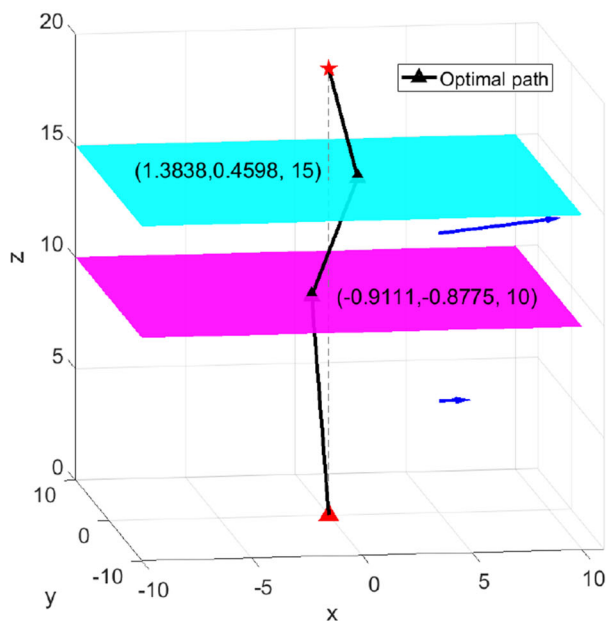
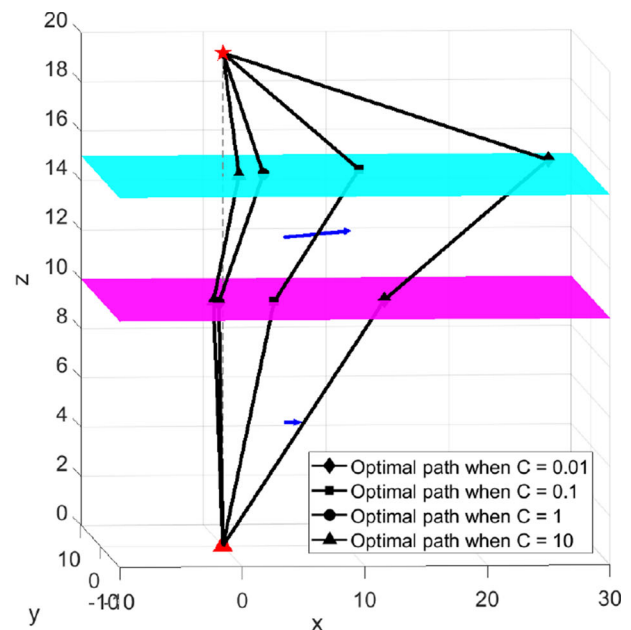


Fig. 3 (Left): Time optimal path planned by the proposed method. (Right): Energy optimal path planned by the proposed method. In both plots, boundaries of the jet flow are denoted by the colored surfaces. The flow speed in the domain is represented by the blue arrows. The



optimal path is marked by black line, while the marker position denotes junction points computed by the proposed method (Color figure online)

4.3 A general convex Lagrangian

In general, if we only assume the Lagrangian $L = L(v)$ is a convex function and the dynamics is $\dot{x} = f(u + v)$ where f is invertible and 0 is in the range of f (there exists some y with $\|y\| < +\infty$ such that $f(y) = 0$), we can have similar optimal path within a constant flow field and the result is stated in Theorem 3.1.

5 Simulation results

In this section, we provide simulations to validate the strength of the proposed method. First, the time-optimal and energy-optimal path planning examples with vehicle travel in simple canonical time flow field are presented. This example serves as a benchmark example wherein, we compare the solution obtained by our algorithm to solution derived from other path planning methods. Then we present a path planning example of using the proposed method to plan the time-optimal and energy-optimal path in a realistic ocean surface flow field. This simulation is intended to verify the performance of the proposed method in a highly complicated and strong real ocean flow field.

5.1 Jet flow in 3D space

For this benchmark example, we present path planning using the proposed method in a jet flow in 3D space. The domain

consists of three regions, divided by two boundary surfaces, $z = 10$ and $z = 15$. In the region where $z \in (0, 10)$, the flow speed is $(0.5, 0, 0)$. There is strong jet flow in the region where $z \in (10, 15)$ with flow speed $(2, 1, 0)$. The flow speed is zero in the region where $z \in (15, 20)$. The starting position is assigned at the origin, while the goal position is assigned at $(0, 0, 20)$.

The left figure in Fig. 3 shows the time-optimal path planned by the proposed method. The time-optimal solution is compared with the time-optimal path planned by the LSM and the A* method. The comparison result is shown in Table 1. In this comparison, assuming the path segment $x_{i+1} - x_i$ travels from the boundary surface $f_{\alpha_i \beta_i}$ to reach the boundary surface $f_{\alpha_{i+1} \beta_{i+1}}$, we define θ_i as the angle between path segment $x_{i+1} - x_i$ and the boundary surface $f_{\alpha_i \beta_i}$. γ_i is defined as the angle between the projection of $x_{i+1} - x_i$ on the boundary surface $f_{\alpha_i \beta_i}$ and the x-axis of $f_{\alpha_i \beta_i}$, $\theta_i \in (0, 90^\circ]$, $\gamma_i \in (-180^\circ, 180^\circ]$. From the table, θ_i and γ_i computed from the proposed method and the LSM are similar, with approximately 1° difference. Travel time of the optimal path planned by the proposed method and LSM are also approximately the same. Since A* method discretizes the domain and only search for the optimal path on the grid points, comparing the angles of the planned path between the proposed algorithm and A* would yield little or no insight in performance of the algorithms. Thus we only compare their optimal total cost. The value found by the A* method is 7.1298, which is comparable to the optimal solution found

Table 1 Comparison between using the proposed method, LSM for time-optimal path planning

	Proposed method	LSM
θ_1	82.7924	83.5659
θ_2	62.0255	63.3118
θ_3	73.7397	73.8027
γ_1	-136.0775	-135.6592
γ_2	30.2293	30.2407
γ_3	-161.6199	-161.2246
Total cost	6.9096	6.9826

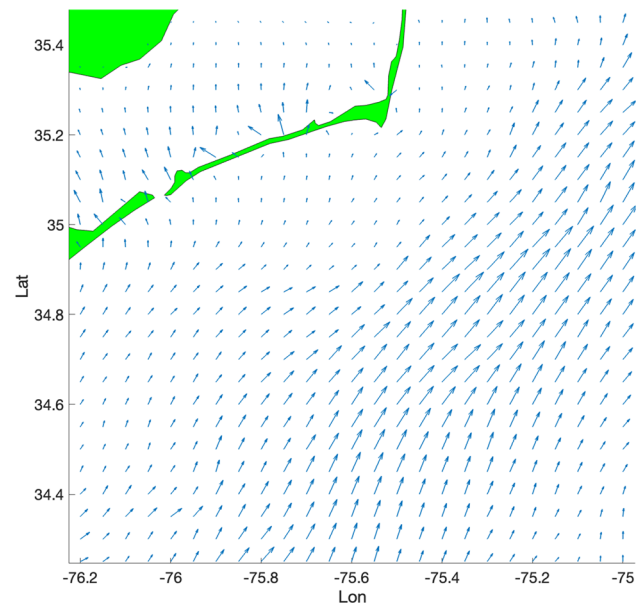
by the proposed algorithm. This shows that the proposed algorithm converges to the optimal solution.

The energy-optimal path is shown in the right figure of Fig. 3. The energy-optimal path when $C = 10$ is exactly the same as the time-optimal path. As the assigned C decreases, the energy cost is attached relatively more weight in the cost function. Therefore, the vehicle tends to save more energy to go with the flow in the bottom region and the jet flow region. Thus, the energy-optimal path deviates from the time-optimal path as C decreases.

5.2 Surface ocean flow

In this section we present path planning simulation of an underwater glider traveling in real ocean surface flow field near Cape Hatteras, North Carolina, a highly dynamic region characterized by confluent western boundary currents and convergence in the adjacent shelf and slope waters. While deployed, the glider is subject to rich and complex current fields driven by a combination and interaction of Gulf Stream, wind, and buoyancy forcing, with significant cross-shelf exchange on small spatial scales that is highly challenging for planning algorithms. While the energy efficiency of the glider's propulsion mechanism permits endurance of weeks to months, the forward speed of the vehicles is fairly limited (0.25–0.30 m/s), which can create significant challenges for navigation in strong currents. Use of a thruster in a so-called “hybrid” glider configuration can increase forward speed to approximately 1 m/s Ji et al. (2019), but at great energetic cost. The continental shelf near Cape Hatteras is strongly influenced by the presence of the Gulf Stream, which periodically intrudes onto the shelf, resulting in strong and spatially variable flow that can be nearly an order of magnitude greater than the forward speed of the vehicle (2+ m/s). Due to the high flow speed, we consider the deployment of a hybrid underwater glider in this simulation, and consider the vehicle speed $V = 1$ m/s.

The input flow map for path planning is given by a 1-km horizontal resolution version of the Navy Coastal Ocean

**Fig. 4** Surface ocean flow field on May 27, 2017, 00:00 UTC at Cape Hatteras, NC

Model (NCOM) (Martin, 2000) made available by J. Book and J. Osborne (Naval Research Laboratory, Stennis Space Center). The domain contains 130×130 grid points. One snapshot of the dynamic flow field is shown in Fig. 4. We partition the flow field using the algorithm proposed in Hou et al. (2019), and choose the number of partitioned cells using the “Elbow criterion” Shi et al. (2021). The domain is partitioned into 19 cells according to the “Elbow criterion”.

We perform 3 sets of simulation, with each set contains 10 test cases. The start and destination position are chosen such that the distance between the two points is 40, 100, or 130 km. For the 3 set of simulations, the computation cost and total travel cost is averaged over the 10 test cases. To verify performance of the proposed algorithm, we compare its simulation result with the A* and the LSM. Both A* and LSM run on the 130×130 rectangular grid cells. To avoid the incompleteness issue of A* (Kularatne et al., 2018; Soullignac, 2011), in the node generation process of A*, we consider each grid point have 16 neighboring nodes. The planned path of each test case is shown in Figs. 5, 6, and 7. Since the input to the proposed algorithm is the partitioned flow field, while the input to the LSM and the A* algorithm is the grid represented flow field, thus the optimal path computed by the proposed algorithm is different from the result of LSM and A*. However, as shown in Fig. 8, solution quality of the proposed algorithm is comparable to that of the A* and the LSM. Note that even though the proposed algorithm computes the planned path in the partitioned flow field, we compute the total travel cost of the vehicle tracking the planned path in the original flow field given by NCOM.

For all 3 sets of simulation, the proposed algorithm takes less computation time to compute the optimal solution to the

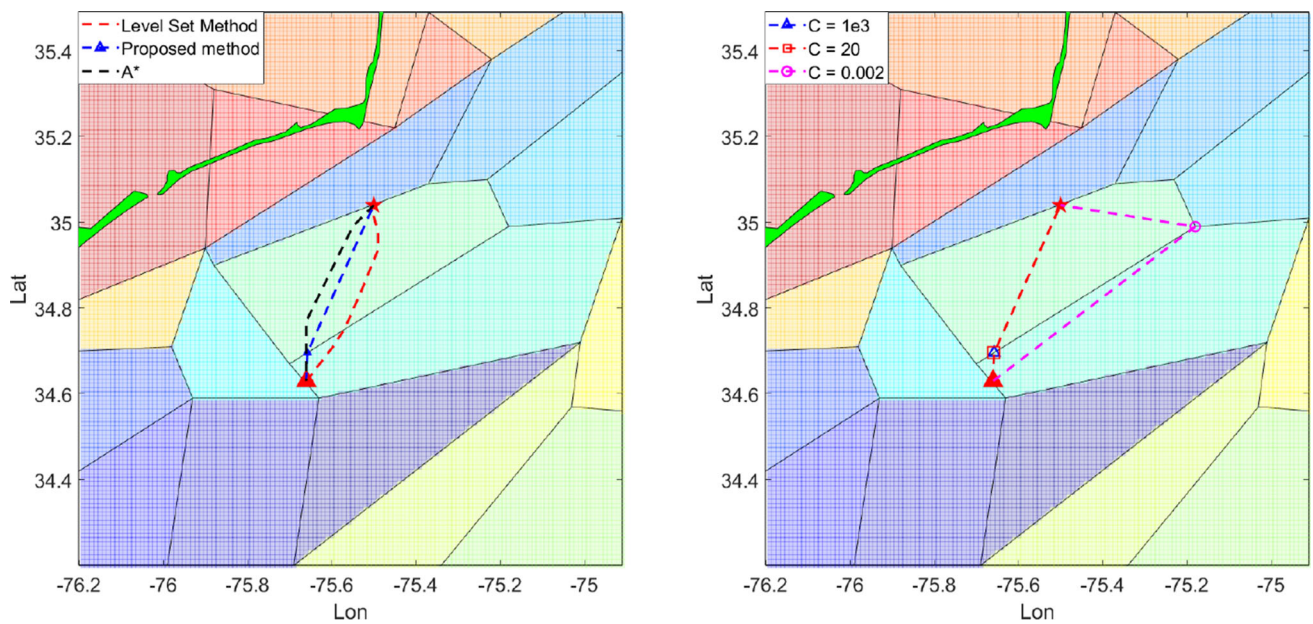


Fig. 5 Optimal path when $d = 40$ km. The colored cells represent the partitioned cells. (Left): Time optimal path planned by the proposed method, the A* method and the LSM. (Right): Energy optimal path planned by the proposed method, when C takes different value.

The optimal path is marked by colored line, while the marker position denotes junction points computed by the proposed method (Color figure online)

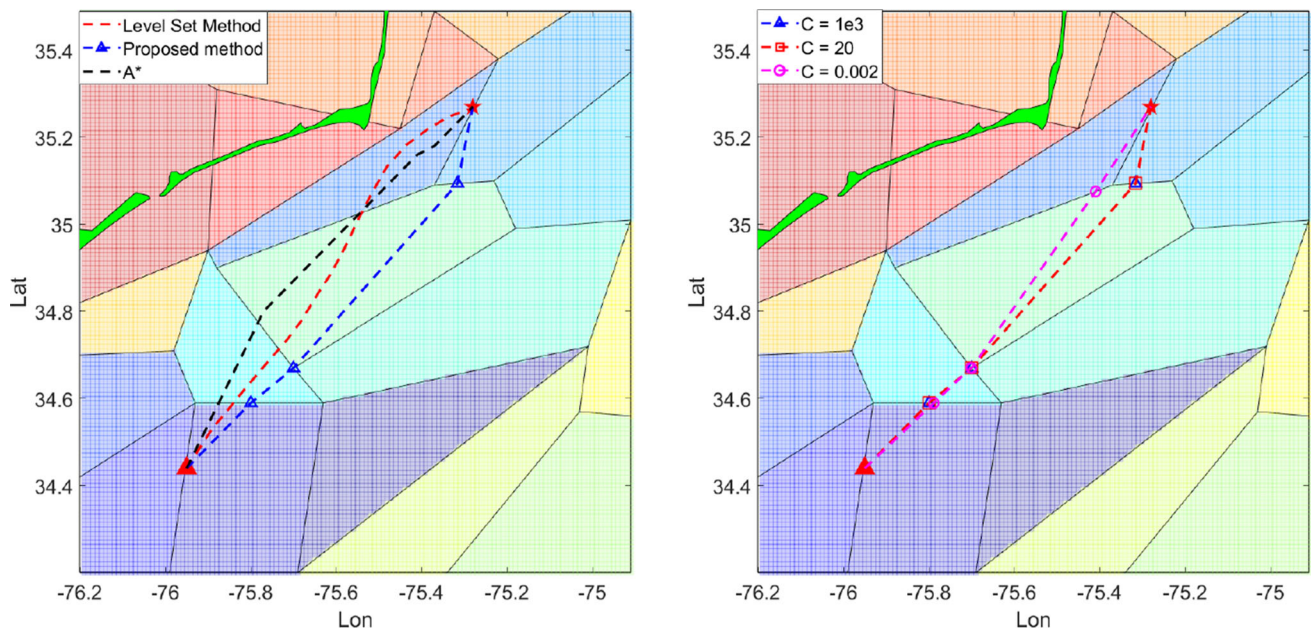


Fig. 6 Optimal path when $d = 100$ km. (Left): Time optimal path planned by the proposed method, the A* method and the LSM. (Right): Energy optimal path planned by the proposed method, when C takes different value

minimum-time planning problem, when the flow field is partitioned into 19 cells according to the Elbow criterion. Even though the total cost of the optimal solution found by the proposed algorithm is slightly larger, about 2% – 9% larger than that of the A* and LSM, it takes more than 40% lower computation cost than the A* and the LSM. The proposed algorithm takes significantly less computation time in the simulation

sets with shorter distance between the start and the destination node. The reason is that with smaller d , the decision tree is shallow, and the proposed algorithm only need to search through a small number of nodes to find the optimal solution. In addition, we increase the number of partitions to 39 and 59 cells, to demonstrate how the computation cost of the proposed algorithm scales with respect to the level of parti-

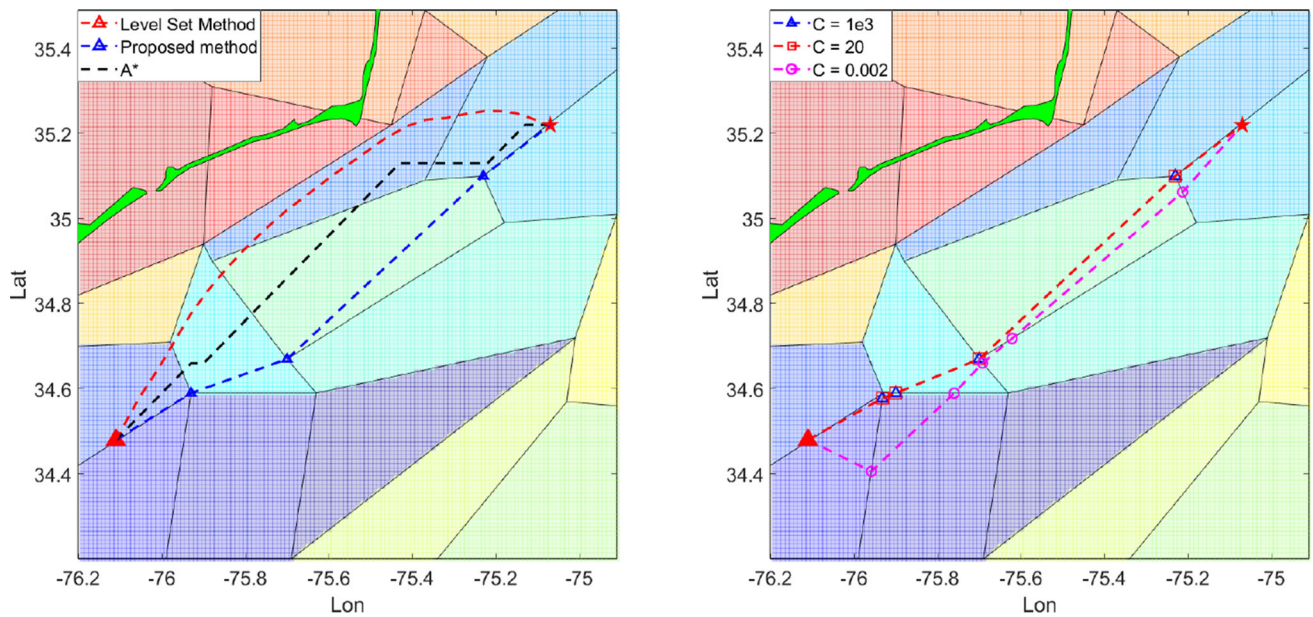


Fig. 7 Optimal path when $d = 130$ km. (Left): Time optimal path planned by the proposed method, the A* method and the LSM. (Right): Energy optimal path planned by the proposed method, when C takes different value

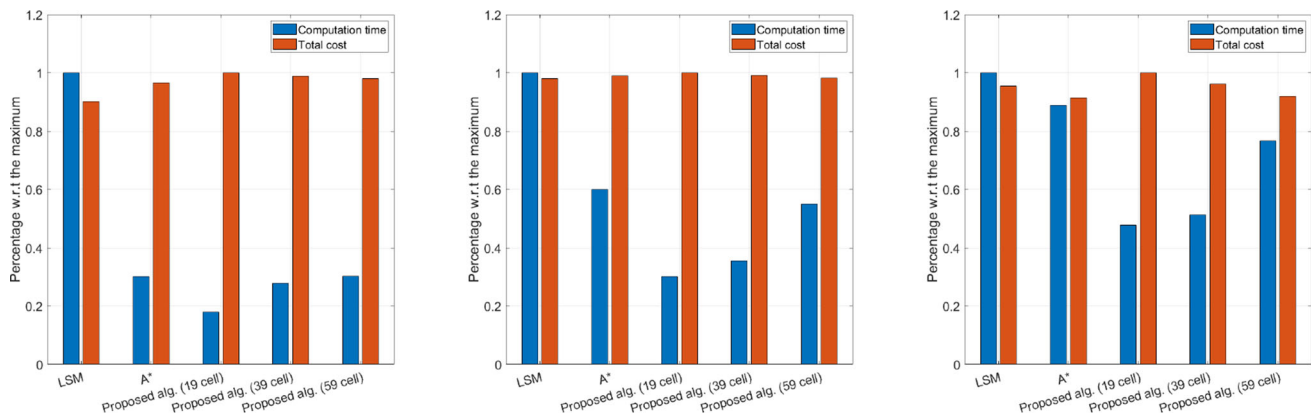


Fig. 8 Comparison between using the proposed method and existing algorithms for time-optimal path planning. We compare the LSM and the A* method with the proposed algorithm, in the cases when the flow field is partitioned into 19, 39 and 59 cells. The bar is showing the computation time and total cost of each algorithm. The computation time is

normalized by the largest computation time among all the algorithms (the largest computation time is normalized as 1), and the total cost is also normalized by the largest among all the algorithms (the largest total cost is normalized as 1). (Left): 40 km case; (Middle): 100 km case; (Right): 130 km case

tion. The proposed algorithm takes more time to compute the optimal solution as the number of partitions increases. Even though the more detailed partition gives improved solution quality, the improvement is marginal. Hence in order for the proposed algorithm to achieve good performance, it is desirable to choose the proper level of partitioning for the best trade-off between solution quality and computation cost.

The Energy optimal path is shown in Figs. 5, 6, and 7. The proposed algorithm generates same results when $C = 1e03$ and $C = 20$. In these cases, the running cost is much larger than the vehicle speed. Thus the energy-optimal planned path is identical to the time-optimal path. When $C = 2e - 3$, the

running cost is much less than vehicle speed. In this case, instead of making use of the strong jet flow, the proposed method plans a path different from the time-optimal one.

6 Conclusion

In this paper, we propose a new method using Method of Evolving Junctions to solve the AUV path planning problem in an arbitrary flow field with the dynamics being $\dot{x} = u + v$ where u is the flow field and v is the vehicle velocity. Taking advantage of the explicit solution in constant flow field being

straight line motion, we partition the flow field into piecewise constant vector field and transform the optimal control problem into a finite dimensional optimization, using Intermittent Diffusion method to get the global minimizer. In this way, we can get rid of the system error induced by discretizing the continuous space. Also, our method can be trivially extended to high dimensional general vehicle path planning problems in the same time complexity without making further assumption.

Acknowledgements The authors would like to thank the support from NSF Grants DMS-1830225, ONR Grant N00014-21-1-2891, ONR Grants N00014-19-1-2556 and N00014-19-1-2266; AFOSR Grant FA9550-19-1-0283; NSF Grants CNS-1828678, S&AS-1849228 and GCR-1934836; NRL Grants N00173-17-1-G001 and N00173-19-P-1412; and NOAA Grant NA16NOS0120028.

Appendix

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

First we present proof of Theorem 4.1. The proof leverages the following theorem in Chow et al. (2013).

Theorem 6.1 *Let Q be the set of global minimizers, U be a small neighborhood of Q and λ_{opt} the optimal solution obtained by the ID process. Then for any given $\epsilon > 0$, there exists $\tau > 0$, $\sigma_0 > 0$ and $N_0 > 0$ such that if $T_i - S_i > \tau$, $\sigma_i < \sigma_0$ (for $i = 1, \dots, N$) and $N > N_0$,*

$$\mathbb{P}(\lambda_{opt} \in U) \geq 1 - \epsilon.$$

Then we provide completeness proof of Algorithm 1.

Proof of Theorem 4.1 The proof includes two steps. First, we show that the decision tree returns all cell sequences with total cost less than or equal to the lowest-cost path found so far. Then we prove that given a fixed cell sequence, the global minimizer can be found by Algorithm 2.

The DFS algorithm, which avoids repeated states in the graph, is complete in finite state spaces (Russell and Norvig, 2002). In a static flow field divided into convex regions, the optimal path will not visit a cell boundary curve more than one time. Hence, the optimal path connecting the root and the target node in the decision tree does not contain loops. Therefore, the BnBDFS returns all cell sequences with total cost less than or equal to the lowest-cost found so far.

Next we show that the ID algorithm is complete. We combine Theorem 4.4, together with Bellman principle, to show that the global optimal path must be in the structure of constant motion within each flow region. To prove that the proposed algorithm is convergent, we only need to show that there exists a global minimizer $\lambda^* = (\lambda_1^*, \dots, \lambda_K^*)$, around which there is a closed neighborhood $U \subset \prod_{i=1}^K D_i$ such

that $\text{vol}(U) > 0$ (vol is the product Lebesgue measure in $\prod_{i=1}^K D_i$) and for all $\lambda \in U$, the gradient flow $\dot{\lambda} = -\nabla J(\lambda)$ converges to λ^* . If this condition holds, we can follow the proof of intermittent diffusion and get the desired results.

To this end, if there exists such U that $\text{vol}(U) > 0$ and for all $\lambda \in U$, we have $J(\lambda) \leq J(\mu)$ for arbitrary $\mu \in S$ for some $S \subset U$, then the proof is done. Now if the global minimizers are isolated, then given any global minimizer $\lambda^* = (\lambda_1^*, \dots, \lambda_K^*)$, since J is continuous differentiable, we can have a closed neighborhood $U \subset \prod_{i=1}^K D_i$ with $\text{vol}(U) > 0$ (within the neighborhood, the dimension of the domain does not change) such that $J(\lambda) > J(\lambda^*)$ and $\nabla J(\lambda) \neq 0$ for all $\lambda \in U \setminus \{\lambda^*\}$, then the gradient flow starting at $\lambda \in U$ converges to λ^* .

Therefore, we prove the algorithm is complete. $\square$

Proof of Lemma 4.2 We write the value function as

$$\psi(x, t) = \frac{\|x - x_0\|}{\|v + u\|}.$$

To make the problem complete, we define $\psi(x, t) = +\infty$ if the vehicle cannot reach x in time t , which gives the final value function to be

$$\psi(x, t) = \begin{cases} \frac{\|x - x_0\|}{\|v + u\|} & \frac{\|x - x_0\|}{\|v + u\|} \leq t \\ +\infty & \text{otherwise} \end{cases}.$$

If only the reachable part is considered, from the above equation, we can calculate $\psi_t = 0$ and

$$\nabla \psi = \frac{1}{\|v + u\|^2} \left(\|v + u\| \frac{x - x_0}{\|x - x_0\|} - \|x - x_0\| \nabla \|v + u\| \right). \quad (21)$$

We can rewrite $v = v^0 + v^\perp$ and $V^2 = \|v^0\|^2 + \|v^\perp\|^2$ if we denote

$$v^0 = \frac{(x - x_0)(x - x_0)^T v}{\|x - x_0\| \|x - x_0\|},$$

$$v^\perp = \left(I - \frac{(x - x_0)(x - x_0)^T}{\|x - x_0\| \|x - x_0\|} \right) v,$$

where I is the identity matrix. And u can be decomposed in the same manner $u = u^0 + u^\perp$. It is easy to see that $v^\perp = -u^\perp$ since $(v + u)/\|v + u\| = (x - x_0)/\|x - x_0\|$. Then, we see that $\|v + u\| = \|v^0\| + \|u^0\|$ and

$$\sqrt{\left(\frac{(x - x_0)^T u}{\|x - x_0\|} \right)^2 + V^2 - \|u\|^2}$$

$$= \left(\|u^0\|^2 + \|v^0\|^2 + \|v^\perp\|^2 - \|u^0\|^2 - \|u^\perp\|^2 \right)^{1/2}$$

$$= \|v^0\|.$$

Hence, we have

$$\begin{aligned}
& \nabla \|v + u\| \\
&= \nabla \left(\frac{(x - x_0)^T u}{\|x - x_0\|} + \sqrt{\left(\frac{(x - x_0)^T u}{\|x - x_0\|} \right)^2 + V^2 - \|u\|^2} \right) \\
&= \frac{\|v + u\|}{\|v^0\|} \nabla \frac{(x - x_0)^T u}{\|x - x_0\|} \\
&= \frac{\|v + u\|}{\|x - x_0\| \|v^0\|} \left(I - \frac{(x - x_0)(x - x_0)^T}{\|x - x_0\|^2} \right) u \\
&= \frac{\|v + u\|}{\|x - x_0\| \|v^0\|} u^\perp.
\end{aligned}$$

Taking $\nabla \|v + u\|$ back to (21) and noticing that $v^\perp = -u^\perp$, we reduce the gradient to be

$$\nabla \psi = \left(\frac{x - x_0}{\|x - x_0\|} - \frac{u^\perp}{\|v^0\|} \right) \frac{1}{\|v + u\|} = \frac{v}{\|v^0\| \|v + u\|}.$$

With the above equation, the Hamiltonian is

$$\begin{aligned}
H &= \sup_{\hat{v}: \|\hat{v}\| \leq V} \left(\nabla \psi^T (\hat{v} + u) - 1 \right) \\
&= \sup_{\hat{v}: \|\hat{v}\| \leq V} \left\{ \frac{v^T}{\|v^0\|} \frac{\hat{v} + u}{\|v + u\|} \right\} - 1 \\
&= \frac{1}{\|v + u\|} \left(\sup_{\hat{v}: \|\hat{v}\| \leq V} \left\{ \frac{v^T \hat{v}}{\|v^0\|} \right\} + \|u_0\| - \frac{\|u^\perp\|^2}{\|v^0\|} \right) - 1 \\
&= \frac{1}{\|v + u\|} \left(\frac{V^2}{\|v^0\|} + \|u^0\| - \frac{\|u^\perp\|^2}{\|v^0\|} \right) - 1 \\
&= \frac{1}{\|v + u\|} (\|v^0\| + \|u^0\|) - 1 = 0,
\end{aligned}$$

which leads to the conclusion that the value function induced by the maximum speed constant velocity motion solves the Hamilton-Jacobi equation, thus is the optimal moving pattern in a constant flow speed region since $\min_t \psi = \psi$. $\square$

Meanwhile, using the same notation and logic, we can give the proof of Proposition 3.1:

Proof of Proposition 3.1 First we show that the objective function is well-defined if there exists a feasible trajectory, and $\|u_{c_i}\| \leq V$. If $(x_i - x_{i-1})^T u_{c_i} \leq 0$, unless $V > \|u_{c_i}\|$, there does not exist a feasible path. Therefore,

$$\begin{aligned}
& (x_i - x_{i-1})^T u_{c_i} < \\
& \sqrt{\left((x_i - x_{i-1})^T u_{c_i} \right)^2 + \|x_i - x_{i-1}\|^2 (V^2 - \|u_{c_i}\|^2)}.
\end{aligned}$$

Since $\|u_{c_i}\|^2 - V^2 < 0$, we have $t_i^* > 0$. On the other hand, if $(x_i - x_{i-1})^T u_{c_i} > 0$, we can have two cases: $V > \|u_{c_i}\|$,

which shares the same conclusion as the first case, and $V < \|u_{c_i}\|$. In the latter circumstance, since $\|u_{c_i}\|^2 - V^2 > 0$ and

$$\begin{aligned}
& (x_i - x_{i-1})^T u_{c_i} > \\
& \sqrt{\left((x_i - x_{i-1})^T u_{c_i} \right)^2 + \|x_i - x_{i-1}\|^2 (V^2 - \|u_{c_i}\|^2)},
\end{aligned}$$

it is still true that $t_i^* > 0$.

Meanwhile, When $(x_i - x_{i-1})^T u_{c_i} > 0$, $t_i^* > 0$ still holds if $V = \|u_{c_i}\|$ and actually

$$\begin{aligned}
t_i^* &= \lim_{V^2 - \|u_{c_i}\|^2 \rightarrow 0} \frac{1}{\|u_{c_i}\|^2 - V^2} \left((x_i - x_{i-1})^T u_{c_i} \right. \\
&\quad \left. - \sqrt{\left((x_i - x_{i-1})^T u_{c_i} \right)^2 + \|x_i - x_{i-1}\|^2 (V^2 - \|u_{c_i}\|^2)} \right) \\
&= \frac{\|x_i - x_{i-1}\|^2}{2(x_i - x_{i-1})^T u_{c_i}} > 0.
\end{aligned}$$

However, if $(x_i - x_{i-1})^T u_{c_i} \leq 0$, $V = \|u_{c_i}\|$ becomes a singular point since there is no feasible path. Thus, in this case, we cannot formally solve the problem.

Since $g_i^t(x_i, x_{i-1}) = g_i^t(x_i - x_{i-1})$ and $g_i^e(x_i, x_{i-1}) = g_i^e(x_i - x_{i-1})$, we only need to consider the differentiability of

$$g(a) = \begin{cases} g_1(a) & \text{if (10) holds} \\ g_2(a) & \text{otherwise} \end{cases}$$

where

$$\begin{aligned}
g_1(a) &= 2\sqrt{\|u_{c_i}\|^2 + C} \|a\| - 2a^T u_{c_i} \\
&= 2\|a\| \left(\sqrt{\|u_{c_i}\|^2 + C} - \|u_{c_i}^0\| \right), \\
g_2(a) &= \frac{V^2 + C}{\|u_{c_i}\|^2 - V^2} \left(a^T u_{c_i} \right. \\
&\quad \left. - \sqrt{(a^T u_{c_i})^2 + \|a\|^2 (V^2 - \|u_{c_i}\|^2)} \right) \\
&= \frac{(V^2 + C)\|a\|}{\|u_{c_i}^0\| + \|v^0\|}.
\end{aligned}$$

First of all, when equality in (10) holds, we have

$$\begin{aligned}
& \sqrt{\|u_{c_i}\|^2 + C} = \|u_{c_i}^0\| \pm \|v^0\| \\
& \implies \sqrt{\|u_{c_i}\|^2 + C} = \|u_{c_i}^0\| + \|v^0\|.
\end{aligned} \tag{22}$$

We take the plus sign since $\sqrt{\|u_{c_i}\|^2 + C} \geq \|u_{c_i}\|$. Meanwhile from (10) and (22), we can derive the following equation

$$V^2 + C = 2\|v^0\|(\|u_{c_i}^0\| + \|v^0\|).$$

Therefore, $g(a)$ is continuous. Similar calculations shows that

$$\nabla g_1 = 2 \left((\sqrt{\|u_{c_i}\|^2 + C}) \frac{a}{\|a\|} - u_{c_i} \right) = v^0 - u_{c_i}^\perp = 2v,$$

$$\nabla g_2 = \frac{V^2 + C}{\|v^0\| \|v + u_{c_i}\|} v = 2v,$$

which gives us the desired result. $\square$

Proof of Lemma 4.3 In the case of minimum energy planning, $L(x, v) = \|v\|^2 + C$ where $C \geq 0$ is a constant running cost. To calculate the optimal solution for the vehicle running from x_0 to the target x in a constant flow velocity field, we again study the constant speed straight line motion. However in this circumstance, the vehicle may no longer travel with maximum speed, hence we take the travel time in the region into consideration. Suppose that the vehicle moves from x_0 to x in time t , we set the vehicle velocity to be

$$v = \frac{x - x_0}{t} - u,$$

assuming that

$$\|v\|^2 = \frac{\|x - x_0\|^2}{t^2} + \|u\|^2 - \frac{2(x - x_0)^T u}{t} \leq V^2. \quad (23)$$

Then the value function is

$$\begin{aligned} \psi(x, t) &= (\|v\|^2 + C)t \\ &= \frac{\|x - x_0\|^2}{t} - 2(x - x_0)^T u + (C + \|u\|^2)t. \end{aligned}$$

Further we take

$$\psi(x, t) = \begin{cases} \frac{\|x - x_0\|^2}{t} - 2(x - x_0)^T u + (C + \|u\|^2)t & \|v\| \leq V \\ +\infty & \text{otherwise} \end{cases}.$$

Then by direct calculation with the finite part of ψ , we have

$$\psi_t = C + \|u\|^2 - \frac{\|x - x_0\|^2}{t^2}, \quad (24)$$

$$\nabla \psi = \frac{2(x - x_0)^T u}{t} - 2u, \quad (25)$$

$$\|\nabla \psi\|^2 = \frac{4\|x - x_0\|^2}{t^2} + 4\|u\|^2 - \frac{8(x - x_0)^T u}{t} \quad (26)$$

The Hamilton-Jacobi equation is in the form of

$$\psi_t + \sup_{v: \|v\| \leq V} \left\{ \nabla \psi^T (v + u) - \|v\|^2 - C \right\} = 0. \quad (27)$$

To solve the optimization part of (27), we denote

$$F(v) = \nabla \psi^T (v + u) - \|v\|^2 - C$$

and calculate its critical point as

$$v^* = \frac{1}{2} \nabla \psi,$$

which means that the optimal is

$$H = \sup_{v: \|v\| \leq V} F(v) = \frac{1}{4} \|\nabla \psi\|^2 + \nabla \psi^T u - C \quad (28)$$

and by (25), we have

$$\begin{aligned} \|v^*\|^2 &= \frac{1}{4} \|\nabla \psi\|^2 = \frac{\|x - x_0\|^2}{t^2} + \|u\|^2 - \frac{2(x - x_0)^T u}{t} \\ &\leq V^2, \end{aligned} \quad (29)$$

which leads to the fact that $F(v^*) = \sup_{v: \|v\| \leq V} F(v)$. Let us take (25), (26) into (28) and the result is

$$H = \frac{\|x - x_0\|^2}{t^2} - \|u\|^2 - C. \quad (30)$$

Combining (24) and (30) finally results in the constructed ψ being the solution of (27).

Based on the solution ψ , we further find the minimizer over time t and solve the minimization problem as follow:

$$\begin{aligned} \min_{t \geq 0} \psi &= \min_{t \geq 0} (\|v\|^2 + C)t \\ &= \frac{\|x - x_0\|^2}{t} - 2(x - x_0)^T u + (C + \|u\|^2)t. \end{aligned}$$

It is easy to see that the global minimizer of ψ over t is

$$t^* = \frac{\|x - x_0\|}{\sqrt{C + \|u\|^2}}$$

and the corresponding minimum is

$$\psi^* = 2\|x - x_0\| \sqrt{C + \|u\|^2} - 2(x - x_0)^T u. \quad (31)$$

Thus, if t^* is reachable, that is, using (23), we have

$$\|v(t^*)\|^2 = C + 2\|u\|^2 - \sqrt{C + \|u\|^2} \frac{2(x - x_0)^T u}{\|x - x_0\|} \leq V^2$$

the optimal is given as (31).

On the other hand, if $\|v(t^*)\| > V$, the global minimizer t^* is on longer in the domain of our problem. In this case, we notice that $\|v\|^2$ is decreasing on the interval

$$\left[t^*, \frac{(x - x_0)^T u}{\|x - x_0\|^2} \right],$$

and is increasing on

$$\left[\frac{(x - x_0)^T u}{\|x - x_0\|^2}, +\infty \right).$$

Also by noticing that $\lim_{t \rightarrow +\infty} \|v\|^2 = \|u\|^2 \leq V^2$, we conclude that there exists $t_0 > t^*$ when $t \geq t_0 > t^*$, $\|v\| \leq V$. Meanwhile, when $t > t^*$, ψ is monotone increasing with respect to t . Hence, to get the minimum, we should take the time $t = t_0$, where $\|v(t_0)\| = V$. By taking the equality in (23), we have then

$$(\|u\|^2 - V^2)t^2 - 2(x - x_0)^T u t + \|x - x_0\|^2 = 0,$$

from which we have

$$t_0 = \frac{1}{\|u\|^2 - V^2} \left((x - x_0)^T u - \sqrt{((x - x_0)^T u)^2 + \|x - x_0\|^2 (V^2 - \|u\|^2)} \right),$$

and $\min_t \psi = (V^2 + C)t_0$. $\square$

Proof of Theorem 3.1 Denoting $g(w)$ to be the inverse of f such that if $w = f(u + v)$ then $u + v = g(w)$, we will show that

$$\psi(x, t) = \begin{cases} tL\left(g\left(\frac{x-x_0}{t}\right) - u\right) & \|g\left(\frac{x-x_0}{t}\right) - u\| \leq V \\ +\infty & \text{otherwise} \end{cases}$$

satisfies the HJB equation (20). First of all, the Hessian matrix $\mathcal{H}(v)$ is positive definite for all $\|v\| \leq V$ since L is convex. Therefore, for any v_1, v_2 in the domain, there exists ξ such that

$$\nabla_v L(v_1) = \nabla_v L(v_2) + \mathcal{H}(\xi)(v_2 - v_1).$$

Further if $\nabla_v L(v_1) = \nabla_v L(v_2)$, then $\mathcal{H}(\xi)(v_2 - v_1) = 0$. Because of the positive definite property for $\mathcal{H}$, we have $v_2 = v_1$, which implies that $\nabla_v L(v)$ is one-to-one.

Then we do the following calculation on the non-infinity part of ψ

$$\begin{aligned} \psi_t &= L\left(g\left(\frac{x-x_0}{t}\right) - u\right) \\ &\quad - \left[\nabla_v L\left(g\left(\frac{x-x_0}{t}\right) - u\right) \right]^T \nabla_w g\left(\frac{x-x_0}{t}\right) \frac{x-x_0}{t}, \end{aligned} \quad (32)$$

$$\nabla \psi = \left[\nabla_w g\left(\frac{x-x_0}{t}\right) \right]^T \nabla_v L\left(g\left(\frac{x-x_0}{t}\right) - u\right). \quad (33)$$

Since L is convex, we further have the relaxed optimization

$$\max_v \{ \nabla \psi^T (v + u) - L(v) \}$$

is a convex problem and get the condition for the optimal v^* to be

$$\begin{aligned} [\nabla_v f(u + v^*)]^T \nabla \psi &= \nabla_v L(v^*) \\ \implies \nabla \psi &= [\nabla_w g(f(u + v^*))]^T \nabla_v L(v^*). \end{aligned}$$

Combining this with (33), we have

$$v^* = g\left(\frac{x-x_0}{t}\right) - u, \quad (34)$$

and $\|v^*\| \leq V$ holds. Thus, v^* is the maximizer of $H(x, \nabla \psi)$. Taking (32), (33) and (34), we have

$$\psi_t + \nabla \psi^T (v^* + u) - L(v^*) = 0,$$

implying that (20) holds. At last notice that

$$\lim_{t \rightarrow \infty} L\left(g\left(\frac{x-x_0}{t}\right) - u\right) = L(g(0) - u) < \infty,$$

since 0 is in the range of f . We have that

$$\lim_{t \rightarrow \infty} tL\left(g\left(\frac{x-x_0}{t}\right) - u\right) = \infty.$$

Thus, we have $t^* > 0$ such that given x ,

$$t^* = \arg \min_{t \geq 0} \psi(x, t).$$

Thus, $v^* = g((x - x_0)/t^*) - u$ gives us a constant velocity motion. $\square$

References

- Chen, Y., He, Z., & Li, S. (2019). Horizon-based lazy optimal RRT for fast, efficient replanning in dynamic environment. *Autonomous Robots*, 43(8), 2271–2292.
- Chow, S.N., Yang, T.S., & Zhou, H.M. (2013). Global optimizations by intermittent diffusion. In: *Chaos, CNN, Memristors and Beyond: A Festschrift for Leon Chua With DVD-ROM*, composed by Eleonora Bilotta. *World Scientific*, pp. 466–479.
- Forrest, J., Hirst, J., & Tomlin, J. A. (1974). Practical solution of large mixed integer programming problems with umpire. *Management Science*, 20(5), 736–773.

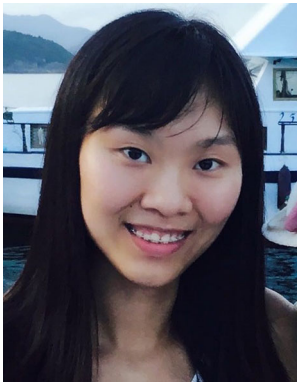
- Gammell, J. D., Barfoot, T. D., & Srinivasa, S. S. (2018). Informed sampling for asymptotically optimal path planning. *IEEE Transactions on Robotics*, 34(4), 966–984.
- Hou, M., Cho, S., & Zhou, H., et al. (2021). Bounded cost path planning for underwater vehicles assisted by a time-invariant partitioned flow field model. *Frontiers in Robotics and AI*, 8.
- Hou, M., Zhai, H., & Zhou, H., et al. (2019). Partitioning ocean flow field for underwater vehicle path planning. In: *OCEANS 2019-Marseille*, IEEE, pp. 1–8.
- Janson, L., Schmerling, E., Clark, A., et al. (2015). Fast marching tree: A fast marching sampling-based method for optimal motion planning in many dimensions. *The International Journal of Robotics Research*, 34(7), 883–921.
- Ji, D. H., Choi, H. S., Kang, J. I., et al. (2019). Design and control of hybrid underwater glider. *Advances in Mechanical Engineering*, 11(5), 1687814019848.
- Kaiser, E., Noack, B. R., Cordier, L., et al. (2014). Cluster-based reduced-order modelling of a mixing layer. *Journal of Fluid Mechanics*, 754, 365–414.
- Karaman, S., & Frazzoli, E. (2011). Sampling-based algorithms for optimal motion planning. *The International Journal of Robotics Research*, 30(7), 846–894.
- Kuffner, J., & LaValle, S. (2000). RRT-connect: An efficient approach to single-query path planning. In: *IEEE International Conference on Robotics and Automation*.
- Kularatne, D., Bhattacharya, S., & Hsieh, M. A. (2017). Optimal path planning in time-varying flows using adaptive discretization. *IEEE Robotics and Automation Letters*, 3(1), 458–465.
- Kularatne, D., Bhattacharya, S., & Hsieh, M. A. (2018). Going with the flow: A graph based approach to optimal path planning in general flows. *Autonomous Robots*, 42(7), 1369–1387.
- LaValle, S. M. (1998). *Rapidly-exploring random trees: A new tool for path planning*. Tech. rep.: Department of Computer Science, Iowa State University.
- Leonard, N. E., Paley, D. A., Davis, R. E., et al. (2010). Coordinated control of an underwater glider fleet in an adaptive ocean sampling field experiment in Monterey Bay. *Journal of Field Robotics*, 27(6), 718–740. <https://doi.org/10.1002/rob.20366>.
- Li, W., Lu, J., Zhou, H., et al. (2017). Method of evolving junctions: A new approach to optimal control with constraints. *Automatica*, 78, 72–78.
- Lolla, S.V.T. (2016). Path planning and adaptive sampling in the coastal ocean. *PhD thesis, Massachusetts Institute of Technology*.
- Martin, P.J. (2000). *Description of the navy coastal ocean model version 1.0*, Tech. Rep. NRL/FR/7322–00-9962, Naval Research Lab.
- Noreen, I., Khan, A., & Habib, Z. (2016). Optimal path planning using RRT* based approaches: a survey and future directions. *International Journal of Advanced Computer Science and Applications*, 7(11), 97–107.
- Ozog, P., Carlevaris-Bianco, N., Kim, A. Y., et al. (2016). Long-term mapping techniques for ship hull inspection and surveillance using an autonomous underwater vehicle. *Journal of Field Robotics*, 33, 265–289.
- Pereira, A. A., Binney, J., Hollinger, G. A., et al. (2013). Risk-aware path planning for autonomous underwater vehicles using predictive ocean models. *Journal of Field Robotics*, 30(5), 741–762. <https://doi.org/10.1002/rob.21472>.
- Poole, D. L., & Mackworth, A. K. (2010). *Artificial Intelligence: foundations of computational agents*. Cambridge University Press.
- Rhoads, B., Mezic, I., & Poje, A. C. (2012). Minimum time heading control of underpowered vehicles in time-varying ocean currents. *Ocean Engineering*, 66(1), 12–31.
- Russell, S., & Norvig, P. (2002). *Artificial intelligence: a modern approach*.
- Ser-Giacomi, E., Rossi, V., López, C., et al. (2015). Flow networks: A characterization of geophysical fluid transport. *Chaos: An Interdisciplinary Journal of Nonlinear Science*, 25(3), 036404.
- Sethian, J. A. (1999). *Level set methods and fast marching methods: Evolving interfaces in geometry, fluid mechanics*. Computer Vision and Material Science: Cambridge University Press.
- Shi, C., Wei, B., Wei, S., et al. (2021). (2021) A quantitative discriminant method of elbow point for the optimal number of clusters in clustering algorithm. *EURASIP Journal on Wireless Communications and Networking*, 1, 1–16.
- Shome, R., Solovey, K., Dobson, A., et al. (2020). dRRT*: Scalable and informed asymptotically-optimal multi-robot motion planning. *Autonomous Robots*, 44(3), 443–467.
- Sinha, A., Malo, P., & Deb, K. (2017). A review on bilevel optimization: from classical to evolutionary approaches and applications. *IEEE Transactions on Evolutionary Computation*, 22(2), 276–295.
- Smith, R. N., Chao, Y., Li, P. P., et al. (2010). Planning and implementing trajectories for autonomous underwater vehicles to track evolving ocean processes based on predictions from a Regional Ocean Model. *The International Journal of Robotics Research*, 29(12), 1475–1497.
- Soulignac, M. (2011). Feasible and optimal path planning in strong current fields. *IEEE Transactions on Robotics*, 27(1), 89–98. <https://doi.org/10.1109/tro.2010.2085790>.

Publisher's Note Springer Nature remains neutral with regard to jurisdictional claims in published maps and institutional affiliations.

Springer Nature or its licensor holds exclusive rights to this article under a publishing agreement with the author(s) or other rightsholder(s); author self-archiving of the accepted manuscript version of this article is solely governed by the terms of such publishing agreement and applicable law.



Haoyan Zhai received the B.S. degree from Zhejiang University in 2014, and the Ph.D. in mathematics degree from Georgia Institute of Technology in 2019. His research interests include optimal transport, and stochastic differential equations with applications in robotics and machine learning.



Mengxue Hou received the B.S. degree from Shanghai Jiao Tong University in 2016. She has been pursuing a Ph.D. degree from the School of Electrical and Computer Engineering, Georgia Institute of Technology, since 2016. Her research interests include control theory and robotics.



Fumin Zhang received the B.S. and M.S. degrees from Tsinghua University, in 1995 and 1998, and the Ph.D. degree from the Department of Electrical and Computer Engineering, University of Maryland, College Park, in 2004. He joined the School of ECE, Georgia Institute of Technology as a Professor in 2007. He was a Lecturer and Postdoctoral Research Associate in the Mechanical and Aerospace Engineering Department, Princeton University from 2004 to 2007. His research

interests include marine autonomy, mobile sensor networks, and unmanned systems.



Haomin Zhou received the B.S. in mathematics from Peking University, the M. Phil. in applied mathematics from the Chinese University of Hong Kong, and the Ph.D. in computational mathematics from University of California, Los Angeles. He was a postdoctoral scholar at California Institute of Technology for three years. He has been a faculty member in the School of mathematics at Georgia Institute of Technology since 2003. His research interests include optimal transport, inverse problems, and

stochastic differential equations with applications in robotics and machine learning.