Set-based optimal control in 3D time-varying velocity fields using α-shapes

Nicholas M.W. Sharp
Department of Computer Science
Virginia Tech
Blacksburg, VA 24060
Email: nsharp3@vt.edu

Shane D. Ross
Department of Biomedical
Engineering and Mechanics
Virginia Tech
Blacksburg, VA 24060
Email: sdross@vt.edu

Abstract— **Vehicles in time-varying, non-uniform currents, such as autonomous vehicles in underwater and atmospheric settings, present a difficult optimization problem in charting time-optimal paths from a given source to a given destination. The problem is especially challenging in the case of weakly propelled vehicles, where the current is stronger than the propulsion and the system is not fully controllable. Recent progress has been made on the general optimal control problem by using front propagation to track the boundary of a reachable set. This work presents a significant advancement on that technique by using α-shapes to dynamically mesh the reachable set boundary in three dimensional space. This process permits the solution, for the first time, of globally time-optimal trajectories in three dimensional spatiotemporally varying flows. The method is presented and demonstrated on an example chaotic flow.**

I. Introduction

Applications in aerial and underwater navigation reduce to the problem of finding an optimal path through a known geophysical fluid environment. Here we consider only time-optimal navigation, where the vehicle has some constant speed and is controlled by its heading. Note that this formulation is equivalent to a vehicle with controlled, bounded speed, because in all optimal trajectories the vehicle's speed is necessarily maximal. Additional complexity is introduced when the vehicle is slower than the surrounding current and the system is not completely controllable, a key focus of recent efforts.

The problem was first stated as Zermelo's Navigation Problem, concerning the solution of minimum-time paths to cross a river with known analytical flow [1]. This problem has closed form solutions for many flows, and is often useful as a test case for more advanced techniques. However, modern efforts have focused on general computational methods for arbitrarily specified flows.

The simplest class of methods are grid-based state space searches, which resemble Dijksta's shortest path algorithm [2]. These methods are highly effective for time-invariant flows, and can be extended to time-varying flows [3]. Additionally, they can be utilized as heuristics to give practically useful solutions to many variants of the problem [4]. Methods in artificial intelligence have been applied to these techniques, yielding significant efficiency improvements in both optimal and approximate methods [5], [6].

Additionally, some authors have investigated numerical methods such as shooting methods and piecewise optimizations [7]. These are often practical for simple flows and finding approximate solutions, but may become unstable and impractical for optimal solutions in complex, time-varying flows [8]. Iterative methods are used to relax an initial path to a locally optimal solution [9]. This method can be effective on simple flows, even in the time-varying case, and furthermore it permits a wide variety of cost metrics in time, energy, and space. However, solution spaces for complex flows contain many local minima, which prevent relaxation methods from finding even approximately optimal solutions. Additionally, finding an initial feasible solution is nontrivial in the case of a vehicle which is slow relative to the surrounding flow [10], [11], [12].

Additionally, recent work has addressed the issue of control schemes for realistic physical vehicles such as underwater gliders [13]. These efforts model realistic vehicle dynamics to find feedback control laws for basic motion. These local methods are necessary to actually traverse the global paths considered here.

Recent efforts for the global problem have utilized set-based methods with great success [14], [15]. These develop the solution by considering a *reachable front* in space which bounds the *reachable set*, the region that can be reached for a given cost (or less). An expansion rule is then used to optimally deform and expand the reachable set. Once the reachable set expands to include a desired target destination, the optimal path to the target can be reconstructed. The front method can be interpreted by analogy to computational front propagation techniques; the boundary of the set is treated as an expanding front [16]. Set-based methods have the additional strength that they find globally optimal solutions to the fully general form of the problem, with incompletely controllable vehicles in time-varying flows. However, previous work [14], [15], [16] has only considered time-varying flow fields on a two dimensional domain, due mainly to the complexity of representing sets in three dimensional space.

In this work, we develop a computational method using α-shapes to expand reachable sets in 3D space. Using this method, we derive optimality constraints from Pontryagin's Minimum Principle and demonstrate the solution of 3D, time-optimal trajectories in the general case of the problem.

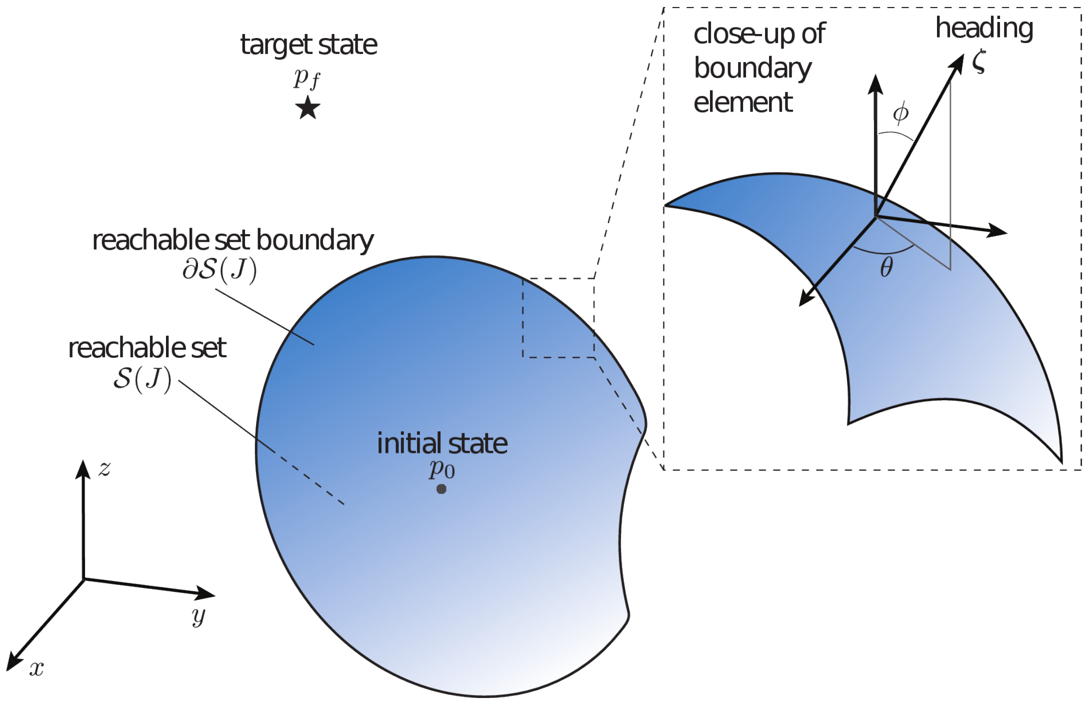


Fig. 1: Schematic of $\mathcal{S}(J)$, the set reachable with cost less than or equal to J starting from state p_0. We propagate the boundary $\partial\mathcal{S}(J)$ outward (towards larger cost) until the boundary reaches the target state p_f at a cost J_f. From the point on $\partial\mathcal{S}(J_f)$ which intersects p_f, we obtain the control profile which affects the optimal trajectory from p_0 to p_f.

All computational methods are available online[1] under an MIT license. Implementations are coded in Python with SciPy and CGAL for computational geometry [17], [18].

II. Set-Based Control

The core concept in set-based control is the development of a reachable set $\mathcal{S}(J)$, the set of states reachable with some cost J. Suppose we want an optimal trajectory from an initial state p_0 to a target state p_f. Calculating an optimal trajectory with a set-based strategy begins by considering an initial reachable set $\mathcal{S}(0) = \{p_0\}$. This set is continually expanded to give all points reachable with $J > 0$. We then find $J_f = \min\{J : p_f \in \mathcal{S}(J)\}$, which necessarily corresponds to the optimal trajectory. Finally, we reconstitute the optimal trajectory by following p_f backwards through the surface of each reachable set with $J < J_f$.

To apply this method to a particular system, we must determine (i) a rule by which to expand $\mathcal{S}(J)$ such that it always contains all reachable states, and (ii) a computational representation of $\mathcal{S}(J)$ in the state space of the system. In practice, we represent $\mathcal{S}(J)$ by its boundary surface $\partial\mathcal{S}(J)$, as in Fig. 1.

In current control problems, we derive a rule for expanding $\mathcal{S}(J)$ from Pontryagin's Minimum Principle. Intuitively, this stems from considering the continuous distribution of states along the boundary of the reachable set $\partial\mathcal{S}(J)$. If each of these states is controlled optimally according to Pontragin's Minimum conditions for the system, they will necessarily yield $\partial\mathcal{S}(J + \Delta J)$.

As previous work has shown, this set-based control yields a powerful technique for the solution of time-optimal trajectory problems [15]. However, it has not been extended to higher dimensions due to the challenges associated with representing and manipulating the $\partial\mathcal{S}$ surface in 3D. In 2D, $\partial\mathcal{S}$ can be represented as a continuous closed curve or set of curves, but in 3D $\partial\mathcal{S}$ is a surface and must be represented as a mesh. Furthermore, this mesh must support three operations:

- We must be able to generate $\partial\mathcal{S}(J + \Delta J)$ from $\partial\mathcal{S}(J)$ to iteratively expand the reachable set.
- The discrete set of points which represents $\partial\mathcal{S}$ must be interpolated during the expansion to maintain sufficient resolution (i.e., adaptive meshing).
- The set will become self-intersecting as the computation progresses [19]. We must then *trim* the surface $\partial\mathcal{S}(J)$ during the computation to remove interior sections. While not strictly necessary for correctness, our findings agree with previous authors that this step is crucial in reducing the computational cost scaling [15].

The first operation is relatively straightforward, and the second operation is nontrivial but not uncommon, but implementing these in addition to the third operation is on the forefront of surface meshing research. Furthermore, the topological management of sets in 3D, such as when one connected $\mathcal{S}$ becomes two disconnected sets, provides further complications.

Recent work has made significant progress in designing

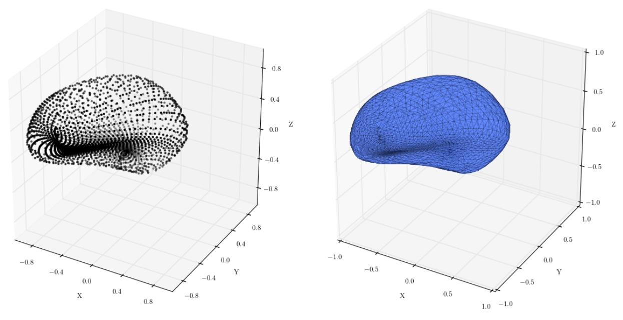
[1]`https://github.com/nsharp3/OptimalAlphaShapes`

exact meshing schemes which provide all of these operations [20]. Indeed, such a scheme would be entirely superior to the method presented here. However, these methods are not yet well-enough understood for use in applications such as this. In the meantime, we provide an alternate approach which allows theoretical developments and practical applications to move forward. We also note that because the meshing procedure is orthogonal to the rest of the optimal trajectory algorithm, these exact meshing schemes could replace the α-shape scheme used here once they are sufficiently mature.

III. α-SHAPES

Consider the problem of generating the connectivity of a surface mesh corresponding to an unconnected point cloud. If the desired shape is convex, then this is the well-understood convex hull problem. However, the mapping of point clouds to non-convex shapes is not well-defined.

The method of α-shapes rigorously define possibly-concave shapes by using an α parameter which effectively defines the radius of a minimum acceptable cavity [21]. The shapes are calculated by finding a Delaunay triangulation and then removing edges to create concavities, which runs in at worst-case $\mathcal{O}(n^2)$ time, where n is the number of points. An example is shown in Fig. 2.



(a) An unconnected point cloud (b) The corresponding α-shape

Fig. 2: α-shapes define a connected mesh over an unconnected point cloud of n points, sidestepping the need for advanced meshing.

This definition permits the manipulation of a dynamic 3D mesh without an advanced meshing scheme by treating the surface as an unconnected point cloud between each iteration, and re-running the α-shape algorithm to generate connectivity for interpolation at the end of each iteration. Points which are not on the surface of the shape must be internal, and can be discarded to affect a trimming operation. The threshold for interpolation along the surface can be related to α, to ensure that the algorithm does not create false exclusions along the surface.

There is some error associated with this process. If two sections of $\mathcal{S}$ are within a distance 2α of each other, they will be joined as a single segment. We note that this is an error in a numerical sense, rather than a logical approximation. As $\alpha \to 0$, this error disappears.

IV. TIME-OPTIMAL CONTROL IN A 3D FLOW

Equipped with this 3D meshing procedure, we can directly extend the method given by Rhoads *et al.* [15] to provide the first method for globally time-optimal control in three dimensions.

The system is defined by a state $\mathbf{x} = (x, y, z) \in \mathbb{R}^3$ (thus p_0 and p_f correspond to two points in $\mathbb{R}^3$) and heading control state $\boldsymbol{\zeta} \in S^2$ viewed as a unit vector in $\mathbb{R}^3$ parametrized by spherical coordinates, i.e., $\boldsymbol{\zeta} = (\theta, \phi)$ where θ is in the xy-plane relative to the x axis, and ϕ is relative to the z axis (see Fig. 1 where we see that $\boldsymbol{\zeta}$ is the local surface normal to $\partial\mathcal{S}(J)$). The state $\mathbf{x}$ evolves according to

$$\frac{d\mathbf{x}}{dt} = \begin{bmatrix} u_x(\mathbf{x}, t) + s\cos\theta\sin\phi \\ u_y(\mathbf{x}, t) + s\sin\theta\sin\phi \\ u_z(\mathbf{x}, t) + s\cos\phi \end{bmatrix}, \tag{1}$$

where u_x, u_y, and u_z are the spatial components of a spatiotemporally varying flow field $\mathbf{u}(\mathbf{x}, t)$. Trajectories which take the least time will necessarily have $s = s_{max}\ \forall\ t$, thus the speed is fixed. (It can be shown by induction that $s < s_{max}$ is never time-optimal.)

We apply Pontryagin's minimum principle to establish necessary conditions for an optimal trajectory [15]. The full procedure is given in Appendix A; the result is the control heading equations given in (9). Since the cost is time, (9) together with (1) are a five-dimensional set of differential equations for the boundary $\partial\mathcal{S}$ which can be propagated in time with initial conditions given by an infinitesimal sphere, $S_0^2 = \partial\mathcal{S}(0)$, surrounding the starting state p_0.

The relations (9) provide a necessary differential constraint for the optimal solution, but the initial control $\boldsymbol{\zeta}_0 = (\theta_0, \phi_0)$ on S_0^2 is unknown. Using the set-based α-shape procedure to resolve optimal trajectories suggests an alternative conceptualization of the control scheme as a shooting method—the surface of the reachable set is merely an effective way to search the 2D manifold S_0^2 of possible initial controls $\boldsymbol{\zeta}_0$.

Results

To demonstrate the performance of this method, we compute time-optimal trajectories through the Arnold-Beltrami-Childress (ABC) flow. This flow, which mimics unsteady vortical behavior in currents, is given by:

$$\mathbf{u}(\mathbf{x}, t) = \begin{bmatrix} -v\sin(ax + t\omega)\cos(by + t\omega)\sin(cz + t\omega) \\ -v\cos(ax + t\omega)\sin(by + t\omega)\sin(cz + t\omega) \\ v\cos(ax + t\omega)\sin(by + t\omega)\cos(cz + t\omega) \end{bmatrix} \tag{2}$$

Results for a representative time-optimal trajectory for a vehicle much slower than the current are given in Fig. 3. A movie visualizing the search procedure and solution for this problem is available in the web repository.[2]

V. DISCUSSION AND FUTURE WORK

The solution of optimal trajectories in known flow fields has important applications in industry and research, but presents a challenging optimal control problem. Recently,

[2]https://github.com/nsharp3/OptimalAlphaShapes/blob/master/ABC_Flow_Movie.mp4

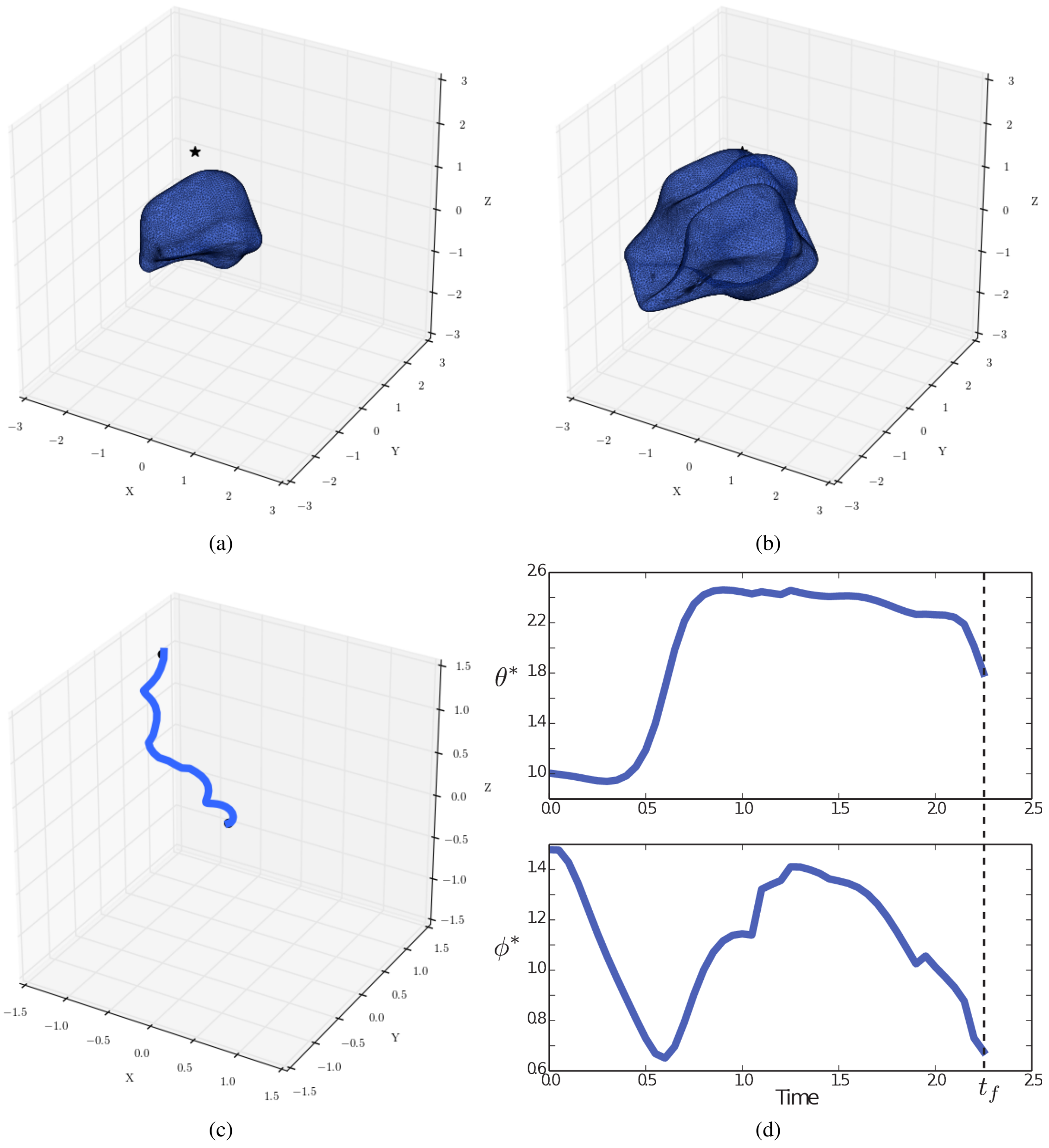


Fig. 3: The solution of a time-optimal trajectory problem in the ABC flow. Flow parameters are $v = 2$, $a = b = c = 2$, and $\omega = 3$. The initial state is $p_0 = (0, 0, 0)$ and the target state is $p_f = (-1.25, 0.8, 1.25)$, indicated by the '$\star$'. The vehicle has speed $s_{max} = 0.5$, smaller than that of the surrounding current. A solution is found with $J_f = t_f = 2.2$. At the final iteration, $\mathcal{S}(J_f)$ is represented by 36256 triangular surface facets defined on 18546 points. Implemented in un-optimized, single-threaded Python code, this computation requires approximately 5 minutes on a 2.4 GHz Intel i7 CPU. (a) The reachable set at $t = 1.5$. (b) The reachable set at $t = 2.2$, when the solution is found. (c) The optimal path in space. Plot is scaled for clarity. (d) The optimal controls $\theta^*(t)$ and $\phi^*(t)$. Small irregularities along the path correspond to interpolation events on $\partial\mathcal{S}(J)$. A corresponding movie is available on the web at `https://github.com/nsharp3/OptimalAlphaShapes/blob/master/ABC_Flow_Movie.mp4`

set-based methods have provided robust routines for the case of time-varying, non-uniform two-dimensional flow fields in which the flow velocity may be larger than the vehicle's.

In this work, we develop a computational method using α-shapes which permits the extension of set-based methods to three dimensions. With this, we directly solve the problem of computing time-optimal trajectories in spatiotemporally varying three-dimensional flows. To our knowledge, this work represents the first general method for finding globally optimal solutions to this problem in three dimensions.

Future Work

As with all numerical methods, every detail in the implementation of this α-shape method has potentially significant implications for numerical stability and convergence. The choices of α, the interpolation strategy, and the step size must be further investigated for these effects. Furthermore, α-shapes can be weighted, with different α values used at different points. This technique could potentially be used to adapt mesh density for regions of high curvature. Additionally, the inherent connection between the optimal control vector and the surface normal vector could be incorporated to avoid compounding numerical error while propagating differential equations.

The general statement of the controlled vehicle problem assumes that the current is fully known at all times. This assumption may not be realistic in practice, and it would be valuable to develop methods which compute trajectories in partially unknown flows. These methods could incorporate statistical models for the flow to compute trajectories with minimum expected time. Additionally, the current method could also be extended to account for obstructions in space.

The front propagation techniques we use here for optimal control could also be applied to the dynamics of reaction fronts in three-dimensional time-varying, non-uniform fluid flows, extending previous work in 2D (see, e.g., [16], [19], [22]). Under the assumptions of (i) the sharp front limit, that is, the reaction proceeds rapidly compared to diffusion, and (ii) negligible feedback from the chemical reaction to the fluid flow, the advection-reaction-diffusion dynamics can be recast as a 5D set of ordinary differential equations, identical to our eqs. (1) and (9), for a front element in a fluid flow $\mathbf{u}$ with position and orientation given by (x, y, z, θ, ϕ). We intend to apply this method to chaotic 3D flow fields found numerically [23] and experimentally [24].

There is no inherent limitation in our method to propagate surfaces in three *spatial* dimensions; we could have space and time as dimensions. Extending to space-time allows one to use front propagation methods for optimality criteria other than just time-optimality. In future work, we intend to use this method to find trajectories which are optimal under a mixed cost involving both time and control effort. Ideally, we seek to implement the method in real time for control of robotic vehicles in an experimental fluid flow (e.g., [25]).

Additionally, the general technique of using α-shapes for optimal trajectory control problems could potentially be applied to control in other dynamical systems. In fact, α-shapes are well-defined for any dimension, opening the possibility that they could be used to resolve trajectories in many previously intractable systems.

Set-based and set-oriented control holds great promise for the solution of optimal trajectories in a wide range of dynamical systems, given proper computational techniques (cf. [26], [27], [28], [29], [30]).

Acknowledgements

The authors acknowledge helpful feedback and discussions with K. Mitchell, J. Mahoney, C. Woolsey, A. BozorgMagham, and A. Hsieh. Funding for this work was provided by the Virginia Space Grant Consortium. Additionally, SDR acknowledges partial support from NSF Grant 1150456.

Appendix A: 3D Time Optimal Derivation

Here, we derive necessary constraints on the 3D time-optimal control problem using Pontryagin's minimum principle. Recall that the state is $\mathbf{x}$, with control $\boldsymbol{\zeta}$, and the state evolves according to (1). The cost of a trajectory $\mathcal{P}$ is given by arrival time, corresponding to time-optimality.

$$J(\mathcal{P}) = \int_0^{t_f} dt = t_f. \tag{3}$$

To derive optimality constraints, we first admit unknown co-states $\boldsymbol{\lambda}(t) = (\lambda_1(t), \lambda_2(t), \lambda_3(t))$ and then construct the Hamiltonian for the system,

$$H(\boldsymbol{\lambda}, \mathbf{x}, \boldsymbol{\zeta}, t) = \boldsymbol{\lambda} \cdot \frac{d\mathbf{x}}{dt} + 1 \tag{4}$$

The Hamiltonian must be minimized along the controls $H(\mathbf{x}^*, \boldsymbol{\zeta}^*, \boldsymbol{\lambda}^*, t) \leq H(\mathbf{x}^*, \boldsymbol{\zeta}, \boldsymbol{\lambda}^*, t) \quad \forall \ t \in [0, t_f]$. The heading controls θ and ϕ are unbounded, so the minimization implies,

$$\begin{aligned}\frac{\partial H}{\partial \theta} &= 0 = -s\lambda_1 \sin\theta \sin\phi + s\lambda_2 \cos\theta \sin\phi \\ \frac{\partial H}{\partial \phi} &= 0 = \quad s\lambda_1 \cos\theta\cos\phi + s\lambda_2 \sin\theta\cos\phi - s\lambda_3 \sin\phi.\end{aligned} \tag{5}$$

Additionally, because the final time is unrestricted, we know,

$$H(\mathbf{x}^*, \boldsymbol{\zeta}^*, \boldsymbol{\lambda}^*, t) = 0 \tag{6}$$

Eqs. (5) and (6) restrict the costates to functions of the spherical coordinate form,

$$\begin{aligned}\lambda_1 &= r\cos\theta\sin\phi \\ \lambda_2 &= r\sin\theta\sin\phi \\ \lambda_3 &= r\cos\phi\end{aligned} \tag{7}$$

where $r(t)$ is a new unknown function representing the extra degree of freedom in solutions to (5). Eq. (6) is necessary because it implies that the co-states are not all zero, which makes (7) a unique representation of each solution.

Now, we consider the final necessary constraint from Pontryagin's minimum principle

$$\frac{\partial H}{\partial x_i} = -\frac{d\lambda_i}{dt} \tag{8}$$

Solving this system for co-states with the form of (7) yields the differential constraints on the optimal trajectory given as,

$$\begin{aligned}\frac{d\theta^*}{dt} &= -\frac{\partial u_x}{\partial y}\cos^2\theta + \left(\frac{\partial u_x}{\partial x} - \frac{\partial u_y}{\partial y}\right)\cos\theta\sin\theta \\ &\quad + \frac{\partial u_y}{\partial x}\sin^2\theta + \cot\phi\left(-\frac{\partial u_z}{\partial y}\cos\theta + \frac{\partial u_z}{\partial x}\sin\theta\right) \\ \frac{d\phi^*}{dt} &= -\left[\frac{\partial u_z}{\partial x}\cos\theta + \frac{\partial u_z}{\partial y}\sin\theta\right]\cos^2\phi + \left[\frac{\partial u_x}{\partial z}\cos\theta\right. \\ &\quad \left. + \frac{\partial u_y}{\partial z}\sin\theta\right]\sin^2\phi - \frac{1}{2}\left[-\frac{\partial u_z}{\partial z} + \frac{\partial u_x}{\partial x}\cos^2\theta\right. \\ &\quad \left. + \left(\frac{\partial u_x}{\partial y} + \frac{\partial u_y}{\partial x}\right)\cos\theta\sin\theta + \frac{\partial u_y}{\partial y}\sin^2\theta\right]\sin(2\phi)\end{aligned} \tag{9}$$

Because the result for $r(t)$ is completely decoupled from the other two, it can be disregarded. The relations for θ^* and ϕ^* are degenerate about $\phi = 0$ and $\phi = \pi$, so for computational implementation we project them on to Cartesian coordinates.

References

[1] E. Zermelo, "Über das navigationsproblem bei ruhender oder veränderlicher windverteilung," *ZAMM-Journal of Applied Mathematics and Mechanics/Zeitschrift für Angewandte Mathematik und Mechanik*, vol. 11, no. 2, pp. 114–124, 1931.

[2] J. N. Tsitsiklis, "Efficient algorithms for globally optimal trajectories," *IEEE Transactions on Automatic Control*, vol. 40, no. 9, pp. 1528–1538, 1995.

[3] L. C. Polymenakos, D. P. Bertsekas, and J. N. Tsitsiklis, "Implementation of efficient algorithms for globally optimal trajectories," *IEEE Transactions on Automatic Control*, vol. 43, no. 2, pp. 278–283, 1998.

[4] T.-B. Koay and M. Chitre, "Energy-efficient path planning for fully propelled AUVs in congested coastal waters," in *OCEANS-Bergen, 2013 MTS/IEEE*, pp. 1–9, IEEE, 2013.

[5] B. Garau, A. Alvarez, and G. Oliver, "Path planning of autonomous underwater vehicles in current fields with complex spatial variability: an a* approach," in *Robotics and Automation, 2005. ICRA 2005. Proceedings of the 2005 IEEE International Conference on*, pp. 194–198, IEEE, 2005.

[6] D. Ferguson and A. Stentz, "Using interpolation to improve path planning: The field d* algorithm," *Journal of Field Robotics*, vol. 23, no. 2, pp. 79–101, 2006.

[7] A. Rantzer and M. Johansson, "Piecewise linear quadratic optimal control," *IEEE Transactions on Automatic Control*, vol. 45, no. 4, pp. 629–637, 2000.

[8] J. T. Betts, "Survey of numerical methods for trajectory optimization," *Journal of Guidance, Control, and Dynamics*, vol. 21, no. 2, pp. 193–207, 1998.

[9] D. Kruger, R. Stolkin, A. Blum, and J. Briganti, "Optimal auv path planning for extended missions in complex, fast-flowing estuarine environments," in *IEEE International Conference on Robotics and Automation*, pp. 4265–4270, IEEE, 2007.

[10] C. Senatore and S. D. Ross, "Fuel-efficient navigation in complex flows," in *Proceedings of 2008 American Control Conference*, pp. 1244–1248, 2008.

[11] S. D. Ross and P. Tallapragada, "Detecting and exploiting chaotic transport in mechanical systems," in *Applications of Chaos and Nonlinear Dynamics in Science and Engineering-Vol. 2*, pp. 155–183, Springer, 2012.

[12] P. G. Thomasson and C. A. Woolsey, "Vehicle motion in currents," *IEEE Journal of Oceanic Engineering*, vol. 38, no. 2, pp. 226–242, 2013.

[13] S. Fan and C. A. Woolsey, "Dynamics of underwater gliders in currents," *Ocean Engineering*, vol. 84, pp. 249–258, 2014.

[14] T. Lolla, M. Ueckermann, K. Yigit, P. J. Haley, and P. F. Lermusiaux, "Path planning in time dependent flow fields using level set methods," in *Robotics and Automation (ICRA), 2012 IEEE International Conference on*, pp. 166–173, IEEE, 2012.

[15] B. Rhoads, I. Mezić, and A. C. Poje, "Minimum time heading control of underpowered vehicles in time-varying ocean currents," *Ocean Engineering*, vol. 66, pp. 12–31, 2013.

[16] J. Mahoney, D. Bargteil, M. Kingsbury, K. Mitchell, and T. Solomon, "Invariant barriers to reactive front propagation in fluid flows," *Europhysics Letters*, vol. 98, no. 4, p. 44005, 2012.

[17] E. Jones, T. Oliphant, P. Peterson, *et al.*, "SciPy: Open source scientific tools for Python," 2001–.

[18] "CGAL, Computational Geometry Algorithms Library." http://www.cgal.org.

[19] K. A. Mitchell and J. R. Mahoney, "Invariant manifolds and the geometry of front propagation in fluid flows," *Chaos: An Interdisciplinary Journal of Nonlinear Science*, vol. 22, no. 3, p. 037104, 2012.

[20] A. Zaharescu, E. Boyer, and R. Horaud, "Topology-adaptive mesh deformation for surface evolution, morphing, and multiview reconstruction," *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, vol. 33, no. 4, pp. 823–837, 2011.

[21] H. Edelsbrunner and E. P. Mücke, "Three-dimensional alpha shapes," *ACM Transactions on Graphics (TOG)*, vol. 13, no. 1, pp. 43–72, 1994.

[22] D. Bargteil and T. Solomon, "Barriers to front propagation in ordered and disordered vortex flows," *Chaos: An Interdisciplinary Journal of Nonlinear Science*, vol. 22, no. 3, p. 037103, 2012.

[23] C. O. Mehrvarzi and M. Paul, "Front propagation in a chaotic flow field," *Physical Review E*, vol. 90, no. 1, p. 012905, 2014.

[24] S. G. Raben, S. D. Ross, and P. P. Vlachos, "Experimental determination of three-dimensional finite-time lyapunov exponents in multi-component flows," *Experiments in Fluids*, vol. 55, no. 10, pp. 1–6, 2014.

[25] M. Michini, M. A. Hsieh, E. Forgoston, and I. B. Schwartz, "Robotic tracking of coherent structures in flows," *IEEE Transactions on Robotics*, vol. 30, no. 3, pp. 593–603, 2014.

[26] L. Grüne, "Subdivision techniques for the computation of domains of attraction and reachable sets," in *Proc. of NOLCOS*, pp. 762–767, 2001.

[27] O. Junge and H. M. Osinga, "A set oriented approach to global optimal control," *ESAIM Control Optim. Calc. Var.*, vol. 10, pp. 259–270, 2004.

[28] L. Grüne and O. Junge, "A set oriented approach to optimal feedback stabilization," *Systems and Control Letters*, vol. 54, no. 2, pp. 169–180, 2005.

[29] S. Jerg, O. Junge, and S. D. Ross, "Optimal capture trajectories using multiple gravity assists," *Communications in Nonlinear Science and Numerical Simulation*, vol. 14, no. 12, pp. 4168 – 4175, 2009.

[30] P. Grover and S. D. Ross, "Designing trajectories in a planet-moon environment using the controlled Keplerian map," *Journal of Guidance, Control and Dynamics*, vol. 32, pp. 436–443, 2009.