Time-optimal and energy-optimal path planning in two-dimensional potential flows

Rose Gebhardt

***Abstract*— The goal of this project to implement time optimal and energy optimal path planning of a nonholonomic system in potential flow fields using graph-based planning methods and wavefront propagation methods. The velocity field and path-planning workspace are be two-dimensional and the moving object is modelled as a one-dimensional point with nonholonomic kinematic constraints. This is implemented on simple potential flow fields, including uniform flow and flow around a cylinder, and builds toward a more complex potential flow models. It is implemented on the infinite periodic potential flow model of a Kármán vortex street, which is the result of a bluff body or foil oscillating in a fluid. This project will not consider the fluid-structure interactions between the swimmer and the flow field or the controls implementation of the computed paths, but may be left as ongoing work.**

I. Introduction

Autonomous underwater vehicles (AUVs) are a class of vehicles that have a wide range of uses from surveillance and maintenance of underwater structures to monitoring of underwater environmental conditions [1]. AUVs often have to navigate in strong dynamic ocean currents, sometimes with velocities larger than the speed of the vehicle itself. Bio-inspired AUVs in particular, often encounter flow patterns which can be represented by a low-dimensional potential flow model [2]. Typically when planning trajectories for these vehicles, speed, energy-efficiency and safety are important considerations.

There is therefore a need for determining paths for AUVs in external fluid velocity fields which optimize time required and energy expenditure while taking kinematic constraints and obstacle avoidance into account. This project will discuss and implement algorithms which solve this problem and apply them to several common test fluid fields.

In Section II, relevant prior work and how that work relates to this project will be discussed. Section III will introduce notation and definitions used throughout the paper and give a formal definition of the problem statement. Section IV will provide details about the implementation of the algorithms used, including derivations and psuedocode. Section V will introduce a few test case flow fields, show the optimal paths found in these fields, and discuss the validity of these results. Section VI will address the work that still needs to be done, and lastly section VII will highlight the key aspects of the project.

II. Related Work

To find an optimal path under a potential fluid flow, Free and Paley define a potential flow model of a Kármán vortex street and reverse Kármán vortex street created by a flapping foil in a fluid. The Kármán vortex street is estimated using distributed pressure sensors on each fish and this information is used to generate the observably optimal path and the energy optimal path, assuming a sinusoidal trajectory. Closed-loop control is simulated to move the robotic fish along this trajectory. [2]. This paper is used to generate a realistic flow field and to verify results. This project, however, will not assume sinusoidal movement or consider controller design.

To solve the problem of path planning in external flow fields, Kularatne *et al* studied underwater vehicles subject to a flow field and dynamic constraints to find time and energy optimal paths using graph-based methods with two different cost functions. The time-optimal path was found using A* while the energy-optimal path is found using Dijkstra's [1]. Similarly, Koay *et al* used A* to find an energy-optimal path for an autonomous underwater vehicle (AUV) in a static ocean flow field under dynamic constraints using one of four admissible heuristics based on the length of the current [3]. These methods will be used to derive cost functions, heuristics, and to solve for the cost minimizing paths in this project.

Rhoads *et al* give an alternative to an A* graph search by finding a time-optimal path of an underwater glider in a two-dimensional environment using feedback control, which is done by solving the dynamic Hamilton Jacobi Bellman for the optimal control law using an extremal field algorithm. This is applied to both time-invariant and time-varying flow fields [4]. This paper investigates a system with kinematic constraints similar to the ones that will be examined in this project and lays out the optimal control problem for the system.

Lolla *et al* provide a method for time-optimal path planning through a time-varying flow field by using forward level-set evolution and backward particle tracking. They include details about implementation and results from a few common flow patterns [5]. This paper is useful for a high-level understanding of level-set evolution and parts of it will be used for the wavefront propagation method. This is similar to the work done by Elston and Frew, who back propagate a cost-to-map from a goal region based on the Hamilton-Jacobi partial differential equations in order to find optimal paths for aircrafts in severe weather conditions [6]. To derive the cost map, they use control-theoretic ordered upwind methods, developed by Sethian and Vladimirsky [7]. Details of the wavefront propagation algorithm and proofs of convergence are provided by Sethian and Vladmirsky, while considerations related to the physical system, such as the

formation of an edge cost function are provided by Elston and Frew.

Lastly, Wolek and Woolsey derived a feasible turn radius for a Dubin's car given some velocity disturbances and prove that, by increasing the radius as a function of the magnitude of the velocity disturbance, one can guarantee that the curve is feasible for the vehicle [8]. The kinematic model of the vehicle in this project is a Dubin's car, so the result from this paper is useful for determining feasibility of the optimal paths found.

III. System Definition

A. External velocity field

We want to consider the problem of finding time-optimal and energy-optimal trajectories in a two-dimension planar potential flow field. An ideal flow is one which is inviscid, incompressible, and irrotational, except at discrete points. Given an ideal flow, we can define a scalar velocity potential function $\phi(\mathbf{x}, t)$ such that

$$\mathbf{v}(\mathbf{x}) = \nabla\phi(\mathbf{x})$$

where $\mathbf{x} = [x, y]^T \in \mathbb{R}^2$ is the position in the two-dimensional workspace and $\dot{\mathbf{x}} = \mathbf{v}(\mathbf{x}) = [v_x, v_y]^T \in \mathbb{R}^2$ is the velocity of the flow field at location $\mathbf{x}$. The velocity components can be written more explicitly as the gradient of the potential:

$$v_x(\mathbf{x}) = \frac{\partial\phi(\mathbf{x})}{\partial x}$$

$$v_y(\mathbf{x}) = \frac{\partial\phi(\mathbf{x})}{\partial y}$$

B. Swimmer kinematics

To simplify the problem, we can think of the swimmer as a one-dimensional point moving in the workspace. Thus, we do not need to consider the fluid-structure interactions between the swimmer and the flow field. The swimmer, however, will still have an associated orientation with respect to the workspace.

Define $\mathbf{X}(t) = [X, Y]^T$ as the position of the swimmer at time t, let V be the magnitude of the velocity of the swimmer in the absence of a flow field, and let θ be the angle of the swimmer velocity with respect to the $[X, Y]^T$ coordinates. Then the kinematic model of the swimmer is [1],

$$\begin{aligned} \dot{X}(t) &= V\cos(\theta) + v_x(\mathbf{X}) \\ \dot{Y}(t) &= V\sin(\theta) + v_y(\mathbf{X}) \end{aligned} \tag{1}$$

C. Formal problem statements

Problem Definition 1 [Time Optimization]: Suppose we are given a discretized workspace $\mathbb{W} \subset \mathbb{R}^2$ with associated external velocity field components $v_x(\mathbf{x})$ and $v_y(\mathbf{x})$ $\forall\ \mathbf{x} \in \mathbb{W}$, a start location $\mathbf{x}_{start} \in \mathbb{W}$, a goal location $\mathbf{x}_{goal} \in \mathbb{W}$, and a differential time cost function dt. Find the path $\Gamma^\star$ which minimizes the functional $\int_\Gamma dt$ with constraints $\Gamma(t = 0) = \mathbf{x}_{start}$, $\Gamma(t = T) = \mathbf{x}_{goal}$ and $V \leq V_{max}$.

.

Problem Definition 2 [Energy Optimization]: Suppose we are given a discretized workspace $\mathbb{W} \subset \mathbb{R}^2$ with associated external velocity field components $v_x(\mathbf{x})$ and $v_y(\mathbf{x})$ $\forall\ \mathbf{x} \in \mathbb{W}$, a start location $\mathbf{x}_{start} \in \mathbb{W}$, a goal location $\mathbf{x}_{goal} \in \mathbb{W}$, and a differential energy cost function dW. Find the path $\Gamma^\star$ which minimizes the functional $\int_\Gamma dW$ with constraints $\Gamma(t = 0) = \mathbf{x}_{start}$, $\Gamma(t = T) = \mathbf{x}_{goal}$ and $V \leq V_{max}$.

IV. Methodology

A. Graph definition

The time-optimization problem will be solved using A* and a control-theoretic anisotropic wavefront propagation algorithm and the energy-optimization problem will be solved using Dijkstra's algorithm. All the methods being considered will utilize a graph data structure to find the time and energy optimal paths. We construct an undirected weighted graph $G = (V, E)$, where vertices V are samples from the workspace which are associated with a position in $\mathbb{R}^2$ and an external velocity vector, and edges E connect neighboring nodes and are associated with a cost function based on the time required or the energy required to move between vertices.

The vertices form a uniform rectangular grid and are connected to 16 neighboring nodes as shown in Figure 1.

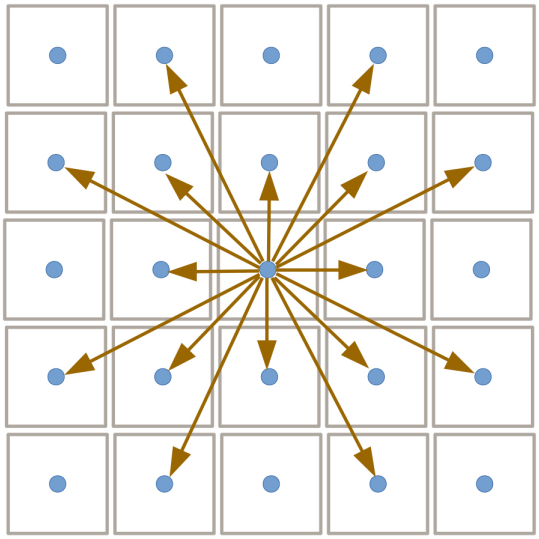


Fig. 1: 16-nodes edge connectivity pattern [1]

Define $v(\mathbf{x})$ as the magnitude of the flow field velocity,

$$v(\mathbf{x}) = \|\mathbf{v}(\mathbf{x})\|_2 = \sqrt{v_x(\mathbf{x})^2 + v_y(\mathbf{x})^2}$$

Then let $\hat{v}_1(\mathbf{x})$ and $\hat{v}_2(\mathbf{x})$ be the orthonormal basis at point $\mathbf{x}$ such that $\hat{v}_1(\mathbf{x})$ is tangent to the streamline and $\hat{v}_2(\mathbf{x})$ is parallel to the potential function level set at point $\mathbf{x}$.

$$\hat{v}_1 = \frac{\vec{v}}{v}$$

$$\hat{v}_2 = \begin{pmatrix} 0 & -1 \\ 1 & 0 \end{pmatrix} \hat{v}_1$$

For each edge, let $\mathbf{x}_i$ be the location of the start node in the directed edge and $\mathbf{x}_{i+1}$ be the end node. We can then define dx as the distance traveled between nodes in the $\hat{v}_1$-direction and dy as the distance traveled between nodes in the $\hat{v}_2$-direction using the inner product of the displacement vector with the coordinate vectors.

$$dx = \hat{v}_1^T(\mathbf{x}_{i+1} - \mathbf{x}_i)$$
$$dy = \hat{v}_2^T(\mathbf{x}_{i+1} - \mathbf{x}_i)$$

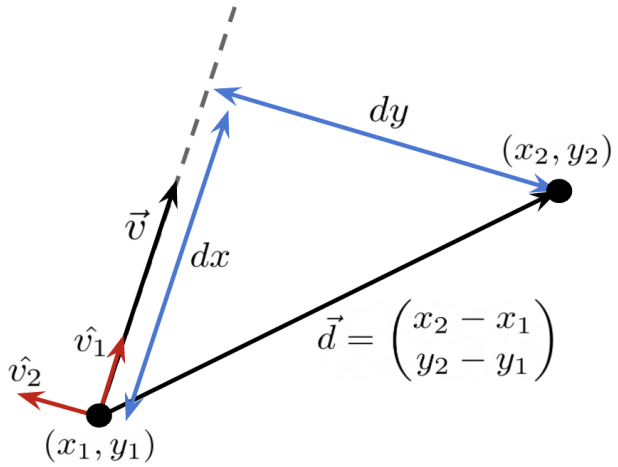


Fig. 2: Orthonormal basis in direction of fluid flow

We can now write the displacement and external velocity vectors in the flow-velocity coordinates [1]:

$$\vec{v} = \begin{pmatrix} v \\ 0 \end{pmatrix}, \ \vec{d} = \begin{pmatrix} dx \\ dy \end{pmatrix}$$

This change of coordinates simplifies the calculations needed for finding the time-costs and energy-costs associated with each edge.

B. Differential cost functions

In order to find each of the edge costs, assume that the swimmer moves at its maximum velocity $V = V_{max}$ and assume that the energy expended comes from work done to overcome the drag force, $\vec{F} = \kappa\vec{v}$, where κ is the drag coefficient of the swimmer.

To find the time-optimizing cost function, notice that travel time is minimized when

$$V_{max}^2 = \left(\frac{dx}{dt} - v\right)^2 + \left(\frac{dy}{dt}\right)^2$$

This can be expanded to a second-order polynomial in dt

$$(v^2 - V_{max}^2)dt^2 - (2vdx)dt + (dx^2 + dy^2) = 0$$

Solving this quadratic yields the time-cost function [1]

$$dt = \frac{vdx - \sqrt{V_{max}^2(dx^2 + dy^2) - v^2dy^2}}{v^2 - V_{max}^2} \qquad (2)$$

Notice, first, that the negative root is used instead of the positive root. We can verify this is correct by plugging in $dy = 0$ to get $dt = \frac{dx}{v+V_{max}}$, as expected. Also note that, if dt is not a non-negative real number, this is not a valid time cost function. This can occur if the velocity field is too large for the swimmer to travel between given nodes. In these cases, the cost function is set to infinity.

To derive the energy-cost function, we assume energy expenditure is due to overcoming drag. Let κ be the drag coefficient of the vehicle in the fluid, and let $\vec{v}_{still}$ and dx_{still} and be the velocity and displacement in the absence of an external fluid velocity. Then the energy cost function is

$$dW = \vec{F} \cdot dx_{still} = \kappa\|\vec{v}_{still}\|^2 dt$$
$$\implies dW = \kappa\left(\left(\frac{dx}{dt} - v\right)^2 + \left(\frac{dy}{dt}\right)^2\right)dt$$

In order to eliminate the dependence of the cost function on dt, we use the fact that the energy-optimal dt is

$$dt = \frac{\sqrt{dx^2 + dy^2}}{v}$$

since this is the differential time cost when the velocity of the vehicle is entirely driven by the external velocity field. This gives the time-independent energy cost function [1]

$$dW = 2\kappa v\left(\sqrt{dx^2 + dy^2} - dx\right) \qquad (3)$$

C. Dijkstra's and A*

The first methods considered are Dijkstra's and A*. These are both graph-based search algorithms which are guaranteed to find resolution-optimal paths from a start node to a goal node if one exists [9].

```
1. UNVISITED ←V\{start}
2. Q.INSERT(start)
3. while Q.TOP() ≠ Ø
   4. v ← Q.POP()
   5. while v.NextNeighbor() ≠ Ø
      6. u ← v.NextNeighbor ()
      7. if u ∈ UNVISITED or u.costToStart > v.costToStart + Cost(v,u)
         8. UNVISITED ← UNVISITED\{u}
         9. u.parent ← v
         10. u.costToStart ← v.costToStart + Cost(v,u)
         11. Q.UPDATE(u,u.costToStart+ h(u,goal))  // insert if u not in Q
   12. if v = goal
         13. return SUCCESS
14. return FAILURE
```

Fig. 3: Dijkstra's and A* Algorithm [10]

The function $h(u, \text{goal})$ on line 5 of Figure 3 the heuristic, which is a function that needs to be less than or equal to the true cost-to-go of node u. For the energy optimization path planning problem, the only admissible heuristic is $h(\mathbf{x}, \mathbf{x}_{goal}) = 0$, so the algorithm simplifies to Dijkstra's.

For the time optimization problem, we can find a lower bound on the time needed to move between vertices by dividing the minimum possible distance traveled between

vertices by the maximum possible velocity, which gives us the admissible heuristic [1]

$$h(\mathbf{x}, \mathbf{x}_{goal}) = \frac{\sqrt{(x - x_{goal})^2 + (y - y_{goal})^2}}{V_{max} + \max_{\mathbf{x}} v(\mathbf{x})}$$

This problem can therefore be solved using A*, which improves the runtime by decreasing the number of nodes considered by Dijkstra's.

D. Wavefront propagation

The wavefront expansion algorithm produces a cost-to-go map over a discretized configuration space using a breadth-first search to approximate level-sets corresponding to the viscosity solutions of the Hamilton-Jacobi partial differential equation [7]:

$$\begin{cases} \|\nabla u\| F\left(\mathbf{x}, \frac{\nabla u}{\|\nabla u\|}\right) = 1 & \mathbf{x} \in \Omega \\ u(\mathbf{x}) = 0 & \mathbf{x} \in \partial\Omega \end{cases}$$

where Ω is the workspace, $\partial\Omega$ is the goal region, $u(\mathbf{x})$ is the viscosity solution of the partial differential equation, or the cost function in the workspace, F is the velocity at position $\mathbf{x}$ in unit direction $\frac{\nabla u}{\|\nabla u\|}$ which can be viewed as a steering control input.

A control-theoretic ordered upwind method for anisotropic problems can be used to solve the time optimization problem. This method starts by assigning the goal region a cost-to-go of zero, then computing the cost-to-go values of adjacent nodes by interpolating cost values over neighboring nodes. The full algorithm is as follows:

1. Start with all the mesh points in Far.
2. Move the mesh points on the boundary ($\boldsymbol{y} \in \partial\Omega$) to $Accepted$ ($U(\boldsymbol{y}) = q(\boldsymbol{y})$).
3. Move all the mesh points $\boldsymbol{x}$ adjacent to the boundary into $Considered$ and evaluate the tentative values

$$V(\boldsymbol{x}) := \min_{\boldsymbol{x_j}\boldsymbol{x_k} \in \mathrm{NF}(\boldsymbol{x})} V_{\boldsymbol{x_j},\boldsymbol{x_k}}(\boldsymbol{x}). \tag{32}$$

4. Find the mesh point $\bar{\boldsymbol{x}}$ with the smallest value of V among all the $Considered$.
5. Move $\bar{\boldsymbol{x}}$ to $Accepted$ ($U(\bar{\boldsymbol{x}}) = V(\bar{\boldsymbol{x}})$) and update the $AcceptedFront$.
6. Move the Far mesh points adjacent to $\bar{\boldsymbol{x}}$ into $Considered$ and compute their tentative values by (32).
7. Recompute the value for all the other $Considered$ $\boldsymbol{x}$ such that $\bar{\boldsymbol{x}} \in \mathrm{NF}(\boldsymbol{x})$

$$V(\boldsymbol{x}) := \min\left\{V(\boldsymbol{x}), \min_{\bar{\boldsymbol{x}}\boldsymbol{x_i} \in \mathrm{NF}(\boldsymbol{x})} V_{\bar{\boldsymbol{x}},\boldsymbol{x_i}}(\boldsymbol{x})\right\}. \tag{33}$$

8. If $Considered$ is not empty, then go to 4.

Fig. 4: Control Theoretic Ordered Upwind Method for Anisotropic Problems [7]

In the algorithm in Figure 4, NF($\mathbf{x}$) are the points on the wavefront within a certain distance of $\mathbf{x}$, $V_{\mathbf{x_j},\mathbf{x_k}}(\mathbf{x})$ is the candidate cost-to-go value, found by interpolating the cost of moving from position $\mathbf{x_j}$ and the cost of moving from position $\mathbf{x_k}$, and $U(\mathbf{x})$ is the accepted cost-to-go value, which is set once a node is added to the accepted front.

Two benefits of this methods are that, for the static case, where the flow field is not changing in time, the cost-to-go map only needs to be computed once and this method is proven to converge to the viscosity solution as the distance between grid points approaches zero [7].

V. Application

A. Complex potential functions

Each of the flow fields that the path-planning algorithms will be applied on can be defined by a complex potential function

$$F(z) = F(x + iy) = \Phi(x, y) + i\Psi(x, y)$$

where $\Phi(z)$ is the real scalar-valued velocity potential function whose partial derivatives are the velocities in the corresponding directions and $\Psi(z)$ is the streamline function.

Notice that, by the definition of complex differentiation and by the Cauchy-Riemann equations

$$\frac{dF}{dz} = \frac{\partial\Phi}{\partial x} + i\frac{\partial\Psi}{\partial x} = \frac{\partial\Phi}{\partial x} - i\frac{\partial\Phi}{\partial y} = v_x - iv_y$$

which can be used to get the velocity vectors without defining them explicitly in terms of x and y.

Using this notation, we can now define the three flow fields in which the path planning methods will be applied:

(1) *Uniform flow:* This will be the most simple flow case that can be used to identify errors in the algorithm implementation. The flow is defined by

$$F(z) = u_0 z, \quad \frac{dF}{dz} = u_0$$

and it describes uniform flow with magnitude u_0 acting the positive x-direction. In simulations, $u_0 = 1$, which is shown in Figure 5.

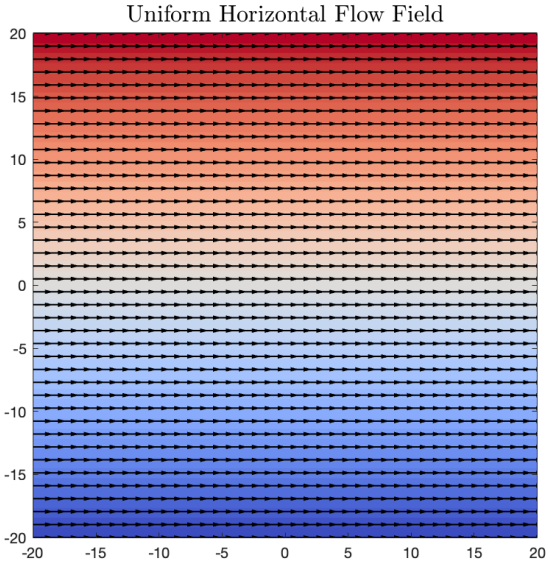


Fig. 5: Uniform Flow Field

(2) *Free stream past a cylinder*: The next case will be the two-dimension cross-section of a uniform free stream passing a cylinder. It is another relatively simple flow field that can be used to identify errors and test the ability of the algorithms to avoid obstacle collision, the obstacle being the cylinder itself. The flow field is defined by

$$F(z) = u_0\left(z + \frac{r_0^2}{z}\right), \quad \frac{dF}{dz} = u_0\left(1 - \frac{r_0^2}{z^2}\right)$$

where u_0 is the magnitude of the free stream acting in the positive x-direction and r_0 is the radius of the cylinder centered at the origin. The parameters used in simulations were $u_0 = 1$ and $r_0 = 8$, which shown in Figure 6.

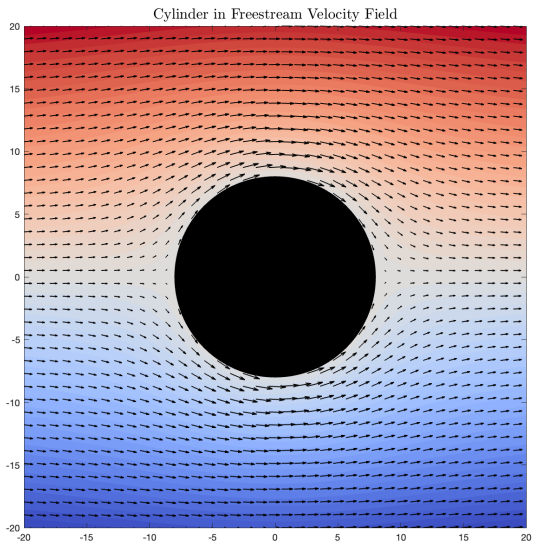


Fig. 6: Freestream Past a Cylinder

(3) *Kármán vortex street*: The final case will the be the Kármán vortex street flow field, that occurs as a result of a flapping airfoil in static fluid and is seen in the wake pattern of fish. This is a pattern of point vortices that appear in two horizontal lines, with the top line and bottom line having opposite rotational directions. The energy optimal path for a swimmer to take is known to be the slaloming around vortices which can be used to verify the resulting paths found.

One model of this flow field is defined by [2]

$$F(z) = \frac{i\Gamma}{2\pi}\left[\log\left(\sin\left[\frac{\pi(z-z_v)}{a}\right]\right) - \log\left(\sin\left[\frac{\pi}{a}\left(z - z_v - \left[\frac{a}{2} + ih\right]\right)\right]\right)\right]$$

$$\frac{dF(z)}{dz} = \frac{i\Gamma}{2a}\left[\cot\left(\frac{\pi[z-z_v]}{a}\right) - \cot\left(\frac{\pi}{a}\left[z - z_v - \left(\frac{a}{2} + ih\right)\right]\right)\right] = v_x - iv_y$$

The parameters, shown in Figure 7, are the horizontal distance between vortices, a, the vertical distance between vortices, h, the strength of each vortex Γ, and the location of the primary vortex with respect to the origin in a fixed inertial frame, $z_v = x_v + iy_v$.

The parameters used in simulations were $z_v = x_v + iy_v = 0 - 0.75i$, $a = 3$, $h = 1.5$, $\Gamma = 3$ and the resulting velocity field is shown in Figure 8.

B. Results: Time-optimization vs. Energy-optimization

The first set of results will compare the time-optimization solutions with the energy-optimization solutions found using Dijkstra's and A*, respectively.

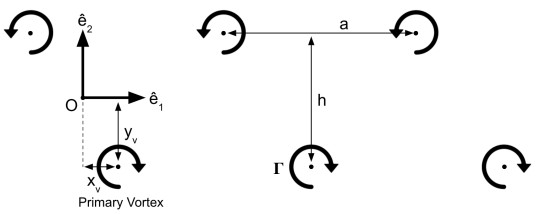


Fig. 7: Time-invariant Kármán vortex street model [2]

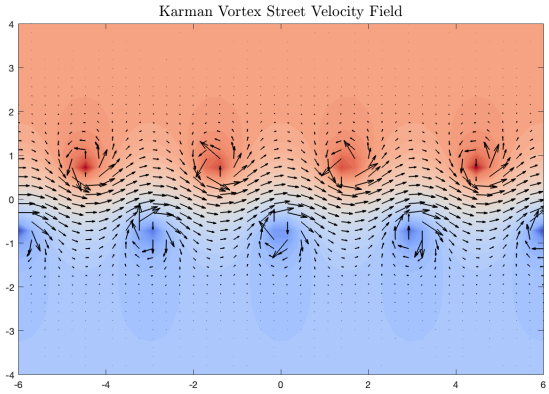
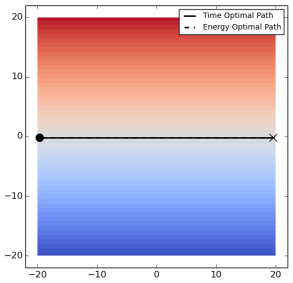
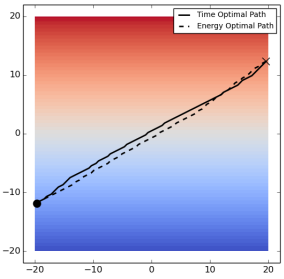


Fig. 8: Kármán vortex street pattern

The results in Figures 9 and 10 were used to verify the accuracy of the algorithms. In Figure 9, the maximum velocity of the swimmer $V_{max} = 2$ in a stream with magnitude $u_0 = 1$ and it shows that, as expected, the optimal path is a direct path between nodes. In Figure 10, the magnitude of the stream remains the same, but $V_{max} = 0.25$ and it is attempting to moving against the stream. Due to this constraint, the swimmer should not be able to reach the goal node and, as expected, Figure 10 shows that the algorithm does not find a correct path, and a warning statement is given in the terminal. This verifies that when path planning is not possible, it will be caught by the algorithm and reported.



(a) No vertical displacement between start and goal



(b) With vertical displacement between start and goal

Fig. 9: Time and energy-optimal planning for uniform flow field in direction of flow

Figure 11 shows time and energy-optimal paths found through a freestream of magnitude $u_0 = 1$ past a cylinder moving with maximum velocity $V_{max} = 1$. The time and

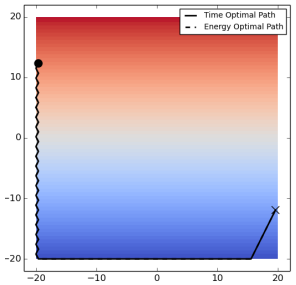
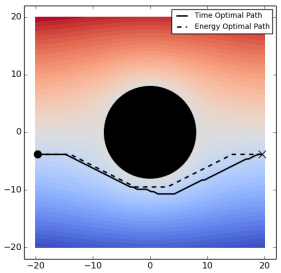
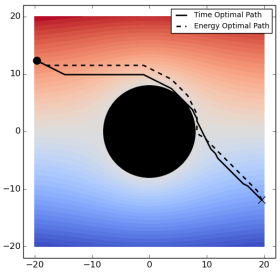


Fig. 10: Time and energy-optimal planning for uniform flow field against direction of flow, intentional failure case

energy-optimal paths are similar and both influenced by the external velocity field. This case is also used to verify that the algorithm is successfully avoiding obstacles by not creating edges which interfere with the cylinder.



(a) No vertical displacement between start and goal



(b) With vertical displacement between start and goal

Fig. 11: Time and energy-optimal planning for freestream around a cylinder in direction of flow

Figures 12 and 13 show the time and energy-optimal paths found through and across a Kármán vortex street. In both plots, the time-optimal path is a more direct path, while the energy-optimal path slaloms around the vortices which results in a oscillatory path. This is a well-known phenomenon known as Kármán gaiting and is an indication that the algorithm was successful at finding the energy-optimal path [2].

C. Results: A* vs. Wavefront expansion method

The next set of results will show a comparison of the time-optimal paths found by the A* algorithm and wavefront expansion algorithms.

Figure 14 was used to verify the accuracy of the wavefront expansion algorithm. The paths found were similar to those found with A*, but less direct. This could be an artifact of the graph-based structure and the interpolation used to determine cost functions.

Figures 15 and 16 show the time-optimal paths found by A* and wavefront expansion while moving in the direction and against the direction of flow with a maximum velocity $V_{max} = 2$. The paths found in the direction of flow are in close agreement with one another, while the wavefront expansion algorithm appears to be finding a more direct route when moving against the direction of flow. Notice again that both algorithms successful avoid obstacle collision with the cylinder.

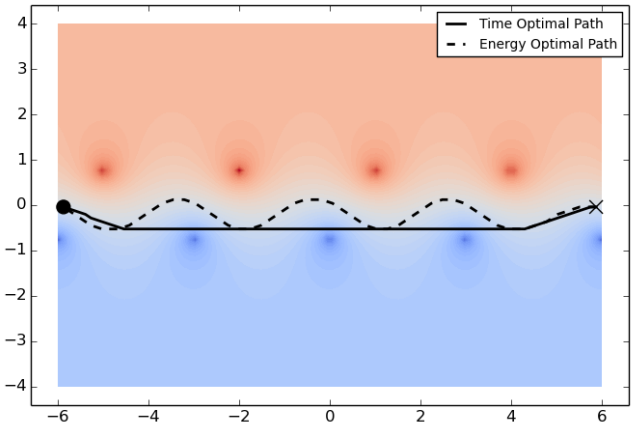


Fig. 12: Time and energy-optimal planning directly through a Kármán vortex street

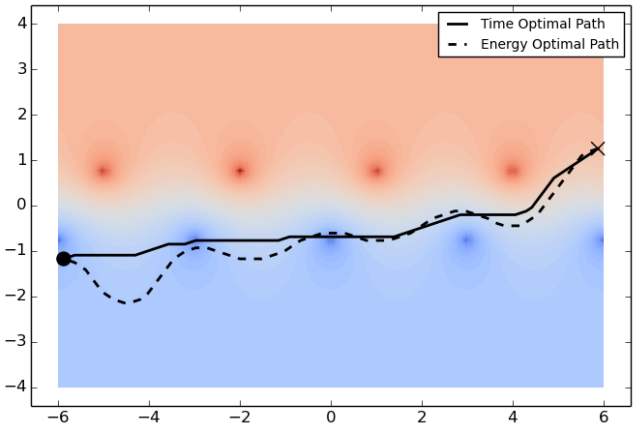
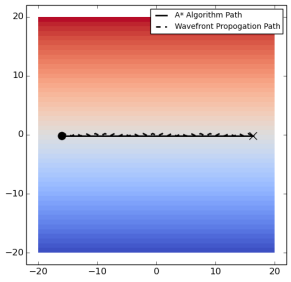
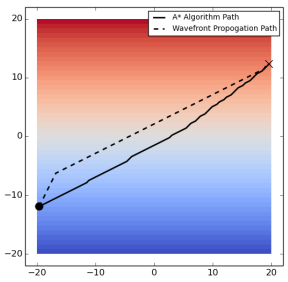


Fig. 13: Time and energy-optimal planning across and through a Kármán vortex street



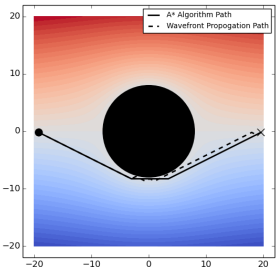
(a) No vertical displacement between start and goal



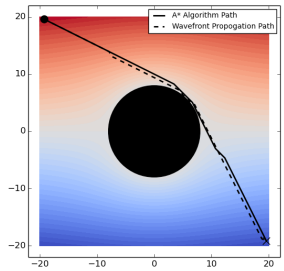
(b) With vertical displacement between start and goal

Fig. 14: Time-optimal planning for uniform flow field in direction of flow

Figures 17 and 18 show the time-optimal paths found moving directly through a Kármán vortex street and moving across a Kármán vortex street, respectively. The paths found by A* are the same as they were in Figures 12 and Figure 13

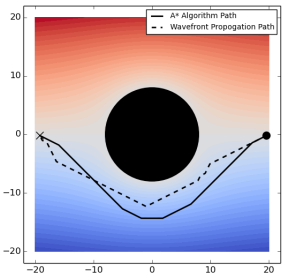


(a) No vertical displacement between start and goal

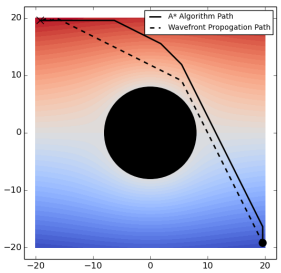


(b) With vertical displacement between start and goal

Fig. 15: Time-optimal planning for freestream around a cylinder in direction of flow



(a) No vertical displacement between start and goal



(b) With vertical displacement between start and goal

Fig. 16: Time-optimal planning for freestream around a cylinder against direction of flow

and are more direct than the typical Kármán gaiting pattern observed in fish. The wavefront expansion method path is similar to A*, but appears to show more interaction with the shed vortices. This deviation from A* intuitively makes sense and is a result of the alternate cost map definition.

VI. Future Work

A. Nonholonomic Kinematic Constraints

The kinematic model specified in equation 1 has kinematic, or nonholonomic constraints, which means the system is not fully integrable and has limitations on how it can move in $\mathbb{R}^2$.

This can be seen if we introduce a control input to the system in equation 1. We get the system [8]

$$\begin{aligned}\dot{X}(t) &= V\cos(\theta) + v_x(\mathbf{X}) \\ \dot{Y}(t) &= V\sin(\theta) + v_y(\mathbf{X}) \\ \dot{\theta}(t) &= u(t)\end{aligned}$$

where the control input $u(t)$ is the turn velocity and is bounded by the inequality

$$|u(t)| \leq \frac{V}{R_0}$$

thereby limiting the direction of movement of the swimmer.

The path planning methods in this project do not explicitly address these kinematic constraints, but if the grid spacing is large with respect to the minimum turn radius of the swimmer, finding a feasible control law should be possible [6]. So, if this turn radius is known, these methods can be used so long as the configuration space is discreteized correctly.

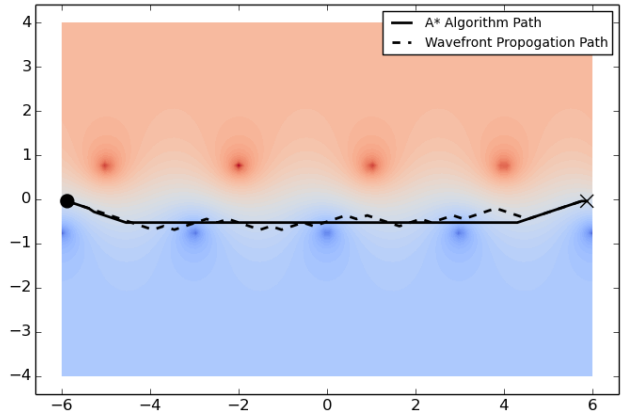


Fig. 17: Time-optimal planning directly through a Kármán vortex street

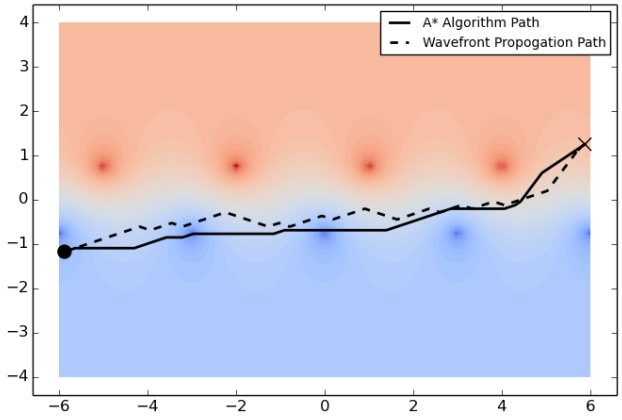


Fig. 18: Time-optimal planning across and through a Kármán vortex street

A useful result is that, if we know the minimum turn radius, R_0 in still water, and define v_{max} and ϵ as

$$v_{max} = \max_{\mathbf{x}\in W} v(t)$$

$$\epsilon = \frac{v_{max}}{V} < 1$$

then the minimum turn radius in the external flow is [8]

$$R_0' = R_0(1+\epsilon)^2$$

Note that this result is limited the case when $V > v_{max}$, which is not always true. Additionally, as the grid spacing gets wider, the paths found will be further from the true optimal paths. Therefore, there is more work to be done to ensure the path planning can be implemented on a physical system.

B. Optimally-observable path planning

For the case of navigating through a Kármán vortex street, in addition to minimizing time and energy expenditure, the ability of the swimmer to estimate the flow pattern is an important consideration for path planning.

To determine the flow pattern, a common set up is for the swimmer to have pressure sensors along its longitudinal axis to make an artificial lateral line. Observability can then be defined as the ability of the system to determine the state of the vortex street from the pressure sensor outputs over time. The observability can be measured numerically as the reciprocal of the minimum singular value of the nonlinear observability Grammian [2]. If the system is unobservable, this value goes to infinity. It has been found that the path taken to maximize the ability to estimate the flow is at odds with the energy-optimal path and requires the swimmer to move into each vortex, as seen in Figure 19 [2].

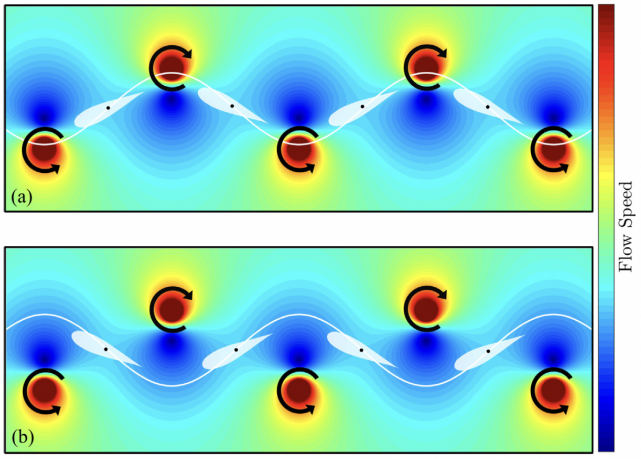


Fig. 19: (a) Optimally observable path and (b) Energy optimal path found by Free and Paley [2]

In the future, it may be useful to combine the metrics for energy-optimization and observability-optimization in order to take both into account.

C. Dynamic flow fields

All of the flow fields considered in the project were static, but both methods used can both be adjusted to solve the path planning problems in a dynamic flow field. This is an important consideration for more realistic flow fields which will be impacted by currents or miscellaneous drifting elements.

For Dijkstra's and A*, the problem can be reinterpreted as path planning over a three-dimensional workspace, with the dimensions representing movement in a plane and time [1]. The cost functions in equations 2 and 3 would be left as functions of dt and information about the time would need to be stored in each node. For the wavefront propagation method, the cost map and starting location can periodically be updated, ideally at a rate similar to the sensor update rate [6]. As the updates are made, the vehicle can follow the most up-to-date gradient descent route.

VII. Conclusions

This project defines the problem of motion planning under an external velocity field and presents two algorithms to solve it over a discretized workspace. The first algorithm is Dijkstra's for energy-optimization and A* for time-optimization and the second algorithm is a wavefront propagation to compute a time cost map and a gradient descent. When applied to three different potential flow fields, we find that, as expected, the energy-optimal path found with Dijkstra's tends to follow the stream of flow while the time-optimal path found with A* tends to be more direct. Additionally we find similar behavior between the time-optimal paths found with A* and wavefront propagation, but the ladder tends to be less direct than the former and more influenced by the external flow field. This difference arises due to the alternate method of computing the cost-to-go value. It is left for future work to determine the proper grid spacing for feasible control, apply a cost function that weights multiple cost values, and to apply the algorithms to dynamic flow fields.

References

[1] D. Kularatne, S. Bhattacharya, and M. A. Hsieh, "Time and energy optimal path planning in general flows," in *Robotics: Science and Systems XII*. Robotics: Science and Systems Foundation. [Online]. Available: https://doi.org/10.15607/rss.2016.xii.047

[2] B. A. Free and D. A. Paley, "Model-based observer and feedback control design for a rigid joukowski foil in a kármán vortex street," *Bioinspiration & Biomimetics*, vol. 13, no. 3, p. 035001, Mar. 2018. [Online]. Available: https://doi.org/10.1088/1748-3190/aaa97f

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

[4] B. Rhoads, I. Mezić, and A. Poje, "Minimum time feedback control of autonomous underwater vehicles," in *49th IEEE Conference on Decision and Control (CDC)*, 2010, pp. 5828–5834.

[5] T. Lolla, M. P. Ueckermann, K. Yigit, P. J. Haley, and P. F. J. Lermusiaux, "Path planning in time dependent flow fields using level set methods," in *2012 IEEE International Conference on Robotics and Automation*. IEEE, May 2012. [Online]. Available: https://doi.org/10.1109/icra.2012.6225364

[6] J. Elston and E. Frew, "Unmanned aircraft guidance for penetration of pretornadic storms," *Journal of Guidance, Control, and Dynamics*, vol. 33, no. 1, pp. 99–107, Jan. 2010. [Online]. Available: https://doi.org/10.2514/1.45195

[7] J. A. Sethian and A. Vladimirsky, "Ordered upwind methods for static hamilton–jacobi equations: Theory and algorithms," *SIAM Journal on Numerical Analysis*, vol. 41, no. 1, pp. 325–363, Jan. 2003. [Online]. Available: https://doi.org/10.1137/s0036142901392742

[8] A. Wolek and C. Woolsey, "Feasible dubins paths in presence of unknown, unsteady velocity disturbances," *Journal of Guidance, Control, and Dynamics*, vol. 38, no. 4, pp. 782–787, Apr. 2015. [Online]. Available: https://doi.org/10.2514/1.g000629

[9] S. M. LaValle, *Planning Algorithms*. Cambridge University Press, 2006.

[10] M. Otte, "Big-o notation, graph search (time permitting: high-level concept survey)," January 2022.