Go With the Flow:
Optimizing the 100 Miler Sailing Race

Connor McBride, Ethan Palenske, and Jackson Pond

Overview

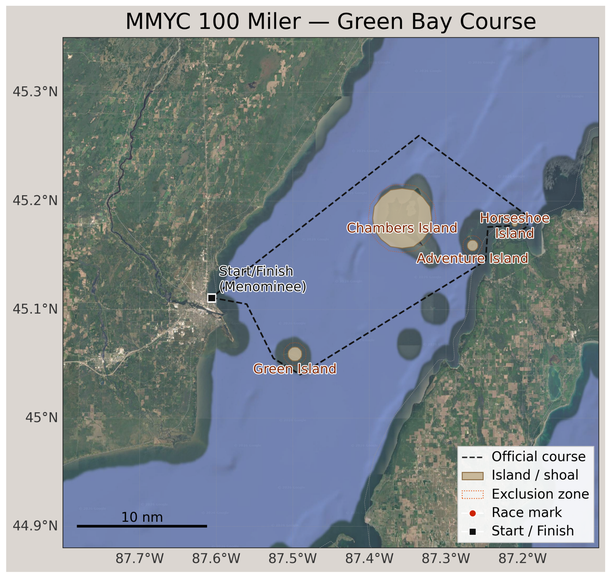

Figure 1: MMYC Race Course for Spinnaker Yachts. ∼41.8 nm. [Mar24]

The historic M&M 100 Miler sailing race takes place annually in Lake Michigan. In this project, we aim to determine the optimal path for a sailboat to take in this race. We use a few different models; piloting a motor boat with constant velocity and variable current, one that matches simple sailing dynamics assuming uniform current and wind, and a more complex one that accounts for real-world sailing dynamics with variable wind and current. Our goal was to determine the simplest model that would re-discover real world sailing techniques, particularly tacking and jibing.

The Problem

The Zermelo Navigation Problem

We begin by solving the standard Zermelo Navigation problem, where we are piloting a motor boat through water with a current and want to travel between two points as fast as possible. We chose θ, the angle between the x-axis and our heading, to be our control, and assumed that the velocity of the boat due to the motor would be given by

$$\begin{bmatrix} u_1 \\ u_2 \end{bmatrix} = v \begin{bmatrix} \cos(\theta) \\ \sin(\theta) \end{bmatrix}.$$

Accounting for current yields the following velocity:

$$\begin{bmatrix} \dot{x} \\ \dot{y} \end{bmatrix} = v \begin{bmatrix} \cos(\theta) \\ \sin(\theta) \end{bmatrix} + \begin{bmatrix} w_1(x,y) \\ w_2(x,y) \end{bmatrix},$$

where w_1 and w_2 refer to the impact of current. So, we are trying to minimize the functional

$$J[u] = \int_{t_0}^{t_f} 1 \, dt,$$

subject to

$$\begin{aligned} \dot{\mathbf{s}} &= \mathbf{u}(t) + \mathbf{w}(\mathbf{s}(t)) \\ \mathbf{s}(t_0) &= \mathbf{s}_0 \\ \mathbf{s}(t_f) &= \mathbf{s}_f \\ \|\mathbf{u}\| &= v\cos(\theta - \phi) \end{aligned}$$

The Lagrangian constraint version will be given by

$$\tilde{J}[u] = \int_{t_0}^{t_f} 1 + \boldsymbol{\lambda}^\top(\dot{\mathbf{s}} - \mathbf{u}(t) - \mathbf{w}(\mathbf{s}(t))) \, dt,$$

and the Hamiltonian will be given by

$$\begin{aligned} H &= \boldsymbol{\lambda}^\top \mathbf{f} - L \\ &= \begin{bmatrix} \lambda_1 & \lambda_2 \end{bmatrix} \left(v \begin{bmatrix} \cos(\theta) + w_1(x,y) \\ \sin(\theta) + w_2(x,y) \end{bmatrix} \right) - 1. \end{aligned}$$

By Pontryagin's Maximization principle, an optimal solution must satisfy

$$\begin{aligned} \dot{x} &= v\cos(\theta) + w_1(x,y) \\ \dot{y} &= v\sin(\theta) + w_2(x,y) \\ \dot{\lambda_1} &= -\lambda_1 \frac{\partial w_1}{\partial x} - \lambda_2 \frac{\partial w_2}{\partial x} \\ \dot{\lambda_2} &= -\lambda_1 \frac{\partial w_1}{\partial y} - \lambda_2 \frac{\partial w_2}{\partial y} \\ H(t_f) &= 0, \end{aligned}$$

where the last equality comes from the fact that we are maximizing over unknown time. We get that $\frac{\partial H}{\partial u} = 0$ as well.

Since we are maximizing H, and the only term where u shows up is $\lambda \cdot u$, we can just maximize that to solve for u. We get $u = \frac{\lambda}{\|\lambda\|} v(\theta)$ at each time t, which we can plug into the above system.

The Simplest Sailing Approach

But what if we are sailing using the wind instead of a motor? This model makes the assumption that current and wind are the only factors we need to account for, and that they directly control velocity. More precisely, the impact of wind on the boat's velocity is

$$\begin{bmatrix} u_1 \\ u_2 \end{bmatrix} = v \begin{bmatrix} \cos(\theta - \phi)\cos(\theta) \\ \cos(\theta - \phi)\sin(\theta) \end{bmatrix},$$

where the new $\cos(\theta - \phi)$ term determines the exposure of the sail to the wind. This makes the boat go at full speed when it's facing the same way as the wind, and is still when its sail is orthogonal to the wind. The derivation of the solution is nearly identical to the previous one.

However, there were some major issues with this approach. This function that determined the boat's velocity assumed that boats could not go upwind, which is not physically accurate. As such, the boat could not complete the course under this optimization problem, and so we had it optimize a single leg of the race.

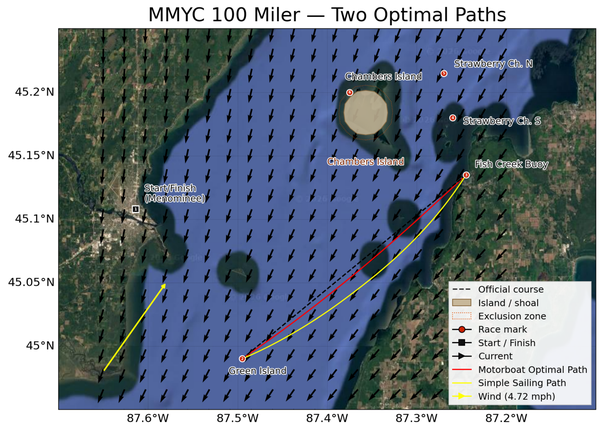

Figure 2: Plotting the optimal paths computed by the two simpler models.

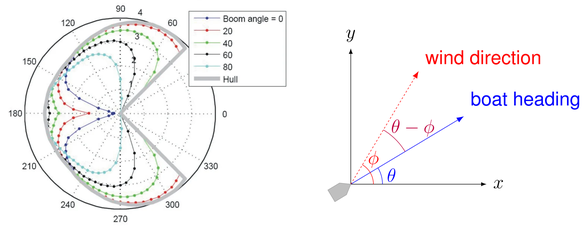

Figure 3: Comparison of theoretical points of sail (left) [RPP11] vs. coordinate system geometry (right).

Physically Accurate Sailing

However, wind doesn't move boats by "blowing on the sails." For boats with triangular sails, it's more accurate to say that the sails act like an airplane wing, creating thrust as wind moves faster over their front than their back. In fact, boats rarely sail directly downwind since this makes them very unstable! A more accurate model would have boats going fastest about 70° off of straight downwind, called a "Broad Reach". Not only does this allow sailboats with triangular sails to sail *across* the wind, or orthogonally to the wind, it also allows the sailboat to sail about 40° from directly upwind! So sailboats can effectively move in any direction, in a straight line for friendly directions, and in zig-zags if close to directly up-wind or directly down-wind.

Switching which zig you are zagging while going upwind is called **tacking**. Switching while going downwind is called **jibing**. One main objective of this project is to *discover* tacking and jibing as part of the optimal solution.

In this case, the speed of the boat is a function of the heading of the boat and the angle of the wind. When we factor in the x and y-components of the current as w_1 and w_2, our optimization problem becomes

$$J[u] = \int_0^{t_f} 1 \, dt,$$

subject to

$$\begin{bmatrix} \dot{x} \\ \dot{y} \end{bmatrix} = v(\theta(t), \phi(x,y)) \begin{bmatrix} \cos(\theta) \\ \sin(\theta) \end{bmatrix} + \begin{bmatrix} w_1(x,y) \\ w_2(x,y) \end{bmatrix}.$$

So our Hamiltonian and costate evolution equations are given by

$$\begin{aligned} H &= \boldsymbol{\lambda}^\top \left(v(\theta,\phi) \begin{bmatrix} \cos(\theta) \\ \sin(\theta) \end{bmatrix} + \begin{bmatrix} w_1(x,y) \\ w_2(x,y) \end{bmatrix} \right) - 1 \\ &= \lambda_1(v(\theta,\phi)\cos(\theta) + w_1) + \lambda_2(v(\theta,\phi)\sin(\theta) + w_2) - 1 \\ \dot{\lambda_1} &= -\left(\lambda_1 \left(\frac{\partial v}{\partial x}\cos(\theta) + \frac{\partial w_1}{\partial x} \right) + \lambda_2 \left(\frac{\partial v}{\partial x}\sin(\theta) + \frac{\partial w_2}{\partial x} \right) \right) \\ \dot{\lambda_2} &= -\left(\lambda_1 \left(\frac{\partial v}{\partial y}\cos(\theta) + \frac{\partial w_1}{\partial y} \right) + \lambda_2 \left(\frac{\partial v}{\partial y}\sin(\theta) + \frac{\partial w_2}{\partial y} \right) \right). \end{aligned}$$

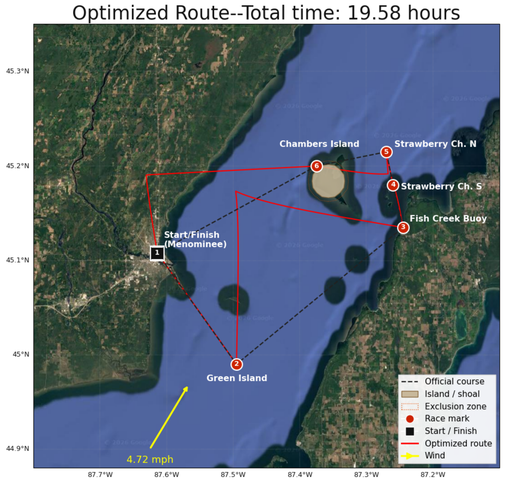

Figure 4: Optimized route of MYMC 100 Miler with constant wind

Early Results & Analysis

Main Results

The simpler model of sailing was not able to uncover methods such as tacking because the boat could not go upwind. However, our more advanced model did discover and use tacking while going upwind. Our Julia implementation may be penalizing tacking too much, resulting in the model making only a few turns during the upwind legs of the race instead of many.

Issues and Troubleshooting

For the Zermelo problem and the simplest sailing model, we used SciPy's `solve_bvp` function with great success. However, our physically accurate model led to discontinuous controls (bang-bang), which lead to sharp corners in the state–tacking. Since `solve_bvp` assumes continuous solutions, it was incompatible with the proper solution. It crashed when trying to resolve the corners by increasing mesh resolution. Similarly, since we did not bound how quickly θ can change, some of our attempts at the solution seemed to result in θ changing direction every timestep, tacking so quickly that it appeared the boat was going in a straight line. In essence, `solvebvp` is theoretically elegant and mathematically clean, but is fragile in ways that did not work well with this problem.

We decided to use Julia's `Ipopt` to attempt to avoid these issues. `Ipopt` is an interior-point optimization solver which only takes an objective function and certain kinds of constraints. We enforced the $\dot{x} = f$ relation via pointwise constraints with forward Euler, and penalized changes in theta in the objective. Whereas `Ipopt` was a bit more brute force, it successfully discovered tacking as the optimal route and is computationally efficient.

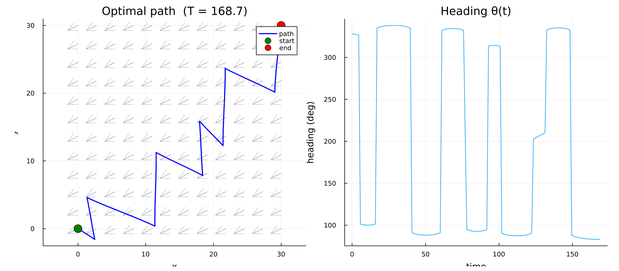

Figure 5: Demonstration of tacking discovered through Ipopt. Comparison of optimal path (left) vs. the heading over time (right).

Conclusions and Next Steps

There is more work to be done, in a few directions. Some we have mentioned, such as re-working the penalty on tacking in order to see more small tacks. Possible further research directions include:

1. The most obvious addition to this work is the penalize sailing over land. This has a straightforward physical interpretation.
2. It is not physically realistic that the wind directly controls the boat's velocity. Instead, the wind should impact the boat's *acceleration*. Solving this optimal problem will yield more accurate results, though is a greater challenge to implement.
3. We also will try putting bounds on how quickly θ can change, since a boat cannot change its heading instantaneously. This alteration place the problem back into the realm of continuous controls and states, which we can perhaps solve with the elegant `solve_bvp` implementation.

Our group has already seen success and we are excited to implement more realistic models that produce more accurate optimal controls for the M&M 100 miler race. We hope to see you in July!

References

[Mar24] Marinette & Menominee Yacht Club, *100 miler race charts*, 2024, Accessed: March 27, 2026.

[RPP11] M Ramirez, Clement Petres, and Frederic Plumet, *Navigation with obstacle avoidance of an autonomous sailboat*, 08 2011, pp. 86–93.

[Sun23] Sunsail, *Racing bvi spring regatta*, 2023, Accessed: 2026-04-04.