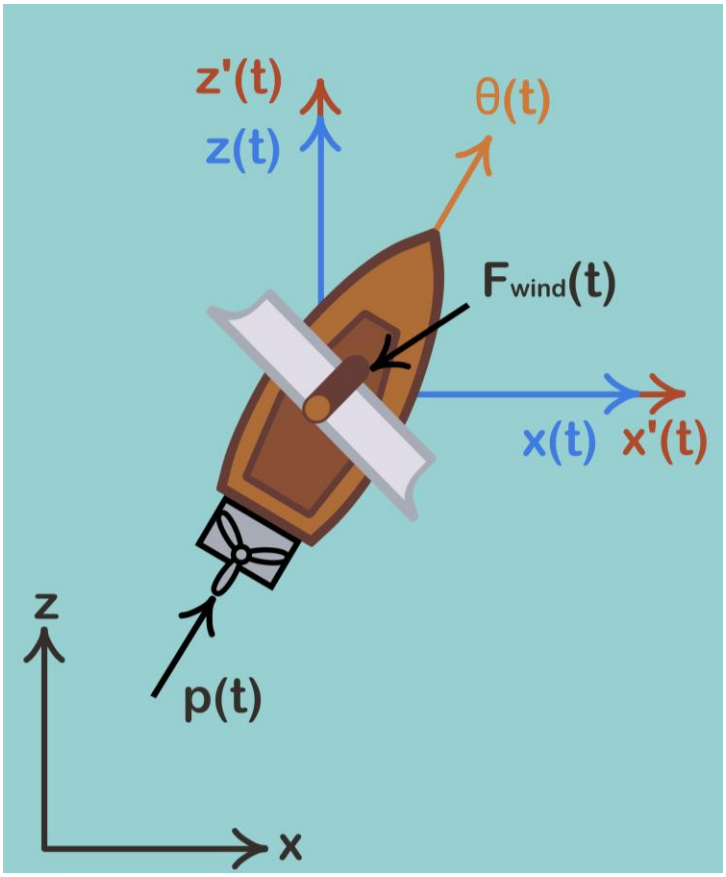
Problem Documentation

The chosen problem was that of a boat on the water, constrained to planar motion with three degrees of freedom: translation in the x-direction, translation in the z-direction, and rotation about its center of mass, with an angle defined from the z-axis as theta.

The system, forces, and coordinate system can be viewed in the illustration provided below.



There are two forces acting on the boat. The propulsion, provided by the motorized propeller on the backside of the craft, and the wind, acting at an angle on the boat's sail.

The assumptions are as follows: the body of water is still and provides no force on the boat, and the sail pivots to always remain perpendicular to the wind force so its force vector is constant relative to the COM of the boat, any movement along the y-axis is negligible and ignored.

The following system of equations defines the discrete dynamics of the system, where $x(t)$ and $z(t)$ are the position coordinates of the boat's center of mass respectively, $\dot{x}(t)$ and $\dot{z}(t)$ are the velocity vectors in those same directions, and $\theta(t)$ the angle of the longitudinal axis of the boat with respect to the z-axis. These define the state of the boat $\boldsymbol{f}(t) = [x(t), z(t), \dot{x}(t), \dot{z}(t), \theta(t)]^T$, and is controlled by the variables of the boat helm turning rate $\omega(t)$ and the motor's propulsion force $p(t)$.

$$x(t+1) = x(t) + \dot{x}(t)\Delta t + \frac{1}{2}p(t)\sin(\theta(t))\,\Delta t^2$$

$$z(t+1) = z(t) + \dot{z}(t)\Delta t + \frac{1}{2}p(t)\cos(\theta(t))\,\Delta t^2$$

$$\dot{x}(t+1) = \dot{x}(t) + \frac{1}{2}p(t)\sin\big(\theta(t)\big)\,\Delta t$$
$$\dot{z}(t+1) = \dot{z}(t) + \frac{1}{2}p(t)\cos\big(\theta(t)\big)\,\Delta t$$
$$\theta(t) = \theta(t) + \omega(t)\Delta t$$

And the controller is defined as:

$$\boldsymbol{u}(t) = [a(t), \omega(t)] = \pi_w\big(\boldsymbol{f}(t)\big)$$

This problem is optimized by minimized the state as an unconstrained problem with respect to w, or

$$\min_w \big|\big|\boldsymbol{f}(T)\big|\big|^2$$

The dynamical system is mathematically modeled as follows.

The force from the wind on the sail as a vector: $[0,0,-d\cdot\Delta t\cdot sin(\varphi),-d\cdot\Delta t\cdot cos(\varphi),0]$, where φ is the (assumed constant direction) angle of the wind force on the sail.

The tensor defining the state: $[0,0,\sin(\theta),\cos(\theta),0]$.

The discretized state vector: $[0,0,\sin(\theta)\cdot\Delta t\cdot prop.accel.const.\cdot p(t),\cos(\theta)\cdot\Delta t\cdot prop.accel.const.\cdot p(t),0]$

Now for Model Predictive Control (MPC), and linearized dynamics of the system, accomplished using Taylor's expansion. This is applied for the state of the system without a controller, at its initial state, where the linear system is given by:

$$f(\boldsymbol{f},u) \approx f(\boldsymbol{f}_0) + A(\boldsymbol{f}-\boldsymbol{f}_0) + Bu$$

Simplified to:

$$f(\boldsymbol{f_0},u) \approx +A\boldsymbol{f} + Bu + c$$

Where the matrices A and B are defined as follows:

$$A = \nabla_x f = \begin{bmatrix} 1 & 0 & \Delta t & 0 & 0 \\ 0 & 1 & 0 & \Delta t & 0 \\ 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 1 \end{bmatrix}$$

$$B = \nabla_u f = \begin{bmatrix} 0 & 0 \\ 0 & 0 \\ \sin(\theta)\,\Delta t & 0 \\ \cos(\theta)\,\Delta t & 0 \\ 0 & \Delta t \end{bmatrix}$$

$$c = [0,0,-d\cdot\Delta t\cdot sin(\varphi),-d\cdot\Delta t\cdot cos(\varphi),0]$$

The optimization problem is redefined, and becomes as follows:

$$\min_{(\boldsymbol{f}(t),u(t))} ||\boldsymbol{f}(T)||^2$$
$$\boldsymbol{s.t.}\, \boldsymbol{x}(t+1) = Af(t) + Bu(t) + c$$
$$f(t)[1] \geq 0$$
$$|u(t)[0]| \leq 2, |u(t)| \leq 1, \forall t = 0, \ldots, T-1, \boldsymbol{f}(0) = f_0$$

Programming

[talk about where the code is]

The Integrated Development Environment PyCharm 2023.3 was used to host and edit the Python code and import both Pytorch and the necessary packages for running the simulation.

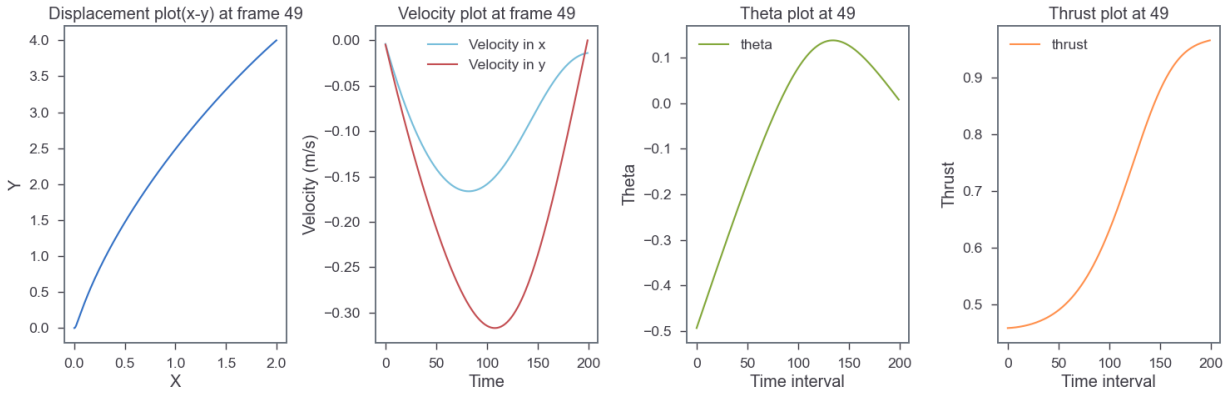
The code modified to the chosen dynamics problem can be found in my GitHub repo, linked here.

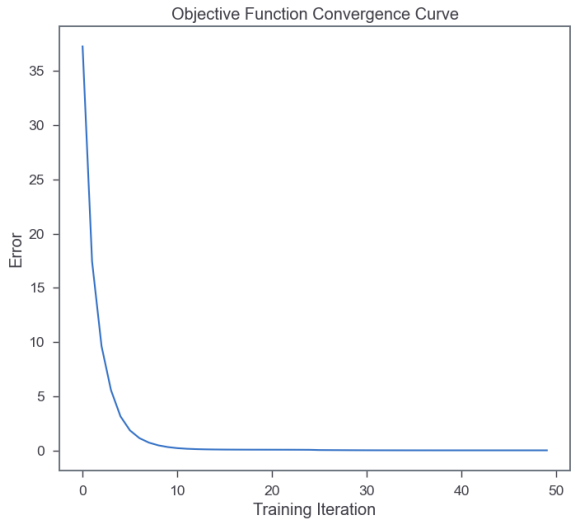
I uploaded the project 1 and 2 code separately, though I ran them together in my original PyCharm file.

Result Analysis

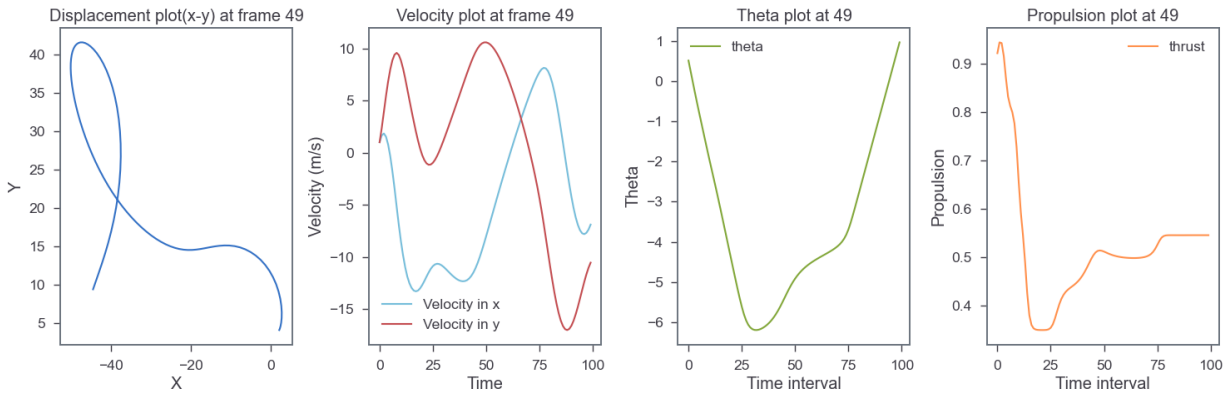
Please explain clearly and in details all issues you encountered and lessons you learned in solving your problem, including incorrect problem formulations, hyperparameter tuning (e.g., for the optimization algorithm), and coding issues (e.g., related to tensor operations).

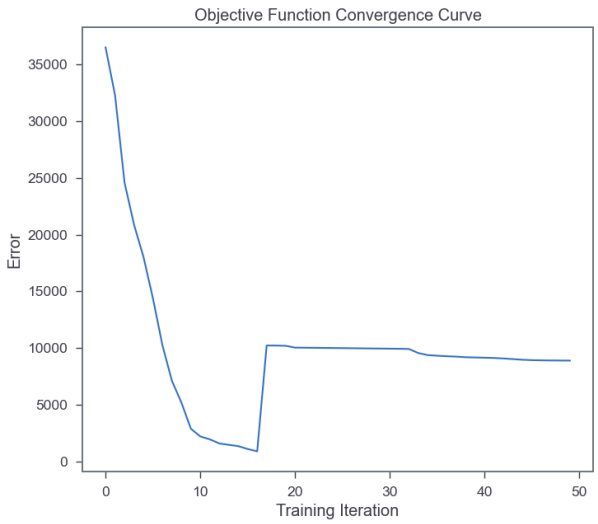
The provided code operated as expected once all the necessary modules were provided, and gave the following plots:





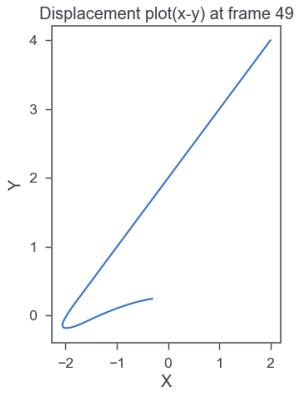
However when modified for the constants for my system, it became unstable, with significant error:

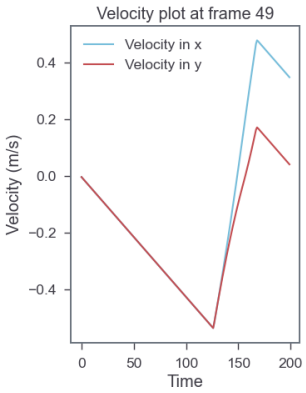


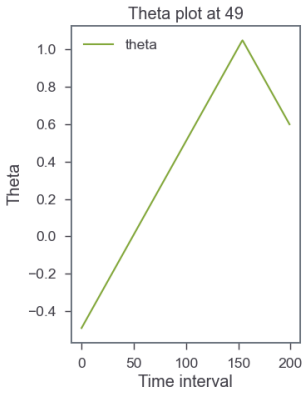


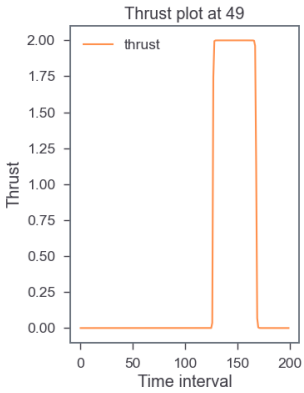
I also had a lot of trouble making the system robust, it tends to break for any input other than the default, especially when adjusting initial conditions.

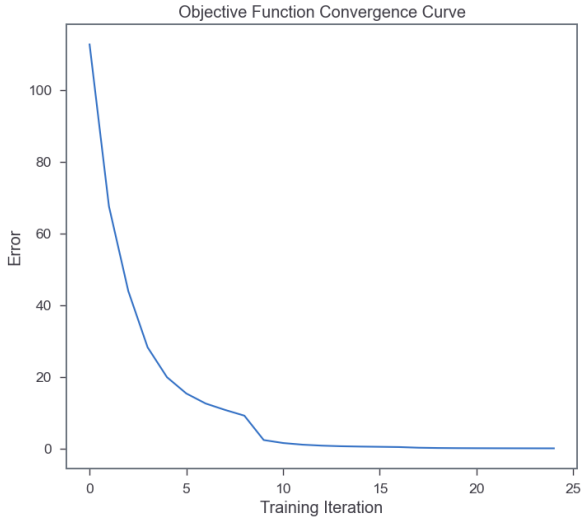
The accelerations took some tuning to make sure the boat didn't overcorrect repeatedly, and that the constants were reasonable for the system in mind.







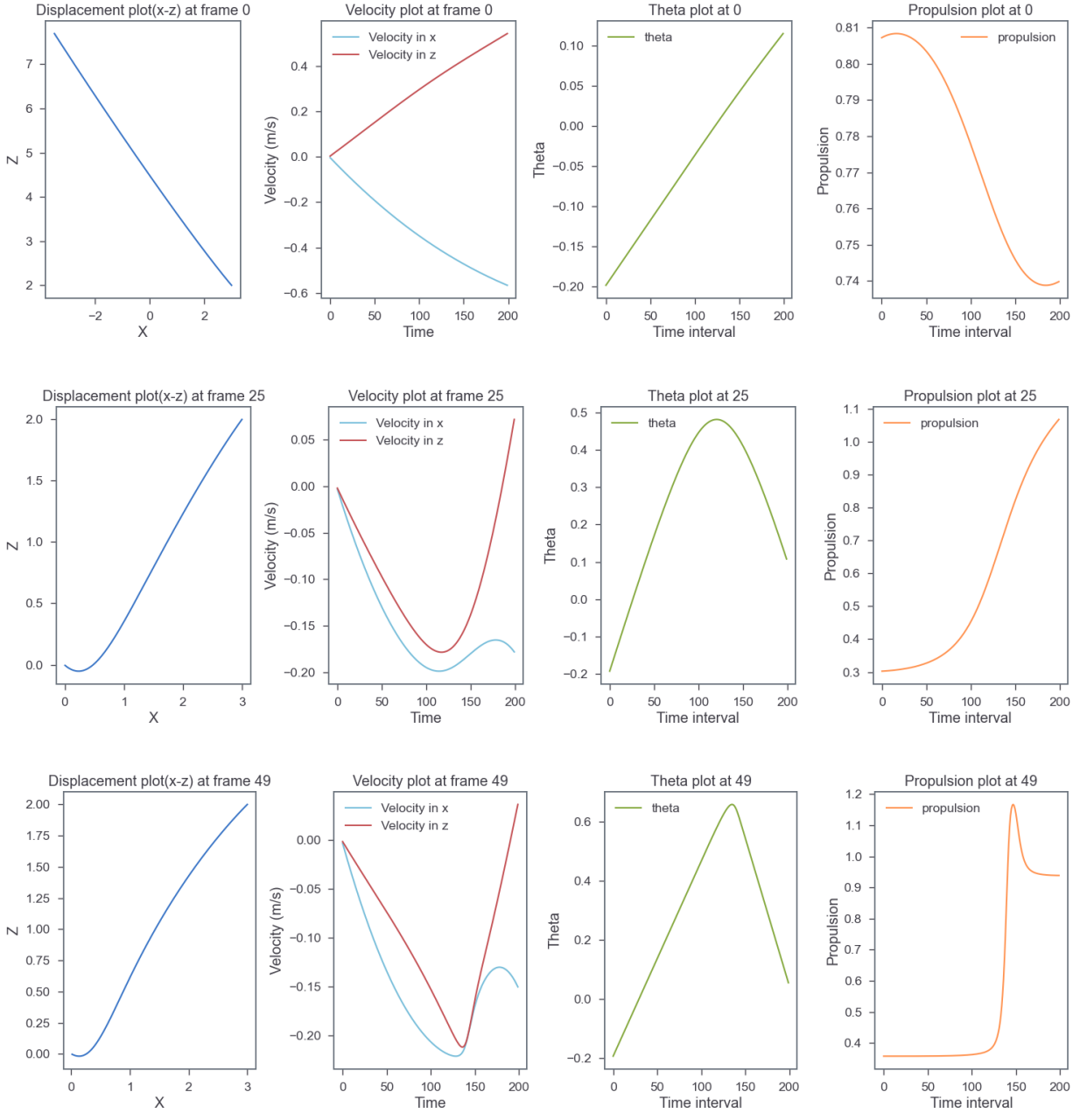


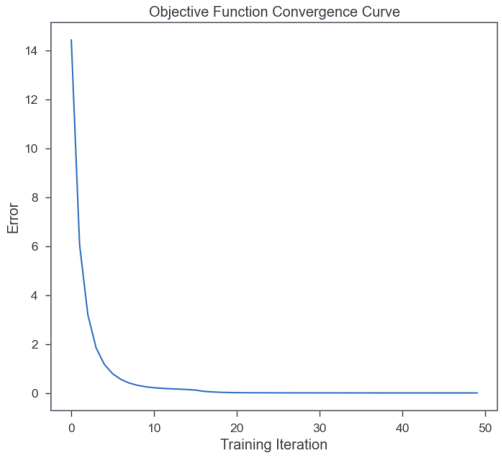


Through experimentation, I found that the system converged fairly rapidly, and that iterations from 15 to 20 were all that were necessary for rough estimates.

The tensors were fairly easy to get working, though I had a couple of problems getting it to accept both wind force directions, with some errors from disparate data types that I had to brute force for part 1.

Final figures can be found below:





I learned that the system of equations will tolerate some finagling and manipulation, but all of the functions are fairly intolerant to changes, and that altering the constants and their relationships can easily cause the SQCP solver to fail or the optimization problem to become infeasible.

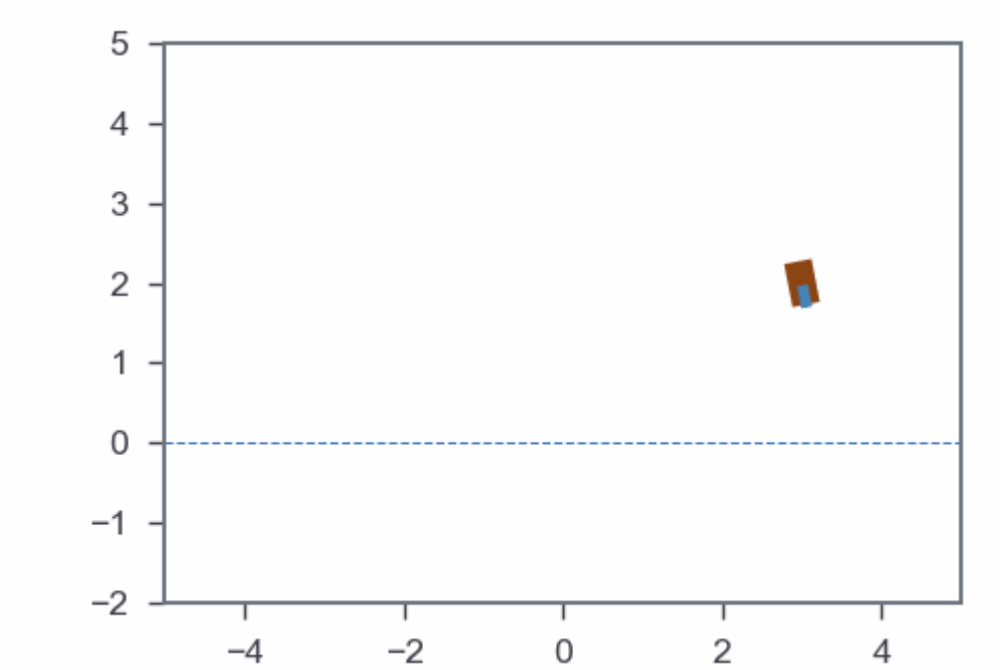
The system converged successfully, but for accurate solutions it took well in the neighborhood of 25+ iterations to produce negligible error.

Visualization

Once you find a converged solution, please do your best to visualize the final results in video and other necessary formats so that other people can easily understand what you achieved.

PART 1

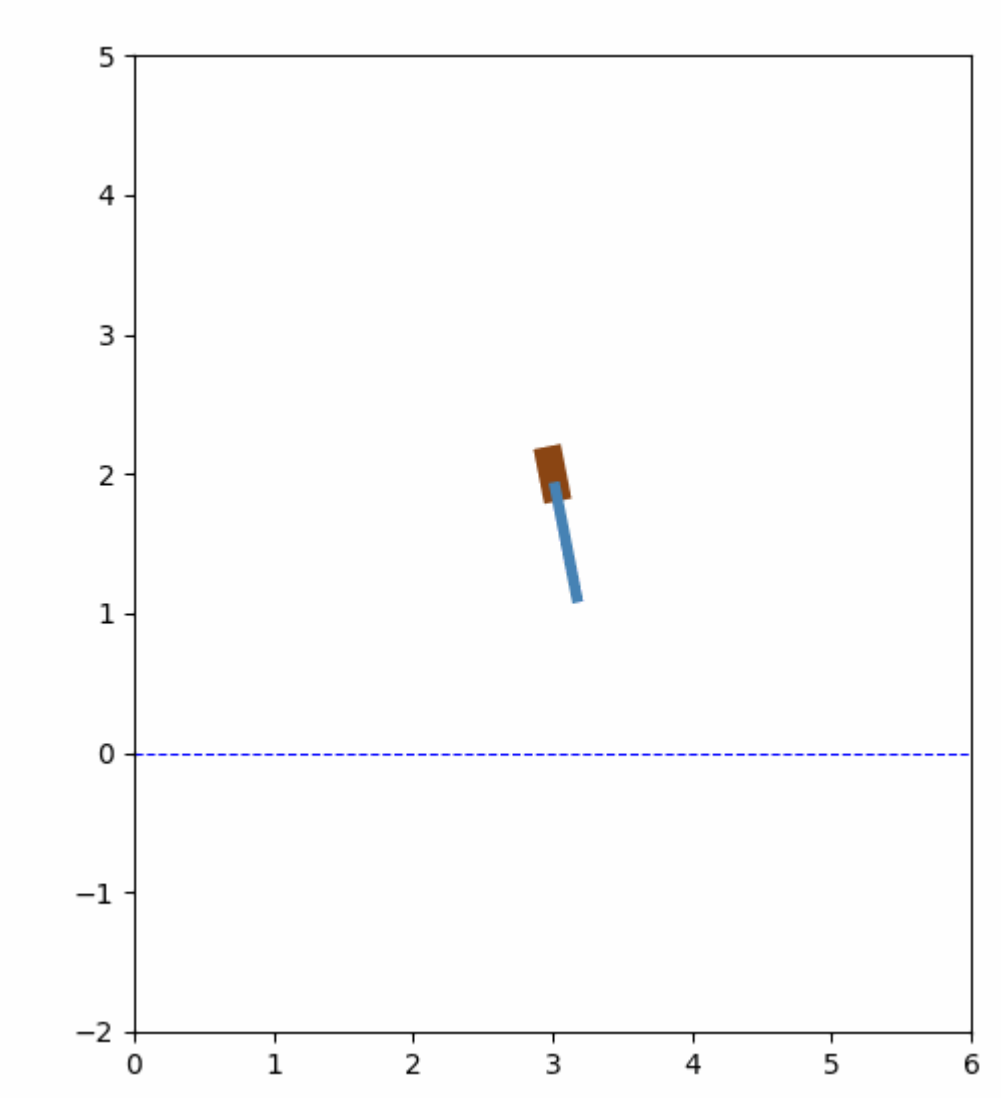
The gif of the simulation for the discrete simulation is below:



PART 2

The gif of my simulation can be found at the provided Github repository, o

The Model Predictive Control animation can be seen below. It displays a much more controlled and well-visualized version of the boat's steering process, with the propulsion force of the boat's motor visible as the blue vector, the boat itself as the brown rectangle, and the path to successfully dock in the desired location marked with the dotted line.


5
4
3
2
1
0
−1
−2
0
1
2
3
4
5
6

- (20%) **Documentation of the problem formulation**
    - Clearly describe the objective function, the variables, the constraints (including the dynamical systems involved), and the assumptions involved in formulating the problem.
- (30%) **Programming**
    - Please push your code to your github repo or a deepnote notebook. Please comment your code so that it is useful to you in the future.
- (30%) **Analysis of the results**
    - Please explain clearly and in details all issues you encountered and lessons you learned in solving your problem, including incorrect problem formulations, hyperparameter tuning (e.g., for the optimization algorithm), and coding issues (e.g., related to tensor operations).
- (20%) **Visualization**
    - Once you find a converged solution, please do your best to visualize the final results in video and other necessary formats so that other people can easily understand what you achieved.