Octobotics Coding Assignment Report

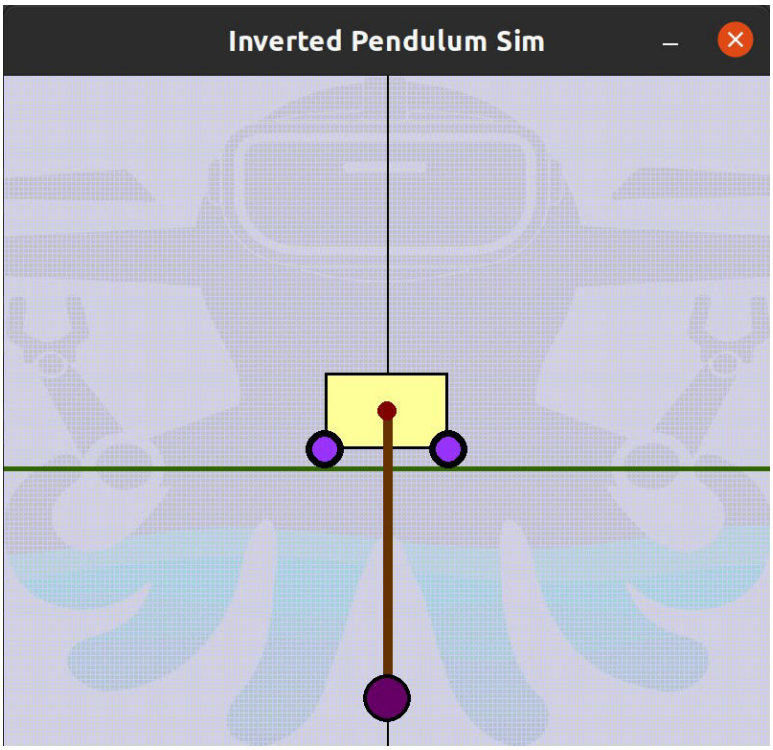
Goal 1: Creating a controller package

To set the initial parameters I made use of the service ***inverted_pendulum/*set_params** specified in the **inverted_pendulum_sim_node.py** python script.
To call the service I created a client node in the **task1_client.py** script and set the initial parameters as specified in the task.
The service and the client scripts are included in the launch file **task1.launch** for convenience.

Run task 1:
roslaunch inverted_pendulum_controller task1.launch

Assumptions:
- Weight and Mass are used interchangeably i.e. gravity is not accounted for.

Output:



Goal 2: Send control input to the pendulum

To provide a sinusoidal force input to the cart, I created a publisher node **task2_pub** that publishes the sinusoidal force to the topic ***inverted_pendulum/*control_force** which cart has subscribed to.
The phase angle for the sin wave is taken as zero.

$$y(t) = A\sin(2\pi ft + \varphi) = A\sin(\omega t + \varphi)$$

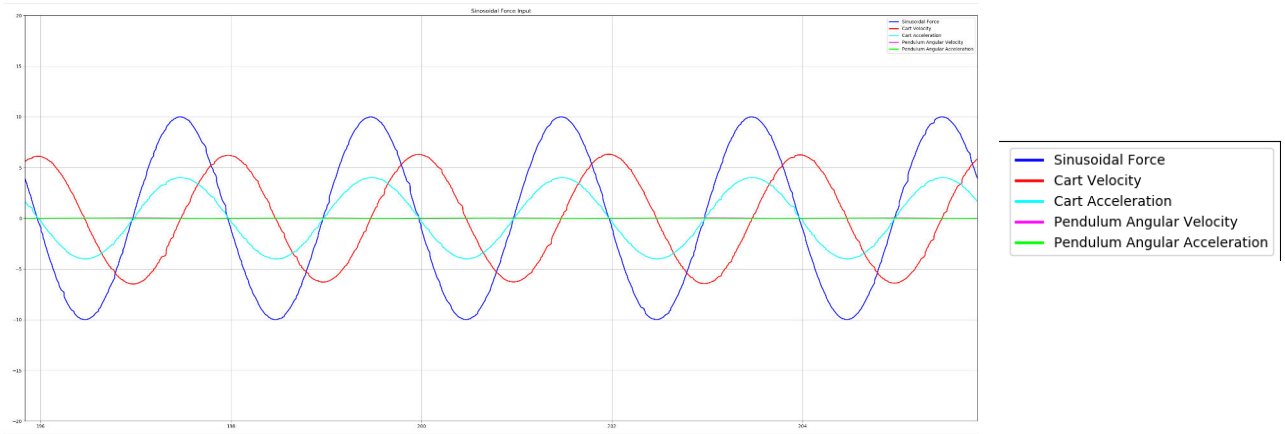
The graphs for cart velocity, acceleration, pendulum angular velocity and acceleration are plotted using **rqt_plot**.

The task1 parameters are taken as the initial condition. The graph plotting, task1 launch file and task2 publisher node are included in the launch file **task2.launch** for convenience.

Result on varying amplitude and frequency is included in the video.

Run task 2:
roslaunch inverted_pendulum_controller task2.launch

Output:



Goal 3: Balance the inverted pendulum

I used PID Control for balancing the pendulum in an inverted position.

First, I created a script called **task3.py** and called the service ***inverted_pendulum/*set_params** to set the initial position of the pendulum as 3 rad.

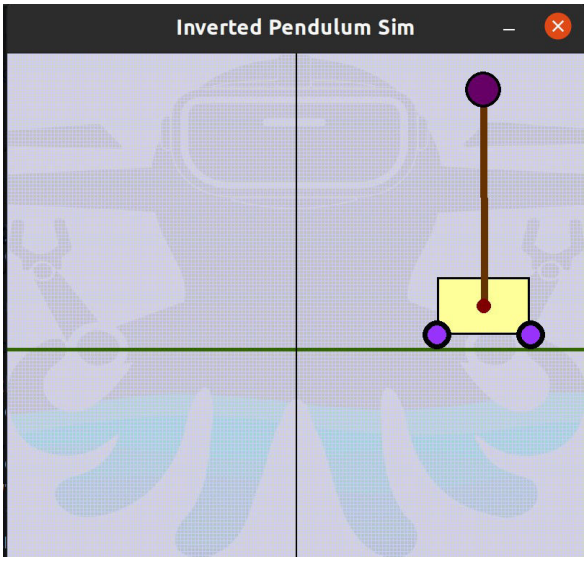
Then, I subscribed to the ***inverted_pendulum/*current_state** topic to get the current theta of the pendulum. In the subscriber's callback function, I made use of the **simple_pid** controller in python to get the value of optimal force for the current position of the pendulum **theta**. Then I manually fine-tuned the Kp, Ki, Kv parameters based on the output in the simulator for optimal performance.

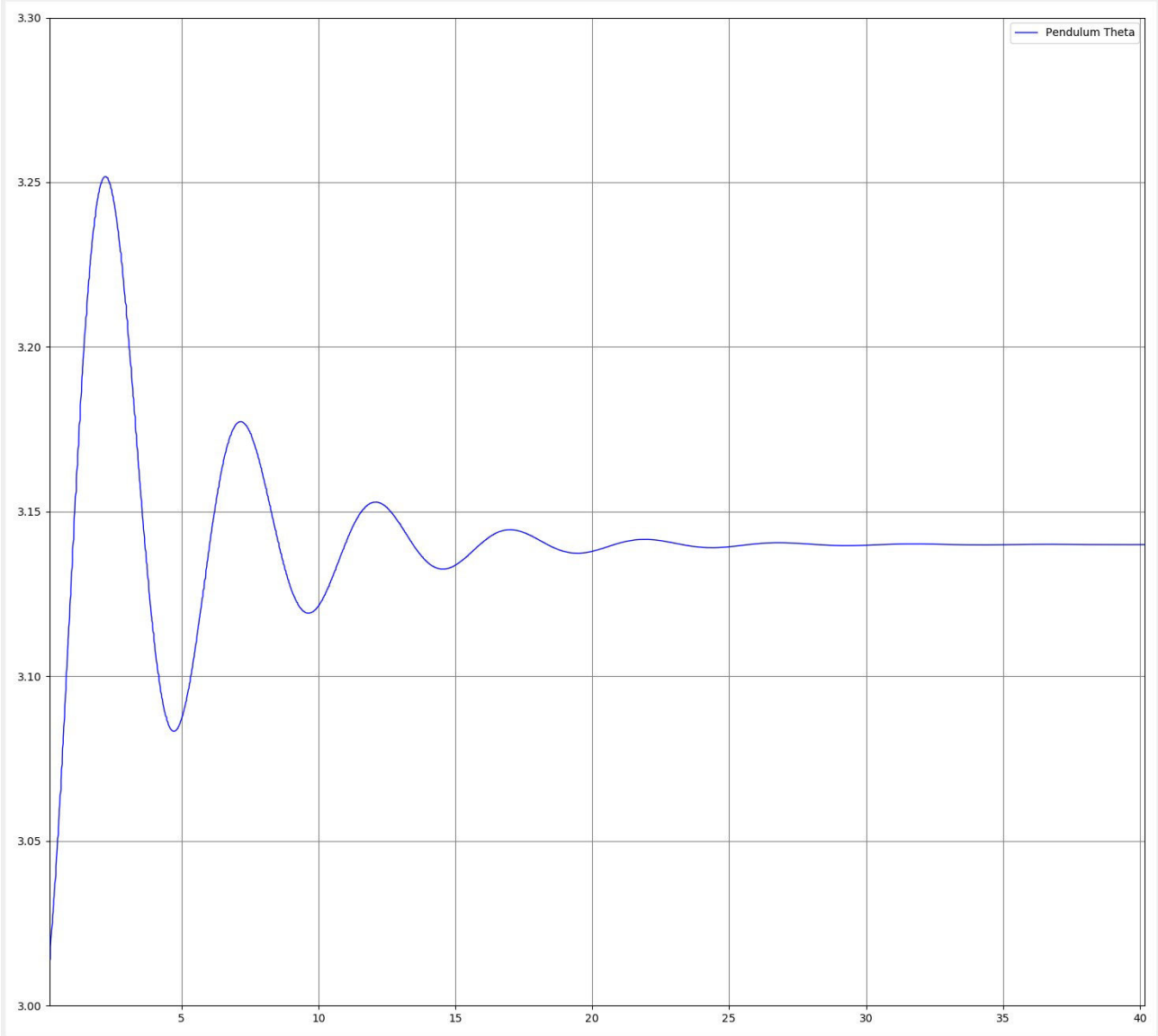
The graphs for theta is plotted using **rqt_plot** which shows the theta position tending to pi rad.

The python script and graph are included in the launch file **task3.launch** for convenience.

Run task 3:
roslaunch inverted_pendulum_controller task3.launch

Output:





Notes:

I used ROS Noetic in python for implementing my solutions.

I tried to make the solutions as simple as I could because of the time constraint and did not experiment much. If the submission deadline was not there, I would've experimented more with the pid controller and tried the lqr implementation also. This was the first time that I implemented PID control.