ME 449 Capstone Project

Christopher Luey

Summary

We create a controller for the youBot mobile manipulator, consisting of a 5R joint arm riding on a base of 4 mecanum wheels. The objective is to use the manipulator to pick up and place a cube at its initial and final configurations. Techniques such as odometry, feedback control, and simple trajectory planning are implemented. Videos of simulations and plots of error are shown as output.

Code Directory

In order to generate the controls, **run best.py**, **overshoot.py**, and **newTask.py**. When each script runs, graphs, logs, and CSV files are generated in the proper results directory. Enhancements can also be enabled for Jacobian singularity avoidance (enhancements = True).

For each situation, the configuration files are as follows: **best_config.py, overshoot_config.py, and newTask_config.py**. Each configuration alters the input parameters. Each configuration file can be altered to change the gains, initial conditions, etc.

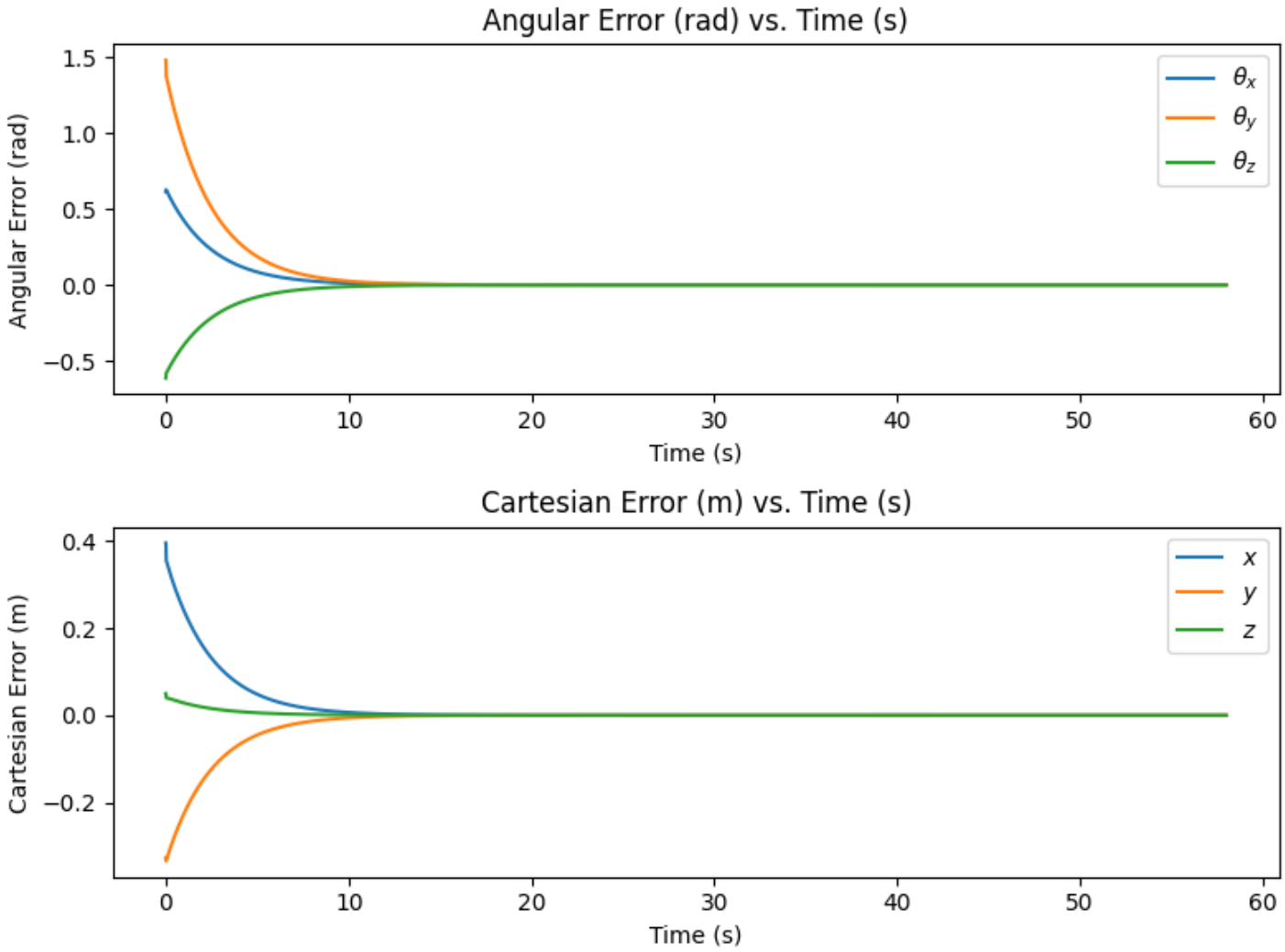
To make the code more understandable, individual files are created for each milestone. **Milestone1.py** includes the NextStep function. **Milestone2.py** includes the TrajectoryGeneration function and **milestone3.py** includes the FeedbackControl function. These scripts can be run by themselves in order to test the correctness of the output to compare with the values on the assignment website.

The **util.py** file includes code useful for generating the error vs. time graphs.

Result Directory

Under the folder named best, is the result of a tuned feedforward plus P controller for the given cube initial and final positions ((1 m, 0 m, 0 rad) and (0 m, -1 m, -π/2 rad)).

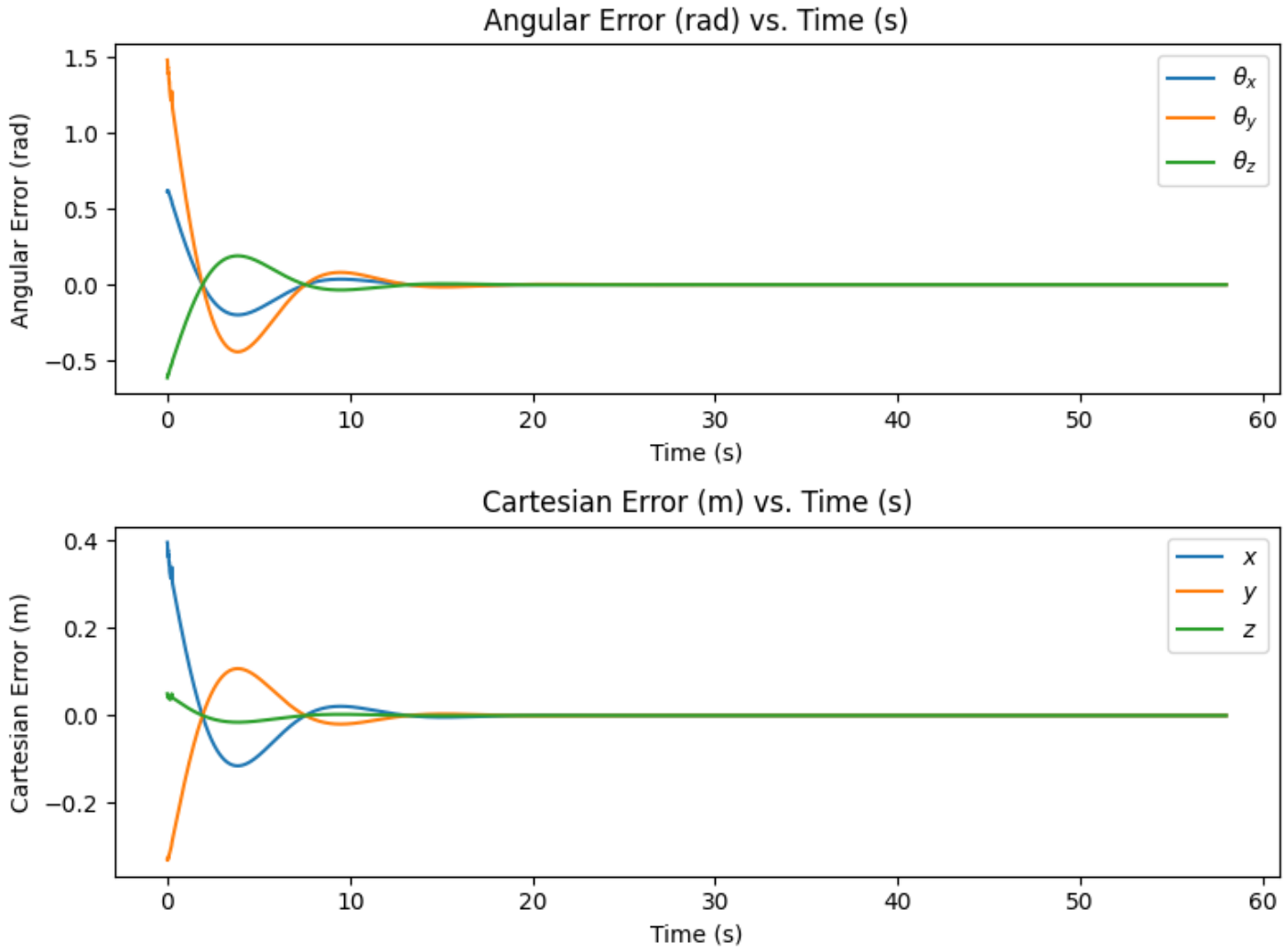
The initial robot configuration used was phi = π/4 rad and x = -0.5 m, everything else 0. The gain used was Kp = (0.4) I_{6x6} per second. Ki = 0 per second squared. The max wheel and joint speeds are 10 rad/s and m/s.



The error approaches 0 within segment 1 of the trajectory (which is 20 seconds long). Error stays near 0 until the end.

Under the folder named overshoot, is the result of a feedforward plus PI controller for the given cube initial and final positions ((1 m, 0 m, 0 rad) and (0 m, -1 m, -π/2 rad)).

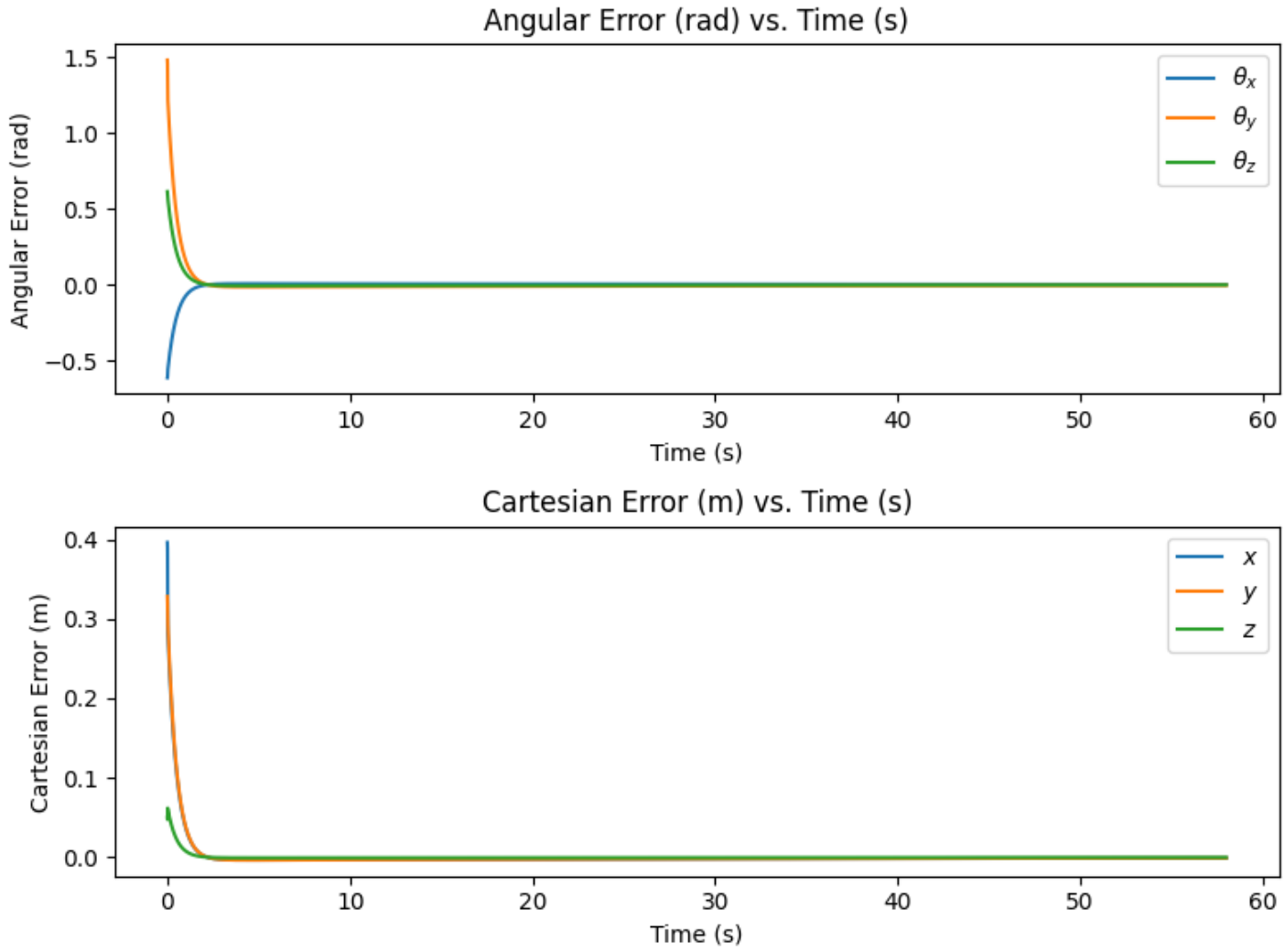
The initial robot configuration used was phi = π/4 rad and x = -0.5 m, everything else 0. The gain used was Kp = (0.6) I_{6x6} per second and Ki = (0.4) I_{6x6} per second squared. The max wheel and joint speeds are 5 rad/s and m/s.



The controller error overshoots and oscillates about and approaches 0 rad and m by the end of trajectory segment 1.

Under the folder named newTask, is the result of a feedforward plus PI controller for the chosen cube initial and final positions ((2 m, 1 m, 1 rad) and (0 m, 1.5 m, -3 rad)).

The initial robot configuration used was phi =- π/4 rad and x = -0.5 m, everything else 0. The gain used was Kp = (2.0) I_{6x6} per second and Ki = (0.05) I_{6x6} per second squared. The max wheel and joint speeds are 99 rad/s and m/s.



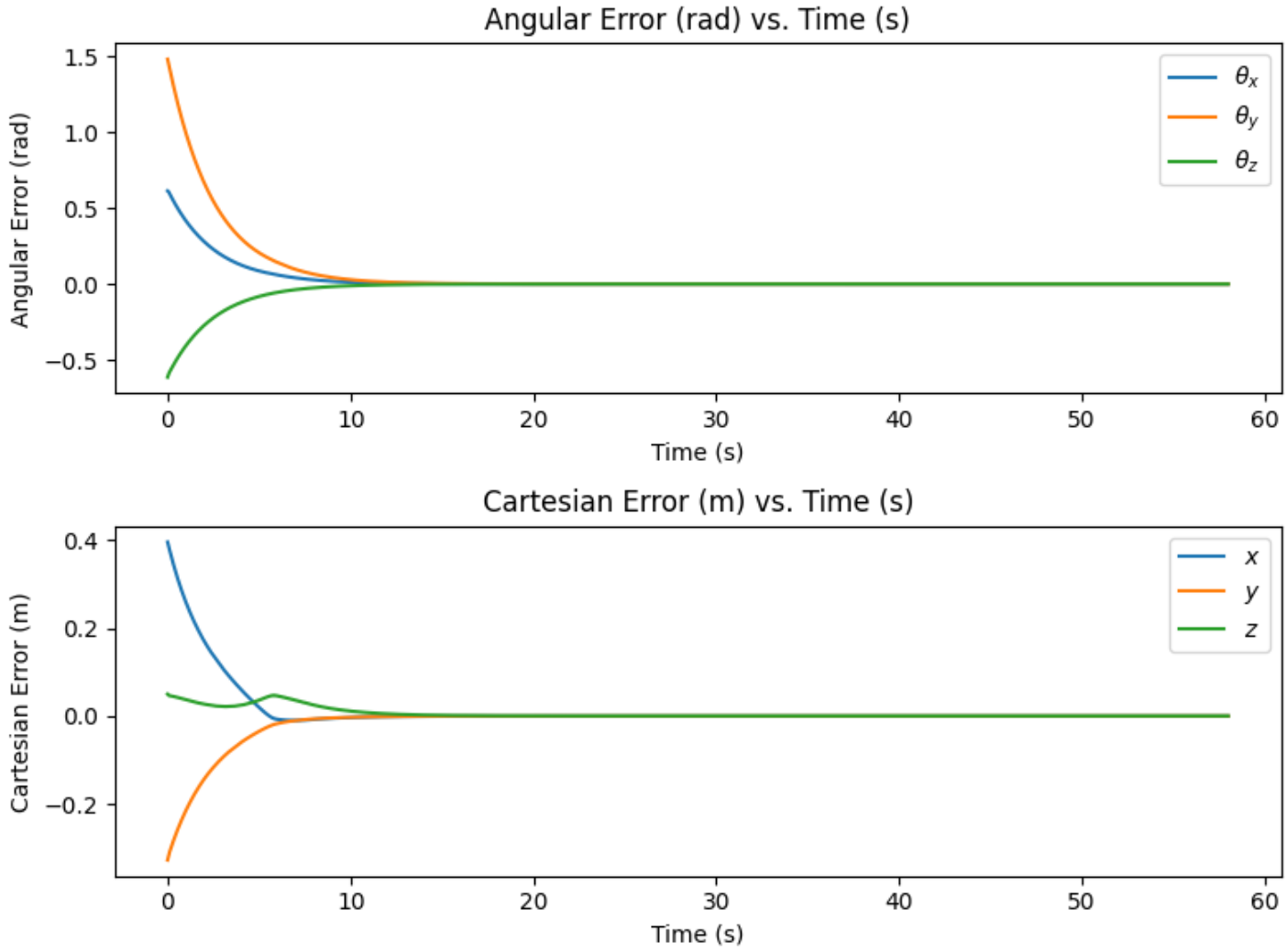
The controller converges to 0 error before the pick up and placement of the cube meaning they are highly accurate for this maneuver.

Enhancements (Singularity Avoidance, Extra Credit)

When enhancements is set to True, jacobian singularity avoidance can be enabled. It detects singularities that are within 10^-2 and considers them to be 0. Thus, no large velocities can emerge after taking the pseudoinverse of the jacobian.

This is the code implemented: *Jt = np.linalg.pinv(J, rcond=10.0**(-2))*

The result of this singularity detection is shown in the “best” configuration.



The z error has an increase in error that occurs at a time of around 6 seconds which indicates that the controller is blocked from following the desired trajectory because of singularity avoidance. The large velocities required to stay on the end effector path cannot be computed. This results in a smoother motion as shown in the video with enhancements enabled. It also increases the activeness of the mobile manipulator base compared to the arms which move more readily.