John Akagi
139 E 400 N
Provo, UT 84604

April 23, 2019

Dr. John Hedengren
EB Brigham Young University
Provo, UT 84606

Dr. Hedengren:

The attached paper details the use of a model predictive controller and a moving horizon estimator in orbital rendezvous. Orbital rendezvous is a key technology that was developed during the Gemini program and used in subsequent NASA programs during the moon landings, space station building and rendezvous, and repair of the Hubble telescope. Despite the continued use, rendezvous remains a challenging problem to solve with multiple approaches being used to solve it.

The approach detailed in this paper uses a model predictive controller, developed with the GEKKO optimization suite, and a moving horizon controller, developed with SciPy packages, to enable a spacecraft to rendezvous with an uncontrolled, orbiting target. The spacecraft is shown to be able to successfully rendezvous with the target. However, this approach is shown to not be as effective if there are large differences in the initial positions of the spacecraft. The estimator is able to determine the actual thrust and the amount of fuel remaining in the spacecraft.

The contribution of this paper is that it is a simple controller that attempts to minimize the difference in the polar location of the two spacecraft. Additionally, it provides a method of estimating remaining fuel reserves without relying on direct measurements of fuel tanks. This method can be used if primary fuel sensors fail or to eliminate the need for sensors inside the tanks, thus simplifying rocket design.

Respectfully,
John Akagi

Main Contributions

- Model predictive controller for allowing two spacecraft in orbit to rendezvous
- First principles model of an orbiting spacecraft with two sets of thrusters
- Moving horizon estimation that allows for estimation of actual thrust based on position and angular velocity measurements
- Accurate estimation of remaining fuel reserves using a moving horizon estimation

Orbital Rendezvous

John Akagi

Abstract— **Orbital rendezvous is an import ability for many space missions such as resupplying the space station or building large structures in space. In this project, a moving horizon estimator is implemented with SciPy packages to estimate the difference in the actual and nominal thrust. The estimator is also able to determine the amount of fuel remaining on the spacecraft. An model predictive controller is used with the GEKKO Python package to allow a spacecraft to rendezvous with an orbiting target. The controller is shown to be able to rendezvous with the target. Although successful, the controller is limited by computation time and resources and the limitations are discussed.**

I. Introduction

The ability for spacecraft to rendezvous is a crucial task for many missions. For example, supply capsules to the International Space Station (ISS) need to be able to match the orbit and position of the ISS in order to dock. Other example of orbital rendezvous include the gradual assembly of the ISS in space, maintenance on the Hubble telescope, and the reunion of the lunar lander and command modules of the Apollo flights after landing on the moon.

Although these tasks are done regularly, orbital rendezvous is rather counter-intuitive due to the physics. For example, changing the velocity of a spacecraft changes the orbit and vice versa. Thus, a spacecraft cannot simply attempt to catch up to another spacecraft by increasing its velocity.

This project uses GEKKO to develop a model and series of objective functions to allow a spacecraft to rendezvous with an orbiting target. We assume that both spacecraft are already in orbit around the Earth. Since the target is unpowered, it only moves based on the gravitational pull of the Earth. Additionally, we assume that both spacecraft are in the same orbital plane and so the problem can be modeled as a 2D problem.

This project also uses a moving horizon estimator to estimate the amount of fuel on the spacecraft. Since fuel in zero gravity can move much more freely than its terrestrial counterpart, devising a system for directly measuring it is difficult at best. Using the moving horizon estimator, the spacecraft can track its fuel usage indirectly, by measuring the effect of thrust on the spacecraft. Since thrust is directly correlated to the amount of fuel used, and since acceleration is directly correlated to the mass of the spacecraft, the fuel usage and amount of remaining fuel can be inferred as the spacecraft moves.

II. Literature Review

Various approaches have been used in order to solve the problem of orbital rendezvous. These approaches have historically used impulse maneuvers where a spacecraft is able to instantaneously change its velocity. This simplifying assumption is able to be used to plan trajectories but since a spacecraft cannot instantly change its velocity, there is often one or more correcting burns needed. Other approaches have modeled on-off thrusters which more accurately represent actual spacecraft engines [1], model predictive control [2], and linear covariance [3].

In [1], the authors use a Hill-Clohessy-Wiltshire model to describe a spacecraft orbiting in a circular orbit and then find an optimal series of inputs assuming variable thrust levels. These inputs are then used to initialize another optimization that finds the optimal inputs assuming the thrusters can only be full on or full off. A model predictive controller was implemented in [2] which was able used in simulations to have a spacecraft rendezvous with a rotating target using dynamically reconfigurable linear constraints. Additionally, they show that they are able to compensate for unmodeled disturbances like solar radiation or air resistance. A third approach is shown in [3] where linear covariance is used and shown to be faster than a similar Monte Carlo approach. Work has also been done in other areas like, for example, in [4] which planned rendezvous trajectories that would ensure collision avoidance even if the thrusters fail on one spacecraft.

The contribution of this project is to use the GEKKO package to plan and execute a trajectory which allows the rendezvous of two spacecraft while also estimating parameters such as the exact thruster outputs to improve rendezvous efficiency. Additionally, the use of the moving horizon estimator to determine current fuel levels allows to accurate fuel measurements without the need for additional sensors inside the fuel tanks.

III. Model

As stated, this simulation consists of two spacecraft orbiting Earth. Consequently, the dynamics of both spacecraft are dominated by the gravitational pull of the Earth such that

$$\mathbf{F}_g = \frac{-\mu \mathbf{r}}{r^3} \tag{1}$$

where μ is the universal gravitational constant times the mass of the Earth, $\mathbf{r}$ is the vector pointing from the center of the Earth to the spacecraft, and r is the distance from the center of the Earth to the spacecraft. Since the Earth's mass is much greater than the mass of the spacecraft, we drop the spacecraft mass from the equation and assume that the spacecraft has no impact on the motion of the Earth. For the target spacecraft, this is the only force acting on it.

For the controlled spacecraft, we assume that there is a main propulsion engine and a series of thrusters to provide

TABLE I: The parameter values used in the simulation. Values are based on the Apollo Command and Service Module.

Parameter	Value
Main Engine Propellant Mass	18410 kg
Main Engine Thrust	91000 N
Main Engine Mass Flow	29.5 kg/s
Attitude Thruster Propellant Mass	155 kg
Attitude Thruster Thrust	440 N
Attitude Thruster Mass Flow	.15 kg/s
Spacecraft Mass	11900 kg
Spacecraft Diameter	3.9 m
Moment of Inertia	108465 m^2kg
μ	3.986e14

attitude control. For both, we assume that the thrust can be commanded with a percentage of the maximum thrust. The main engine provides a force that accelerates the spacecraft in the direction the spacecraft is facing. The attitude thrusters provide thrust that rotates the spacecraft. Since we are assuming a 2D model, the only attitude variable is the heading or yaw of the spacecraft, pitch and roll and not considered.

The force from the main engine is then

$$\mathbf{F}_m = \alpha F_{m\ max} \mathbf{h} \tag{2}$$

where α is the commanded percentage, $F_{m\ max}$ is the maximum force of the engine, and $\mathbf{h}$ is a unit vector describing the direction the spacecraft is facing.

The force from the attitude thrusters is

$$\mathbf{F}_t = \beta F_{t\ max} \tag{3}$$

where β is the commanded percentage and $F_{t\ max}$ is the maximum possible thrust from the thrusters. The total torque on the spacecraft is found as

$$\tau = r \times \mathbf{F}_t \tag{4}$$

where τ is the torque on the spacecraft and r is the distance from the thrusters to the center of the spacecraft.

The linear accelerations due to gravity and the main engine are found by dividing the forces by the mass of the spacecraft and then summing them. The angular acceleration is found by dividing the torque by the moment of inertia of the spacecraft. The position, velocity, heading, and angular velocity are all propagated using these accelerations as well as the current velocities.

A list of all parameters used in the simulation is found in Table I. Parameters are based on the Apollo Command and Service Module using values obtained from released NASA records and Wikipedia. While they may not be entirely accurate, they are accurate enough to demonstrate the effectiveness of the simulation. Note that the attitude thruster values for thrust and mass flow rate are doubled in the simulation since the values given in Table I are for each individual thruster but they are always used in pairs to get an angular acceleration with no linear acceleration.

IV. Estimation Model

When controlling the spacecraft, it is unlikely that the thrusters will fire perfectly and supply the desired acceleration. There is likely going to be some offset such that the thruster acceleration is consistently off by the same amount. Additionally, imprecise throttles, valves, and pumps can further change the amount of thrust that is produced.

While the uncertainty in the actual thrust command can cause problems with regards to the position of the spacecraft drifting, it also creates uncertainty in the amount of fuel remaining. The spacecraft model used in this project assumes that there is no instrument available to directly measure the fuel and that the fuel usage is estimated by integrating the design fuel usage per thruster. As long as the actual fuel usage matches the nominal fuel usage, this method works. However, as mentioned, various source cause the actual usage to vary from the nominal usage. In a worst case scenario, the spacecraft could consistently use more fuel than expected and run out in the middle of a maneuver. This obviously has fatal implications for the spacecraft and any crew members.

An estimator was built in order to gauge the actual thrust of the thrusters and to predict the amount of fuel remaining. The actual thrust was determined by multiplying the commanded thrust by a noise value. The noise value was determined by adding a bias value to a disturbance value. Both of these values were ratios such that a value of 1 indicated that there was no bias or noise.

For the estimation model, we assume that the spacecraft is able to measure its position and the angular velocity of its rotation. Based on these values, and the known thruster commands, the spacecraft is able to minimize the distance between the position it should be in if it was using the design thrust and the position it is currently in. Additionally, the spacecraft is able to estimate its velocity and current heading although these values were not directly used for anything.

V. Estimator Implementation

The estimator was implemented using the minimize and odeint function in the SciPy Python library. The spacecraft was modeled using differential equations that accounted for gravity, a main propulsive thruster, and a smaller control thruster. The commands to the spacecraft were two values to control the two thrusters. The main thruster could take any value between 0 and 1 while the control thruster could take any value between -1 and 1 where a negative value indicated that a negative angular acceleration was being applied to the spacecraft.

The total disturbance was determined by combining the bias and the noise. The bias was set at a fixed value of 1.05 for the main thruster and 0.95 for the control thruster. The noise was selected each timestep from a Gaussian distribution about 0 with a mean of 0.05. The noise of each thruster was independent of the other. The total disturbance was the sum of these two values. The actual thrust was then the commanded thrust times the respective disturbance for each thruster.

A series of thruster commands was decided before the simulation was begun. For each simulation step, the true attitude of the spacecraft was recorded. Then the thursters were fired based on the commanded thrust and noise, and the attitude was propgated forward. The estimated attitude and bias was then used to propagate forward the estimated position. The minimization function in SciPy was used to adjust the bias in order to minimize the difference between the actual and estimated positions over the time horizon. Once the bias was estimated, the estimated attitude was updated.

The objective function being minimized was based around the error between the measured and estimated x position, y position, and angular velocity. For each step in the time horizon, the difference between these values was found, squared, and then summed across all values and all timesteps. Since the error in angular velocity was much smaller than the error in position, that error was weighted with a constant value to increase the error to be approximately of the same order of magnitude. Additionally, the difference between the current bias and the new estimated bias was also found, squared, and added to the objective function. The purpose of this was to slow the change in bias estimation so that a single outlying data point would have a limited effect on bias.

VI. Estimation Results

The estimator was able to accurately determine the bias in the thruster commands and to keep the estimated attitude from deviating from the actual attitude. The simulation was first run with no estimator running in order to determine how much of an effect the bias and noise had in the actual deviation. The nominal bias was set at 1.05 and .95 for the main and control thrusters respectively. The additional noise was set to be drawn from a Gaussian distribution centered on 0 with a standard deviation of 0.05. The simulation was run for a total of 160 seconds with command inputs changing approximately every 30 seconds.

In Figure 1, the estimated and actual fuel levels can be seen when the estimator is turned off. Since the bias results in more or less fuel being used than anticipated, the main thruster fuel is overestimated and the control thruster fuel is underestimated. This causes a divergence in the actual and modeled fuel amounts that continually widens as time goes on.

When the estimator is running, it has a time horizon of 30 seconds. When the estimator is initialized, it waits for data to be gathered over the full horizon and then begins to estimated variables. The nominal bias, total noise, and estimated bias are shown in Figure 2. Although the bias estimation is not perfect, overall, the estimator is able to maintain a general idea of what the nominal bias is. Additionally, an incorrect bias estimation typically does not remain for too long and is eventually corrected.

With a better bias estimation, the estimator is able to accurately determine that attitude of the spacecraft. Although the estimator does get off at some points, is it generally able

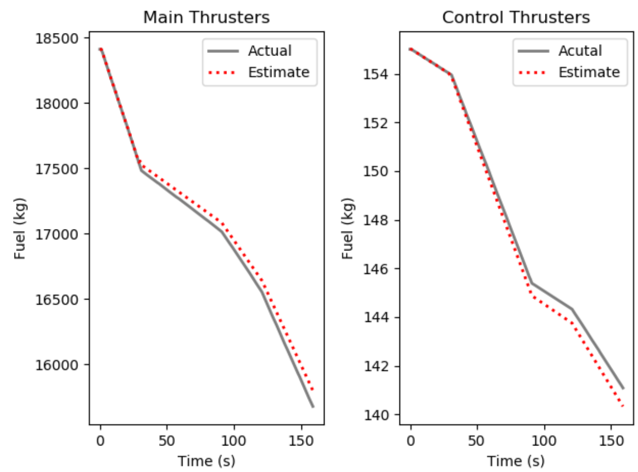

Fig. 1: Actual and estimated fuel levels with no estimator running

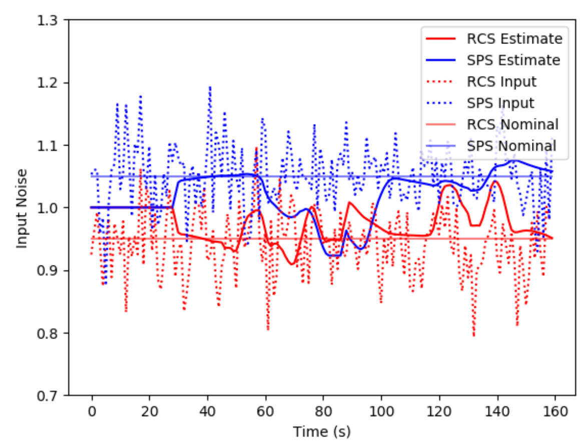

Fig. 2: The actual nominal bias, total command noise, and estimated nominal bias for the thrusters when the estimator was running.

to correct itself so that any errors in the estimated state do not continue to perpetuated. A closeup of the estimated fuel states (Figure 3) shows that the fuel is much more accurate and much less prone to being influenced by the bias. There is still some difference between the actual and estimated states, but overall, it is able correct any errors that arise.

VII. Sensitivity Analysis

A sensitivity analysis was performed in order to determine how the thrusts affect the attitude of the spacecraft. The position and the angular velocity were the control variables that were being measured. However, as a result of the dynamics of the system, the position variables were converted from Cartesian to polar for the controller. As a result, the sensitivity analysis was done using polar coordinates with a radius and an angle.

Since the main thruster controls the magnitude of acceleration and the control thruster controls the direction,

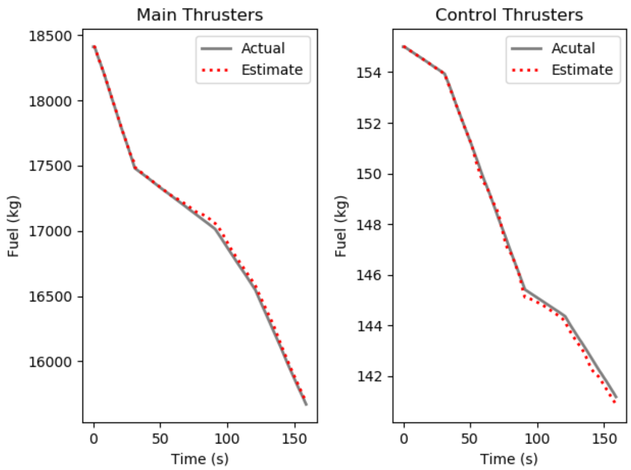

Fig. 3: Actual and estimated fuel levels when the estimator was running

the sensitivity analysis required the examination of how they interact. To this end, varying inputs that cover the total range of inputs for both thrusters were used for the sensitivity analysis. The spacecraft was initialized with the same starting attitude and the difference after one timestep of each combination of inputs was calculated. The attitude of the spacecraft after one timestep of no commands was also found and that attitude was subtracted off from all others to provide a baseline evaluation.

In Table II, the change in the orbit radius of the spacecraft can be seen. The control thrusters have no practical impact on the radius. The radius changes roughly linearly with increasing the main thruster.

In Table III, the angular positions of the spacecraft are shown. The control thruster has no impact if the main thruster is not firing. However, if the main thruster is firing, the interplay between the main thruster and control thruster can be seen. If either input changes, the angular position changes as well.

Finally, in Table IV, the angular velocity of the spacecraft's rotation is seen. In this instance, the main thruster has no impact on the rotation of the spacecraft but it scales roughly linearly with increasing the control thruster. Additionally, the magnitude of the rotation is the same for identical magnitudes of the thruster, which is to be expected.

VIII. Control and Optimization Results

The spacecraft was controlled to match the position and orbit of an uncontrolled target spacecraft orbiting the Earth. The optimizer was assumed to have perfect information about all the state of the spacecraft. The states, in Cartesian coordinates, were converted into a radius and an angle for easier and more intuitive optimization function. The optimizer attempted to minimize the difference in the radii of the two spacecraft, the angle of the spacecraft, and the speeds of the spacecraft. While the speed of the spacecraft was not necessary for matching the position, it theoretically assists

TABLE II: Change in radius, or distance from center of Earth, with varying control inputs. Inputs are in fraction of full thrust for controls and meters for radius. Reported values are the difference between the final attitude and the final attitude if no thrusters were used.

Main Thrust	Control Thrust										
	-1.0	-0.8	-0.6	-0.4	-0.2	0.0	0.2	0.4	0.6	0.8	1.0
0.0	0.005	0.005	0.005	0.005	0.005	0.	0.005	0.005	0.005	0.005	0.005
0.2	0.303	0.303	0.303	0.303	0.303	0.303	0.303	0.303	0.303	0.303	0.303
0.4	0.602	0.602	0.602	0.602	0.602	0.602	0.602	0.602	0.602	0.602	0.602
0.6	0.901	0.901	0.901	0.901	0.901	0.901	0.901	0.901	0.901	0.901	0.901
0.8	1.2	1.2	1.2	1.2	1.2	1.2	1.2	1.2	1.2	1.2	1.2
1.0	1.499	1.499	1.499	1.499	1.499	1.499	1.499	1.499	1.499	1.499	1.499

TABLE III: Angular position of the spacecraft with varying control inputs. Inputs are in fraction of full thrust for controls and degrees for angular velocity. Reported values are the difference between the final attitude and the final attitude if no thrusters were used.

Main Thrust	Control Thrust										
	-1.0	-0.8	-0.6	-0.4	-0.2	0.0	0.2	0.4	0.6	0.8	1.0
0.0	1.817e-11	1.817e-11	1.817e-11	1.817e-11	1.817e-11	0.000e+00	1.817e-11	1.817e-11	1.817e-11	1.817e-11	1.817e-11
0.2	-4.374e-09	-3.835e-09	-3.297e-09	-2.771e-09	-2.232e-09	-1.680e-09	-1.127e-09	-6.114e-10	-5.509e-11	4.755e-10	1.014e-09
0.4	-9.011e-09	-7.986e-09	-6.611e-09	-5.550e-09	-4.474e-09	-3.378e-09	-2.309e-09	-1.220e-09	-1.302e-10	1.056e-09	2.077e-09
0.6	-1.327e-08	-1.172e-08	-1.022e-08	-8.310e-09	-6.714e-09	-5.076e-09	-3.455e-09	-1.813e-09	-1.473e-10	1.429e-09	3.022e-09
0.8	-1.761e-08	-1.549e-08	-1.342e-08	-1.142e-08	-8.953e-09	-6.775e-09	-4.608e-09	-2.362e-09	-2.684e-10	1.859e-09	4.104e-09
1.0	-2.198e-08	-1.931e-08	-1.667e-08	-1.415e-08	-1.139e-08	-8.474e-09	-5.756e-09	-2.974e-09	-3.583e-10	2.310e-09	5.101e-09

TABLE IV: Angular velocity of spacecraft rotation with varying control inputs. Inputs are in fraction of full thrust for controls and degrees per second for angular velocity. Reported values are the difference between the final attitude and the final attitude if no thrusters were used.

Main Thrust	Control Thrust										
	-1.0	-0.8	-0.6	-0.4	-0.2	0.0	0.2	0.4	0.6	0.8	1.0
0.0	-9.065e-01	-7.252e-01	-5.439e-01	-3.626e-01	-1.813e-01	-2.013e-16	1.813e-01	3.626e-01	5.439e-01	7.252e-01	9.065e-01
0.2	-9.065e-01	-7.252e-01	-5.439e-01	-3.626e-01	-1.813e-01	-2.013e-16	1.813e-01	3.626e-01	5.439e-01	7.252e-01	9.065e-01
0.4	-9.065e-01	-7.252e-01	-5.439e-01	-3.626e-01	-1.813e-01	-2.013e-16	1.813e-01	3.626e-01	5.439e-01	7.252e-01	9.065e-01
0.6	-9.065e-01	-7.252e-01	-5.439e-01	-3.626e-01	-1.813e-01	-2.013e-16	1.813e-01	3.626e-01	5.439e-01	7.252e-01	9.065e-01
0.8	-9.065e-01	-7.252e-01	-5.439e-01	-3.626e-01	-1.813e-01	-2.013e-16	1.813e-01	3.626e-01	5.439e-01	7.252e-01	9.065e-01
1.0	-9.065e-01	-7.252e-01	-5.439e-01	-3.626e-01	-1.813e-01	-2.013e-16	1.813e-01	3.626e-01	5.439e-01	7.252e-01	9.065e-01

the optimizer by adding an objective function that helps the controlled spacecraft to continue to match the orbit of the target spacecraft. Since the difference in radius is much larger than the difference in angle, the angle objective function was scaled up so that it was of a similar order of magnitude to the radius function. In attempts previous to this, the optimizer was able to match the orbital radius of the two spacecraft but was never able to get the angles lined up correctly. Once the objective functions were scaled properly, the optimizer was able to control both parameters accurately.

In Figure 4, the distance between the two spacecraft can be seen. Additionally, the relative speed between the spacecraft can also be seen. Although the separation initially increases, it is able to quickly minimize the distance. Additionally, the chaser is able to match velocities with the target well. There are some slight oscillations which indicate that the controller could be tuned a little better to damp out some overshoot, but overall it is able to hold the same velocity well.

Figure 5 shows the difference in angular position of the spacecrafts. As with the separation, the error between the two initial increases but then is able to quickly match the target. Figure 6 shows the orbital radii of the two spacecraft. The chaser spacecraft initially holds its radius and then comes in to match the radius of the target. Again, there is some overshoot so the controller could be tuned to damp out some of that unneeded motion. The delay in the change in radius is due to the fact that the spacecraft first needs to turn to the correct direction before it can start firing its main thruster. This is also seen in Figure 7 (which shows the optimized thrusts) where the control thruster fires for the first 15 seconds or so before the main thruster fires. It should be noted that as the positions get closer, the main thruster begins to use smaller but more frequent impulses to keep the spacecraft on track.

There are a few areas of improvement that could be made in the controller. As noted, it could be tuned better so that there is less overshoot. The thruster model could also be made more accurate to reality. Generally, thrusters are binary and can only be turned on full or turned off. Even those that are variable are likely to not perform well if commanded with fasts oscillations. Thus, a better optimizer would restrict the thruster to binary decisions or attempt to smooth the thrust profile. Finally, the control thruster is giving an almost continually torque on the spacecraft. This is unlikely to be desired since there are structural and biological limitations to how fast the spacecraft can rotate. If a torque is continually applied for too long, the angular acceleration of the spacecraft will cause the crew to pass out and eventual cause the spacecraft to tear itself apart.

Despite the limitations in the thruster model, the controller has proven it is able to optimize a course that minimizes the difference in attitude between itself and a target spacecraft. Overall, the performance is acceptable and the chaser spacecraft meets all the requirements placed on it.

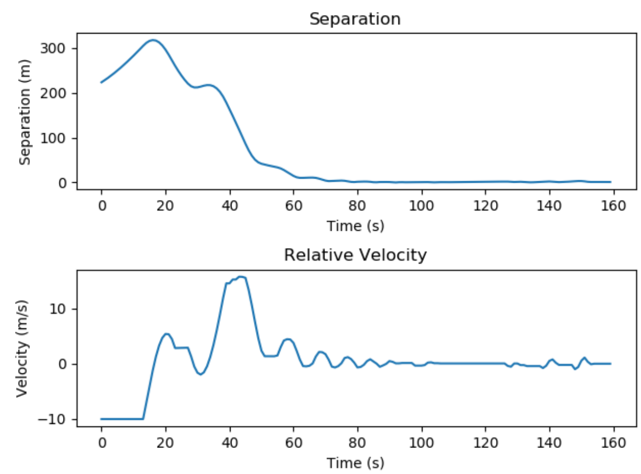

Fig. 4: The distance between the two spacecraft as a function of time can be seen in the top graph. The difference in velocity can be seen in the bottom graph.

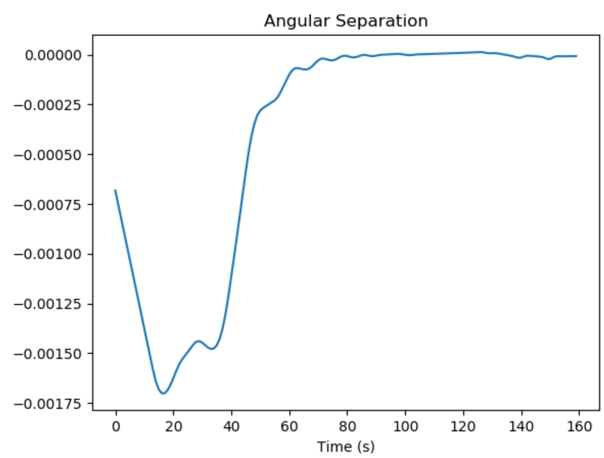

Fig. 5: The difference in angles of the spacecraft.

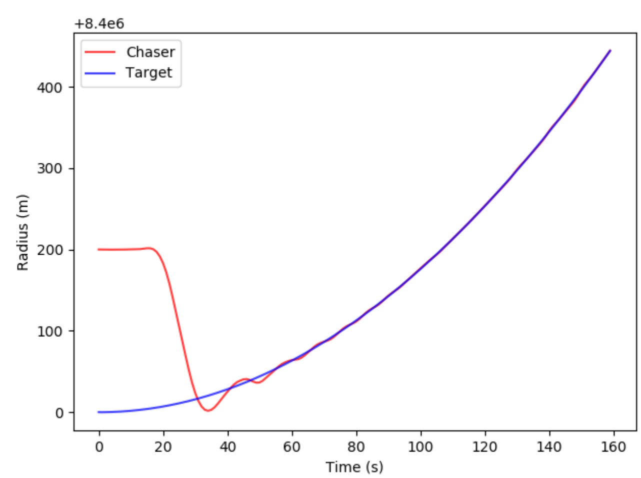

Fig. 6: The orbital radius of the chaster and target spacecraft as a function of time.

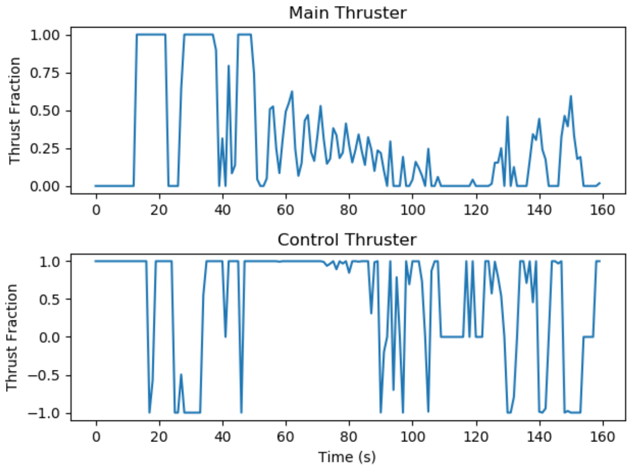

Fig. 7: The chosen thrusts for the main and control thrusters as a function of time.

IX. Conclusion

The MPC controller was shown to work effectively and to be able to match the position and velocity of a target spacecraft while minimizing fuel usage. The MHE was also shown to be able to estimate the offset in the actual thruster force and to correctly estimate the amount of fuel that was remaining on the spacecraft. Although these results were promising, tests with greater separation between the spacecraft and target were unsuccessful due to high computation requirements, large time horizons, and incorrect angle wrapping. Further development on this controller should implement angle wrapping as one method to allow the controller to function better with large separations. Additionally, the thruster inputs should be constrained to on-off values in order to better model the dynamics of an actual spacecraft.

References

[1] R. Vazquez, F. Gavilan, and E. F. Camacho, *Trajectory planning for spacecraft rendezvous with on/off thrusters*, vol. 44. IFAC, 2011.

[2] S. Di Cairano, H. Park, and I. Kolmanovsky, "Model Predictive Control approach for guidance of spacecraft rendezvous and proximity maneuvering," *International Journal of Robust and Nonlinear Control*, vol. 22, no. 12, pp. 1398–1427, 2012.

[3] R. S. Christensen and D. Geller, "Linear covariance techniques for closed-loop guidance navigation and control system design and analysis," *Proceedings of the Institution of Mechanical Engineers, Part G: Journal of Aerospace Engineering*, vol. 228, no. 1, pp. 44–65, 2014.

[4] L. S. Breger and J. P. How, "Safe Trajectories for Autonomous Rendezvous of Spacecraft," *Journal of Guidance, Control, and Dynamics*, vol. 31, no. 5, pp. 1478–1489, 2008.