Design of a Spacecraft Docking Maneuver MPC Algorithm with Debris Avoidance

Aditya Gupta, Asees Sarna, Colin Jensen, Pranjal Sinha, Rithin Venkatesh

Abstract*—The concern for safe and reliable space docking is becoming paramount as the market for space missions exponentially grows. Thus, MPC will be applied for the efficient docking of a spacecraft given stationary debris to a stationary platform. Ideal trajectories are generated with fuel consumption and time to dock as non-linear cost functions to be minimized. In addition, to reduce the risk of damage during docking, minimizing jerk is also considered as a cost. The main constraints of the system are on thrust, approach velocity, debris position, debris velocity, and spacecraft positioning within the Line-of-Sight cone from the docking port[2]. There are three unique factors in our solution. First, we are considering debris location within the calculation of our trajectory. Second, trying to get the spacecraft docked to the exact location on the stationary platform. Lastly, docking force is also minimized to avoid forceful collision with the target platform. *Link to YouTube video

I. Introduction

Space commercialization is an ever-growing industry in today's age. Our project goal was to gain exposure within this market by creating an MPC docking program that factors in environmental parameters and operating conditions. As our starting point, we used the reference paper *Nonlinear model predictive control for spacecraft rendezvous and docking with a rotating target*. We devised our own set of input and output parameters through research into existing spacecraft specifications. We started out by taking the cost function provided in the above-mentioned paper, and made changes to it so we were tracking errors in states. Moving on to the constraints, there was not a lot going on in here, there were state constraints and input constraints in addition to discretization constraints. The place where we were heavily dependent on intuition was choosing the lower an upper bounds for different state parameters. The reason being that a lot of this was not given in the reference paper and thus we had to make reasonable assumptions about these. In addition, we also had to make some reasonable assumptions when it came to choosing some constants for calculations, like platform radius, parameters related to the spacecraft's size and more. While the paper utilized MATLAB Simulink, we used Pyomo and IPOPT as the solving method for the system. We also included the calculation of error between closed and open loop predicted trajectories generated via MPC. Finally, our program included the additional feature of calculating trajectories to avoid stationary debris lying within the docking path.

II. Problem Formulation

To begin modeling this goal, the dynamics of the system must be defined. The general dynamics equations describing orbital relative motion can be given by the Clohessy-Wiltshire (CWH) equations, and are given by

$$\ddot{x} = 3n^2x + 2n\dot{y}, \qquad \ddot{y} = -2n\dot{x}, \qquad \ddot{z} = -n^2z$$

where n is $\sqrt{\frac{\mu}{a^3}}$ and a is the targets orbital radius, and μ is the standard gravitational parameter [1].

Specifically, these equations will describe the relative position of a spacecraft to a target through a time when both are in respective orbits. By making an assumption while considering the problem statement, these equations can be simplified. The assumption is that in the model, the spacecraft and target docking position is close enough under a small enough time span that the CWH equations can simply become a double integrator dynamical system. This means that the dynamics are modeling a simple mass in space under a time-dependent force [3].

Thus, by simplifying the number of degrees of freedom for translational and rotational motion, to two and one respectively, the double integrator dynamics can become

$$\ddot{x} = \frac{F_x}{m}, \qquad \ddot{y} = \frac{F_y}{m}, \qquad \ddot{\theta} = \frac{\tau}{I_z}$$

where m is the mass of the spacecraft, F is the force, I is the moment of inertia for a point mass, mR^2, and τ is the torque. Note that R is the radius from the axis of rotation [2]. The moment of inertia for a point mass was chosen for simplicity due to the variety of spacecraft currently. Another moment of inertia that could be effective, could be for a cylindrical rod about its center, which is defined as $\frac{1}{12}mL^2$, where L is the length of the rod.

Having derived the dynamics of the spacecraft's motion during docking, the above equations can be used to form a state-space representation of the problem. This representation will allow us to model our problem into an optimal control problem, which can be written as:

$$\dot{\mathbf{x}} = A\mathbf{x} + B\mathbf{u}$$

The chosen states and control inputs for the problem can be formulated into the following vectors:

$$\mathbf{x} = \begin{bmatrix} x \\ y \\ \theta \\ \dot{x} \\ \dot{y} \\ \dot{\theta} \end{bmatrix} \qquad \mathbf{u} = \begin{bmatrix} F_x \\ F_y \\ \tau \end{bmatrix}$$

The choice of states can be explained by observing the dependence of the position and heading variables in the dynamics equations described earlier. The choice of inputs is also explained by the dynamics but also can be considered as parameters that can be adjusted or "controlled" in order to affect the motion of the spacecraft. Using these state and input vectors, the state and input transition matrices, A and B respectively, can be formulated as:

$$A = \begin{bmatrix} 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 \end{bmatrix}$$

$$B = \begin{bmatrix} 0 & 0 & 0 \\ 0 & 0 & 0 \\ 0 & 0 & 0 \\ \frac{1}{m} & 0 & 0 \\ 0 & \frac{1}{m} & 0 \\ 0 & 0 & \frac{1}{I_z} \end{bmatrix}$$

Note that these lead to the following discrete-time model of the system:

$$\mathbf{x}(k+1) = A_d\mathbf{x}(k) + B_d\mathbf{u}(k)$$

Where $A_d = I + T_s \cdot A$ and $B_d = T_s \cdot B$ using Euler discretization with sampling period $T_s = 2s$.

It is crucial to consider constraints in any MPC problem and for this application, an important constraint to formulate is on the input, the thrust force, so as to avoid collision between the spacecraft and platform.

$$|u_1(k)| \leq u_{max}, \quad |u_2(k)| \leq u_{max}$$

u_{max} was chosen to be 1.5 N which represents a reasonable maximum possible thrust force value for both directions for docking. As to define upper and lower limits for the state variables, the following values were chosen:

$$\begin{bmatrix} -100 \\ -100 \\ -2\pi \\ -100 \\ -100 \\ -10 \end{bmatrix} \leq \mathbf{x} \leq \begin{bmatrix} 100 \\ 100 \\ 2\pi \\ 100 \\ 100 \\ 10 \end{bmatrix}$$

The limits for the position states, x and y, are chosen to confine the problem to a rectangular region around the platform as that is the region of interest. The heading angle is set to rotate to represent a full rotation range of 360 degrees so that the spacecraft can maneuver accordingly in tight spaces and orient itself to navigate to the platform freely.

The next set of constraints will focus on the debris avoidance aspect of the model. These constraints serve two purposes: for the spacecraft to avoid collision with debris in its trajectory, and so that the spacecraft enter the docking position from the correct side instead of going "through" the docking station. The constraints are similar and arise from using the equation of a circle.

$$x^2 + y^2 \geq r_{debris}^2$$

$$x^2 + y^2 \geq r_{station}^2$$

Thus, all of the constraints of the system have been defined.

The next piece of the model is to define the MPC cost function. This cost function in this model is chosen based on error rather than a terminal condition. This was done in case the spacecraft could not reach the exact docking coordinates in the specified time steps. Thus, following the framework of a finite horizon LQR, our cost function takes the form:

$$J = (\mathbf{x}(N))^T P(\mathbf{x}(N)) \quad + \sum_{i=0}^{N-1} ((\mathbf{x}(i))^T Q(\mathbf{x}(i)) + \mathbf{u}(i)^T R\mathbf{u}(i))$$

where $\mathbf{x}(N)^T P(\mathbf{x}(N))$ is the terminal cost, with P as the terminal weight, $\mathbf{x}(i)^T Q(\mathbf{x}(i))$ is the cost from states, with Q as the state weight, and $\mathbf{u}(i)^T R(\mathbf{u}(i))$ is the cost from input, with R as the input weight.

Then, instead of using this, an error is introduced for the state vector, x. Thus, the cost function would model the minimization of this error as the minimization of the cost and use this as a feasible solution. Here the x_d variable is the terminal condition for the state variables. Thus, after the substitution of $x - x_d$ for x, the cost function becomes

$$J = (\mathbf{x}(N) - \mathbf{x}_d(N))^T P(\mathbf{x}(N) - \mathbf{x}_d(N)) \quad + \sum_{i=0}^{N-1}((\mathbf{x}(i) - \mathbf{x}_d(i))^T Q(\mathbf{x}(i) - \mathbf{x}_d(i)) + \mathbf{u}(i)^T R\mathbf{u}(i))$$

Note that there needs to be no error added to the input, as these values are exact throughout the model and for the initial and terminal conditions.

In the cost function, three important parameters in an MPC problem are the weight matrices P, Q, and R. While Q and R are manually tuned to provide more weights on certain states and inputs, P is calculated as the positive-definite solution of the discrete Riccati equation below:

$$P_k = A^{'}P_{k+1}A + Q - A^{'}P_{k+1}B(B^{'}P_{k+1}B + R)^{-1}B^{'}P_{k+1}A$$

The manual tuning of the Q and R matrices led to the following values:

$$Q = \begin{bmatrix} 10 & 0 & 0 & 0 & 0 & 0 \\ 0 & 10 & 0 & 0 & 0 & 0 \\ 0 & 0 & 10 & 0 & 0 & 0 \\ 0 & 0 & 0 & 300 & 0 & 0 \\ 0 & 0 & 0 & 0 & 300 & 0 \\ 0 & 0 & 0 & 0 & 0 & 500 \end{bmatrix}$$

$$R = \begin{bmatrix} 10 & 0 & 0 \\ 0 & 10 & 0 \\ 0 & 0 & 10 \end{bmatrix}$$

It can be seen that the weights on the derivatives of the first three states are significantly higher since the velocity and angular rate of the spacecraft were observed to be important when calculating the cost of the problem. This is because a change in the value of the last three states, the derivatives, affects the value of the first three states significantly and not the other way, so factoring in the velocity, both linear and angular, of the spacecraft is more important than the position itself. Note that the magnitude of the scalar values which act as weights for each state and input are not critical themselves but it's the relative ratio between the states which plays a role.

Moreover, the weights on inputs as seen in R is equal meaning that the thrust in both directions and the torque applied are equally important for the formulation of the problem.

III. Results

The main results that we found were for spacecraft docking into a non-moving platform without debris, and transitioning into the spacecraft docking into a non-moving platform with large debris and multiple debris. The parameters used for this simulation were a spacecraft weight of 4000kg, the Radius of the docking platform of 2m, and a sampling time of 2s.

This plot indicates the trajectory of the spacecraft given a non-moving platform. In addition, the docking position on this particular maneuver is within the line of sight of the spacecraft. As a result, the trajectory to the docking point is linear as shown. The spacecraft starts at a point (30,30) meters away from the platform which has a center at point (0,0).

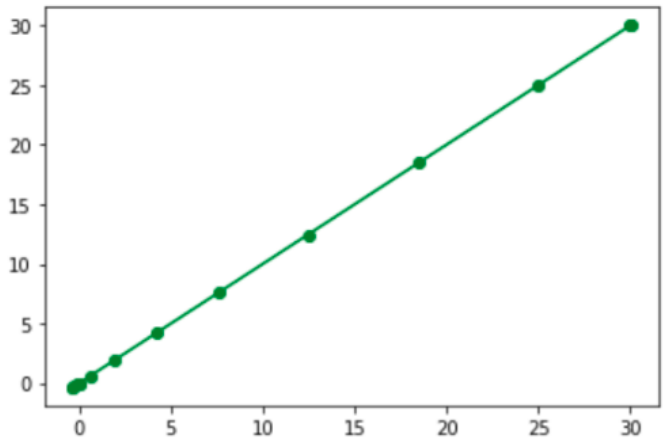

Fig. 1. Spacecraft docking given a docking position within the line of sight.

This is significant as it provides a solid framework that the MPC model is creating an optimized trajectory for the spacecraft to a specified docking position. Note that the linear shape arises from the spacecraft having an initial position that is in the line of sight of the docking position. The next step is to verify that the MPC model can successfully create a trajectory in which the spacecraft is not in the line of site.

The following plot indicates the trajectory of the spacecraft given a non-moving platform. In addition, the docking position on this particular maneuver is not within the spacecraft's line of sight. As a result, the trajectory to the docking point is curved as shown. The spacecraft starts at a point (5,5) and docks to the platform which has a center at point (3.5,3.5).

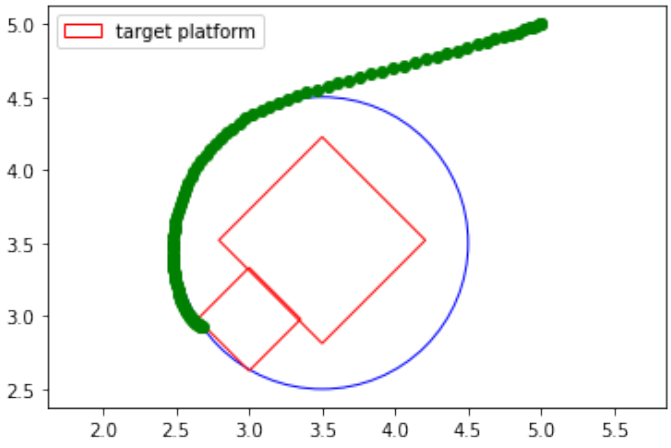

Fig. 2. Spacecraft docking given a docking position outside of the line of sight.

This next plot provides even greater support for a successful MPC simulation. Having the spacecraft take a circular trajectory as it nears the docking station is essential for safe and effective docking. If the circular constraints were not added, this plot would not account for the docking station and thus the spacecraft would have a trajectory that runs through the station rather than around.

The final plot below portrays spacecraft docking with asteroid obstacles to a non-rotating platform. The docking platform is not within the line of sight of the spacecraft. Furthermore, two asteroid obstacles are introduced to this problem, resulting in the spacecraft taking a somewhat curved trajectory in order to avoid the two asteroids, which have equal sizes. In this case, the spacecraft starts at a point (10, 10) meters away from the docking platform, centered at the point (0, 0). To achieve this simulation, the MPC horizon, N, was increased in order for the system to react to the upcoming asteroids.

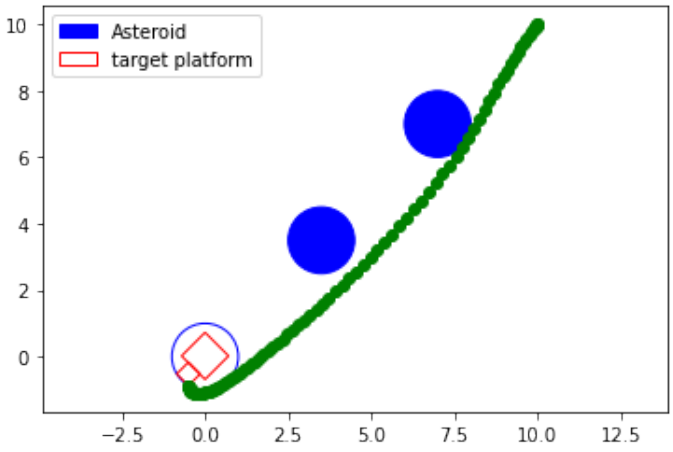

Fig. 3. Spacecraft Docking with Two Asteroid Obstacles

This plot is the accumulation of the previous two plots. Here, the MPC is solidified as the spacecraft successfully docks to the specified position while having debris within the trajectory. This can be majorly seen by both the curve trajectory around the blue asteroids and the curved trajectory, as seen before, around the docking station. The final step would be to run the MPC for a given horizon to show the difference between closed and open loop solutions.

The open-loop and closed-loop trajectories predicted by the simulated MPC can be seen in the following graph. This simulation was created with Prediction Horizon: $N = 100$ and Simulation Horizon: $M = 150$. It can be observed that the open-loop trajectory diverges from the closed-loop trajectory at each step due to it being a sub-optimal path. The closed-loop trajectory ends at the terminal location which is the point (2.75, 3.0).

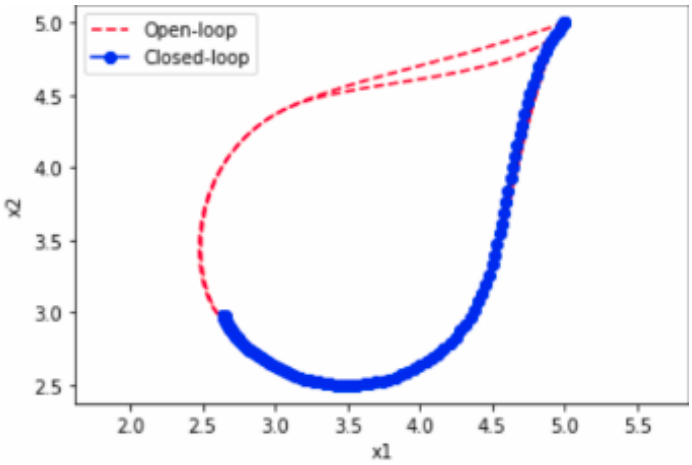

Fig. 4. Closed-Loop Trajectory and Open-Loop Trajectory Predicted by MPC with Simulated Horizon

Here, the error between the open loop and closed loop trajectory is clear. Although somewhat symmetric, and seemingly both viable, the open loop takes a suboptimal path. The trajectory of the open loop is larger than that of the closed loop implying a greater cost function and thus a suboptimal trajectory. When finding the prediction error between these two models, the result is large, around 100. This is sound as the trajectories of the open-loop and closed-loop systems are almost opposite, and thus the error at some points becomes very large in the trajectory.

IV. Conclusion

Through this project, we were able to achieve some of the basic goals of the project. We were able to use Python along with the Pyomo library and the concepts of MPCs to achieve what was originally done in MATLAB

Simulink, which required a deep understanding of both how the concept of MPC works and how spaceship docking works. Through this project, we were able to design an MPC model that is used to predict the trajectories of a spaceship to help it dock to a non-rotating docking platform. The MPC problem was solved using the IPOPT solver.

There is future work that can be done on the project as well. Understanding and adding constraints to make this problem valid for docking on a rotating and moving platform is one of the tasks that can be undertaken in the future. Another thing could be, instead of having stationary debris, we could add constraints to have dynamically moving debris pieces that would need to be dodged by the spacecraft, and finally have some practical data to have more accurate upper and lower state limits to improve the accuracy of the model would be great. Even though the upper and lower limits have been assumed right now (as they were not clearly stated in the reference paper), having these numbers backed by some practical data would make our algorithm more reliable.

V. BIBLIOGRAPH

1) Jayamon, A. C., Chethipuzha, M., S, J., R, A., amp; S, S. (2020). Spacecraft Docking using model predictive control with STK-based visualization. 2020 IEEE 17th India Council International Conference (INDICON). https://doi.org/10.1109/indicon49873.2020.9342043
2) Park, Hyeongjun Zappulla II, Richard Zagaris, Costantinos Virgili-Llop, Josep Romano, Marcello. (2017). Nonlinear Model Predictive Control for Spacecraft Rendezvous and Docking with a Rotating Target.
3) IEEE. (2016). Double-integrator dynamics for multiagent systems with antagonistic ... IEEE. Retrieved December 16, 2022, from https://ieeexplore.ieee.org/document/8844271/