AIAA SPACE Forum
13 - 16 September 2016, Long Beach, California
AIAA/AAS Astrodynamics Specialist Conference

10.2514/6.2016-5273

Analysis and Experimentation of Model Predictive Control for Spacecraft Rendezvous and Proximity Operations with Multiple Obstacle Avoidance

Hyeongjun Park * Costantinos Zagaris † Josep Virgili-Llop ‡ Richard Zappulla §
Ilya Kolmanovsky ¶ and Marcello Romano ‖

In this paper, Model Predictive Control (MPC) approaches are applied to multiple obstacle avoidance maneuvers for spacecraft rendezvous and docking. For safe obstacle avoidance, keep-out constraints are introduced by bounding ellipsoids around obstacles. In a linear quadratic MPC (LQ-MPC) framework, the rotating hyperplane method is used to convexify the obstacle avoidance constraints. A new method using two hyperplanes for convexification of the constraints is also proposed to improve performance of the LQ-MPC approach. A nonlinear MPC (NMPC) approach that deals with the nonlinear obstacle constraints directly is also applied to solve the spacecraft proximity maneuvering problems by using the nonlinear programming solver IPOPT (Interior Point OPTimizer). Real-time implementation of the MPC solutions is analyzed and compared on a physical test bed using several test cases. Numerical simulations and experiments demonstrate the obstacle avoidance as well as real-time operation capabilities of the considered control approaches.

Nomenclature

Abbreviation

MPC	Model Predictive Control
LQ-MPC	Linear Quadratic Model Predictive Control
NMPC	Nonlinear Model Predictive Control
FSS	Floating Spacecraft Simulator
RPO	Rendezvous and Proximity Operations
NPS	Naval Postgraduate School
QP	Quadratic Programming
IPOPT	Interior Point OPTimizer
RH	Rotating Hyperplane
DH	Dual Hyperplane

Symbol

I_z	Moment of the floating spacecraft simulator
m	Mass of the floating spacecraft simulator, kg
F_x, F_y	Control forces in the x and y direction, N
F_{max}	Maximum control force, N
τ	Control torque, Nm
$\mathbf{x}$	State vector
$\mathbf{u}$	Control vector

*Postdoctoral Research Associate, Mechanical and Aerospace Engineering Department, Naval Postgraduate School (NPS), Monterey, CA, AIAA Member.
†PhD Student, Mechanical and Aerospace Engineering Department, NPS, Monterey, CA, AIAA Student Member.
‡Postdoctoral Research Associate, Mechanical and Aerospace Engineering Department, NPS, Monterey, CA.
§PhD Candidate, Mechanical and Aerospace Engineering Department, NPS, Monterey, CA, AIAA Student Member.
¶Professor, Department of Aerospace Engineering, University of Michigan, Ann Arbor, MI, AIAA Member.
‖Associate Professor, Mechanical and Aerospace Engineering Department, NPS, Monterey, CA, AIAA Associate Fellow.

N	Length of the prediction horizon
$\mathbf{p}$	Position vector of the chaser spacecraft
$\mathbf{p}_{c0}$	Initial position of the chaser spacecraft
$\mathbf{r}_{obs,j}$	Position of j^{th} obstacle
J_1	Control effort of the Chaser

I. Introduction

Spacecraft rendezvous and proximity operations (RPO) are required for a broad range of space missions, such as, docking, berthing, servicing, inspection and active debris removal. Over the past decades, several space missions involving RPO have been successfully completed with varying degrees of spacecraft autonomy.[1–6] There is a growing interest in achieving autonomous real-time collision-free RPO due to increasing number of debris around the Earth[7] and demands to conduct operations in the vicinity of other spacecraft vehicles. For instance, future missions may approach resident space objects with complex shapes, e.g., large appendages, or navigate around clusters of multiple space objects. These missions impose multiple geometrically complex obstacle avoidance constraints requiring a high level of guidance and control flexibility and autonomy. While achieving a collision-free RPO is certainly required, minimizing the propellant consumption during these missions is also an important factor for spacecraft operations.

Various approaches capable of handling obstacle avoidance along with various other constraints, e.g., maneuver requirements and actuation limits, have been proposed for real-time spacecraft guidance and control. Methods presented in literature are based on artificial potential functions (APF),[8–10] LQR/APF,[11,12] Gauss pseudo-spectral approach,[13] inverse dynamics,[14–16] and mixed-integer linear programs (MILP).[17] A survey of RPO guidance and control algorithms suitable for on-board implementation is also available.[18]

As an effective feedback control approach, model predictive control (MPC) has also received significant attention. Variants of the MPC framework[19–22] have been proposed for the spacecraft rendezvous and docking control problem. For instance, Richards and How[23] proposed an MPC strategy using the solution of an MILP.[17] Another approach to MPC design has been also proposed with a variable horizon wherein a constant number of moves is initially used and then reduced.[24]

This paper focuses on the analysis and experimentation with autonomous guidance and control algorithms using two different MPC approaches for RPO maneuvers with multiple obstacle avoidance. One approach is based on utilizing linear quadratic MPC (LQ-MPC)[25] coupled with dynamically reconfigurable linear constraints. An LQ-MPC approach has been developed for spacecraft RPO in the presence of a single obstacle.[20,21,26] In this previously developed approach, the inherently non-convex exclusion constraint is replaced with a convex constraint using a rotating hyperplane on the exclusive area encircling the obstacle. This LQ-MPC problem formulation is feasible for real-time implementation since the LQ-MPC problem reduces to a quadratic programming (QP) problem for which effective numerical solvers exist. This approach has been experimentally implemented and validated on various robotic test beds.[16,27] However, this approach tends to over-constrain the problem since the LQ-MPC uses an approximation of the constraint. Additionally, it is questionable how to deal with multiple obstacles, specifically, when obstacles are located in parallel to the target spacecraft, that is, two or more obstacles affect the trajectory of a chaser spacecraft at the same time. In this paper, this previously developed LQ-MPC formulation is extended for a multiple obstacle case. In addition, a new method for dealing with the inherently non-convex constraints imposed by the obstacles is developed and experimentally evaluated.

On the other hand, there have been attempts to directly handle nonlinear constraints[28,29] in a Nonlinear MPC (NMPC) framework. For example, nonlinear constraints on thrust magnitude can be directly dealt with by using a fast algorithm.[28] In this approach it is not required to resort to saturation or polyhedral norm approximations. Jewison et al.[29] proposed an NMPC algorithm which is able to deal with multiple nonlinear (non-convex) obstacle constraints by exploiting ellipsoids as exclusion zones that completely surround the considered obstacles. NMPC can yield lower control effort, i.e., propellant use, and reduced maneuver times since it does not require approximation of the constraints. However, computing the optimal control input in real time is a challenging task. To successfully achieve obstacle avoidance, a long prediction horizon may be required and, consequently, this leads to computational difficulties for real-time implementation. The inability to solve the NMPC problem in real time can result in loss of stability, degraded performance, and violation of constraints.[30] Hence, computationally efficient NMPC algorithms are desired in order to obtain

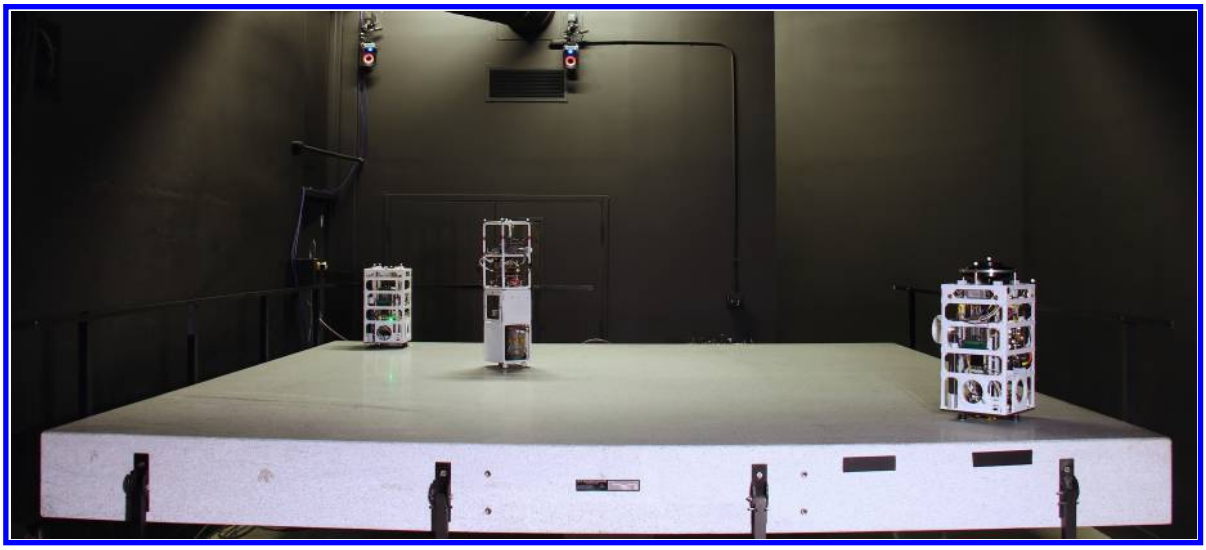

Figure 1. Floating sapacecraft simulator test bed with the overhead Vicon motion capture cameras, floating spacecraft simulators, and granite monolith in the Spacecraft Robotics Laboratory at the Naval Postgraduate School.

the optimal control solutions online with negligible computational delay.

Fast numerical algorithms have been proposed for solving NMPC problems. Cannon[31] provided an overview of efficient numerical methods and algorithms for NMPC. The nonlinear real-time iteration scheme[32] achieves fast computation using one iteration of root finding in each sampling period. To avoid the potential issues associated with the early termination approaches, the advanced step algorithm[33] relies on a complete Newton-type interior point procedure run to convergence. The integrated perturbation analysis and sequential quadratic programming (IPA-SQP) algorithm[34] has been also proposed to solve NMPC, which combines solution updates derived using perturbation analysis (PA) and sequential quadratic programming (SQP) and improves computational efficiency. In application to ship power management, the IPA-SQP approach has demonstrated real-time optimization, constraint enforcement, and fast reference following capabilities.[30]

In this paper, we report the results of applying the LQ-MPC and NMPC approaches to spacecraft RPO with multiple obstacle avoidance constraints. A new method using two hyperplanes is proposed to deal with obstacle avoidance constraints in the LQ-MPC framework more effectively. The performance of the developed MPC algorithms is analyzed using a MATLAB/Simulink-based simulator. Finally, the MPC controllers are implemented on a physical test bed in real time. For several test cases, the performance of the control algorithms and the real-time control capability with limited hardware resources are experimentally validated.

The paper is organized as follows. In Section II, the physical test bed is introduced. Then, the model and constraints are described. The general MPC problem is formulated with obstacle avoidance constraints. The rotating hyperplane method and a new method approximating the obstacle avoidance constraints are described in Section III. In Section IV, test cases of the proposed MPC controller are discussed. The simulation results are reported and analyzed. In Section V, the experimental results of the MPC controllers applied to the spacecraft test bed are presented, analyzed, and compared. Finally, concluding remarks are made at the end of the paper in Section VI.

II. System Description and Model

A. System Description

The developed MPC controllers are experimentally evaluated on the Naval Postgraduate School Floating Spacecraft Simulator Test Bed (NPS-FSSTB) . The test bed consists of floating spacecraft simulators (FSS), a polished granite table, a Vicon motion capture system, and a ground station computer. An overview of the NPS-FSSTB is shown in Fig. 1.

The floating surface is a 4-by-4 meter, 15 ton granite monolith with a planar accuracy of ± 0.0127 mm and

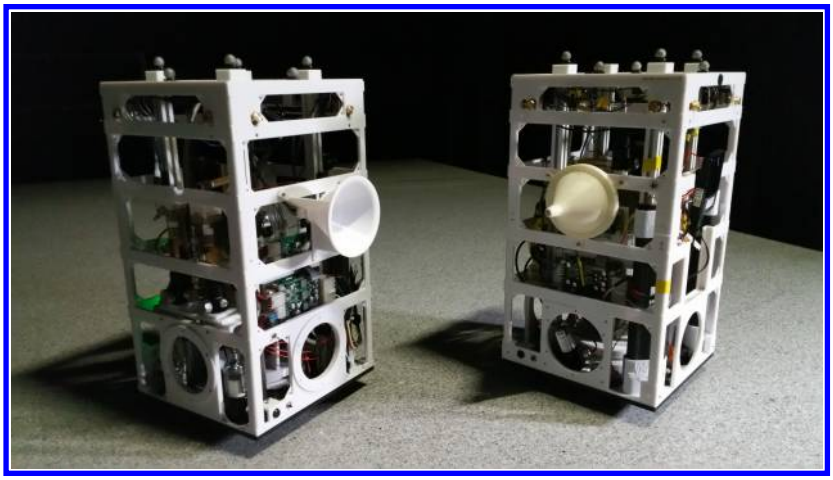

Figure 2. The Deputy (left) and Chaser (right) FSS atop the granite monolith.

a horizontal leveling accuracy of less than 0.01 deg. The FSS[10] are approximately 10 kg vehicles that float over the granite monolith surface via three flat air bearings. The quasi-frictionless environment with the low residual acceleration of the FSS emulates the environment encountered during close proximity operations in space. The FSS is propelled via eight cold gas thrusters fed by compressed air from an onboard tank. Each on-off thruster generates around 0.16 N of thrust, and when operated in a coordinated manner, the thrusters can provide the requested forces and torques. The FSS is able to perform real-time computation using the onboard computer. A power system gives the FSS complete autonomy.

The focus of this paper is on the guidance and control. The relative navigation problem is solved by using the Vicon motion capture system. This system, composed of ten overhead cameras that track reflectors mounted on the FSS, provides accurate position and orientation data. This data is processed on an external computer and streamed to the FSS using a Wi-Fi connection. It is then augmented by an onboard fiber-optic gyroscope, which provides angular velocity measurements. A discrete Kalman filter processes the data and provides a full state estimate. Detailed descriptions of the NPS-FSSTB and examples of its use for other spacecraft RPO research can be found elsewhere.[10, 16, 35, 36]

B. Model and Constraints

The model of the FSS is expressed by the following equations of motion,

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

where m denotes the mass of the FSS, I_z is the moment of inertia about the vertical axis, and $F_x,\ F_y,\ \tau$ are the control forces and torque, respectively. In this paper, we only consider control of the translational states. The attitude control is achieved through a fuel-optimal, bang-off-bang, slew combined with a proportional-derivative (PD) controller to maintain attitude pointing.

The state-space representation of the FSS translational model is written as

$$\dot{\mathbf{x}} = A\mathbf{x} + B\mathbf{u}, \tag{1}$$

where $\mathbf{x} = [x,\ y,\ \dot{x},\ \dot{y}]^T$ denotes the state vector, $\mathbf{u} = [F_x,\ F_y]^T$ denotes the control vector, and A and B are the corresponding state and control matrices, respectively. For the development of the MPC controllers, the discrete-time model is obtained as

$$\mathbf{x}(k+1) = A_d\mathbf{x}(k) + B_d\mathbf{u}(k), \tag{2}$$

where $\mathbf{x}(k) \in \mathbb{R}^4$ and $\mathbf{u}(k) \in \mathbb{R}^2$ denote the state and input vectors at the sampling instant $k \in \mathbb{Z}_{0+}$, respectively. The matrices A_d and B_d are the discrete state and control matrices, which can be derived from the continuous dynamics in Eq. (1).

The objective of the real-time guidance and control algorithms developed in this paper is to achieve autonomous docking between a deputy (stationary) spacecraft and a chaser (controlled) spacecraft (see Fig. 2) while avoiding multiple obstacles or debris on the spacecraft rendezvous path. We make the following assumptions that are representative of the physical system in the NPS-FSSTB.

1. Each thruster generates maximum thrust force of 0.15 N.
2. An entry cone constraint is enforced at the docking port of the deputy spacecraft to ensure safe docking.
3. For safe obstacle avoidance, an exclusion zone constraint is introduced by defining an exclusion zone in a form of the ellipsoid around an obstacle.[21,29] In simulations and experiments, a circle is used as the keep-out zone constraint for simplicity.

C. MPC Problem Formulation

Both the LQ-MPC and NMPC controllers are developed to address the MPC problem formulated as follows:
Minimize:

$$\begin{aligned} J &= (\mathbf{x}(N)-\mathbf{x}_t)^T P(\mathbf{x}(N)-\mathbf{x}_t) \\ &\quad + \sum_{i=0}^{N-1} (\mathbf{x}(k+i)-\mathbf{x}_t)^T Q(\mathbf{x}(k+i)-\mathbf{x}_t) + \mathbf{u}(k+i)^T R\mathbf{u}(k+i), \end{aligned} \tag{3a}$$

subject to:

$$\mathbf{x}(k+1) = A_d\mathbf{x}(k) + B_d\mathbf{u}(k), \tag{3b}$$
$$|u_1(k)| \leq u_{max}, \tag{3c}$$
$$|u_2(k)| \leq u_{max}, \tag{3d}$$
$$-\hat{n}_{c1}\cdot\mathbf{p}(k) \leq -\hat{n}_{c1}\cdot\mathbf{p}_{dock}, \tag{3e}$$
$$\hat{n}_{c2}\cdot\mathbf{p}(k) \leq \hat{n}_{c2}\cdot\mathbf{p}_{dock}, \tag{3f}$$
$$1 \leq (\mathbf{p}(k)-\mathbf{r}_{obs,j})^T S_j(\mathbf{p}(k)-\mathbf{r}_{obs,j}),\ j=1,\ldots,n, \tag{3g}$$

where N is the length of the prediction horizon, and $\mathbf{x}_t$ is the targeted final condition. As mentioned before, only the translational motion is included in the MPC formulation. The matrices P, Q, and R in Eq. (3a) define the cost function weights on the final state, intermediate states, and control variables, respectively. The matrix P is chosen as the positive-definite solution of the discrete Riccati equation for the unconstrained infinite horizon variant of the MPC problem to ensure local stability.[37,38] Eq. (3b) defines the equality constraint enforcing the dynamics of the system. Eq. (3c) and (3d) enforce constraints on the control variables. The approach cone constraint is linearized by constructing two hyperplanes that define the edges of the cone, and intersect at the target docking point. When these constraints are activated, the FSS will be forced to stay within the two hyperplanes until docking is achieved. Eq. (3e) and (3f) enforce hyperplane constraints for the entry cone, where $\hat{n}_{()}$ defines the normal vector of the hyperplane, $\mathbf{p}_{dock}$ defines a point on the hyperplane, $\mathbf{p}(k)$ is the position vector at the sampling instant k, i.e., $\mathbf{p}(k) = [x(k),\ y(k)]^T$. The obstacle avoidance constraints for n obstacles are expressed in Eq. (3g) where $\mathbf{r}_{obs,j}$ is the position vector of the obstacle j and S_j is the shape matrix of the ellipsoid j. Note that the obstacle avoidance constraints are nonlinear and non-convex. Whereas NMPC is able to deal with these constraints directly, the LQ-MPC method requires a convex, linear approximation. Two approximation methods for the obstacle avoidance constraints are introduced and discussed in the following section.

III. Obstacle Avoidance Methods

A. Linear-Quadratic MPC (LQ-MPC)

To utilize the LQ-MPC framework, the obstacle avoidance constraints are linearized. The MPC problem then becomes a constrained optimization problem, wherein a quadratic cost function is minimized subject to linear inequality constraints. This problem formulation results in a convex, QP problem that can be solved using readily available solvers.[39] The rotating hyperplane method has been used in the literature for

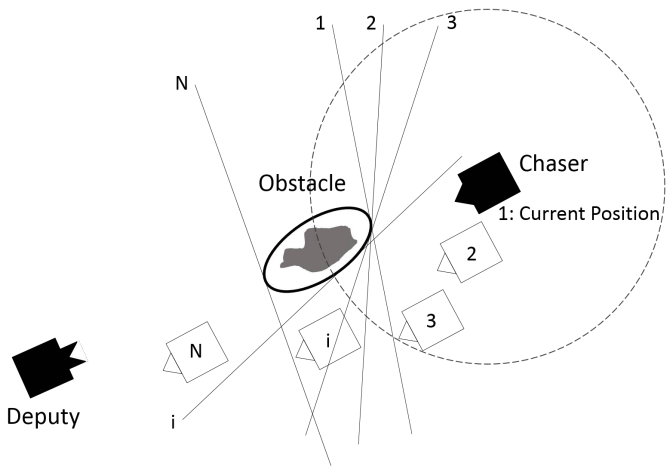

Figure 3. Depiction of the Rotating Hyperplane method for avoidance of one obstacle.

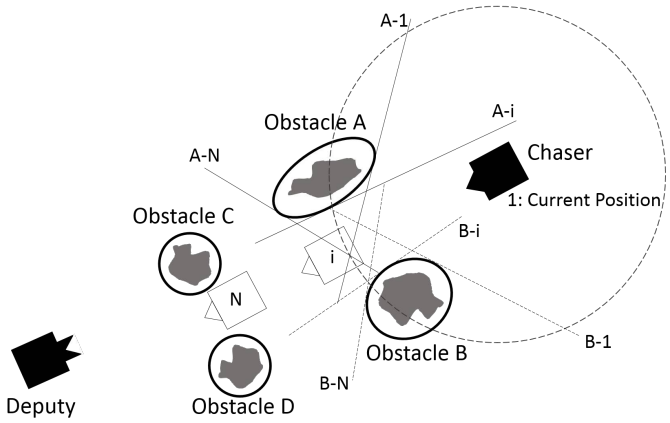

Figure 4. Depiction of the Rotating Hyperplane method for avoidance of multiple obstacles.

obstacle avoidance maneuvers in the context of spacecraft relative motion,[20,21,26] and has been successfully demonstrated, through simulation and experiments, in cases with one obstacle.[16] A brief overview of the method is provided, and an extension for multiple obstacles is introduced in this section. A new method considering two non-rotating hyperplanes is introduced to achieve avoidance of multiple obstacles and to improve performance.

1. Rotating Hyperplane (RH) Approach

For the case with one obstacle, given the current position, a hyperplane is here placed on the keep-out boundary and rotated at a constant rate throughout the horizon, in order to guide the spacecraft around the obstacle. Hence, the constraint is dynamically reconfigured to force the spacecraft to stay on one side of the hyperplane. The constraint is activated when the spacecraft is within a user-defined distance (range of influence) from the obstacle. When the constraint is activated, and assuming the horizon of length is N, this method requires the construction of N hyperplanes for every iteration. The constraint is deactivated after the spacecraft passes the obstacle so that the constraint does not interfere with the spacecraft motion. Figure 3 depicts the rotating hyperplane scenario for one obstacle and the chaser spacecraft which is to approach the deputy spacecraft position.

This method can be similarly extended to handle multiple obstacles. If an obstacle or obstacles are within the range, the avoidance constraints corresponding to the obstacles are activated. Then, a dynamically reconfigurable, rotating hyperplane is computed for each obstacle. Figure 4 illustrates the rotating hyperplane

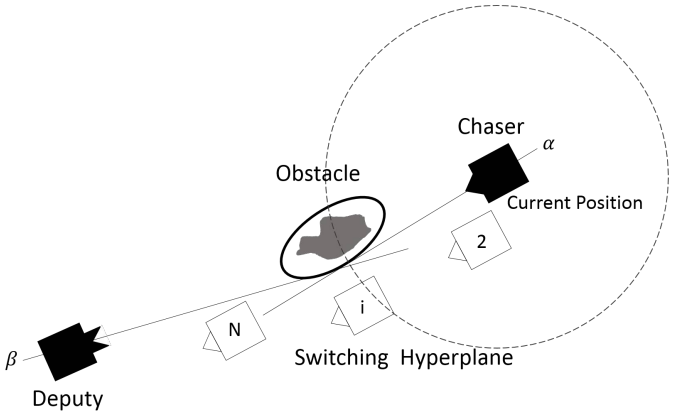

Figure 5. Depiction of Dual Hyperplane method for avoidance of one obstacle.

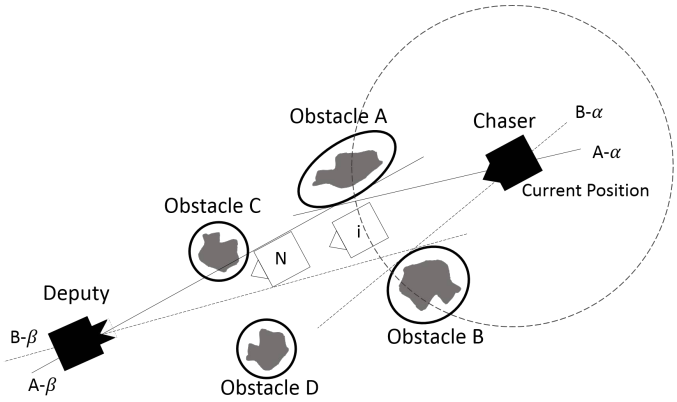

Figure 6. Depiction of Dual Hyperplane method for avoidance of multiple obstacles.

scenario for multiple obstacles. Avoidance constraints of Obstacle A and Obstacle B are activated at the current position of the chaser spacecraft. The constraint corresponding to Obstacle A is the rotating hyperplane from $A-1,\ \ldots,\ A-i,\ \ldots,\ A-N$. Similarly, $B-1,\ \ldots,\ B-i,\ \ldots, B-N$ represent the rotating hyperplane on Obstacle B.

The RH method has shown promising results with one obstacle, but may be computationally expensive for a problem with multiple obstacles or a large horizon. The dynamically reconfigured obstacle avoidance constraints may be infeasible or lead to ill-conditioned problem numerically depending on the locations and shapes of obstacles thereby increasing the complexity of the optimization problem. In the RH method, the rotation rates of hyperplanes also become tuning parameters, as they change the resulting trajectory.

2. Dual Hyperplane (DH) Approach

A new approach using two fixed hyperplanes is proposed to handle obstacle avoidance constraints and to improve performance of LQ-MPC. Figure 5 illustrates the dual hyperplane method with one obstacle. As with the RH method, the constraint becomes activated when the chaser is within the range of influence of the obstacle. First, given the current position, two candidate points are found that are considered as points of closest approach. This is accomplished by considering the intersection of two circles - a circle of radius equal to the current distance to obstacle, and a circle defining the keep-out-zone. One of the two candidate points is selected based on how close it is to the current position, and how close it is to the desired target approach axis. Fig. 5 shows that hyperplane α is constructed based on the candidate point selection. Similarly, another hyperplane, β, is constructed along the vector from the candidate point to the deputy spacecraft.

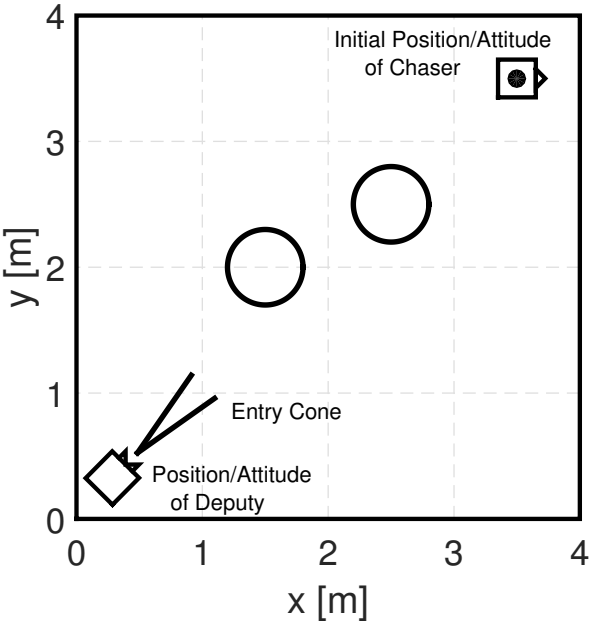

Figure 7. Schematic of the obstacle avoidance problem set-up.

In the MPC problem, the first few steps of the horizon enforce the first hyperplane, guiding the vehicle towards the edge of the obstacle along the hyperplane α. Then the second hyperplane constraint β is enforced to guide the vehicle towards the target. In this method, the obstacle avoidance constraint is deactivated after the chaser passes the obstacle so that the spacecraft motion is not unnecessarily influenced by these constraints. The extension of this method to a multiple obstacle avoidance scenario is fairly straight-forward. Figure 6 depicts the DH scenario for multiple obstacles. For two activated obstacles (Obstacle A and B), four hyperplanes $A-\alpha$, $A-\beta$, $B-\alpha$, and $B-\beta$ are constructed to avoid both obstacles.

This new method provides simpler dynamically reconfigured constraints compared with the RH method and thus reduces the complexity of the algorithm. As with the rotating hyperplane, a tuning parameter is inherent in this method which determines at which point within the horizon the switch is made to the second hyperplane. Switching too late may increase the control effort, while switching too early may result in an ill-conditioned problem. There are two possible approaches to implement this in the LQ-MPC framework. One is using a fixed switching timing, e.g., i-th step within the prediction horizon. The other approach is to determine an appropriate switching time using the predicted trajectory from the previous iteration. Although this approach would require an initial guess, the final implementation may be more robust that the fixed switching point approach. In this paper, we use the fixed switching point approach, and the dynamic switching approach is left as a topic for future research. For the simulations and experiments presented here the switch was implemented at $\frac{1}{4}N$.

B. Nonlinear MPC (NMPC)

In the NMPC framework, the nonlinear ellipsoid obstacle avoidance constraints can be handled directly so that the performance in terms of time-to-dock or fuel consumption may be improved compared to the approximated convex obstacle constraints. However, the ellipsoidal constraints define a non-convex feasible region which may cause optimization difficulties and loss of guarantees on convergence.[29,40] To solve the nonlinear optimization problem in real time, the IPOPT software package[41] is utilized.

IV. Numerical Simulation Results

The implementation of the controllers based on LQ-MPC and NMPC has been tested in simulation, using a MATLAB/Simulink based numerical simulator that represents the FSS plant and emulates the on-board

Table 1. Parameters in the test cases

Parameter	Value
Initial state	$\mathbf{p}_{c0} = (3.5,\ 3.5)$ m
Target location	$\mathbf{p}_d = (0.3,\ 0.3)$ m
Entry cone orientation	$\theta_c = 45$ deg.
Entry cone half-angle	$\theta_{hc} = 10$ deg.
Entry cone range	$r_{\text{cone}} = 0.75$ m
Obstacle keep out radius	$r_{\text{obs}} = 0.3$ m
Case	Obstacle location
A	$\mathbf{p}_{\text{obs}_1} = (2.4,\ 2.5)$ m, $\mathbf{p}_{\text{obs}_2} = (2.5,\ 1.5)$ m
B	$\mathbf{p}_{\text{obs}_1} = (2.0,\ 2.5)$ m, $\mathbf{p}_{\text{obs}_2} = (3.0,\ 2.5)$ m
C	$\mathbf{p}_{\text{obs}_1} = (2.5,\ 2.5)$ m, $\mathbf{p}_{\text{obs}_2} = (1.5,\ 2.0)$ m

sensors and actuators. The case study scenarios used in the numerical simulations and experiments are also provided in this section.

A. Test Scenarios

Three test cases were designed differing from the positions of obstacles in order to test and compare the guidance algorithms. In extending previous results for single obstacle avoidance,[16] each test case includes two obstacles placed such that the algorithms must actively avoid them. We assume that keep-out zones are circles with the radius of 0.3 m. Figure 7 shows the problem set-up with the keep-out zones placed around two obstacles. The initial position and attitude of the chaser and the target conditions were the same for all three cases, while the obstacle positions changed: the initial position of the chaser is (3.5, 3.5) m and the initial attitude angle is 0 deg. The deputy is located at (0.3, 0.3) m and its orientation is 45 deg. The half angle and length of the entry cone are 10 deg and 0.75 m. Table 1 summarizes the set-up for the cases.

In the simulations and experiments, we have used the same weighting matrices for the MPC cost functional chosen as:

$$Q = \begin{bmatrix} \mathbf{I}_2 & \mathbf{0}_{2\times 2} \\ \mathbf{0}_{2\times 2} & 3\times 10^3 \mathbf{I}_2 \end{bmatrix}, \quad R = 10^2 \mathbf{I}_2. \tag{4}$$

The prediction horizon of is set to $N = 30$, and the sampling period is set to 3 s to balance the controller execution time with the prediction horizon. Hence, the prediction window is 90 s. The maximum thrust force in the x and y direction is selected as $F_{max} = 0.15$ N. As a fuel-related metric, the control effort J_1 is defined using the ℓ_1-norm as

$$J_1 = \int_{t_0}^{t_f} \sum_{i=1}^{8} u_i dt, \tag{5}$$

where $u_i \in 0, 1$ is the on-off state of the thruster with a resolution of 0.01 s for each thruster in real implementation, t_0 is the initial time when the FSS guidance is enabled, and t_f is the final time when the docked conditions are met.[10,16]

B. Simulation Results

The simulation results for Case A are shown in Figure 8. The trajectory generated by NMPC is a fairly straight path, and the LQ-MPC/DH trajectory is similar. The LQ-MPC/RH trajectory, however, stays further away from the obstacles and has a more distinct S-shape. As a result, fuel consumption is larger for LQ-MPC/RH. Table 2 reports the simulation results on time-to-dock and control effort for all three algorithms. Note that the computation time is not used as a performance metric for simulation results because simulations were executed on different computers, and it would not be a fair comparison. The computation time is taken into account for the experiments. The NMPC and LQ-MPC/DH results show similar performance in terms of fuel consumption. The LQ-MPC/DH design achieves smaller control effort than NMPC while it requires longer time to finish the docking maneuver.

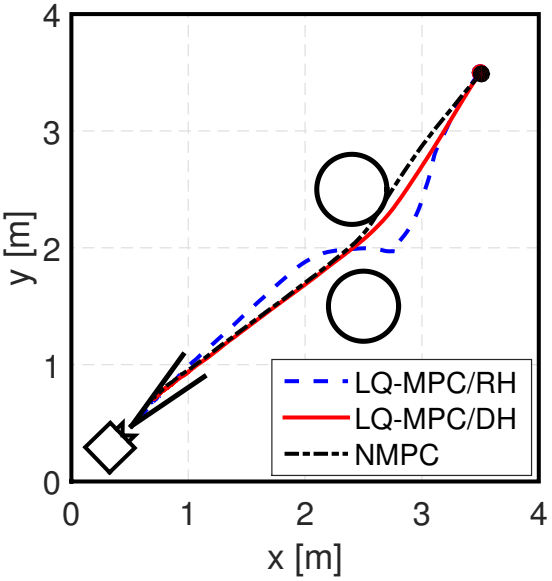

Figure 8. Simulated trajectory comparison for Case A.

Table 2. Simulation Results

Case	Metric	LQ-MPC/RH	LQ-MPC/DH	NMPC
A	Time-to-dock (s)	216.81	237.81	199.06
	Control effort (Ns)	6.87	3.04	3.12
B	Time-to-dock (s)	216.81	240.81	205.01
	Control effort (Ns)	7.93	3.17	2.82
C	Time-to-dock (s)	213.81	252.81	205.06
	Control effort (Ns)	7.22	4.27	3.52

Figure 9 shows the simulation results for Case B. As in Case A, the NMPC and LQ-MPC/DH trajectories are fairly similar straight paths, while the LQ-MPC/RH trajectory has a distinct S-shape, resulting in increased fuel consumption. These results indicate that the feasible regions created by the rotating hyperplanes at every step in the horizon are not well-matched with the velocity of the chaser spacecraft. Table 2 shows that in this case, NMPC is able to achieve docking in the shortest time with the best fuel consumption performance, with LQ-MPC/DH close behind it.

Case C is the most constraining case for the algorithms as the second obstacle is located behind the first, requiring the chaser to maneuver significantly in order to avoid it, i.e., there is no feasible path close to a straight line. Note that the first obstacle is placed on the diagonal, which corresponds to a line of symmetry at the beginning of the simulation, when the chaser is outside of the second obstacle's range of influence. With this set-up the chaser could avoid the first obstacle on either side, with the same control effort, but would only encounter the second obstacle if its trajectory is to the left side of the first obstacle. In order to test the most stressing case, we consider only trajectories passing between two obstacles by "forcing" the chaser to avoid the first obstacle on the left side.

Figure 10 shows the simulation results for Case C. The NMPC trajectory in this case "hugs" the two obstacles in order to avoid them and achieve docking with minimal control effort. The LQ-MPC/DH result shows a significant trajectory change when the second obstacle is encountered, which then converges to the NMPC trajectory. The LQ-MPC/RH solution, however, shows quite large trajectory deviations resulting in higher fuel consumption. Table 2 shows that in this case, NMPC is able to achieve docking in the shortest time with the best fuel consumption performance, while the LQ-MPC algorithms resulted in much higher fuel consumption.

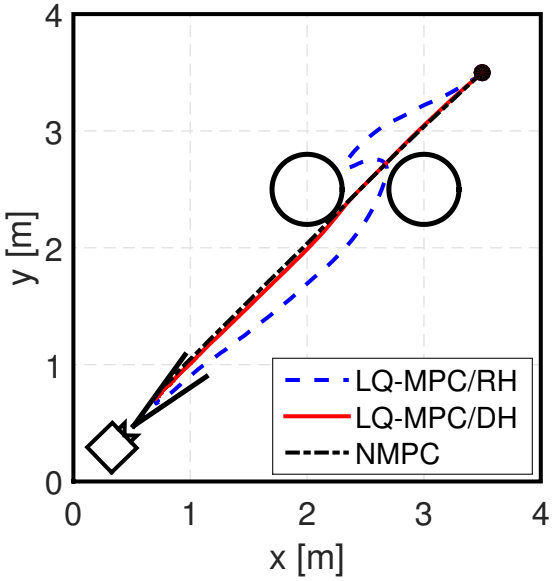

Figure 9. Simulated trajectory comparison for Case B.

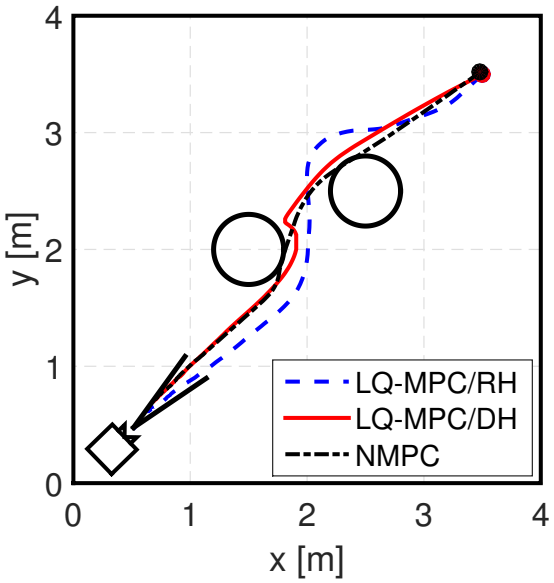

Figure 10. Simulated trajectory comparison for Case C.

V. Experimental Results on the NPS-FSSTB

The experiments were conducted on the NPS-FSSTB at the Spacecraft Robotics Laboratory. A detailed description of the system can be found in Section II-A and in the literature.[10,42] The chaser spacecraft Simulator has an onboard PC-104 form-factor computer based on an Intel Atom 1.6 GHz 32 bit processor with a 2 GB of RAM and a 8 GB solid-state drive. The onboard computer accommodates computing for guidance, navigation, and control (GNC) algorithms developed in a MATLAB/Simulink environment. The operating system is an RTAI-patched Ubuntu 14.04 Sever Edition which provides real-time execution capabilities. The developed MPC controllers are included in a MATLAB/Simulink model with the sensor and actuator blocks. Then, the model is cross-compiled for the FSS 32-bit architecture and later transferred

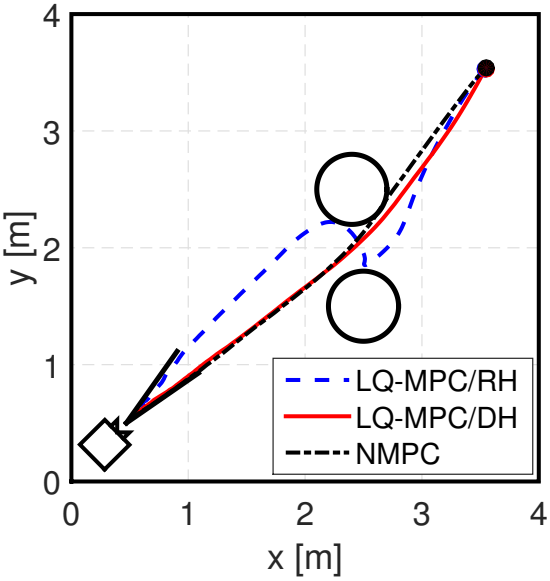

Figure 11. Experimental trajectory comparison for Case A.

to the FSS simulator via Wi-Fi connection.

The LQ-MPC methods use a MATLAB-based QP solver[43] implemented in an embedded MATLAB function. In order to avoid overruns in real-time operations the solver is terminated if a maximum number of iterations is exceeded. In this implementation, the maximum number of iterations is set to 170. For solving the NMPC problem, the IPOPT solver is wrapped as a Simulink S-function. In order to avoid overruns in real-time operations the IPOPT solver is terminated if a maximum number of iterations, or a maximum computation time is exceeded. In this implementation, the maximum number of iterations is set to 400, and maximum computation time is set to 2.7 seconds. Then, a C code of the hardware model including the solvers can be automatically generated by Simulink, and executed in the real-time operating system of the FSS.

Figure 11 shows the experiment results for Case A. The NMPC and LQ-MPC/DH trajectories are very similar to the simulated trajectories, following a fairly straight path in between the two obstacles. The LQ-MPC/RH trajectory, as in simulation, shows fairly large deviations, resulting in higher fuel consumption. After conducting at least 3 tests on each case using the aforementioned methods, Table 3 reports the median experimental results in terms of performance metrics. Time-to-dock results are well-correlated to the simulation results in Case A. Control effort for the experiments, however, is slightly larger for all the methods. For this experiment, NMPC exhibited the shortest docking time with the best fuel consumption, with LQ-MPC/DH close behind it. From a computational aspect, although the NMPC algorithm shows a significantly larger number of iterations to converge that the LQ-MPC methods, the average computation time is still lower. This result was surprising, since the LQ-MPC methods formulate a convex problem. However, the IPOPT has the advantage of being a mature solver, written in C, versus a MATLAB-based function. Therefore, the specific implementations of the solvers in these algorithms likely affect their computational performance. A consideration for future work could be to use IPOPT's convex solver option in the LQ-MPC methods and compare the algorithms' computational performance. Comparison between the LQ-MPC/RH and LQ-MPC/DH methods shows that the DH method has a slightly smaller average computation time. The RH method also reaches the maximum allowable number of iterations, while the DH method is able to find solutions within a relatively small number of iterations.

The experimental results for Case B are illustrated in Fig. 12. As in Case A, the experimental trajectories match the simulated results. NMPC and LQ-MPC/DH show fairly straight trajectories to the target, while LQ-MPC/RH again has the distinct S-shape. Table 3 shows that in this case, the LQ-MPC/DH method has the most efficient result on fuel consumption, but with a longer docking time than the NMPC result. As in Case A, NMPC shows a slight advantage in average computation time, although the number of iterations

Table 3. Experimental Results

Case	Metric	LQ-MPC/RH	LQ-MPC/DH	NMPC
A	Time-to-dock (s)	270.51	240.34	202.43
	Control Effort (Ns)	7.30	3.99	3.68
	Avg Computation Time (s)	0.29	0.22	0.13
	Max Computation Time (s)	2.02	0.36	0.34
	Avg Number of Iterations	17.94	9.55	23.19
	Max Number of Iterations	171	13	67
B	Time-to-dock (s)	353.40	276.47	244.41
	Control Effort (Ns)	8.24	4.51	4.70
	Avg Computation Time (s)	0.24	0.24	0.16
	Max Computation Time (s)	2.02	0.41	0.32
	Avg Number of Iterations	12.28	11.10	28.90
	Max Number of Iterations	171	15	60
C	Time-to-dock (s)	256.88	274.56	Violation of
	Control Effort (Ns)	7.87	5.17	an obstacle constraint
	Avg Computation Time (s)	0.33	0.23	0.79
	Max Computation Time (s)	2.85	0.43	2.68
	Avg Number of Iterations	20.98	10.43	34.35
	Max Number of Iterations	171	16	347

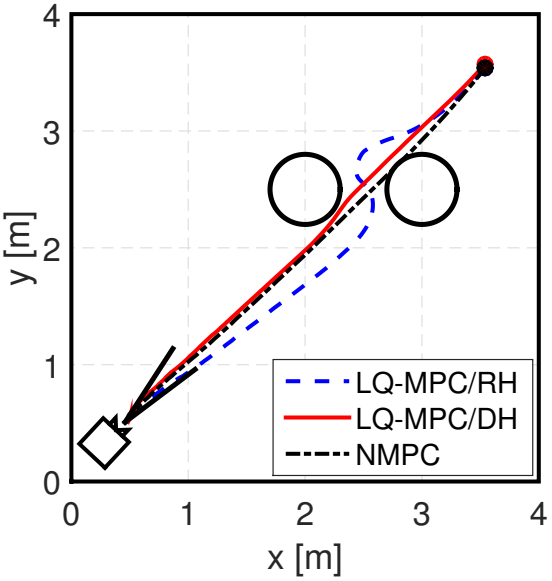

Figure 12. Experimental trajectory comparison for Case B.

required is significantly larger.

Figure 13 shows the experimental for Case C, the most stressing case. In this case, the LQ-MPC methods successfully achieve docking without constraint violations; however, NMPC violated the constraint on the second obstacle. In repeated experimental trials with NMPC, the chaser invades the keep-out zone as shown in Figure 13. Due to the limited thrust capability, the optimization problem with the non-convex obstacle constraint and high approaching velocity of the chaser becomes a difficult problem to solve in limited time or iterations. In this scenario, the solver exceeds the maximum allowable computation time, resulting in

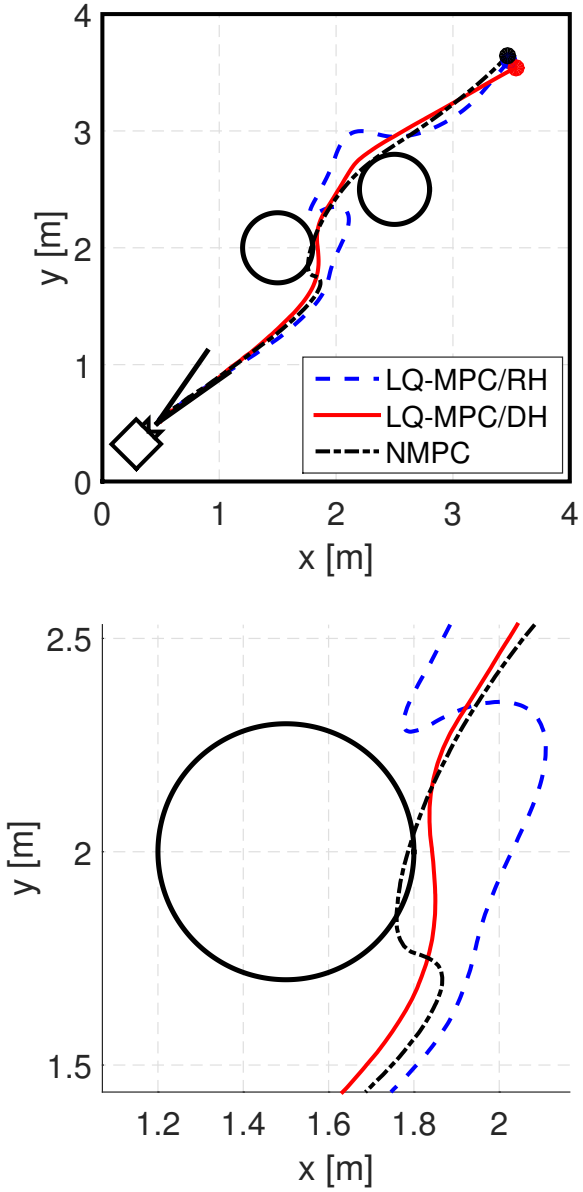

Figure 13. Experimental trajectory comparison for Case C: trajectories (top) and zoomed-in trajectories (bottom).

solutions that do not satisfy constraints. The maximum computation time is exceeded at two sampling time instants and the chaser invades the keep-out zone as shown in Fig. 13. The numbers of iterations when the maximum computation time is exceeded are 347 and 332, respectively.

Table 3 shows the computational effort comparison between the tree algorithms. NMPC, in this case, shows the largest average computation time due to the constraint violation. The average computation time and number of iterations for the LQ-MPC methods in this case is similar to the results of the previous cases.

The experimental results show that the LQ-MPC methods provides a certain level of robustness, since all three cases were able to be solved without re-tuning the controller. However, the LQ-MPC/RH method encountered a time where the maximum allowable number of iterations was reached, whereas the LQ-MPC/DH method solved with a relatively small number of iterations. The NMPC method, as shown in Case C, was not always successful. This is a direct result of the convexity characteristics exploited in the LQ-MPC methods. The NMPC method could, however, be re-tuned so that a successful docking can be

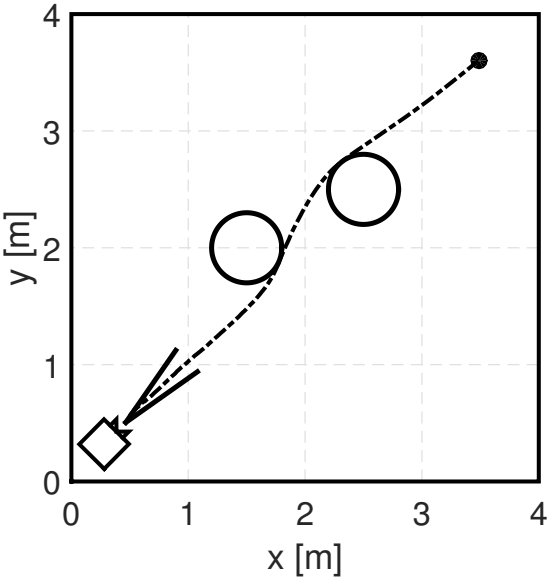

Figure 14. Experimental trajectory of NMPC with the modified state weighting matrix for Case C.

achieved for Case C. For example, modifying the state weighting matrix would result in a successful docking. Figure 14 shows the experimental trajectory of NMPC with a modified state weighting matrix $\bar{\mathbf{Q}}$,

$$\bar{\mathbf{Q}} = \begin{bmatrix} \mathbf{I}_2 & \mathbf{0}_{2\times 2} \\ \mathbf{0}_{2\times 2} & 5\times 10^3 \mathbf{I}_2 \end{bmatrix}. \tag{6}$$

With larger weight on the velocity components, the chaser moves slower and is able to avoid the second obstacle. As a result, NMPC could be well tuned for a specific scenario but the same set-up may not work in different conditions.

VI. Conclusions

In this paper, we have considered spacecraft rendezvous and docking maneuvering with multiple obstacle avoidance. LQ-MPC and NMPC were utilized to enforce the obstacle avoidance constraints, thrust magnitude constraints, and the entry cone constraints. In the LQ-MPC framework, the non-convex obstacle constraints was approximated to convex constraints, and the optimization problem was solved by a QP algorithm. The dynamically reconfigurable linear constraints have been employed to accomplish this. The rotating hyperplane method has been implemented as an approach of the constraint linearization. A new method, double hyperplane method, using two hyperplane has been proposed to improve performance and to reduce numbers of tunable parameters in the rotating hyperplane method. An NMPC controller dealing with the nonlinear obstacle avoidance constraints directly using an efficient solver, IPOPT.

The developed MPC controllers have been analyzed and tested on the simulation model and the NPS-FSSTB. The dual hyperplane method achieved performance improvement and is applicable for a general case whereas the rotating hyperplane method required to tune rotation rates of hyperplanes in simulations and experiments. The NMPC controller with the same MPC parameter setting has accomplished good performance in nonreal-time simulations and in experiments overall; however, non-converged solutions due to computational delays made violation of the obstacle avoidance constraints in a test case. This demonstrated that the well-defined convexification of the obstacle avoidance constraints enables more robust implementation of the LQ-MPC without significant computational delays in real time. As a future work, the extensions to three dimensional spacecraft of the dual hyperplane method and to docking with a rotating target will be pursued in the NPS-FSSTB.

References

[1]Davis, T. M. and Melanson, D., "XSS-10 microsatellite flight demonstration program results," *Defense and Security*, International Society for Optics and Photonics, 2004, pp. 16–25.

[2]Zimpfer, D., Kachmar, P., and Tuohy, S., "Autonomous rendezvous, capture and in-space assembly: past, present and future," *1st Space Exploration Conference: Continuing the Voyage of Discovery*, Vol. 1, Orlando, Florida, USA, 2005, pp. 234–245.

[3]Howard, R. T. and Bryan, T. C., "DART AVGS flight results," *Defense and Security Symposium*, International Society for Optics and Photonics, 2007, pp. 65550L–65550L.

[4]Henshaw, C. G., "The DARPA Phoenix Spacecraft Servicing Program: Overview and Plans for Risk Reduction," *Proceedings of 12th International Symposium on Artificial Intelligence, Robotics and Automation in Space (i-SAIRAS 2014),(Montreal, Canada)*, 2014.

[5]De Pasquale, E., "ATV Jules Verne: a Step by Step Approach for In-Orbit Demonstration of New Rendezvous Technologies," *Proc. SpaceOps Conference, Stockholm*, 2012.

[6]Xie, Y., Hu, J., Wang, M., Hu, H., and Zhang, H., "Accurate and stable control of shenzhou spacecraft in rendezvous and docking," *IFAC Proceedings Volumes*, Vol. 46, No. 19, 2013, pp. 524–528.

[7]Klinkrad, H., *Space debris*, Wiley Online Library, 2010.

[8]Lopez, I. and McInnes, C. R., "Autonomous rendezvous using artificial potential function guidance," *Journal of Guidance, Control, and Dynamics*, Vol. 18, No. 2, 1995, pp. 237–241.

[9]Muñoz, J. D. and Fitz-Coy, N. G., "Rapid Path-Planning Options for Autonomous Proximity Operations of Spacecraft," *Proceedings of the AIAA/AAS Astrodynamics Specialist Conference, Toronto, Ortario Canada*, 2010.

[10]Richard Zappulla, I., Park, H., Virgili-Llop, J., and Romano, M., "Experiments on autonomous spacecraft rendezvous and docking using an adaptive artificial potential field approach," *26th AAS/AIAA Space Flight Mechanics Meeting, Napa, CA*, 2016, pp. 14–18.

[11]McCamish, S. B., Romano, M., Nolet, S., Edwards, C. M., and Miller, D. W., "Flight testing of multiple-spacecraft control on SPHERES during close-proximity operations," *Journal of Spacecraft and Rockets*, Vol. 46, No. 6, 2009, pp. 1202–1213.

[12]Bevilacqua, R., Lehmann, T., and Romano, M., "Development and experimentation of LQR/APF guidance and control for autonomous proximity maneuvers of multiple spacecraft," *Acta Astronautica*, Vol. 68, No. 7, 2011, pp. 1260–1275.

[13]Boyarko, G., Yakimenko, O., and Romano, M., "Optimal rendezvous trajectories of a controlled spacecraft and a tumbling object," *Journal of Guidance, Control, and dynamics*, Vol. 34, No. 4, 2011, pp. 1239–1252.

[14]Ciarcia, M. and Romano, M., "Spacecraft proximity maneuver guidance based on inverse dynamic and sequential gradient-restoration algorithm," *Advances in the Astronautical Sciences*, Vol. 142, 2011.

[15]Ciarcià, M., Grompone, A., and Romano, M., "A near-optimal guidance for cooperative docking maneuvers," *Acta Astronautica*, Vol. 102, 2014, pp. 367–377.

[16]Virgili-Llop, J., Zagaris, C., Park, H., Zappulla II, R., and Romano, M., "Experimental evaluation of model predictive control and inverse dynamics control for spacecraft proximity and docking maneuvers," .

[17]Richards, A., Schouwenaars, T., How, J. P., and Feron, E., "Spacecraft trajectory planning with avoidance constraints using mixed-integer linear programming," *Journal of Guidance, Control, and Dynamics*, Vol. 25, No. 4, 2002, pp. 755–764.

[18]Zagaris, C., Baldwin, M., Jewison, C., and Petersen, C., "Survey of spacecraft rendezvous and proximity guidance algorithms for on-board implementation," *Proc. AAS/AIAA Space Flight Mechanics Meeting*, 2015.

[19]Hartley, E. N., Trodden, P. A., Richards, A. G., and Maciejowski, J. M., "Model predictive control system design and implementation for spacecraft rendezvous," *Control Engineering Practice*, Vol. 20, No. 7, 2012, pp. 695–713.

[20]Di Cairano, S., Park, H., and Kolmanovsky, I., "Model predictive control approach for guidance of spacecraft rendezvous and proximity maneuvering," *International Journal of Robust and Nonlinear Control*, Vol. 22, No. 12, 2012, pp. 1398–1427.

[21]Petersen, C., Jaunzemis, A., Baldwin, M., Holzinger, M., and Kolmanovsky, I., "Model Predictive Control and extended command governor for improving robustness of relative motion guidance and control," *Proc. AAS/AIAA Space Flight Mechanics Meeting*, 2014.

[22]Weiss, A., Baldwin, M., Erwin, R. S., and Kolmanovsky, I., "Model predictive control for spacecraft rendezvous and docking: Strategies for handling constraints and case studies," *IEEE Transactions on Control Systems Technology*, Vol. 23, No. 4, 2015, pp. 1638–1647.

[23]Richards, A. and How, J., "Performance evaluation of rendezvous using model predictive control," *AIAA Guidance, Navigation, and Control Conference and Exhibit*, 2003, p. 5507.

[24]Hartley, E. N. and Maciejowski, J. M., "Model predictive control for spacecraft rendezvous," 2010.

[25]Rawlings, J. B. and Mayne, D. Q., *Model predictive control: Theory and design*, Nob Hill Pub., 2009.

[26]Park, H., Di Cairano, S., and Kolmanovsky, I., "Model predictive control for spacecraft rendezvous and docking with a rotating/tumbling platform and for debris avoidance," *Proceedings of the 2011 American Control Conference*, IEEE, 2011, pp. 1922–1927.

[27]Goodyear, A., Petersen, C., Pierre, J., Zagaris, C., Baldwin, M., and Kolmanovsky, I., "Hardware implementation of Model Predictive Control for relative motion maneuvering," *2015 American Control Conference (ACC)*, IEEE, 2015, pp. 2311–2316.

[28]Park, H., Kolmanovsky, I., and Sun, J., "Model predictive control of spacecraft relative motion maneuvers using the IPA-SQP approach," *ASME 2013 Dynamic Systems and Control Conference*, American Society of Mechanical Engineers, 2013, pp. V001T02A001–V001T02A001.

[29]Jewison, C., Erwin, R. S., and Saenz-Otero, A., "Model predictive control with ellipsoid obstacle constraints for spacecraft rendezvous," *IFAC-PapersOnLine*, Vol. 48, No. 9, 2015, pp. 257–262.

[30]Park, H., Sun, J., Pekarek, S., Stone, P., Opila, D., Meyer, R., Kolmanovsky, I., and DeCarlo, R., "Real-time model predictive control for shipboard power management using the IPA-SQP approach," *IEEE Transactions on Control Systems Technology*, Vol. 23, No. 6, 2015, pp. 2129–2143.

[31]Cannon, M., "Efficient nonlinear model predictive control algorithms," *Annual Reviews in Control*, Vol. 28, No. 2, 2004, pp. 229–237.

[32]Houska, B., Ferreau, H. J., and Diehl, M., "An auto-generated real-time iteration algorithm for nonlinear MPC in the microsecond range," *Automatica*, Vol. 47, No. 10, 2011, pp. 2279–2285.

[33]Zavala, V. M. and Biegler, L. T., "The advanced-step NMPC controller: Optimality, stability and robustness," *Automatica*, Vol. 45, No. 1, 2009, pp. 86–93.

[34]Ghaemi, R., Sun, J., and Kolmanovsky, I. V., "An integrated perturbation analysis and sequential quadratic programming approach for model predictive control," *Automatica*, Vol. 45, No. 10, 2009, pp. 2412–2418.

[35]Romano, M., Friedman, D. A., and Shay, T. J., "Laboratory experimentation of autonomous spacecraft approach and docking to a collaborative target," *Journal of Spacecraft and Rockets*, Vol. 44, No. 1, 2007, pp. 164–173.

[36]Bevilacqua, R., Lehmann, T., and Romano, M., "Development and experimentation of LQR/APF guidance and control for autonomous proximity maneuvers of multiple spacecraft," *Acta Astronautica*, Vol. 68, No. 7, 2011, pp. 1260–1275.

[37]Maciejowski, J. M., *Predictive control: with constraints*, Pearson education, 2002.

[38]Mayne, D. Q., Rawlings, J. B., Rao, C. V., and Scokaert, P. O., "Constrained model predictive control: Stability and optimality," *Automatica*, Vol. 36, No. 6, 2000, pp. 789–814.

[39]Brand, M., Shilpiekandula, V., Yao, C., Bortoff, S. A., Nishiyama, T., Yoshikawa, S., and Iwasaki, T., "A parallel quadratic programming algorithm for model predictive control," *IFAC Proceedings Volumes*, Vol. 44, No. 1, 2011, pp. 1031–1039.

[40]Nocedal, J. and Wright, S., *Numerical optimization*, Springer Science & Business Media, 2006.

[41]Wächter, A. and Biegler, L. T., "On the implementation of an interior-point filter line-search algorithm for large-scale nonlinear programming," *Mathematical programming*, Vol. 106, No. 1, 2006, pp. 25–57.

[42]Zappulla, R., Virgili-Llop, J., Zagaris, C., Park, H., Sharp, A., and Romano, M., "Floating spacecraft simulator test bed for the experimental testing of autonomous guidance, navigation, and control of spacecraft proximity maneuvers and operations," *AIAA SPACE 2016, Long Beach, CA*, 2016, accepted.

[43]Currie, J., *Practical Applications of Industrial Optimization: From High-Speed Embedded Controllers to Large Discrete Utility Systems*, Ph.D. thesis, Auckland University of Technology, 2014.