EXPERIMENTAL EVALUATION OF MODEL PREDICTIVE CONTROL AND INVERSE DYNAMICS CONTROL FOR SPACECRAFT PROXIMITY AND DOCKING MANEUVERS

Josep Virgili-Llop, Costantinos Zagaris, Hyeongjun Park, Richard Zappulla II, and Marcello Romano

Naval Postgraduate School
Mechanical and Aerospace Engineering Department
1 University Circle, Monterey, CA 93940

ABSTRACT

An experimental campaign has been conducted to evaluate the performance of docking maneuvers of two different guidance and control algorithms based on Model Predictive Control (MPC) and on Inverse Dynamics in the Virtual Domain (IDVD) control. A Linear Quadratic approach with a Quadratic Programming solver has been used for the MPC and the nonlinear Interior Point OPTimizer solver is used for the IDVD approach. The docking scenario is constrained by the presence of a keep-out zone and an entry cone. The provided performance metrics for the conducted experiments and simulations include control effort, time to target and constraint handling. The experiments have been conducted on a planar dynamic simulator, using spacecraft simulators that float over a granite monolith, creating a reduced gravity and friction environment. In addition, a standard test framework for experimental evaluation of different guidance, navigation and control approaches for planar dynamic spacecraft simulators is proposed.

***Index Terms*—** spacecraft proximity and docking maneuvers, Model Predictive Control, inverse dynamics, testing

1. INTRODUCTION

Rendezvous and proximity operations (RPO) are essential to a wide range of space missions [1, 2]. Ground-based experimental evaluation of emerging guidance, navigation, and control (GNC) approaches may be useful to raise their technological readiness level, while helping to determine their performance and limitations on flight-equivalent hardware (i.e. sensors, actuators and computational systems) [2].

An experimental campaign to evaluate the performance of Model Predictive Control (MPC) and Inverse Dynamics in the Virtual Domain (IDVD) has been performed at the Naval Postgraduate School Floating Spacecraft Simulator (NPS-FSS) test bed. By utilizing this test bed, the focus of the research can be limited in scope to the guidance and control of the floating spacecraft simulators, as the navigation problem can be considered solved. The NPS-FSS motion capture system, augmented by onboard sensors, is used to provide accurate navigation data to the vehicles (including the location of the deputy and of the keep out zone).

A spacecraft docking problem has been selected for the experimental evaluation of the two different control approaches. A keep-out zone and an entry cone have been added to the scenario to evaluate the controllers constrain handling ability. A Linear Quadratic MPC (LQMPC) with a Quadratic Progamming (QP) solver and an IDVD with a Nonlinear Programming (NLP) solver were chosen for this comparison. While the LQMPC method requires linearization of the constraints in order to formulate the QP problem, the IDVD method can directly handle the nonlinear constraints, but requires a more complex and less robust NLP solver. In particular the open-source NLP Interior Point OPTimizer (IPOPT) is used. These two controllers are then executed in real-time on-board the autonomous Floating Spacecraft Simulators (FSS) in order to achieve autonomous docking.

To help standardize the experimental evaluation, a standard test framework is proposed. A qualitative set of test scenarios, designed to represent a wide set of common RPO scenarios (unconstrained and constrained, cooperative and uncooperative docking and proximity operations with or without obstacle avoidance) is proposed. The definition of the quantitative parameters (e.g. initial conditions or size of the keep-out zone) associated with each test scenario are let to be defined by each individual test bed. Standard metrics to compare different GNC approaches are also proposed. The goal is to define a standard framework that can be used to experimentally benchmark different guidance algorithms so that a meaningful comparison of different approaches can be made.

After this framework is introduced, the problem that has been used to evaluate the MPC and IDVD approaches is presented, and brief overview of the experimental set-up is provided. An overview of the theoretical formulation of the two different control approaches is then presented. The practical implementation details, as well as the simulation and experimental results, are then given. Finally, a discussion about the experimental results and the differences between the two control approaches is provided.

2. STANDARD TEST FRAMEWORK

When experimentally comparing the performance of multiple GNC approaches for RPO, it may be useful to have a standard framework to benchmark and compare them. As the experimental evaluation is heavily dependent on the available test bed, this proposed framework provides qualitative guidelines that are used to then specify the particular test scenarios for each individual test bed. Additionally, the experimental evaluation results are also dependent on the test bed and thus the comparison will only be valid between experiments made at the same test bed and conducted with the same underlying hardware (e.g. same sensors and actuators).

The goal of this test framework is to define a standard set of test scenarios so that the different GNC approaches can be evaluated under equal conditions. A test facility may use these test scenarios to evaluate different control approaches and thus build up a historical data-set helping them compare all of the different GNC approaches that have been tested. A standard set of metrics is also provided, generating an objective and quantitative measure of the GNC approach performance.

As an initial attempt to define this standard test framework the following qualitatively standard test scenarios are proposed.

1. Straight-on docking. The chaser starts in-line with the deputy and it has to simply move in a straight line to dock with the deputy. Obstacles may be present forcing the deputy to deviate from a simple straight line approach.

2. Lateral approach. The chaser starts from a lateral position where it must first navigate around the deputy and any obstacles before docking. The purpose of this test is evaluate the constraint handling capabilities of the Guidance algorithm. A variation on this would be an approach from the back, where the chaser must entirely circumnavigate the deputy to get into the entry cone.

3. Trajectory Following. In a more general case the chaser may need to follow a prescribed trajectory (e.g. circumnavigation for inspection).

Adding a keep out zone (i.e. obstacle) can be used as a variation of the above described scenarios to test obstacle avoidance capability and path re-planning. Another variation may be to have a moving deputy (e.g. spinning) or a moving obstacle so the ability of the GNC approach to adapt to a dynamic environment is also tested.

As these scenarios are intended to be used for experimental campaigns they need to be adapted for a specific experimental test-bed. Therefore the initial conditions of the chaser, the location and orientation of the deputy, and the size and location of the keep-out zones will need to be specified according to the properties of that specific test bed .

2.1. Metrics

In order to compare the performance of the different control approaches, three different metrics are proposed. The first proposed metric is the control effort, u_T, which is defined as

$$u_T = \int_{t_0}^{t_f} \|u\| \, dt. \tag{1}$$

The control effort measures how efficient the guidance and control approach is. An L^1-norm has been selected as its output is in Ns, providing results that are intuitive and that can be converted into other meaningful quantities (i.e. fuel and thruster on-time). The time to complete the maneuver, $\Delta t = t_f - t_0$ and the constraint handling (i.e. constraint violation) are also important metrics. In some applications a faster maneuver time may be preferred over reducing the control effort and when constraints are imposed it is important to establish whether the proposed approach can handle such constraints, which may be a limiting factor for other proposed approaches.

The computational cost to solve the control algorithm is the final proposed metric. An example of computational cost can be the CPU time required to solve the control algorithm. Some advanced control approaches can be very computationally intensive, limiting their applicability in computationally constrained vehicles.

It has to be noted that the performance of a GNC approach is heavily dependent on its implementation and the available hardware used during the testing. Substantially different results can potentially be obtained by using a different implementation or different hardware.

Zappulla [3] used the same methodology to experimentally compare the performance of Artificial Potential Functions (APF) and an Adaptive APF using the NPS-FSS test facility. In that particular work three different initial conditions and multiple obstacle configurations are explored.

3. PROBLEM FORMULATION

One case, from the multiple test cases defined in the previous section, has been selected to experimentally compare the MPC and IDVD control approaches at the NPS-FSS experimental facility. Figure 1 schematically shows the test setup with the problem initial conditions at the NPS-FSS. The numerical parameters for this test case are provided in Table 1. The selected test case is a straight-on docking with a static obstacle and an entry cone constraint.

It is worth noting that the FSS positions are measured from their respective VICON motion capture object frames, which have been setup to be approximately coincident with their geometric center. The desired end state of the chaser will then have an offset with respect to the deputy's position. From the same reason, and to ensure a safe obstacle avoidance, a keep-out constraint is placed around the obstacle. The

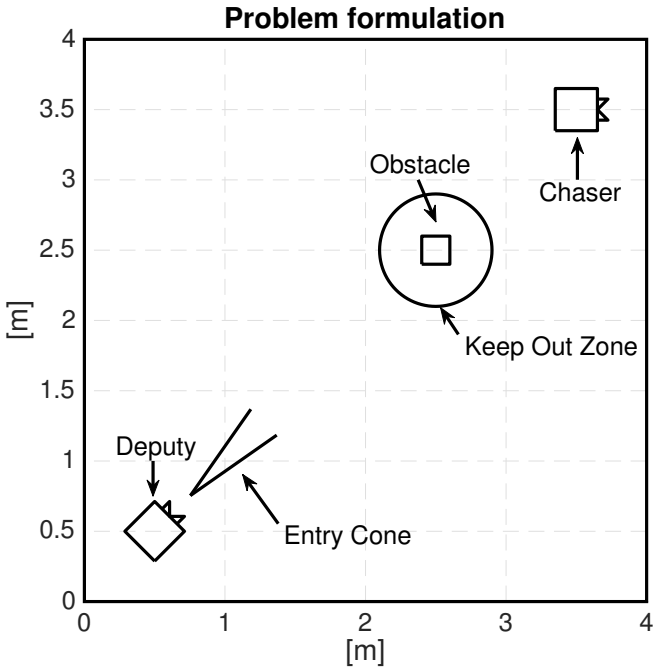

Fig. 1: Experiment initial conditions on the NPS-FSS test bed.

Table 1: Numerical parameters of the test case.

Parameter	Value
Chaser initial state	$x_{c0} = \begin{bmatrix} 3.5 & 3,5 \end{bmatrix}$ m
Deputy location	$x_d = \begin{bmatrix} 0.5 & 0.5 \end{bmatrix}$ m
Obstacle location	$x_{\text{obs}} = \begin{bmatrix} 2.5 & 2.5 \end{bmatrix}$ m
Obstacle keep out radius	$r_{\text{obs}} = 0.4$ m
Entry cone orientation	$\theta = 45$ deg.
Entry cone half-angle	$\theta_{hc} = 10$ deg.
Entry cone range	$r_{\text{cone}} = 0.75$ m

keep-out zone size, imposed on the chaser's reference frame origin, is defined to avoid any collision (regardless of the obstacle or chaser's relative orientation). It is also important to note that the chaser will need to avoid the keep-out zone, as this one is in the middle of what would be the optimal straight line trajectory if no obstacle was included. This can be clearly seen in Fig. 1.

The focus of this experiment campaign is to compare two control approaches and thus the navigation problem will be considered solved. The obstacle and the deputy's position will be available to the chaser vehicle. Additionally, the algorithms to be evaluated will only be used to control the vehicle's position - the attitude will be controlled using a combination of a fuel-optimal slew and a simple Proportional-Derivative (PD) controller. Resultantly, only the translational forces will be considered in the control effort evaluation.

3.1. Experimental Set-Up

The experiments have been conducted using two approximately 10 kg Floating Spacecraft Simulators (FSS). These vehicles float via air-pads over a 4-by-4 meter polished granite monolith surface recreating a reduced gravity and a quasi-frictionless motion in two translational and one rotational degrees-of-freedom (planar motion) [3, 4, 5, 6, 7, 8, 9, 10, 11, 12]. Figure 2 shows the deputy, chaser and obstacle FSS over the granite surface in the initial conditions used for this particular test.

Eight cold-gas thrusters provide autonomous motion capability to the FSS. An onboard tank of compressed air (propellant), a power system and on-board computer give them full autonomy. All the required processing (sensor readings, communications, navigation, guidance and control, and actuator commanding) is handled on-board in real-time.

Navigation data is provided by an overhead optical motion capture system (VICON). The provided position and attitude information is augmented by an on-board one-axis Fiber Optic Gyroscope (FOG). Communication between multiple FSS, the VICON workstation and other PC (used for telemetry monitoring and software upload) is achieved via TCP/UDP protocol over an ad-ho Wi-Fi network.

3.2. FSS Dynamics Model

The FSS can be modeled as a double integrator with two translational and one rotational degree-of-freedom. The equations of motion of the FSS are written as follows

$$\begin{aligned} \ddot{x} &= \frac{F_x}{m}, \\ \ddot{y} &= \frac{F_y}{m}, \\ \ddot{\theta} &= \frac{\tau}{I_z}, \end{aligned} \tag{2}$$

where m denoting the mass of the FSS, I_z the moment of inertia about the vertical axis, and $F_x,\ F_y,\ \tau$ the control forces and torque, respectively. These equations can then be written in state-space form as

$$\dot{\mathbf{x}} = A\mathbf{x} + B\mathbf{u}, \tag{3}$$

where $\mathbf{x} = [x,\ y,\ \theta,\ \dot{x},\ \dot{y},\ \dot{\theta}]^T$ denoting the state vector, $\mathbf{u} = [F_x,\ F_y,\ \tau]^T$ the control vector, and A and B the corresponding state and control constant matrices, respectively.

A control method can then be designed to control the linear-time invariant (LTI) system described by Eq. (3). In this case, the MPC and IDVD algorithms will be used to control the translational states, as the focus of this experimental campaign. Attitude control of the FSS is achieved through a fuel-optimal, bang-off-bang slew, which switches to a PD control law to maintain pointing at the desired angle.

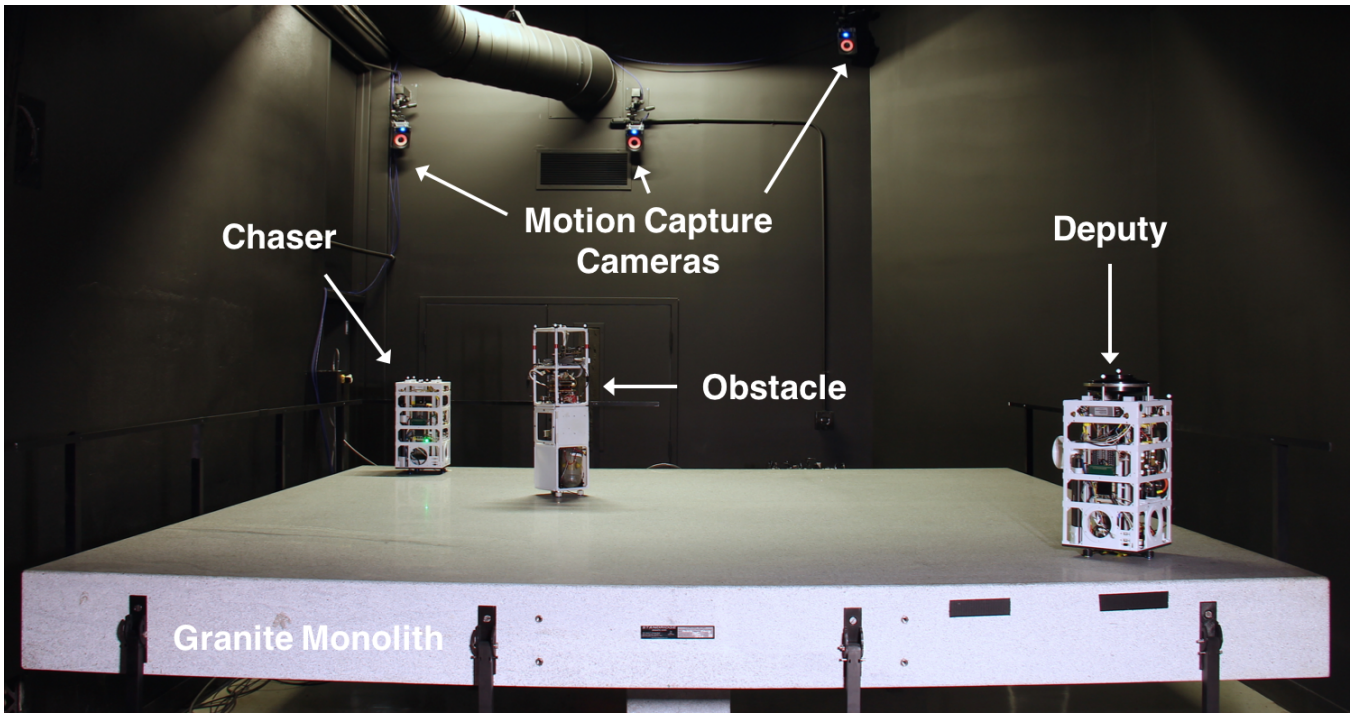

Fig. 2: Floating Spacecraft Simulators on top of the 4-by-4 meter granite surface in the initial conditions used for the experiment campaign .

4. CONTROLLER DESIGN

Two different control approaches have been experimentally evaluated for this paper.

4.1. Linear-Quadratic Model Predictive Control

Model Predictive Control (MPC) is a receding horizon control approach that can be used to solve constrained trajectory optimization problems. MPC is fundamentally based on the Linear Quadratic Regulator (LQR) problem, but can also be implemented in a nonlinear fashion. The main advantage is that the optimization problem formed through the MPC framework can be solved efficiently since only the solution over a finite prediction horizon is considered. A LQ-based approach (LQMPC) formulation has been considered for this comparative study.

Implementation of MPC for spacecraft RPO maneuvers have been studied in the past. A survey of guidance algorithms that can be used for on-board RPO trajectory planning is presented in [13], which includes an implementation of MPC in simulation. An experimental validation of an MPC algorithm provides further confidence in its ability to be implemented on-board a real system. Simulation results of applications of MPC to a constrained rendezvous problem have also been shown in [14] and [15]. The types of constraints enforced in these simulations include thrust constraints, a line-of-sight constraint linearized through polyhedral approximations [14], and an obstacle avoidance constraint linearized through a rotating hyperplane [15]. These references provide the basic framework for the MPC formulation implemented for this experimental campaign.

As an LQ-based method, LQMPC can be used to solve a constrained optimization problem, where a quadratic cost function is minimized subject to linear dynamic constraints, and linear inequality constraints. This problem formulation results in a convex, quadratic programming (QP) problem that can be solved using readily available solvers [16]. The obstacle avoidance constraint is linearized through a rotating hyperplane method [15]. The approach cone constraint is linearized by constructing two hyperplanes that define the edges of the cone, and intersecting 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. The LQMPC problem, in discrete form, is formed as follows.

Minimize:

$$\begin{aligned} J &= (\mathbf{x}(N)-\mathbf{x}_t)^T P(\mathbf{x}(N)-\mathbf{x}_t) + \\ &\quad \sum_{i=0}^{N-1} (\mathbf{x}(k+i)-\mathbf{x}_t)^T Q(\mathbf{x}(k+i)-\mathbf{x}_t) \\ &\quad + \mathbf{u}(k+i)^T R\mathbf{u}(k+i), \end{aligned} \tag{4a}$$

subject to

$$\begin{aligned}
\mathbf{x}(k+1) &= A_d\mathbf{x}(k) + B_d\mathbf{u}(k), && \text{(4b)}\\
|u_1(k)| &\leq u_{max}, && \text{(4c)}\\
|u_2(k)| &\leq u_{max}, && \text{(4d)}\\
\hat{n}_{obs}\cdot\mathbf{r}(k) &\geq \hat{n}_{obst}\cdot\mathbf{p}_{obs}, && \text{(4e)}\\
\hat{n}_{c1}\cdot\mathbf{r}(k) &\geq \hat{n}_{c1}\cdot\mathbf{p}_{dock}, && \text{(4f)}\\
\hat{n}_{c2}\cdot\mathbf{r}(k) &\leq \hat{n}_{c2}\cdot\mathbf{p}_{dock}. && \text{(4g)}
\end{aligned}$$

The length of the horizon is denoted by N, and $A_d,\ B_d$ are the discrete state and control matrices, which can be derived from the continuous dynamics in Eq. (2), 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. (4a) define the cost function weights on the final condition, state, and control variables, respectively. Eq. (4b) defines the equality constraint enforcing the dynamics of the system. Eq. (4c) and (4d) enforce constraints on the control variables. Finally, Eq. (4e-4g) enforce hyperplane constraints for the obstacle and cone, where $\hat{n}_{()}$ defines the normal vector of the hyperplane and $\mathbf{p}_{()}$ defines a point on the hyperplane. This problem is transformed into a QP problem [16], and solved using a publicly available MATLAB-based solver. The QP solver outputs the required control inputs for the entire horizon. In order to introduce a degree of robustness through feedback action, the first control input is extracted from the solution and applied. The QP is then resolved at the next iteration.

The LQMPC formulation is implemented in Simulink as an embedded MATLAB function, which is suitable for automatic code generation. The C code is then automatically generated from Simulink. It is then later compiled targeting the embedded hardware architecture and executed on-board the FSS's real time operating system.

4.2. Inverse Dynamics in the Virtual Domain

An IDVD controller uses a function that depends on a set of parameters (e.g. as polynomials, splines) to parametrizes the complete trajectory (from the current state up to the desired end state) and also the time [17, 18, 19, 20, 21]. The trajectory parameters (e.g. polynomial coefficients) are found via an optimization algorithm such that the control effort is minimized while meeting the imposed constraints. Once the trajectory parameters are obtained, the control input can be easily determined. This procedure is repeated at every time step to guide the vehicle towards the desired end state.

In this IDVD implementation, the trajectory will be constructed as a polynomial that is a function of the virtual time κ as

$$x(\kappa) = \sum_{i=0}^{n_x} a_i\kappa^i, \qquad y(\kappa) = \sum_{i=0}^{n_y} b_i\kappa^i. \tag{5}$$

The time t is also modeled as a polynomial of the virtual time κ as

$$t_x(\kappa) = \sum_{i=1}^{n_t} d_{ai}\kappa^i, \qquad t_y(\kappa) = \sum_{i=1}^{n_t} d_{bi}\kappa^i. \tag{6}$$

The virtual time is then $\kappa \in [0, \kappa_f]$. The time is also parameterized as to provide extra optimitzation variables and thus reduce the final control effort or provide an improved constraint handling.

It is worth noting that different polynomial orders can be used for each of the trajectory components $n_x \neq n_y$. Additionally, different virtual time polynomials (see Eq. (6)) can be used for each of the trajectory components, imposing $t_x(\kappa_f) = t_y(\kappa_f)$. It is also worth pointing that the time t has to be monotonically increasing and thus $t' = dt/d\kappa$ needs to be positive (so that time t is univocally determined by κ),

$$t' = \frac{dt}{d\kappa} = \sum_{i=1}^{n_t} i d_i\kappa^{i-1} > 0. \tag{7}$$

The time derivatives of the trajectory are then computed as follows,

$$t''(\kappa) = \frac{d^2t}{d\kappa} = \sum_{i=2}^{n_t} i(i-1)d_i\kappa^{i-2}, \tag{8}$$

$$\dot{x} = \frac{dx}{dt}\frac{d\kappa}{d\kappa} = \frac{x'}{t'}, \tag{9}$$

$$x' = t'\dot{x}, \tag{10}$$

$$\ddot{x} = \frac{d\dot{x}}{dt}\frac{d\kappa}{d\kappa} = \frac{d(x'/t')}{d\kappa}\frac{1}{t'} = \frac{x''}{t'^2} - \frac{x'}{t'^3}t'' = \frac{x''}{t'^2} - \frac{\dot{x}}{t'^2}t'', \tag{11}$$

$$x'' = t'^2\ddot{x} + t''\dot{x}. \tag{12}$$

The derivatives with respect to the virtual time are easily computed as follows,

$$x' = \sum_{i=1}^{n_x} i a_i\kappa^{i-1}, \tag{13}$$

$$x'' = \sum_{i=2}^{n_x} i(i-1)a_i\kappa^{i-2}. \tag{14}$$

The initial conditions of the FSS are known and the final conditions of the FSS are set based on the desired final state,

$$x(0) = x_0, \quad \dot{x}(0) = \dot{x}_0, \tag{15}$$

$$x(t_f) = x_f, \quad \dot{x}(t_f) = \dot{x}_f, \quad \ddot{x}(t_f) = \ddot{x}_f. \tag{16}$$

In general, for a docking scenario, the final acceleration will be set to zero $\ddot{x}_f = 0$ and the final velocity may be also set to zero or to certain small terminal velocity as to ensure a successful latching.

As the trajectory must comply with the initial and the desired final states, these are used to set the first five trajectory coefficients. The polynomial order will then need to be larger or equal to four $n_{x,y} \geq 4$. The coefficients that are left to shape the trajectory, minimizing cost function while meeting constraints, are the trajectory coefficients a_i and b_i with $i \geq 5$, the virtual time coefficients d_i where $i \geq 1$ and the final virtual time κ_f. It has to be noted that if different time polynomials are used for the different components then the d_{b1} is set to that $t_x(\kappa_f) = t_y(\kappa_f)$ is met.

4.2.1. Constraints

The final time shall be less than the maximum user defined time as

$$t(\kappa_f) < t_{max}. \tag{17}$$

Additionally, the time shall be monotonically increasing with κ (see Eq. (7)). Although not providing an optimal solutions, this last constrain will be enforced by a imposing a lower bound on the d_i coefficient as $d_i > 0$.

The force that the FSS can produce is limited and thus this also included as a constraint.

$$F_x(\kappa) < F_{max}, \qquad F_y(\kappa) < F_{max}. \tag{18}$$

Finally, the scenario's obstacle and entry cone constraints are also imposed.

4.2.2. IPOPT implementation

The IDVD problem, as implemented here, is inherently non-linear. The open-source NLP Interior Point OPTimizer (IPOPT) [22] solver is used to find the solution to the IDVD problem at each time step.

The IPOPT routine is wrapped as a Simulink S-functions suitable for automatic code generation. The resulting C code is compiled for the target hardware and is executed in a real-time operating system on-board the FSS. The IDVD problem is subsequently solved at a 5 Hz frequency.

5. SIMULATION AND EXPERIMENTAL RESULTS

A simulator that recreates the FSS dynamics and emulates the different on-board sensors and actuators is first used to design, validate and tune the controllers.

Table 2 shows the parameters used for the LQMPC algorithm to run both the simulator and experimental cases, where $\bar{P}$ is the solution to the Algebraic Riccati equation for the discrete LQR problem.

Table 3 shows the parameters used for the IDVD controller. The total number of available variables for the IPOPT solver is 6 (a_5, b_5, d_{a1}, d_{a2}, d_{b2} and κ_f). As the IDVD approach has a tendency to hug the constraints it was observed that small cone constrain violations could occur on the edge of the cone. To alleviate this concern of not meeting the problem's cone constraint as defined in Table 1, the IDVD cone constraint was extended.

Table 2: LQMPC Parameters.

Parameter	Value
Q	$diag\left(10^2\ 10^2\ 10^5\ 10^5\right)$
R	$diag\left(10^3\ 10^3\right)$
P	$100\bar{P}$
Maximum force	$F_{\max} = 0.30$ N
Sample Time	$T_s = 5$ s
Horizon Length	$N = 20$
Maximum number of iterations	$n = 100$

Table 3: IDVD parameters.

Parameter	Value
Maximum time	$t_{\max} = 150$ s
Polynomial orders	$n_{x,y} = 5\ n_{tx,y} = 2$
Entry cone range	$r_{\text{IDVD cone}} = 1$ m
Maximum force	$F_{\max} = 0.075$ N
Sample Time	$T_s = 0.2$ s
Maximum number of IPOPT iterations	$n = 25$

Figure 3 shows the trajectories followed by the FSS, while Table 4 provides the performance metrics for the MPC and IDVD controllers when executed on the simulator. The solid black circles located along the trajectory represent the control effort expended during the last 10 seconds, thus visually indicating the distribution of the control effort along the trajectory. As these black circles are evenly spaced in time they also indicate the velocity at which the FSS traversed the trajectory.

These controllers are then compiled and executed on a real-time environment onboard the FSS. Figure 4 shows the

Table 4: Comparison of simulation performance metrics for MPC and IDVD.

Metric (Simulation)	MPC	IDVD
Control effort [Ns]	5.5	2.4
Time [s]	132	148.0
Constraint Violation	No	No

Table 5: Comparison of experiment performance metrics for MPC and IDVD.

Metric (Experiment)	MPC	IDVD
Control effort	5.1	2.7
Time [s]	108.5	146.1
Constraint Violation	No	No

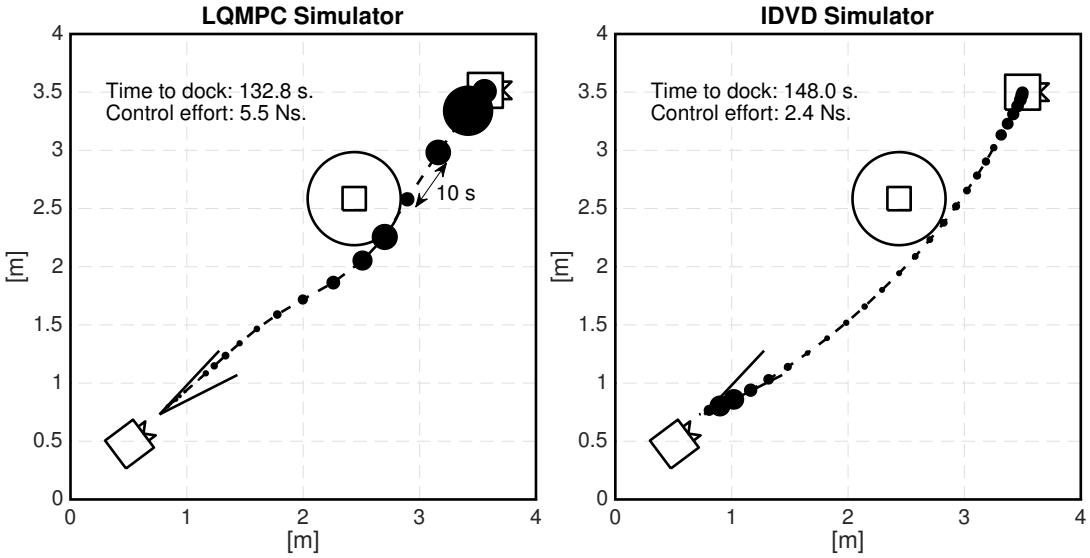

Fig. 3: Trajectories for MPC and IDVD on the simulator.

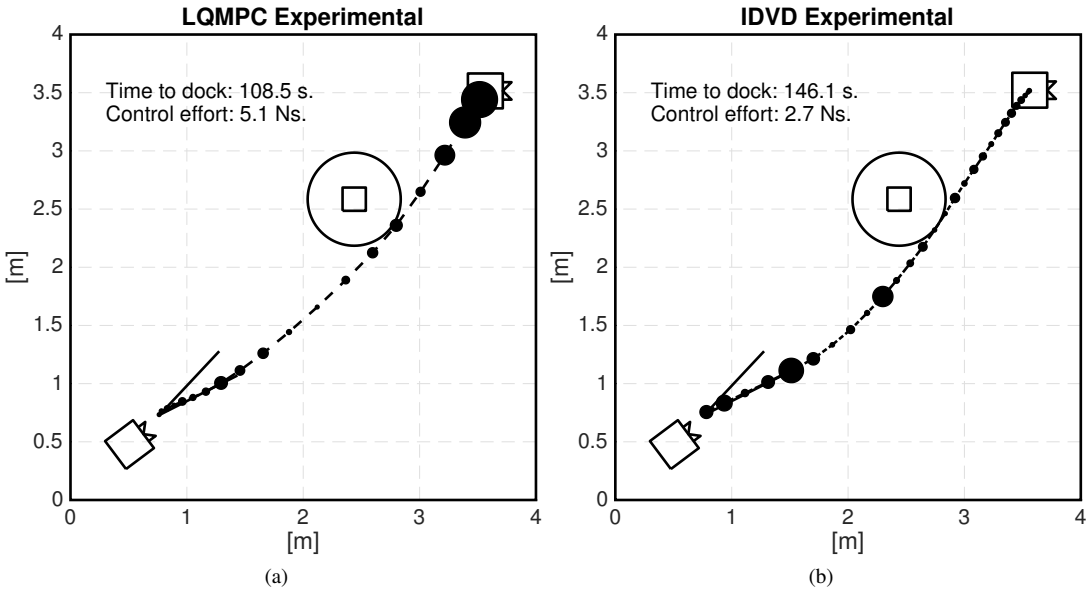

Fig. 4: Trajectories for MPC and IDVD for the experimental campaign.

trajectories followed by the FSS while Table 5 provides the performance for the experiment campaign.

6. DISCUSSION

The results in the previous section showed slight differences between the simulation and experiment, most notable in the case of the LQMPC algorithm. Differences are expected due to physical attributes of the test bed that cannot be fully modeled in simulation, such as thruster misalignment or uncertainties in the FSS state and in the thrust levels. Although the navigation problem has been considered solved, the VICON motion capture system only provides position and attitude information. A discrete Kalman filter is then used to obtain the velocity estimates, which in the case of the angular rate, are augmented by an onboard FOG. The velocities thus suffer from a certain amount of noise which can slightly alter the controller's behavior. Additionally, the output from both

control methods go to a delta-sigma modulator for thruster actuation.

However, it can be seen from Figures 3 and 4 that similar trajectories were followed by the FSS in simulation and experiments. It is also clearly seen in these figures that both algorithms successfully reach the final docking condition while meeting constraints.

It is also worth pointing that the simulation results for the IDVD are much smoother, in terms of thrust, than the experiment results. With increasing polynomial order and the subsequent solving at each time step, the IDVD approach tends to converge to a bang-off-bang type of solution. This can be clearly seen on the simulation results (see Fig. 3). During the experiments, a slightly different thrust output was observed. Mainly, it can be seen that when it gets within range of the entry cone constraint (see Table 3) the thrusting become more aggressive. Due to the nature of the IDVD approach, the FSS tends to get very close to the constraint limits. Under these conditions, small uncertainties in state estimation and thruster misalignment can bring the FSS in a trajectory that violates the constraint and thus the IDVD approach tries to correct this condition at all cost. The IDVD approach is then sensitive to these type of uncertainties when navigating close to the constraint border.

Another, potentially undesirable, effect of to the IDVD approach is the tendency to only decelerate when it gets very close to its desired end state. This behavior can be attributed to the tendency to behave in a similar manner as a bang-off-bang type controller. Safety and operational constraints may limit the velocity at which the chaser approaches the deputy, or my limit the amount of thrusting that is allowed in the immediate vicinity of the deputy. To limit this aggressive deceleration, the force constraint on IDVD controller has been artificially reduced. The available maximum force has been lower to a quarter of the actual achievable level by the FSS to limit. By adding an upper velocity limit constraint a similar modification of this tendency could be obtained (at the expense of increasing the computational burden).

In comparing the two algorithms from the results in Table 5, a few conclusions can be drawn. The IDVD algorithm provided a more fuel efficient solution, since the control effort was smaller. However, the LQMPC solution took less time to achieve final docking condition. This result is expected since IDVD solves the nonlinear optimization problem without approximating the obstacle avoidance, whereas LQMPC approximates the non-convex constraint through the rotating hyperplane method. This approximation tends to over-constrain the problem, leading to a less optimal solution, but provides a problem that can be solved much more efficiently with guaranteed of convergence. Additionally, the LQMPC cost function includes both state and control terms, whereas IDVD minimizes the control effort directly. This is illustrated by the fact that the LQMPC method minimizes both state error and control effort via the Q and R matrices respectively. Increasing the R weighting values could, for example, result in a trajectory that uses less control effort, but would take longer to reach docking conditions.

Another interesting difference between the MPC and the IDVD approach is that the IDVD approach includes a maximum docking time constraint. The MPC approach time to dock is governed, by the problem itself, but can be controlled via the Q and R matrices. In the conducted experiment the time to dock between the MPC and the IDVD approach are significantly different. A fairer comparison could be drawn if the controllers where tuned to achieve a similar docking time.

The computational cost of the MPC and IDVD approaches has not been specifically measured and thus a direct quantitative comparison on this magnitude can not be made at this time. With its guaranteed convergence the LQMPC approach limited its number of iterations to 100. This limit was rarely reached, only when the FSS found itself close to a region of local infeasibility. However, in these cases, the resulting LQMPC control input pushes the FSS towards the feasibility region and thus the LQMPC did not suffer from reaching this limit. The NLP IPOPT solver used for the IDVD approach has no guaranteed convergence and thus more precautions had to be taken. In order to meet the 5 Hz rate, a 25 iteration limit on the IPOPT solver was imposed. That limit was actually reached on a few occasions, mainly when the FSS found itself close to an infeasible region (e.g. close to a constraint boundary). In the IPOPT, failing to converge can produce a completely erroneous control input. To help the FSS recover from this situation the IDVD algorithm had to run at a faster rate. Using the IPOPT warm start capabilities, convergence was usually achieved in under 10 iterations.

These experimental and simulation results show a good example of the trade-offs that must be considered between optimality in terms of time and fuel. The set of standard cases presented in this paper, along with the results from a specific test case, can be used to benchmark these guidance algorithms and study these trade-offs in order to compare algorithm performance. These types of studies could be used as a basis for providing quantitative information when selecting algorithms for specific applications.

7. CONCLUSIONS

A Linear Quadratic Model Predictive Control (MPC) based approach and an Inverse Dynamics in the Virtual Domain (IDVD) based approach have been experimentally evaluated on a planar air-bearing table experimental set-up. The chaser spacecraft had to autonomously dock with the deputy in the presence of a keep out zone and an entry cone constraint. Both approaches achieved its goal with IDVD exhibiting a slightly smaller cost while MPC achieved faster docking times. A real-time implementation of both controllers has been achieved. Additionally a standard experimental test framework has been proposed with the aim to help provide

meaningful experimental comparisons of different spacecraft guidance, navigation and control algorithms.

8. REFERENCES

[1] Douglas Zimpfer, Peter Kachmar, and Seamus Tuohy, *Autonomous Rendezvous, Capture and In-Space Assembly:Past, Present and Future*, American Institute of Aeronautics and Astronautics, 2005.

[2] Marco B Quadrelli, Lincoln J Wood, Joseph E Riedel, Michael C McHenry, MiMi Aung, Laureano A Cangahuala, Richard A Volpe, Patricia M Beauchamp, and James A Cutts, "Guidance, navigation, and control technology assessment for future planetary science missions," *Journal of Guidance, Control, and Dynamics*, vol. 38, no. 7, pp. 1165–1186, 2015.

[3] Richard Zappulla II, Hyeongjun Park, Josep Virgili-Llop, and Marcello Romano, "Experiments on autonomous spacecraft rendezvous and docking using an adaptive artificial potential field approach," in *26th AAS/AIAA Space Flight Mechanics Meeting, Napa, CA. 14-18 February 2016. AAS 16-459*, 2016.

[4] Marcello Romano and Jason Hall, "A testbed for proximity navigation and control of spacecraft for on-orbit assembly and reconfiguration," in *Proceedings of the AIAA Space 2006 Conference and Exhibit*, 2006, pp. 1–11.

[5] Marcello Romano, David A. Friedman, and Tracy J. Shay, "Laboratory experimentation of autonomous spacecraft approach and docking to a collaborative target," *Journal of Spacecraft and Rockets*, vol. 44, no. 1, pp. 164–173, 2007.

[6] Jason S. Hall and Marcello Romano, "Novel robotic spacecraft simulator with mini-control moment gyroscopes and rotating thrusters," in *Advanced intelligent mechatronics, 2007 IEEE/ASME international conference on*. 2007, pp. 1–6, IEEE.

[7] Riccardo Bevilacqua, Jason S. Hall, James Horning, and Marcello Romano, "Ad hoc wireless networking and shared computation for autonomous multirobot systems," *Journal of Aerospace Computing, Information, and Communication*, vol. 6, no. 5, pp. 328–353, 2009.

[8] Riccardo Bevilacqua, Andrew Caprari, Jason Hall, and Marcello Romano, "Laboratory experimentation of multiple spacecraft autonomous assembly," in *AIAA Guidance, Navigation and Control Conference and Exhibit*, 2009.

[9] Claudio Lugini and Marcello Romano, "A ballistic-pendulum test stand to characterize small cold-gas thruster nozzles," *Acta Astronautica*, vol. 64, no. 5, pp. 615–625, 2009.

[10] Fabio Curti, Marcello Romano, and Riccardo Bevilacqua, "Lyapunov-based thrusters' selection for spacecraft control: analysis and experimentation," *Journal of guidance, control, and dynamics*, vol. 33, no. 4, pp. 1143–1160, 2010.

[11] Jason S. Hall and Marcello Romano, *Laboratory experimentation of guidance and control of spacecraft during on-orbit proximity maneuvers*, INTECH Open Access Publisher, 2010.

[12] Marco Ciarcia, Alessio Grompone, and Marcello Romano, "A near-optimal guidance for cooperative docking maneuvers," *Acta Astronautica*, vol. 102, pp. 367–377, 2014.

[13] Costantinos Zagaris, Morgan Baldwin, Christopher Jewison, and Christopher Petersen, "Survey of spacecraft rendezvous and proximity guidance algorithms for on-board implementation," in *Advances in the Astronautical Sciences (AAS/AIAA Spaceflight Mechanics 2015)*, 2015, vol. 155.

[14] A. Weiss, I. Kolmanovsky, M. Baldwin, and R.S. Erwin, "Model predictive control of three dimensional spacecraft relative motion," in *American Control Conference (ACC), 2012*, June 2012, pp. 173–178.

[15] Hyeongjun Park, S. Di Cairano, and I. Kolmanovsky, "Model predictive control for spacecraft rendezvous and docking with a rotating/tumbling platform and for debris avoidance," in *American Control Conference (ACC), 2011*, June 2011, pp. 1922–1927.

[16] Matthew Brand, Vijay Shilpiekandula, Chen Yao, Scott A. Bortoff, Takehiro Nishiyama, Shoji Yoshikawa, and Takash Iwasaki, "A parallel quadratic programming algorithm for model predictive control," in *Proc. 18th World Congress of the International Federation of Automatic Control (IFAC)*, 2011, vol. 18, pp. 1031–1039.

[17] George A Boyarko, Marcello Romano, and Oleg A Yakimenko, "Time-optimal reorientation of a spacecraft using a direct optimization method based on inverse dynamics," in *Aerospace Conference, 2010 IEEE*. IEEE, 2010, pp. 1–13.

[18] George Boyarko, Oleg Yakimenko, and Marcello Romano, "Real-time 6dof guidance for of spacecraft proximity maneuvering and close approach with a tumbling object," in *AIAA/AAS Astrodynamics Specialist Conference, Toronto, Ontario, Canada, August*, 2010, pp. 2–5.

[19] Marco Ciarcia and Marcello Romano, "Spacecraft proximity maneuver guidance based on inverse dynamic and

sequential gradient-restoration algorithm," *Advances in the Astronautical Sciences*, vol. 142, 2011.

[20] M Ciarcià and M Romano, "Suboptimal guidance for orbital proximity maneuver with path constraints capability," *Proceedings of the AIAA Guidance Navigation and Control Conference*, 2012.

[21] Jacopo Ventura, Marcello Romano, and Ulrich Walter, "Performance evaluation of the inverse dynamics method for optimal spacecraft reorientation," *Acta Astronautica*, vol. 110, pp. 266–278, 2015.

[22] Andreas Wächter and T. Lorenz Biegler, "On the implementation of an interior-point filter line-search algorithm for large-scale nonlinear programming," *Mathematical Programming*, vol. 106, no. 1, pp. 25–57, 2005.