AAS 14-249

MODEL PREDICTIVE CONTROL AND EXTENDED COMMAND GOVERNOR FOR IMPROVING ROBUSTNESS OF RELATIVE MOTION GUIDANCE AND CONTROL

Christopher Petersen,* Andris Jaunzemis,† Morgan Baldwin,‡
Marcus Holzinger§ and Ilya Kolmanovsky¶

The paper describes two approaches to improving on-board robustness of spacecraft relative motion guidance and control. The first approach is based on Model Predictive Control (MPC), and this paper demonstrates its capability for robust trajectory execution by changing the environment with a sudden obstacle appearing along its path. The second, novel approach uses an Extended Command Governor (ECG) that augments a nominal LQ controller. The ECG modifies commanded set-points to the inner loop LQ controller when it becomes necessary to avoid constraint violation. Simulations are used to compare the two approaches and experimental results of MPC, based on an omnibot system developed for validation of spacecraft relative motion control algorithms, are reported.

INTRODUCTION

The need for robust Earth spacecraft guidance and control is becoming greater as of recently, especially in the realm of low orbits where the danger of debris collision is prominent. The crash between the U.S. satellite manufactured by Iridium and a non-operational Russian Cosmos satellite in 2009 is one such collision that not only resulted in the destruction of both satellites, but also in the creation of a large number of smaller debris, any of which could be the cause of another downed satellite.[1] Much research has already been done in the field of robust spacecraft maneuver planning in order to prevent such an event from happening.[2] Traditionally, such approaches to spacecraft control lead to open-loop maneuvers, see References 3 and 4 for examples.

Feedback guidance and control, however, is becoming increasingly attractive as the necessity for spacecraft autonomy increases. Model Predictive Control (MPC) is a frequently used feedback control approach for systems with constraints. Relative motion control for line-of-sight rendezvous using MPC has been considered in References 5–9 and has been shown to improve maneuver robustness. It has also been studied for spacecraft formation flying applications in the presence of sensing noise.[10] In terms of maneuver efficiency, MPC appears to be most advantageous when its prediction and control horizon span the entire maneuver. A long horizon such as this however requires considerable computational power. In this paper, MPC is applied to relative motion obstacle avoidance, where the obstacle may be sensed in the process of executing a maneuver. Simulations

*Graduate Student, Department of Aerospace Engineering, The University of Michigan, Ann Arbor, MI.
†Graduate Student, School of Aerospace Engineering, Georgia Institute of Technology, Atlanta, GA.
‡Research Aerospace Engineer, Space Vehicles Directorate, Air Force Research Laboratory, Kirtland Air Force Base, NM.
§Assistant Professor, School of Aerospace Engineering, Georgia Institute of Technology, Atlanta, GA
¶Professor, Department of Aerospace Engineering, The University of Michigan, Ann Arbor, MI.

are first performed with a reasonably long horizon, but this presents computational difficulties for on-board implementation. Therefore, on-board implementation and testing of MPC in this paper is done with a considerably shorter horizon, still showing that it may be used for fast, robust obstacle avoidance.

An Extended Command Governor (ECG), which modifies set-point commands to satisfy flight constraints, can also be used for spacecraft relative motion control. As opposed to MPC, ECG is an add-on to an existing controller, which is a Linear Quadratic (LQ) controller in this paper. Its simpler variant, the reference governor, has been previously applied to line-of-sight spacecraft rendezvous problems in Reference 11. The ECG can provide a larger constrained domain of attraction and faster response, even when disturbances are present,[12] but at the price of increased computational effort. This LQ/ECG approach will be applied to robust relative motion obstacle avoidance in this paper and compared to conventional MPC.

This work is organized as follows. First, the relative motion spacecraft dynamics model in both MPC and LQ/ECG design is presented. The MPC problem is then formulated and solved for spacecraft maneuvering in a changing environment in simulations. Then the Linear Quadratic and Extended Command Governor configuration for spacecraft maneuvers is explained, and simulation results are presented. To further demonstrate each scheme, a planar-motion robot implementation is introduced. The paper is ended with concluding remarks.

SPACECRAFT RELATIVE MOTION

Continuous Relative Motion Dynamics

The spacecraft equations of motion in this paper are formulated in the non-inertial Hill's frame. The origin of the frame is a chief spacecraft in a nominal circular orbit. The unit vectors are **i**, **j**, and **k**, and represent the radial track, the in-track, and the out-of-plane orbital positions, respectively. The relative position vector of a spacecraft with respect to the center of the Earth is

$$\mathbf{R} = \mathbf{R_0} + \delta\mathbf{r} = (R_0 + x)\mathbf{i} + y\mathbf{j} + z\mathbf{k}, \tag{1}$$

where $\mathbf{R_0}$ is the nominal orbital position vector of the chief spacecraft, $\delta\mathbf{r}$ is the relative position vector with respect to the chief's location, x, y, z are the positions of the spacecraft in the relative frame, and R_0 is the nominal orbital radius. Using Equation (1), the nonlinear equations of motion have the following form,

$$\ddot{\mathbf{R}} = -\mu\frac{\mathbf{R}}{R^3} + \frac{1}{m_c}\mathbf{F}, \tag{2}$$

where μ is the gravitational constant, $R = |\mathbf{R}|$, m_c is the mass of the spacecraft, and $\mathbf{F}$ is the vector of external forces applied to the spacecraft. $\ddot{\mathbf{R}}$ can be further separated into its vector components as

$$\ddot{\mathbf{R}} = (\ddot{x} - 2n\dot{y} - n^2x)\mathbf{i} + (\ddot{y} + 2n\dot{x})\mathbf{j} + (\ddot{z})\mathbf{k}, \tag{3}$$

where $n = \sqrt{\frac{\mu}{R_0^3}}$ is the mean motion on the nominal orbit. Equations (2)-(3) represent the nonlinear equations of motion of spacecraft in the relative motion Hill's frame.[13]

These nonlinear equations of motion can then be linearized by assuming (i) a circular chief orbit and (ii) that the distance relative to the chief is small in comparison to the radius of the chief orbit ($|\delta\mathbf{r}| \ll |\mathbf{R}|$). This results in the Clohessy-Wiltshire equations,

$$\begin{aligned} \ddot{x} - 3n^2x - 2n\dot{y} &= \frac{F_x}{m_c}, \\ \ddot{y} + 2n\dot{x} &= \frac{F_y}{m_c}, \\ \ddot{z} + n^2z &= \frac{F_z}{m_c}, \end{aligned} \tag{4}$$

where F_x, F_y, F_z are the components of the external force vector $\mathbf{F}$ in the Hill's frame.

Remark 1: We note that the linearization of Equation (4) is time-invariant as the nominal orbit is assumed to be circular. For relative motion maneuvers near a chief spacecraft on a nominally elliptic orbit, a set of time-varying linearized, equations can be derived. Since in this work the optimization solver used to generate the MPC solution is on-line, the case of elliptic orbits can be handled analogously.

Discrete Relative Motion Dynamics

For the MPC and LQ/ECG implementation, the linearized spacecraft dynamics in Equation (4) are transformed into the standard continuous-time system realization (A_c, B_c) and then converted to discrete-time assuming a sampling period of ΔT. This yields

$$\begin{aligned} X(k+1) &= AX(k) + BU(k), \\ Y_t(k) &= C_tX(k) + D_tU(k), \\ Y_c(k) &= C_cX(k) + D_cU(k), \end{aligned} \tag{5}$$

where $X(k) = [x(k), y(k), z(k), \dot{x}(k), \dot{y}(k), \dot{z}(k)]^T$ is the state vector of relative positions and velocities at time instant k, $Y_t(k)$ is the output vector that tracks set-point spacecraft equilibrium positions, $Y_c(k)$ is the output vector that constraints are imposed on, A is the discrete dynamics matrix obtained by $A = e^{A_c\Delta T}$, $U(k)$ is the control vector that represents an instantaneous change in the velocity vector (ΔV) of the spacecraft due to thruster burns, and B is the discrete control matrix, obtained by

$$B = e^{A_c\Delta T}\begin{bmatrix} 0_{3x3} \\ I_{3x3} \end{bmatrix}. \tag{6}$$

MODEL PREDICTIVE CONTROL

The Model Predictive Controller (MPC) functions first by optimizing the control sequence over a finite, receding horizon to minimize a selected cost subject to constraints. Then, the first element of the optimal sequence is applied over the first discrete time interval. The optimization horizon afterwards recedes by one step and the process is repeated starting with the current state as the initial condition. By recomputing the solution for the current state as the initial condition, MPC generates a feedback action that can compensate for the uncertainties and disturbances. MPC can also react to changes in the constraints such as an obstacle appearing along the maneuver path.

MPC Problem Formulation

For spacecraft maneuver planning, MPC solves the following minimization problem for discrete state $X(k)$ and control $U(k)$,

$$\min_{\bar{\mathbf{U}}} \sum_{i=0}^{N-1} \left((Y_t(k+i) - r(k+i))^T Q (Y_t(k+i) - r(k+i)) + U(k+i)^T R U(k+i) \right) + (X(k+N) - X_f)^T P (X(k+N) - X_f), \tag{7}$$

where

$$\bar{\mathbf{U}} = [U(k),\ U(k+1), ...U(k+N-1)]^T \tag{8}$$

$$\begin{aligned} \text{subject to} \quad & X(k+i+1) = AX(k+i) + BU(k+i), \\ & U_{min} \leq U(k+i) \leq U_{max}, \\ & Y_t(k+i) = \begin{bmatrix} I_{3\times 3} & 0 \end{bmatrix} X(k+i), \\ & i = 0, 1, \ldots, N-1, \end{aligned} \tag{9}$$

where N is the length of the horizon, $r(k+i)$ is the reference vector at discrete step $k+i$, X_f is the target end state, Q and R are the state and control weight matrices, and P is the solution to the discrete Riccati equation for the unconstrained infinite horizon variant of the MPC problem. In this paper it is assumed that the goal is to rendezvous with the chief satellite at the origin. Therefore, X_f and $r(k)$ are both zero vectors. This results in a cost function that is similar to a nominal LQ controller. The constraint on the second line of Equation (9) represents the control constraints. Other variants of this constraint can be considered depending on the thruster type and configuration, such as polyhedral approximations to 2-norm, which is relevant to the case of a single thruster spacecraft configuration.

Under appropriate assumptions, Equations (8)-(9) reduce to a convex quadratic programming problem of the form,

$$\min_{\bar{\mathbf{U}}} \ \frac{1}{2} \bar{\mathbf{U}}^T S U + H^T \bar{\mathbf{U}}, \tag{10}$$

subject to

$$V \bar{\mathbf{U}} \leq W, \tag{11}$$

where S and H are appropriately defined and depend on A, B, Q, R, and P.

Obstacle Avoidance Constraints

Obstacle avoidance, in contrast to the control constraint in Equation (9), is a non-convex problem. Utilizing MPC on such a problem is prone to difficulties, e.g. no guarantees exist to find a global minimal solution. Numerous approaches to dealing with obstacle avoidance have been developed, and in this paper we follow the approach in our previous works (see References 14, 5, and 6), where the obstacle is separated from the spacecraft by a rotating hyperplane and the MPC problem is solved subject to a convex rotating hyperplane constraint. When an obstacle is sensed by the spacecraft, a hyperplane tangent to the spacecraft is projected on the obstacle and rotated at a constant rate. At each discrete time step, the following linear constraint is imposed on the predicted state $X(k)$,

$$-\eta^T(k+i)Y_t(k+i) \leq -\eta^T(k+i)r_O(k+i), \;\; i = 0, \cdots, N \tag{12}$$

where $\eta(k)$ is a normal unit vector of the hyperplane at time k pointing outside of the obstacle and $r_O(k)$ is a point on the edge of the obstacle at time k. The rate and direction of rotation are governed by the spacecraft's initial and final positions, and the rotation is stopped once the normal from the center of the obstacle to the final position is aligned with the normal of the hyperplane. Including the rotating hyperplane constraint adds another layer of complexity to the MPC formulation. Therefore, this constraint is only active when the spacecraft senses an obstacle and is within a certain distance of it.

Simulation Results

In these simulations, MPC is used to guide a spacecraft, initially located 5 km ahead in the in-track direction, toward the origin, with a convergence radius of 50 m. The spacecraft dynamics are discretized at a sample rate of $\Delta T = 120$ seconds to provide sufficient time for robust maneuver changes. The horizon length is set at 360 steps (total time 43,200 sec), approximately half an orbit period at geostationary orbit, which allows for full utilization of the Clohessy-Wiltshire relative motion dynamics. The weighing matrices Q and R in Equation 7 weighting matrices are chosen as $1 \times 10^{-3}[I_{6\times6}]$ and $2 \times 10^{3}[I_{3\times3}]$, respectively. For the control constraint, a max ΔV of 0.01 km/sec in the radial track, the in-track, and the out-of-plane directions are imposed (i.e., $U_{min} = [-0.01 \;\; -0.01 \;\; -0.01]^T$ and $U_{max} = [0.01 \;\; 0.01 \;\; 0.01]^T$). Figures 1 - 2 show the spacecraft's path and ΔV inputs for a nominal case with no obstructions.

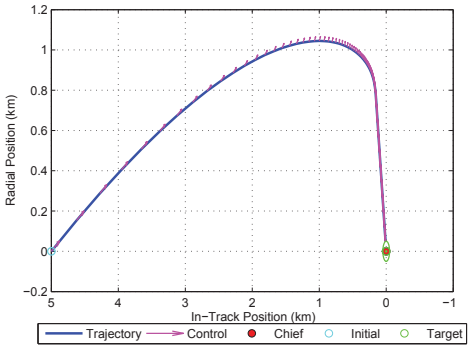
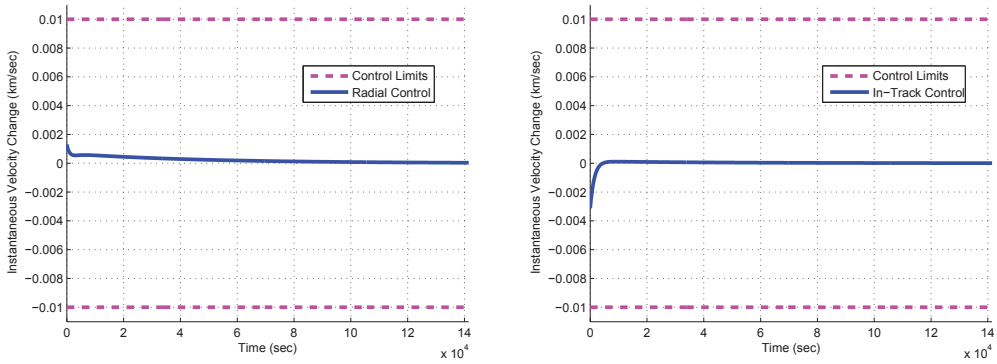


Figure 1. MPC Spacecraft Trajectory With Long Horizon and No Obstacle



(a) (b)

Figure 2. (a) Radial Track ΔV for MPC With Long Horizon and No Obstacle (b) In-Track ΔV for MPC With Long Horizon and No Obstacle

In Figure 1, the trajectory of the spacecraft is shown in blue while the ΔV vector at each time step is shown in pink. For this simulation, the spacecraft can be seen going out of the in-track direction after an initial large ΔV, and then exploits the relative motion dynamics to approach the chief efficiently with little control effort. In Figure 2, the control in the radial and in-track are shown for the entire duration of the maneuver in blue, while the control limit is represented by the pink dashed lines. As seen, the maneuver is completed well within the control limits. Since R is large, the control is heavily penalized, resulting in a ΔV efficient path. These results demonstrate that MPC can guide a spacecraft back to the chief satellite within control constraints, but since an optimal trajectory is recalculated at every time step, the algorithm is computationally expensive. To demonstrate the robustness of MPC to obstacles, another simulation is performed where an obstacle, represented as a sphere with a radius of 0.2 km, appears mid-trajectory.

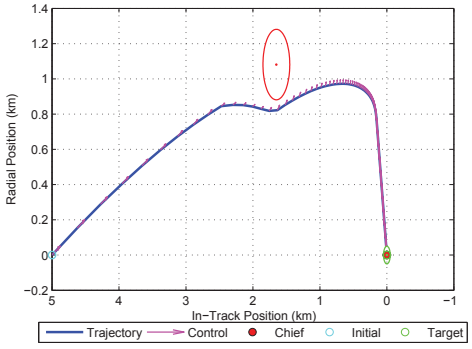
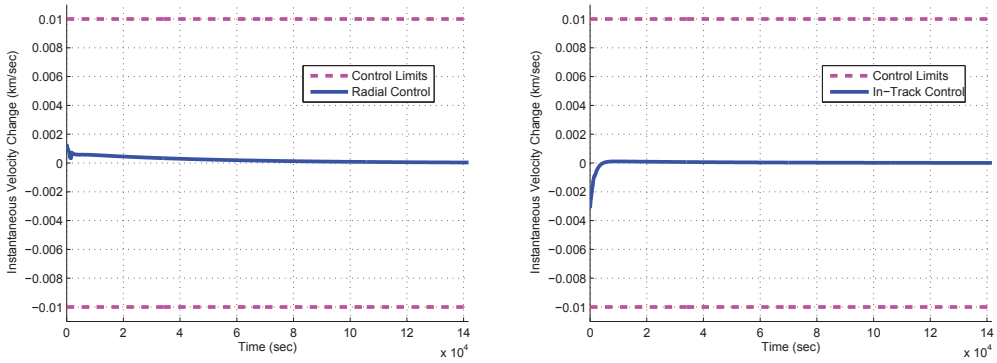


Figure 3. MPC Spacecraft Trajectory With Long Horizon and One Obstacle

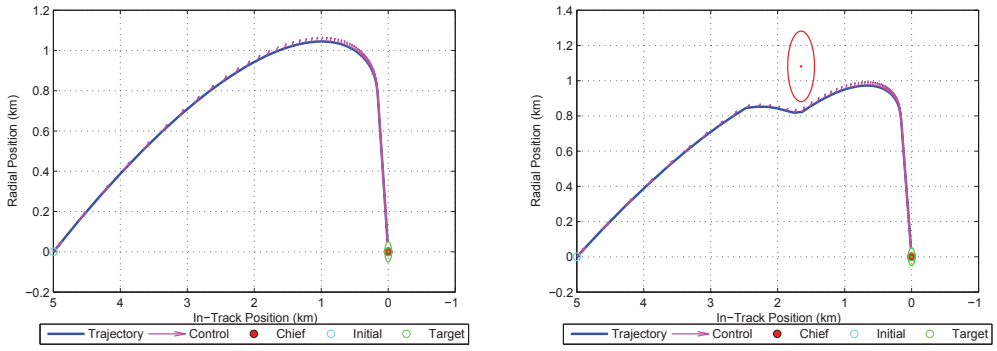


(a) (b)

Figure 4. (a) Radial Track ΔV for MPC With Long Horizon and One Obstacle (b) In-Track ΔV for MPC With Long Horizon and One Obstacle

It can be seen in Figure 3 that in the beginning of this maneuver, the spacecraft travels along the same path as in the previous simulations. When the obstacle appears, the spacecraft avoids it, but then returns to the previous nominal path so that it can utilize the relative motion dynamics and minimize the control effort. In Figure 4, the control is still well within the predefined limits.

The above simulations were performed with a large prediction horizon of 360 steps, which for MPC, introduces a great amount of computational complexity, thereby making it less appealing for on-board implementation. There are various options to reduce computation time, the easiest being to limit the number of steps in the horizon. This can be done in one of two ways: by increasing the discrete time step while keeping the number of steps in the horizon constant, or by reducing the number of steps in the horizon while keeping the discrete time step the same. The first of these methods may reduce the robustness of the algorithm because if an obstacle appears, the spacecraft must wait longer to react and change its maneuver. The second method shortens how far MPC can look into the future, which may degrade maneuver efficiency but maintains faster reaction time and is preferable from the standpoint of obstacle collision avoidance, and hence this method was chosen. Figure 5 shows the resulting trajectories if the horizon length was significantly shortened to 50 steps (6000 sec).



(a) (b)

Figure 5. (a) MPC Trajectory with Short Horizon and No Obstacle (b) MPC Trajectory with Short Horizon and One Obstacle

Table 1 shows the maneuver time (in hr), total control effort (in km/sec), and computational time of the full simulation (in sec) for each case. It should be noted that all simulations were performed using MATLAB, and that the computational time in real application is dependent upon the computational power available. These results were produced using a computer with an Intel i7 processor.

	Maneuver Time (hr)	Total ΔV (km/sec)	Computation Time (sec)
Long Horizon, No Obstacle	39.3000	0.33455	1764.5013
Long Horizon, One Obstacle	39.3667	0.33645	2441.9776
Short Horizon, No Obstacle	39.3000	0.33455	22.2595
Short Horizon, One Obstacle	39.3667	0.33645	23.3119

Table 1. Maneuver Time, Control Effort, and Computation Time Required for MPC Simulations

From the results presented, it can be seen that the trajectories, maneuver time, and control effort using long and short horizons are similar. While the results suggest the MPC with a shorter horizon is adequate, further study of the effects of constant/bounded disturbances and multiple obstacles will be pursued in the future to delineate the differences between the two horizon solutions.

Additionally, from the numerical results presented, adding an obstacle in the path of the spacecraft does increase the computational time, which is to be expected since the added obstacle constraint is active for a portion of this trajectory. Maneuver time and total ΔV are also greater, but it should be noted that the results are relativity close to the no obstacle case, showing that the MPC controller attempts to avoid the obstacle with little change to its nominal path. The addition of obstacles in the long horizon case is quite computationally expensive, since the added steps in the horizon and the obstacle constraint both increase the computation time. Note that with ΔT = 120 seconds, sufficient time to perform the computations is available even in the long horizon, one obstacle case. Overall, these results indicate that MPC is effective to guiding a spacecraft in a changing environment, with a $\Delta T = 120$ seconds.

EXTENDED COMMAND GOVERNOR WITH LINEAR QUADRATIC CONTROLLER

The ECG is an add on to an existing controller in order to enforce state and control constraints by modifying the set-point commands to the closed-loop system, as shown in Figure 6. In this system, $\hat{x}(k)$ is the state estimate*, $y(k)$ is the system output constrained by the set inclusion conditions $y(k) \in Y$ for all k, $w(k)$ is the disturbance/uncertainty, $v(k)$ is the ECG output, $u(k)$ is the controller output (in this paper, the LQ output), and $r(k)$ is the reference command.

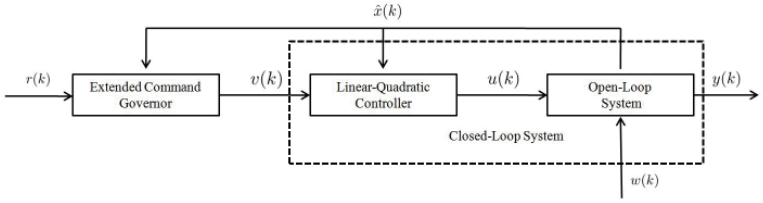


Figure 6. Extended Command Governor Applied to a Closed-Loop System

LQ Controller Formulation

The nominal controller for the spacecraft is of LQ type, designed on the basis of the discrete-time linear system model,

$$\begin{aligned} X(k+1) &= AX(k) + BU(k), \\ Y_t(k) &= C_t X(k) + D_t U(k), \\ Y_c(k) &= C_c X(k) + D_c U(k), \end{aligned} \tag{13}$$

where X is the state vector of spacecraft positions and velocities, U is the control vector of impulse velocities, Y_t is the output vector that tracks set-point spacecraft equilibrium positions, and Y_c is the output vector that constraints are imposed on. The optimal feedback gain K_{LQ} is determined by minimizing the cost function,

$$J = \sum_{n=1}^{\infty} (X(n)^T Q X(n) + U(n)^T R U(n)), \tag{14}$$

such that

$$U(n) = K_{LQ} X(n), \tag{15}$$

where $Q = Q^{\mathrm{T}} \geq 0$ and $R = R^{\mathrm{T}} > 0$ are the weight matrices on the state and the control. The controller is then augmented for set-point tracking,

$$U(k) = K_{LQ} X(k) + \Gamma v(k), \tag{16}$$

where

$$\Gamma = (C_t (I - A - B K_{LQ})^{-1} B + D_t)^{-1}, \tag{17}$$

such that for constant v, it is guaranteed that

$$Y_t(k) \to v \text{ as } k \to \infty. \tag{18}$$

*$\hat{x} = x$ in this paper and navigation errors will be considered in future work.

ECG Problem Formulation

The ECG uses the set, $\check{O}_\infty$, of states and parameterized inputs such that the constraints are satisfied for all time. The ECG output is defined by

$$v(k) = \rho(k) + \bar{C}\bar{x}(k), \tag{19}$$

where $\rho(k)$ and $\bar{x}(k)$ are solutions to the following optimization problem

$$\begin{array}{ll} \text{minimize} & ||\rho(k) - r(k)||_Q^2 + ||\bar{x}(k)||_P^2, \\ \text{subject to} & (\rho(k), \bar{x}(k), X(k)) \in \check{O}_\infty, \end{array} \tag{20}$$

The $\bar{x} \in \mathbf{R}^{\bar{n}}$, $\bar{n} \geq 0$, and ρ are states of a stable auxiliary dynamic system which evolves over the semi-infinite prediction horizon according to

$$\begin{array}{c} \bar{x}(k+j+1) = \bar{A}\bar{x}(k+j), j \geq 0, \\ \rho(k+j) = \rho(k). \end{array} \tag{21}$$

The set $\check{O}_\infty$ in Equation (20) is a finitely determined inner approximation to the set of all $(\rho(k), \bar{x}(k), X(k))$ that do not induce subsequent violation of the constraint $Y_c(k+j) \in Y$ when the input sequence $v(k+j)$ is determined by the fictitious dynamics per Equations (19) and (21). In the case when Y is polyhedral, computational procedures exist that lead to polyhedral $\check{O}_\infty$.[15] Without the fictitious states, i.e., when $\bar{n} = 0$, the ECG becomes the simple command governor.[16] The optimization problem can be solved numerically using conventional quadratic programming techniques. If the model and constraints do not change online, iterative procedures can be avoided by using explicit multi-parametric quadratic programming,[17] leading to ρ and $\bar{x}$ being given as a piecewise-affine function of the state $x(k)$ and reference r.

Various choices of $\bar{A}$ and $\bar{C}$ can be made in Equations (19) and (21). The shift sequences used in Reference 16 are generated by the fictitious dynamics with,

$$\begin{aligned} \bar{A} &= \begin{bmatrix} 0 & I_m & 0 & 0 & \cdots \\ 0 & 0 & I_m & 0 & \cdots \\ 0 & 0 & 0 & I_m & \cdots \\ 0 & 0 & 0 & 0 & \cdots \\ \vdots & \vdots & \vdots & \vdots & \ddots \end{bmatrix}, \\ \bar{C} &= \begin{bmatrix} I_m & 0 & 0 & 0 & \cdots \end{bmatrix}, \end{aligned} \tag{22}$$

where m is the dimensionality of v and I_m is an $m \times m$ identity matrix. Another approach uses Laguerre sequences.[18] These sequences possess orthogonality properties and are generated by the fictitious dynamics with

$$\begin{aligned} \bar{A} &= \begin{bmatrix} \epsilon I_m & \zeta I_m & -\epsilon\zeta I_m & \epsilon^2\zeta I_m & \cdots \\ 0 & \epsilon I_m & \zeta I_m & -\epsilon\zeta I_m & \cdots \\ 0 & 0 & \epsilon I_m & \zeta I_m & \cdots \\ 0 & 0 & 0 & \epsilon I_m & \cdots \\ \vdots & \vdots & \vdots & \vdots & \ddots \end{bmatrix}, \\ \bar{C} &= \sqrt{\zeta}\begin{bmatrix} I_m & -\epsilon I_m & \epsilon^2 I_m & -\epsilon^3 I_m & \cdots & (-\epsilon)^{N-1} I_m \end{bmatrix}, \end{aligned} \tag{23}$$

where $\zeta = 1 - \epsilon^2$ and $0 \leq \epsilon \leq 1$ is a selectable parameter that corresponds to the time-constant of the fictitious dynamics. Note that with the choice of $\epsilon = 0$, Equation (23) coincides with the shift register considered in Reference 16. The implementation of the ECG is feasible using partial state information and reduced order models following the approach in Reference 19.

Obstacle avoidance is achieved by extending the LQ/ECG combined scheme developed for the case of time-invariant control constraints to satisfy time-varying, obstacle-related constraints. In the MPC case, the obstacle avoidance constraint was treated by a rotating hyperplane approach, shown in Equation (12). This constraint can be rewritten in the general form,

$$H(k)Y_c \leq h(k). \tag{24}$$

Let,

$$\mathcal{X} = \begin{bmatrix} \bar{x} \\ \rho \\ X \end{bmatrix}, \tag{25}$$

which is driven from the following augmented dynamics,

$$\mathcal{X}(k+1) = \mathcal{A}\mathcal{X}(k), \tag{26}$$
$$Y_c(k) = \mathcal{C}\mathcal{X}(k), \tag{27}$$

where,

$$\mathcal{A} = \begin{bmatrix} \bar{A} & 0 & 0 \\ 0 & I & 0 \\ B\Gamma\bar{C} & B\Gamma & A + BK_{LQ} \end{bmatrix}, \tag{28}$$
$$\mathcal{C} = [D_c\Gamma\bar{C} \quad D_C\Gamma \quad C_c + D_cK_{LQ}].$$

To implement obstacle avoidance, the minimization in Equation (20) is performed at current time instant k subject to the additional constraint,

$$H(k)\mathcal{C}\mathcal{A}^j\mathcal{X} \leq h(k), \;\; j = 0, \ldots, n_c, \tag{29}$$

where n_c is a constraint horizon that is comparable to the settling time of the closed-loop system with the nominal LQ controller.

Since the constraint in Equation (29) adds another layer of complexity to the problem, the obstacle avoidance LQ/ECG controller is more computationally intensive. Therefore, control constraints are only enforced by the $\check{O}_\infty$ condition, while the avoidance constraint in Equation (29) is only active when the spacecraft senses and is within a certain distance of the obstacle. This approach reduces the average computing effort. Even though the processor is sized for the worst case computing effort, reducing average computing time is beneficial as it may reduce onboard electric power consumption.

Simulation Results

The simulations presented in this section have the same initial conditions as the MPC simulations. The spacecraft is displaced 5 km in the in-track direction and has the objective to rendezvous to an area with a 50 m radius around the chief satellite. As in the MPC case, $\Delta T = 120$ sec. The n_c was chosen to be 3 steps, or 360 sec. The Q and R matrices are set to $1 \times 10^{-3}[I_{6\times6}]$ and $2 \times 10^{3}[I_{3\times3}]$, which are the same weights when the discrete Riccati equation is solved for MPC. The same control constraint of 0.01 km/sec for ΔV is also used. Figures 7- 8 show results for spacecraft motion without an obstacle.

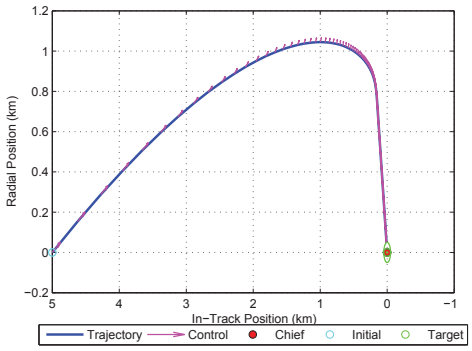
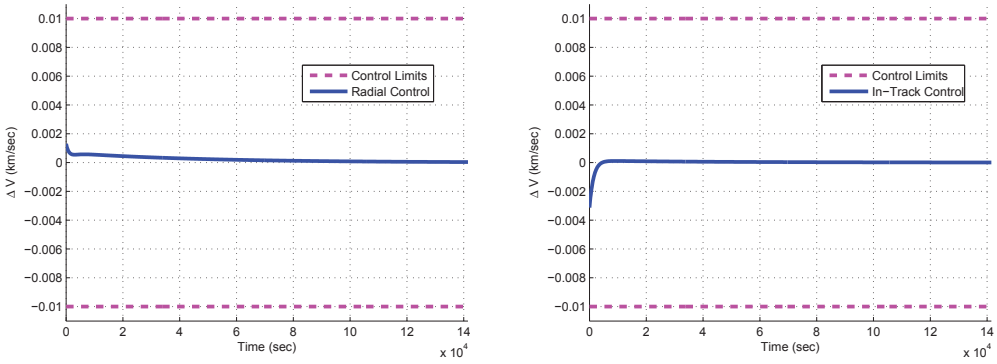


Figure 7. LQ/ECG Spacecraft Trajectory With No Obstacle



(a) (b)

Figure 8. (a) Radial ΔV for LQ/ECG With No Obstacle (b) In Track ΔV for LQ/ECG With No Obstacle

The trajectory in Figure 7 is a result of the LQ control exploiting the relative motion dynamics in order to conserve on control effort. This is due to choosing R large and putting more cost on control. As a consequence of using these weight matrices, the control is smooth and small, well within the control limits, as shown in Figure 8.

To demonstrate obstacle avoidance, a second simulation is performed where an obstacle of radius 0.2 km is placed midway in the spacecraft's trajectory. The results are shown in Figures 9-10.

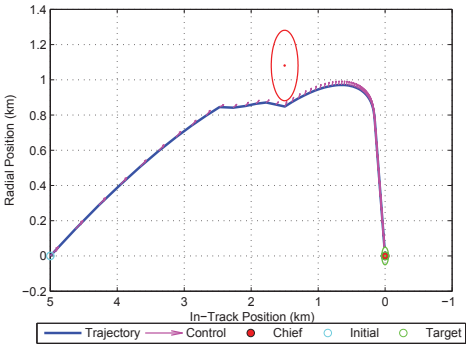
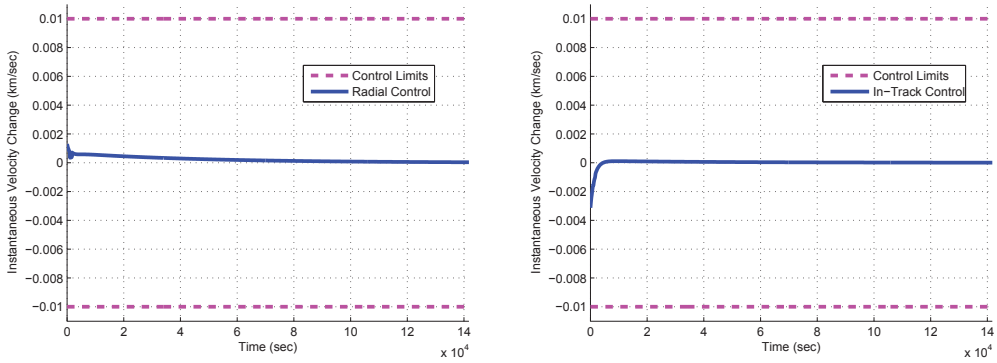


Figure 9. LQ/ECG Spacecraft Trajectory With One Obstacle



(a) (b)

Figure 10. (a) Radial ΔV for LQ/ECG With One Obstacle (b) In Track ΔV for LQ/ECG With One Obstacle

The trajectory for the obstacle avoidance case is shown to begin the same as the no obstacle case. Once the spacecraft is within range of the obstacle, the LQ/ECG controller exerts sufficient ΔV to avoid it. Afterwards, the controller commands a trajectory that uses the relative motion dynamics to rendezvous with the chief. The overall control of the spacecraft stays small and does not approach its limits, as shown in Figure 10. As a result of these simulations, it can be concluded that the LQ/ECG configuration can guide a spacecraft successfully to the chief satellite, even in presence of an obstacle.

The numerical results for both LQ/ECG simulations are shown in Table 2. The values for maneuver time and total ΔV in both simulations are close to one other, as well as to the MPC controller (Table 1). The computation time taken for the obstacle case was only slightly more than the no obstacle case, which is a consequence of the avoidance constraint being active for only a portion of

the trajectory. The computing time is significantly less in the LQ/ECG case when compared to the MPC case, however, a part of the benefit is attributed to near range obstacle constraint activation, which has not been implemented in the MPC.

	Maneuver Time (hr)	Total ΔV (km/sec)	Computation Time (sec)
No Obstacle	39.3000	0.33455	2.6371
One Obstacle	39.3667	0.33695	2.7389

Table 2. Maneuver Time, Control Effort, and Computation Time Required for LQ/ECG Simulations

ROBOT IMPLEMENTATION

In order to demonstrate the real-time, on-board implementation, a robotic test-bed was developed using available hardware. The robot parts for this project were from the LEGO Mindstorms NXT set and were used to build a holonomic "omnibot" robot. This robot uses three motors with omni-directional wheels offset at 120-degree angles (seen in Figure 11), allowing for uncoupled planar motion, which is particularly important to emulate spacecraft motion.

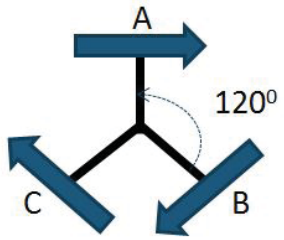


Figure 11. Geometry of Omnibot, Showing Forward Wheel Directions

To estimate the omnibot's position and provide feedback for the algorithm, a PhaseSpace inertial measurement system is used. This system combines a series of cameras and an LED controller to track specific LEDs and output their positions in inertial space. A series of algorithms were developed to distill this data into a usable form and generate states in the relative Hill's frame.

An inertial measurement unit (IMU) was also used to develop a dead-reckoning algorithm for when PhaseSpace data is lost. The IMU used was a VectorNav VN-100R IMU, which contains a 3-axis accelerometer and 3-axis gyroscope, and a MATLAB interface was developed to communicate with it using built-in serial functions. Since the motion modeled by the omnibot is planar, two dimensions of the acceleration and one dimension of rotation rate were used to propagate the IMU's state estimate in a dead-reckoning algorithm. This was done by using rotation rate data to track orientation estimates, transforming the body-frame acceleration to the inertial frame, and then integrating once for velocity and once more for position. Unfortunately, this method of state updating is well known to be inaccurate unless the precision of the IMU hardware is flawless, so there is considerable error propagation in this method. This is discussed further in the results below.

Omnibot Model

An important aspect of demonstrating the desired guidance algorithms on a robotic platform is being able to command the robot to a desired state. With the MPC and LQ/ECG method, it was necessary to ensure the ability to command a position and simulate the velocity of the vehicle between waypoints. With this in mind, a kinematic model was developed for the omnibot.

The omnibot motion is commanded by integer-value power settings between -100 and 100 for each motor, with negative values indicating rotation opposite of the direction indicated in Figure 11. The main insight that allowed for a kinematic model was that there was an observed, nearly linear correlation between the power input and rotation rate of the wheel. For the model, the power settings allowed were limited such that they were within this linear range. This lead to the overall conclusion that a given power setting implies a certain velocity for the robot. Another key insight in the model development was the velocity matching requirement. Since there is an equal change in time between each waypoint of the trajectories, it is desired to drive the robot toward each waypoint within a fixed time step, which implies a desired velocity matching requirement.

Given the above insights, an algorithm was developed to calculate motor power settings for the omnibot, given a desired displacement and fixed time step. When traveling from one waypoint to another, the displacement magnitudes, dx for radial and dy for in-track, and defined time step, dt, imply the velocities in the x and y direction that the omnibot must match. The desired velocity can then be converted to a desired power in the x and y directions, P_x and P_y respectively. Using the geometry of the omnibot as shown in Figure 11, the following power setting equations were developed,

$$P_x = P_A - P_B \cos(\pi/6) - P_C \cos(\pi/6), \tag{30}$$

$$P_y = -P_B \sin(\pi/6) + P_C \sin(\pi/6), \tag{31}$$

where P_A, P_B, and P_C are the power commands to motors A, B, and C, respectively (as designated in Figure 11). An additional constraint was placed on the robot to ensure that it maintained a zero rotation rate about the center of mass,

$$0 = P_A + P_B + P_C. \tag{32}$$

With P_x and P_y prescribed by the desired trajectory, the system of three-equations and three-unknowns is trivial to solve. Once the wheel power commands are calculated, they are rounded to satisfy the robot's integer-valued, power inputs requirement. After these power settings are calculated, the robot is guided to the waypoint by slightly modifying the motor powers to adjust the direction of displacement. By preserving the 2-norm of the powers and modifying the unit-vectors slightly, the robot maintains a given velocity and drives toward a waypoint with feedback control.

The model developed in this section was tested and validated experimentally using the PhaseSpace system to track the true position and provide feedback to adjust the motors. The model's performance to command was accurate within the domain of motor powers, where the correlation between motor power and wheel rotation rate is roughly linear.

Results

With a robot that effectively drives to a given waypoint in a fixed amount of time, the MPC and LQ/ECG controllers could finally be tested and demonstrated with hardware. The figures in this section show the results of testing the MPC method on the omnibot.

Figure 12 shows the performance of the omnibot using the MPC algorithm without any obstacle. It can be seen from this result that the IMU data cannot be trusted without frequent reinforcement from the PhaseSpace system. This is partially due to the manner in which IMU data is collected. Since both the robot control and data collection (PhaseSpace and IMU) are run from a single MATLAB script, the program is gathering data from the IMU once every loop iteration. If the IMU data could be collected and averaged more frequently, the results from the dead-reckoning algorithm should improve. This could be accomplished by having the IMU data algorithm run separately from the robot control script. From a hardware perspective, one possible solution is to use a Raspberry Pi to collect data and run the dead-reckoning algorithm to track the IMU state estimate. The main control algorithm could query the Raspberry Pi over wi-fi for the most recent state estimate, similar to how the control algorithm currently queries the PhaseSpace server for the most recent position estimate.

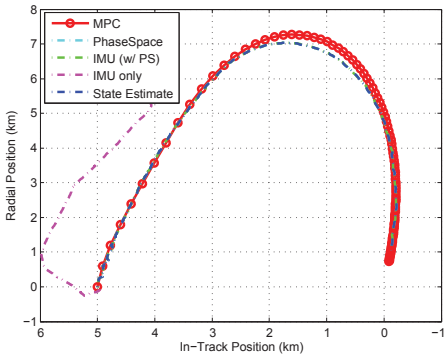


Figure 12. MPC Implementation on Omnibot

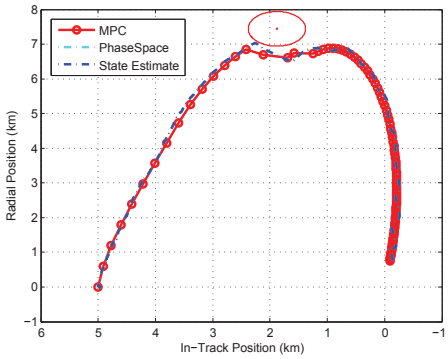


Figure 13. MPC Implementation on Omnibot With One Obstacle

One of the primary advantages of the MPC algorithm for optimal trajectory is its robustness to a suddenly appearing obstacle. Figure 13 shows the performance of the omnibot with a appearing halfway through the trajectory. The experimental data shows that MPC is able to compute a new trajectory, resulting in the omnibot's avoidance of the obstacle.

An interesting side effect of the hardware used is the addition of random noise. Since the robot is not made for precision maneuvers, there are variable time lags associated with commanding the robot. This results in different trajectories despite the same initial conditions. However, it was observed that even with the added noise, MPC is still able to maneuver around the obstacle and rendezvous with the chief.

The results from this section show that this robotic test-bed can be used to effectively demonstrate the MPC algorithm and visualize the simulation results, as well as demonstrate the robustness of the MPC algorithm to random noise brought on by the inaccuracies of the hardware. In the future, the LQ/ECG will be tested under the same conditions and compared to the MPC results. Also, it is desired to perform tests using more accurate hardware such that precision maneuvers are simulated accurately.

CONCLUSION

Autonomy in spacecraft guidance and control has become an increasingly important subject as the space environment becomes more populated and mission objectives require more precision and robustness. This paper explores two methods of fast on-board autonomous spacecraft guidance and control based on Model Predictive Control and a Linear Quadratic controller augmented by an Extended Command Governor. Through simulations and robotic, test-bed experiments in the MPC controller case, each controller has been shown to successfully rendezvous with a chief satellite and avoid suddenly appearing obstacles. Both methods result in similar trajectories, but the LQ/ECG configuration is less complex and more computationally efficient due to its add-on nature. This new innovative method for obstacle avoidance is attractive for on-board implementation. MPC in contrast is a more conventional controller, and while more computationally taxing, can produce a more efficient solution than the LQ/ECG if a relatively long horizon is used. By shortening the prediction and control horizon, it was also shown that MPC's computational requirement can be reduced. It is desired in future works to address disturbances and multiple obstacles with both methods.

ACKNOWLEDGMENTS

The authors would like to thank AFRL Scholars Program and the AFRL Guidance, Navigation and Control group for providing the opportunity to implement the algorithms with hardware, Uros Kalabic of University of Michigan for providing ECG design software, James Svacha of Clemson University for aiding in the robot implementation, and Jacob Darling and Dr. Kyle DeMars of Missouri University of Science and Technology for the use of their IMU diagnostic algorithms in characterizing and testing the IMU dead-reckoning algorithm. The last author (Ilya Kolmanovsky) would like to acknowledge the support of the National Science Foundation under Award 1130160.

REFERENCES

[1] "Satellite Collision Leaves Significant Debris Clouds," *Orbital Debris Quarterly News*, Vol. 13, No. 2, 2009, pp. 1–2.

[2] L. Breger and J. How, "Safe Trajectories for Autonomous Rendezvous of Spacecraft," *Journal of Guidance, Control, and Dynamics*, Vol. 31, No. 5, 2008, pp. 1478–1489.

[3] R. Farrenkopf, "Optimal Open Loop Maneuver Profiles for Flexible Spacecraft," *Proceedings of Guidance and Control Conference*, 1978.

[4] J. L. Junkins, M. R. Akella, and R. D. Robinett, "Nonlinear Adaptive Control of Spacecraft Maneuvers," *Proceedings of Guidance, Navigation, and Control Conference*, 1997.

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

[6] A. Weiss, I. Kolmanovsky, M. Baldwin, and R. S. Erwin, "Model Predictive Control of Three Dimensional Spacecraft Relative Motion," *Proceedings of American Control Conference*, 2012.

[7] H. Park, S. D. Cairano, and I. Kolmanovsky, "Linear quadratic model predictive control approach to spacecraft rendezvous and docking," *Proceedings of 21st AAS/AIAA Space Flight Mechanics Meeting, Spaceflight Mechanics, Part III of Advances in the Astronautical Sciences*, Vol. 140, February, 2011.

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

[9] M. Saponara, V. Barrena, A. Bemporad, E. Hatley, J. Maciejowski, A. Richards, A. Tramutola, and P. Trodden, "Model Predictive Control application to spacecraft rendezvous in Mars return scenario," *Proceedings of 4th European Conference for Aerospace Sciences (EUCASS)*, St. Petersburg, Russia, July, 2011.

[10] L. Breger, J. How, and A. Richards, "Model Predictive Control of Spacecraft Formations with Sensing Noise," *Automatica*, Vol. 45, No. 10, 2009, pp. 2412–2418.

[11] U. Kalabic, E. Gilbert, and I. Kolmanovsky, "Reference governors for linear systems with nonlinear constraints," *Proceedings of 50th IEEE Conference on Decision and Control*, 2011, pp. 2680–2686.

[12] E. Gilbert and C. Ong, "An Extended Command Governor for Constrained Linear Systems with Disturbances," *Proceedings of Joint 48th IEEE Conference on Decision and Control and 28th Chinese Control Conference*, 2009.

[13] R. Clohessy and R. Wiltshire, "Terminal Guidance System for Satellite Rendezvous," *J. Aerospace Sci.*, Vol. 27, No. 9, 1960, pp. 653–658.

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

[15] I. Kolmanovsky, U. Kalabic, and E. Gilbert, "Developments in constrained control using reference governors," *Proceedings of the IFAC Conference on Nonlinear Model Predictive Control (NMPC)*, 2012, pp. 282–290.

[16] E. Gilbert and C. Ong, "Constrained linear systems with hard constraints and disturbances: An extended command governor with large domain of attraction," *Automatica*, Vol. 47, 2011, pp. 334–340.

[17] A. Bemporad, M. Morari, V. Dua, and Pistikopoulos, "The explicit solution of Model Predictive Control via multiparametric quadratic programming," *Automatica*, Vol. 38, No. 3, 2002.

[18] U. Kalabic, I. Kolmanovsky, J. Buckland, and E. Gilbert, "Reference and extended command governors for control of turbocharged gasoline engines based on linear models," *Proceedings of IEEE Multi-Conference on Systems and Control*, 2011, pp. 319–325.

[19] U. Kalabic, E. Gilbert, and I. Kolmanovsky, "Reduced order extended command governor," *Automatica, in press*.