A model predictive control approach for a thrust vector controlled sounding rocket

Tom Lijding 6318037, Wesley Nijhuis 4965590,

***Abstract*—We study a model predictive control (MPC) approach for a thrust vector controlled sounding rocket. Three approaches are applied: regulation, reference tracking and disturbance rejection and finally trajectory tracking. The MPC was found to perform better than a typical LQR controller on all tasks.**

I. Introduction

The project is largely based on [1]. The reader is encouraged to peruse the paper themselves to see (slightly more) in depth derivations of the dynamics of the system, as well as see the more simplified LQR and PID control schemes used. All relevant code for this project is given in a Github repository in [2].

For the purpose of this project, a thrust vectored rocket will be discussed with a single gimbal control. In the following three chapters, the rocket is analyzed around an equilibrium point. First the rocket will be controlled to an equilibrium point, this corresponding to the rocket floating in mid-air. In this setup, all states are assumed to be usable for control. Next, the rocket will be controlled via an output-feedback scheme, where 6 out of the 9 states can be observed. Additionally, a disturbance is estimated using Luenberger observer. In chapter IV and V, a trajectory planning and tracking MPC is built that, aside from optimized trajectories, can track arbitrary trajectories. For this a state feedback MPC is for tracking.

The report is structured into six parts. First, in the introduction we introduce the model used for the rocket, as well as around which points the rocket is linearized. Next, the MPC problem is formulated for the three cases in Section II: MPC Formulation. Next, the regulated MPC rocket is shown to be asymptotically stable in Section III: Asymptotic Stability. Then, in chapter IV and V a trajectory planning and tracking MPC is proposed. Finally, we show numerical simulations of all systems in Section VI, as well as the effect of choices such as discretization time. weighting matrices and soft or hard constraints.

A. Model

1) Assumptions

- The rocket is assumed to be axially symmetric (in the x and y axis)
- The Earth is assumed to be flat

2) Reference frame transformations

For the definition of our dynamics, it is first required to define two reference frames, namely an inertial reference frame and the body-fixed reference frames as within [1].

It is important to also define the coordinate transformations between these reference frames. For this, we use the Euler angle notation

$$\lambda = \begin{bmatrix} \phi & \theta & \psi \end{bmatrix}^T$$

where ϕ, θ, ψ denote the roll, pitch and yaw angles respectively. Using this, define then the transformation from the body reference frame to the intertial reference frame as $\mathbf{R}(\lambda) = \mathbf{R}_y\psi \cdot \mathbf{R}_y\theta \cdot \mathbf{R}_x\phi$ where $\mathbf{R}_y\psi$ denotes a rotation matrix in 3D around the z-axis of ψ degrees.

Note that c_θ and s_θ denote $\cos(\theta)$ and $\sin(\theta)$ respectively. Further on, the notation t_θ will denotes $\tan(\theta)$.

3) Dynamics and kinematics

From the above reference frame and from the Newton-Euler equations for rigid body dynamics, the 6DOF equations of motion are given by

$$\begin{cases} \dot{\mathbf{p}} = \mathbf{R}(\lambda)\mathbf{v} \\ \dot{\mathbf{R}}(\lambda) = \mathbf{R}(\lambda)\mathbf{S}(\omega) \\ m\dot{\mathbf{v}} = -\mathbf{S}(\omega)m\mathbf{v} + \mathbf{f} \\ \mathbf{J}\dot{\omega} = -\mathbf{S}(\omega)\mathbf{J}\omega + \tau \end{cases} \tag{1}$$

where $\mathbf{p} \in \mathbb{R}^3, \mathbf{v} \in \mathbb{R}^3, m \in \mathbb{R}, \mathbf{J} \in \mathbb{R}^{3\times3}, \mathbf{f} \in \mathbb{R}^3, \tau \in \mathbb{R}^3, \omega \in \mathbb{R}^3, \mathbf{S}(\omega) \in \mathbb{R}^{3\times3}$ denotes the position in the intertial frame of reference, the velocity in the body frame of reference, the inertia matrix, the forces and the torques acting on the body in the body frame of references, the angular velocity and the skew-symmetric acceleration matrix respectively. The angular vectors are further defined below.

$$\mathbf{p} = \begin{bmatrix} x_i & y_i & z_i \end{bmatrix}^T, \quad \mathbf{v} = \begin{bmatrix} u & v & w \end{bmatrix}^T, \quad \omega = \begin{bmatrix} p & q & r \end{bmatrix}^T$$

4) External forces and torques

The forces can then be expressed as in [1] and the reader is encouraged to peruse all derivations as there with two exceptions. The first being that we assume no aerodynamic force on the rocket. This being the case as we linearize around relatively low velocity points. Secondly, we add a roll control via a flywheel. How this translates into state space is given Appendix A

B. Complete Model (No Aerodynamic Force)

By substituting the equations of the forces as in [1] as well as Equation 25 in 1 we arrive at a complete nonlinear model as in [1]. As stated before, we remove the aerodynamic forces on the model. The model for this case is then given in Equation 2.

Tom Lijding 6318036 and Wesley Nijhuis 4965590 are master students at TU Delft, The Netherlands. E-mail addresses: {t.lijding, w.p.a.nijhuis}@student.tudelft.nl.

$$\begin{cases} \dot{u} = -gc_\theta c_\psi - \frac{T}{m} c_{\mu_1} c_{\mu_2} - qw + rv \\ \dot{v} = -g\left(s_\phi s_\theta c_\psi - c_\phi s_\psi\right) - \frac{T}{m} c_{\mu_1} s_{\mu_2} - ru + pw \\ \dot{w} = -g\left(c_\phi s_\theta c_\psi + s_\phi s_\psi\right) - \frac{T}{m} s_{\mu_1} - pv + qu \\ \dot{p} = J_l^{-1}\left(\tau_r\right) \\ \dot{q} = J_t^{-1}\left(-T s_{\mu_1} l\right) \\ \dot{r} = J_t^{-1}\left(T c_{\mu_1} s_{\mu_2} l\right) \\ \dot{\phi} = p + \left(q s_\phi + r c_\phi\right) t_\theta \\ \dot{\theta} = q c_\phi - r s_\phi \\ \dot{\psi} = \frac{q s_\phi + r c_\phi}{c_\theta} \end{cases} \tag{2}$$

C. Parameter selection

In defining the model, all described constants must be chosen to have realistic values. We take inspiration from [1] where possible and from [3] as well. The chosen parameters are listed in Appendix C in Table I.

D. Linearization around a Floating Point

In model predictive control finding a stabilizing nonlinear controller is a nontrivial task. This task may be simplified by linearizing the system around an equilibrium point, and assuming that the dynamics of the system are suitably close to these linear dynamics. We take this approach as well. The dynamics of the rocket are linearized at a point above the ground, (note that we defined this point in the x,y,z axes as (0,0,0) velocity in the inertial frame of reference), with a constant thrust of $m \cdot g$. The full linearization point is described by

$$\begin{bmatrix} u & v & w & p & q & r & \phi & \theta & \psi \end{bmatrix}^T = \begin{bmatrix} 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \end{bmatrix}^T \\ \begin{bmatrix} \mu_1 & \mu_2 & \tau_r & T \end{bmatrix}^T = \begin{bmatrix} 0 & 0 & 0 & m \cdot g \end{bmatrix}^T \tag{3}$$

which gives symbolic dynamics. We refer the reader to the appendix for the full linear dynamics in matrix form F, as the dynamics are too large to show here.

II. MPC FORMULATION

Based on the derived dynamics in Section I, three MPC formulations were tackled.

1) Regulation MPC, for full state feedback
2) Output feedback MPC, where an observer is integrated into the design to estimate the states of the system and any disturbances
3) Adaptive MPC, for the time varying trajectory

The general formulation of the (linear) MPC problem is given by

$$\min_{\mathbf{u}_N} V_N(x_0, \mathbf{u}_N) = \min_{\mathbf{u}_N} \sum_{k=0}^{N-1} \ell(x(k), u(k)) + V_f(x(N)) \tag{4}$$

such that

$$x(k+1) = Ax(k) + Bu(k) \quad \forall k = \{0, 1, ..., N-1\}, \\ u \in \mathbb{U}, x \in \mathbb{X} \tag{5}$$

where $V_N(x_0, \mathbf{u}_N)$ denotes the cost function from state x_0 until time-step N, $\ell(x(k), u(k))$ denotes the stage cost at time-step k and $V_f(x(N))$ denotes the terminal cost for the state at $x(N)$ at time-step N and $\mathbb{X}, \mathbb{U}$ denote the set of all feasible states and inputs respectively.

To solve the MPC problem the above cost function is minimized at each timestep for a given stage cost $\ell(x(k), u(k))$, terminal cost $V(N)$ and prediction horizon N.

A. Regulation MPC

From the linearization in Section I-D the following state space equation is derived:

$$\dot{x}(k) = Ax(k) + Bu(k) \\ y(k) = Cx(k)$$

Where $C = I_n$.

The stage cost and terminal cost can than be chosen as Linear Quadratic cost functions with $P, Q \succeq 0$, $R \succ 0$.

$$\ell(x(k), u(k)) = x(k)^T Q x(k) + u(k)^T R u(k) \tag{6}$$

$$V_f(x(N)) = x(N)^T P x(N) \tag{7}$$

Here Q and R are a design choice and should be tuned to achieve desired system behavior and P is the solution to the DARE. Now depending on the choice of definition of terminal set (we refer to Section III for more information) either an extra constraint must be introduced to ensure the MPC ends in the terminal set, i.e.

$$x(N) \in \mathbb{X}_f$$

or the terminal cost must be adjusted to

$$V_f(x(N)) = \beta \cdot x(N)^T P x(N)$$

We place constraints on the input and the states as follows. The inputs are bounded by physical limits, namely the maximum angles that the gimbals can take, non-negative and maximum thrust, as well as a maximal torque corresponding to a flywheel of around 10 kg with a 20 cm diameter.

Furthermore, constraints are placed upon the states, such that the rockets nonlinear dynamics stay sufficiently close to our linear operating range.

Summarized:

$$|u_1| \leq u_{1,lim} = 0.15708 \tag{8}$$
$$|u_2| \leq u_{2,lim} = 0.15708 \tag{9}$$
$$|u_3| \leq u_{3,lim} = 1.66 \tag{10}$$
$$u_4 \leq u_{4,lim} = 140.93 \tag{11}$$
$$u_4 \geq u_{4,lim} = -m \cdot g \tag{12}$$
$$|x_i| \leq x_{i,lim} = 10 \text{ for } i = 1, 2, 3 \tag{13}$$
$$|x_j| \leq x_{j,lim} = 0.174533 \text{ for } j = 4, 5, 6 \tag{14}$$
$$|x_k| \leq x_{k,lim} = 0.261799 \text{ for } k = 8, 9, 10 \tag{15}$$
$$\tag{16}$$

Additionally, as the state constraints are self-imposed for linearity reasons, we found that softening these constraints allowed the MPC to perform in a wider set of states. How one may introduce slack variables in the constraints is shown in Appendix B.

We furthermore, discretize the system using the ZOH hold method and a sampling time of 0.1 seconds. We refer to [4] for further discussion on sampling time and discretization.

Furthermore, in Section VI numerical simulations are run showing results for differing sampling times. 0.1 seconds is found to be a good sampling time.

B. Output, Reference Tracking and Disturbance Rejection MPC

In this section, we discuss the creation of an output and reference tracking MPC controller. In previous sections we have assumed full state feedback or in other words

$$C = I$$

In actual fact, most real systems do not have observable states and their states must be observed via an observer. Through the properties of the separation principle, in which the eigenvalues of an observer and a controller can be completely separately placed, it is possible to design an observer of the system whose eigenvalues of the error dynamics are stable. By placing these eigenvalues sufficiently fast, we ensure that after convergence to zero error, only the controller dynamics take over.

Additionally, we have assumed no other disturbances up until now work on our system. In reality, we have wind disturbances acting on our system.

Formally we can define such a system as follows

$$\begin{aligned} x^+ &= Ax + Bu + B_d \\ y &= Cx + C_d d \end{aligned} \tag{17}$$

Now, it is important to define B_d. We define in our case a disturbance acting on the angular velocity around the x-axis, on p. Connecting this to reality, one may see this as a miscalibration of the flywheel, or misbalanced wind spinning the rocket around, or perhaps slightly misproduced geometry. The disturbance input matrix is then defined as

$$B_d = \begin{bmatrix} 0 & 0 & 0 & J_t^{-1} & 0 & 0 & 0 & 0 & 0 \end{bmatrix}^T$$

C. Sensor suite

In [1] it is assumed all states can be measured (through an advanced estimator architecture). We assume contrary to this that not all states can be measured, and mainly that angular velocity cannot be measured directly. For this we use a Leuenberger observer and C is then defined as

$$C = \begin{bmatrix} I_3 & 0 & 0 \\ 0 & 0 & I_3 \end{bmatrix}$$

The observer dynamics are then given by

$$\begin{bmatrix} \hat{x}^+ \\ \hat{d}^+ \end{bmatrix} = \underbrace{\begin{bmatrix} A & B_\mathrm{d} \\ 0 & I \end{bmatrix}}_{\tilde{A}} \begin{bmatrix} \hat{x} \\ \hat{d} \end{bmatrix} + \begin{bmatrix} B \\ 0 \end{bmatrix} u + \underbrace{\begin{bmatrix} L_1 \\ L_2 \end{bmatrix}}_{\tilde{L}} \left(y - \underbrace{\begin{bmatrix} C & C_\mathrm{d} \end{bmatrix}}_{\tilde{C}} \begin{bmatrix} \hat{x} \\ \hat{d} \end{bmatrix} \right)$$

where L denotes the observer gain matrix, $\hat{d}$ denotes the estimed disturbance, and $\hat{x}$ denotes the estimated system states. We set the poles of the observer error dynamics, given by

$$\tilde{A} - \tilde{L}\tilde{C}$$

iteratively, but such that the eigenvalues of the above matrix are within the unit circle. Setting these too quick led to aggressive and wildly inaccurate estimates, while too low lead to unstable dynamics. Eventually a good balance was found to be around $p = 0.6 - 0.8$. In implementing the observer system, the system is required to be observable. This can be easily verified by checking if the observability matrix is full rank, in addition to

$$\operatorname{rank} \begin{bmatrix} I - A & -B \\ C & 0 \end{bmatrix} = n + n_d = 9 + 1$$

which both are confirmed to be.

D. OTS

Finally, in order to track a reference, the following problem was solved at each time step

$$(x_\mathrm{ref}, u_\mathrm{ref}) = \begin{cases} \underset{x_\mathrm{r}, u_\mathrm{r}}{\operatorname{argmin}} J(x_\mathrm{r}, u_\mathrm{r}) = \ell(x_r, u_r) \\ \text{s.t.} \begin{bmatrix} I - A & -B \\ C & 0 \end{bmatrix} \begin{bmatrix} x_\mathrm{r} \\ u_\mathrm{r} \end{bmatrix} = \begin{bmatrix} 0 \\ y_\mathrm{ref} - \hat{d} \end{bmatrix} \\ (x_\mathrm{r}, u_\mathrm{r}) \in \mathbb{Z} \end{cases}$$

where $\hat{d}$ is the current best estimate of d from the observer. Note that we assume a constant disturbance on the state.

III. Asymptotic Stability

In this section the MPC controller is shown to asymptotically stabilize the linearized system given the initial conditions are within the region of attraction $\mathbf{\Gamma}_\mathbf{N}^\beta$.

We make a number of (standard) choices, and prove afterwards why given these choices the MPC is asymptotically stable, as well as some caveats. As stated before, we choose $\ell(x(k), u(k))$ and $V_f(x(N))$ as in Equations 6 and 7 to begin.

Now, we are left with three well-described choices for a terminal set $\mathbb{X}_f$ as in Chapter 2 of [5]. The three choices of (explicit or implicit) terminal sets are either, the maximal constraint admissible set for LQR control, a sublevel set of $V_f(x(N))$ defined by $W(c) := \{x \mid x^T P x \leq c\}$ such that all states give feasible LQR control inputs, or implicitly defining the terminal set by defining a new $V_f(\cdot) = \beta x^T P x$ as described in Section 2.6 of [5] with $\beta \geq 1$.

The maximal constraint admissible set was found to be a set of 50 linear constraints, defining a convex polytope. The large amount of constraints lead to difficult and computationally expensive implementation in the MPC. The quadratic terminal set $x^T P x \leq c$ was found to be small $c = 0.1973$ leading to a suboptimal (and often infeasible) MPC. For this reason, we defined the terminal set implicitly.

By adjusting the terminal cost to $V_f(\cdot) = \beta x^T P x$, we implicitly define a new region of attraction $\Gamma_N^\beta := \{x \mid \hat{V}_N^\beta(x) \leq Nd + \beta c\}$ which is asymptotically stable where $\hat{V}_N^\beta(x)$ is the value function of the optimal solution to the MPC problem for $V_f(x) = \beta \cdot x^T P x$ and d is a lower bound on $\ell(x, u)$ for $x \in \mathbb{X} \setminus \mathbb{X}_f$ and $u \in \mathbb{U}$. The advantages of this are multiple. First, MPC problems are easier to solve with exclusively (hard) input constraints [5]. Secondly, including the terminal set explicitly penalizes optimality as it forces us to be within a set that is smaller than the maximal constraint admissible set for any stabilizing input.

Our goal is then to first prove Theorem 2.19 from Section 2.4 of [5], on the asymptotic stability of linear model predictive control for the maximal constraint admissible set for LQR

control. Following, this we quickly run through Lemma 2.40 and Theorem 2.4.1 of Section 2.6 to prove asymptotic stability for no terminal constraint. For Theorem 2.19 to hold, we must prove the following four assumptions:

1) Assumption 2.17 (Weak controllability). There exists a $\mathcal{K}_\infty$ function $\alpha(\cdot)$ such that
$$V_N^0(x) \leq \alpha(|x|) \quad \forall x \in \mathcal{X}_N$$
2) Assumption 2.2 (Continuity of system and cost). The functions $f : \mathbb{Z} \to \mathbb{X}, \ell : \mathbb{Z} \to \mathbb{R}_{\geq 0}$ and $V_f : \mathbb{X}_f \to \mathbb{R}_{\geq 0}$ are continuous, $f(0,0) = 0$, $\ell(0,0) = 0$ and $V_f(0) = 0$.
3) Assumption 2.3 (Properties of constraint sets). The set $\mathbb{Z}$ is closed and the set $\mathbb{X}_f \subseteq \mathbb{X}$ is compact. Each set contains the origin. If $\mathbb{U}$ is bounded (hence compact), the set $\mathbb{U}(x)$ is compact for all $x \in \mathbb{X}$. If $\mathbb{U}$ is unbounded, the function $\mathbf{u} \mapsto V_N(x, \mathbf{u})$ is coercive, i.e., $V_N(x, \mathbf{u}) \to \infty$ as $|\mathbf{u}| \to \infty$ for all $x \in \mathbb{X}$).
4) Assumption 2.14 (Basic stability assumption). $V_f(\cdot), \mathbb{X}_f$ and $\ell(\cdot)$ have the following properties: (a) For all $x \in \mathbb{X}_f$, there exists a u (such that $(x, u) \in \mathbb{Z}$) satisfying
$$f(x,u) \in \mathbb{X}_f$$
$$V_f(f(x,u)) - V_f(x) \leq -\ell(x,u)$$
(b) There exist $\mathcal{K}_\infty$ functions $\alpha_1(\cdot)$ and $\alpha_f(\cdot)$ satisfying
$$\ell(x,u) \geq \alpha_1(|x|) \forall x \in \mathcal{X}_N, \forall u \text{ such that } (x,u) \in \mathbb{Z}$$
$$V_f(x) \leq \alpha_f(|x|) \forall x \in \mathbb{X}_f$$

Assumption 2.17 can be proven by checking the controllability of the system, as this is a stronger notion of controllability and implies weak controllability [5]. This is done by checking the rank controllability matrix which is full rank, and thus the system is controllable. Next, Assumption 2.2 is proven by inspection of our functions. f is a linear function, and furthermore ℓ and $V_f(\cdot)$ are positive (semi-)definite and continuous, since they are quadratic functions of positive semi-definite matrices. Finally $f(0,0) = 0$, $\ell(0,0) = 0$ and $V_f(0) = 0$.

Assumption 2.3 is satisfied since $\mathbb{Z}$ is closed, the terminal set $\mathbb{X}_f$ is compact and contains the origin and $\mathbb{U}$ is compact. $\mathbb{U}$ is unbounded, but $V_N(x,u)$ is coercive $\forall x \in \mathbb{X}_f$ since R is positive definite.

Assumption 2.14 holds since we have chosen $V_f(\cdot) = x^T P x$ where P satisfies the discrete algebraic Riccati equation $P = A_K^T P A_K + Q + K^T R K$, where $u = Kx$. Then, it follows that

$$V_f(A_K x) + x^T (Q + K^T R K) x - V_f(x) \leq 0 \forall x \in \mathbb{R}^n$$

Notice that $\ell(x,u) = x^T(Q + K^T R K)x$ for $u = Kx$. Furthermore, we have chosen $\mathbb{X}_f$ exactly such that it is positive control invariant, namely first by calculating the exact maximally control invariant set under $u = Kx$, then via a sublevel set of $x^T P x$ for which the optimal LQR control law is fesaible thus satisfying a). Furthermore, since Q, R and P are all positive definite, $\ell(\cdot)$ and $V_f(\cdot)$ are themselves $\mathcal{K}_\infty$ functions globally and thus b) is satisfied.

Since $P = A_K^T P A_K + Q + K^T R K \implies \beta \cdot P \geq \beta A_K^T P A_K + Q + K^T R K$ Assumption 2.2 holds, and Assumption 2.3 trivially holds. Then from Theorem 2.41 from Section 2.6 in [5] it follows that since $\beta \cdot V_f(\cdot)$ and $\mathbb{X}_f = W(c)$ satisfy Assumptions 2.2 and 2.3 that the origin is asymptotically or exponentially stable for $x^+ = f(x, \kappa_N^\beta(x))$ with region of attraction Γ_N^β and that Γ_N^β is positive invariant for $x^+ = f(x, \kappa_N^\beta(x))$.

Intuitively, we say that the LQR control law stabilizes the system, and thus as long as we end up in a region where all states lead to feasible LQR control (and stay within the region), we can control the system.

A. Finding Γ_N^β

Finally, just to precisely define the region of attraction, we could optimize

$$d = \min_x \ell(x,0) = x^T Q x \quad s.t. \quad x^T P x \geq c$$

(since R is positive definite, $u = 0$ is always optimal), however this problem is nonconvex and thus difficult to solve. We therefore settle for knowing that such a bound exists.

IV. Trajectory planning for landing a rocket

The problem of landing a rocket using thrust vectored control requires a model that captures the rocket dynamics and allows for the evaluation of the rocket's position. For this the previous model needs to be expanded to incorporate the rocket's position

To augment the state space with the position of the rocket in the inertial frame, the rotation matrix $\mathbf{R}(\lambda)$ is used to derive the velocities $\dot{x}_I, \dot{y}_I, \dot{z}_I$ in the inertial frame. the extended states then became:

$$\begin{cases} \dot{x}_I = c_\theta c_\psi u + (s_\psi s_\theta c_\phi - c_\phi s_\psi) v + (c_\phi s_\theta c_\psi + s_\phi s_\psi) w \\ \dot{y}_I = c_\theta s_\psi u + (s_\phi s_\theta s_\psi + c_\phi c_\psi) v + (c_\phi s_\theta s_\psi - s_\phi c_\psi) w \\ \dot{z}_I = -s_\theta u + s_\phi c_\theta v + c_\phi c_\theta w \end{cases} \tag{18}$$

A. Design concerns for a nonlinear MPC controller

Two methods for implementing a trajectory optimizing were considered. The first one is the iterative LQR method. This method uses dynamic programming. In short, the iLQR method optimized a quadratically approximated version of the nonlinear trajectory optimization problem for all stages. This method can adjust the planned trajectory to external disturbances, but requires solving the nonlinear optimization problem at each timestep.

The second method that is proposed here, is a modified version of iLQR. To investigate the possibility of tracking an arbitrary trajectory $\mathbf{x}_t$ for which $x_t(k+1) = \{\phi(k; x_t(k), u_t(k)) | x_t(k) \in \mathbb{X}, u_t(k) \in \mathbb{U}\}$, the trajectory optimization problem was split into two parts: an offline trajectory planning problem and an online trajectory tracking problem. This approach has the benefit of not having to recompute the trajectory at every timestep, which could save computation costs especially if the planning horizon M is large (which is interesting for the use case of a sounding rocket). The trajectory tracking is done by linearizing the system at each timestep around the next trajectory point, and the optimal input is calculated using a linear sub-MPC of horizon N. This approach is very similar to the approach of a tube-based MPC (chapter 3.6 of [6]).

B. Linearization around a trajectory

To avoid drift due to a non-zero first order term when linearizing around a non-equilibrium point (x_l, u_l), tracking error was $\tilde{x}^+ = x^+ - x_t^+$ used to accomplish trajectory racking. Linearizing x^+ around the desired trajectory point $x_l = x_t$ then results in:

$$\tilde{x}^+ = A_t\tilde{x} + B_t\tilde{u} \tag{19}$$

where (A_t, B_t) are Jacobian matrices for $(x_l, u_l) = (x_t, u_t)$.

C. The trajectory planning problem

For computing the optimized trajectory, the `nlmpcmove` and `nlmpcmultistage` tools were used to utilize an iLQR algorithm only for the initial state and omit the receding horizon.

Using the nonlinear state space equations from 18, a trajectory was optimized using a finite horizon nonlinear MPC. For the cost function the following LQR cost function was used:

$$V_t(\mathbf{x}_t, \mathbf{u}_t) = \sum_{k=0}^{M-1} x_t(k)^T Q_t x_t(k) + u_t^T(k) R_t u_t(k) + x_t^T(M) Q_M x_t(M) \tag{20}$$

Where M is the trajectory horizon and $x_t(M)^T Q_M$ is the final stage cost.

Note that this trajectory can be computed offline before the trajectory tracking sub-MPC is activated. The trajectory has been constructed using an optimization algorithm, but can be any trajectory that is desirable, given $x_t(k+1) = \{\phi(k; x_t(k), u_t(k)) | (x_t(k), u_t(k)) \in \mathbb{Z}\}$. This means that not only stabilizing trajectories can be realized, but also non-stabilizing trajectories like a rocket launching trajectory.

V. Trajectory Tracking for landing a rocket

The problem of optimizing a sequences of inputs to reach a desired final state with a nonlinear MPC has now been transformed into the problem of tracking a predefined nominal trajectory. The nominal trajectory can be tracked even for disturbances as long as the the state remains within the attractor set $x \in \mathcal{X}_N$

As mentioned before, the nominal trajectory is defined over a horizon M, and the sub-MPC is defined over a horizon N. At each timestep k the sub-MPC is ran which aims to steer the current state to the next nominal trajectory state in N steps. For the nominal trajectory the index k is reserved, and for the sub-MPC problem the index i is reserved.

The target reference at $x_r(k)$ is taken to be exactly trajectory point $(x_t(k+1), u_t(k+1))$. When accounting for unknown disturbances like for example a wind disturbance and/or when using output control, the following target optimization is proposed to run online at every step i.

$$(x_r, u_r)(y_r) \in \begin{cases} \arg\min_{x_r, u_r} J(x_r, u_r) \\ \text{s.t.} \\ \begin{bmatrix} I & -A & -B \\ 0 & C & 0 \end{bmatrix} \begin{bmatrix} x_r(k+1) \\ x_r(k) \\ u_r(k) \end{bmatrix} = \begin{bmatrix} 0 \\ y_t(k) - \hat{d}(k) \end{bmatrix} \\ \qquad\qquad \forall k = 1, 2, ..., M-1 \\ \begin{bmatrix} (I-A) & -B \\ C & 0 \end{bmatrix} \begin{bmatrix} x_r(M) \\ u_r(M) \end{bmatrix} = \begin{bmatrix} 0 \\ y_t(M) - \hat{d}(M) \end{bmatrix} \\ (x_r, u_r) \in \mathbb{Z} \\ Cx_r \in \mathbb{Y} \end{cases}$$

A. Numerical stability

Implementing this strategy directly proved to be insufficient. Numerical instabilities arose from the Hessian of the cost function becoming indefinite. This problem was solved by preconditioning the Hessian using the LQR optimal gain K.

The total cost function of over the complete horizon can be written as:

$$V(\tilde{\mathbf{u}}_N, x_0) = \frac{1}{2}\tilde{\mathbf{x}}^T \bar{Q} \tilde{\mathbf{x}} + \frac{1}{2}\tilde{\mathbf{u}}^T \bar{R} \tilde{\mathbf{u}}$$

where $\bar{Q} = \begin{bmatrix} I_M \otimes K & 0 \\ 0 & \beta P \end{bmatrix}$, and $\bar{R} = I \otimes R$.

Now the input will be based on the stabilizing feedback control gain K.

$$\tilde{u}(i) = K\tilde{x}(i) + v(i)$$

where v is the new optimization variable.

the state sequence can then be written as

$$\tilde{\mathbf{x}} = T\tilde{\mathbf{x}}_0 + S\bar{K}\tilde{\mathbf{x}} + S\mathbf{v}$$
$$\tilde{\mathbf{x}} = \underbrace{(I - S\bar{K})^{-1}T}_{T_K}\tilde{x}_0 + \underbrace{(I - S\bar{K})^{-1}S}_{S_K}\tilde{\mathbf{v}}$$

or alternatively:

$$T_K = \begin{bmatrix} I \\ A_K \\ \vdots \\ A_K^N \end{bmatrix}, \quad S_K = \begin{bmatrix} 0 & 0 & ... & 0 \\ B & 0 & ... & 0 \\ A_K B & B & \ddots & \vdots \\ \vdots & \ddots & \ddots & 0 \\ A_K^{N-1}B & A_K^{N-2}B & ... & B \end{bmatrix}$$

As the eigenvalues of A_K are in the unit circle, the matrices $A_K^c \to 0$ for $c \to \infty$. This avoids large numerical values in the prediction matrices which solves the problems arising form numerical instability.

The resulting well-posed cost function will be

$$V(\mathbf{v}, x_0) = \frac{1}{2}\mathbf{v}^T(S_K^T \bar{Q} S_K + \bar{R})\mathbf{v} + \tilde{x}_0^T T_K^T \bar{Q} S_K \mathbf{v}$$
$$V(\mathbf{v}, x_0) = \frac{1}{2}\mathbf{v}^T H_K \mathbf{v} + h_K \mathbf{v}$$

The algorithm for the sub-MPC will then be the following:

The results of this algorithm are simulated in section VI.

B. Stability

The input now contains three terms:

$$u = \begin{cases} u_t, \text{ the nominal inputs calculated offline / via OTS} \\ K\tilde{x}, \text{ a LQR optimal feedback to precondition H} \\ v, \text{ to enforce stability of the tracking error} \end{cases}$$

For stability of the tracking error, the same stability analysis as before can be extended to $\tilde{x}$. In [6], a proposition is made to adress the terminal set of the tracking error at each timestep as a tube or sequence of sets $\mathbf{X}_c(\bar{\mathbf{x}}) = \{\mathbb{X}_0^c(\bar{\mathbf{x}}), \mathbb{X}_1^c(\bar{\mathbf{x}}), ...\}$ where $\bar{\mathbf{x}}$ is the nominal trajectory and $\mathbb{X}_i^c(\bar{\mathbf{x}}) \coloneqq \{x | V_N^0(x, i) \leq c\}$.

The selection of c is not done directly as this would mean calculating a terminal set for every timestep, which is unpractical. Instead, the method of scaled terminal stage cost is used.

Proposition 3.16 *(Implicit satisfaction of $\mathbb{X}_f$) [6]*

For all $c > 0$ there exists a $\beta_c \coloneqq c/\alpha$ such that, for any $i \in \mathbb{R}_{\geq 0}$ and any $x \in \mathbb{X}_i^c(\bar{\mathbf{x}})$, the terminal state $x_f(N; x_0, i)$ lies in $\mathbb{X}_f$ if $\beta \geq \beta_c$

Here again β is used to penalize the final stage cost $V_f(\tilde{x}(N)) \leftarrow \beta V_f(\tilde{x}(N))$. β should now be chosen in such a way that $\mathbb{X}_i^c(\bar{\mathbf{x}}) \subset$

To allow for imperfections in the model and uncertainties, the nominal state and input constraints should be chosen such that for $\bar{\mathbf{x}} \in \bar{\mathbb{X}}, \bar{\mathbf{u}} \in \bar{\mathbb{U}} \rightarrow \bar{\mathbb{X}} \subset \mathbb{X}, \bar{\mathbb{U}} \subset \mathbb{U}$. So either the constraints should be relaxed for the sub-MPC or the constraints should be tightened for the trajectory planning optimization.

C. Choice of weighting matrices

The choice of weighting matrices of the sub-MPC when tracking an optimized trajectory turned out to be crucial. As the nominal trajectory planning already used the Q_t and R_t matrices, it is recommended to tune the desired nominal behaviour using Q_t and R_t and choose Q and R to both be identity. This way the overall nominal state-input trajectory is tracked as close as possible, decreasing the chance of the rocket becoming unstable.

VI. Numerical Simulations

In this section we show the numerical simulations of the MPC system. First we show the regulation MPC and it's performance compared to LQR control, as well as the results for different choices of weighting matrices, prediction horizons and discretization times. Following this, the observer system is simulated, where the effect of an external disturbance is shown and how the system tracks, and finally when we add an additional reference to the system. Finally, we show numerical simulations for the trajectory tracking case. All simulations were ran on the full nonlinear system.

A. Regulation MPC

1) Prediction Horizon

The prediction horizon is a key parameter in MPC design. Simulations showed that starting far from the terminal set $\mathbb{X}_f$ often led to infeasibility, while initial conditions near $\mathbb{X}_f$ resulted in immediate control saturation, mimicking a saturated LQR response regardless of horizon length. To address this, state constraints were relaxed using slack variables as described in Section 1.2 of [5], allowing for feasible MPC solutions over a broader range of conditions and reducing computational load. Figure 1 illustrates the impact of varying the horizon N: values below 5 caused instability, while horizons above 20 yielded no performance gains and even degraded performance due to model nonlinearity. Since increasing N enlarges the region of attraction (by d per step in an ellipsoidal sense), we selected $N = 20$ as a trade-off between performance and computational efficiency.

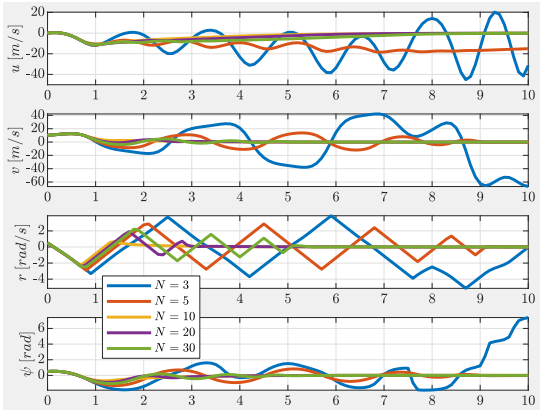


Fig. 1: Effect of Prediction Horizon on Response

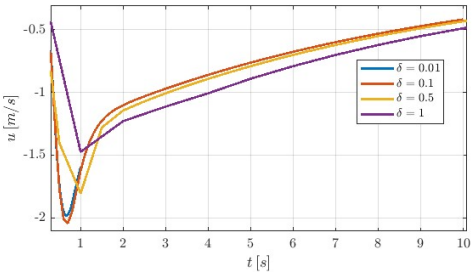


Fig. 2: Effect of differing sampling times on response

2) Sampling time

Numerical simulations showed that a sampling time of 0.5 seconds lead to a good balance between MPC performance and MPC simulation time. Despite this, we additionally see a rise-time of 1 second, leading us to chice a sampling time of 0.1 seconds to capture all relevant dynamics in accordance with theory [4]. Additionally, one can observe increased performance with higher sampling time. One could argue that this is due to the MPC being able to "see" farther into the future, and thus making more optimal choices.

3) Weighting Matrices

A good weighting matrix combination was found to be

$$Q = \text{diag}([1000 \cdot \text{ones}(3,1) 100 \cdot \text{ones}(6,1)]), \quad (21)$$

$$R = \text{diag}([0.1\ 0.1\ 0.1\ 0.001]) \quad (22)$$

Numerical simulations for various choices of Q and R are shown in Figure 3. Again, we take the same initial conditions as in the previous section.

4) Initial conditions

Below we show the MPC simulated on the nonlinear system for varying initial conditions. Since the MPC is ran on a nonlinear system, the theoretical guarantees for the region of

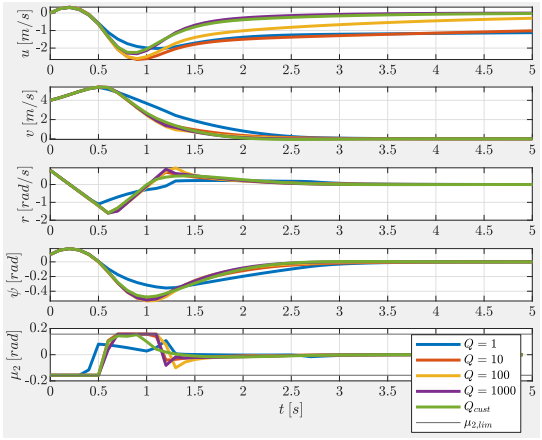


Fig. 3: Effect of weighting matrices on system response

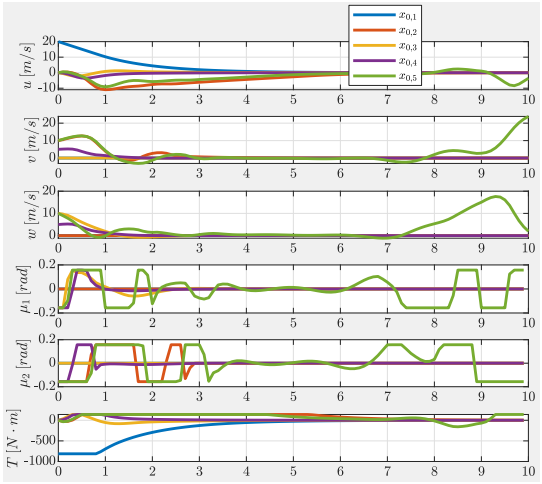


Fig. 4: Response to varying initial conditions

attraction do not apply, and we verify a posteriori if we are able to stabilize for a range of initial conditions. We simulate for positive upwards velocity, an offset to the rocket velocity in the y-direction, an offset to the rocket velocity in the z-direction, an offset to the velocity in both directions and an offset in both directions with positive angular velocity around the x-axis. Results are shown in Figure 4. Only for the last case does the system destabilize showing a good ability to regulate, even far away from equilibrium.

5) Comparison to LQR

As described in [1], the "usual" approach to the rocket regulation problem is via LQR. We show in Figure 5 that our MPC performs significantly better than the LQR control law, given that we tune the LQR controller to not saturate our controls. Note that only states with significant responses are shown. In the case of vertical and sidewards velocity as well as the yaw angle we note an improvement of around 1 second in the settling time. In the case of the angular velocity around the z-axis, the MPC is more aggressive, however settles more quickly as well. For these simulations, weighting matrices as

$$Q = \text{diag}([100\ 0.1 \cdot \text{ones}(1,8)]), \tag{23}$$

$$R = \text{diag}([1000 \cdot \text{ones}(1,3)\ 0.1]) \tag{24}$$

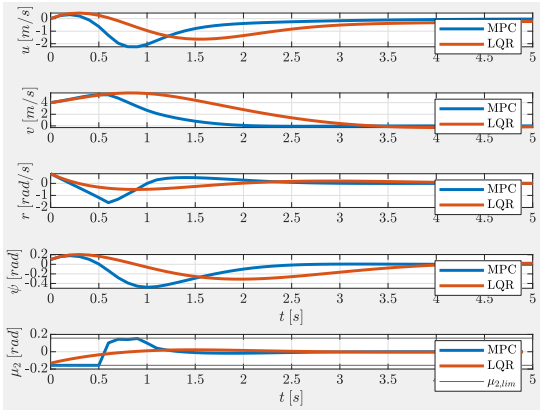


Fig. 5: Response Comparison LQR and MPC

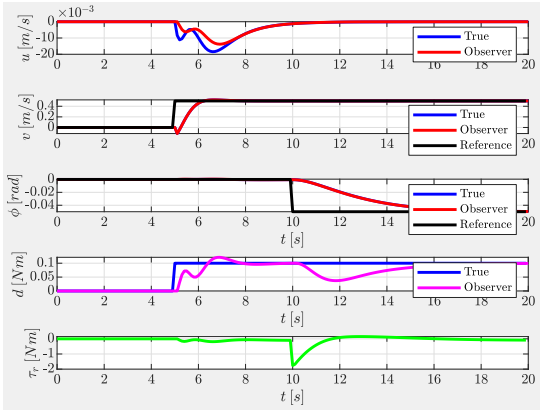


Fig. 6: Reponse to both Disturbances and Reference Changes

were chosen for the LQR controller such that the controls were not saturated but still found a quick response.

VII. Output MPC

This section demonstrates successful disturbance rejection and state estimation using the Leuenberger observer. Figure 6 shows a time-varying disturbance on the angular velocity p, which the observer effectively tracks. Note that this is simulated on the nonlinear system, where nonlinearities may be misinterpreted as disturbances, potentially leading to instability. We assume a noise-free environment with disturbances acting only on the states.

Figure 6 also shows accurate tracking of a varying reference. As detailed in Section II, we observe body-frame velocities u, v, w and Euler angles ϕ, θ, ψ. Although we use an OTS program for state and input optimization, this is not strictly necessary: the disturbance can be countered directly via τ_r, and the reference states are directly observable. In principle, one could set $p, q, r = 0$ and assign the remaining states directly from y. We retain the OTS in our MPC to demonstrate its implementation, using softened constraints to expand the feasible set to $\mathbb{R}^9$ for improved performance.

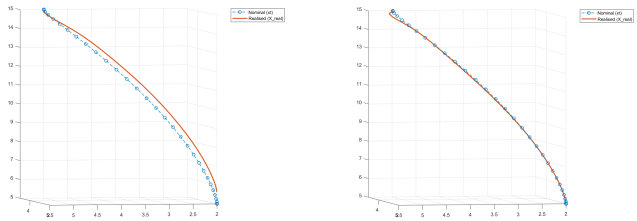
Additional system responses for disturbance rejection and reference tracking are shown in Appendix G, Figures 9 and 10.

VIII. Trajectory tracking

First the planned trajectory was fine-tuned by choosing different weighting matrices based on a certain initial state and terminal state (the terminal position of the rocket in the inertial frame will be [x,y,z]=[5,2,2]). The following weights achieved well-behaved dynamics (note that for reference tracking inputs τ and T are switched:

$$\begin{cases} Q = \text{diag}(1,1,1,10,10,10,1,10,10,1,100,100) \\ R = \text{diag}(0.1,0.1,10,0.01) \end{cases}$$

Next step is tuning the trajectory tracking behaviour. As mentioned before, a scaled final cost by a factor β can enforce stability by creating a tube of terminal sets $\mathbf{X_c}(\bar{\mathbf{x}})$. The factor β should not be chosen to be too high, as this will lead to very aggressive tracking behavior. After some tweaking, a value of $\beta = 10^3$ was found to give both stable and smooth tracking behavior. To see how the stabilizing input term v affected the system, simulation of the MPC with and without this term where compared in 7a and 7b. Although the LQR based controller yields a smooth trajectory, a steady state error can be seen. Adding the v term stabilizes the tracking error, and achieves accurate tracking. The action of the input terms reveal...



(a) LQR based trajectory tracking: $u(i) = u_r + Kx(i)$

(b) LQR + error correction based trajectory tracking: $u(i) = u_r + Kx(i) + v(i)$

Fig. 7: Comparison of LQR and MPC

A. Arbitrary trajectories

A motivation for the sub-MPC reference tracking setup given in IV-C is that, in contrary to optimizing a nonlinear optimization at each timestep to recalculate an optimal trajectory, this method can be used for *any* (possibly non-stabilizing) trajectory. An interesting, non-stabilizing trajectory could be a launch trajectory. In figure 8, the application of this can be seen as a in appendix H predetermined launch trajectory is followed by the rocket. In IV it was mentioned that the trajectory should satisfy $x_t(k+1) = \{\phi(k; x_t(k), u_t(k)) | (x_t(k), u_t(k)) \in \mathbb{Z}\}$. It is however noteworthy to mention that when this is not the case, the MPC still maintains stability when the trajectory is well behaved.

B. Discretization

As the system is treated by the sub-MPC as a linear system sampled at timesteps k, a faster sampling then the time between trajectory points is chosen as T_p.

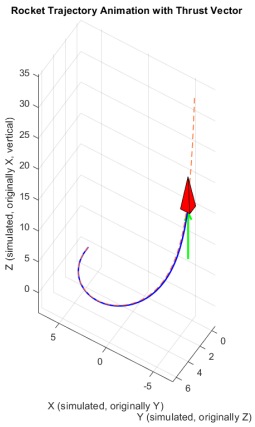


Fig. 8: A predetermined launch trajectory

IX. Conclusion

The regulation and output MPC was found to achieve the tasks well, given that the initialization point was close to the linearization point. Increasing the prediction horizon did not at all times lead to a better MPC remarkably, presumably owing to the highly nonlinear nature of the problem. Stability guarantees are made, with a terminal set defined in two ways, where it was chosen for an implicit terminal set.

The proposed trajectory tracking algorithm is one that prioritizes a reasonable computing load during tracking. By linearizing every kth step, trajectory tracking can be very fast which might be desirable for fast-moving vehicles such as a sounding rocket. The trajectory tracking MPC was found to perform better than an LQR and removed steady-state error.

References

[1] P. dos Santos and P. Oliveira, "Thrust vector control and state estimation architecture for low-cost small-scale launchers," arXiv e-prints, pp. arXiv–2303, 2023.

[2] tomlijding and WesleyNijhuis, tomlijding/TVC-rocket-mpc, 4 2025. [Online]. Available: https://github.com/tomlijding/TVC-rocket-mpc

[3] K. Hammargren, "Aerodynamics modeling of sounding rockets: A computational fluid dynamics study," 2018.

[4] K. J. Åström and B. Wittenmark, Computer-controlled systems: theory and design. Courier Corporation, 2013.

[5] A. Parisio, E. Rikos, and L. Glielmo, "A model predictive control approach to microgrid operation optimization," IEEE Transactions on Control Systems Technology, vol. 22, no. 5, pp. 1813–1827, 2014.

[6] J. Rawlings and D. Mayne, Model Predictive Control: Theory and Design. Nob Hill Publishing, 2008.

APPENDIX

A. Flywheel dynamics

Contrary to the simplified model in [1], we do not assume that the roll of the system is controlled via an external control system. For this, we extend the model of [1] by assuming that we have access to a flywheel control system to control the roll of the aircraft. We define this torque to be

$$\tau_f = \begin{pmatrix} \tau_r \\ 0 \\ 0 \end{pmatrix} \tag{25}$$

B. Slack variables

To facilitate loosening the state constraints, the state constraints are adjusted to be

$$x \leq x_{ub} + \xi_{ub} \tag{26}$$
$$x_{lb} \leq x + \xi_{lb} \tag{27}$$
$$\xi_{lb}, \xi_{ub} \in \mathbb{R}^9_{\geq 0} \tag{28}$$

and the stage and terminal cost adjusted to be

$$\ell(x(k), u(k)) = x(k)^T Q x(k) + u(k)^T R u(k) \tag{29}$$
$$+\lambda \|\xi_{ub}\|_2^2 + \lambda \|\xi_{lb}\|_2^2 \tag{30}$$
$$V_f(x(N)) = x(N)^T P x(N) \tag{31}$$
$$+\lambda \|\xi_{ub}\|_2^2 + \lambda \|\xi_{lb}\|_2^2 \tag{32}$$

where λ is a hyperparameter used to adjust how "important" we find it that our constraints are not violated. More on this choice in Section VI.

C. Parameter Selection

Parameter	Symbol	Value	Unit
m	Mass	82.9	$[kg]$
J_t	(Transverse) Inertia	88	$[\frac{kg}{m^2}]$
l	Length	2.77	$[m]$
d	Diameter (of fuselage)	0.24	$[cm]$
S	Cross-sectional area	0.04524	$[m^2]$
x_{cp}	Center of pressure (distance to tip)	0.75	$[m]$
x_{cm}	Center of mass (distance to tip)	0.8	$[m]$
C_{Y_β}	Derivative C_Y w.r.t. β	-0.5	$[-]$
C_{N_α}	Derivative of C_N w.r.t. α	-0.5	$[-]$
C_{m_q}	Derivative of C_m w.r.t q	-10	$[-]$
C_{n_r}	Derivative of C_n w.r.t. r	-10	$[-]$
$C_{m_{\dot{\alpha}}}$	Derivative of C_m w.r.t. $\dot{\alpha}$	-10	$[-]$
$C_{n_{\dot{\beta}}}$	Derivative of C_n w.r.t $\dot{\beta}$	-10	$[-]$
ρ	Air density	1.225	$[\frac{kg}{m^3}]$
ϕ	Roll angle	0	$[rad]$

TABLE I: Parameter definition

D. Algorithm 1

Algorithm 1 Fast MPC for trajectory tracking without disturbances

Require: $f(x,u), (\mathbf{x}_t, \mathbf{u}_t)$ or $\mathbf{y}_t, N$.

for k = {1,2,..., M} **do**
  ▷Calculate optimal target
  $(x_r, u_r) \leftarrow (x_t(k+1), u_t(k+1))$ (or use OTS)
  ▷Linearize system around (x_r, u_r):
  $A \leftarrow \frac{\partial}{\partial x}\big|_{x=x_r} f(x,u)$
  $B \leftarrow \frac{\partial}{\partial u}\big|_{u=u_r} f(x,u)$
  ▷Discretize the system
  $\Phi, \Gamma \xleftarrow{\mathcal{Z}} A, B$
  ▷Solve the DARE by backward iter..
  $\xrightarrow{\text{DARE}} K, P$
  ▷Utelize sub-MPC to find $\mathbf{u}(k)_N$:
  for i=1,2,...,N **do**
    $(\tilde{x}(i), \tilde{u}(i)) \leftarrow (x(i) - x_r, u(i) - u_r)$
    ▷Find optimal control sequence
    $\mathbf{v}_N^* = \arg\min_{\mathbf{v}_N} V_N(\tilde{x}(k), \mathbf{v}_N)$
    ▷Determine optimal control
    $v(i) = \mathbf{v}_N^*(:, 1)$
    $u(i) = u_t(k+1) + K\tilde{x}(i) + v(i)$
    ▷Feed input to system
    $x^+(i) = f(x(i), u(i))$
  end for
end for

E. Algorithm 2

Algorithm 2 Finding Maximal Constraint Admissible Set

Initialization: Set $k \leftarrow 0$.
repeat
  for $i = 1, 2, \ldots, s$ **do**
    Compute
$$x_i^\star \leftarrow \underset{x}{\operatorname{argmax}}\, f_i\big(A_K^{k+1} x\big)$$
    subject to
$$f_j\big(A_K^t x\big) \le 0, \quad \forall\, j = 1, 2, \ldots, s,\ \forall\, t = 0, 1, \ldots, k.$$
  end for
  if $f_i\big(A_K^{k+1} x_i^\star\big) \le 0,\ \forall\, i = 1, 2, \ldots, s$ **then**
    Set $k^\star \leftarrow k$
    Define
$$\mathbf{X}_f \coloneqq \Big\{ x \in \mathbb{R}^n \mid f_i\big(A_K^t x\big) \le 0,\ \forall\, i = 1, 2, \ldots, s,\ \forall\, t = 0, 1, \ldots, k^\star \Big\}.$$
  else
    Set $k \leftarrow k + 1$
  end if
until termination condition is met

F. Linearized Rocket Dynamics (Floating Rocket)

$$A = \begin{bmatrix} 0 & r & -q & 0 & -w & v & 0 & g\,c_\Psi\,s_\theta & g\,s_\Psi\,c_\theta \\ -r & 0 & p & w & 0 & -u & -g\,s_\Psi\,s_\phi - g\,c_\Psi\,c_\phi\,s_\theta & -g\,c_\Psi\,c_\theta\,s_\phi & g\,c_\Psi\,c_\phi + g\,s_\Psi\,s_\phi\,s_\theta \\ q & -p & 0 & -v & u & 0 & g\,c_\Psi\,s_\phi\,s_\theta - g\,s_\Psi\,c_\phi & -g\,c_\Psi\,c_\phi\,c_\theta & g\,s_\Psi\,c_\phi\,s_\theta - g\,c_\Psi\,s_\phi \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & s_\phi\,t_\theta & c_\phi\,t_\theta & t_\theta\Big(q\,c_\phi - r\,s_\phi\Big) & \Big(t_\theta^2+1\Big)\Big(r\,c_\phi + q\,s_\phi\Big) & 0 \\ 0 & 0 & 0 & 0 & c_\phi & -s_\phi & -r\,c_\phi - q\,s_\phi & 0 & 0 \\ 0 & 0 & 0 & 0 & \dfrac{s_\phi}{c_\theta} & \dfrac{c_\phi}{c_\theta} & \dfrac{q\,c_\phi - r\,s_\phi}{c_\theta} & \dfrac{s_\theta\Big(r\,c_\phi + q\,s_\phi\Big)}{c_\theta^2} & 0 \end{bmatrix}$$

$$B = \begin{bmatrix} -\dfrac{T\,c_{\mu_2}\,s_{\mu_1}}{m} & -\dfrac{T\,c_{\mu_1}\,s_{\mu_2}}{m} & 0 & \dfrac{c_{\mu_1}\,c_{\mu_2}}{m} \\ \dfrac{T\,s_{\mu_1}\,s_{\mu_2}}{m} & -\dfrac{T\,c_{\mu_1}\,c_{\mu_2}}{m} & 0 & -\dfrac{c_{\mu_1}\,s_{\mu_2}}{m} \\ -\dfrac{T\,c_{\mu_1}}{m} & 0 & 0 & -\dfrac{s_{\mu_1}}{m} \\ 0 & 0 & \dfrac{1}{J_t} & 0 \\ -\dfrac{l\,T\,c_{\mu_1}}{J_t} & 0 & 0 & -\dfrac{l\,s_{\mu_1}}{J_t} \\ -\dfrac{l\,T\,s_{\mu_1}\,s_{\mu_2}}{J_t} & \dfrac{l\,T\,c_{\mu_1}\,c_{\mu_2}}{J_t} & 0 & \dfrac{l\,c_{\mu_1}\,s_{\mu_2}}{J_t} \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \end{bmatrix}$$

G. Disturbance rejection and output tracking simulations

In this section we show some plots for exclusively the disturbance rejection (Figure 9) or output tracking (Figure 10) cases.

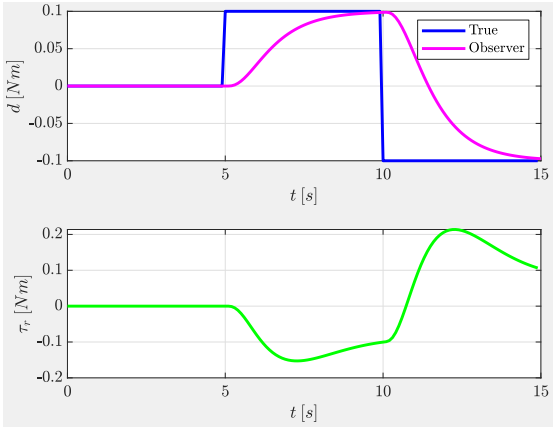


Fig. 9: Disturbance Response

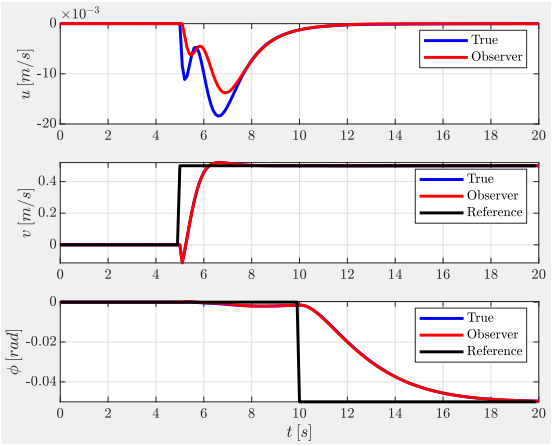


Fig. 10: Reference Tracking Response

H. Launch trajectory

The launch trajectory was predetermined using the simple following function:

$$\mathcal{S} = \left\{ (x_I, y_I, z_I) \middle| \begin{cases} x_I(r_t, \theta_t) = r_t \cos(\theta_t) \\ y_I(r_t, \theta_t) = r_t \sin(\theta_t) \\ z_I(r_t, \theta_t) = z_0 \tan(\theta_t/2) \end{cases} , r_t = 6, 0 < \theta_t < 0.97\pi, z_0 = 3 \right\}$$