École Polytechnique Fédérale de Lausanne

Model Predictive Control Mini-Project

Tom Herrmann (355 973)
Alexandros Dellios (355 873)
Salah Slaoui Hasnaoui (342 907)
Group BG

January 2026

2. Linearization

2.1 Deliverable 2.1

After linearizing around a trim point, we can then, at this equilibrium, make the choice of decomposing the system into **4 subsystems**. This separation is justified by the *small-angle approximation* valid near the equilibrium. Moreover, we see that it makes sense from a physical point of view. In fact, we only keep the variables that are relevant for each subsystem.

For the **subsystem** x, the deflection angle δ_2 will create a torque around axis y_b. We then keep ω_y, the angular speed around y, and β being the pitch angle around y_b between the world and body frame. This tilt will drive the position x and v_x.

For the **subsystem** y it's practically the same thing with δ_1 creating a torque around axis x_b, changing the angle α, which drives v_y and y.

However, for the **subsystem** z, the vertical motion is dominated by the average total thrust power P_{avg} who will act on the height z and velocity v_z.

Finally, for the **subsystem** $roll$, the difference in the thruster power P_{diff} will create a torque around axis z_b and influence the rocket's roll angle.

3. Design MPC Velocity Controllers

3.1 Deliverable 3.1

3.1.1 Design procedure

Here we created a subclass for each subsystem. We had to be aware of all our constraints:

$$|\alpha| \leq 10^\circ, \quad |\beta| \leq 10^\circ, \quad 40 \leq P_{avg} \leq 80$$
$$|P_{diff}| \leq 20, \quad |\delta_1| \leq 15^\circ, \quad |\delta_2| \leq 15^\circ$$

In order to ensure recursive satisfaction, we computed O_∞, the maximum invariant set for the closed-loop system under the local LQR controller K_{LQR} using the algorithm seen in class. We enforce $x_N \in O_\infty$, and this ensures that once the terminal set is reached, a feasible control law exists for all future time steps.

To ensure stability and approximate the infinite-horizon cost, we also computed the terminal weight matrix Q_f by solving the Discrete Algebraic Riccati Equation (DARE) for the unconstrained LQR problem.

Input: f, $\mathbb{X}$
Output: $\mathcal{O}_\infty$

$\Omega_0 \leftarrow \mathbb{X}$
loop
  $\Omega_{i+1} \leftarrow \text{pre}(\Omega_i) \cap \Omega_i$
  if $\Omega_{i+1} = \Omega_i$ **then**
    return $\mathcal{O}_\infty = \Omega_i$
  end if
end loop

Figure 3.1: *Algorithm from Professor Colin Jones MPC class*

3.1.2 Choice of tuning parameters

An horizon of 3.0 seconds was chosen as a trade-off. It is sufficiently long to capture the dominant dynamics of the

system, while keeping the computational load low.

Subsystem	**State Penalty** (Q)	**Input Penalty** (R)
X	$10 \cdot I$	$10 \cdot I$
Y	$10 \cdot I$	$10 \cdot I$
Roll (γ)	diag(10, **200**)	$\mathbf{1} \cdot I$
Z	$\mathbf{50} \cdot I$	$10 \cdot I$

Table 3.1: Tuning parameters for the Velocity MPC Controllers.

We chose balanced weights for X and Y subsystem with ratio ($Q/R = 1$) which provides smooth tracking performance without causing aggressive actuator usage or "chattering", which is critical for the stability of the roll and lateral dynamics.

Initially, a balanced tuning ($Q = 10, R = 10$) for the roll subsystem resulted in a sluggish response, failing the settling time requirement (> 7s). To fix this, we adopted a more aggressive tuning strategy for the roll subsystem. We heavily penalized the roll angle error. This forces the optimizer to treat any deviation from upright as a "high cost" state that must be eliminated immediately. We lowered the cost of using differential thrust to allow the controller to command large torque values.

For the Z subsystem, we increased the state penalty to $Q = 50 \cdot I$ (keeping $R = 10 \cdot I$). This higher ratio ($Q/R = 5$) makes the vertical controller more aggressive. This is physically justified because the rocket must constantly fight gravity; a reactive controller is required to minimize steady-state errors and prevent altitude loss during maneuvers.

3.1.3 Terminal invariant set

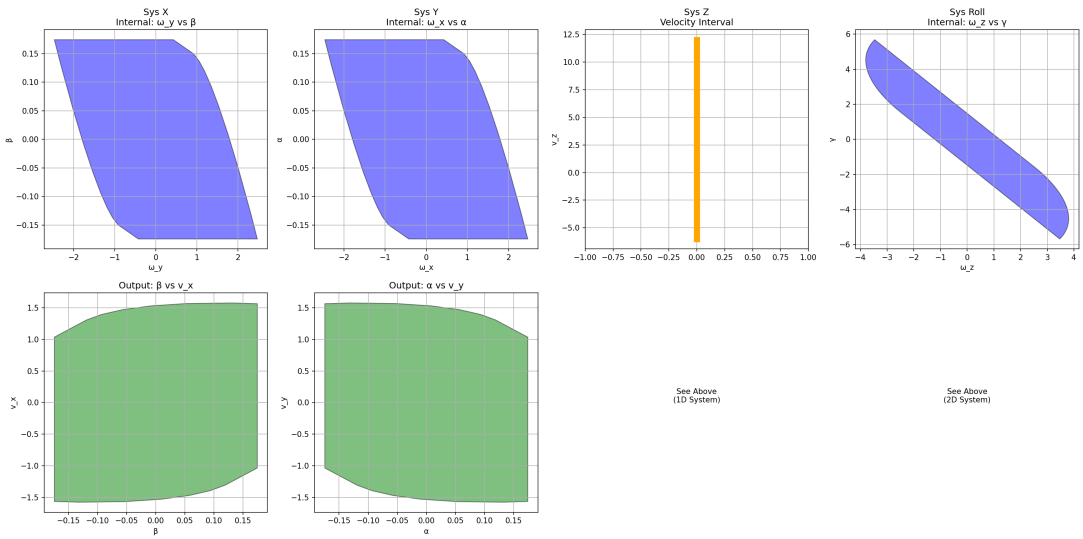

Figure 3.2: Maximum Invariant Sets for 3.1

The shape and volume of these sets are directly determined by the LQR gain, which depends on our tuning weights. The terminal invariant sets (X_f) represent the "safe" operating regions where the unconstrained LQR controller can stabilize the rocket without violating any constraints.

- **Lateral Systems (X and Y):** With balanced tuning ($Q = 10, R = 10$), the sets are relatively large. This indicates a significant stability margin.

- **Roll System:** The aggressive tuning ($Q_\gamma = 200, R = 1$) results in a high-gain controller. Consequently, the invariant set is much narrower.
- **Altitude System (Z):** Since position was removed for velocity control, the system is 1D. The set is an interval of vertical velocities.

3.1.4 Open-loop and closed loop plots

Here we can see that we respected our constraints and that the system stabilizes in less than 7 seconds.

1) When starting at speed 5 m/s:

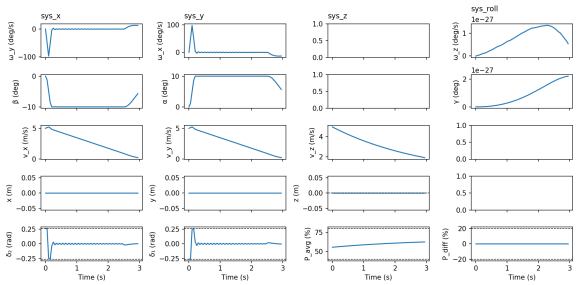

Figure 3.3: Open Loop Prediction

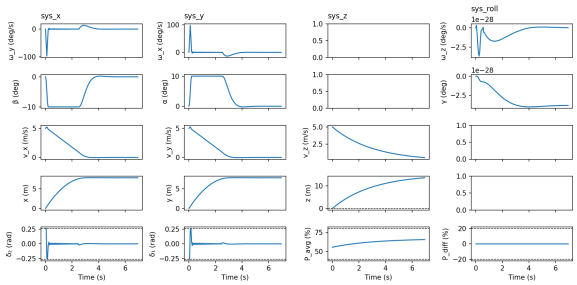

Figure 3.4: Closed Loop Trajectory

2) When starting at $\gamma = 30°$:

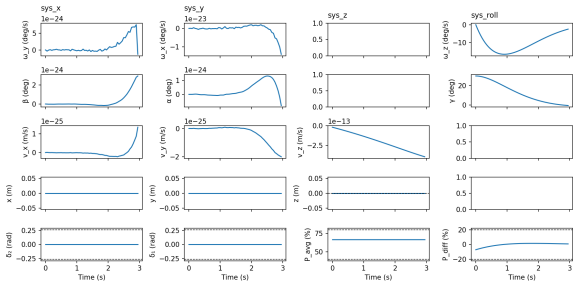

Figure 3.5: Open Loop Prediction

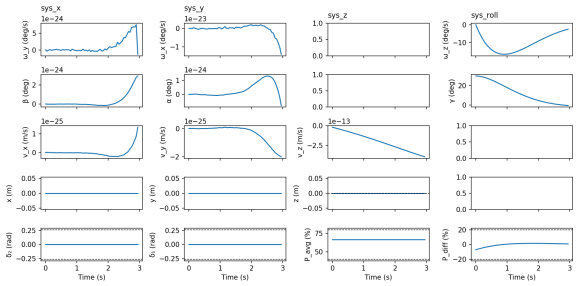

Figure 3.6: Closed Loop Trajectory

3.2 Deliverable 3.2

3.2.1 Design procedure and tuning

The objective function was generalized to penalize the error relative to a target state ($x - x_{targ}$) and a reference input ($u - u_{ref}$). This allows the rocket to track constant velocity setpoints (v_x, v_y, v_z) and a target roll angle (γ) instead of just returning to zero.

We increased the prediction horizon to 5.0s. A longer horizon is crucial for tracking because it allows the controller to "see" further ahead, ensuring a smoother approach to the target velocity and reducing overshoot during transitions.

Please also note that we kept the same matrices as in 3.1 because the primary dynamics of the rocket remain unchanged. Furthermore, we maintained the hard terminal invariant set constraint ($x_N \in O_\infty$) to guarantee recursive feasibility.

3.2.2 Open-loop and closed-loop plots

Results of tracking a target state with a velocity reference of 3 m/s for each direction and a roll angle of 35°.

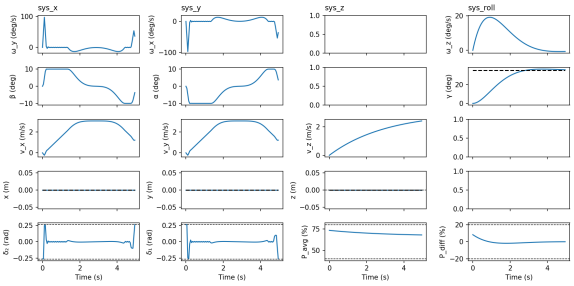

Figure 3.7: Open Loop

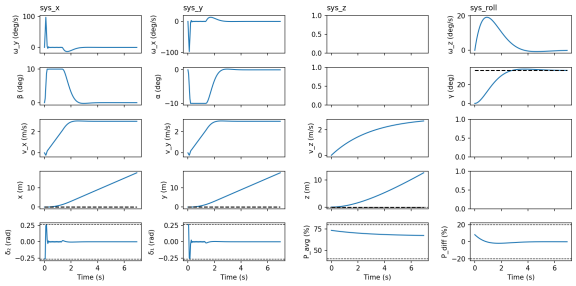

Figure 3.8: Closed Loop

As shown in the figure above, the closed-loop controller successfully drives the system to the reference targets. The roll angle γ settles smoothly at 35° within approximately 3 seconds with little overshoot. Similarly, the linear velocities $v_x, v_y,$ and v_z converge to the 3 m/s setpoint. The controller respects the actuator constraints, with the gimbal angles δ_1 and δ_2 exhibiting brief saturation during the initial transient before stabilizing at their equilibrium values.

3.3 Deliverable 3.3

3.3.1 Design procedure and tuning

For this deliverable, we added a PID position tracking controller on top of our MPC velocity controllers. The PID translates position errors into velocity setpoints, which are then tracked by the MPC. No changes were made to the MPC weighting matrices (Q, R) or the horizon $(H = 5.0s)$. The robustness of the velocity-level tuning allows it to track the dynamic setpoints generated by the PID without needing further adjustment (except the simulation time that was raised to 40s). Despite the large initial position error, the MPC successfully maintains the rocket within the limits during the entire approach.

3.3.2 Closed-loop plots

The closed-loop plots show a successful maneuver from $pos = [50, 50, 100]$ m to the target at $[0, 0, 10]$ m.

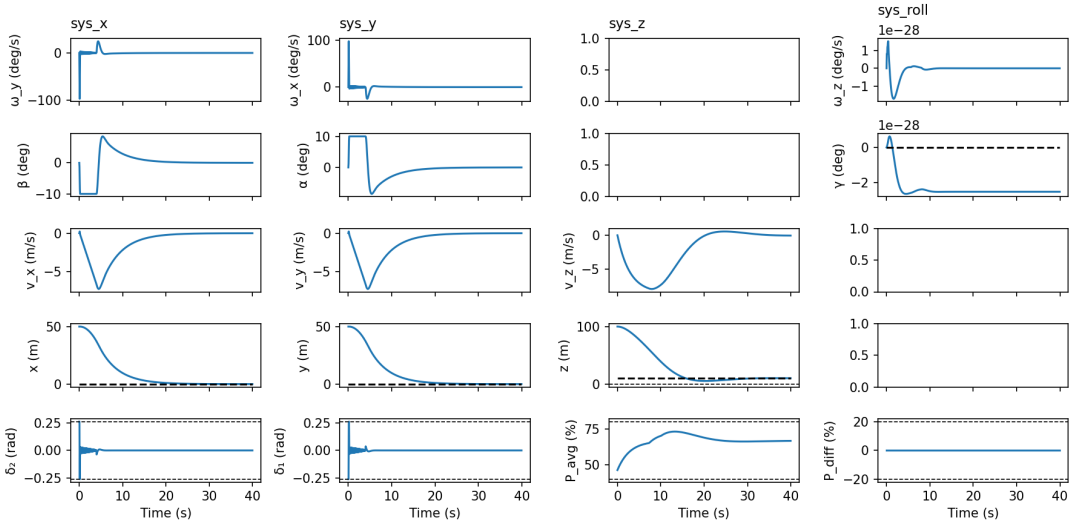

Figure 3.9: Closed Loop Trajectory (3.3)

The controller successfully guides the rocket to the target coordinates.

- **Transient Response:** In the first 5 seconds, the controller aggressively actuates the gimbal (δ) and changes the vehicle orientation (α, β) to generate the necessary lateral velocities. This results in an initial saturation of the gimbal constraints and angle of attack limits, which is expected behavior when correcting large initial position errors.
- **Convergence:** The vertical position (z) descends smoothly from 100m to 10m, stabilizing around $t = 25$s. The horizontal positions (x, y) follow a similar trajectory, converging to the origin without significant overshoot.

The little oscillations that we see for the plot of the γ angle is of order of magnitude 10^{-28}, so we can deduce that it is simply numerical noise.

4. Simulation with Nonlinear Rocket

4.1 Deliverable 4.1

We tested the cascaded PID-MPC control architecture on the full nonlinear rocket model. To address the inherent model mismatch introduced by the non-linear dynamics, which would lead to infeasibility, we introduced **soft state constraints** using slack variables (while keeping the same cost weights as previously). These variables allow minor, heavily penalized violations of state constraints. The closed-loop plots demonstrate a successful maneuver from the initial state to the target position $[0, 0, 10]$m. The z position converges smoothly to the 10m setpoint without the significant undershoot seen in earlier tests.

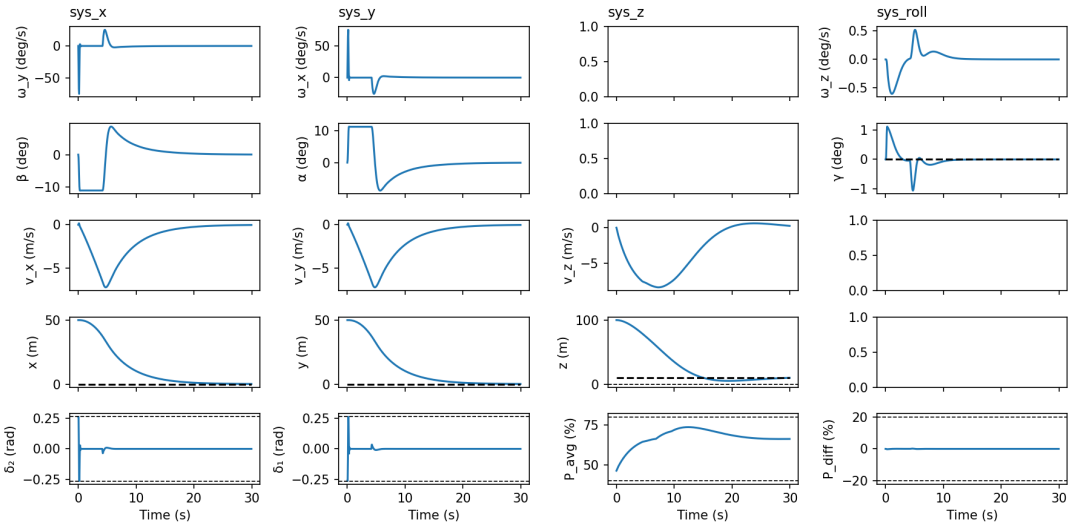

Figure 4.1: Closed Loop Trajectory (4.1)

Despite the model mismatch, the system remains stable. Unlike the linear simulation where the roll angle γ was perfectly zero, here we observe small transients due to dynamic coupling, which the controller successfully rejects.

The gimbal angles (δ_1 and δ_2) and angles of attack (α, β) saturate during the initial phase to generate maximum lateral force. Thanks to the slack variables, the transition out of saturation is smooth, and the recursive feasibility is maintained throughout the trajectory.

5. Offset-Free Tracking

5.1 Deliverable 5.1

5.1.1 Design procedure and choice of tuning parameters

The objective is to eliminate the steady-state error caused by a **change in the rocket's mass** (increased to 1.5× nominal). Under these conditions, the standard MPC's gravity compensation is insufficient. We implemented an Offset-Free MPC using a disturbance observer.

The standard linear model was augmented with a constant disturbance $d_k \in \mathbb{R}$ affecting the vertical dynamics:

$$\begin{bmatrix} x_{k+1} \\ d_{k+1} \end{bmatrix} = \begin{bmatrix} A & B_d \\ 0 & 1 \end{bmatrix} \begin{bmatrix} x_k \\ d_k \end{bmatrix} + \begin{bmatrix} B \\ 0 \end{bmatrix} u_k \tag{5.1}$$

where B_d represents the influence of the mass offset on the z-acceleration. We designed a **Luenberger** observer to estimate the augmented state.

We chose the poles to be $p = [0.7, 0.75]$. These poles are located inside the unit circle and not too close to the origin, as an excessively fast observer would become too sensitive to measurement noise.

For the vertical subsystem Z, the observer was tuned using $Q_{aug} = \text{diag}(50.0, 10.0)$. The high weight on the first diagonal element ensures that the observer prioritizes the accuracy of the altitude estimate. Finally : $R = 0.1 \cdot I$, indicating high confidence in the altitude measurements. Note that for this deliverable, the terminal set constraint was removed to maintain feasibility.

5.1.2 Comparison with non Offset-Free tracking

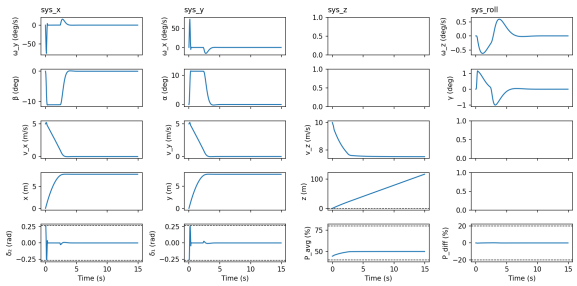

Figure 5.1: Part 4 Controller

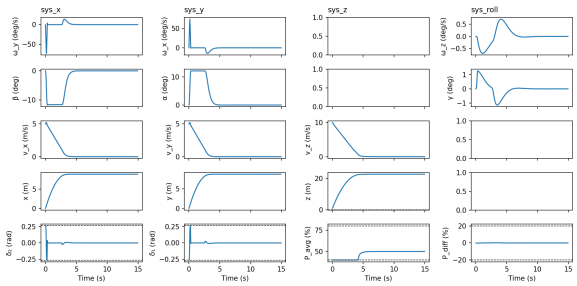

Figure 5.2: Offset-Free Tracking

Without offset-free control, the model mismatch leads to a constant error in the vertical velocity v_z. Since the velocity is non zero, the rocket continues to drift along the z-axis. The addition of the disturbance observer allows the controller to regain a true steady-state where $v_z = 0$.

5.1.3 Disturbance estimation

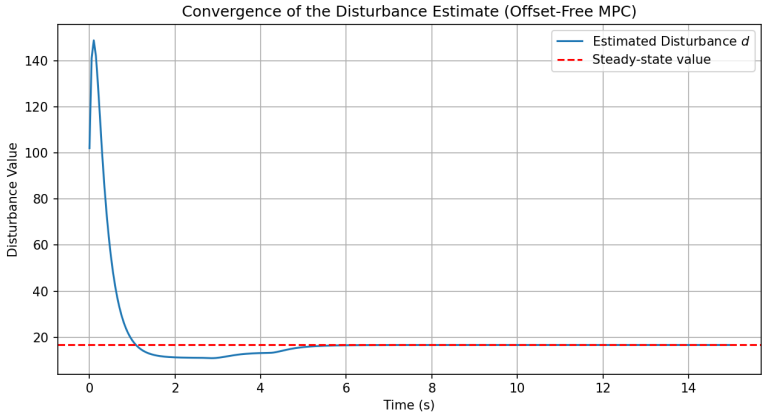

Figure 5.3: Evolution of d estimation

We observe that d converges to a constant steady-state value (approx. 18) after the transient phase. Since the gravitational force depends linearly on the mass, and the mass remains constant (fuel rate is zero here), the resulting disturbance is **constant**. That is because the error originates from the difference between the actual gravity force ($m_{real} \cdot g$) and the modeled force ($m_{nom} \cdot g$). Since both are constant, the resulting acceleration offset is constant.

5.2 Deliverable 5.2

5.2.1 Closed-loop trajectories

We now simulate the system with a initial mass ($m = 2.0$) and a fuel consumption rate ($fuel\_rate = 0.1$).

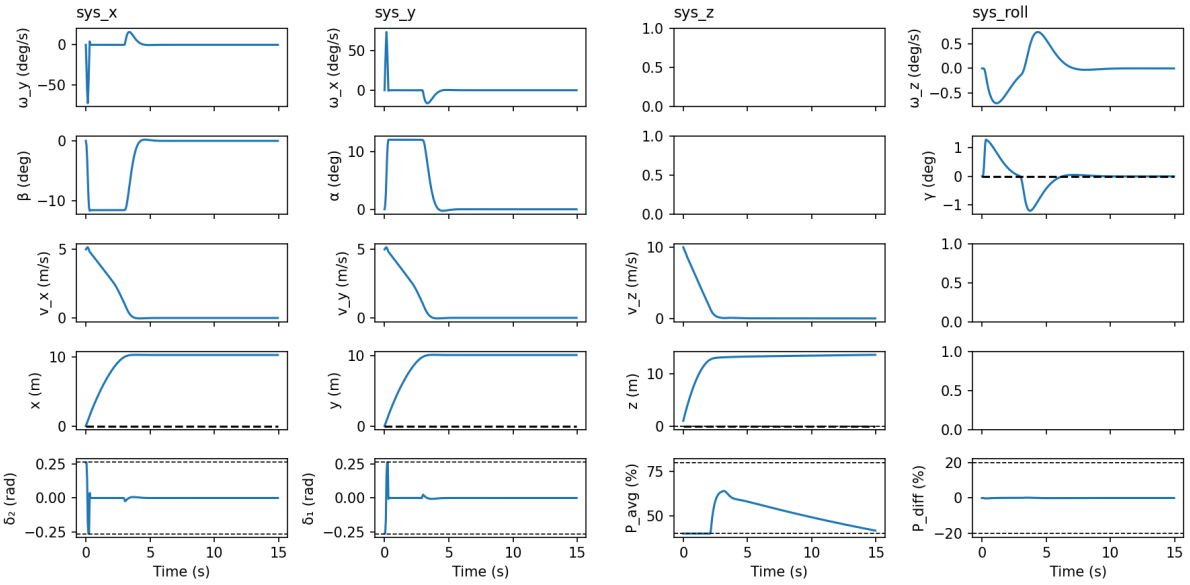

Figure 5.4: Simulation with $rocket.mass = 2.0$ and $fuel\_rate = 0.1$

5.2.2 Tracking offset at the beginning

Although the Offset-Free MPC eliminates steady-state error, a significant tracking offset is observed during the first few seconds. This behavior is due to the convergence dynamics of the disturbance observer. At $t = 0$, the observer is initialized with $\hat{d}_0 = 0$. However, the rocket immediately experiences the gravitational force of an enhanced mass : there is a massive initial model mismatch. As the prediction error accumulates, the Luenberger observer updates $\hat{d}$ and the offset disappears.

To reduce the duration of this initial error, the observer dynamics could be tuned to be more aggressive (poles closer to origin). However, faster observer dynamics make the system more sensitive to measurement noise.

5.2.3 Behaviors explanation

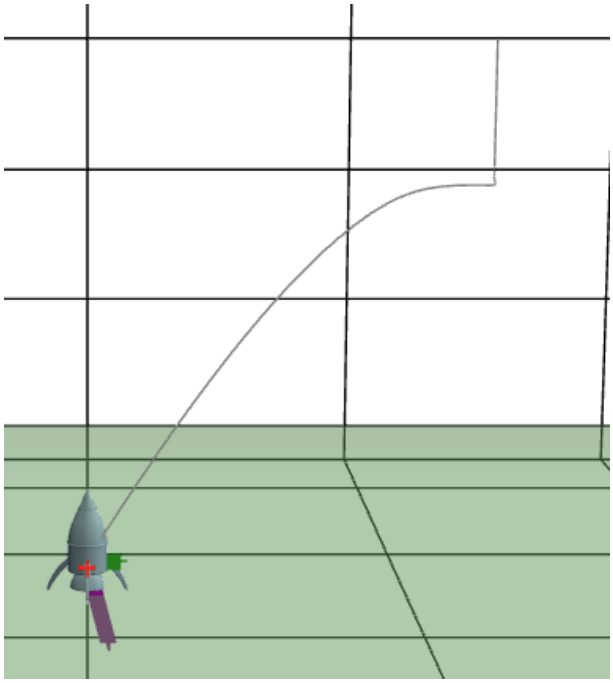

Figure 5.5: Trajectory of the rocket

We identify three distinct phases in the trajectory:

- **Phase 1:** At the very beginning, the rocket rises slower than expected (explained in 5.2.2).
- **Phase 2:** As the estimator converges, the average thrust increases sharply.
- **Phase 3:** The estimate $\hat{d}$ stabilizes, and the controller attempts to hold the position. As fuel is consumed, the rocket becomes lighter and we observe P_{avg} steadily decreasing.

However, towards the end, we see an unexpected behavior where the rocket overshoot our desired position along the z-axis. This is caused by the mismatch between the linear and nonlinear models regarding orientation. Consequently, the controller provides slightly more thrust than necessary, causing the rocket to drift upwards. This behavior highlights the limitation of the constant disturbance assumption ($d_{k+1} = d_k$) used in our augmented state model.

6. Robust Tube MPC for landing

6.1 Deliverable 6.1

6.1.1 Design Procedure

To ensure the rocket lands safely ($z \geq 0$) despite external disturbances and model mismatches (modeled as an additive disturbance $w \in \mathbf{W} = [-15, 5]$), we implemented a **Robust Tube MPC**. This control architecture ensures that the actual system state remains within a bounded "tube" centered around the nominal trajectory for all possible disturbances.

The design procedure follows these steps:

1. **Feedback Stabilization:** We computed a stabilizing feedback gain K using LQR. This gain is responsible for rejecting disturbances and keeping the system close to the nominal path.

2. **Minimal Robust Invariant Set (mRPI):** We computed the set $\mathcal{E}$ such that if the error state $e_k \in \mathcal{E}$, then $e_{k+1} \in \mathcal{E}$ for all admissible disturbances $w \in \mathbf{W}$. This set defines the cross-section of the tube.

3. **Constraint Tightening:** To guarantee robust feasibility, the constraints for the nominal MPC were tightened by the size of $\mathcal{E}$:

   - **State Constraints:** $\mathbf{X}_{tight} = \mathbf{X} \ominus \mathcal{E}$. The physical ground constraint is $z \geq 0$. In our delta-formulation ($\Delta z = z - 3$), this translates to $\Delta z \geq -3$. We defined the constraint matrix such that $f = [3.0]$, allowing the rocket to descend to the ground level even if the disturbance pushes it below the target altitude of 3m.
   - **Input Constraints:** $\mathbf{U}_{tight} = \mathbf{U} \ominus K\mathcal{E}$. This reserves a portion of the actuator authority specifically for disturbance rejection.

4. **Terminal Set:** We computed the maximal invariant set $\mathcal{O}_\infty$ for the nominal system under the tightened constraints to ensure recursive feasibility.

6.1.2 Choice of Tuning Parameters

We utilized a unified tuning approach where the same weights define both the robust feedback and the nominal performance:

- **Weighting Matrices:**
  - $Q = \text{diag}(48, 100)$: We placed a high penalty on the position state (index 1) to enforce precise tracking of the altitude target. The velocity weight (index 0) was set lower to allow necessary acceleration.
  - $R = 0.1$: A low input penalty was chosen to allow the controller to use the full range of thrust.
- **Horizon:** $H = 20$ steps (2.0 seconds) was sufficient to look ahead to the landing target while keeping computation times low.

6.1.3 Invariant Sets and Tightened Constraints

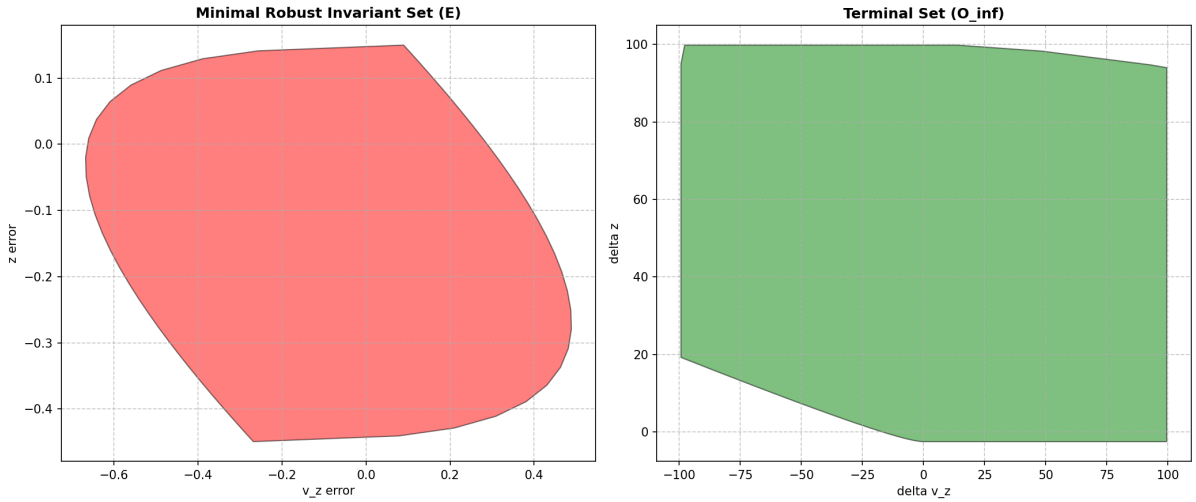

Figure 6.1: Minimal Robust Invariant Set **E** (Red) and Terminal Set $\mathcal{O}_\infty$ (Green).

The input constraints were tightened to account for the disturbance rejection effort. The vertices of the tightened input set $\tilde{\mathbf{U}}$ are:

$$\tilde{u}_{min} \approx -8.8580, \quad \tilde{u}_{max} \approx 2.4169$$

6.1.4 Closed-Loop Simulation Results

The controller was tested under 'Random' and 'Extreme' noise profiles. Under both disturbance levels, the system maintains feasibility. The aggressive tuning allows the rocket to apply maximum thrust to counteract the downward force, stabilizing the descent without violating the ground safety constraint ($z \geq 0$).

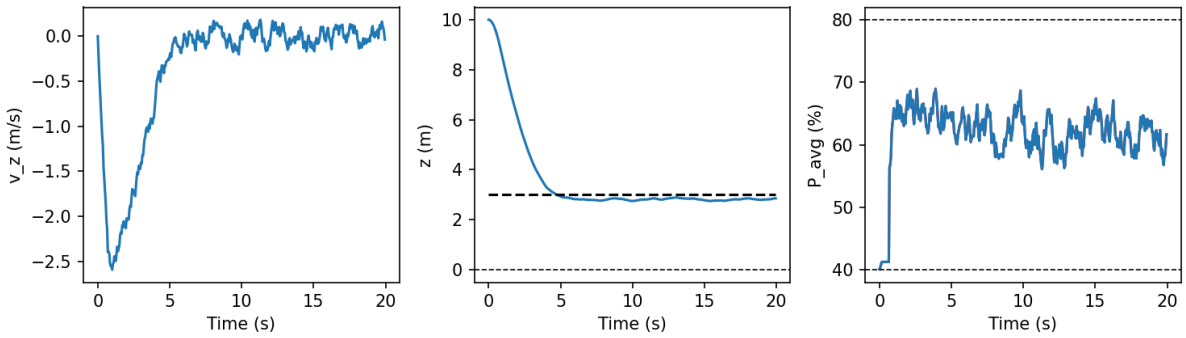

Figure 6.2: Trajectory under "random" noise

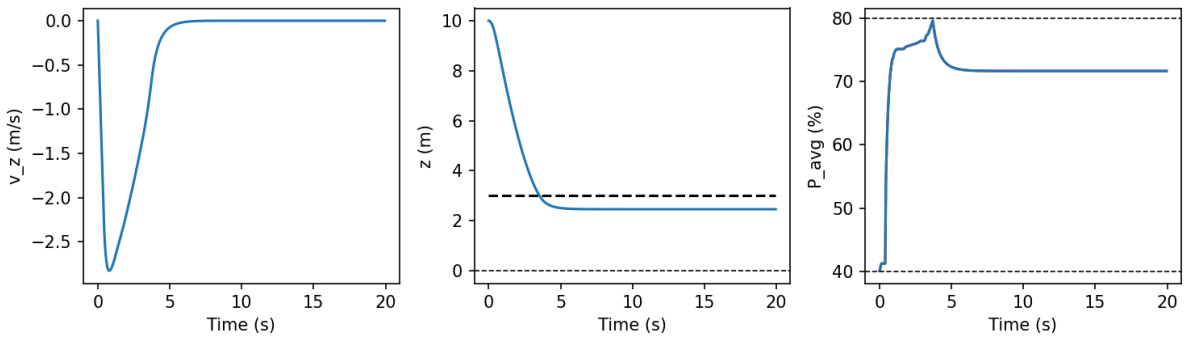

Figure 6.3: Trajectory under "extreme" noise

6.2 Deliverable 6.2: Nominal Controllers and Merging

6.2.1 Design Procedure

To control the full 3D position and orientation of the rocket during the landing phase, we added the 3 other subsystems and merge them together. While the z-subsystem utilizes the Robust Tube MPC designed in Part 6.1, the remaining three subsystems $(x, y, roll)$ are controlled by **Nominal MPC** controllers.

6.2.2 Choice of Tuning Parameters

We applied a balanced tuning approach to ensure smooth convergence within the required 4-second settling time.

- **Weights** (Q, R)**:**
  - We selected $Q = 10 \cdot I$ and $R = 10 \cdot I$ for the x, y.
  - This 1:1 ratio between state error and control effort provides a smooth trajectory without aggressive oscillations, which is crucial when operating near the ground.
  - For the $roll$ system, we needed to increase Q to $100 \cdot I$ to make the controller more aggressive as it was converging in approximately 10 seconds.
- **Soft Constraint Penalty:**
  - $\rho = 10^6$: A very high penalty is placed on the slack variables. This ensures that the constraints (max tilt angle $\approx 10°$) are respected strictly whenever physically possible, and violated only to prevent solver failure.
- **Constraints:**
  - **State (Tilt):** Limited to ≈ 0.1745 rad (10°) for x and y to prevent the rocket from becoming unstable during lateral moves.
  - **Input (Gimbal):** Limited to 0.26 rad ($\approx 15°$) for x/y and 20 for roll.

6.2.3 Closed-Loop Simulation Results

The merged controller was simulated descending from $[x = 3, y = 2, z = 10]$ with a roll of 30° to the target $[1, 0, 3, 0°]$.

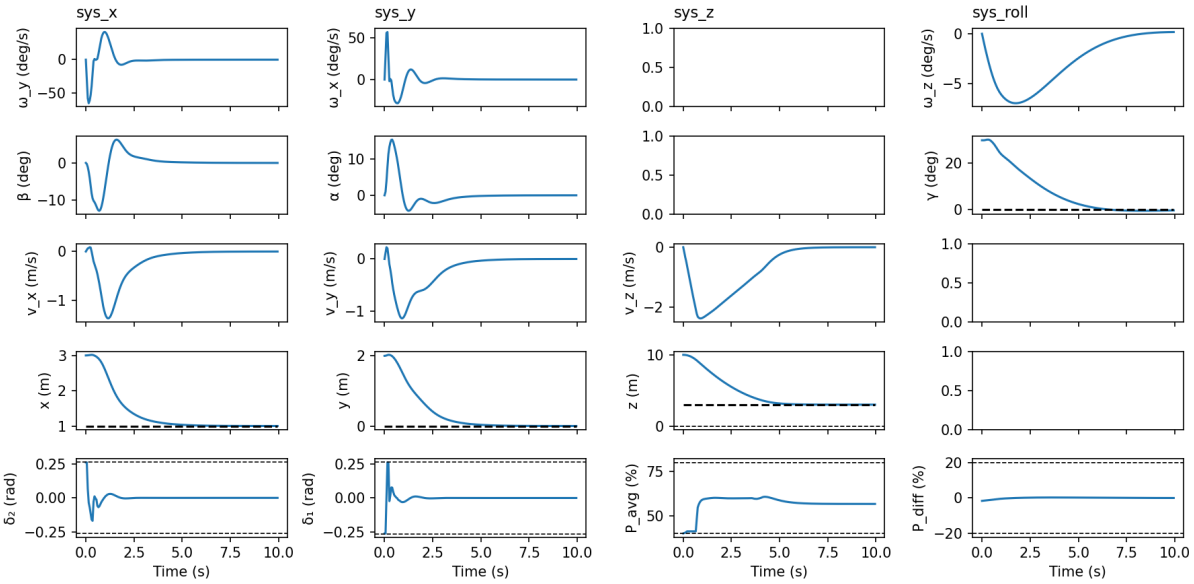

Figure 6.4: Closed-loop trajectory of the merged controller (Robust Z + Nominal X/Y/Roll).

The simulation shows that the rocket successfully stabilizes all states. The lateral positions (x, y) converge to the target $(1, 0)$ smoothly. The settling time is observed to be approximately 4 seconds, satisfying the performance requirement. The Robust Z-controller (from 6.1) successfully handles the altitude descent in parallel with the lateral corrections.

7. Nonlinear MPC

7.1 Deliverable 7.1

7.2 Deliverable 7.1: Nonlinear MPC

7.2.1 Design Procedure

For the final part of this project, we implemented a **Nonlinear MPC (NMPC)** using the `CasADi` optimization framework. This controller optimizes the full state vector and input vector simultaneously, without decoupling the dynamics into subsystems.

We utilized the continuous-time nonlinear dynamics $\dot{x} = f(x, u)$ provided by the rocket model. We implemented a custom **Runge-Kutta 4 (RK4) integrator** to discretize these dynamics within the optimization problem. This ensures high prediction accuracy even for aggressive maneuvers.

To ensure stability with a finite horizon while keeping computation times reasonable, we approximated the infinite horizon cost using a terminal cost term:

$$J_{term}(x_N) = (x_N - x_{ref})^T P_{LQR}(x_N - x_{ref})$$

The matrix P_{LQR} was computed by solving the Algebraic Riccati Equation for the linearized system at the target hover state.

We strictly constrained the pitch angle $|\beta| \leq 80°$ to avoid the Euler angle singularity at 90°. To prevent infeasibility during the initial aggressive recovery, we implemented the maximum thrust constraint as a soft constraint: $u_{thrust} \leq 80\% + \sigma$. The slack variable σ is heavily penalized ($\rho = 10^3$) in the cost function.

7.2.2 Choice of Tuning Parameters

To achieve the settling time requirement ($\leq$ 4s) without significant overshoot, we adopted an aggressive but sufficiently damped tuning strategy.

- **State Weights (Q):**
  - **Position (x, y, z):** $Q_{pos} = 1000$. We assigned the highest priority to the position states (x, y, z) to ensure the rocket converges exactly to the target $[1, 0, 3]$.
  - **Velocity (v_x, v_y, v_z):** We applied heavy penalties on velocities to provide damping and prevent overshoot:
    * v_z is set to **500**. This high damping on the vertical velocity ensures a soft landing and prevents altitude oscillations.
    * v_x, v_y are set to **200**. This prevents the rocket from drifting past the target during lateral corrections.
  - **Attitude:** The Roll angle γ is set to **200**. This high weight forces the controller to immediately correct the initial 30° roll error. The Pitch/Yaw angles α, β are kept low (**10**) to allow the rocket to tilt freely as needed to generate lateral thrust.
- **Input Weights (R):**
  - $R = 10 \cdot I$. We significantly reduced the input penalties. A high R made the gimbal response sluggish, preventing the rocket from braking in time to stop at the target.

7.2.3 Simulation Results

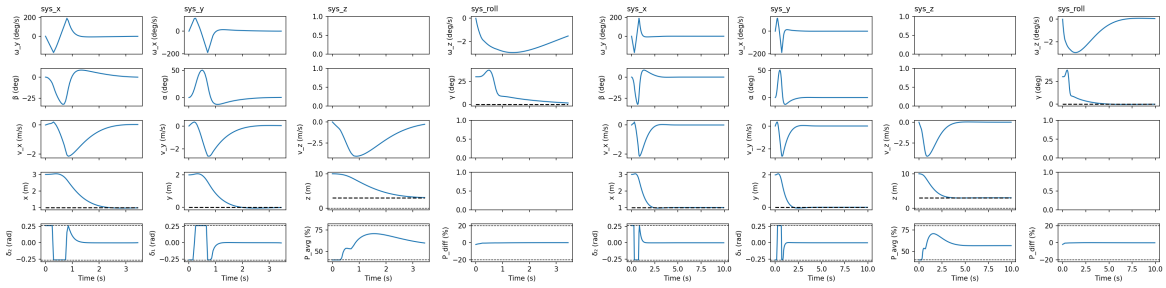

Figure 7.1: Left: Open-loop prediction at $t = 0$. Right: Closed-loop NMPC trajectory.

The simulation confirms the effectiveness of the NMPC. Starting from a difficult initial state ($Roll = 30°$, $z = 10m$), the controller predicts a coordinated maneuver (Figure 7.1, Left) and executes it successfully (Right). The roll angle is corrected within the first 2 seconds, and the rocket settles at the target $[1, 0, 3]$ in under 4 seconds. The trajectory is smoother compared to the linear controllers because the NMPC anticipates the coupling between the roll correction and the lateral movement.

7.2.4 Comparison: NMPC vs. Robust/Nominal MPC

We compared the centralized NMPC (Part 7.1) with the decentralized architecture (Robust Tube z + Nominal $x, y, roll$) from Part 6.2.

Nonlinear MPC (Part 7)	**Robust + Nominal MPC (Part 6)**
Pros: - Uses the exact model. Can handle large deviations (like 30° roll) optimally by exploiting nonlinear coupling. - Handles complex constraints naturally and no decoupling into subsystems.	**Pros:** - Solves Quadratic Programs (QP), which are convex and extremely fast (ms range). - The Tube formulation provides theoretical guarantees of constraint satisfaction under noise.
Cons: - Computationally expensive and requires a good initial guess to converge. - Standard NMPC does not inherently guarantee safety against disturbances without complex robust formulations.	**Cons:** - Performance degrades significantly far from the equilibrium. - Decoupling the axes ignores the fact that rolling induces lateral motion, leading to less efficient trajectories.

Table 7.1: Comparison of Control Architectures