Mini Project

Developing MPC controllers for a rocket prototype

Jiaxuan Zhang (322684)
Pengbo Zhu (323791)
Wei-Sheng Hung (322678)

February 14, 2022

Course:	Model Predictive Control
Professor:	Prof. Colin Jones
Date:	February 14, 2022

1 Part 2 | Linearization

Deliverable 2.1 | Explain from an intuitive physical / mechanical perspective, why this separation into independent subsystems occurs.

The state, input and output of the system is defined as follows:

$X = [\omega_x, \omega_y, \omega_z, \alpha, \beta, \gamma, v_x, v_y, v_z, x, y, z]^T = [\dot{\alpha}, \dot{\beta}, \dot{\gamma}, \alpha, \beta, \gamma, \dot{x}, \dot{y}, \dot{z}, x, y, z]^T$
$U = [\delta_1, \delta_2, P_{avg}, P_{diff}]^T$
$Y = X$

After the trimming and linearization, the state-space model and the matrices A and B are shown below:

$\dot{x} = A(x - x_s) + B(u - u_s) =$

$$\begin{bmatrix} 0 & & & & \dots & & & & & & & \\ 0 & & & & \dots & & & & & & & \\ 0 & & & & \dots & & & & & & & \\ 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 9.81 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & -9.81 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & & & & \dots & & & & & & & \\ 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 \end{bmatrix} x^* + \begin{bmatrix} -55.68 & 0 & 0 & 0 \\ 0 & -55.68 & 0 & 0 \\ 0 & 0 & 0 & -0.104 \\ 0 & \dots & & \\ 0 & \dots & & \\ 0 & \dots & & \\ 0 & 9.81 & 0 & 0 \\ -9.81 & 0 & 0 & 0 \\ 0 & 0 & 0.1731 & 0 \\ 0 & \dots & & \\ 0 & \dots & & \\ 0 & \dots & & \end{bmatrix} u^*$$

From the whole matrix, it can be organized as subsystems:

$$\begin{cases} \dot{\omega_x} = -55.68\delta_1 \\ \dot{\omega_y} = -55.68\delta_2 \\ \dot{\omega_z} = -0.104P_{diff} \end{cases} \quad \begin{cases} \dot{v_x} = 9.81\beta + 9.81\delta_2 \\ \dot{v_y} = -9.81\alpha - 9.81\delta_1 \\ \dot{v_z} = 0.1731P_{avg} \end{cases}$$

4-6 rows are just derivatives of 1-3, as well as rows 10-12 (derivatives of 7-9).

The equations of the angular acceleration $\dot{\omega_x}$ and $\dot{\omega_y}$ are the linearized version of Newton's Second Law for rotation, the moment arms are linearized as δ_1 and δ_2. For $\dot{\omega_z}$, it is relevant to the differential throttle P_{diff} because of the conservation of angular momentum.

The equations of $\dot{v_x}$ and $\dot{v_y}$ are also the linearized version of Newton's Second Law, $F = m\dot{v} = mg(\phi + \delta)$, the throttle force is equal to the gravitational force because the trimming point is $\dot{x} = 0$. For $\dot{v_z}$, it is influenced by $P_{avg}(P_{avg} = m\dot{v_z})$.

As a result, it can be separated to 4 subsystems as follows:

$$\begin{cases} \text{sys\_x} \mid \text{input: } \delta_2, \text{ states: } \omega_y, \beta, v_x, x \\ \text{sys\_y} \mid \text{input: } \delta_1, \text{ states: } \omega_x, \alpha, v_y, y \\ \text{sys\_z} \mid \text{input: } P_{avg}, \text{ states: } v_z, z \\ \text{sys\_roll} \mid \text{input: } P_{diff}, \text{ states: } \omega_z, \gamma \end{cases}$$

2 Part 3 | Design MPC Controllers for Each Sub-System

Deliverable 3.1 |

- **Explanation of design procedure that ensures recursive constraint satisfaction.**

In this part, MPC can be implemented on four linearized subsystems to constrain the state x_i and input u_i from step 0 to $N-1$, also the terminal state x_N after step N. The MPC is formulated as follows:

$$\begin{aligned} \min \sum_{i=0}^{N-1} & x_i^T Q x_i + u_i^T R u_i + x_N^T Q_f x_N \\ \text{s.t.} \quad & x_{i+1} = A x_i + B u_i \\ & H_x x_i \leq h_x \\ & H_u u_i \leq h_u \\ & H_f x_N \leq h_f \\ & x_i \in \mathbb{X} \\ & u_i \in \mathbb{U} \end{aligned} \tag{1}$$

where the matrices A and B denote the dynamics of the subsystems, H and h are the constraints of states, input, and the terminal states respectively. The constraints are shown in Table 1 below.

Subsystem	State constraint	Input constraint
sys_x	$\lvert\beta\rvert \leq 5^\circ$ (0.0873 rad)	$\lvert\delta_2\rvert \leq 15^\circ$ (0.26 rad)
sys_y	$\lvert\alpha\rvert \leq 5^\circ$ (0.0873 rad)	$\lvert\delta_1\rvert \leq 15^\circ$ (0.26 rad)
sys_z	-	$50\% \leq P_{avg} \leq 80\%$
sys_roll	-	$\lvert P_{diff}\rvert \leq 20\%$

Table 1: State and input constraints of the system

To ensure recursive feasibility and stability, the LQR control law u = Kx is applied as the terminal maximal invariant set, and makes the terminal cost a Lyapunov function. The corresponding variable K and Q_f can be obtained by using dlqr function in MATLAB.

The discrete-time algebraic Riccati equation for the LQR controller can be computed as follows, the infinite horizon solution P will be used as Q_f:

$$\begin{aligned} P &= Q + A^T P A - A^T P B (R + B^T P B)^{-1} B^T P A \\ K &= -(R + B^T P B)^{-1} B^T P A \\ Q_f &= P \end{aligned}$$

According to the implementation of the equations above, the solver finally returns the first value in the the control optimal sequence u^* , i.e., u_0^*, as an input to the system.

- **Explanation of choice of tuning parameters. (e.g., Q, R, H, terminal components).**

Parameter	sys_x and sys_y	sys_z	sys_roll
H	5	5	5
Q	I_4	$25I_2$	$60I_2$
R	1	1	1

Table 2: Tuning parameters of the MPC

The selected parameters are shown in the Table 2. Horizon H can influence the stability, feasibility, and the invariance. Increasing H can make the performance better as the infinite horizon is the ideal solution, but the computational effort grows as well. It is found that the performance does not change much after $H \geq 5$, so H is chosen as 5 for all subsystems.

Q and R are relevant to the overall cost and the size of the terminal set. To achieve the objective that the settling time is less than eight seconds when starting stationary at five meters from the origin (for x, y and z) or stationary at 45° for roll. Q and R are finally selected as shown in Table 2. It is found that penalizing on sys_x and sys_y does not influence much, so Q and R remains I. Nevertheless, it improves obviously when penalizing on the states of sys_z and sys_roll. Thus, their Q are tuned as 25 and 60 respectively, and R remain I.

The terminal component Q_f is already obtained in the LQR solution.

- **Plot of terminal invariant set for each of the dimensions, and explanation of how they were designed and tuning parameters used.**

The maximum terminal invariant set is designed by intersecting the current set x_i with its pre-set $pre(x_i)$ until convergence. The iteration ends when $x_i = pre(x_i)$, and the obtained set is the terminal invariant set x_f. The preset is formulated as $pre(S) = \{x|F(A+BK_f)x \leq f\}$ if $S := \{x|Fx \leq f\}$, and K_f is obtained from LQR. This visualization of the terminal invariant set for sys_x, y, z, and sys_roll are shown in Figure 1, 2, 3(a), and 3(b). The projection of the sys_x and sys_y are multi-dimension due to higher dimension of states.

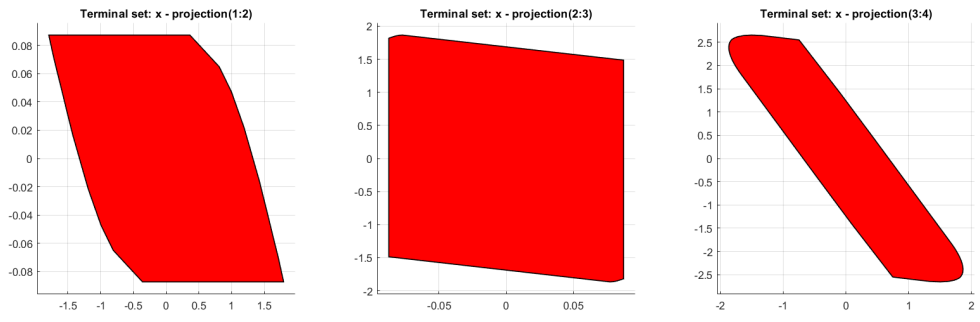


Figure 1: Projection of terminal invariant set of sys_x

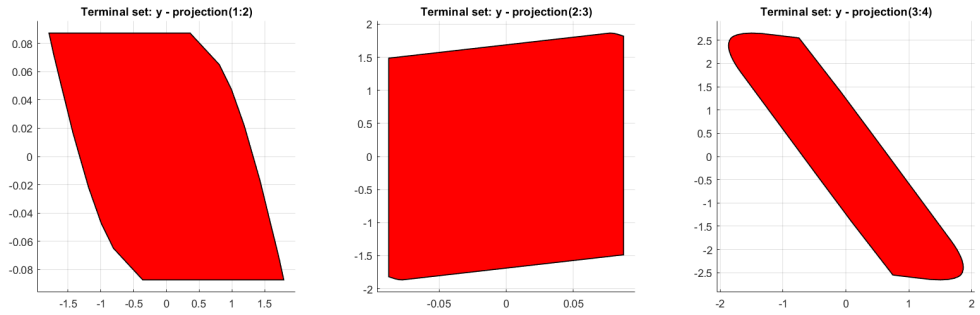
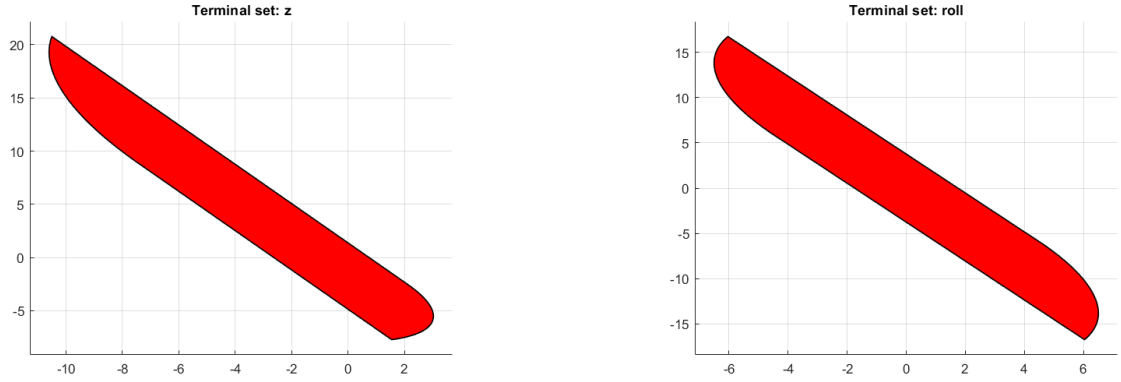


Figure 2: Projection of terminal invariant set of sys_y



(a) (b)

Figure 3: Terminal invariant set of sys_z and sys_roll

- **Plot for each dimension starting stationary at 5 meters from the origin (for x, y and z) or stationary at** $45°$ **for roll.**

The performance of each subsystem under linear MPC are shown in Figure 4, 5, 6, and 7. All of the states converge to the origin as expected.

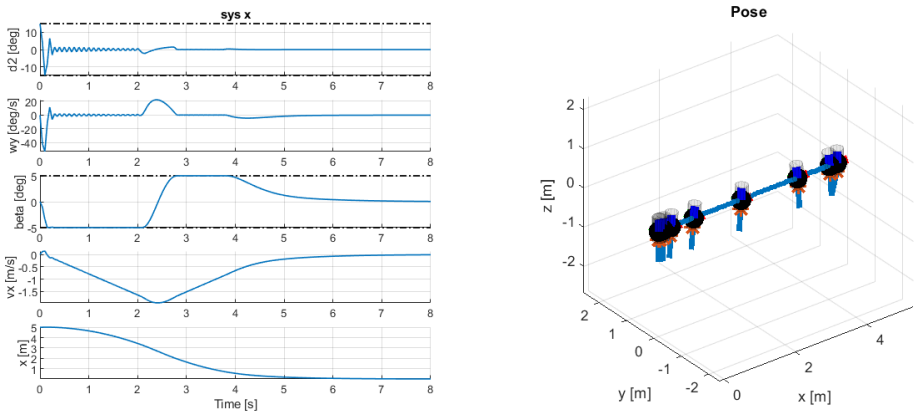


Figure 4: States and input variations of sys_x

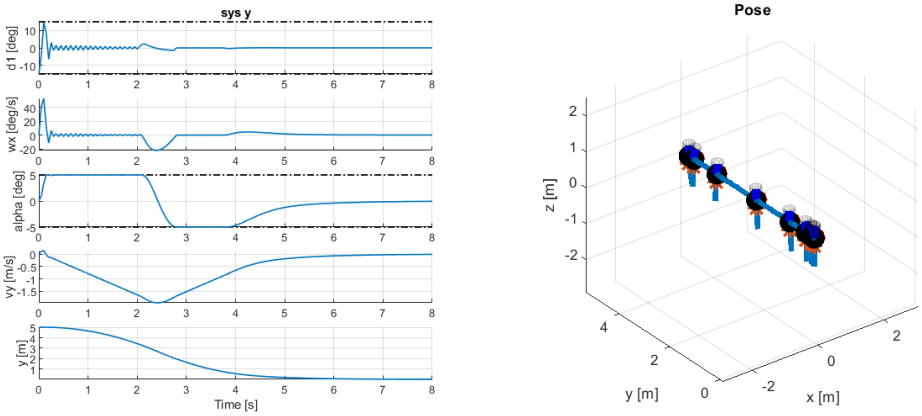


Figure 5: States and input variations of sys_y

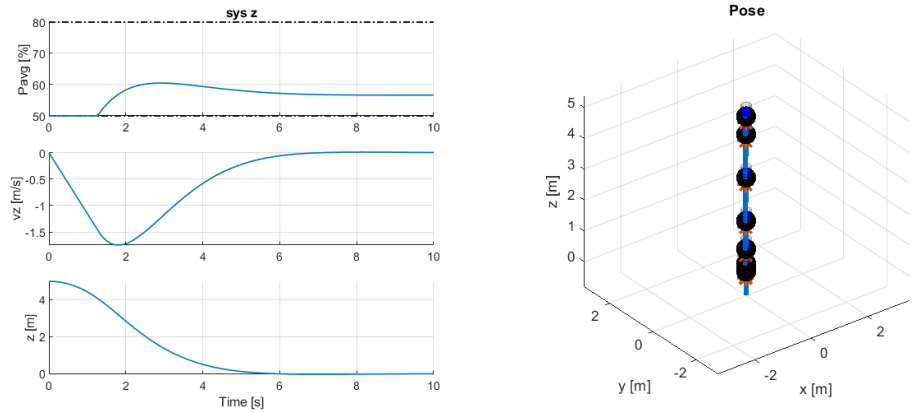


Figure 6: States and input variations of sys_z

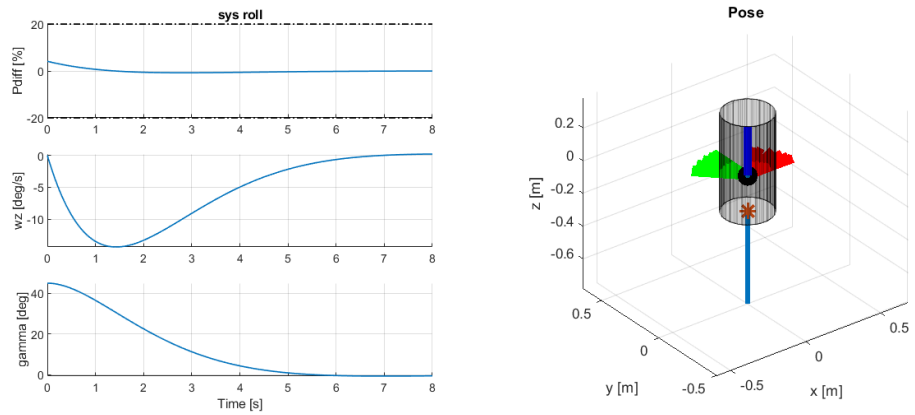


Figure 7: States and input variations of sys_roll

Deliverable 3.2 |

- **Explanation of your design procedure and choice of tuning parameters.**

For the tracking problem, we need to set a target for tracking. The tracking target should be a steady state. We calculate it by solving the following optimization problem. Where u_s is the input x_s is the steady state and r is the reference target. During simulation, we can always obtain a feasible solution for this optimization problem.

$$\begin{aligned} \min_{x_s,u_s} \quad & u_s^T u_s \\ s.t. \quad & x_s = Ax_s + Bu_s \\ & r = Cx_s + Du_s \end{aligned} \tag{2}$$

The intuition of this optimization problem is to find the minimum control cost, so that the system output y is equal to the reference value r and the rocket will stay at x_s with the given control input.

Another change that need to be made is on the optimizer part. After we obtain the x_s and u_s, we need to add them into our model. The model is shown as follows, here, we use the Delta-Formation. Where $\Delta x = x - x_s$, $\Delta u = u - u_s$

$$\begin{aligned} \min_{u} \quad & \sum_{i=0}^{N-1} \Delta x_i^T Q \Delta x_i + \Delta u_i^T R \Delta u_i + \Delta x_N^T Q_f \Delta x_N \\ s.t. \quad & \Delta x_{i+1} = A\Delta x_i + B\Delta u_i \\ & H_x \Delta x_i \leq h_x \\ & H_u \Delta u_i \leq h_u \\ & H_f \Delta x_N \leq h_f \\ & \Delta x_i \in \mathbb{X} \\ & \Delta u_i \in \mathbb{U} \end{aligned} \tag{3}$$

For parameters, we tried the parameter we used in the `Deliverable 3-1` and it works quite well.

- **Plot for each dimension of the system starting at the origin and tracking a reference to -5 meters from the origin (for x, y and z) and to 45◦ for roll.**

The performance of the system on each dimension is shown in Figures 8, 9, 10 and 11. From the figures, we can notice that the system can converge to the reference point very quickly with almost no overshooting.

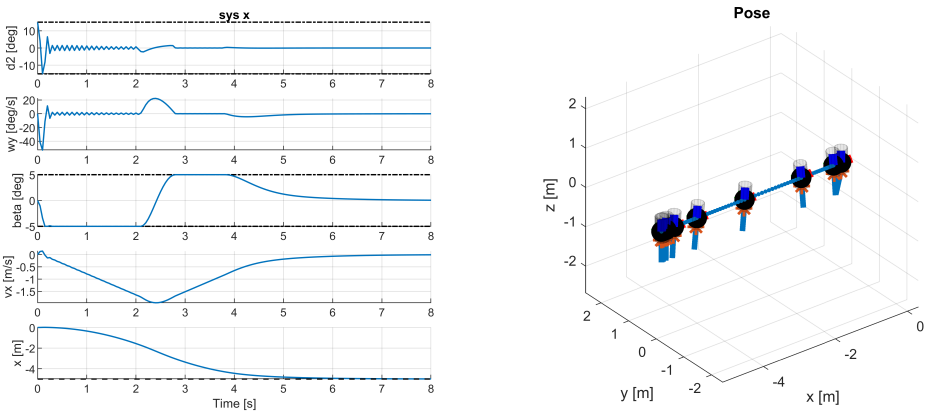


Figure 8: Deliverable 3-2: System X

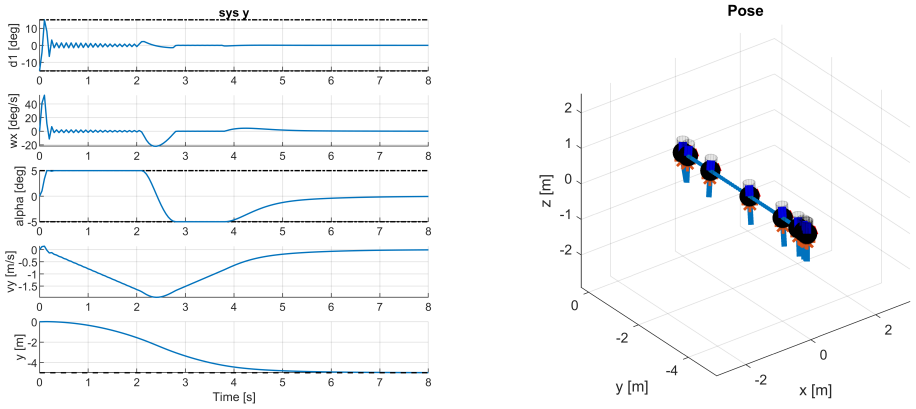


Figure 9: Deliverable 3-2: System Y

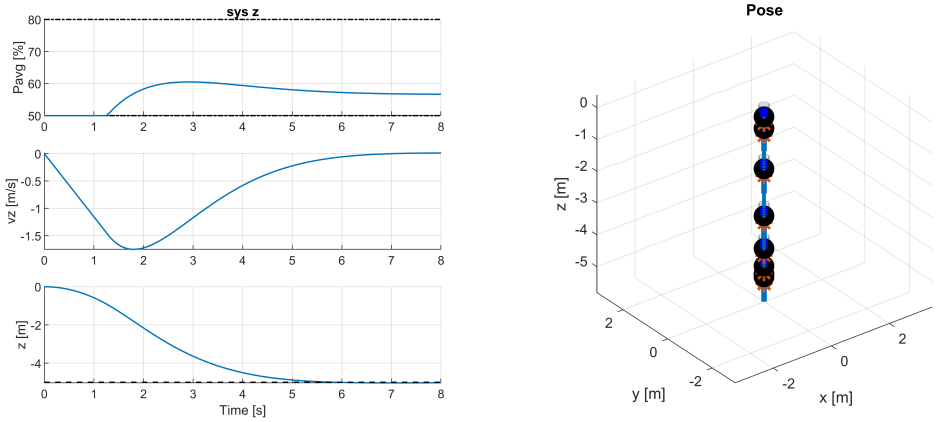


Figure 10: Deliverable 3-2: System Z

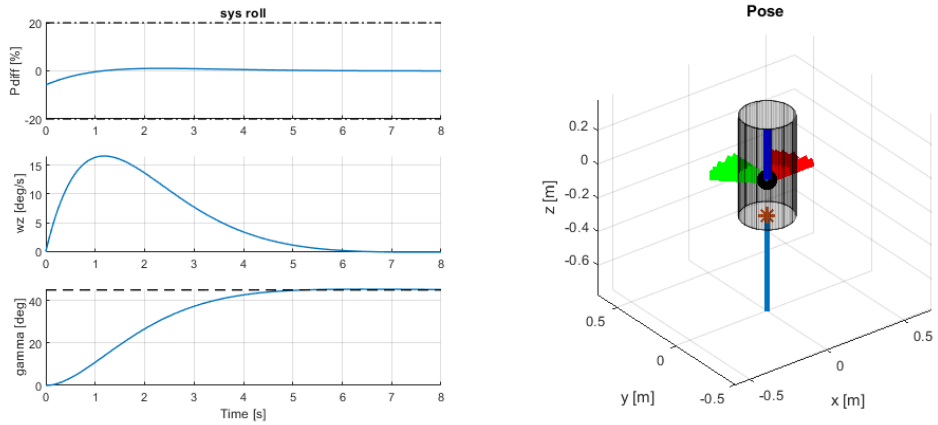


Figure 11: Deliverable 3-2: System Roll

- **Matlab code for the four controllers, and code to produce the plots in the previous step.**

The code is submitted in the attachment of this project.

3 Part 4 | Simulation with Nonlinear Rocket

Deliverable 4.1 |

- **A plot of your controllers successfully tracking the MPC reference path and reference roll angle. If your tracking performance is not good, explain how you adapt your tuning to improve it.**

We first test the parameter we used in `Deliverable 3.1` and `Deliverable 3.2` because the rocket reached reference point quickly and with almost no overshooting in the previous questions. The trajectory is shown in Figure 12 and the performance of each dimension is shown in Figure 13.

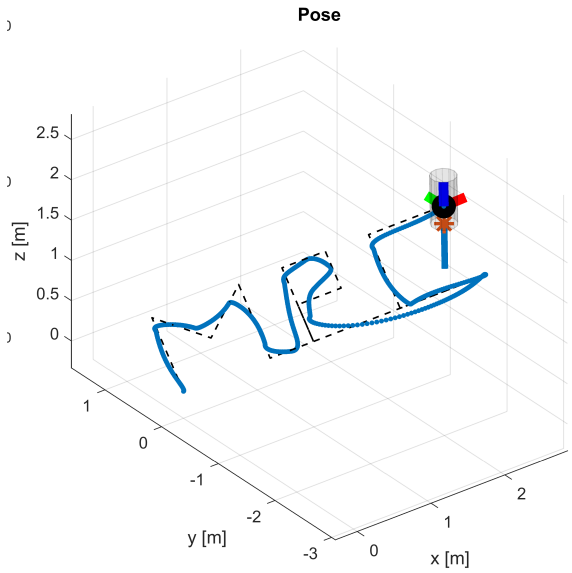


Figure 12: Parameter 1: Trace

The performance of the rocket is not good, which means we need to find another group of parameters. A direct and simplest method is to increase the weight of x, y, z and γ. When we change them to a larger value, the result shows that the system may have `infeasible or unbounded solution`. An example of an infeasible solution graph is shown in Figure 14. From the figure, we think the main reason for the infeasibility is due to subsystem x and subsystem y:

1. the reference point of subsystem x and y has a rapid increase at the stopping time and then the `wy` has a rapid change which violates the constraints.
2. Instead of changing all subsystems together, we change each subsystem one by one, and the result shows that the feasible set of weight of x and y is much smaller than other subsystems

One way to change the infeasibility is to increase the weight of other components like angular speed and velocity because it will push the optimizer to decrease their magnitude. One drawback is that increasing the weight of x, y, z and γ may induce overshooting, which means when the system is at the reference position, the corresponding speed may not be zero. Based on these reasons, we optimize our parameter in the following way:

1. First we increase the weight of all other variables except the weight of the control input. By doing that, we can push variables like velocity to zero when the system reach the reference position.

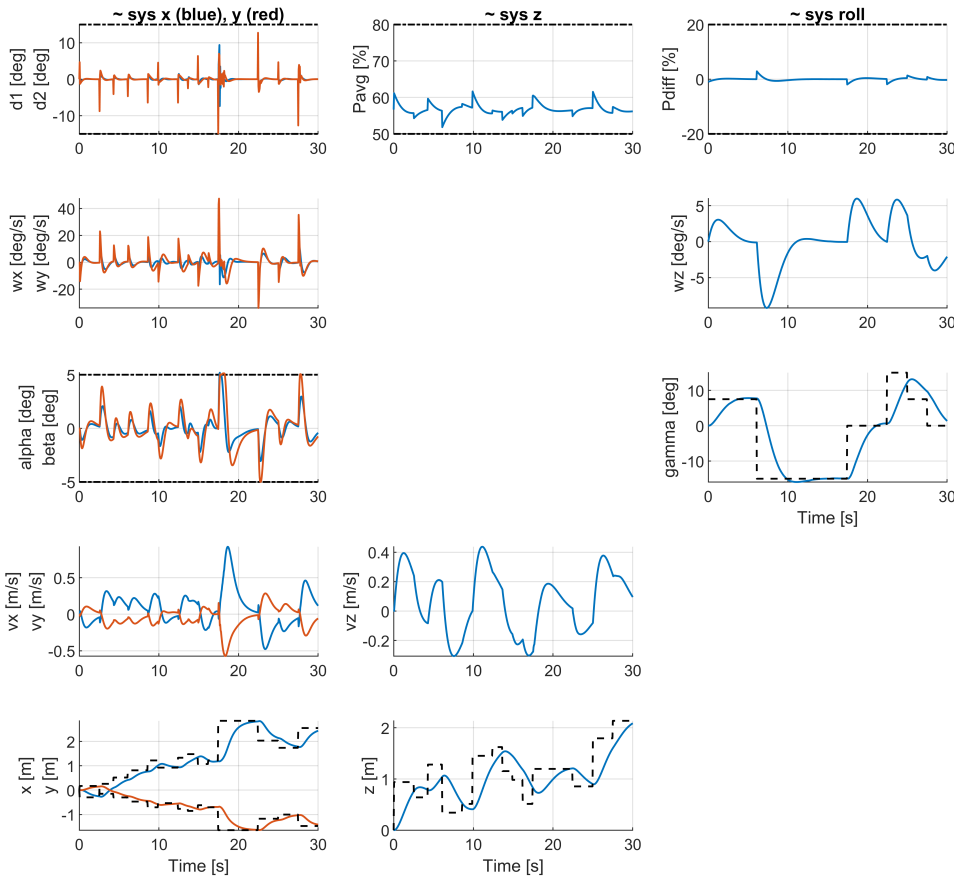


Figure 13: Parameter 1: Each Dimension

2. Then we increase the weight of x, y, z, and γ.

Finally, we find another group of parameters shown in Table 3. The `diag` means a diagonal matrix.

Parameter	sys_x and sys_y	sys_z	sys_roll
H	5	5	5
Q	diag(3,3,3,6)	diag(3,150)	diag(3,180)
R	1	1	1

Table 3: 4-1: Tuning parameters of the MPC

The performance of this group of parameter is shown in Figure 15 and Figure 16. It can be seen that the overall trajectory is much better and the accuracy of each dimension is improved.

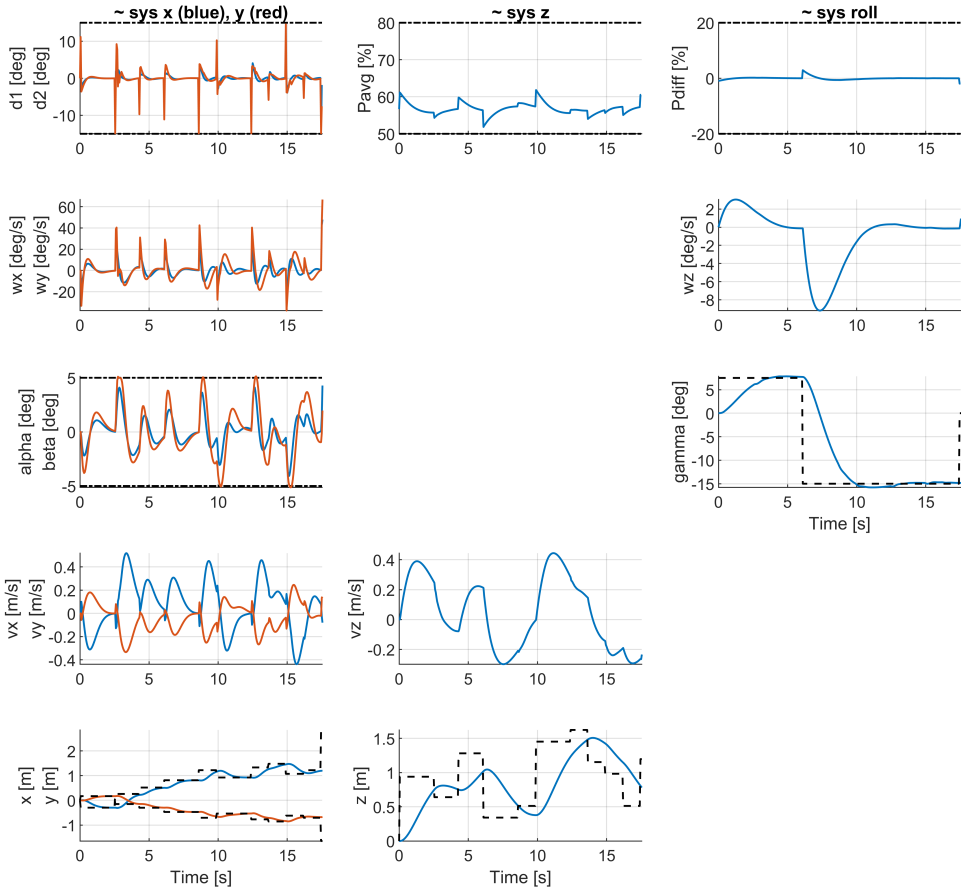


Figure 14: Infeasible Solution Parameter

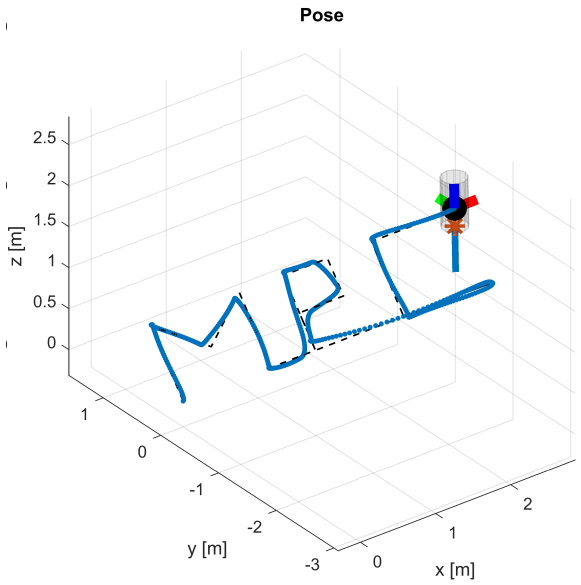


Figure 15: 4-1: Tuned Parameter: Trace

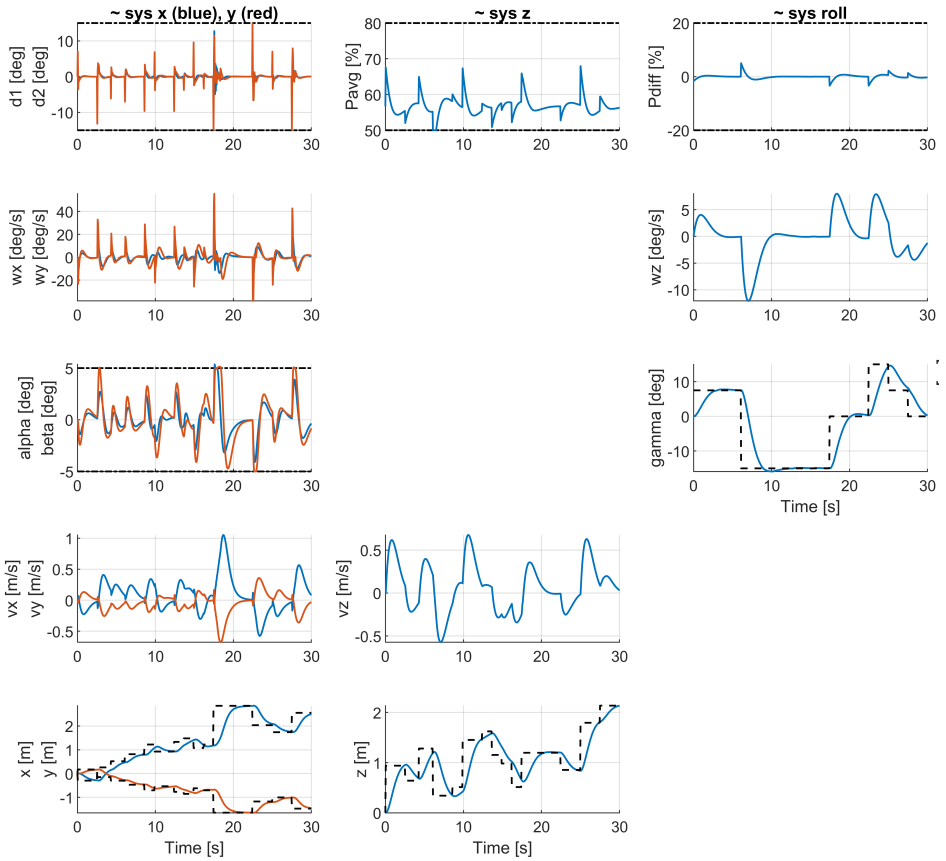


Figure 16: 4-1: Tuned Parameter: Plot

However, there are still some flaws in the trajectory. We think there are mainly two reasons:

- The setup time is long, but some parts of the trajectory require rapid change. The letter "P" is a good illustration. Because of the setup time, the right part of the upper "O" in "P" does not obtain a right angle trace. The same thing happens in the bottom "I" in "P", two parts of the trace do not converge to a single line because of setup time.
- The system is a linear system, but now we use a linearized system to design our controller. Which means we have some accuracy trade-off. Besides, generally, the further the system leaves away from the trimming point, the accuracy will be lower.

4 Part 5 | Offset-Free Tracking

Deliverable 5.1 |

Goal: Design a controller to reject a constant disturbance and track setpoint references with no offset.

In the z-direction, we use an augmented model to depict the dynamics of the system as

$$\begin{aligned} x_{k+1} &= Ax_k + Bu_k + B_d d_k \\ d_{k+1} &= d_k \\ y_k &= Cx_k + d_k \end{aligned} \tag{4}$$

where $B_d = B, C = I$. If $\begin{bmatrix} A-I & B_d \\ C & 0 \end{bmatrix}$ has full column rank, we can use an observer to estimate the states x and the disturbance d based on the augmented model

$$\begin{bmatrix} \hat{x} \\ \hat{d}_{k+1} \end{bmatrix} = \begin{bmatrix} A & B \\ 0 & I \end{bmatrix} \begin{bmatrix} \hat{x}_k \\ \hat{d}_k \end{bmatrix} + \begin{bmatrix} B \\ 0 \end{bmatrix} u_k + \begin{bmatrix} L_x \\ L_d \end{bmatrix} (C\hat{x}_k - y_k) \tag{5}$$

where $\hat{x}, \hat{d}$ are the estimates of the state and disturbance. We can obtain the error dynamics as

$$\begin{aligned} \begin{bmatrix} x_{k+1} - \hat{x}_{k+1} \\ d_{k+1} - \hat{d}_{k+1} \end{bmatrix} &= \left(\begin{bmatrix} A & B \\ 0 & I \end{bmatrix} + \begin{bmatrix} L_x \\ L_d \end{bmatrix} \begin{bmatrix} C & 0 \end{bmatrix} \right) \begin{bmatrix} x_k - \hat{x}_k \\ d_k - \hat{d}_k \end{bmatrix} \\ &= (\hat{A} + \hat{L}\hat{C}) \begin{bmatrix} x_k - \hat{x}_k \\ d_k - \hat{d}_k \end{bmatrix} \end{aligned} \tag{6}$$

We should choose $L = \begin{bmatrix} L_x \\ L_d \end{bmatrix}$ such that the error dynamics are stable and converge to zero, which is equivalent to designing a K such that $(\hat{A}' + \hat{C}'K)$ is stable. This can be easily done by pole placement, the eigenvalues must be strictly less than 1.

- **Explanation of your design procedure and choice of tuning parameters.**

While small norms of eigenvalues will lead to a fast-speed estimation process, it might increase the initial overshoot of estimation. So here we use eigenvalues as (0.2, 0.23, 0.25). For the other MPC parameters, we use the same settings as the previous section in Table 3.

Then, we need to calculate the steady-state target (x_s, u_s) using the disturbance estimate by solving

$$\begin{aligned} \min_{u_s} \quad & u_s^T u_s \\ s.t. \quad & \begin{bmatrix} I-A & -B \\ C & 0 \end{bmatrix} \begin{bmatrix} x_s \\ u_s \end{bmatrix} = \begin{bmatrix} B_d \hat{d} \\ r \end{bmatrix} \\ & H_u u_s \leq h_u \end{aligned} \tag{7}$$

Then the rest of this part is almost the same procedure as we solved the MPC reference tracking problem in Delta-formulation(in Deliverable 3.2) using the disturbance estimate.

- **Plot showing the impact of changing the mass on your original controller from Part 4, and then another plot showing that your controller now achieves offset-free tracking.**

Without estimator: If we change the mass of the rocket and run the controller in Part 4, it shows an off-set along z-direction in Figure 17 and Figure 18. It cannot reject the impact of the constant disturbance. (Please uncomment Line 41, and comment Line 44 in Deliverable_5_1.m.)

With estimator: When we use the offset-free controller, as it shows in Figure Fig. 19 and Figure 20, the reference trajectory are achieved without offset. (Please uncomment Line 44, and comment Line 41 in Deliverable_5_1.m.)

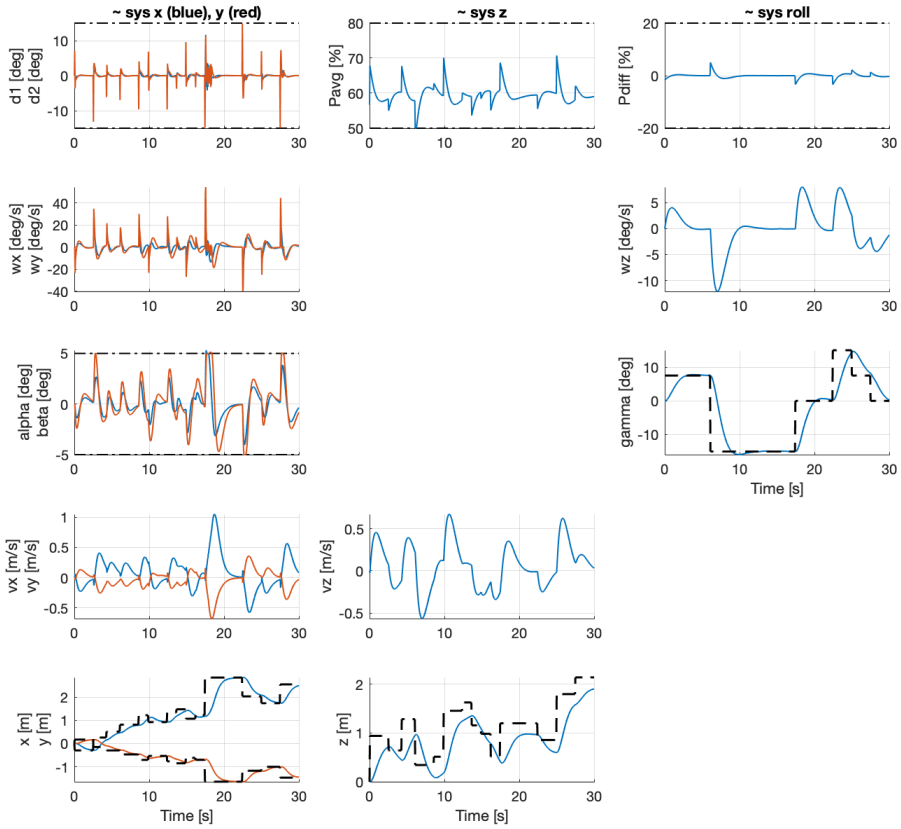


Figure 17: Plot of states with the same controller in Part 4 (without estimator)

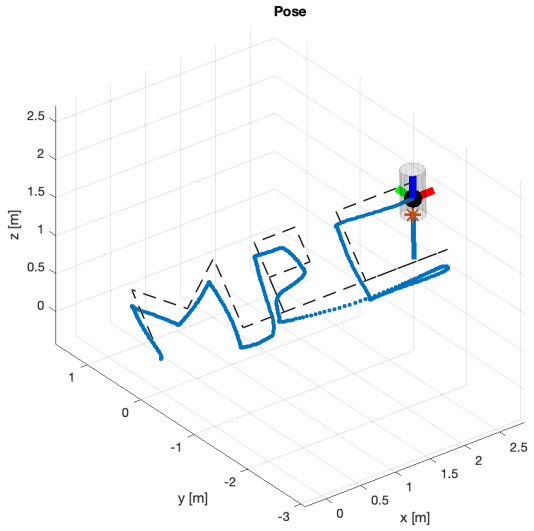


Figure 18: Trajectory of rocket with a controller in Part 4 (without estimator)

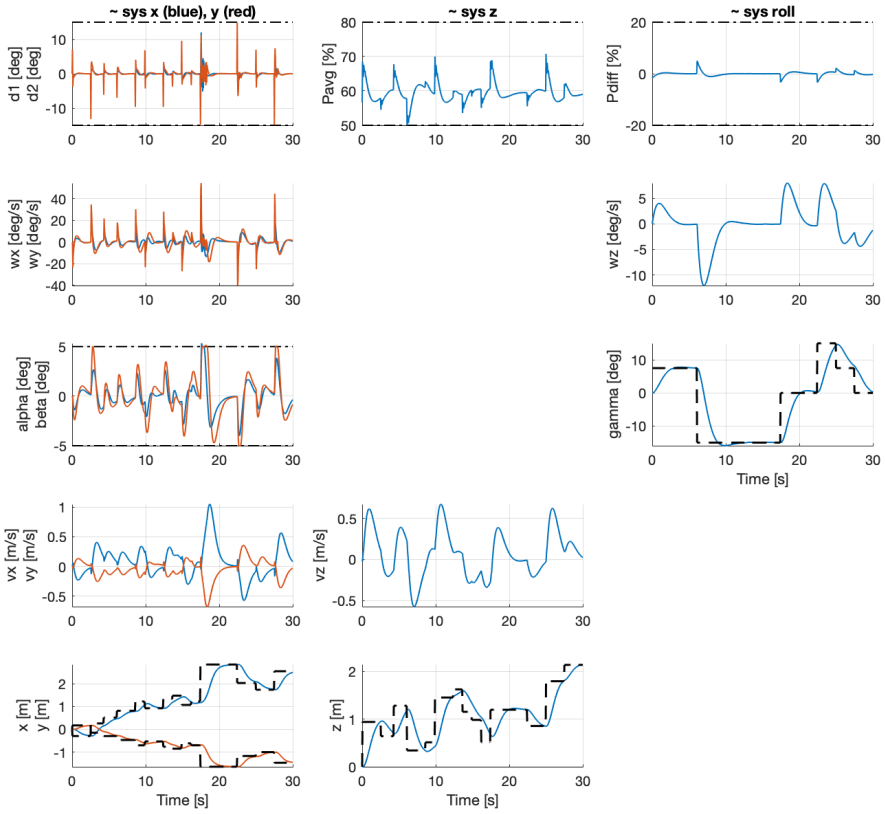


Figure 19: Plot of states with offset-free controller(with estimator)

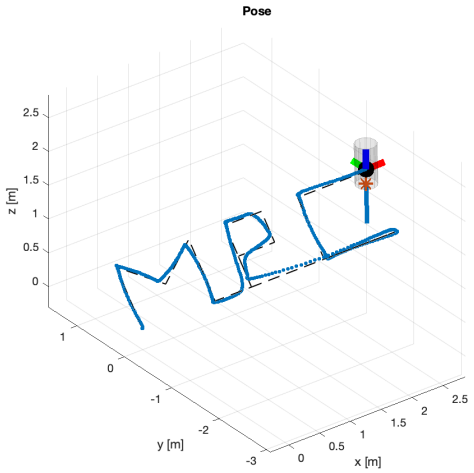


Figure 20: Trajectory of rocket with offset-free controller(with estimator)

5 Part 6 | Nonlinear MPC

Deliverable 6.1 |

- **Explanation of your design procedure and choice of tuning parameters.**

Goal: Develop a nonlinear MPC controller for the rocket.

For a nonlinear MPC controller, we do not need to decompose the rocket into four sub-systems anymore. Here we choose Runge-Kutta 4 integrator to approximate this nonlinear system. Considering the ODE

$$\begin{aligned} \dot{x} &= f(x,u) \\ x_{k+1} &= x_k + h(\frac{k_1}{6} + \frac{k_2}{3} + \frac{k_3}{3} + \frac{k_4}{6}) \end{aligned} \tag{8}$$

where

$$\begin{aligned} k_1 &= f(t_k, x_k) \\ k_2 &= f(t_k + \frac{h}{2}, x_k + \frac{h}{2}k_1) \\ k_3 &= f(t_k + \frac{h}{2}, x_k + \frac{h}{2}k_2) \\ k_4 &= f(t_k + h, x_k + hk_3) \end{aligned} \tag{9}$$

To achieve a good approximation of states, we have to set up much stricter constraints for a linear MPC controller. However, the nonlinear controller covers the whole state space, so we can remove all these 'virtual' constraints. But to avoid violating the singularity of the model, we only constrain $|\beta| \leq 85°$.

Following the hints, as we use a shorter horizon to improve computation speed, we linearize the system to help us find a proper terminal cost. (In NMPC_Control.m, please uncomment Line 27-34 and Line 89, also comment Line 83 to implement.) We discretized the linear system and tuned the corresponding entries of positionp_x, p_y, p_z and γ in Q_f with a higher value to improve the tracking accuracy. While during the simulation, it is usually hard to tune both Q_f and Q. Also, with a relatively rough linearization of the system, when we use S instead of Q as the terminal cost, the optimizer usually cannot return a feasible solution. So we changed it to only tune parameter values in Q.

NL MPC controller with max. $|\gamma_{\mathbf{ref}}| = 15°$

As we mentioned before, in the nonlinear MPC implementation, the parameters of the cost function are replaced by a series of weights on each states. It is applied on not only the recursive horizon but also the terminal cost. The weight of the states for tuning the controller with $|\gamma_{ref,max}| = 15°$ is listed below.

$$W\begin{bmatrix}\omega_x\\ \omega_y\\ \omega_z\end{bmatrix} = \begin{bmatrix}20\\ 10\\ 200\end{bmatrix} \quad W\begin{bmatrix}\alpha\\ \beta\\ \gamma\end{bmatrix} = \begin{bmatrix}40\\ 40\\ 1000\end{bmatrix} \quad W\begin{bmatrix}v_x\\ v_y\\ v_z\end{bmatrix} = \begin{bmatrix}200\\ 100\\ 200\end{bmatrix} \quad W\begin{bmatrix}p_x\\ p_y\\ p_z\end{bmatrix} = \begin{bmatrix}800\\ 800\\ 800\end{bmatrix} \tag{10}$$

The weight of input is set to be I, and the horizon H is shortened to 1 due to speed up the simulation. The weights of positions p and euler angle γ are first tuned larger because they are the decisive variables for tracking. Later, the weight of ω_z is tuned higher, which makes the trace smoother. Finally, the other parameters are fine-tuned to make the tracking more precise. The plots are shown as Figure 21 and Figure 22.

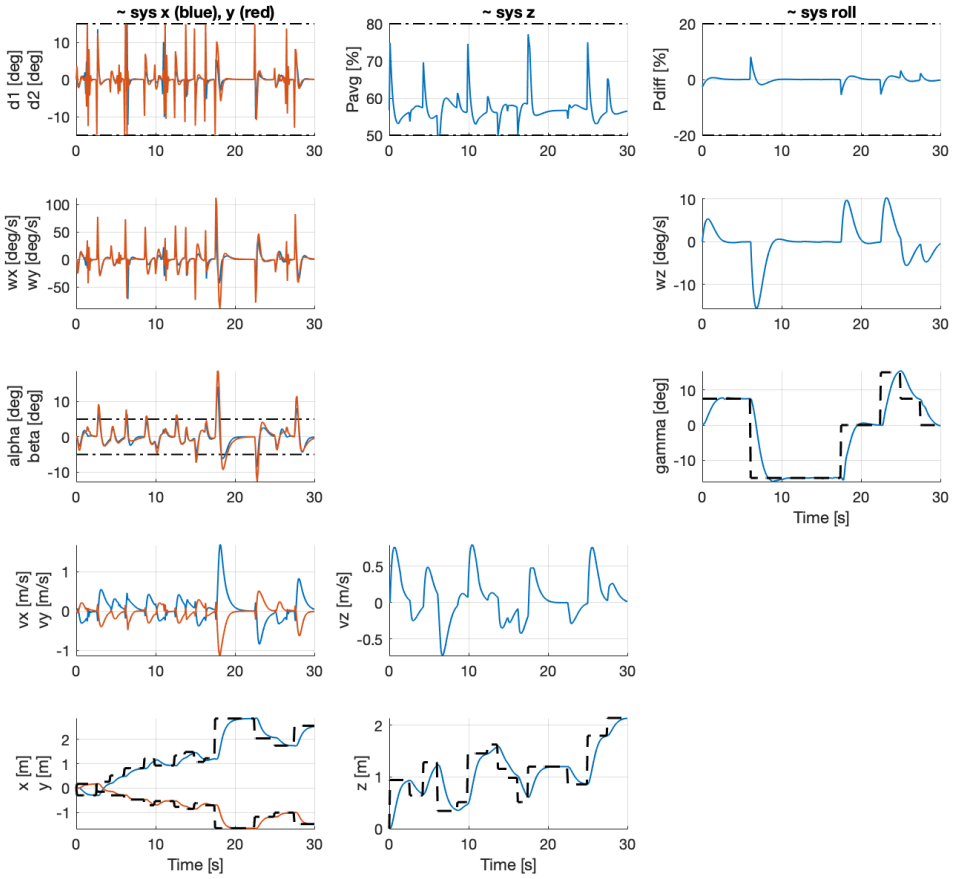


Figure 21: Nonlinear MPC plots with max. $|\gamma_{ref}| = 15°$

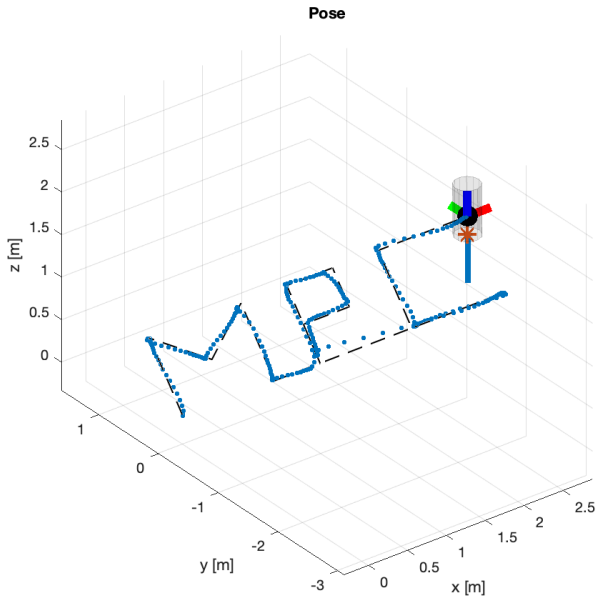


Figure 22: Nonlinear MPC trace with max. $|\gamma_{ref}| = 15°$

NMPC controller with max. $|\gamma_{ref}| = 50°$

When we change the maximum reference value of γ from 15° to 50°, we run the NMPC simulation by uncomment Line 16 and 17 and comment Line 13 in Deliverable_6_1.m. As it is shown in Figure 23. Generally speaking, the tracking performance is good. But we can observe that because of this extremer reference value, there is a position deviation in the letter "M" along x and y-directions. So

we increase the corresponding weights of p_x and p_y. While from the physical perspective, ω_z and γ contribute to the roll dynamics, so they are set to be larger to track the γ_{ref} better. However, the penalty on the γ cannot be too large because the dramatic rolling dynamics will influence the other motions. The weights are choosen as listed below(to run this, please uncomment Line 73 - 91 and comment Line 52 - 69 in NMPC_Control.m.) and the final performance and trace are shown in the Figure 24 and Figure 25.

$$W\begin{bmatrix}\omega_x\\ \omega_y\\ \omega_z\end{bmatrix}=\begin{bmatrix}20\\ 10\\ 100\end{bmatrix}\quad W\begin{bmatrix}\alpha\\ \beta\\ \gamma\end{bmatrix}=\begin{bmatrix}40\\ 40\\ 400\end{bmatrix}\quad W\begin{bmatrix}v_x\\ v_y\\ v_z\end{bmatrix}=\begin{bmatrix}200\\ 100\\ 200\end{bmatrix}\quad W\begin{bmatrix}p_x\\ p_y\\ p_z\end{bmatrix}=\begin{bmatrix}1000\\ 1000\\ 800\end{bmatrix} \tag{11}$$

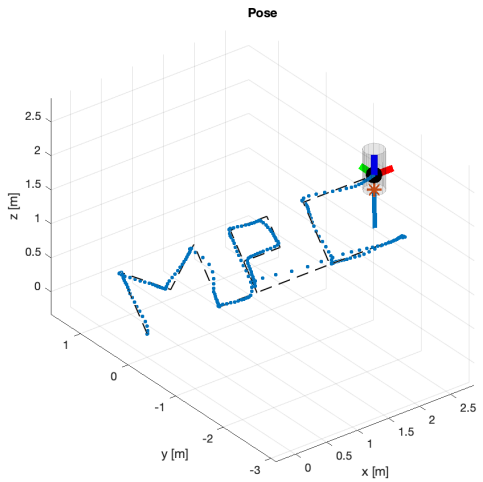


Figure 23: untuned Nonlinear MPC trace with max. $|\gamma_{ref}| = 50°$

Linear MPC controller with $|\gamma_{ref}| = 50°$

After changing to $|\gamma_{ref}| = 50°$ and keeping the linear controller's setting like the one in Deliverable 4.1, the optimizer says now the problem is infeasible at time 12, as shown in Figure 26. From the figure, we think the main reason for the infeasibility is because α and γ. One thing to be explained is that from the curve of α and γ we can see there are several sample points their value is out of constraints but the system is still working. The possible reason is that when computing the control input, we use the linear model however when simulating, the program uses nonlinear dynamic. This means in the optimizer, the trace may meet the constraints while in simulation, it will violate the dynamics.

Based on the analysis, we increase the weight of α and γ. Besides, we also changed the angular speed related to w_x and w_y based on the hint of Deliverable 4.1. The final Qs and Rs we use are in the table 4. The performance of the group of the parameter is shown in Figure 27 and Figure 28. The system can generate feasible traces, however, the tracking performance is not good. It may due to the flaws of linearization and will be discussed later.

Parameter	sys_x and sys_y	sys_z	sys_roll
H	3	5	5
Q	diag(70,150,3,10)	diag(3,150)	diag(3,180)
R	1	1	1

Table 4: 6-1: Tuning parameters of Linear MPC

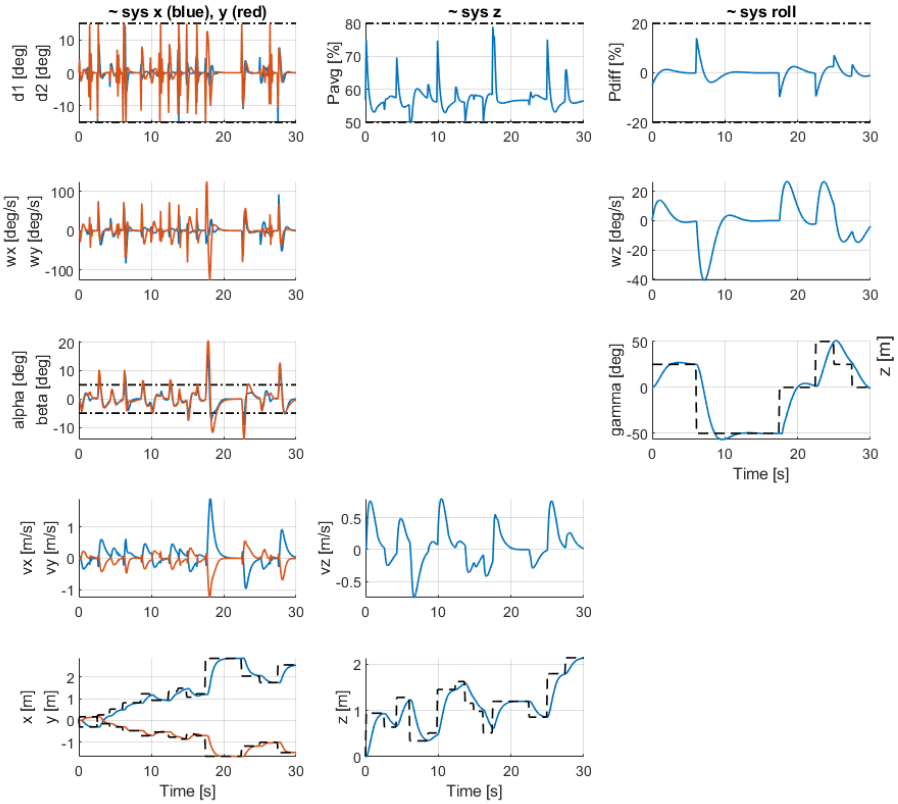


Figure 24: tuned Nonlinear MPC plots with max. $|\gamma_{ref}| = 50°$

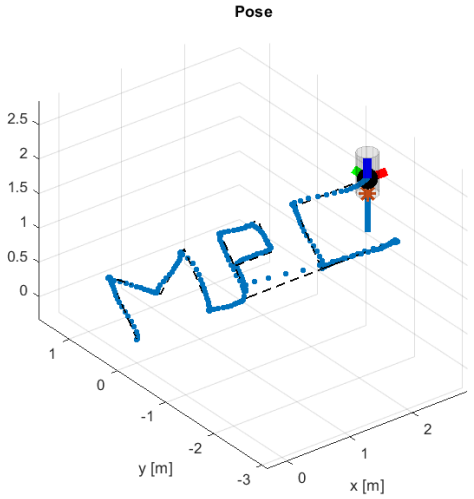


Figure 25: tuned Nonlinear MPC trace with max. $|\gamma_{ref}| = 50°$

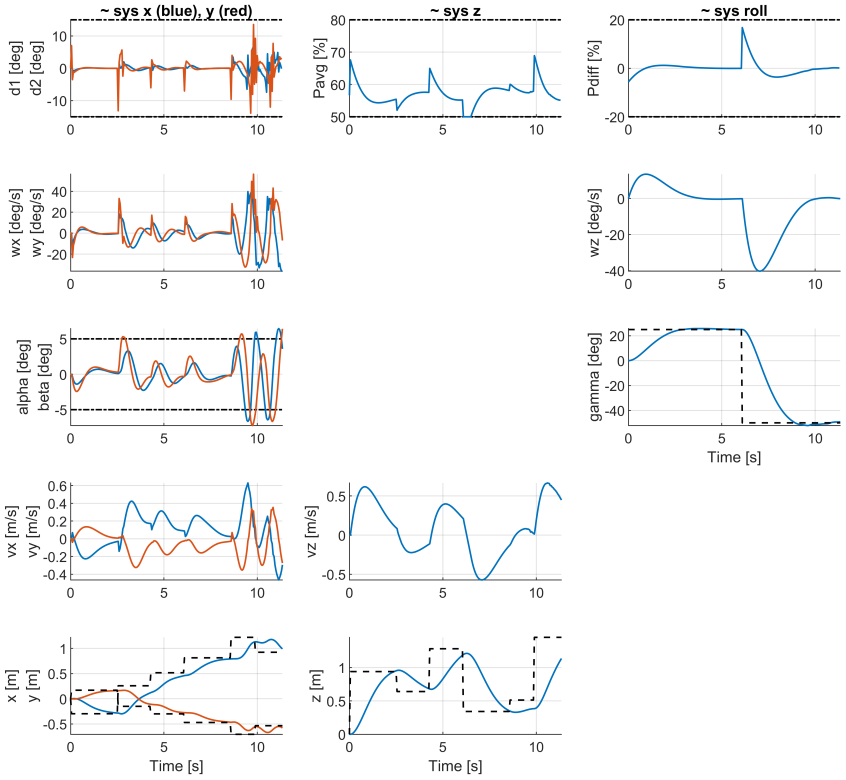


Figure 26: Linear Model Roll 50 Infeasible Plots

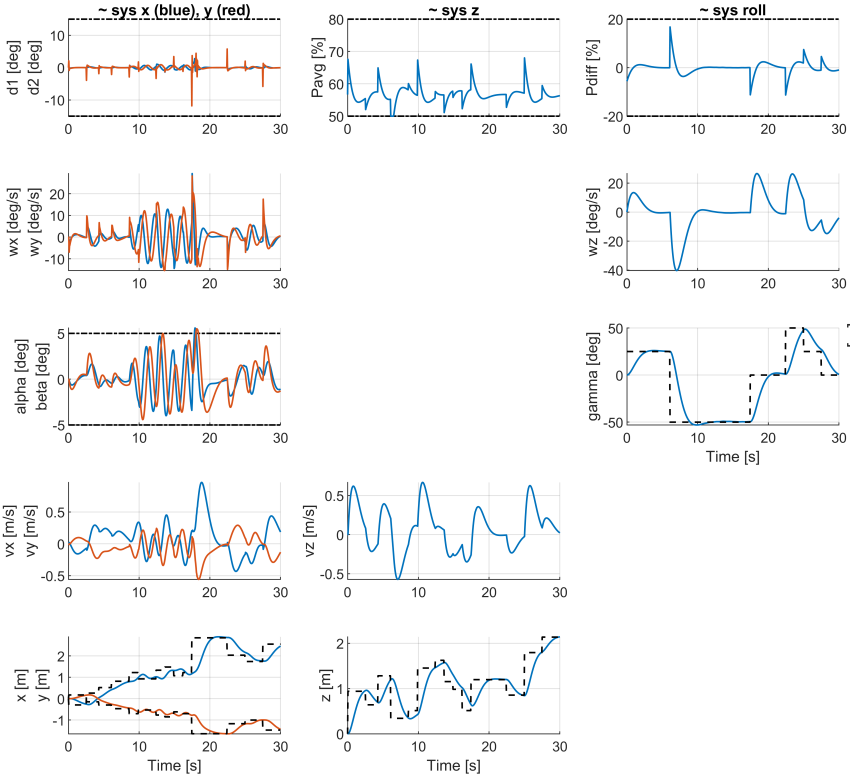


Figure 27: Linear Model Roll 50 Feasible Plots

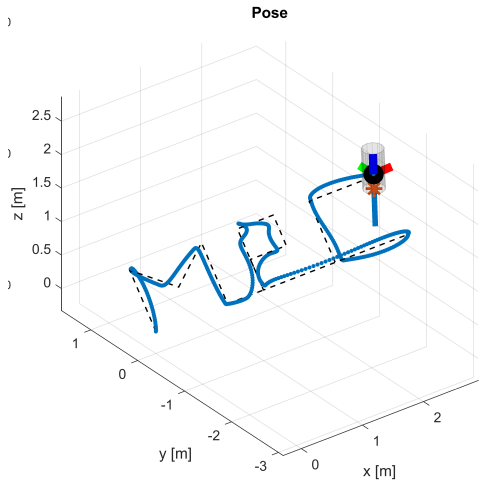


Figure 28: Linear Model Roll 50 Feasible Trace

- **Discuss the pros and cons of your nonlinear controller vs the linear ones you developed earlier.**

Pros: Firstly, nonlinear system does not decompose the system and cover the whole state. Generally speaking, it can depict the system dynamics preciser than a linear one. What's more, compared to linear controller, NMPC can loosen the constrains since the linear approximation is no longer required. In this part, when we change the maximum reference value of γ, NMPC can still achieve a very good tracking performance, while linear one might have an infeasible result because of this much extremer reference. Therefore with a proper choice of parameter settings, we can obtain better performance than a linear controller.

Cons: The computation time is longer than the linear one. Hence in practice, we should consider whether a NMPC can be implemented when given the timescale of the rocket. If the computing time is longer than the acceptable time step of the rocket, it will have a big impact on the performance. As we known, linear controller is usually less time-consuming. In addition, for NMPC, we might only obtain a local minimum or there can be multiple optimal solutions existing which are also challenging issues.