Model Predictive Control Design

ASSIGNMENT A5

Hayden Gomes & Ben Russell
GROUP 4 DATE: 16/02/2024

INTRODUCTION

This report outlines the design of a Model Predictive Controller for an aircraft. The state variables, which represent the current state of the aircraft, include:

Angle of attack (α) - This is the angle between the aircraft's longitudinal axis and the relative wind direction.

Pitch rate (q) - The rate of change of pitch angle over time.

Pitch angle (θ) - The angle between the aircraft's longitudinal axis and the horizontal plane.

The manipulated input, which the control system will adjust to achieve desired pitch behaviour, is the elevator deflection angle (δ). The elevator is a control surface on the aircraft's tail that controls pitch by changing its angle of deflection.

The given system dynamics represent the evolution of the aircraft's angle of attack (α), pitch rate (q), and pitch angle (θ) over time. It's described by a state transition equation where the next state depends on the current state and the input provided by the elevator deflection angle (δ). The coefficients in the transition matrix and the input matrix determine how each state variable changes with respect to the others and the effect of the elevator deflection angle.

This dynamic model, derived with a sampling time of 0.05 seconds, assumes that there are no errors in the measurements of the angle of attack, pitch rate, and pitch angle. This means that the control system can accurately sense and utilize these state variables without any measurement uncertainty.

The system dynamics are given by:

$$\begin{bmatrix} \alpha_{t+1} \\ q_{t+1} \\ \theta_{t+1} \end{bmatrix} = \begin{bmatrix} 0.9835000 & 2.782 & 0 \\ -0.0006821 & 0.978 & 0 \\ -0.0009730 & 2.804 & 1 \end{bmatrix} \begin{bmatrix} \alpha_t \\ q_t \\ \theta_t \end{bmatrix} + \begin{bmatrix} 0.01293 \\ 0.00100 \\ 0.001425 \end{bmatrix} \delta_t \qquad (1)$$

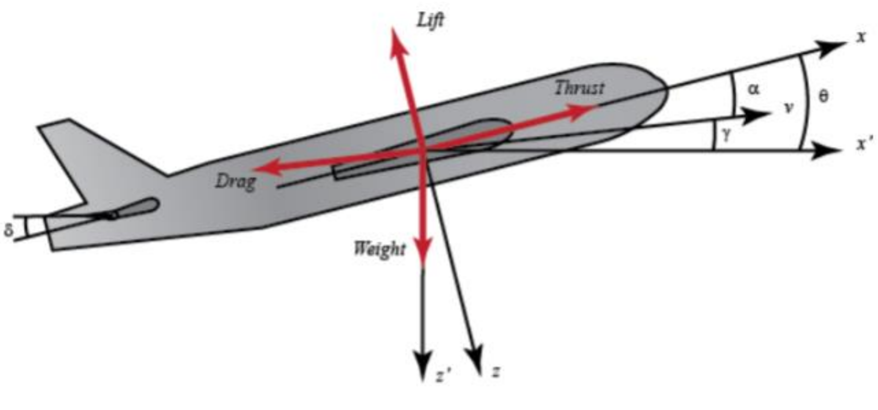


Figure 1: Illustration of an aircraft with its pitch angle, θ, the angle of attack, α, and the deflection angle of the elevators, δ. The angle γ = θ − α is known as the slope.

Methodology

The code used to formulate the MPC can be found via the following link:

https://github.com/ben120-web/Model-Predictive-Control-for-aircraft-pitchDesign/blob/main/MPC_Controller.py

MPC FORMULATION

The Model Predictive Control (MPC) approach implemented in the provided script is designed for controlling the pitch dynamics of an aircraft. The system is represented by the linear state-space model:

$$y_{t+1} = Ax_t + Bu_t \quad (2)$$

where:

- $x_t \in R^3$ is the state vector at time *t*, including the angle of attack (α), pitch rate (q), and pitch angle (ϑ).
- $u_t \in R$ is the control input at time *t*, representing the elevator deflection angle.
- *A* and *B* are the system matrices defining the dynamics.

The MPC seeks to minimize a cost function over a finite prediction horizon *N*, subject to system dynamics and constraints on states and control inputs. The cost function is given by:

$$J = \sum_{k=0}^{N-1}(x_k^T Q x_k + u_k^T R u_k) \quad (3)$$

where *Q* and *R* are weighting matrices for states and control inputs, respectively.

DESIGN CHOICE

- **Cost Function Parameters**: *Q = diag(1000,0,0)* emphasizes minimizing deviations in the angle of attack while de-emphasizing pitch rate and angle. *R=1* balances control effort.
- **Prediction Horizon (*N*)**: Chosen as 10 steps, balancing computational complexity and the need to anticipate future system behaviour.
- **Constraints**: Include bounds on the elevator deflection angle (δ), angle of attack (α), pitch angle (ϑ), and pitch rate (q) to ensure safe operation.

STANDARD FORM

The MPC problem is brought to the standard quadratic programming form:

$$min_u \frac{1}{2} U^T H U + F^T U \quad (4)$$

subject to linear constraints $GU \leq h$, where:

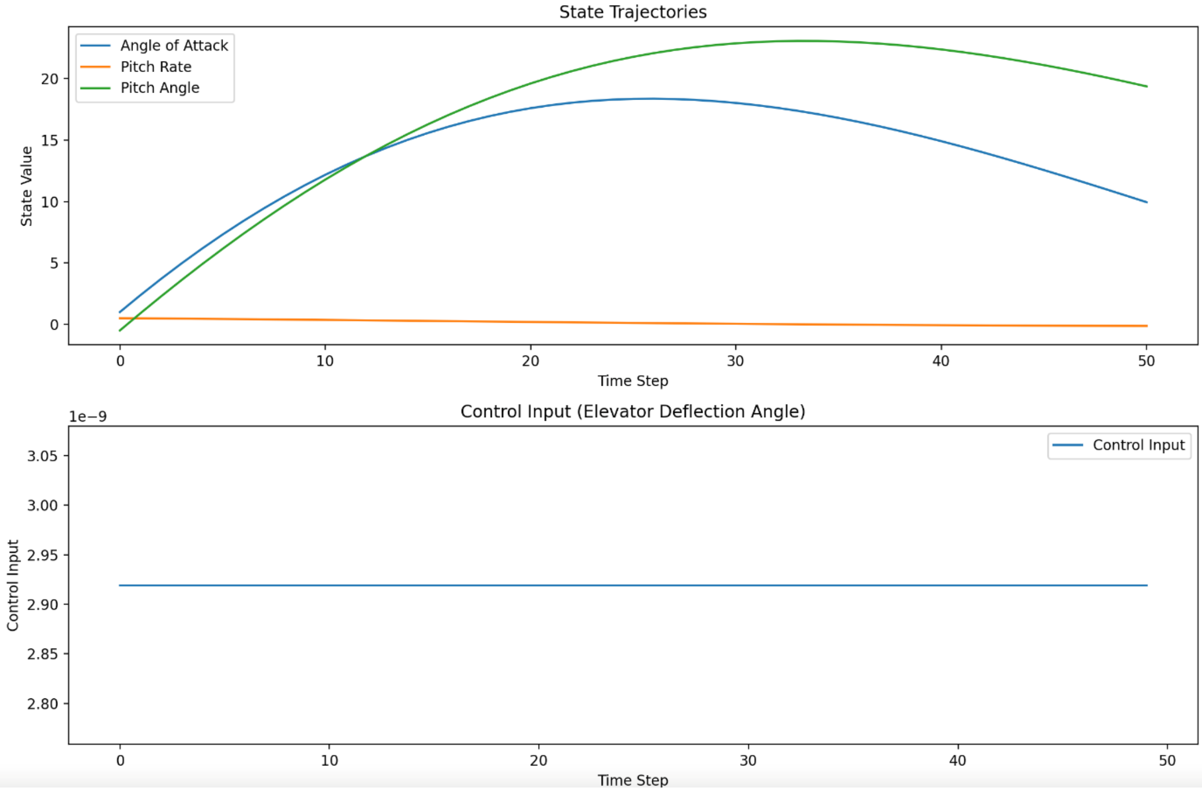
- *U* is the vector of control inputs over the prediction horizon.
- *H* and *F* are derived from *Q* and *R*, incorporating the cost of states and control inputs over the horizon.
- *G* and *h* represent the constraints on control inputs and states.

State sequences are implicitly considered through the prediction matrices *Px and Pu*, which relate the initial state and sequence of control inputs to future.

RESULTS

Simulations

Simulations demonstrate the system's response to initial conditions at the extremes of the allowable state space (*XN*). These highlight the controller's ability to bring the system towards desired states while respecting constraints. This is shown in figure X



CONCLUSION

The implementation of Model Predictive Control (MPC) presented offers a foundational approach to managing the pitch dynamics of an aircraft, balancing the need for constraint satisfaction and optimization of control efforts. However, a significant area for improvement lies in the integration of the polytope library to enhance constraint handling, particularly for the computation and utilization of the maximal invariant set. By leveraging polytope, more sophisticated constraint formulations can be introduced, allowing for the dynamic adjustment of state and control input constraints based on the evolving state of the system. This would enable a more refined control strategy that adapts in real-time to ensure robustness and stability under a wider range of operating conditions. Specifically, the computation of the maximal invariant set using polytope could provide a mathematically rigorous way to define the safe operational envelope of the system, ensuring that all state trajectories remain within acceptable bounds indefinitely. Integrating this capability would mark a substantial advancement in the control strategy, significantly enhancing the system's safety and performance by ensuring that the MPC controller always operates within a feasible and safe set, even in the face of disturbances or model inaccuracies.

Project management and individual contributions

This was a collaborative project with joint responsibility. The primary four software used to manage the project were 1) GitHub, 2) Sourcetree, 3) One Drive and 4) Whatsapp. The link to the Github repository is contained at the start of the report and contains the main script used to develop the controller, as well as test script that did not work out. Sourcetree is a GUI for git which allows developers to push/pull commits easily without using the command terminal. One Drive was used to store this report and both of us were able to work on this together. Whatsapp was used for continuous communication on any updates and was used to plan meetings in QUB graduate library which was held once per week to work together.

Both students contributed to the project, attending meetings when required and planning for the week ahead. Both students pushed up ideas to Github initially, before deciding on the method above as some models were not working at all. Both students contributed towards this report in the formatting and adding of technical detail.