CAPITAL UNIVERSITY OF SCIENCE AND TECHNOLOGY, ISLAMABAD

Fast Maneuver Control of Fighter Aircraft using Model Predictive Control

by

Muhammad Faheem Manzoor

A thesis submitted in partial fulfillment for the degree of Doctor of Philosophy

in the

Faculty of Engineering
Department of Electrical and Computer Engineering

2025

Fast Maneuver Control of Fighter Aircraft using Model Predictive Control

By

Muhammad Faheem Manzoor

(DEE183005)

Dr. Qadeer Ahmed, Assistant Professor

Ohio State University, Ohio, USA

(Foreign Evaluator 1)

Dr. Shen Yin, Professor

Norwegian University of Science and Technology (NTNU), Norway

Foreign Evaluator 2

Dr. Fazal-ur-Rehman

(Research Supervisor)

Dr. Noor Muhammad Khan

(Head, Department of Electrical and Computer Engineering)

Dr. Imtiaz Ahmad Taj

(Dean, Faculty of Engineering)

DEPARTMENT OF ELECTRICAL & COMPUTER ENGINEERING

CAPITAL UNIVERSITY OF SCIENCE AND TECHNOLOGY

ISLAMABAD

2025

I would like to dedicate this dissertation to my loving parents and family.

CAPITAL UNIVERSITY OF SCIENCE & TECHNOLOGY
ISLAMABAD

Expressway, Kahuta Road, Zone-V, Islamabad
Phone:+92-51-111-555-666 Fax: +92-51-4486705
Email: info@cust.edu.pk Website: https://www.cust.edu.pk

CERTIFICATE OF APPROVAL

This is to certify that the research work presented in the dissertation, entitled "**Fast Maneuver Control of Fighter Aircraft using Model Predictive Control**" was conducted under the supervision of **Dr. Fazal ur Rehman**. No part of this dissertation has been submitted anywhere else for any other degree. This dissertation is submitted to the **Department of Electrical & Computer Engineering, Capital University of Science and Technology** in partial fulfillment of the requirements for the degree of Doctor in Philosophy in the field of **Electrical Engineering.** The open defence of the dissertation was conducted on **May 02, 2025**.

Student Name :	Muhammad Faheem Manzoor (DEE183005)	

The Examination Committee unanimously agrees to award PhD degree in the mentioned field.

Examination Committee :

(a)	External Examiner 1:	Dr. Suheel Abdullah Associate Professor IIU, Islamabad	
(b)	External Examiner 2:	Dr. Qudrat Khan Associate Professor COMSATS University, Islamabad	
(c)	Internal Examiner :	Dr. Noor Muhammad Khan Professor CUST, Islamabad	
Supervisor Name :		Dr. Fazal ur Rehman Professor CUST, Islamabad	
Name of HoD :		Dr. Noor Muhammad Khan Professor CUST, Islamabad	
Name of Dean :		Dr. Imtiaz Ahmad Taj Professor CUST, Islamabad	

AUTHOR'S DECLARATION

I, **Muhammad Faheem Manzoor** (Registration No. **DEE183005**), hereby state that my dissertation titled, "**Fast Maneuver Control of Fighter Aircraft using Model Predictive Control**" is my own work and has not been submitted previously by me for taking any degree from Capital University of Science and Technology, Islamabad or anywhere else in the country/ world.

At any time, if my statement is found to be incorrect even after my graduation, the University has the right to withdraw my PhD Degree.

(**Muhammad Faheem Manzoor**)

Dated: 02 May, 2025

Registration No : DEE183005

PLAGIARISM UNDERTAKING

I solemnly declare that research work presented in the dissertation titled "**Fast Maneuver Control of Fighter Aircraft using Model Predictive Control**" is solely my research work with no significant contribution from any other person. Small contribution/ help wherever taken has been duly acknowledged and that complete dissertation has been written by me.

I understand the zero-tolerance policy of the HEC and Capital University of Science and Technology towards plagiarism. Therefore, I as an author of the above titled dissertation declare that no portion of my dissertation has been plagiarized and any material used as reference is properly referred/ cited.

I undertake that if I am found guilty of any formal plagiarism in the above titled dissertation even after award of PhD Degree, the University reserves the right to withdraw/ revoke my PhD degree and that HEC and the University have the right to publish my name on the HEC/ University Website on which names of students are placed who submitted plagiarized dissertation.

(Muhammad Faheem Manzoor)

Dated: 02 May, 2025

Registration No : DEE183005

List of Publications

It is certified that following publication(s) have been made out of the research work that has been carried out for this dissertation:-

- **Muhammad Faheem Manzoor**, Joel A. Paulson, Yasir Awais Butt, Fazal-ur Rehman Aamer Iqbal Bhatti. "Real-time implementation of nonlinear model predictive control for high angle of attack Maneuvers in fighter aircrafts using deep learning", Systems Science & Control Engineering, 12:1, 2342819, DOI: 10.1080/21642583.2024.2342819,2024
- **Muhammad Faheem Manzoor**, Yasir Awais Butt, Fazal-ur Rehman, and Aamer Iqbal Bhatti. "Fault-Tolerant Control of Fighter Aircraft using MPC". In 2022 19th International Bhurban Conference on Applied Sciences and Technology" (IBCAST), pp. 598-602. IEEE, 2022.

(Muhammad Faheem Manzoor)
Registration No: DEE183005

Acknowledgement

First and foremost, I want to express my gratitude to Almighty Allah, WHO gifted me with both strength and intellect. I was able to persevere through the challenging circumstances to finish my dissertation due to the patience and persistence granted to me by Allah.

I would like to thank HEC Pakistan for financial support during PhD studies. It has been an honor and a privilege to work with Dr Fazal-ur-Rehman, Dr Yasir Awais Butt, Dr. Joel Paulson, Dr. Aamer Iqbal Bhatti, and I want to express my deepest gratitude to all of them especially. I would like to express my sincere gratitude and warm regards to Dr. Mansoor Ahmed (Vice Chancellor, CUST), Dr. Imtiaz Ahmad Taj (Dean, Faculty of Engineering) and Dr. Noor Muhammad Khan (HoD Electrical Engineering) for their assistance in facilitating the administrative aspects of this research project. I also want to convey my appreciation to the Capital University of Science and Technology (CUST), Islamabad, for providing me with the opportunity to acquire my Ph.D. degree. My wish to attend the prestigious institution has turned out to be true. My gratitude is especially directed at Engr. Khalid Mahmood (Director of Graduate Studies), for his prompt assistance and guidance. Furthermore, I would like to thank the CASPR Dynamics team. My seniors: Dr. Qadeer Ahmad, Dr. Abrar Hashmi, Mr. Usman Zafar, Dr. Zohaib Latif, Dr. Sidra Bhatti, Ms. Aqsa Hussain, Mr. Farhan, Dr. Yasir Naeem, Dr. Azmat Saeed, Dr. Ahmad Mehmood Mughal, and Mr. Faheem Gulzar deserve special thanks for their insightful comments and ideas that helped me to clarify various concepts. Last but not least, I would like to extend my heartfelt thanks to my parents and siblings for their unwavering support and encouragement throughout my Ph.D. I owe special gratitude to my caring and supportive wife for her emotional support and encouragement during this challenging Ph.D. journey.

(Muhammad Faheem Manzoor)

Abstract

The aviation industry is one of the leading examples of innovation, human intellect, and modernization. The dependence on aircraft and other aerial platforms, such as drones, guided missiles, etc., has increased significantly in modern combat. One of the key research problems confronting modern super-maneuverable combat aircraft design is flight control. The maneuvering feats of modern fighter aircraft are certainly impossible without a modern flight control system, as these aircraft are highly agile and inherently unstable. Consequently, advanced control techniques are required to augment the human pilot while conducting highly agile maneuvers.

Various control techniques are available in the literature that guarantee robustness as well as optimality for fighter aircraft. Among these, MPC is the most effective and widely used technique in industrial applications, however it is computationally intensive. Many methods have evolved for reducing the computational effort, still, research is underway to make it a real-time solution for fast dynamic systems. This research is intended to explore crafting low computation-based MPC with due consideration to robustness and optimization in flight control of modern fighter aircraft.

The computational issue has been addressed in different ways in this research. One of the methods includes the use of move-blocking MPC. This research proposes an optimal LQR-based blocking algorithm. The proposed algorithm has been tested against Cessna aircraft. This method provides good results for slow aircraft, however, for fast aircraft, this scheme doesn't give much advantage. A new way has been proposed for reducing the computational complexity of the MPC. This method uses the simplified function of the aircraft obtained through the Neural Network and symbolic simplification of the MPC model as much as possible. The symbolic simplification has been performed through a tool called CasADi. This scheme has been further modified by offset-free MPC for reducing steady-state error. The proposed method solves the non-linear MPC problem with limited resources. A high angle of attack maneuver has been performed for validation of

the proposed method. A non-linear state estimator has been used for unmeasurable states under uncertainties and disturbances.

To improve the performance of the aircraft against uncertainties and disturbances and to reduce the effect of time-varying faults while performing fast maneuvers, a new control scheme named scenario-based MPC has been applied. This scheme has been further modified by the Neural Network-based model approximation, symbolic simplification of the controller, and offset-free MPC. This scheme has been applied to the F-16 non-linear model for performing the Herbst-type maneuver. While scenario-based MPC incurs higher computational complexity due to solving for uncertainties at each receding horizon step, it delivers superior performance even under fault conditions. A performance comparison with established work has been made for validation.

Contents

Author's Declaration v

Plagiarism Undertaking vi

List of Publications vii

Acknowledgement viii

Abstract ix

List of Figures xiv

List of Tables xvi

Abbreviations xvii

Symbols xviii

1 Introduction 1

1.1 Background . . . 1
1.2 Aircraft Model . . . 4
1.2.1 Axes System . . . 4
1.2.2 6 DOF Equations of Motion . . . 5
1.3 Air Combat Maneuvers . . . 8
1.3.1 Immelmann Turn . . . 9
1.3.2 Split-S . . . 10
1.3.3 Cobra . . . 11
1.3.4 Kulbit . . . 11
1.3.5 Herbst . . . 12
1.4 Control Techniques . . . 13
1.5 MPC . . . 16
1.5.1 Linear MPC . . . 19
1.5.2 Non-linear MPC . . . 23
1.5.3 MPC with State Estimation . . . 25
1.5.4 Disadvantage of MPC . . . 26

1.6 Conclusion . . . 28

2 Literature Review **29**
2.1 Introduction . . . 29
2.2 Problem Statement . . . 35
2.3 Objectives and Significance . . . 37
2.4 Methodology and Technique . . . 38
2.4.1 Open Loop Simulation Model . . . 40
2.4.1.1 Longitudinal Model . . . 40
2.4.2 MPC based Control . . . 41
2.4.3 Closed-Loop Simulation . . . 42
2.4.3.1 Performance Characteristics . . . 42
2.4.3.2 Closed Loop Model . . . 42
2.4.3.3 Closed Loop Results . . . 43
2.4.4 Discussion on Results . . . 45
2.5 Applications . . . 46
2.6 Contributions . . . 46
2.7 Conclusion . . . 47

3 Fault-Tolerant Control of Fighter Aircraft using MPC **48**
3.1 Introduction . . . 48
3.2 Aircraft Model . . . 50
3.2.1 Longitudinal Model . . . 50
3.2.2 Actuator Model . . . 51
3.3 MPC . . . 51
3.3.1 MPC Controller Model . . . 51
3.4 Fault Detection (FD) Algorithm . . . 53
3.5 Closed-Loop Simulations . . . 55
3.5.1 Closed-loop Formulation . . . 55
3.5.2 Closed-Loop Results . . . 56
3.5.3 Discussion on Results . . . 57
3.6 Conclusion . . . 58

4 LQR-based Optimal Blocking Strategy **60**
4.1 Introduction . . . 60
4.2 Blocking MPC Problem . . . 63
4.3 LQR-based Optimal Blocking MPC . . . 65
4.4 Numerical Example . . . 66
4.5 Conclusion . . . 70

5 Non-linear MPC **73**
5.1 Introduction . . . 73
5.2 Aircraft Model . . . 77
5.3 Observability . . . 79
5.4 Controllability . . . 82

5.5 Output Feedback Non-linear MPC Strategy . . . 83
5.6 Computationally Efficient NMPC using Neural Network . . . 88
5.7 Numerical Example . . . 92
5.8 Stability Analysis . . . 101
5.9 Conclusion . . . 103

6 NN-based Offset-free Scenario-based MPC 105
6.1 Introduction . . . 105
6.2 Fighter Aircraft Specifications . . . 108
6.3 NN-based Offset-Free Scenario-based MPC . . . 111
6.4 Computationally Efficient Implementation . . . 116
6.5 Time Varying Faults Tolerance . . . 118
6.6 Numerical Example . . . 120
6.7 Complexity Analysis . . . 126
6.8 Conclusion . . . 128

7 Conclusion and Future Work 130
7.1 Conclusion . . . 130
7.2 Future Work . . . 133

Bibliography 134

Flying & Handling Qualities 147

List of Figures

1.1 Mechanical Flight Control System . . . 2
1.2 Stability Augmentation System in the Feedback Path . . . 3
1.3 Fly-by-Wire Control System . . . 3
1.4 Aircraft Axes . . . 5
1.5 Aircraft Control Surface . . . 9
1.6 Immelmann Turn . . . 10
1.7 Split-S . . . 11
1.8 Cobra Maneuver . . . 12
1.9 Kulbit . . . 12
1.10 Herbst . . . 13
1.11 Flight Control Problem . . . 14
1.12 General MPC Block Diagram . . . 17
1.13 Simulation Results for Tracking . . . 18
1.14 Linear MPC Closed-loop Model . . . 23
1.15 General Closed-loop Block Diagram with Estimator . . . 26

2.1 General Move Blocking Scheme . . . 33
2.2 Closed Loop Simulink Model for MPC . . . 43
2.3 Pitch Angle tracking using Linear MPC . . . 44
2.4 Elevator Control Deflection for Linear MPC . . . 44
2.5 Thrust Control input for Linear MPC . . . 45

3.1 Categories of Faults . . . 49
3.2 Block Diagram with Fault Detection Module . . . 56
3.3 Closed-Loop α Tracking without FD . . . 56
3.4 Elevator Deflection without FD . . . 57
3.5 Closed-Loop α Tracking with FD Module . . . 57
3.6 Elevator Deflection with FD . . . 58

4.1 Fixed Blocking MPC Reference Tracking . . . 69
4.2 Moving Window Blocking MPC Reference Tracking . . . 69
4.3 LQR based Blocking MPC Reference Tracking . . . 70
4.4 Fixed Blocking MPC Input Response . . . 70
4.5 Moving Window Blocking MPC Input . . . 71
4.6 LQR-based Blocking MPC Input . . . 71

5.1 Observability Test Result in Matlab . . . 82

5.2 Kalman Filter State Estimation Process 87
5.3 NN Based NMPC Flow 93
5.4 NN-based Closed Loop MPC Implementation 94
5.5 AOA Tracking using NN-based NMPC 95
5.6 Speed Response using NN-based MPC 95
5.7 Side-Slip Angle Response using NN-based MPC 96
5.8 Roll Angle Response using NN-based MPC 96
5.9 Pitch Angle Response using NN-based MPC 97
5.10 Yaw Angle Response using NN-based MPC 97
5.11 Roll Rate Response using NN-based MPC 98
5.12 Pitch Rate Response using NN-based MPC 98
5.13 Yaw Rate Response using NN-based MPC 99
5.14 Altitude Response using NN-based MPC 99
5.15 Longitudinal Input Controls using NN-based MPC 100
5.16 Lateral Input Controls using NN-based MPC 100
5.17 Lyapunov Stability Analysis 102

6.1 Herbst Maneuver 110
6.2 Scenario Tree 113
6.3 UKF Cycle 117
6.4 NN-based OSMPC Structure 119
6.5 Stuck Fault 120
6.6 Stall Load 121
6.7 Time-Varying Stall Load 121
6.8 NN-based OSMPC Closed Loop Block Diagram 122
6.9 Angle of Attack tracking using OSMPC 123
6.10 Roll Rate Response using OSMPC 124
6.11 Side Slip Angle Response using OSMPC 124
6.12 Speed Response using OSMPC 125
6.13 Pitch Rate Response using OSMPC 125
6.14 Yaw Rate Response using OSMPC 126
6.15 Thrust Input using OSMPC 127
6.16 Elevator Angle using OSMPC 127
6.17 Aileron Angle using OSMPC 128
6.18 Rudder Angle using OSMPC 128
6.19 Computation Time 129

List of Tables

1.1 Control Surface Deflection Convention 9

2.1 Literature Survey 36
2.2 Research Methodology 38
2.3 Performance Characteristics of Longitudinal Model 43

3.1 Fault-Detection Algorithm 54

4.1 Blocking Inputs 61
4.2 Aircraft Limits 67

5.1 F-16 Parameters 78
5.2 F-16 Constraints 78

6.1 Aircraft Parameters 109
6.2 Aircraft Constraints 109

7.1 Performance Comparison of different MPC schemes 132

Abbreviations

AOA	Angle of Attack
DOF	Degree of freedom
EKF	Extended Kalman Filter
FNN	Feedforward Neural Network
FT	Fault Tolerant
LQG	Linear Quadratic Guassian
LQR	Linear Quadratic Regulator
MPC	Model Predictive Control
NN	Neural Network
NMPC	Non-linear Model Predictive Control
SMPC	Scenario-based Model Predictive Control
SMC	Sliding Mode Control
UKF	Unscented Kalman Filter

Symbols

C_e	Cost function for fault
$E(x)$	Expected Value of x
$f(x)$	A non-linear function of x
h	Height
P_n	North Position
P_e	East Position
p	Roll Rate
q	Pitch Rate
r	Yaw Rate
t_r	Rise time
t_s	Settling Time
u	Control input vector
U	Longitudinal Velocity
V	Lateral Velocity
W	Normal Velocity
x	states
ϕ	Roll Angle
θ	Pitch Angle
ψ	Yaw Angle
α	Angle of Attack
β	Side-slip Angle
δ_e	Elevator Angle
δ_a	Aileron Angle

δ_r	Rudder Angle
th	Throttle
$\%OS$	Percentage Overshoot
J	Objective function for linear system
l	Objective function for non-linear system
R_{ref}	Reference input
y	Output Vector
k	Time step
e	Error
w	Disturbance
N	Prediction Horizon
n_x	Number of states
n_i	Number of inputs
q	Cost matrix for states
r	Cost matrix for inputs
dd	Detection Delay
$\hat{u}$	blocked input
T	Blocking Matrix

Chapter 1

Introduction

1.1 Background

The idea of fighter jets immediately followed the first successful flight by the Wright brothers. Aircraft were first introduced in the military before World War I for transportation and reconnaissance. In 1914 there was an exchange of fire between Austro-Hungarian and Serbian pilots during a reconnaissance flight that sparked the idea of aerial combat and fighter aircraft. Since inception, fighter aircraft have evolved through several generations with fifth-generation stealth aircraft such as J-20, SU-57, F-35, and F-22 already deployed and the sixth-generation aircraft moving past the conceptual design stage. Modern fighter aircraft in their features and characteristics certainly appear like Sci-fi machines conceptualized in the movies in the latter half of the previous century.

Whereas other technologies related to aircraft e.g., materials, propulsion, weapons, airfoils progressed at an accelerated pace, the use of mechanical controls introduced by Lilienthal and the Wright Brothers remained prevalent until even the fourth-generation fighter aircraft. The control system was entirely manual, comprised of a system of pulleys, rods, cables, levers, etc., and the levels of the stick and rudder-pedal forces to steer and maneuver an airplane were constrained by the physical capabilities of the pilot Figure 1.1. The rudimentary control despite its inherent inefficiencies and delay was acceptable because the earlier generation aircraft were

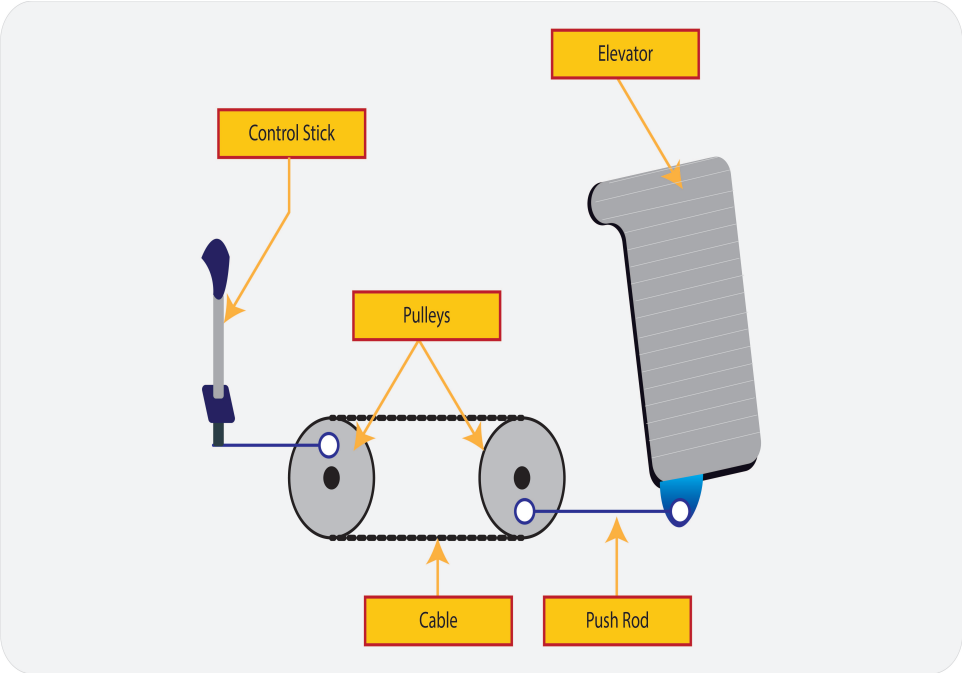

FIGURE 1.1: Mechanical Flight Control System

naturally aerodynamically stable. In terms of control theory, the aircraft had their poles placed well into the complex plane with real parts to be negative for ensuring sufficient damping.

Due to the pursuit for higher speed, the race to break the sound barrier, and the requirements of maneuverability there was a lot of pressure on designers to develop more responsive and robust aircraft with allied flight control methods. Regarding control theory, the aircraft were made responsive by bringing the poles closer to the imaginary axis in the left half of a complex plane and with lower damping rations to allow more responsiveness. Such aircraft could not be controlled with the primary control systems of that time. Consequently, newer systems were introduced in which a flight control system was augmented in the airframe characteristics Figure 1.2. In these systems, the pilot was still generating control commands through rudder sticks, but a mechanical the controller in the feedback path aided in stabilizing the aircraft-based on the measurements of the variables of interest.

Craving for maneuverability pushed the stability augmentation systems to the limits with the development of fourth-generation aircraft such as the F-16 in the 1980s. To enhance maneuverability, the aircraft were made inherently unstable

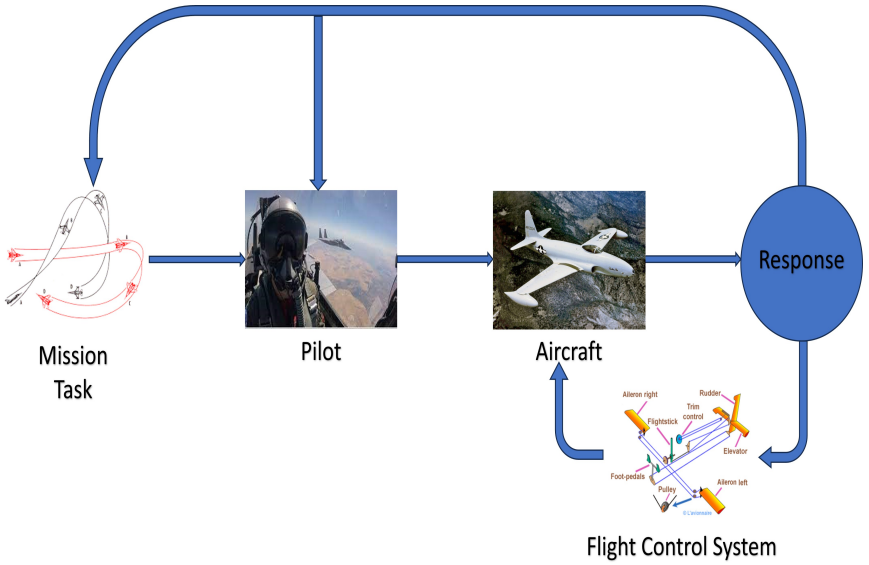

FIGURE 1.2: Stability Augmentation System in the Feedback Path

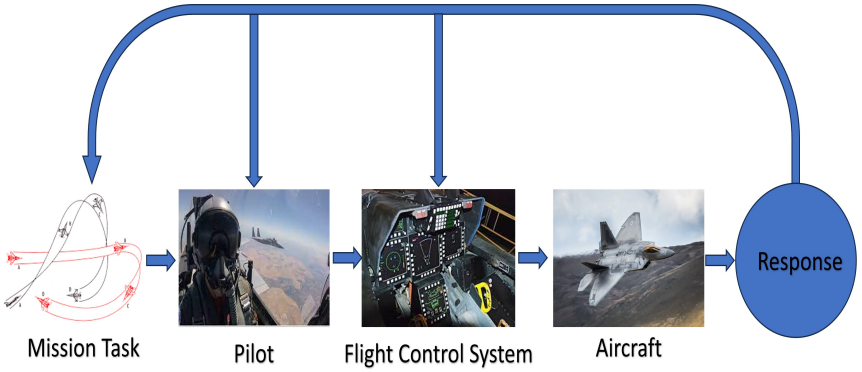

FIGURE 1.3: Fly-by-Wire Control System

with their poles placed into the right half of the complex plane. The use of conventional mechanical and hydro-mechanical flight control systems was depreciated in these advanced aircraft and fly-by-wire (FBW) primary flight controls integrated with the stability augmentation system were incorporated in Figure 1.3. In this setting, the flight control system becomes an integral part of the primary signal flow path and becomes responsible for interpreting and implementing the pilot commands. These control systems were based on digital computers with electrical hydro-mechanical actuators.

In the last decade of the $20th$ century, fifth-generation aircraft were conceptualized with capabilities such as supercruise, stealth, net centricity, etc. The aircraft configuration and corresponding control system were redesigned to cater to super

maneuverability. Fifth-generation aircraft employ purer electrical actuation by the use fibre controlled optical systems. There are numerous advantages of using fiber. The main advantage of using fiber is that the weight due to heavy copper has reduced, thereby reducing the weight of the aircraft. It is evident that aircraft responsiveness and maneuverability are directly dependent upon the sophisticated mathematics of aerodynamics and control systems and hence the need to pledge serious effort to harness this technology cannot be over-emphasized.

1.2 Aircraft Model

1.2.1 Axes System

The aircraft axes system is usually defined by the body of the aircraft. The direction of the fuselage tip shows the positive x-axis in the body axis system. The direction of the right wing (perpendicular to the x-axis) shows the y-axis. The downward direction usually represents a positive z-axis. The resulting system of axes is called the body fixed axis system or simply the body axis system. Since aerodynamic coefficients and moments depend on the airflow and the orientation of the aircraft concerning the air, it is common practice to define the axis system concerning the airflow for control purposes. The two orientation angles used are called the angle of attack (α) and the side slip angle(β). The angle of attack is usually obtained by rotating the tip of the aircraft around the y-axis in the direction of the velocity. The resulting system of axes is called the stability axis system. In effect, the resulting axis system is the same as the body-fixed axis system except that the x-axis points in the direction of velocity. The other system is the wind axis system. The wind axis system is the same as the stability axis system except that the y-axis of the system is rotated by an angle called the side slip angle. The side-slip angle is obtained by rotating the tip of the aircraft about the z-axis in the relative wind direction, resulting in a wind-axis system. These three axes' systems are defined for the body of the aircraft, hence are usually called body axes systems. However, for differentiation, the three axes systems are called

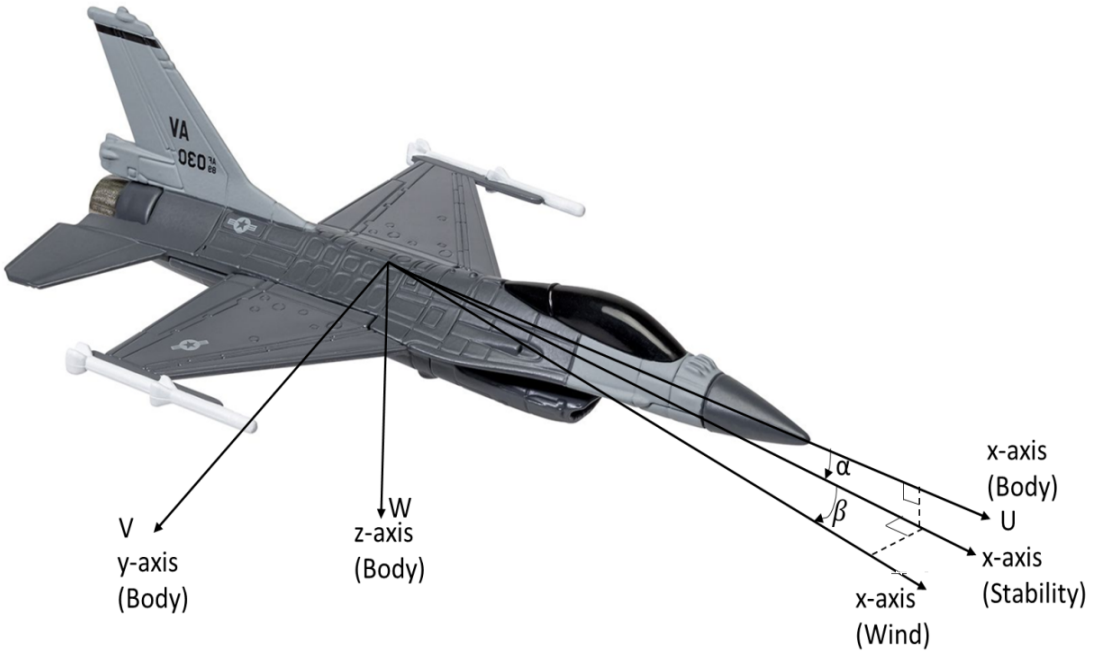

FIGURE 1.4: Aircraft Axes

body, stability, and wind axis systems. Three different axes system can be easily understood by the Figure 1.4.

1.2.2 6 DOF Equations of Motion

The aircraft has 6 DOF motion that include three linear motions in the x, y, and z directions and three angular motions along the x,y, and z axes. The standard aircraft non-linear model that is used for non-linear simulations and control purposes has four vectors. The first one is the velocity vector $\boldsymbol{v}$ which includes longitudinal, lateral, and normal velocities which are represented by the symbols $\boldsymbol{U}$ (longitudinal velocity), $\boldsymbol{V}$ (lateral velocity), and $\boldsymbol{W}$ (normal velocity). The second vector is the Euler angle vector Φ. The components of Φ are roll angle ϕ, pitch angle θ, and yaw angle ψ. The third one is the vector of angular rates ω which is comprised of roll rate p, pitch rate q, and yaw rate r. The last vector is the position vector P which has north position P_n, east position P_e, and height h. These form four sets of differential equations which include three force equations, three kinematic equations, three moment equations, and three navigation equations. The state vector is then defined as

$$\boldsymbol{x}^T = [\boldsymbol{U}\ \ \boldsymbol{V}\ \ \boldsymbol{W}\ \ \phi\ \ \theta\ \ \psi\ \ p\ \ q\ \ r\ \ P_n\ \ P_e\ \ h]$$

The non-linear equations of motion for all the states are given as

$$\dot{\boldsymbol{U}} = r\boldsymbol{V} - q\boldsymbol{W} - g\sin\theta + \frac{\boldsymbol{F_x}}{m} \tag{1.1}$$

$$\dot{\boldsymbol{V}} = -r\boldsymbol{U} + p\boldsymbol{W} + g\sin\phi\cos\theta + \frac{\boldsymbol{F_y}}{m} \tag{1.2}$$

$$\dot{\boldsymbol{W}} = q\boldsymbol{U} - p\boldsymbol{V} + g\cos\phi\cos\theta + \frac{\boldsymbol{F_z}}{m} \tag{1.3}$$

$$\dot{\phi} = p + \tan\theta(q\sin\phi + r\cos\phi) \tag{1.4}$$

$$\dot{\theta} = q\cos\phi + r\sin\phi \tag{1.5}$$

$$\dot{\psi} = \frac{q\sin\phi + r\cos\phi}{\cos\theta} \tag{1.6}$$

$$\dot{p} = (k_1 r + k_2 p)q + k_3\boldsymbol{L} + k_4\boldsymbol{N} \tag{1.7}$$

$$\dot{q} = k_5 pr - k_6(p^2 - r^2) + k_7\boldsymbol{M} \tag{1.8}$$

$$\dot{r} = (k_8 p - k_2 r)q + k_4\boldsymbol{L} + k_9\boldsymbol{N} \tag{1.9}$$

$$\begin{aligned}\dot{p_n} = \ & \boldsymbol{U}\cos\theta\cos\psi + \boldsymbol{V}(-\cos\psi\sin\psi + \sin\phi\sin\theta\cos\psi) \\ & +\boldsymbol{W}(\sin\phi\sin\psi + \cos\phi\sin\theta\cos\psi)\end{aligned} \tag{1.10}$$

$$\begin{aligned}\dot{p_e} = \ & \boldsymbol{U}\cos\theta\sin\psi + \boldsymbol{V}(\cos\phi\cos\psi + \sin\phi\sin\theta\sin\psi) + \\ & \boldsymbol{W}(-\sin\phi\cos\psi + \cos\phi\sin\theta\sin\psi)\end{aligned} \tag{1.11}$$

$$\dot{h} = \boldsymbol{U}\sin\theta - \boldsymbol{V}\sin\phi\cos\theta - \boldsymbol{W}\cos\phi\cos\theta \tag{1.12}$$

The constants are given as:

$$k_1 = \frac{(M_{yy} - M_{zz})M_{zz} - M_{xz}^2}{M_{xx}M_{zz} - M_{xz}^2} \tag{1.13}$$

$$k_2 = \frac{(M_{xx} - M_{yy} + M_{zz})M_{xz}}{M_{xx}M_{zz} - M_{xz}^2} \tag{1.14}$$

$$k_3 = \frac{M_{zz}}{M_{xx}M_{zz} - M_{xz}^2} \tag{1.15}$$

$$k_4 = \frac{M_{xz}}{M_{xx}M_{zz} - M_{xz}^2} \tag{1.16}$$

$$k_5 = \frac{M_{zz} - M_{xx}}{M_{yy}} \tag{1.17}$$

$$k_6 = \frac{M_{xz}}{M_{yy}} \tag{1.18}$$

$$k_7 = \frac{1}{M_{yy}} \tag{1.19}$$

$$k_8 = \frac{M_{xx}(M_{xx} - M_{yy}) + M_{xz}^2}{M_{xx}M_{zz} - M_{xz}^2} \tag{1.20}$$

$$k_9 = \frac{M_{xx}}{M_{xx}M_{zz} - M_{xz}^2} \tag{1.21}$$

Where M is used to denote the moment of inertia. For control purposes, a stability axis is used which changes the velocity vector from $[u \;\; v \;\; w]$ to $[V_T \;\; \alpha \;\; \beta]$ i.e., total velocity, angle of attack (AOA), and side-slip angle respectively. The equation of transformation is given as:

$$\tan\alpha = \frac{\boldsymbol{W}}{\boldsymbol{U}} \tag{1.22}$$

$$\sin\beta = \frac{\boldsymbol{V}}{V_T} \tag{1.23}$$

$$V_T = \sqrt{\boldsymbol{U}^2 + \boldsymbol{V}^2 + \boldsymbol{W}^2} \tag{1.24}$$

These new states are obtained by the transformation:

$$\dot{V_T} = \frac{\boldsymbol{U\dot{U}} + \boldsymbol{V\dot{V}} + \boldsymbol{W\dot{W}}}{V_T} \tag{1.25}$$

$$\dot{\alpha} = \frac{\boldsymbol{U\dot{W}} - \boldsymbol{W\dot{U}}}{\boldsymbol{U}^2 + \boldsymbol{W}^2} \tag{1.26}$$

$$\dot{\beta} = \frac{\boldsymbol{\dot{V}}V_T - \boldsymbol{V}\dot{V_T}}{V_T^2 \cos\beta} \tag{1.27}$$

There are mainly four main inputs of the aircraft, two are for longitudinal motion and two for lateral motion. The longitudinal motion is controlled by thrust and elevator. Thrust is used to control the speed while elevator is used for the pitching motion. The lateral control surfaces are used to control the roll motion by aileron and yaw motion by rudder. The four control surfaces are defined as the control vector u

$$\boldsymbol{u^T} = [th \;\; \delta_e \;\; \delta_a \;\; \delta_r]$$

The control surfaces in the aircraft can be seen in Figure 1.5 with highlighted areas [1]. The direction of arrows around the control surfaces indicates the positive motion of the control surface or the negative motion of the relative state. The sign convention for the F-16 control surfaces is summarized in Table 1.1. The sign convention for other aircraft can be different. The sign convention helps in controller implementation and performance comparison.

1.3 Air Combat Maneuvers

Over the years, many complex maneuvers have been introduced to gain air superiority. These complex maneuvers include the Immelmann turn, Split-S, Cobra

FIGURE 1.5: Aircraft Control Surface

TABLE 1.1: Control Surface Deflection Convention

Surface	**Deflection Side**	**Deflection Sense**	**Effect on motion**
Elevator	Down (Trailing Edge)	Positive	Negative Pitch
Aileron	Down (Right wing Trailing Edge)	Positive	Negative Roll
Rudder	Left (Trailing Edge)	Positive	Negative Yaw

Maneuver, Kulbit, Herbst Maneuver, etc. The details of these maneuvers are given as:

1.3.1 Immelmann Turn

Immelmann turn was first performed in WWI by a German named Lieutenant Max Immelmann, hence named after him. In this maneuver, the aircraft performs both longitudinal and lateral motion. It involves the control of pitch, yaw, and speed. In this maneuver, the pilot first achieves enough speed to perform the turn in a close loop. It then pitches up the aircraft while maintaining the roll

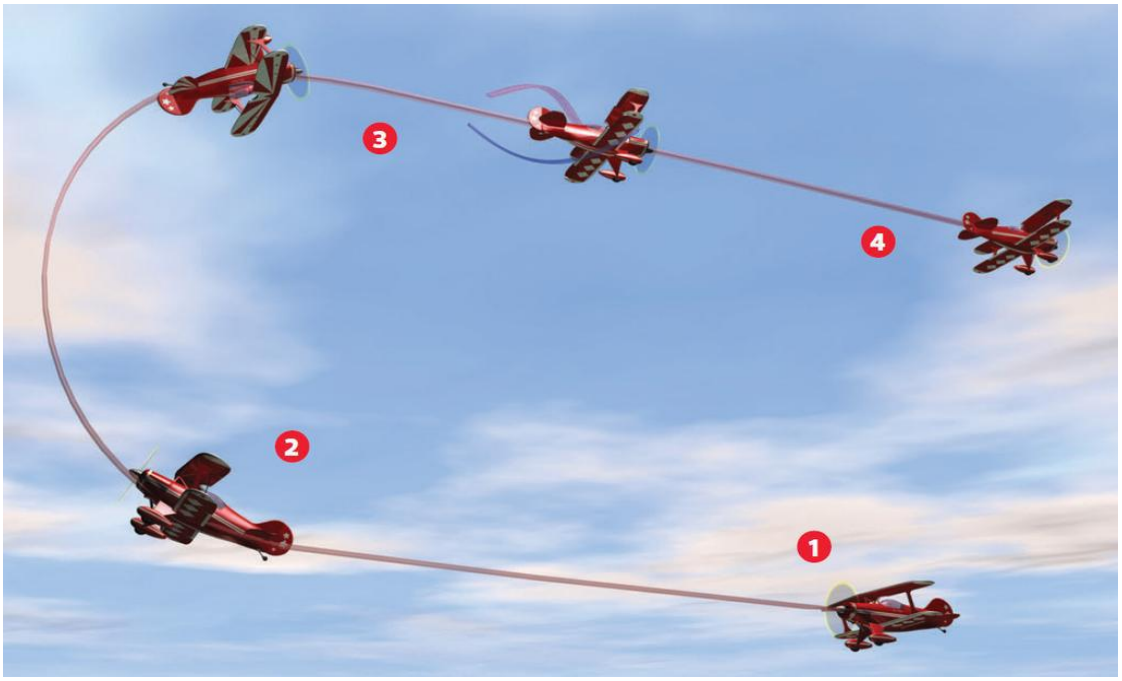

FIGURE 1.6: Immelmann Turn

and yaw motion (longitudinal motion only). After attaining enough altitude, it then performs the roll motion in a half circle. Thus, reversing the direction of velocity at 180 degrees. In effect, the aircraft goes at a high altitude pointing in the opposite direction, thus called roll-of-the-top. This maneuver is performed to gain the attacking position after passing by an aircraft. This maneuver is given [2] can be seen in Figure 1.6.

1.3.2 Split-S

Split-S is more likely the Immelmann maneuver. It also involves longitudinal and lateral motion. However, in this maneuver, the pilot starts by rolling in a half circle at 180 degrees. Then the pilot pitches the aircraft resulting in a downward motion. In effect, the aircraft reduces altitude with an increased velocity but in the opposite direction. The main difference between an Immelmann turn and a Split-S is that in the Immelmann turn the aircraft increases altitude at the cost of speed reduction while in Split-S the aircraft gains speed at the cost of altitude reduction. This maneuver is mainly performed in a dog fight for withdrawing from combat. However, the aircraft must be at enough height to perform the maneuver. The

FIGURE 1.7: Split-S

motion of the aircraft given in [3] in the Split-S maneuver can be seen in Figure 1.7.

1.3.3 Cobra

The Cobra maneuver or simply Cobra is also a complex maneuver that involves longitudinal motion only. In this maneuver, the aircraft instantly reduces speed to zero while changing the angle of attack by about 90 degrees. However, the aircraft doesn't attain an effective altitude while maintaining enough thrust. Then the aircraft reduces pitch before going to level flight. This maneuver is performed in close combat in which a chases overshoot. This maneuver needs an aerodynamically unstable aircraft but must have the ability to return to level flight before becoming unstable. Thrust vectoring or canard is used for this maneuver. This maneuver can be seen in Figure 1.8 and is given in [4].

1.3.4 Kulbit

This maneuver is a little like the Cobra maneuver. As compared to the cobra, in this maneuver the goes on pitching up. In this maneuver, the aircraft completes

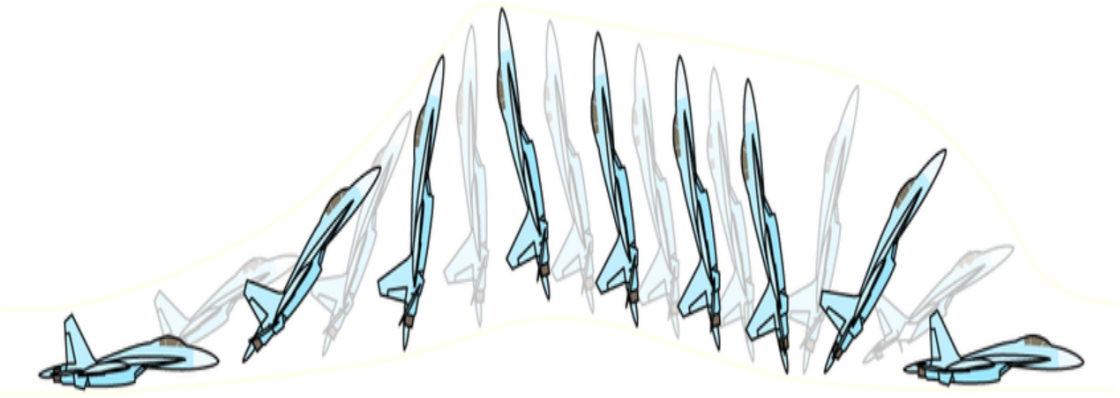

FIGURE 1.8: Cobra Maneuver

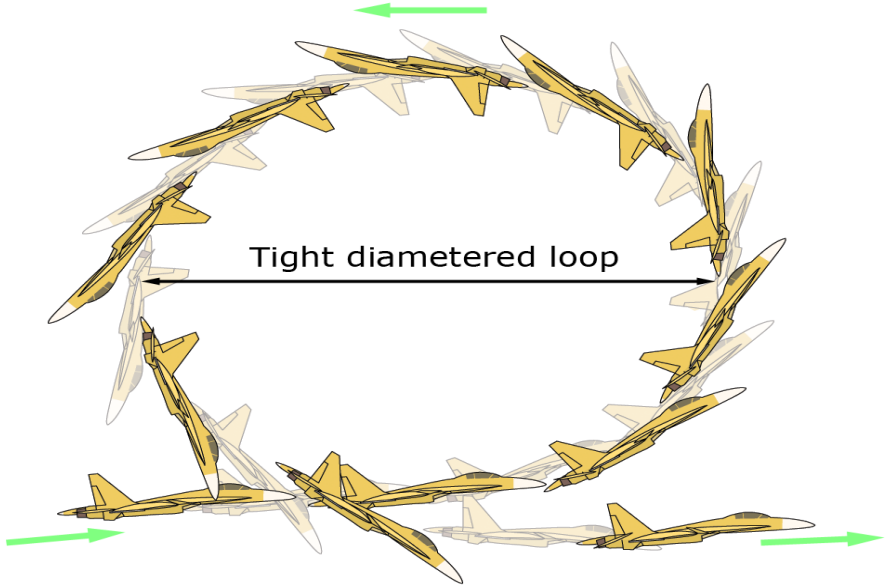

FIGURE 1.9: Kulbit

the loop of pitching motion. The pitching motion goes from 0 to 360 degrees but in a tight loop. However, the cobra cut off instantly. This maneuver is also controlled by the longitudinal motion of the aircraft. The Kulbit maneuver can be understood by the Figure 1.9 which is obtained from [5].

1.3.5 Herbst

This maneuver involves both longitudinal and lateral motion. It involves the control of pitch and roll motion. The maneuver starts at high speed, then it

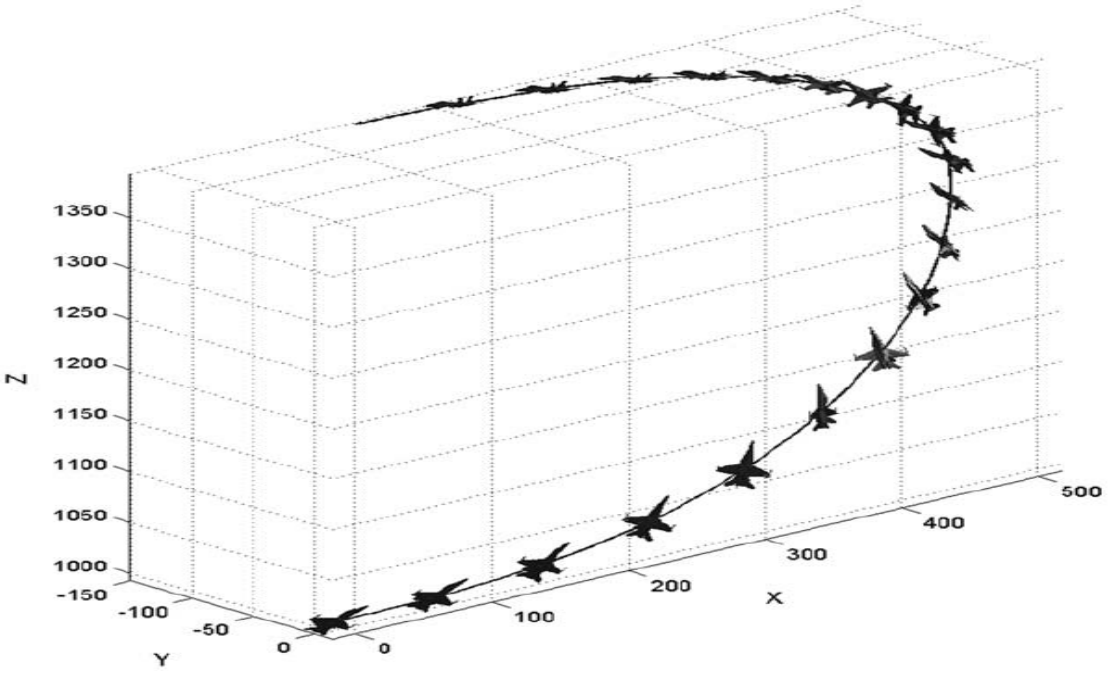

FIGURE 1.10: Herbst

increases the angle of attack to post-stall level. After this, the aircraft performs a rolling motion and reduces the angle of attack, thus pointing downward and 180 degrees to the direction of initial velocity. By pointing downward it regains its original speed. This maneuver is quite popular in air shows and combat. This maneuver can be seen in Figure 1.10, which isthe same as in [6].

All of these maneuvers have their significance and are hard to control. Some of these maneuvers involve the longitudinal motion of the aircraft e.g., Cobra maneuver. Some of them involve the control of the longitudinal as well as lateral motion e.g., Herbst Maneuver. This research is focused on the control of complex maneuvers. The maneuver chosen here is the Herbst maneuver because it involves the motion of longitudinal and lateral motion thus increasing the complexity of the maneuver.

1.4 Control Techniques

Flight Control Problem (FCP) has two perspectives from a control point of view. One is stability and the other is reference tracking. From a stability point of view, it is further classified as Short Period Mode and Phugoid Mode in the Longitudinal

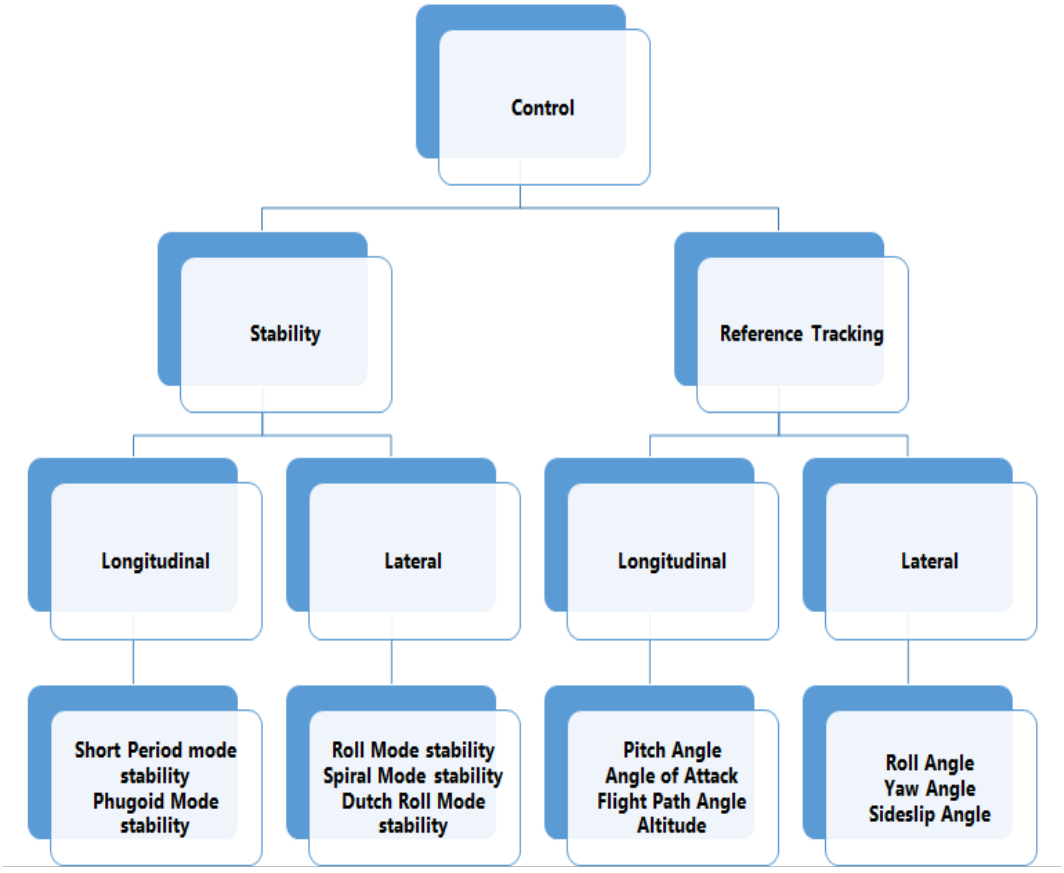

FIGURE 1.11: Flight Control Problem

model, while Roll Mode, Spiral Mode, and Dutch Roll Mode in the lateral model. The reference tracking includes the tracking of different variables of interest. These include pitch, angle of attack, flight path angle in longitudinal model and roll, yaw, and side-slip angles in lateral mode. The flight control problem is summarized in Figure 1.11.

Various control techniques have been applied in the flight control system of fighter aircraft to achieve level 1 flying and handling qualities, e.g., PID control, Eigenstructure assignment, Linear Quadratic Regulator (LQR), Linear Quadratic Gaussian (LQG), H_∞ control, Sliding Mode Control (SMC) and Model Predictive Control (MPC). All of these techniques have their own pros and cons.

PID control is the most famous and most widely used control technique because of easy implementation, adequate performance, and low computations [7]. It can easily stabilize the system giving zero steady-state error. PID controller has been applied on different aircraft in [8–11] and several other references for flight control.

In [11], it has been applied to the high-fidelity model of the F-16 fighting falcon for pitch attitude control. PID controller gives desired handling qualities and robustness to uncertainties, noises, and turbulences. However, it amplifies the high-frequency noise, gives a low stability range, and is used for SISO systems.

Linear Quadratic Regulator (LQR) is the feedback control scheme that finds the minimum of the quadratic cost function. LQR is one of the famous control schemes in modern control theory used for MIMO systems because of optimality. Also, a designer can manage between control effort and error. LQR has been applied on aircraft in [12–15]. In [12, 13], LQR has been applied for longitudinal motion control of F-16. It has been shown through simulations that LQR provides good performance and robustness against external disturbances. In [14], LQR has been used for pitch control of general aviation airplanes and a comparison has been made with PID controller regarding time domain specifications. Genetic Algorithm (GA) is used for finding optimal parameters and for reducing the time of tuning. LQR proved to be better than PID concerning settling time t_s, percentage overshoot $\%OS$, and steady-state error. But in terms of rise time t_r, PID is better. So the overall performance specifications of LQR is better. However, this technique is unconstrained and provides infinite horizon optimal control. Besides, model uncertainties and non-linearity may cause robustness issues.

LQG is a combination of LQR for state feedback controller and Kalman filter for optimal observer. It has all the pros and cons of LQR; besides it is an optimal controller for non-linear systems as well. In [16–18], LQG has been applied over aircraft. LQR provides optimality in performance and the Kalman filter provides optimal estimation in the presence of process noise and disturbance. Hence, suitable for applications where noise and disturbance reduce performance. In [16], a comparison has been made between LQR and LQG based on step response. Results showed that LQR gives better performance in the absence of noise and disturbance. However, when we include process noise and disturbance in the model, LQG gives better results because of the optimal estimation of the Kalman filter in the presence of noise.

H_∞ is a robust control scheme providing robust stabilization of the plant with

unstructured uncertainty. H_∞ has been applied in [19–21] and in [22] a comparison has been made with LQG for longitudinal (pitch angle) control of combat aircraft, proving that H_∞ is a better controller than LQG in terms of performance specifications. The performance of H_∞ controller requires the plant model to be good. Mathematical understanding of a high level is also required. The success rate depends upon the choice of weight functions.

Sliding mode control (SMC) is a non-linear robust controller in which model parameter variations and disturbances don't affect the performance [23]. It is used to enforce the non-linear system towards normal behavior by altering the dynamics. Sliding Mode Control has been applied in [24–27] over aircraft for improving performance. In [23], SMC and integral SMC have been applied over F-18 fighter aircraft for tracking pitch angle and pitch rate at a high angle of attack. Controller performance has been evaluated based on transient response. Simulation results showed that SMC gives quite good tracking and performance.

1.5 MPC

MPC (Model Predictive Control), as is evident from its name, makes predictions about future states (finite horizon) based on current information. It finds the optimal control trajectory by minimizing some cost function. This optimization under constraints needs to be done within a certain time limit. It applies only to the first input and discards the others. After that prediction horizon shifts, one step ahead, and the same procedure is repeated after each sampling interval. Since MPC computes optimal control input after every sampling interval, this technique adopts online computation. MPC is the most popular control scheme in the industry after PID [7] because of many advantages e.g., constrained optimization, finite-horizon control, and transient & steady-state response improvement. There is no comparative study of MPC with other controllers on aircraft. However, a comparison of MPC with LQR, H_∞, and m-synthesis has been made in [28] over a compliant link mechanism for controlling position and vibration suppression. Simulation results showed that MPC gives better results as compared to other control techniques for the given performance specifications.

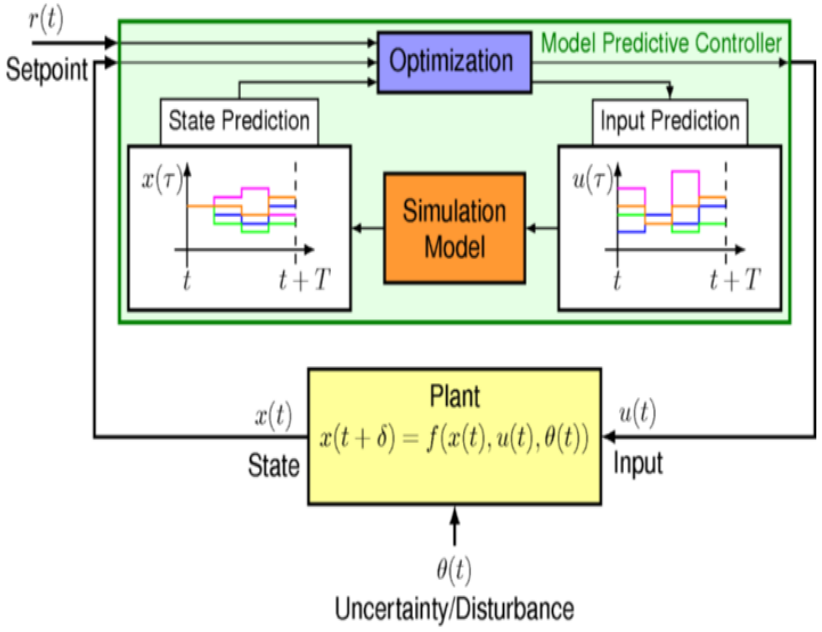

FIGURE 1.12: General MPC Block Diagram

MPC is used here in this research because it is a well-known technique in many industrial applications e.g., process industry, chemical plants, etc. The major advantages of MPC are

- Handle Constraints
- Handle MIMO systems
- Preview Capability
- Provides an optimal solution
- The designer has freedom of management between control effort and error by tuning the weighting factors

The general model for MPC and a tracking result are shown in Figure 1.12 and Figure 1.13 respectively.
Certain terms are commonly used for any MPC formulation. The terms are defined as:

- **Moving Horizon Window**: This defines the number of future time steps used to predict the model of the system. It starts with the current time step

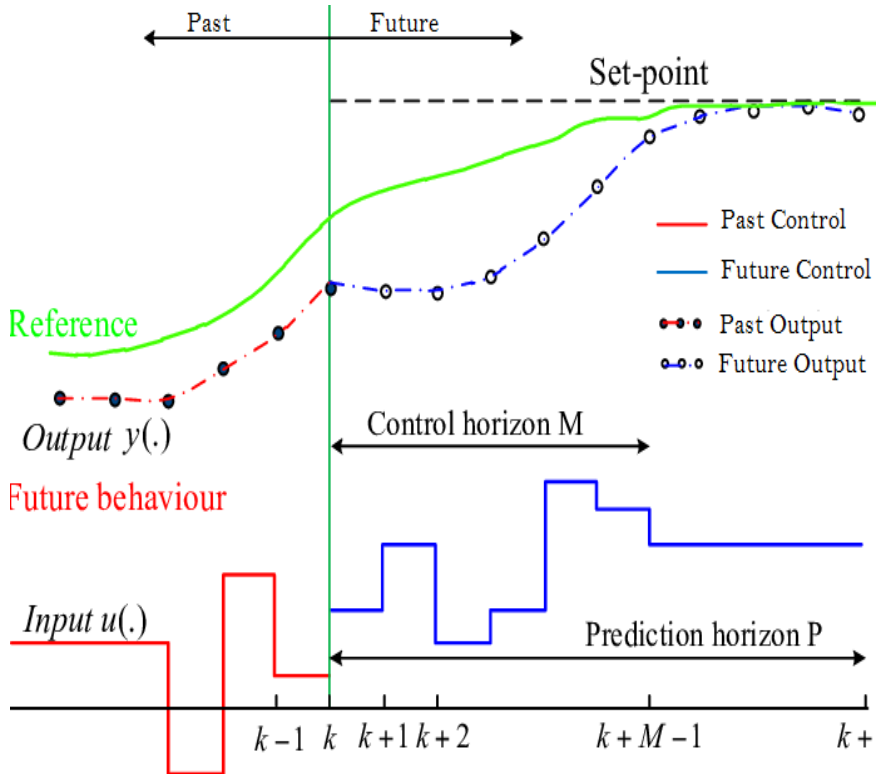

FIGURE 1.13: Simulation Results for Tracking

to the time step defined by the prediction horizon. Since the current time always changes this window is always moving.

- **Prediction Horizon**: This defines the number of future time steps. This is used to calculate the response of the system in the future based on the current states and inputs of the system. Usually, the prediction horizon remains fixed, however, sometimes the designer prefers to use a variable prediction horizon. The prediction horizon is defined by the variable N in this research. The length of the Moving Horizon Window and Prediction Horizon are the same.

- **Current State**: Current state shows the value of the state in current time. This is usually defined by the variable $X(k)$ or x_k, where k is used to denote the current time step.

- **Prediction Model**: The prediction model is the extension of the system model for the prediction horizon. The MPC controller uses a prediction model instead of a simple model to find the optimized control sequence.

- **Objective Function**: The MPC problem finds the optimal sequence by minimizing a function of states or inputs or a combination of both. This function is called the objective function. Usually, the objective is to minimize the squared error or squared input in this research.

Various versions of MPC controllers are available these days. The main difference between all these variations is the system model used. In the early formation of MPC, the Finite impulse response (FIR) and step response model were used. These types of models were limited to stable systems only. This problem was then solved by the use of the transfer function model which can be used for both stable and unstable plants. However, the transfer function model was limited to SISO systems and ineffective for MIMO systems. This problem was then solved by using the state-space model in MPC, which is named the linear MPC model these days. Linear MPC can be used for both stable and unstable plants and it can be used for MIMO systems. With increasing computational capacity, the use of non-linear models in MPC is getting popular which is called non-linear MPC (NMPC). The details of linear and non-linear MPC are given as:

1.5.1 Linear MPC

Given a discrete-time linear plant with a state-space model given as:

$$\boldsymbol{x_p}(k+1) = A_p\boldsymbol{x_p}(k) + B_p\boldsymbol{u_p}(k) \tag{1.28}$$

$$\boldsymbol{y}(k) = C_p\boldsymbol{x_p}(k) \tag{1.29}$$

The state space model can be augmented with an integrator to eliminate steady-state error. For that, the system will be like

$$\Delta\boldsymbol{x_p}(k+1) = A_p\Delta\boldsymbol{x_p}(k) + B_p\Delta\boldsymbol{u_p}(k) \tag{1.30}$$

$$\boldsymbol{y_p}(k+1) = \boldsymbol{y_p}(k) + C_pA_p\Delta\boldsymbol{x_p}(k) + C_pB_p\Delta\boldsymbol{u_p}(k) \tag{1.31}$$

Augmenting eqn (1.30) and eqn (1.31), we have

$$\overbrace{\begin{bmatrix}\Delta \boldsymbol{x_p}(k+1)\\ \boldsymbol{y_p}(k+1)\end{bmatrix}}^{\boldsymbol{x_a}(k+1)} = \overbrace{\begin{bmatrix}A_p & 0\\ C_pA_p & 1\end{bmatrix}}^{A_a}\overbrace{\begin{bmatrix}\Delta \boldsymbol{x_p}(k)\\ \boldsymbol{y_p}(k)\end{bmatrix}}^{\boldsymbol{x_a}(k)} + \overbrace{\begin{bmatrix}B_p\\ C_pB_p\end{bmatrix}}^{B_a}\Delta \boldsymbol{u_p}(k) \tag{1.32}$$

$$\boldsymbol{y_a}(k) = \overbrace{\begin{bmatrix}0 & 1\end{bmatrix}}^{C_a}\overbrace{\begin{bmatrix}\Delta \boldsymbol{x}(k)\\ \boldsymbol{y}_m(k)\end{bmatrix}}^{\boldsymbol{x_a}(k)} \tag{1.33}$$

Future state variables for prediction horizon P and control horizon N are calculated as

$$\boldsymbol{x_a}(k_i+1/k_i) = A_a\boldsymbol{x_a}(k_i) + B_a\Delta \boldsymbol{u_p}(k_i) \tag{1.34}$$

$$\boldsymbol{x_a}(k_i+2/k_i) = A_a\boldsymbol{x_a}(k_i+1) + B_a\Delta \boldsymbol{u_p}(k_i+1) \tag{1.35}$$

$$\boldsymbol{x_a}(k_i+2/k_i) = A_a(A_a\boldsymbol{x_a}(k_i) + B_a\Delta \boldsymbol{u_p}(k_i)) + B_a\Delta_p\boldsymbol{u}(k_i+1) \tag{1.36}$$

$$\boldsymbol{x_a}(k_i+2/k_i) = A_a^2\boldsymbol{x_a}(k_i) + A_aB_a\Delta \boldsymbol{u_p}(k_i) + B_a\Delta \boldsymbol{u_p}(k_i+1) \tag{1.37}$$

Defining equation for the general term at P prediction horizon

$$\boldsymbol{x_a}(k_i+P/k_i) = A_a^P\boldsymbol{x_a}(k_i) + A_a^{P-1}B_a\Delta \boldsymbol{u_p}(k_i) + \cdots + A^{P-C}B_a\Delta \boldsymbol{u_p}(k_i+N-1) \tag{1.38}$$

Defining prediction output and future control inputs in vector form

$$\boldsymbol{Y_a} = \begin{bmatrix}\boldsymbol{y_a}(k_i+1/k_i) & \boldsymbol{y_a}(k_i+2/k_i) & \boldsymbol{y_a}(k_i+3/k_i) & \cdots & \boldsymbol{y_a}(k_i+N_p/k_i)\end{bmatrix}^T \tag{1.39}$$

$$\boldsymbol{\Delta U} = \begin{bmatrix}\Delta \boldsymbol{u_p}(k_i/k_i) & \Delta \boldsymbol{u_p}(k_i+1/k_i) & \cdots & \Delta \boldsymbol{u_p}(k_i+N_c-1/k_i)\end{bmatrix}^T \tag{1.40}$$

Output prediction model will become

$$\boldsymbol{Y} = F\boldsymbol{x}(k_i) + \Phi\boldsymbol{\Delta U} \tag{1.41}$$

where

$$F = \begin{bmatrix} C_a A_a \\ C_a A_a^2 \\ C_a A_a^3 \\ \vdots \\ C_a A_a^P \end{bmatrix} \tag{1.42}$$

$$\Phi = \begin{bmatrix} C_a B_a & 0 & 0 & \cdots & 0 \\ C_a A_a B_a & C_a B_a & 0 & \cdots & 0 \\ C_a A_a^2 B_a & C_a A_a B_a & C_a B_a & \cdots & 0 \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ C_a A_a^{P-1} B_a & C_a A_a^{P-2} B_a & C_a A_a^{P-3} B_a & \cdots & C_a A_a^{P-N} B_a \end{bmatrix} \tag{1.43}$$

Assuming that the data vector that contains the set point information is

$$\boldsymbol{R}_{ref} = \overbrace{\begin{bmatrix} 1 & 1 & \cdots & 1 \end{bmatrix}^T}^{P} r(k_i) \tag{1.44}$$

We define the cost function as

$$J = (\boldsymbol{R}_{ref} - \boldsymbol{y_a})^T (R_{ref} - \boldsymbol{y_a}) + \boldsymbol{\Delta U}^T \bar{R} \boldsymbol{\Delta U} \tag{1.45}$$

From equation (3.5)

$$J = (\boldsymbol{R}_{ref} - F\boldsymbol{x}(k_i) - \Phi\boldsymbol{\Delta U})^T (\boldsymbol{R}_{ref} - F\boldsymbol{x}(k_i) - \Phi\boldsymbol{\Delta U}) + \boldsymbol{\Delta U}^T \bar{R} \boldsymbol{\Delta U} \tag{1.46}$$

Rearranging
$$\begin{aligned} J = \; & (\boldsymbol{R}_{ref} - F\boldsymbol{x}(k_i))^T (\boldsymbol{R}_{ref} - F\boldsymbol{x}(k_i)) - 2\boldsymbol{\Delta U}^T \Phi^T (\boldsymbol{R}_{ref} - F\boldsymbol{x}(k_i)) \\ & + \boldsymbol{\Delta U}^T (\Phi^T \Phi + \bar{R}) \boldsymbol{\Delta U} \end{aligned}$$

Then, constraints are defined in three different forms

- Constraints on incremental inputs Δu

$$\boldsymbol{\Delta u}^{min} \leq \boldsymbol{\Delta u}(k) \leq \boldsymbol{\Delta u}^{max}$$

- Constraints on Inputs u

$$\boldsymbol{u}^{min} \leq \boldsymbol{u}(k) \leq \boldsymbol{u}^{max}$$

- Constraints on output y

$$\boldsymbol{y}^{min} \leq \boldsymbol{y}(k) \leq \boldsymbol{y}^{max}$$

Since the control problem is in the form of Δu, all of the constraints are then converted to Δu, which will take the form

$$-C_2\boldsymbol{\Delta U} \leq C_1\boldsymbol{u}(k_i - 1) - \boldsymbol{U}^{min} \tag{1.47}$$

$$C_2\boldsymbol{\Delta U} \leq -C_1\boldsymbol{u}(k_i - 1) + \boldsymbol{U}^{max} \tag{1.48}$$

$$-I\boldsymbol{\Delta U} \leq -\boldsymbol{\Delta U}^{min} \tag{1.49}$$

$$I\boldsymbol{\Delta U} \leq \boldsymbol{\Delta U}^{max} \tag{1.50}$$

$$-\Phi\boldsymbol{\Delta U} \leq -\boldsymbol{Y}^{min} + F\boldsymbol{x}(k_i) \tag{1.51}$$

$$\Phi\boldsymbol{\Delta U} \leq \boldsymbol{Y}^{min} - F\boldsymbol{x}(k_i) \tag{1.52}$$

Combining them

$$\begin{bmatrix} M_1 \\ M_2 \\ M_3 \end{bmatrix} \boldsymbol{\Delta U} \leq \begin{bmatrix} N_1 \\ N_2 \\ N_3 \end{bmatrix} \implies M\boldsymbol{\Delta U} \leq \gamma \tag{1.53}$$

The closed-loop model of the system will be as in Figure 1.14.

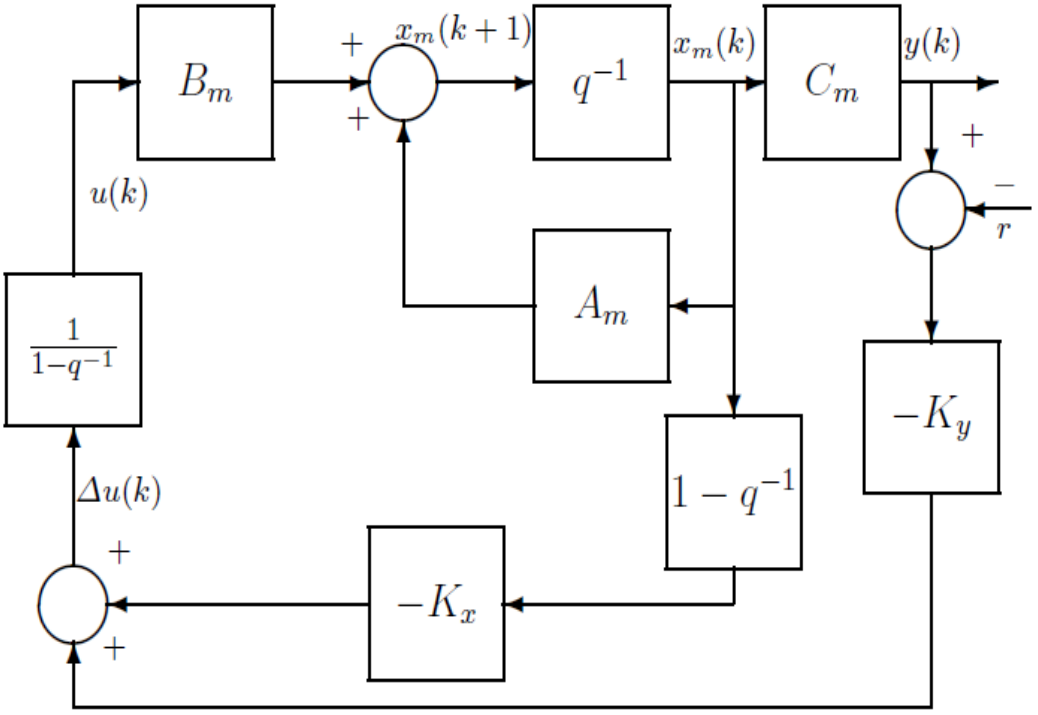

FIGURE 1.14: Linear MPC Closed-loop Model

1.5.2 Non-linear MPC

The NMPC uses the non-linear model of the system, which is represented as

$$\boldsymbol{x}_{np}(k+1) = f(\boldsymbol{x}_{np}(k), \boldsymbol{u}_{np}(k)) \tag{1.54}$$

Where f is used to denote the non-linear function of the model. The prediction model for the non-linear system is found by repeating the equation 1.54 with the value of k from 1 to N.

The necessary condition assumed to find the feedback control law is that there exists an equilibrium state $\boldsymbol{x}_* \in \boldsymbol{X}$ which would be referenced in this problem. This follows that there exists a control input $u_* \in U$ such that

$$\boldsymbol{x}_{m*} = f(\boldsymbol{x}_{m*}, \boldsymbol{u}_{m*}) \tag{1.55}$$

The MPC law reduces the cost function which is defined as the norm of the deviation of the state from the reference state (i.e., error) and/or the norm of the input. The cost function usually involves tuning matrices for states and controlling effort.

These matrices are a trade-off between fast response and low control effort. The non-linear cost function is represented by l and is defined as

$$\min_{u_i} \quad l(\boldsymbol{x}_i, \boldsymbol{u}_i) = \sum_{i=1}^{N-1} \boldsymbol{e}_i^T Q \boldsymbol{e}_i + \boldsymbol{u}_i^T R \boldsymbol{u}_i \tag{1.56}$$

We required that for the system at state $\boldsymbol{x}_*$, we need a control input $\boldsymbol{u}_*$ for staying at that state. This requires the cost function to be zero at the equilibrium point and a positive value otherwise. This is represented as

$$l(\boldsymbol{x}_*, \boldsymbol{u}_*) = 0 \tag{1.57}$$

and

$$l(\boldsymbol{x}_i, \boldsymbol{u}_i) > 0 \quad \text{for} \quad \boldsymbol{x}_i \neq \boldsymbol{x}_* \tag{1.58}$$

The NMPC law is subject to constraints

$$\boldsymbol{x}_{np}(k) \in \boldsymbol{X} \tag{1.59}$$

$$\boldsymbol{u}_{np}(k) \in \boldsymbol{U} \tag{1.60}$$

Where $\boldsymbol{X}$ and $\boldsymbol{U}$ are admissible sets for states and control inputs. The NMPC can used as an algorithm for a given cost function l and a given prediction horizon N. The algorithm is given as

1. Find the value of current state $\boldsymbol{x}_{np} \in \boldsymbol{X}$ and current input $\boldsymbol{u}_{np} \in \boldsymbol{U}$.
2. Solve the non-linear optimal control problem given by equation 1.61 for the given constraints given in equations 1.55,1.59, and 1.60 and find the optimal control sequence $\boldsymbol{u}_* \in \boldsymbol{U}$.
3. Use the first value of optimal control sequence $\boldsymbol{u}_{*0} \in \boldsymbol{U}$ and repeat the process from step 1.

In theory, many variants of NMPC are also available. Depending upon the application, the control designer chooses different MPC variants. The algorithm

discussed above is to track a constant reference (i.e., $\boldsymbol{x}_{ref} = const$). One variant of MPC is the reference varying with time which has a trajectory to follow instead of a constant value. Now the cost function "l" would be a time-varying function, and the tracking of the reference trajectory would require l to be zero.
Another variant of NMPC is to include terminal constraints in the cost function. The cost function then becomes:

$$\min_{\boldsymbol{u}_i} \quad l(\boldsymbol{x}_i, \boldsymbol{u}_i) = \sum_{i=1}^{N-1} (\boldsymbol{e}_i^T Q \boldsymbol{e}_i + \boldsymbol{u}_i^T R \boldsymbol{u}_i) + \boldsymbol{e}_N^T Q \boldsymbol{e}_N \tag{1.61}$$

The condition for the terminal constraint is also the same as for other state constraints. Another variation could be to use weights on different states and control inputs.

1.5.3 MPC with State Estimation

The fighter aircraft contains immeasurable states. That's why the controller needs a modification by including a state estimator in the feedback path. The general block diagram with a state estimator in the feedback path looks like Figure 1.15. The estimator estimates the non-linear states of the system from noisy information and in the presence of external disturbances. The open-loop system will be of the form

$$\boldsymbol{x}_{np}(k+1) = f(\boldsymbol{x}_{np}(k), \boldsymbol{u}_{np}(k), d(k)) \tag{1.62}$$

$$\boldsymbol{y}(k+1) = g(\boldsymbol{x}_{np}(k), \boldsymbol{u}_{np}(k), d(k)) \tag{1.63}$$

The disturbances d can be additive or multiplicative or a combination of both. An estimator finds the best estimation from available noisy data. One of the recent estimator for MPC is the moving horizon estimator (MHE), which is made as an advancement in the early estimators. However, this research is based on reducing the computational complexity of MPC. Therefore, MHE is not considered here due to its high computational demand. This research focuses on the estimation through low computational methods e.g., Kalman Filter.

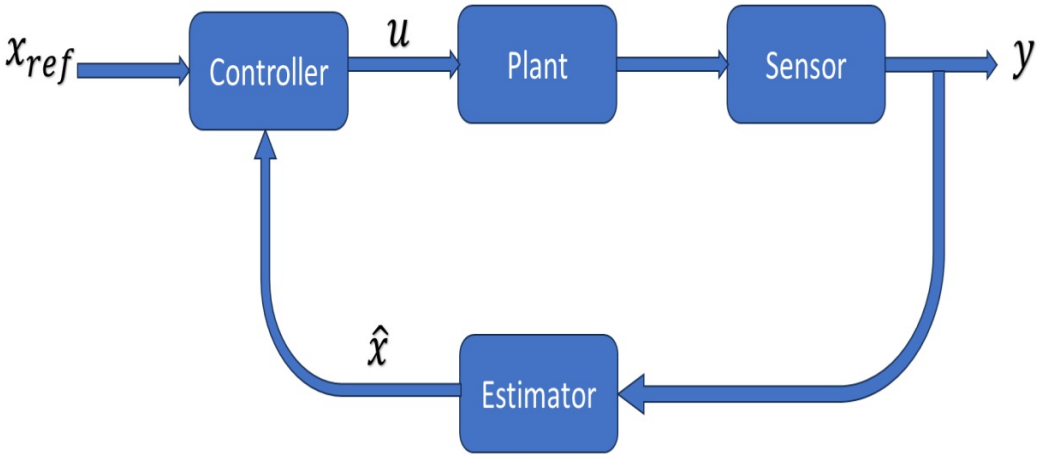

FIGURE 1.15: General Closed-loop Block Diagram with Estimator

Extended Kalman Filter (EKF) is being widely used for non-linear state estimation. The procedure for EKF includes the linearization of the system, followed by the state estimation by using the linear Kalman filter equations. The EKF is the most widely used estimation technique for non-linear systems. However, this technique has implementation difficulties, and tuning difficulties, and gives the best approximation for almost linear systems. The major problem with EKF is that the linearization process may not produce an accurate estimation of the non-linear model. The estimated error propagates throughout each sampling time and will give an inaccurate estimation.

The problem with EKF has been resolved with the use of unscented Kalman Filter (UKF). The UKF samples the non-linear system at several points (called sigma points). The location and weights of these sigma points are chosen to satisfy the starting mean and covariance. The UKF finds applications in many non-linear processes like chemical plants and aircraft.

1.5.4 Disadvantage of MPC

Considering the industrial applications and advancements in model predictive control (MPC), this research is focused on exploring MPC with due consideration to robustness. MPC and its variants have been applied to different aircraft, including civil and military aircraft. It has been applied on F-16 in [29–32] and quite good and effective responses have been obtained. Although MPC is superior to other

techniques in many aspects, it has its disadvantages as well. A major difficulty in MPC is computation complexity, which makes it unrealistic for many applications i.e., aircraft control, etc. There are many reasons for high computations in MPC, which are:

- Prediction matrices are of higher order as compared to simple state-space matrices. Iteratively solving these matrices makes the computations heavy. Matrices size goes on increasing with increasing perdition horizon.
- Long prediction and control horizon also make the computation complex. A long control horizon means more decision variables in the optimization process, which takes more time for calculation.
- Incorporation of constraints also makes the computation heavy.
- The use of fast sampling time also contributes to the computation load. Since MPC calculates the optimal control input in each sampling interval hence use of a fast sampling interval leads to more computation burden.
- Computation is also dependent upon the optimization technique used.

The computations required [33] in solving QP for linear MPC for a centralized controller can be calculated as

$$N \times \left[2N_x^3 + 3N_x^2 N_u + 2N_x N_u^2 + \frac{4}{3}N_u^3 + N_c\left(N_x + N_u\right)^2\right] \tag{1.64}$$

where N is used for the number of time points in the prediction horizon which is obtained by dividing the prediction horizon by the sampling time, N_x is used to denote the number of states, N_u for number of control states and N_c for the number of constraints. The equation (1.64) gives the computations in terms of double-precision floating point operations in each iteration of the QP solver. For the F-16 model given in [34], the total number of open-loop states is 15 including longitudinal as well as lateral states. The total number of inputs of the system is 5. After augmenting with integrators for each state, the number of states would become 30. Including the actuators in the model, the number of states would be $N_x = 35$ and the number of inputs would be $N_u = 5$. The total number of

constraints would $N_c = 90$ including the constraints on states, inputs, and rate of change of inputs. The sampling time and prediction horizon have been obtained from [30] to be $t_s = 1msec$ based on the minimum time constant (less than $\tau = 50msec$ for elevator) and $P = 80\,(4sec)$ based on the maximum time constant (greater than $\tau = 1sec$ for thrust). The number of double precision floating point operations would be approximately 4.744×10^{10} in each iteration of the QP solver. This number would be double for single precision floating point operations and would be approximately 9.488×10^{10}. Suppose there are 20 iterations in each sample time, the number of operations per second would be 1.898×10^{15}. This would be approximately 1.898×10^9 MFLOPS (Millions of single precision floating point operations per second). The maximum achievable computations by using supercomputers claimed by Fujitsu Fugaku by year 2020 are 4.42×10^{11} MFLOPS [35]. However, the computations required for aircraft control are less than the computation speed of the fastest processor. The computation requirement for aircraft will increase for faster systems having shorter sampling times, high prediction horizon, more number of constraints, and use of non-linear MPC. All of these factors indicate that there are limitations to the implementation of MPC in real-time applications involving complex and fast dynamics. However, some modifications in MPC may reduce the computations to make it real-time implementable for fast systems like fighter aircraft.

1.6 Conclusion

The need for advanced control techniques with guaranteed stability and efficiency is the necessity of the modern application i.e., fighter jets. MPC provides both benefits with the advantage of constraint satisfaction. However, the computational complexity of the MPC is a hurdle for many applications. There are numerous ways of reducing the computational effort which involve mathematical reformulation as well as optimization improvement. These methods will be discussed with their pros and cons in the next chapters.

Chapter 2

Literature Review

2.1 Introduction

Computational challenge in MPC is the main focus of research these days to make it real-time applicable in most of the dynamic systems. There are many ways of reducing computational effort in MPC, they are mainly divided into two categories. The first one is to process MPC in a way to reduce initial computational cost e.g., approximating the MPC model, reducing constraints, reducing horizon length, etc. The other way is to reduce the optimization time by using computationally efficient solvers. These computational reduction techniques are listed as

- Reduction of Prediction Horizon
- Simplified System Model
- Explicit MPC
- Warm Start Initialization
- Suboptimal MPC
- Decentralized or Distributed MPC
- Move Blocking
- Constraint Relaxation
- Fast Solver and Hardware Acceleration

- Multirate MPC
- Parametrized Control Policy
- Machine Learning Based Approximations

These ways can be implemented in parallel. However, this research is focused on prepossessing the MPC, while using the same optimization solver. In [36], an NN-based MPC approach has been discussed. NN is used to approximate the nonlinearities and uncertainties of the system, which are further solved by a nonlinear MPC problem. For optimization, two different methods are used, i.e., Mixed Integer Programming (MIP) and Linear Relaxation (RL) methods are used. A numerical example of an inverted pendulum demonstrates the effectiveness and computational efficiency of the proposed optimization methods for its real-time application.

Data-driven MPC (DDMPC) is another approach for reducing the computational complexity of MPC. This technique relies on the input-output data for the formation of the MPC problem instead of deriving the mathematical formulation. The performance of the controller depends on the availability of the data. However, by increasing the date, the computational complexity also increases. In [37], a modified approach for data-driven MPC has been proposed which uses fewer sample points and less number of variables for computational reduction. This scheme has been verified analytically as well as numerically against computational time reduction. The computational complexity reduces by around 17% for a prediction horizon of $N = 10$ with respect to standard Data Driven MPC. However, the performance analysis of the proposed controller is not available.

Another approach for increasing the computational efficiency of MPC is Neural Horizon MPC in [38]. This approach approximates the optimal horizon length and open loop MPC problem through a Neural Network and computes the optimal input based on the approximated model. This method has been applied on the inverted pendulum and compared with baseline MPC for efficiency. The computational complexity can reduce up to 48%. However, this technique only incorporates the state constraints but not input constraints. This scheme also gives sub-optimal performance.

In [39] a different approach for achieving computational efficiency of MPC in which the sampling time and control horizon don't change. This strategy is called Over-Lapping Horizon MPC (OH-MPC). The main idea for OH-MPC is to hold the previous input sequence as long as the new optimized input is not available. While the controller is performing optimization at each sampling interval, the previously available input is utilized. Once the controller computes the new input sequence, the previous input sequence can be discarded and the new input sequence can be utilized as long as the next updated sequence is unavailable. The controller holds the previous input and overlaps it with the current input, which is why it's called the Overlapping MPC. The performance versus computational efficiency can be tuned as per requirement. The results show that this scheme reduces the computational burden by less than 1%.The problem with this scheme is that if the controller is unable to provide the optimized input for more than one sampling time, the performance deteriorates, leading to instability. This could result in disastrous effects in the case of a fast dynamic system i.e., fighter aircraft.

In [40] another approach for reducing the computational burden of the MPC has been introduced. This method involves finding the optimal meta-parameters (i.e., control horizon and event triggering parameter) based on Reinforcement Learning. The controller first computes these parameters offline and finds the most optimal solution to the problem in the first step. Then it uses these parameters in online MPC computations and uses an LQR-based controller while finding this solution. The results claim that the proposed solution reduces computational burden upto 36% than the standard MPC. This method reduces the computational effort to some extent for slow dynamic systems but the problem would be with highly unstable fast dynamic systems i.e., fighter aircraft. This method doesn't guarantee the stability of the system as well.

In [41] another way of implementing MPC called Linearized-Trajectory MPC (LTMPC) is proposed. This method uses a non-linear model of the system and a non-linear estimator to find the future forecast of the states based on the non-linear prediction model and previously available input sequence. Based on this non-linear state information, it linearize the system about the desired trajectory

and then optimizes the control input sequence using QP and a linearized model. It utilizes the first input and uses the remaining available sequence in the next iteration for future prediction of the states. The use of a linear MPC model makes possible the implementation of MPC in real-time. This has been validated using HIL-based simulation. However, for fast systems i.e., fighter aircraft, this idea is tough to implement as there's a need to linearize the system at each sampling instant and then optimize the control input based on the prediction model of the linearized system.

In the paper [42], another way of achieving explicit MPC has been proposed. The explicit MPC solves the optimization problem for each possible combination of inputs, outputs, and their constraints. Then, it utilizes the offline computed control law in the MPC problem. However, this requires the use of a large memory in some cases, thus limiting the use of this technique because of memory constraints. This paper solves this issue by NN-based approximation of the explicit MPC solution, thus reducing the memory requirements for large systems. This paper claims to reduce the computational burden by 6%. However, this paper validates the performance of linear systems and linear MPC. For non-linear MPC this method doesn't say anything about the performance or stability. Secondly, this method is dependent on the NN-based approximation of the controller. Any mismatch, nonlinearities, and external disturbance may deteriorate the performance. For applications like fighter aircraft, the performance reduction is not acceptable.

In [43], a Reinforcement Learning (RL) based MPC has been proposed. This method approximates the MPC problem and cost function and uses this information in the online computation of the controller. This method reduces the online computations by carefully approximating the problem and solving only the RL-based function during online computation. Although the online computations can be reduced, however, the optimal performance is dependent on the accurate approximation of the MPC controller. Any mismatch in the controller model or external disturbance may affect the controller performance for safety-critical applications.

In [44] a feedback control strategy is used in place of MPC for the prediction

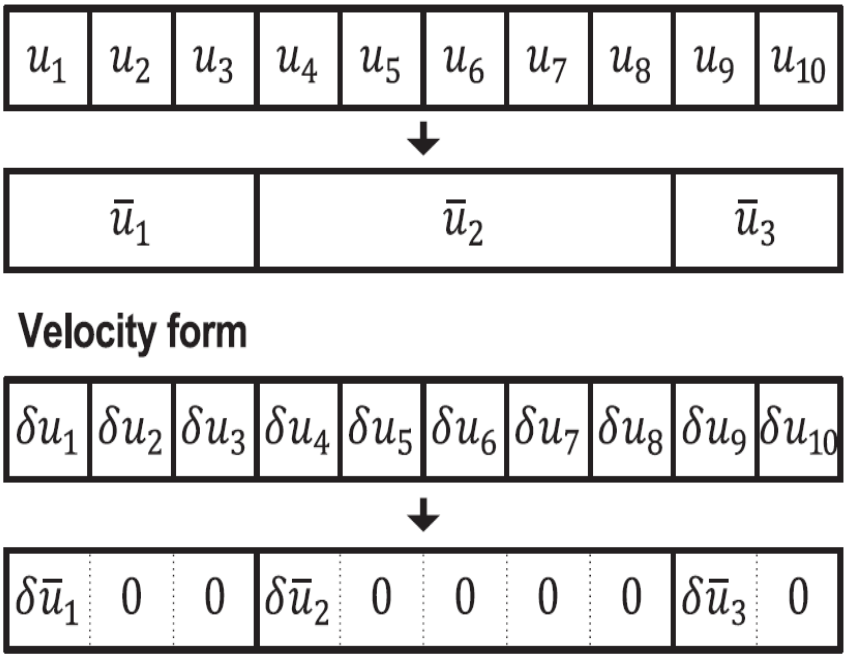

FIGURE 2.1: General Move Blocking Scheme

horizon. It finds different outputs for different values of the inputs and finds the best based on the least mean square error. Thus reducing the computational effort by removing the optimization part from the controller. This is called structured MPC.

Another research area for enhancing computation speed is move-blocking MPC. Move blocking MPC is an input parametrization technique that reduces the number of decision variables by fixing them or their derivatives for a few sampling intervals, thus making blocks as shown in Figure 2.1. The main advantage of using this scheme is that the physical meaning of parametrized input remains the same as the original input. Thus hardware implementation also remains straightforward.

In [45] different move-blocking schemes with their pros and cons have been reviewed. These move-blocking schemes include Input Blocking (IB), Offset Blocking (OB), Delta Blocking (DIB, DOB), and Moving Window Blocking (MWB). IB keeps the value of the input to be constant for a few time steps, but it is not possible to prove recursive feasibility and stability. OB fixes the offset to the feedback control for a certain number of times steps. This scheme provides good closed-loop performance and recursive feasibility, but it doesn't guarantee

closed-loop stability. DB fixes the value of the difference between two consecutive inputs $u_k - u_{k+1}$ or offset from a small number for a few steps. This scheme is also unable to provide closed-loop stability. MWB is time-dependent whereas the blocking structure is time-varying. This scheme, besides improving closed-loop performance, provides recursive feasibility and stability. Optimal blocking positions (called blocking structures) have to be found online in each sampling interval for optimal performance. Blocking positions are integer values and finding the optimal blocking structure along with a regular MPC problem makes it a mixed-integer problem (MIP). MIP is non-convex, hence there are no global minima for this problem. Thus solving MIP makes the problem computationally more complex. So, there's a need to find schemes to derive optimal blocking structure without solving MIP.

While considering optimal time-dependent blocking structure, some schemes convert MIP to a simple floating-point problem by finding the blocking structure separately. In [46] a variable horizon model predictive control scheme has been proposed in which control horizon length can also be found by solving an optimization problem. To reduce the computational burden for long horizon length time-varying blocking MPC has been applied. In this scheme, every admissible blocking structure has been solved using an optimization problem in parallel. The proposed method provides finite time completion and recursive feasibility despite disturbances but it is unable to reduce the computational complexity of MPC because of parallel processing.

In [47] move blocking MPC has been modified by finding the minimum horizon length and the number of blocks. In this MIP has been solved explicitly for finding the optimal control horizon and number of blocks used. Then in the next step, a blocking structure with the maximum region of attraction (ROA) has been chosen as the optimal blocking structure. However, the problem with this scheme is that it uses a time-invariant blocking structure. It's quite probable that the blocking structure which is optimal in a one-time step may not be optimal in another time step. Hence this scheme can't provide the optimal solution.

In [48] time-varying blocking structure has been computed by using an algorithm

(based upon mean square error). However, this method is unable to provide guaranteed stability. The performance of the algorithm also depends upon an initial guess about input.

In [49] a semi-explicit approach has been proposed for finding the optimal time-varying blocking structure. The proposed scheme first finds the critical region for every admissible blocking structure explicitly. Then searches for the critical region online according to the current state. This scheme reduces computational complexity while preserving recursive feasibility and stability. However extra computation appears in searching for the critical region online. This complexity grows with increasing horizon length.

The literature is summarized in Table (2.1). The table shows the disadvantages associated with the specific computational reduction technique. However, all of these schemes have been applied to different systems and the resulting simulation time has been reported. So, it's not possible to construct a direct computational comparison between schemes. However, the performance analysis and the disadvantages associated with the performance show that to reduce computational complexity and preserve performance there is a need for the design and application of a MPC-based control scheme for reduced computation time and optimal performance. The designed scheme should also provide closed-loop stability in the presence of bounded disturbances.

2.2 Problem Statement

Piloting advanced fighter aircraft without advanced flight control is impossible. The flight control system serves as a medium of communication between the pilot and the aircraft, interpreting and executing commands of the pilot. Advanced control is central to modern aircraft and with increasing capabilities, the use of conventional control schemes is proving less fruitful. MPC provides a better solution in terms of constraint satisfaction and performance. However, the problem with MPC is a high computation which limits its application. Fighter aircraft have fast dynamics that require a fast response from the controller. Therefore, it is necessary to find a scheme that will run MPC faster. This research, therefore, targets

TABLE 2.1: Literature Survey

Ref	Description	Drawback
[37]	Data Driven MPC	Doesn't guarantee performance
[38]	Neural Horizon MPC	Doesn't incorporates input constraints
[39]	Over Lapping Horizon MPC	Non-optimal Unstable
[40]	Finds the optimal meta-parameters based on Reinforcement Learning	Not Suitable for fast systems
[41]	Linearized-Trajectory MPC	Linearization at each sampling instant
[42]	NN-based approximation of the explicit MPC	Doesn't guarantee performance or stability for non-linear systems
[43]	Reinforcement Learning (RL) based MPC	No optimization at each sampling interval
[44]	Structured MPC	Removes optimization part of MPC
[45]	Input Blocking (IB) Offset Blocking (OB) Delta Blocking (DIB,DOB) Moving Window Blocking (MWB)	Uses MIP for finding optimal blocking structure
[46]	Variable horizon optimal blocking structure Time-varying blocking structure	Finds optimal by solving all admissible blocking structures in parallel
[47]	Finds minimum horizon length and minimum number of blocks in each iteration	Uses time invariant blocking structure
[48]	Proposed an algorithm for finding time-varying blocking structure	Unable to provide guaranteed stability
[49]	Finds critical region for every admissible blocking structure explicitly Searches for optimal region online	Online search of critical region affects computations

the Computationally efficient Model predictive Control design for modern fighter aircraft with optimal performance during complex maneuvers in the presence of external disturbances.

2.3 Objectives and Significance

Model Predictive Control has proved to be a very efficient and popular control technique in various applications. MPC was primarily developed for the petroleum industry and power plants. The performance of MPC attracted researchers from other fields e.g., refining, petrochemical, chemical, Pulp & Paper, Air & Gas, Mining, Food processing, polymer, Furnaces, Automobiles, and Aerospace. Different vendors like Aspen, Honeywell, Adersa, Invensys, and SGS prepared different packages of linear as well as non-linear MPC for different applications of the above-mentioned fields. Nowadays, MPC is the most widely used technique in different industries and got the highest impact rating after PID from industry [7]. MPC implementation is still limited to slow processes because of extra computation. Enhancing computation time in MPC is still a research challenge. Therefore, research will be the optimal and computationally efficient Model Predictive control-based design for modern fighter aircraft. MPC has proved to be an efficient control technique for flight control and carefree maneuvers. The proposed research will target the development and simulation of novel robust MPC-based control techniques for modern fighter aircraft and achieve level 1 flying and handling qualities [50, 51] as given in the Appendix. The fighter aircraft to be used for validating the efficiency of the proposed controller will be the F-16 Fighting Falcon. The major advantage of using F-16 in research is that its model and parameters are easily available in the literature and comparison of results is easy. Deployment of the flight control system on actual aircraft and field trials is not included in the scope of the project. Several objectives will be achieved during the project timeline.

1. Analysis of linear and non-linear (6 DOF) mathematical models of fighter aircraft in MATLAB

2. Finalization of performance characteristics that will ensure adequate level 1 Flying & Handling qualities
3. Simulation and analysis of existing approaches (LQR) for control of aircraft
4. Application of linear MPC, simulation, and results validation
5. Design and application of a novel control strategy for reduced computation and optimal performance

2.4 Methodology and Technique

The research procedure and methodology is summarized in Table 2.2.

TABLE 2.2: Research Methodology

Tasks	**Sub Tasks**	**Key Deliverables**
Literature survey	Modeling and simulation of aircraft Numerical characterization of level 1 flying and handling qualities Review of the controls applied to the aircraft Move Blocking MPC schemes for computational reduction	Detailed report covering the chosen mathematical model, relevant aircraft data, simulation methodology, gap analysis of applied control and numerical values of various parameters for flying and handling qualities.
Analysis of model		State space model of aircraft in MATLAB Open-loop simulation and Model validation Comparison of results with the published work

continued...

Tasks	**Sub Tasks**	**Key Deliverables**
Implementation of LQR over linear state-space model		LQR design Simulation and validation of results
Benchmark MPC	Mathematical Validation	Choice of the cost minimization algorithm Mathematical validation of the derivation for the linear MPC regulation problem MATLAB function for linear MPC regulation problem Mathematical validation of Linear MPC Tracking problem Modification of MPC-based regulation function according to the tracking problem
	Linear Simulation	Implementation of linear MPC over the linear model Comparative analysis of MPC-based results against LQR controlled/H-infinity Comparison with the LQR-controlled model
Novel MPC Design	Review of previous schemes Design of novel scheme	Mathematical modeling Closed loop simulations Performance Comparison

continued...

Tasks	Sub Tasks	Key Deliverables
Novel Controller over 6 DOF Model		Implementation of controllers in MATLAB over 6 DOF model Results comparison with linear MPC-based results
Conference / Journal Papers		Write-up Review Presentation
Thesis		Write-up Review Final Defense

2.4.1 Open Loop Simulation Model

The open-loop model of the aircraft at $v = 500$ ft/s and $h = 15000$ft is given in state-space form [34]. Since aircraft control is a complex problem, designing a single controller for the whole aircraft may lead to a non-optimal solution. For this reason, it is assumed that aircraft have decoupled motion in longitudinal and lateral axes named as longitudinal and lateral modes. These models have been obtained by assuming negligible coupling between the two modes. The longitudinal decoupled model of F-16 in state-space form is given in the next section.

2.4.1.1 Longitudinal Model

A longitudinal model of the system in state-space form is as

$$A = \begin{bmatrix} 0 & 500 & 3.553 \times 10^{-10} & -500 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 & 0 \\ 1.074e4 & -32.17 & -1.321e-2 & -2.669 & -1.186 & 1.565e-3 & 3.870e-2 \\ 2.076e-6 & -3.681e-13 & -2.552e-4 & -6.761e-1 & 9.392e-1 & -2.48e-7 & -1.437e-3 \\ 9.632e-12 & 0 & -1.184e-9 & -5.757e-1 & -8.741e-1 & 0 & -1.188e-1 \\ 0 & 0 & 0 & 0 & 0 & -1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & -20.20 \end{bmatrix}$$

$$B = \begin{bmatrix} 0 & 0 & 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 20.20 \end{bmatrix}^T$$

where states are h, θ, v, α,q,δ_t and δ_θ representing height (ft), pitch angle (rad), longitudinal velocity (ft/s), angle of attack (rad), pitch rate (rad/s), throttle deflection (lb), and elevator deflection (deg) respectively. Here, only the longitudinal model has been considered. The lateral model of the aircraft in closed-loop control is considered in later chapters.

2.4.2 MPC based Control

MPC is an online control technique that predicts the system's future behavior based on current states. MPC solves FHOCP (Finite Horizon Optimal Control Problem) and finds the optimal control effort for future time steps. Then it applies only the first input and discards the others. After that shift the prediction horizon one step ahead and solve the optimization problem again at the next sampling interval. This process is repeated at each sampling instant, hence called online computation.

A linear MPC model is used as a benchmark MPC. The linear model used here is the augmented model of the system with an integrator. The state space augmented model is defined as

$$\boldsymbol{x}_a(k+1) = A_a \boldsymbol{x}_a(k) + B_a \delta \boldsymbol{u}_p(k) \tag{2.1}$$

The output model is defined as:

$$\boldsymbol{y}_a(k) = C_a \boldsymbol{x}_a(k) \tag{2.2}$$

Where the subscript a shows the augmented model. The objective is to minimize error function and the control effort

$$J = \min_{\delta \boldsymbol{u}} \quad \boldsymbol{e}_i^T Q \boldsymbol{e}_i + \delta \boldsymbol{u}_i^T r \delta \boldsymbol{u}_i \tag{2.3}$$

where i denotes the current time step and $i = 1 \rightarrow N$. The constraints on the system are

$$\boldsymbol{x}_m(k+1) = A_m \boldsymbol{x}_m(k) + B_m \boldsymbol{u}_m(k) \tag{2.4}$$

Where m denotes the model of the system. The inequality constraints are given as

$$\boldsymbol{x}_m^{min} \leq \boldsymbol{x}_m \leq \boldsymbol{x}_m^{max} \tag{2.5}$$

$$\boldsymbol{u}_m^{min} \leq \boldsymbol{u}_m \leq \boldsymbol{u}_m^{max} \tag{2.6}$$

$$\Delta\boldsymbol{u}_m^{min} \leq \Delta\boldsymbol{u}_m \leq \Delta\boldsymbol{u}_m^{max} \tag{2.7}$$

The tuning matrices q and r are defined as

$$q = eye(n_x) \tag{2.8}$$

$$r = eye(n_i) \tag{2.9}$$

Where n_x denotes the number of states and n_i denotes the number of inputs.

2.4.3 Closed-Loop Simulation

2.4.3.1 Performance Characteristics

Performance characteristics required by the system are given in Table 2.3 [34, 52].

2.4.3.2 Closed Loop Model

A closed-loop longitudinal model of the system in Simulink is given in Figure 2.2. The Simulink blocks are mainly S-function for controller implementation, state-space for aircraft model, and delay functions.

TABLE 2.3: Performance Characteristics of Longitudinal Model

Longitudinal Mode			
States	**Constraints**	**Control Objectives**	
Angle of Attack (α)	$-20° \leq \alpha \leq 90°$	$60°$	
Control Surface	**Min. Deflection**	**Max. Deflection**	**Rate**
Elevators	$-25°$	$25°$	$60°$ $/sec$
Throttle (lb)	1000	19000	1000 lb/sec

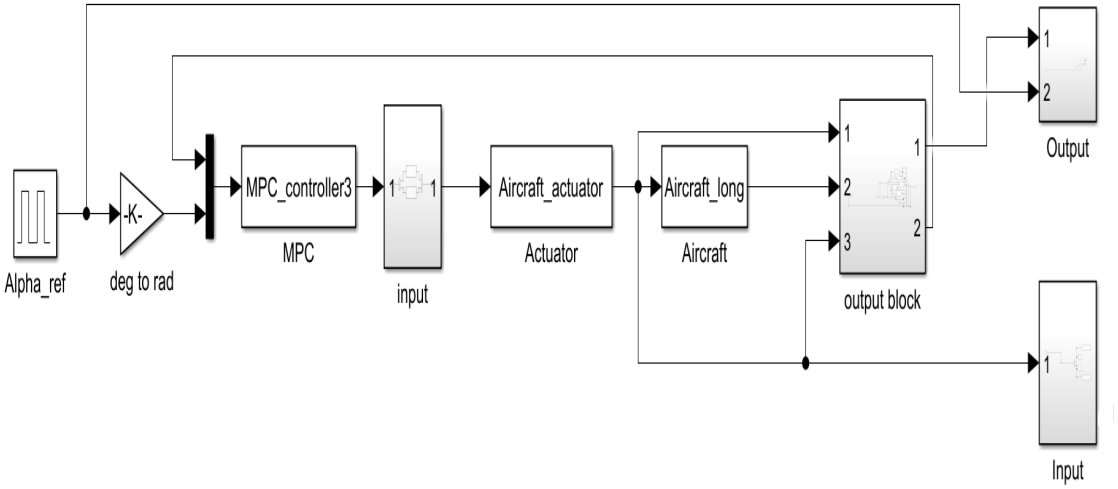

FIGURE 2.2: Closed Loop Simulink Model for MPC

2.4.3.3 Closed Loop Results

Closed loop simulations are performed using linear MPC and a linear model of the aircraft around the equilibrium point, considering the high angle of attack performance specifications. For performing high angle of attack maneuver requirement for the longitudinal mode is 60° angle of attack at 1 sec and then back to 0°at 6 sec [52] based on the constraints given Table 2.3. The reference signal is generated by an impulse block. The Cost matrices chosen are for output angle of attack α, thrust δ_t and elevator deflection δ_e

$$q = \begin{bmatrix} 1 & 0 & 0 \\ 0 & 0 & 0 \\ 0 & 0 & 0 \end{bmatrix}$$

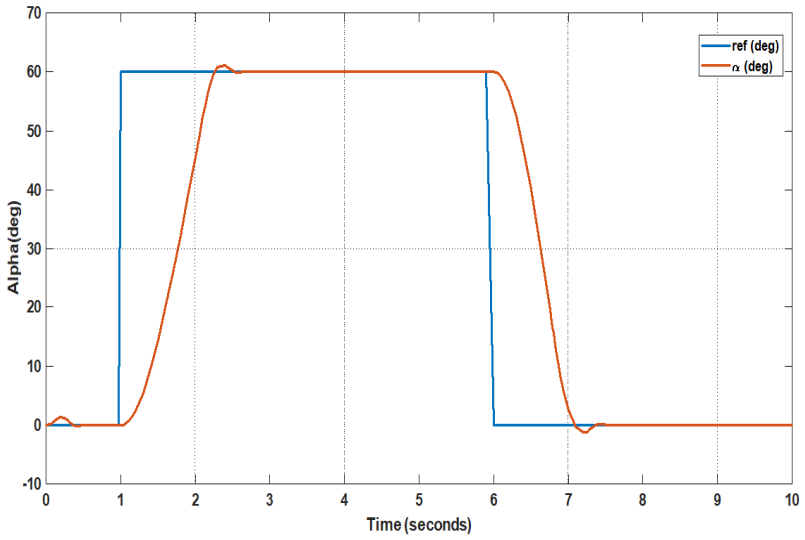

FIGURE 2.3: Pitch Angle tracking using Linear MPC

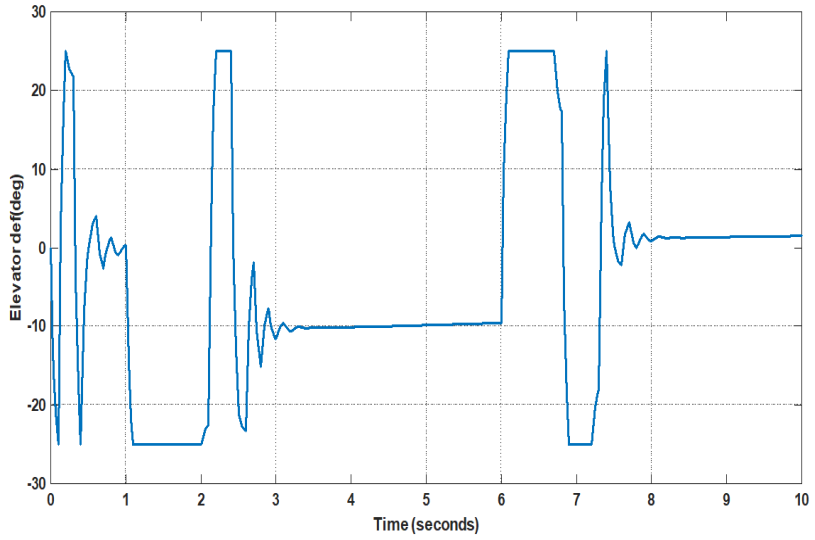

FIGURE 2.4: Elevator Control Deflection for Linear MPC

$$r = \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix}$$

The main reason for choosing values of Q corresponding to the states δ_t and δ_e are to free the control deflection so that it can attain maximum deflection if necessary. Closed-loop response to 60° rectangular reference input in the angle of attack α is given in Fig (2.3), control surface deflections in Figure 2.4-2.5.

The Figures 2.4-2.5 show that the designed controller provides good performance.

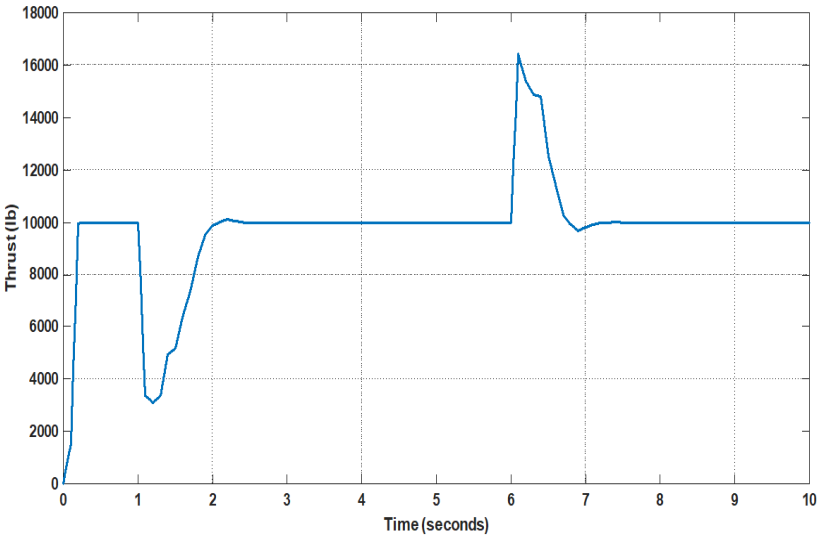

FIGURE 2.5: Thrust Control input for Linear MPC

It gives settling time of about 1.5 *sec* for 60° angle of attack while maintaining the control input constraints.

2.4.4 Discussion on Results

Simulation is performed to test the controller's efficiency against other schemes and to compare the results against already published work. In [53] the high angle of attack maneuver is performed by using SMC (Sliding Mode Controller), which provides a good reference for fast maneuvers. Simulation results show that MPC provides better performance in terms of time domain specifications , e.g., rise time, settling time, etc. The performance of MPC is quite good as it achieves a settling time of 1.5 sec for 60° angle of attack while in [53] settling time is around 4*sec*. The main reason behind this fast response is that MPC makes the best utilization of control surfaces while maintaining the constraints. In other controllers, constraints are dealt with by changing the tuning parameters. In those controllers, it's probable that the controller prevent the elevators from full deflection by tightening the tuning parameters or the controller may require from the elevators to deflect more than their capacity because of choosing low values of tuning parameters. However, MPC enforces constraints by design which not

only provides optimal performance but also reduces the pilot's effort in carefree maneuvering.

2.5 Applications

This research project is based on theoretical as well as practical applications. The major aim of the project is to provide a computationally efficient MPC controller for aircraft control. MPC provides the best solution in terms of optimal performance and constraint satisfaction. However, MPC is not suitable for high-speed systems because of computational complexity. Although modern controllers solve the high computational problem easily, for applications like fighter aircraft faster response is required. A lack of desired response will lead to a catastrophic disaster. This research is focused on the high-speed response to the MPC problem which will be directly implementable on fighter aircraft.

Additional benefits:

Pakistan has developed its own UAV's like Burraq, Satuma Jasoos II, Satuma Mukhbar, Ababeel Aerial drone, and GIDS Shahpar, etc., for reconnaissance, surveillance, and strike. The design of an advanced controller is also a challenging task in UAVs. In addition to the control of fighter aircraft, this technique is equally useful for designing an optimal controller for guidance and performance improvement of UAVs. Control algorithms being pursued in this research can also be extended to indigenous missile and space systems being developed by various organizations.

2.6 Contributions

The proposed solution to the problems discussed in the literature survey and the contribution of the current research are summarized as follows:

- We proposed time-varying faults during complex maneuvers of fighter aircraft and an algorithm for fault detection and fault control.

- We proposed an LQR-based optimal blocking scheme for MPC for computational reduction of MPC.
- An NN-based computationally efficient offset-free NMPC is proposed.
- A computationally efficient NN-based offset-free scenario-based NMPC is proposed.

2.7 Conclusion

Many researchers in the modern era have tried to solve the computational disadvantage of MPC. All of the computational reduction techniques have their limitations along with computational advantages. For applications like fighter aircraft, the performance can't be compromised. So, there's a need to find an optimal solution along with computational efficiency. The next chapter discusses the application of linear MPC for high angle of attack maneuver control of fighter aircraft along with limitations due to aerodynamic forces.

Chapter 3

Fault-Tolerant Control of Fighter Aircraft using MPC

3.1 Introduction

In flight control, the aircraft may encounter different types of faults. The faults include actuator fault and sensor fault [54]-[55]. This research focuses on the actuator fault. The actuator fault can further be categorized as a permanent and temporary fault. The permanent fault is also known as the permanent jamming or stuck fault. In this fault, the actuator gets stuck in a certain position and the aircraft is unable to recover from this fault. This fault can be catered to by redefining the reference command and utilizing other control surfaces. The temporary fault is also known as temporary jamming or stall load. In this type of fault, the aircraft can't achieve full deflection of the control surfaces while performing complex maneuvers due to high aerodynamic forces. The original control limits can be recovered after the maneuver [56, 57]. This type of fault is commonly encountered in most fighter aircraft and will be the focus of the research.

In temporary fault or stall load, it is common practice in the literature to assume that control surfaces can't deflect above the reduced fixed limit. However, in actuality, this is not true. In a real environment, the stuck position may not be fixed and may change with time or with the increasing complexity of the maneuver. So,

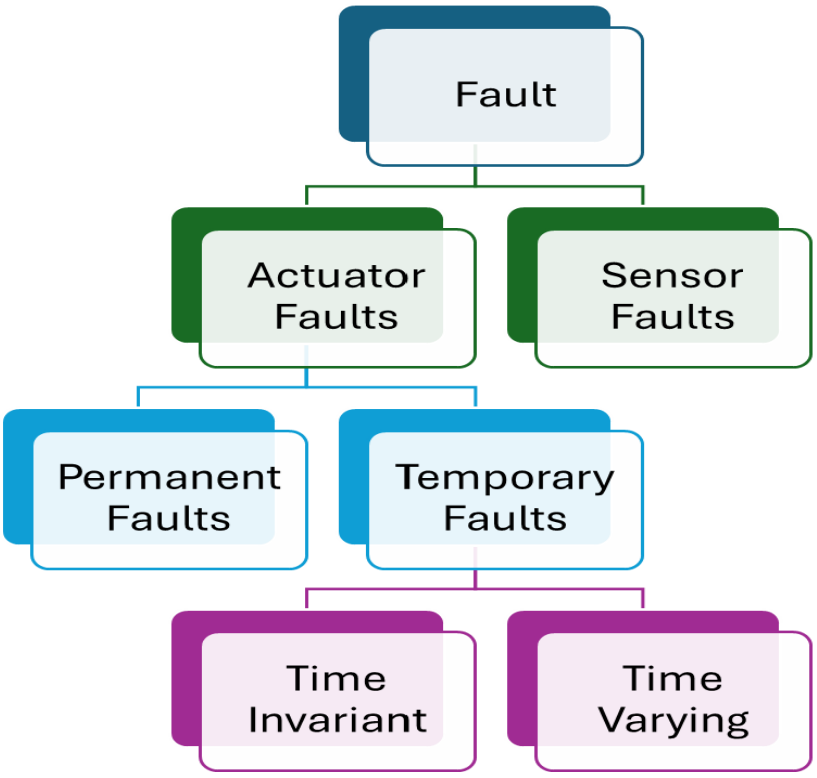

FIGURE 3.1: Categories of Faults

this research is focused on time-varying faults. The types of faults are summarized in Figure 3.1

Various control techniques are available in the literature that guarantee robustness as well as optimality for fighter aircraft. Most of the techniques have been employed for fault-tolerant control efficiently [58–61]. The control techniques that we can apply include PID (Proportional Integral Derivative) control, EA (eigenstructure Assignment), LQR (Linear Quadratic Regulator) control, MPC (Model Predictive Control), LQG (Linear Quadratic Gaussian) control, H_∞ control, and SMC (Sliding Mode Control). This research is focused on the use of MPC as it can handle constraints by design and has a proven record of handling faulty scenarios [60, 62–65]. MPC, for handling faults, applies reconfiguration at the actuator limits for fault tolerance.

MPC can't handle faults until it gets some information about when the fault arises and what is the new control limit of the actuator. This information can be provided by a fault detection (FD) block working in parallel with the control block. The fault detection block actively monitors the timing and control limits of

the fault and informs the controller to apply reconfiguration. The integration of FD and MPC along with guaranteed fault tolerance has been studied in [66, 67]. Different fault detection algorithms exist in the literature. In this research, the root mean square value of the mean deviation from the original control limit is calculated. The fault detection module also provides information about the end of stall load so that the controller may reconfigure for original control limits [68]. The rest of the chapter is organized as follows: Section 3.2 describes the fighter aircraft model. In section 3.3, the linear MPC controller has been discussed. Section 3.4 details the fault detection algorithm and interaction with the MPC controller. Section 3.5 gives the simulation results for the proposed time-varying fault-tolerant scheme. Section 3.6 gives the conclusions and future work.

3.2 Aircraft Model

The open-loop model of the aircraft at $v = 500\ ft/s$ and $h = 15000 ft$ is given in state-space form [69]. Since aircraft control is a complex problem, designing a single controller for the whole aircraft may lead to a non-optimal solution. For this reason, it is assumed that aircraft have decoupled motion in longitudinal and lateral axes named longitudinal and lateral modes. These models have been obtained by assuming negligible coupling between the two modes. The longitudinal decoupled model of F-16 in state-space form is given as:

3.2.1 Longitudinal Model

A longitudinal model of the system in state-space form is as

$$A = \begin{bmatrix} 0 & 500 & 3.553e-10 & -500 & 0 \\ 0 & 0 & 0 & 0 & 1 \\ 1.074e4 & -32.17 & -1.321e-2 & -2.669 & -1.186 \\ 2.076e-6 & -3.681e-13 & -2.552e-4 & -0.6761 & 0.9392 \\ 9.632e-12 & 0 & -1.184e-9 & -0.5757 & -0.8741 \end{bmatrix}$$

$$B = \begin{bmatrix} 0 & 0 & 1.565e-3 & -2.48e-7 & 0 \\ 0 & 0 & 3.87e-2 & -1.437e-3 & -0.0188 \end{bmatrix}^T$$

where states are h, θ, v, α, q, and inputs δ_t and δ_θ representing height (ft), pitch angle (rad), longitudinal velocity (ft/s), angle of attack (rad), pitch rate (rad/s), throttle deflection (lb) and elevator deflection (deg) respectively.

3.2.2 Actuator Model

The actuator model used in F-16 is given as

$$A = \begin{bmatrix} -1 & 0 \\ 0 & -20.20 \end{bmatrix}$$

$$B = \begin{bmatrix} 1 & 0 \\ 0 & 20.20 \end{bmatrix}$$

Where states are the same as the inputs of the aircraft model and the inputs are the control inputs for the throttle and elevator respectively.

3.3 MPC

MPC is an online control technique that predicts the future behavior of the system based on current states. MPC solves FHOCP (Finite Horizon Optimal Control Problem) and finds the optimal control effort for future time steps. Then it applies only the first input and discards the others. After that shift the prediction horizon one step ahead and solve the optimization problem again at the next sampling interval. This process is repeated at each sampling instant, hence called online computation.

3.3.1 MPC Controller Model

The process of MPC is given in [70, 71].
Given a discrete-time linear plant with a state-space model given as

$$pmbx_m(k+1) = A_m \boldsymbol{x}_m(k) + B_m \boldsymbol{u}_m(k) \tag{3.1}$$

$$\boldsymbol{y}_m(k) = C_m \boldsymbol{x}_m(k) \tag{3.2}$$

The state space model can be augmented with an integrator to eliminate steady-state error. Also, it fulfills the constraints on the rate of change of input. For that, the system will be like

$$\overbrace{\begin{bmatrix}\Delta \boldsymbol{x}_m(k+1)\\ \boldsymbol{y}_p(k+1)\end{bmatrix}}^{\boldsymbol{x}_a(k+1)} = \overbrace{\begin{bmatrix}A_m & 0\\ C_m A_m & 1\end{bmatrix}}^{A_a}\overbrace{\begin{bmatrix}\Delta \boldsymbol{x}_m(k)\\ \boldsymbol{y}_p(k)\end{bmatrix}}^{\boldsymbol{x}_a(k)} + \overbrace{\begin{bmatrix}B_m\\ C_m B_m\end{bmatrix}}^{B_a}\Delta \boldsymbol{u}_m(k) \tag{3.3}$$

$$y_a(k) = \overbrace{\begin{bmatrix}0 & 1\end{bmatrix}}^{C_a}\overbrace{\begin{bmatrix}\Delta \boldsymbol{x}(k)\\ \boldsymbol{y}_m(k)\end{bmatrix}}^{\boldsymbol{x}_a(k)} \tag{3.4}$$

Defining prediction output and future control inputs in vector form, the output prediction model will become

$$\boldsymbol{Y} = F\boldsymbol{x}(k_i) + \Phi \Delta \boldsymbol{U} \tag{3.5}$$

where

$$F^T = \begin{bmatrix}C_a A_a & C_a A_a^2 & C_a A_a^3 & \dots\\ C_a A_a^P & & & \end{bmatrix} \tag{3.6}$$

$$\Phi = \begin{bmatrix} C_a B_a & 0 & \cdots & 0\\ C_a A_a B_a & C_a B_a & \cdots & 0\\ \vdots & \vdots & \ddots & \vdots\\ C_a A_a^{P-1} B_a & C_a A_a^{P-2} B_a & \cdots & C_a A_a^{P-N} B_a \end{bmatrix} \tag{3.7}$$

We define the cost function as

$$J = (\boldsymbol{R}_{ref} - \boldsymbol{Y}_a)^T (\boldsymbol{R}_{ref} - \boldsymbol{Y}_a) + \Delta \boldsymbol{U}^T \bar{R} \Delta \boldsymbol{U} \tag{3.8}$$

Then constraints are defined in three different forms of the Δu taking the form

$$\begin{bmatrix}M_1\\ M_2\\ M_3\end{bmatrix} \Delta \boldsymbol{U} \leq \begin{bmatrix}N_1\\ N_2\\ N_3\end{bmatrix} \implies M \Delta \boldsymbol{U} \leq \gamma \tag{3.9}$$

3.4 Fault Detection (FD) Algorithm

The fault detection module works by finding the root mean square of the error in elevator deflection over a certain range of values [68]. The cost function that determines the fault is given as

$$C_e = \sqrt{\frac{1}{dd}\sum_{t+1-d}^{t}\left(\delta_{e_{actual}} - \delta_{e_{predicted}}\right)} \tag{3.10}$$

Where dd is used to cater to the detection delay. The value of dd is not fixed. Choosing a low value of dd may lead to a false alarm. The high value of dd caters to detection delays. However, a very high value may also give a fault alarm after the fault has gone long ago. So, its choice should be such that it caters to detection delays as well as false alarms. The cost can be further compared to a certain threshold for the detection of the fault. The fault is detected as

$$If\ C_e(t) \leq C_e^{th}\ ,\ \text{The elevator is fault-free} \tag{3.11}$$

$$If\ C_e(t) \geq C_e^{th}\ ,\ \text{The elevator has fault} \tag{3.12}$$

Where C_e^{th} is the threshold value of the error. This value is also a trade-off between false alarms and detection delays. A very high value may lead to a false alarm and a low value may not cater for the detection delays. Its value can be found by getting the root mean squared value of the error in fault-free scenarios.

The fault-detection algorithm works in parallel with MPC as shown in Table 3.1. Where δ_e^f is the elevator deflection in the event of fault detection and δ_e^o is used to represent the original limits of the elevator. The variables $\bar{\delta}_e$ and $\underline{\delta}_e$ are the upper and lower bounds, respectively. The delay in time should be large enough to complete the maneuver.

Thus fault detection module in parallel with the MPC module finds the fault in the control surface for a specific amount of time. If the fault persists for a certain amount of time, then the controller reduces the constraint limits. The FD module continues to look for a fault. When the fault is removed, the controller restores its original limits.

TABLE 3.1: Fault-Detection Algorithm

Sq	**FD**	**MPC**
1	Check C_e if $C_e(t) \leq C_e^{th}$ The elevator has no fault Else if $C_e(t) \geq C_e^{th}$ Fault Detected	
2		If Fault Detected while $t < t + delay$ if the fault on the upper bound $\bar{\delta}_e = \delta_e^f - \gamma$ if the fault on the lower bound $\underline{\delta}_e = \delta_e^f + \gamma$ end
3		At $t \geq t + delay$ if the fault on the upper bound $\bar{\delta}_e = \delta_e^f + \alpha$ if the fault on the lower bound$\underline{\delta}_e =$ $\delta_e^f - \alpha$ end
4	If $\left\|\delta_e - \delta_e^f\right\| \leq \alpha$ Stall load active Goto step 2 else: Stall load ended	

continued...

Sq	**FD**	**MPC**
5		Restore original bounds
		$\bar{\delta}_e = \delta_e^o$
		$\underline{\delta}_e = \delta_e^o$

3.5 Closed-Loop Simulations

3.5.1 Closed-loop Formulation

The proposed algorithm is tested in Matlab Simulink for validation. A linear model of the aircraft and linear MPC is considered for simulation. The fault-detection module works in parallel with the MPC module. It signals the MPC module for reconfigurations in case of a faulty scenario. The FD module keeps on checking the controller output as well as the actuator's physical parameters. In case of any fault, it signals the controller to adjust limits. The reference for command for controller is to perform high angle of attack maneuver i.e., $\alpha = 60°$ MPC and FD module is implemented in closed-loop formulation as shown in Figure 3.2. For implementing faults in simulation, a saturation block is used in which the control limits vary with time.

The closed-loop results become better because of this parallel combination. In closed loop simulations, the FD block detects any failure in control surface based on its commanded value and original value. If difference persists for a certain amount of time, the FD block suggest for reduced control limit in upper or lower bound based on the fault. The controller then perform online optimization based on reduced limits thus giving better performance.

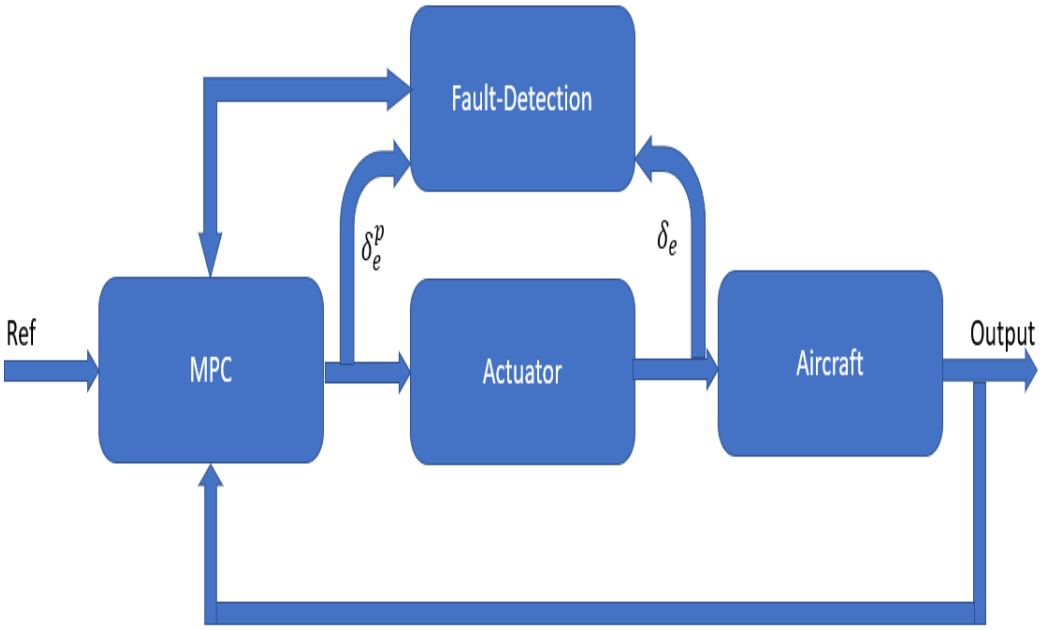

FIGURE 3.2: Block Diagram with Fault Detection Module

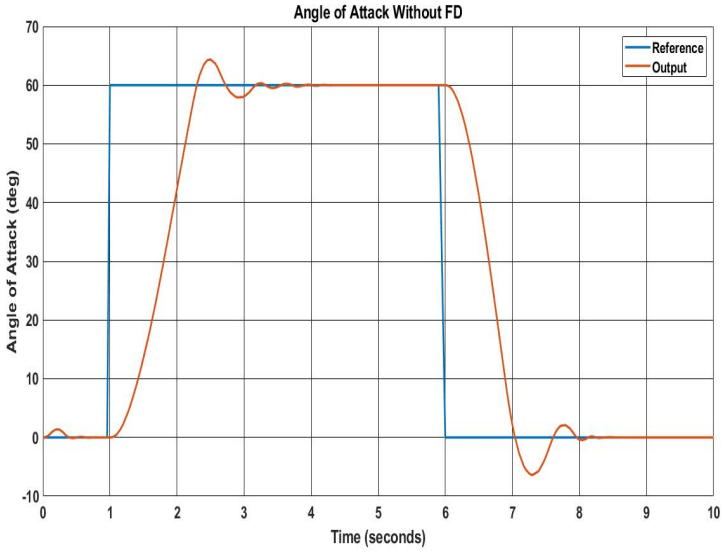

FIGURE 3.3: Closed-Loop α Tracking without FD

3.5.2 Closed-Loop Results

The closed-loop results of MPC have been compared with and without fault-detection modules. The results without the FD module are given in Figure 3.3. The control effort for achieving α without the FD module is given in Figure 3.4. The angle of attack tracking for closed-loop system after inclusion of FD module is given in Figure (3.5).

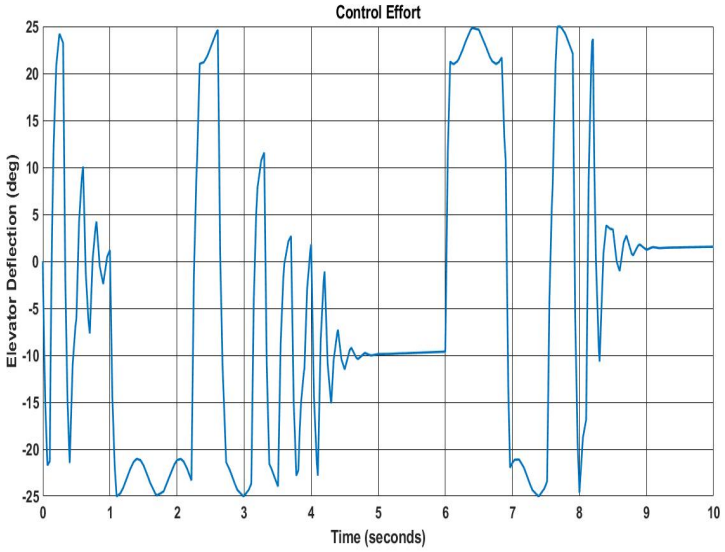

FIGURE 3.4: Elevator Deflection without FD

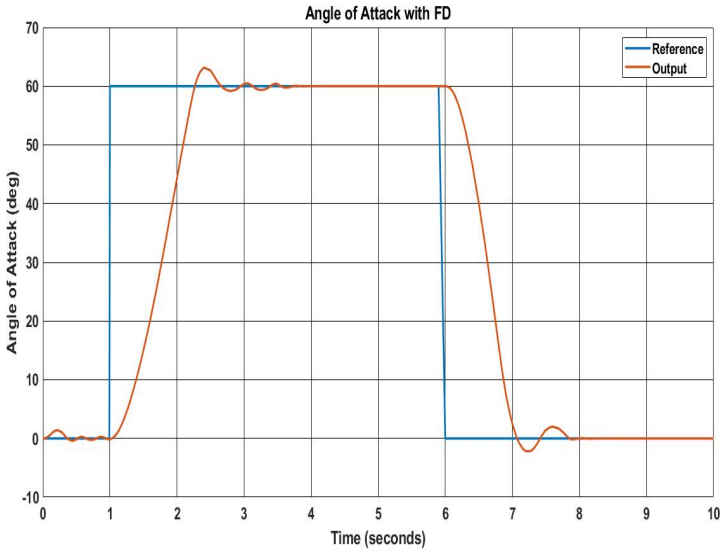

FIGURE 3.5: Closed-Loop α Tracking with FD Module

The corresponding control effort is given in Figure 3.6.

3.5.3 Discussion on Results

The results from closed-loop results indicate that the closed-loop results for MPC are quite efficient with and without the FD module. However, tracking can be improved by including the FD module in parallel with the MPC module as is clear from Figure 3.3 and Figure 3.5. The rise time, settling time, and overshoot for tracking in the presence of the FD module is better than without the FD module.

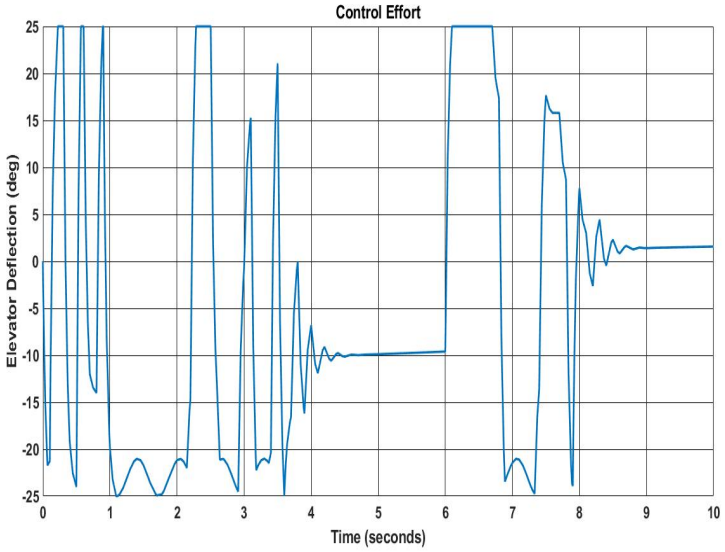

FIGURE 3.6: Elevator Deflection with FD

The rise time, settling time, and percentage overshoot for using MPC without fault detection module is given as

$$\begin{aligned} t_r &= 1.30s \\ t_s &= 3.00s \\ M_p &= 8.3\% \end{aligned}$$

These parameters for MPC with fault detection module are given as

$$\begin{aligned} t_r &= 1.20s \\ t_s &= 2.50s \\ M_p &= 3.3\% \end{aligned}$$

3.6 Conclusion

In modern fighter aircraft, the main emphasis of the designer is on increasing maneuverability. The fast maneuvers suffer from heavy aerodynamic forces on their control surfaces. An MPC controller caters to the input constraints. However, it can't cater to faults produced by strong aerodynamic forces that occurred during the maneuvers. The faults can be catered for using reconfigurable MPC and FD modules working in parallel. The closed-loop simulation results show that the maneuver can be improved by combining the FD module with reconfigurable

MPC. FD module intimate about the occurrence of the fault to the MPC and MPC reconfigures its setting according to the fault scenario. This technique has an advantage over other fault detection techniques that it considers time-varying faults, while other techniques consider time-invariant faults.

The main limitation of the proposed method is that the FD module works in parallel MPC, and adjusts the controller in each sample time. Thus, it increases the computational complexity of the controller. The other limitation is that different variables in the FD module have been selected on a trial and error approach which is time-consuming and non-optimal. Also, this method is tested in simulations only, physical validation of the proposed algorithm is beyond the scope of the research.

Future work includes the finding of different variables in the FD module in a systematic way. FD algorithm can also be improved according to time-varying faults. MPC itself is computationally complex. The working of MPC in parallel with FD makes the simulations slower. So, different techniques can be employed to make the MPC work faster. Physical validation can also contribute to the validation and improvement of the proposed algorithm.

Chapter 4

LQR-based Optimal Blocking Strategy

4.1 Introduction

MPC is quite efficient in reference tracking and constraint handling but inefficient from a computational point of view. The computational burden limits the applications of the MPC for slow dynamic systems with a few number of states. There are many reasons for high computations in MPC, which are:

- Prediction matrices are of higher order as compared to simple state-space matrices. Iteratively solving these matrices makes the computations heavy. Matrix size increases with the increasing perdition horizon.
- Long prediction and control horizons also make the computation complex. A long control horizon means more decision variables in the optimization process, which takes more time for calculation.
- Incorporation of constraints also makes the computation heavy.
- The use of fast sampling time also contributes to the computation load. Since MPC calculates the optimal control input in each sampling interval, hence use of a fast sampling interval leads to a greater computation burden.
- Computation is also dependent upon the optimization technique used.

TABLE 4.1: Blocking Inputs

u_0	u_1	u_2	u_3	u_4	u_5	u_6	u_7	u_8	u_9
$\hat{u}_0$	$\hat{u}_1$			$\hat{u}_2$			$\hat{u}_3$		

Over the years, scientists have been working on reducing the computational complexity of the MPC in both offline as well as online manner. These ways include changing the optimization strategy as well as changing the MPC model. One research area for enhancing computation speed is blocking MPC. The blocking MPC is an input parametrization technique that reduces the number of decision variables by fixing them or their derivatives for a few sampling intervals, thus making blocks as shown in Table 4.1. The main advantage of using this scheme is that the physical meaning of the parametrized input remains the same as the original input. Thus hardware implementation also remains straightforward.

In [72] different blocking schemes with their pros and cons have been reviewed. These blocking schemes include Input Blocking (IB), Offset Blocking (OB), Delta Blocking (DIB, DOB), and Moving Window Blocking (MWB). IB keeps the value of the input to be constant for a few time steps, but it is not possible to prove recursive feasibility and stability. OB fixes the offset to the feedback control for a certain number of times steps. This scheme provides good closed-loop performance and recursive feasibility, but it doesn't guarantee closed-loop stability. DB fixes the value of the difference between two consecutive inputs $\boldsymbol{u}_k - \boldsymbol{u}_{k+1}$ or offset from a small number for a few steps. This scheme is also unable to provide closed-loop stability. MWB is time-dependent whereas the blocking structure is time-varying. This scheme, besides improving closed-loop performance, provides recursive feasibility and stability. Optimal blocking positions (called blocking structures) must be found online in each sampling interval for optimal performance. Blocking positions are integer values and finding optimal blocking structure along with a regular MPC problem makes it a mixed-integer problem (MIP). MIP is non-convex, hence there are no global minima for this problem. Thus solving MIP makes the problem computationally more complex. So, there's a need to find schemes to derive optimal blocking structures without solving MIP.

While considering optimal time-dependent blocking structure, some schemes convert MIP to a simple floating-point problem by finding the blocking structure separately. In [73] variable horizon model predictive control scheme has been proposed in which control horizon length can also be found by solving an optimization problem. To reduce the computational burden for long horizon length time-varying blocking MPC has been applied. In this scheme, every admissible blocking structure has been solved using an optimization problem in parallel. The proposed method provides finite time completion and recursive feasibility despite disturbances but it is unable to reduce the computational complexity of MPC because of parallel processing.

In [74] move blocking MPC has been modified by finding the minimum horizon length and the number of blocks. In this MIP has been solved explicitly for finding the optimal control horizon and number of blocks used. Then in the next step, a blocking structure with the maximum region of attraction (ROA) has been chosen as the optimal blocking structure. However, the problem with this scheme is that it uses a time-invariant blocking structure. It's quite probable that the blocking structure that is optimal in a one-time step may not be optimal in another time step. Hence this scheme can't provide the optimal solution.

In [75] time-varying blocking structure has been computed by using an algorithm (based on mean square error). However, this method is unable to provide guaranteed stability. The performance of the algorithm also depends upon an initial guess about the input.

In [76], a semi-explicit approach has been proposed for finding the optimal time-varying blocking structure. The proposed scheme first finds the critical region for every admissible blocking structure explicitly. Then searches for the critical region online according to the current state. This scheme reduces computational complexity while preserving recursive feasibility and stability. However extra computation appears in searching for the critical region online. This complexity grows with increasing horizon length. To reduce computational complexity and to preserve performance there is a need for the design and application of a scheme that computes time-varying optimal blocking structure based on current states online.

The designed scheme should also provide closed-loop stability in the presence of bounded disturbances.

A different blocking scheme is proposed in this chapter which benefits from the optimal behavior of LQR and makes the blocking structure accordingly. The proposed scheme solves the infinite horizon LQR problem offline to get the optimal inputs over the entire simulation period. Based on these inputs it then finds the difference between consecutive inputs and compares it with a threshold value. If the difference between two consecutive values is less than the threshold value then it blocks the input, otherwise doesn't block it. The number of blocked variables doesn't remain fixed over the horizon but changes according to the behavior of the input, hence providing a time-varying blocking strategy.

MPC has many applications in the control field, and it has already been applied to many applications. This chapter focuses on the application of the MPC for fast dynamic systems. In [77], MPC has been applied to F-16 aircraft for performing high Angle of Attack Maneuvers in the presence of external disturbances and uncertainties. However, the computational complexity limits the real-time application of the MPC for these applications. This chapter focuses on the application of the proposed scheme for fast dynamic systems i.e., aircraft.

The rest of the chapter is as follows. The basic formation of MPC and blocking MPC is given in section 4.2. An LQR-based optimal blocking strategy for computationally efficient MPC is given in section 4.3. The proposed control scheme is applied to a Cessna aircraft in section 4.4. The concluding remarks and future directions are given in section 4.5.

4.2 Blocking MPC Problem

Model Predictive Control (MPC) is a model-dependent controller. Based on the model of the system, MPC can be categorized as a linear or non-linear controller. Linear MPC [78] makes predictions of the system by using the discrete linear model

$$\boldsymbol{x}_m(k+1) = A_m\boldsymbol{x}_m(k) + B_m\boldsymbol{u}_m(k) \tag{4.1}$$

$$\boldsymbol{y}_m(k) = C_m\boldsymbol{x}(k) + D_m\boldsymbol{u}(k) \tag{4.2}$$

The idea of blocking MPC is to solve the MPC problem for a reduced number of control inputs and a reduced number of constraints in each iteration. The MPC solves the optimization problem for a fixed horizon in the future for the cost function given

$$J = \min_{\Delta\boldsymbol{u}(i)} \sum_{i=1}^{N-1} (\boldsymbol{e}(i)^T q_d \boldsymbol{e}(i) + \Delta\boldsymbol{u}(i)^T r_d \Delta\boldsymbol{u}(i)) + \boldsymbol{e}(N)^T q_d \boldsymbol{e}(N) \tag{4.3}$$

where $\boldsymbol{e}(i) = \boldsymbol{x}_m(i) - \boldsymbol{x}_{ref}$, and $\boldsymbol{x}_m$, $\boldsymbol{y}_m$ follow equation (4.1). The variables q_d and r_d are the tuning variables for the state and input. These variables are mostly the diagonal variables. The state and input constraints are defined as

$$\begin{aligned} \boldsymbol{x}^{min} \leq \quad & \boldsymbol{x}_m(k) \leq \quad \boldsymbol{x}^{max} \\ \Delta\boldsymbol{u}^{min} \leq \quad & \Delta\boldsymbol{u}_m(k) \leq \quad \Delta\boldsymbol{u}^{max} \end{aligned} \tag{4.4}$$

Where $\boldsymbol{x}^{min}$ and $\Delta\boldsymbol{u}^{min}$ are the lower limits of the states and change in inputs, respectively. While $\boldsymbol{x}^{max}$ and $\Delta\boldsymbol{u}^{max}$ are the upper limits of the same.

The MPC problem solves the optimization problem in equation (4.3) at each sample time to find the optimal values of the control inputs $\boldsymbol{u}_i$. However, in the case of blocking MPC, the control variables are reduced by a blocking matrix T, which fixes the values of the control variable for a few time steps. The transformed input vector is then obtained by

$$\begin{aligned} \Delta\boldsymbol{U} &= \begin{bmatrix} \Delta\boldsymbol{u}(1) & \cdots & \Delta\boldsymbol{u}(N-1) \end{bmatrix}^T \\ &= (T \otimes I_m) \begin{bmatrix} \Delta\hat{\boldsymbol{u}}(1) & \cdots & \Delta\hat{\boldsymbol{u}}(M-1) \end{bmatrix}^T \end{aligned} \tag{4.5}$$

where I_m is the identity matrix of order m, m are the number of inputs of the system. N is the horizon length, M is the reduced length and $M < N$. The sign $\otimes$ is the Kronecker product.

The efficiency of the blocking MPC is dependent on the blocking matrix T. There are various ways in the literature to find the blocking matrix. This research is focused on an optimal time-varying blocking structure based on offline LQR computations.

4.3 LQR-based Optimal Blocking MPC

This section focuses on the optimal way of finding the blocking matrix. Other than making it optimal, the other concern is to make the blocking time-varying (moving window blocking) to guarantee stability. The steps involved in finding the blocking matrix are given:

- Assumption: The desired trajectory is known
- For the system model in equation (4.1), solve the unconstrained LQR problem [79] using the augmented model (4.6)

$$\begin{bmatrix} \boldsymbol{x}_m(k+1) \\ \boldsymbol{e}(k+1) \end{bmatrix} = \begin{bmatrix} A_m & 0 \\ C_m & 1 \end{bmatrix} \begin{bmatrix} \boldsymbol{x}_m(k) \\ \boldsymbol{e}(k) \end{bmatrix} + \begin{bmatrix} A_m \\ 0 \end{bmatrix} \boldsymbol{u}_m(k) + \begin{bmatrix} 0 \\ 1 \end{bmatrix} \boldsymbol{R}_{ref} \tag{4.6}$$

$$\boldsymbol{u} = -K_{LQR} \begin{bmatrix} \boldsymbol{x} \\ \boldsymbol{e} \end{bmatrix}$$

- Find the difference between two consecutive inputs for all period

$$\delta \boldsymbol{u} = \boldsymbol{u}(k) - \boldsymbol{u}(k-1) \qquad k = 1 \longrightarrow \infty$$

- Find the change (in input) vector and the number of blocked variables
  - Define that minimum change e_{th} acceptable for blocking
  - Initialize the number of blocked variables to zero (i.e., $c = 0$)
  - Find the change vector

$$\boldsymbol{e}(k) = \delta \boldsymbol{u}(k) - \delta \boldsymbol{u}(k+1) \qquad k = 1 \longrightarrow t$$

    where t is the total simulation time
  - Increment c by 1 if $|\boldsymbol{e}(k)| \geq \boldsymbol{e}_{th}$
- Initialize the blocking matrix T

$$T = zeros(length(t) + N, c + N)$$

and

$$T(1,1) = 1$$

- Set $d = 1$, for $k = 1 \longrightarrow length(t) - 1$
  - if $|\boldsymbol{e}(k)| \leq \boldsymbol{e}_{th}$
    $$T(k+1, d) = 1$$
  - else $d = d + 1$
    $$T(k+1, d) = 1$$
- Discard zero columns in T

After finding the blocking matrix T, the next step is to update the MPC problem. At every sampling time, take part of T for that horizon only

$$T_1 = T(kk : kk + N - 1, f : f + N - 1)$$

where f is used for switching to the next column if the first entry in the column is zero.

$$columnWithAllZeros = all(T_1 == 0)$$

and

$$T_2 = T_1(:, \sim columnWithAllZeros)$$

and update the MPC problem by

$$\Delta \boldsymbol{u} = T_2 \, \Delta \hat{\boldsymbol{u}}$$

4.4 Numerical Example

The designed scheme is tested at an aircraft longitudinal model for pitch control. The states of the system are

$$\boldsymbol{x}^T = \begin{bmatrix} \alpha & \theta & q \end{bmatrix}$$

TABLE 4.2: Aircraft Limits

Sr	Symbol	Variable Name	Limit
1	θ	Pitch Angle	20°
2	δ_e	Elevator Deflection	15°

where α shows the angle of attack measured in rad, θ is for the pitch angle in rad, and q is used for the pitch rate of the aircraft in rad/s. The control input of the system is elevator deflection δ_e which is used to control the pitching movement of the aircraft. Throttle has not been considered in this model. The linear model of the system at 5000m height and 128.2m/s speed is given [80] as

$$\begin{aligned} A &= \begin{bmatrix} -1.2822 & 0 & 0.98 \\ 0 & 0 & 1 \\ -5.4293 & 0 & -1.8366 \end{bmatrix} \\ B &= \begin{bmatrix} -0.3 \\ 0 \\ -17 \end{bmatrix} \\ C &= \begin{bmatrix} 0 & 1 & 0 \end{bmatrix} \\ D &= 0 \end{aligned} \tag{4.7}$$

The limits on the input and output of the system are given in Table 4.2.

The sampling time is 0.1sec for this case, and the horizon length is chosen as 10. The error threshold e_{th} is chosen as 0.001 here. For now, the error threshold is defined on a trial-and-error approach. However, in the future, this can be found systematically depending on the application. The reference input for pitch angle tracking is 20° from 1sec to 7sec and 0° otherwise.

The designed new scheme is tested against two different blocking schemes, fixed blocking and moving window blocking. The fixed blocking scheme is selected as blocking 3 consecutive inputs. The output response for the fixed blocking scheme is shown in Figure 4.1. The computation time for this scheme is around 1.7sec. The output result shows that the control scheme tracks the desired reference with a good rise time and a little overshoot. The performance parameters for this

method are given as

$$\begin{aligned} t_r &= 0.90s \\ t_s &= 2.10s \\ M_p &= 10\% \end{aligned}$$

Then, a moving window-blocking scheme is tested at the same system. Now the blocking variables are different at each sampling time. However, the pattern is fixed. This pattern is repeated over a window for guaranteed stability but doesn't cater to it if the results are not good. This scheme has the computational advantage over the previous scheme and takes about 1.14 sec. The output response for this scheme is shown in Figure 4.2. The performance parameters for this scheme are given as

$$\begin{aligned} t_r &= 0.80s \\ t_s &= 1.80s \\ M_p &= 7.5\% \end{aligned}$$

The LQR-based optimal control scheme, on the other hand, shows quite a good response in Figure 4.3. The computational time for this scheme is around 1.13sec. The rise time and settling time are quite better than the previous schemes and there's no overshoot in the system. The response is better at the start as well as the end of the maneuver. The performance is given as

$$\begin{aligned} t_r &= 0.70s \\ t_s &= 0.70s \\ M_p &= 0.5\% \end{aligned}$$

A comparison in terms of rise time, settling, and percentage overshoot for different blocking indicates that the proposed control scheme outperforms the others. The proposed control scheme reacts quickly, which is required for fast dynamic systems (indicated in the form of a short rise time). The proposed scheme also gives a shorter settling time, indicating that the system achieves the desired response faster with less oscillation. Similarly, a small overshoot in the proposed control scheme points toward stability and indicates that the system stays within the limits.

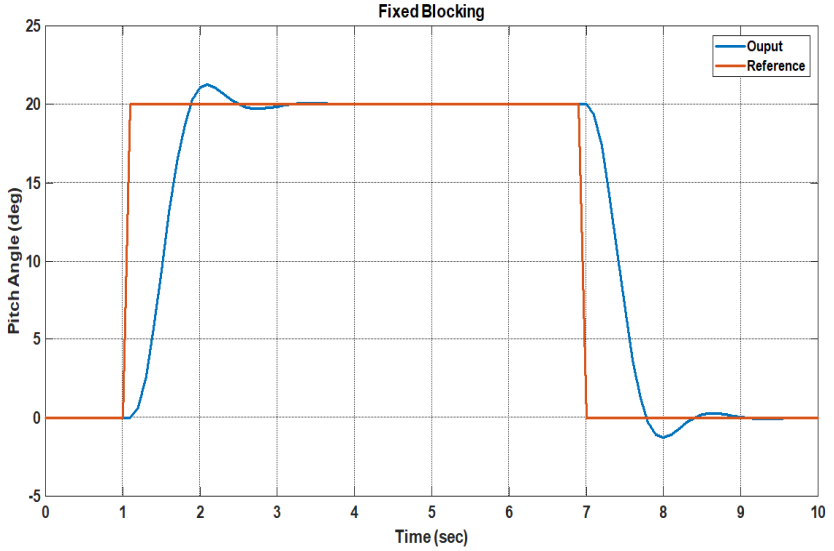

FIGURE 4.1: Fixed Blocking MPC Reference Tracking

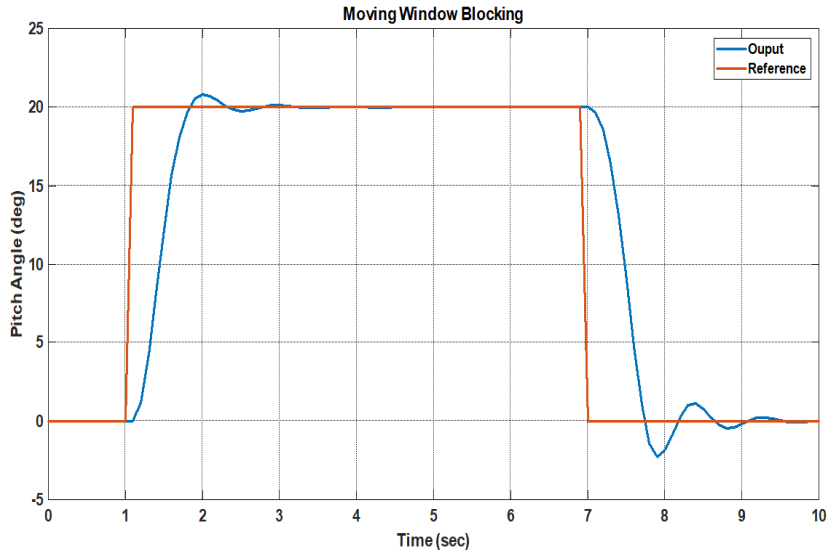

FIGURE 4.2: Moving Window Blocking MPC Reference Tracking

The elevator response for performing the desired maneuver for all three schemes is shown in Figure 4.4-4.6. The elevator's deflection reacts faster in the proposed control scheme as compared to the schemes for better performance. The effect of elevator's deflection can be seen in the fast response with short rise time and settling time and a small overshoot. The performance characteristics of the proposed controller are much better than the available techniques. The input response for all three schemes is within the constraints. The designed controller also performs in a computationally efficient way, while satisfying the constraints

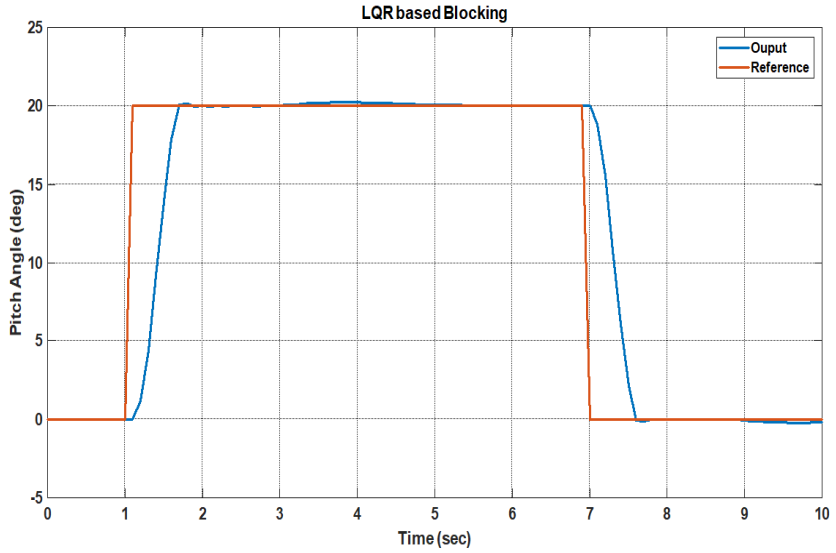

FIGURE 4.3: LQR based Blocking MPC Reference Tracking

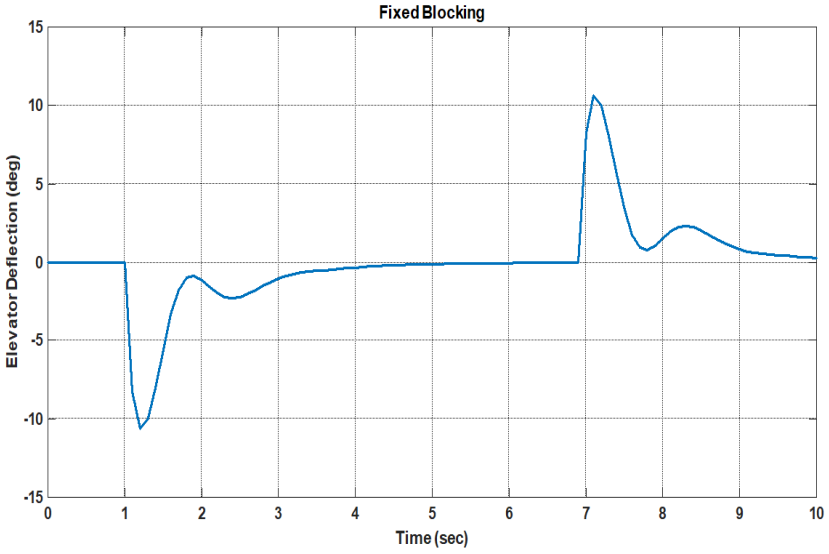

FIGURE 4.4: Fixed Blocking MPC Input Response

4.5 Conclusion

A new optimal blocking scheme for MPC is presented for the computational efficiency of the online controller. The proposed blocking scheme takes advantage of the optimal behavior of the LQR controller and ensures stability by the idea of moving window blocking. The LQR-based optimal blocking scheme is tested on Cessna aircraft and compared against various blocking schemes in the literature. The proposed scheme is better in efficiency and provides good tracking with low

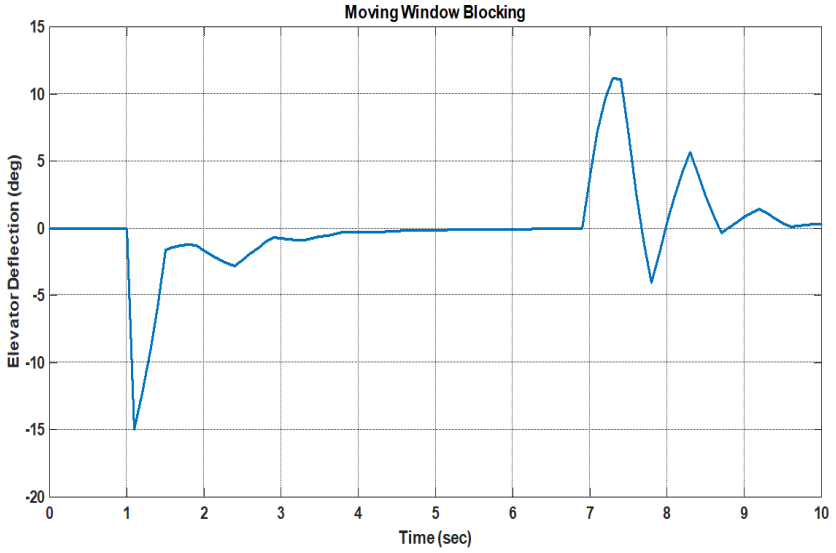

FIGURE 4.5: Moving Window Blocking MPC Input

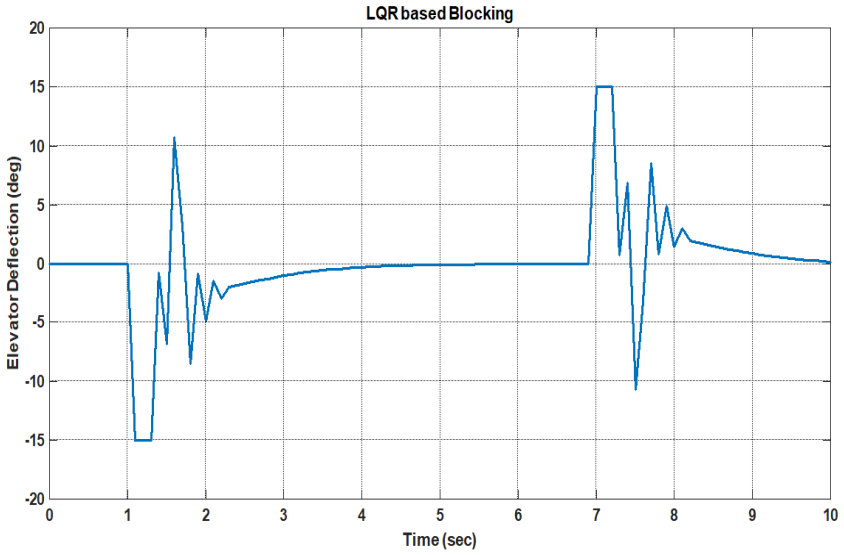

FIGURE 4.6: LQR-based Blocking MPC Input

rise time, settling time, and overshoot. There's no extra computational burden for finding the blocking matrix during online computations, as it can be found offline based on the LQR law.

The proposed scheme is tested against the SISO system. One future direction would be to extend this scheme for the MIMO system and application to a more complex system. Another future direction would be to implement this scheme with other schemes used to reduce the computational burden of the MPC. HIL-based implementation can also be future work for this.

The performance and computational complexity of the proposed algorithm are dependent on the value of the error threshold. A high value of the parameter leads to poor control performance. This will lead to increased oscillations or overshoot, reduced accuracy, and lower robustness to disturbances. However, the computational burden will be reduced as the controller will use a few iterations. A low value contributes to increasing performance, thus providing better tracking, smoother response, and robustness to disturbances. However, this will affect negatively in the form of increasing computations because of requiring more iterations. The choice of this variable is critical in critical applications, e.g., aerospace, robotics, or medical systems. The poor performance can lead to system instability, however, the computational burden will make real-time implementation infeasible. A systematic way of finding this value according to the application can also be done in the future.

Chapter 5

Non-linear MPC

5.1 Introduction

Model predictive control (MPC) is one of the most popular advanced control techniques due to its ability to seamlessly handle constrained multivariable dynamic systems [80]- [81]. The concept also naturally extends to systems best described by nonlinear models. The main downside of MPC is that it requires the real-time solution of a large-scale (dynamic) optimization problem, which has limited its applicability for several decades. Although significant advances have been made in accelerating the solution of MPC problems, it is still challenging to implement in many fast dynamic systems (e.g., fighter aircraft).

Many versions of MPC have been designed to achieve efficient tracking, which can be divided into further categories named Linear, Non-Linear, Explicit, and Robust MPC. A linear MPC approach has been used in [82] for performing high angle of attack (AOA) maneuvers in fighter aircraft, i.e., F-16. This approach performs well for linear systems but is inefficient for nonlinear systems. Another drawback of linear MPC is that it needs the linearized model of the system at each sampling interval, and based on that model, it solves the MPC problem. For the fast dynamic systems e.g., fighter aircraft, the sampling time requirement is in ms, which needs a huge number of linearized models for the whole flight envelope. So it's almost impossible to save this number of linearized models offline or to linearize the system model online along with solving MPC at this sampling interval.

The other way is to use Non-linear MPC [83] or robust MPC [84]-[85], which gives better performance because these techniques can cater to the nonlinearities in the system by design. However, non-linear MPC is unable to provide perfect tracking for some applications, i.e., gives offset error. The offset error has been catered to in different ways in different control schemes. The PID controller applies an integrator to remove offset, while LQR augments the model with output error to reduce it as much as possible. There are multiple methods in the literature for achieving offset-free tracking in MPC. One of them is Dynamic Matrix Control, in which the correction term is achieved by finding the difference between actual vs. predicted output [86]. However, this scheme does not apply to closed-loop systems. There are several different approaches for offset-free tracking in linear MPC, which usually use disturbance or observer models [87]-[88]. Similar types of approaches exist in the literature for the velocity form of the linear system [89]-[90]. These systems usually cater to the difference between the current and previous state/input instead of the simple state/input. These linear concepts are then transformed into non-linear MPC in [91]. This method controls non-linear models by using non-linear MPC and a non-linear estimator. In this, the system is augmented with disturbance, and a secondary reference is achieved by solving a non-linear problem at steady-state along with the normal MPC problem. So, the tracking is much better as compared to the simple MPC model. However, using nonlinearities in the model and controller design increases the computational complexity.

Offset-free NMPC involves augmenting the system with the disturbance. This disturbance can be usually estimated by finding the difference between the actual output as well as the estimated output. So, for using this model, a good estimator is required. For state estimation, the Kalman filter provides optimal estimation in the presence of model uncertainties and external noise [[92]]. The Kalman filter uses several measurements instead of one for estimation and provides the best estimation. Some non-linear versions of the Kalman filter are also available in the literature, which involves the Extended Kalman Filter (EKF) and the Unscented Kalman Filter (UKF). Extended Kalman Filter gives poor results when

the model is highly nonlinear. For this type of system, the UKF provides good approximations [93].

NMPC provides efficient tracking in the presence of nonlinearities and removes steady-state error by using UKF and offset-free MPC. The major disadvantage is this method that it is not suitable for real-time applications involving fast dynamic systems. For example the applications like fighter aircraft F-16, the system has around 13 states and 4 inputs. However, the sampling time requirement is around $1ms$ (depending on the minimum time constant of the elevator) [94]. The computational complexity of NMPC for this type of system may lead to reduced performance. This research focuses on reducing the computational complexity of NMPC by using Neural Network and symbolic approximations.

In [95], a Reinforcement Learning (RL) based MPC has been proposed. This method approximates the MPC problem as well as the cost function and uses this approximation in the online computation of the controller. This method reduces the online computations by carefully approximating the problem and solving only the RL-based function during online computation. Although the online computations can be reduced, the optimal performance depends on the accurate approximation of the MPC controller. Any mismatch in the controller model or external disturbance may affect the controller's performance.

The main idea of this research, which distinguishes it from the available techniques is to find the solution of NMPC offline by solving the problem symbolically instead of numerically, and NN-based approximation for the part that resists symbolic resolution (i.e., lookup tables). With this unique approach, the problem can be transformed into equations with algebraic expressions instead of matrices. Subsequently, leveraging this offline-derived simplified expression, it finds an optimal solution in each iteration online using the current state estimate. A noteworthy addition to this technique is the integration NLP solver and UKF to mitigate the steady-state error and uncertainties due to approximations and external disturbances. NLP solver finds a secondary reference at the steady state. UKF estimates the immeasurable states amidst uncertainties. This way of solving the problem for controller design and application reduces the computational complexity, and

steady-state error while achieving good performance.

In summary, the novel approach proposed in this research reduces the computational challenge of NMPC by combining symbolic and NN-based approximations. This integration of NLP solver and UKF helps reduce steady-state error, resulting in robustness enhancement. The overall result is reduced computation, diminished steady-state error, and improved performance in controller design and application. For symbolic solution and numerical optimization, the CasADi toolbox provides an efficient, as well as a rapid way of implementation [96]. This platform provides a symbolic platform for offline as well as online optimization. It helps solve complex engineering problems and has a wide range of applications in process control, robotics, and aerospace.

CasADi toolbox works in the symbolic environment and solves the optimization problem by considering the system's parameters to be constant. However, if there are state/input-dependent parameters e.g., aerodynamics coefficients in aircraft models, the CasADi toolbox is not applicable. These aerodynamic coefficients are neither constants nor variables with algebraic expressions. The coefficients are states and inputs dependent parameters and are available in lookup table form in the mathematical model of the system. Thus, these parameters can't be solved symbolically by using the CasADi toolbox to find a simplified expression for the controller. This problem is catered to by the application of the Neural Network (NN) function approximation [97]- [98] in this research. The proposed approach uses a feedforward-based NN approach to approximate the function of state/input-dependent parameters, which are accepted by the CasADi toolbox.

A numerical analysis for performing a highly complex maneuver for aircraft helps prove the efficiency of the proposed approach. The fighter aircraft have been tested for many control applications [99] - [79]. However, with the introduction of highly maneuverable 5th and 6th generation aircraft, the need for control design is even more demanding. With increased maneuverability, modern aircraft are highly unstable to perform even complex maneuvers that are impossible for earlier aircraft. This requires improved tracking while maintaining stability in milliseconds. In effect, all the performance parameters for control, i.e., the rise time,

settling time, and overshoot, should be improved. The proposed control approach provides a computationally efficient NMPC implementation for efficient tracking of such high-speed systems while catering to the physical constraints.

This chapter follows the following sequence. Section 5.2 describes the aircraft 6 DOF model, the states and inputs of the aircraft, the parameters of the system, and constraints that need to be followed by the aircraft. Section 5.3 and section 5.4 describe the observability and controllability of the system. Section 5.5 gives the description of the NMPC as well as its offset-free variation. Section 5.6 tells about the proposed approach for a computationally efficient method using an NN-based approach. In section 5.7, the proposed approach is applied to a non-linear fighter aircraft model, i.e., F-16 to verify the controller efficiency in the presence of uncertainties and non-linearities. Section 5.8 gives the stability analysis of the controller. The conclusion is summarized in section 5.9.

5.2 Aircraft Model

The aircraft model is highly non-linear. Since the controller design in this chapter is based on the non-linear model, only the non-linear model will be considered here. Overall, the 6 DOF model of the aircraft consists of 14 states including longitudinal Velocity "$\boldsymbol{u}$", lateral velocity "$\boldsymbol{v}$", vertical velocity "$\boldsymbol{w}$", roll angle "ϕ", pitch angle "θ", yaw angle "ψ", roll rate "p", pitch rate "q", yaw rate "r", north position "P_n", east position "P_e", height "h", and engine power "P_a". However, in fighter aircraft, it is sometimes preferred to use velocity axes instead of body axes. This will result in a change of states $\boldsymbol{u}$, $\boldsymbol{v}$, and $\boldsymbol{w}$ to V_t, α, and β (i.e., Total Velocity, Angle of Attack, and Side-Slip Angle). However, for simplicity, some uncontrollable states are removed from the model i.e., P_n, P_e. The model follows the following system states

$$\boldsymbol{x}^T = [V_t \;\; \alpha \;\; \beta \;\; \phi \;\; \theta \;\; \psi \;\; p \;\; q \;\; r \;\; h \;\; P_a]$$

There are four control inputs. Two of them are for longitudinal control i.e., thrust "th", and elevator deflection "δ_e". The other two inputs are aileron deflection

TABLE 5.1: F-16 Parameters

Symbol	Name	Value
B	Wing Span	$30\ ft$
S	Wing Area	$300\ ft^2$
$\bar{c}$	Mean Aerodynamics Chord	$11.32\ ft$
X_{CGR}	Reference CG in chord length	0.35
H_g	Engine Angular Momentum	$160\ slug - ft^2/s$
α_T	Thrust Vector Angle	$0°$
$\boldsymbol{W}_{GT}$	Weight	$20500\ lb$
M_{xx}	Moment of Inertia about x-axis	$9496\ slug - ft^2$
M_{yy}	Moment of Inertia about y-axis	$55814\ slug - ft^2$
M_{zz}	Moment of Inertia about z-axis	$63100\ slug - ft^2$
M_{xz}	Product of Inertia	$982\ slug - ft^2$

TABLE 5.2: F-16 Constraints

Symbol	Name	Constraint
th	Thrust	$0 \leq th \leq 1$
δ_e	Elevator Deflection	$-25° \leq \delta_e \leq 25°$
δ_a	Aileron Deflection	$-21.5° \leq \delta_a \leq 21.5°$
δ_r	Rudder Deflection	$-30° \leq \delta_r \leq 30°$
V_t	True Velocity	$390 \leq V_t \leq 900\ ft/s$
h	Altitude	$0 \leq h \leq 40000\ ft$
P_a	Engine Power	$0 \leq P_a \leq 100\ \%$

"δ_a", and rudder deflection "δ_r". The overall input vector is as follows:

$$\boldsymbol{u}^T = [th \;\; \delta_e \;\; \delta_a \;\; \delta_r]$$

The model of F-16 is taken from [100]. The system's parameters are as in Table 5.1. There are certain constraints on the inputs and states of the model. These constraints are summarized in Table 5.2.

All other aerodynamics coefficients and the non-linear model of F-16 can be taken from [100], which includes the system's equations and the aircraft's parameters.

5.3 Observability

The nonlinear Aircraft model doesn't contain fully measurable states. There are a few states, e.g., side slip angle, linear velocities, which are not directly measurable. Hence, these unmeasurable states are estimated through a nonlinear observer, e.g., Kalman filter, etc. Since, the 6 DOF model of the aircraft contains longitudinal and lateral motion, so atleast one state in each mode is required for the nonlinear observer. This research is focused on the angle of attack maneuver, so α is chosen as a measurable state in longitudinal motion. In lateral motion, yaw angle ψ is considered as a measurable state because it is directly measurable. Hence, two states α and ψ are considered as measurable states in this research. All other states are estimated through nonlinear observer. For nonlinear observer, a nonlinear state observability test is required.

The observability test can be performed by linearizing the system around the equilibrium and finding the rank of the observability matrix given in the equation

$$O^T = \begin{bmatrix} C & CA & CA^2 & ...CA^{n-1} \end{bmatrix} \tag{5.1}$$

For linear analysis, the system's linear model at equilibrium condition $\boldsymbol{x}0 = [500; 0; 0; 0; 0; 0; 0; 0; 0; 20000; 90]$, $\boldsymbol{u}0 = [0.9; 0; 0; 0]$ is defined in equation 5.2.

The observability matrix derived by using the linear model of the system is shown in the equation 5.3. The rank of the observability matrix is 11, which is the number of states, indicating that it is a full-rank matrix. Thus, the system is locally observable.

In [101], a nonlinear observability test has been defined. For the system of the form

$$\dot{\boldsymbol{x}} = f(\boldsymbol{x}(t), \boldsymbol{u}(t)) \tag{5.4}$$

$$A = \begin{bmatrix} 1.004 & 3.1577 & 0 & 0 & -3.2174 & 0 & 0 & 0.0291 & 0 & -0.0001 & 0.0178 \\ 0 & 0.9416 & 0.0001 & 0 & 0 & 0 & 0 & 0.0951 & 0 & 0 & 0 \\ 0 & 0 & 0.9795 & 0.0064 & 0 & 0 & -0.0001 & 0 & -0.0996 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 & 0.1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0.1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0.1 & 0 & 0 \\ 0 & -0.0587 & -1.2910 & 0 & 0 & 0 & 0.8020 & 0 & 0.0255 & 0 & 0 \\ 0 & 0.1055 & 0 & 0 & 0 & 0 & 0 & 0.9428 & -0.0003 & 0 & 0 \\ 0 & -0.0042 & 0.4576 & 0 & 0 & 0 & 0.0003 & 0.0003 & 0.9748 & 0 & 0 \\ 0 & -50 & 0 & 0 & 50 & 0 & 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0.4734 \end{bmatrix}$$

$$\mathrm{B} = \begin{bmatrix} 0 & 0.0005 & 0 & 0 \\ 0 & -0.0001 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & -0.0385 & 0.007 \\ 0 & -0.0092 & 0 & 0 \\ 0 & 0 & -0.0017 & -0.0032 \\ 0 & 0 & 0 & 0 \\ 112.7531 & 0 & 0 & 0 \end{bmatrix}$$

$$\mathrm{C} = \begin{bmatrix} 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 \end{bmatrix} \tag{5.2}$$

$$O = \begin{bmatrix} 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0.94 & .0001 & 0 & 0 & 0 & 0 & .0951 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & .1 & 0 & 0 \\ 0 & 0.8965 & .0002 & 0 & .0001 & 0 & 0 & .1792 & 0 & 0 & 0 \\ 0 & -.0004 & .0458 & 0 & 0 & 1 & 0 & 0 & .1975 & 0 & 0 \\ -.0001 & -.8629 & .0002 & 0 & .0002 & 0 & 0 & .2543 & -.0001 & 0 & 0 \\ 0 & -.0012 & .1352 & .0003 & 0 & 1 & .0001 & 0 & 0.2879 & 0 & 0 \\ -0.0001 & .8391 & .0002 & 0 & .0004 & 0 & 0 & .3218 & -.0002 & 0 & 0 \\ 0 & -.0024 & .2641 & .0012 & 0 & 1 & .0001 & 0 & 0.3672 & 0 & 0 \\ -0.0001 & .8238 & .0002 & 0 & .0006 & 0 & 0 & .3833 & .0003 & 0 & 0 \\ 0 & -.0038 & .4265 & .0029 & 0 & 1 & .0003 & -.0001 & .4316 & 0 & 0 \\ -0.0001 & .8157 & .0002 & 0 & 0.001 & 0 & 0 & .4397 & -0004 & 0 & 0 \\ 0 & -.0054 & .6149 & .0056 & 0 & 1 & .0006 & -.0004 & .4783 & 0 & 0 \\ -0.0002 & .8140 & 0 & 0 & .0014 & 0 & 0 & .4923 & -.0006 & 0 & 0 \\ 0 & -.0072 & .8204 & .0096 & 0 & 1 & .0011 & -.0008 & .5050 & 0 & 0 \\ -0.0002 & .8179 & -.0002 & 0 & .002 & 0 & 0 & .5416 & -.0007 & 0 & 0 \\ 0 & -.0091 & 1.0332 & .0148 & 0 & 1 & .0019 & -.0013 & .5105 & 0 & 0 \\ -0.0002 & .8266 & -.0004 & 0 & .0026 & 0 & 0 & .5886 & .-0008 & 0 & 0 \\ 0 & -0.0109 & 1.2432 & .0215 & 0 & 1 & .0031 & -.0019 & .4948 & 0 & 0 \\ -0.0003 & .8396 & -.0007 & 0 & .0034 & 0 & 0 & .6338 & -.0009 & 0 & 53 \\ 0 & -.0128 & -1.4401 & .0295 & 0 & 1 & .0047 & -.0027 & .4858 & 0 & 0 \end{bmatrix} \tag{5.3}$$

where f is the nonlinear function of the state $\boldsymbol{x}$, and input $\boldsymbol{u}$ at time t (here nonlinear 6DOF aircraft model defined in Chapter 1). The system is fully observable if nonlinear observability matrix O has full rank. The observability matrix O is defined as

$$O^T(x) = \nabla \begin{bmatrix} L_f^0 \boldsymbol{h} & L_f^1 \boldsymbol{h} & ... & L_f^{n-1} \boldsymbol{h} \end{bmatrix} \tag{5.5}$$

The operator $L_f^0 \boldsymbol{h}$ is the Lie derivative of $\boldsymbol{h}$ in the direction of f, and 0 denotes the $0th$ Lie derivative. The operator $\boldsymbol{\nabla}$ is the gradient of the matrix with respect to the state $\boldsymbol{x}$. Lie derivative represents the directional derivative of the function $\boldsymbol{h}$ in the direction of the function f, and can be calculated in general as

$$L_f \boldsymbol{h} = \boldsymbol{\nabla h(x)} \tag{5.6}$$

where

$$\boldsymbol{\nabla h} = \frac{\partial \boldsymbol{h}}{\partial \boldsymbol{x}} \tag{5.7}$$

which is a row vector of the derivatives of h with respect to each state, with each element of the column is represented by $\frac{\partial \boldsymbol{h}}{\partial \boldsymbol{x}_i}$. In this way, Lie derivative in each row of the observability matrix O can be calculated as given in the equations

$$L_f^0 \boldsymbol{h} = \boldsymbol{h(x)} \tag{5.8}$$

$$L_f^1 \boldsymbol{h} = \frac{\partial L_f^0 \boldsymbol{h}}{\partial \boldsymbol{x}} f \tag{5.9}$$

which further implies in nth lie derivative as

$$L_f^n \boldsymbol{h} = \frac{\partial L_f^{n-1} \boldsymbol{h}}{\partial \boldsymbol{x}} f \tag{5.10}$$

The observability condition is then defined as

$$\text{rank}(O(\boldsymbol{x}_0)) = n \tag{5.11}$$

The simulation is performed in MATLAB using the CasADi platform. The observability matrix obtained is given in the equation 5.12.

The MATLAB simulation result in Figure 5.1 show that the system is fully observable.

$$O = \begin{bmatrix} 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 2 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 3 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 20 & 0 & 0 & 0 & 0 & 0 & 2 & 0 & 0 & 0 \\ 0 & 0 & 2 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 108 & 0 & 0 & 0 & 0 & 0 & 13 & 0 & 0 & 0 \\ 0 & 0 & 21 & 0 & 0 & 1 & 0 & 0 & 3 & 0 & 0 \\ 0 & -416 & 0 & 0 & 0 & 0 & 0 & 81 & 0 & 0 & -1 \\ 0 & 1 & -74 & 1 & 0 & 1 & 0 & 0 & 15 & 0 & 0 \\ 0 & 7814 & 0 & 0 & -1 & 0 & 0 & -360 & 0 & 0 & 53 \\ 0 & 1 & -111 & 2 & 0 & 1 & 0 & 0 & 56 & 0 & 0 \end{bmatrix} \tag{5.12}$$

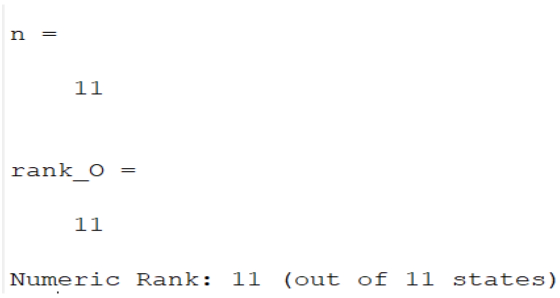

FIGURE 5.1: Observability Test Result in Matlab

5.4 Controllability

The controllability test can be performed by using the linear model of the system around the equilibrium point in equation 5.2. The controllability matrix given as

$$Cr = \begin{bmatrix} B & AB & A^2B & ... & A^{n-1}B \end{bmatrix} \tag{5.13}$$

The system is locally controllable if the rank of the matrix Cr is equal to the number of states.

$$\text{rank}(Cr) = n \tag{5.14}$$

The controllability matrix for the linear system in equation 5.2 is calculated in Matlab by using the equation 5.13. The rank of the controllability matrix is 11 which shows that the system is locally controllable.

5.5 Output Feedback Non-linear MPC Strategy

NMPC uses a non-linear prediction model (instead of a simple model) to find the optimal input required to achieve a certain reference based on the current input and state estimate. It does this on each sampling interval. So, most of the disturbances can be catered to by this repetition. Hence, this controller is better than other controllers regarding performance.

Over the years, a variety of different models of MPC have been designed. These models include linear, non-linear, explicit, and robust MPC. Among them, non-linear MPC is the most suitable choice for non-linear systems. So, for a prediction horizon and control horizon to be N, the general non-linear MPC problem is

$$\min_{\boldsymbol{U}} \; l = \; \boldsymbol{E}^T Q \boldsymbol{E} + \boldsymbol{U}^T R \boldsymbol{U} \tag{5.15}$$

where

$$\boldsymbol{E}^T = \begin{bmatrix} \boldsymbol{e}(0)^T & \boldsymbol{e}(1)^T & \boldsymbol{e}(2)^T & \dots & \boldsymbol{e}(N)^T \end{bmatrix} \tag{5.16}$$

$$\boldsymbol{e}(k) = \boldsymbol{x}(k) - \boldsymbol{R}_{ref} \tag{5.17}$$

$$\boldsymbol{X}^T = \begin{bmatrix} \boldsymbol{x}(0)^T & \boldsymbol{x}(1)^T & \boldsymbol{x}(2)^T & \dots & \boldsymbol{x}(N)^T \end{bmatrix} \tag{5.18}$$

$$\boldsymbol{U}^T = \begin{bmatrix} \boldsymbol{u}(1)^T & \boldsymbol{u}(2)^T & \dots & \boldsymbol{u}(N-1)^T \end{bmatrix} \tag{5.19}$$

$$\boldsymbol{x}(k+1) = f(\boldsymbol{x}(k), \boldsymbol{u}(k)) \tag{5.20}$$

$$\boldsymbol{y}(k) = g(\boldsymbol{x}(k), \boldsymbol{u}(k)) \tag{5.21}$$

$$\boldsymbol{x}^{min} \leq \boldsymbol{x}(k) \leq \boldsymbol{x}^{max} \tag{5.22}$$

$$\boldsymbol{u}^{min} \leq \boldsymbol{u}(k) \leq \boldsymbol{u}^{max} \tag{5.23}$$

$$Q = \text{diag}(q) \tag{5.24}$$

$$R = \text{diag}(r) \tag{5.25}$$

$\boldsymbol{x}_{ref}$ is the reference input. $\boldsymbol{x}^{min}, \boldsymbol{x}^{max}, \boldsymbol{u}^{min}$, and $\boldsymbol{u}^{max}$ are the upper and lower limits of state $\boldsymbol{x}$ and input $\boldsymbol{u}$. Q and R are of the order of $N \times n$ and $N \times l$. These are the tuning matrices for state and input, respectively. The MPC solves the optimization problem by using the prediction model of the system over a current state estimate. It finds the optimal input required to reach a reference for a prediction horizon of "N" and applies only the first of the inputs. After this, it solves the optimization problem again at the next sampling interval and repeats this process until the end of the process. This way, optimizing each sampling interval makes it more efficient and caters to the disturbances.

MPC is not always accurate. Since MPC is model-dependent, any uncertainties in the system and disturbances may give a steady-state error. One way is to put an integrator in the forward path of the controller. Another way is to use offset-free MPC instead of using another controller in a single system.

The nominal MPC solves the optimization problems efficiently and gives quite good performance. However, in some applications, it's unable to provide perfect tracking. In some applications, there is an offset error, which is caused by plant model mismatches and external disturbances. Ignoring these disturbances and uncertainties while designing the controller may lead to drastic results. The offset-free MPC solves this issue.

Offset-free MPC is a modification of the general MPC problem. This model takes account of the system uncertainties and disturbances based on the non-linear estimation of the states. Then it solves another optimization problem to find the secondary reference state and input, which is

$$\min_{\boldsymbol{x}_{ss},\boldsymbol{u}_{ss}} \; l_{ss} = \boldsymbol{e}_{ss}^T Q_{ss} \boldsymbol{e}_{ss} + \boldsymbol{u}_{ss}^T R_{ss} \boldsymbol{u}_{ss} \tag{5.26}$$

where

$$\boldsymbol{e}_{ss} = \boldsymbol{x}_{ss} - \boldsymbol{R}_{ref} \tag{5.27}$$

$$\boldsymbol{x}_{ss} = f(\boldsymbol{x}_{ss}, \boldsymbol{u}, \hat{\boldsymbol{w}}) \tag{5.28}$$

$$\hat{\boldsymbol{w}} = \hat{\boldsymbol{x}} - \boldsymbol{x} \tag{5.29}$$

$\boldsymbol{x}_{ss}$ is the secondary reference state required to be tracked at steady-state, $\boldsymbol{u}_{ss}$ is the steady-state input required to cater for the disturbances, $\hat{w}$ is the estimated disturbance, $\hat{\boldsymbol{x}}$ is the estimated state. The initial and final states will be "$\boldsymbol{R}_{ref} + \hat{\boldsymbol{w}}$" and "$\boldsymbol{R}_{ref}$" respectively. Now, the MPC problem will be to track the new reference state $\boldsymbol{x}_{ss}$ instead of the original reference $\boldsymbol{R}_{ref}$, and the MPC problem will take the form

$$\min_{\boldsymbol{U}_e} \; l_e = \boldsymbol{E}^T Q_e \boldsymbol{E} + \boldsymbol{U}_e^T R_e \boldsymbol{U}_e \tag{5.30}$$

where

$$\boldsymbol{E}^T = \begin{bmatrix} \boldsymbol{e}(0)^T & \boldsymbol{e}(1)^T & \boldsymbol{e}(2)^T & \dots & \boldsymbol{e}(N)^T \end{bmatrix} \tag{5.31}$$

$$\boldsymbol{e}(k) = \boldsymbol{x}(k) - \boldsymbol{x}_{ss} \tag{5.32}$$

$$\boldsymbol{U}_e^T = \begin{bmatrix} \boldsymbol{u}_e(0)^T & \boldsymbol{u}_e(1)^T & \boldsymbol{u}_e(2)^T & \dots & \boldsymbol{u}_e(N-1)^T \end{bmatrix} \tag{5.33}$$

$$\boldsymbol{u}_e(k) = \boldsymbol{u}(k) - \boldsymbol{u}_{ss} \tag{5.34}$$

$$\boldsymbol{x}(k+1) = f(\boldsymbol{x}(k), \boldsymbol{u}(k)) \tag{5.35}$$

$$\boldsymbol{y}(k) = g(\boldsymbol{x}(k), \boldsymbol{u}(k)) \tag{5.36}$$

$$\hat{\boldsymbol{x}}(k+1) = f(\hat{\boldsymbol{x}}(k), \boldsymbol{u}(k), \hat{\boldsymbol{w}}) \tag{5.37}$$

$$\hat{\boldsymbol{w}} = \hat{\boldsymbol{x}} - \boldsymbol{x} \tag{5.38}$$

$$\boldsymbol{x}^{min} \ \leq \ \boldsymbol{x}(k) \ \leq \ \boldsymbol{x}^{max} \tag{5.39}$$

$$\boldsymbol{u}^{min} \ \leq \ \boldsymbol{u}_e(k) \ \leq \ \boldsymbol{u}^{max} \tag{5.40}$$

$$Q_e = \text{diag}(q_e) \tag{5.41}$$

$$R_e = \text{diag}(r_e) \tag{5.42}$$

Now, the controller solves the optimization problem on each sampling interval to find the secondary reference state and input required to cater to that state. This secondary input will play a part as a reference input to the general MPC problem. The most important part here is the disturbance estimate, which can be estimated by $\hat{\boldsymbol{w}} = \hat{\boldsymbol{x}} - \boldsymbol{x}$.

Another way of catering to uncertainties is the use of a non-linear state estimator. The Kalman filter is known to be an optimal estimator. Kalman filter, which is also known as Linear Quadratic Estimator (LQE) was initially designed by Rudolf E. Kalman and gained popularity even in its early days of design. The main

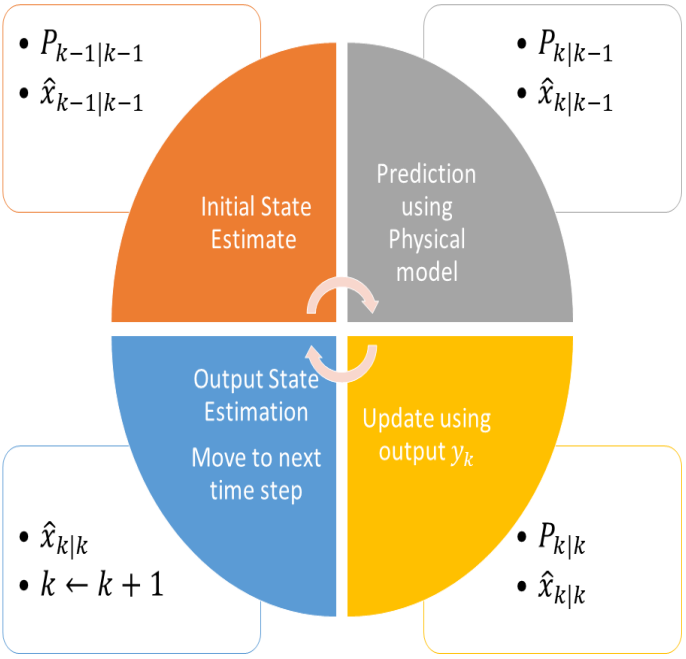

FIGURE 5.2: Kalman Filter State Estimation Process

reason for its popularity is that it provides a more accurate estimate of the states despite noisy data and uncertainties in the system model. Kalman filter uses a series of state values instead of a single measurement, hence providing a better approximation. There are mainly two phases of Kalman filter estimation, namely prediction and update. In the prediction phase, the current states are estimated, which includes uncertainties. In the update phase, the system's states are updated according to noisy measurements, and this process is repeated. The process of the Kalman filter can be better understood by Figure 5.2.

This technique was further extended for non-linear systems. The updated forms of the algorithm were named EKF and UKF. In EKF, the function f for state estimation and function h for output measurement need not be linear. Non-linear models can be used to estimate the next states. But EKF finds the Jacobian in each iteration for a current state estimate, thus linearizing the model in each iteration. This linearization gives an error that propagates in each iteration. In addition, it's really hard to find Jacobian analytically as well as numerically for complex systems. This problem was further solved by the UKF.

UKF finds sigma points around the mean by using Unscented transformation (UT) instead of funding Jacobian. These sigma points are then used to find the new

covariance and mean by using the non-linear model. The accuracy of the filter depends on the statistics of UT and the choice of sigma points. UKF has many applications in aerospace, underwater vehicles, and other navigation applications. In offset-free MPC, there is a need to estimate the system's state in each iteration. Based on the pros and cons of the different estimating schemes, UKF best suits this application [93]. The aircraft model is a highly non-linear function operating in a highly uncertain environment, and controller requirements are changing in milliseconds. This application needs an accurate estimate because any inaccuracy may lead to disastrous effects.

5.6 Computationally Efficient NMPC using Neural Network

Implementation of the NMPC is quite challenging because of the high computational requirements and limited hardware resources. The applications of MPC are mostly restricted to the slow dynamic system i.e., process industry, etc. However, for fast dynamic systems i.e., fighter aircraft, it's still a challenge (F-16 needs a sampling time of about $1ms$ with 14 states and 4 inputs). Other than the high dimensional model, the aircraft model has constraints on the states and inputs. Online solution of NMPC involves solving high dimensional matrices and then optimization of the solution for non-linear systems under given constraints. The online solution may lead to delayed controller response or no solution in the worst-case scenario. The ultimate effect would be disastrous, and this is the reason for the use of NMPC for fast dynamic systems.

This chapter proposes a way to reduce online computations as much as possible by solving the aircraft control problem symbolically (offline) and then optimizing the solution online in each iteration. This is done by using two different tools named CasADi and NN.

CASADI [96] is a symbolic tool that works with MATLAB, Python, and other applications to enhance computational efficiency. It simplifies the problem in symbolic expression and solves the expression or finds the derivatives by using AD

(Algorithm Differentiation). This is done in CasADi's virtual machine or by using CasADi's generated C code. CasADi is mainly inspired by MATLAB (Matrix expressions). It treats all matrices as sparse matrices or column matrices and solves them symbolically once at the start instead of solving them numerically in each iteration. CasADi provides support to modeling in mathematical optimization, including linear, quadratic as well as non-linear optimization. The non-linear optimization in CasADi uses objective and constraint functions that are dependent on optimization variables and constant known parameters. The non-linear solvers supported by CasADi are IPOPT, BONMIN, KNITRO, WORHP, and SNOPT. This toolbox has gained popularity in educational as well as industrial applications since its start in 2011. In the educational sector, it is used to teach optimization tools and to test new algorithms and software. In industry it's being used in the energy sector, automotive industry, process industry, robotics, and assorted applications. The reason for its popularity is it's easy to design and compatible with the hardware.

In today's world, function approximation through NN has attracted most researchers. These approximations proved to be very accurate in most of the applications. One such approximator is the Feed-forward Neural Network (FNN), which is the universal approximator. The function approximations don't need any compact formula for approximations. They just need the data from input to output in the form of $(\boldsymbol{x}, f(\boldsymbol{x}))$. So for an input "$\boldsymbol{x}$" of vector length "N" and output $\boldsymbol{y}$ of the same length as $\boldsymbol{x}$, the function approximation is to find

$$\boldsymbol{y} = f(\boldsymbol{x})$$

A feedforward neural network is used here in this research for approximating the aerodynamic coefficients of the aircraft. The limits of the inputs are obtained by the state's constraints Table 5.2 and outputs are available in the form of interpolation for each aerodynamic coefficient. So, the data set for the function can be easily calculated. The whole aircraft model can also be approximated by using NN, but more approximations lead to more uncertainty. To reduce uncertainty, only necessary parameters have been approximated. Then the data is scaled to

avoid the unstable behavior of the function. In this research, multiple coefficients need to be approximated. All of these coefficients have different numbers of inputs and outputs. By hit and trial, a different number of layers and a different number of nodes have been tuned to get the optimal result. A network training function "translm" is used for the backpropagation algorithm, which provides the updated weights and bias by using the Levenberg-Marquardt optimization. The hyperbolic tangent sigmoid function introduces the non-linearity.

After choosing the network structure and activation function, the neural network can be trained by reducing the mean square error between the estimated value and actual value as

$$MSE = \frac{1}{k}\sum_{n=1}^{k}(C_{out} - \hat{C}_{out})^2 \tag{5.43}$$

A good fit of the function will be if there's no error between the approximated value and the actual value. The function can then be approximated by

$$z_1 = 2(x_{in} - data_{min})./(data_{max} - data_{min}) - 1 \tag{5.44}$$

for all layers $i = 1\ to\ L$

$$z_i = tansig(W(i+1)\ z_i + b(i+1)) \tag{5.45}$$

The variables "W" and "b" are tuning parameters called weight and bias respectively. These parameters have been tuned for the best approximation of the function. Thus

$$y_{out} = W(L+1)\ z_i + b(L+1) \tag{5.46}$$

The approximated function would be

$$f = (y_{out} + 1)/2 \times (t_{max} - t_{min}) + t_{min} \tag{5.47}$$

A complete process for the controller implementation involving NN and symbolic approximation, integration of NLP solver, and utilization of UKF can be described

in the following steps:

1. **Problem Formulation**: The 1st step involves defining the system's dynamics, including constraints and control problems with control objectives.

2. **Symbolic Representation**: The system dynamics and control problem is then transformed into a symbolic representation by using CasADi or similar tools.

3. **Lookup Table Integration Issue**: The next step involves identifying the system's parameters (e.g., lookup tables for aerodynamics coefficients) that can't be integrated into a symbolic representation.

4. **NN Approximation**: The lookup tables and other parameters identified in the previous step are then approximated through Neural Network.

5. **Integration with Symbolic Framework**: The trained neural network is then incorporated into symbolic representation, thus replacing lookup tables with symbolic approximation.

6. **Offline Optimization**: This step involves the utilization of the symbolic representation for the offline solution of the NMPC problem, thus converting the problem from matrices to algebraic expressions.

7. **NLP Solver Integration**: An NLP solver then finds the secondary reference trajectory for mitigating the steady-state error.

8. **Online Computation**: An optimal control sequence is then computed based on the offline computed cost function and feedback control strategy by using the current state estimate. The system's states are estimated by utilizing the UKF, incorporating uncertainties and disturbances.

9. **Application of Control**: From the computed control sequence, 1st of them is applied on the system that need to be controlled. The system's states are continuously estimated by utilizing UKF.

10. **Iterative Process**: The online process is repeated at each sampling interval, and the control sequence is optimized based on the updated states and reference trajectory.

11. **Performance Evaluation and Tuning**: The controller performance is evaluated by using simulations or real-time experiments. The tuning parameter and NN architecture can be updated for better performance and stability.

12. **Deployment and Monitoring**: The designed controller with optimal tuning parameters can then be deployed in the actual environment.

The effective control of complex dynamic systems can be achieved by following the above steps, thus reducing the computational complexity and removing steady-state error. The complete structure of the process is shown in Figure 5.3.

5.7 Numerical Example

The aircraft control problem is tested in MATLAB within the CasADi framework. The focus of the simulation is to perform a high angle of attack maneuver, so AOA tracking is performed in this simulation. The aim is to track AOA as close as possible to the maximum limit of the model. As compared to other control problems of fighter aircraft, a single controller is used to control the aircraft. The controller is applied in a closed-loop formation. The closed-loop block diagram of the system is shown in Figure 5.4.

In Figure 5.4, $\boldsymbol{x}_{ref}$ is the reference input of the system, which in our case is the angle of attack (AOA). The second block named "NLP" is the Non-linear problem solver, which gets the information about the reference input, currently estimated states, and states calculated through a mathematical model. It then processes information to get a second reference input required to be tracked in the presence of uncertainties and disturbances. "NN" is the Neural Network block that approximates the aerodynamic coefficients from the lookup table of the mathematical model, which is then used to find the function value. These aerodynamic

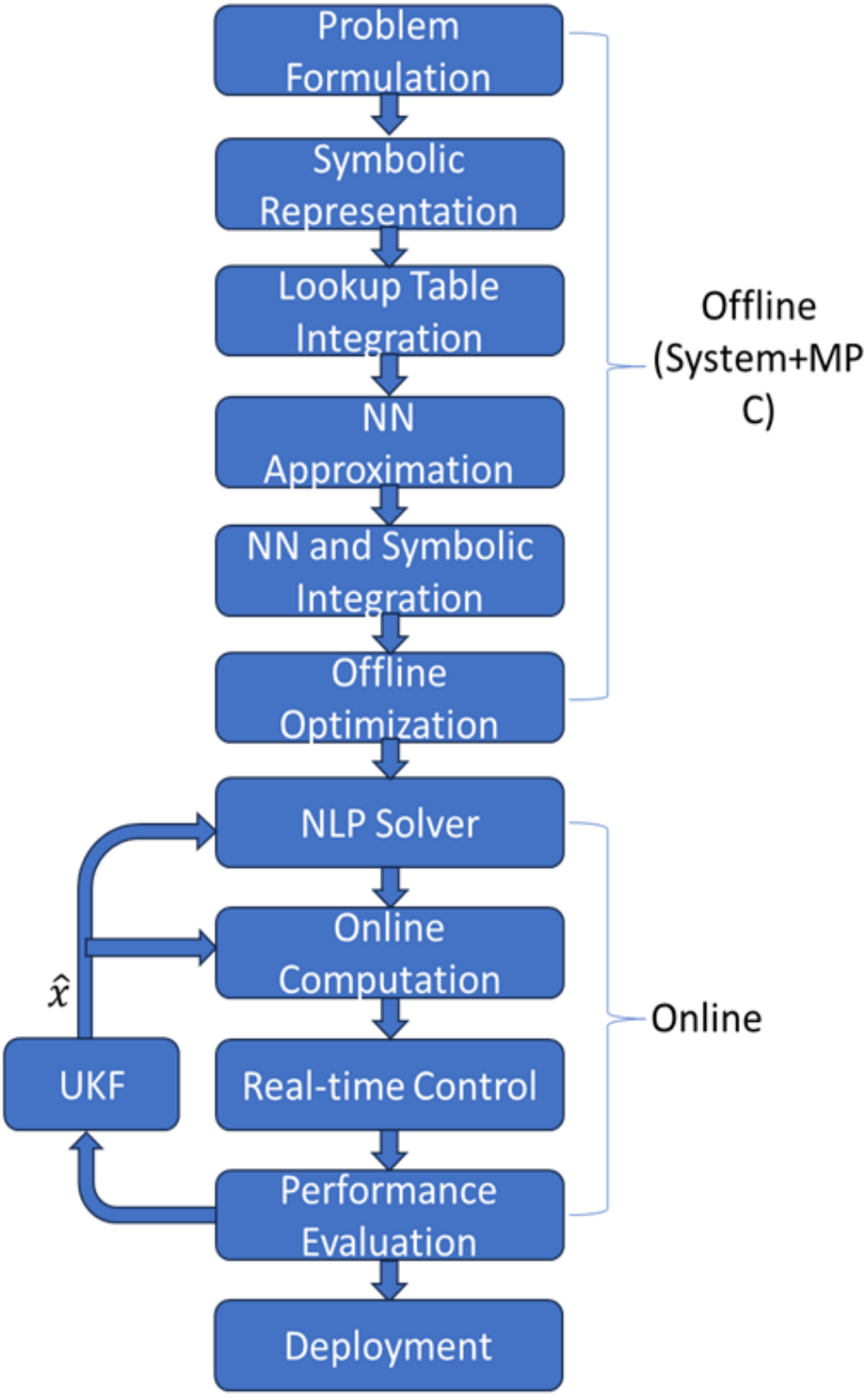

FIGURE 5.3: NN Based NMPC Flow

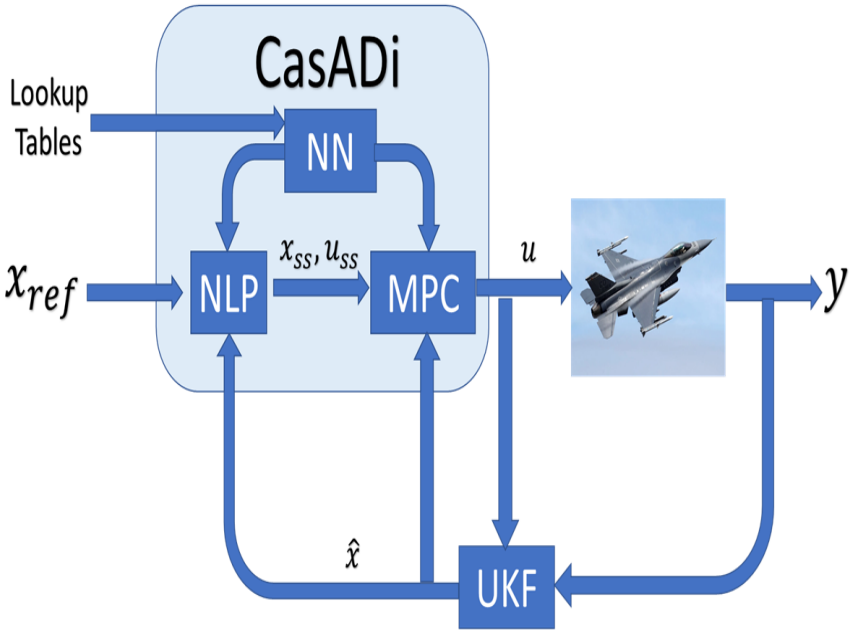

FIGURE 5.4: NN-based Closed Loop MPC Implementation

coefficients are constrained by the physical limits of the aircraft such as angle of attack, side slip angle, velocity, elevator angle, aileron angle, and rudder angle. The range of these aerodynamic coefficients depends on the physical limits of the control inputs and the operational envelope of the states. Hence, there's no need to approximate these variables beyond limits. The controller catered to these limits as well. The MPC block further bounds the system within limits defined by finding optimal solutions within the constraints defined. The "MPC" block finds the optimal controller to get the desired response. These three blocks i.e., NLP, NN, and MPC work in the CasADi environment to lessen the computational burden of the hardware. The output of MPC is then fed to the Aircraft which based on the input gives the output. The "UKF" block is the Unscented Kalman Filter, which is used to estimate the non-linear states of the system which are used as feedback to the controller.

The simulation process follows the sequence in which the first step is the non-linear function approximation of the model through NN. UKF estimates the system's non-linear states before the start of the iteration. NLP block then finds a secondary reference based on the available inputs. The MPC block then finds the optimal control input for perfect tracking of the AOA of the aircraft. The aircraft then

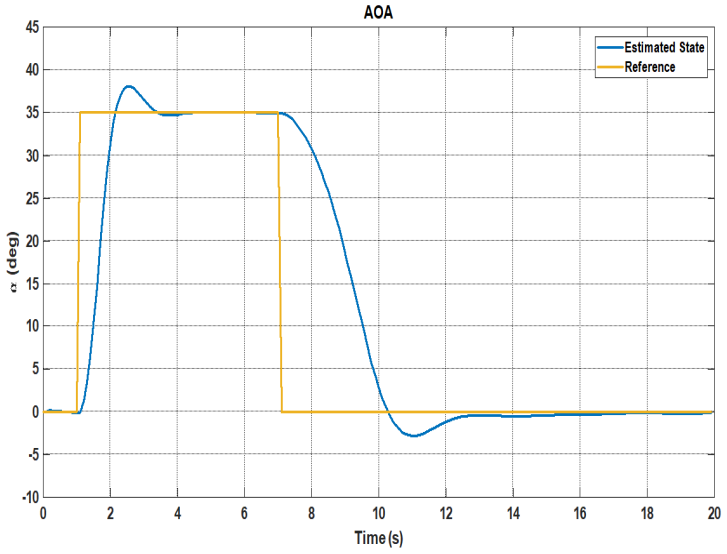

FIGURE 5.5: AOA Tracking using NN-based NMPC

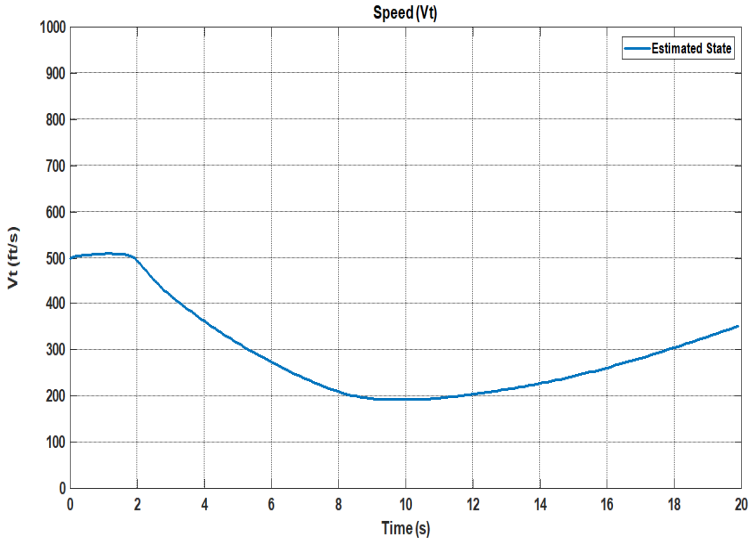

FIGURE 5.6: Speed Response using NN-based MPC

receives the control command and responds accordingly, and then gives the output to the UKF, and the process repeats on each cycle. The tuning parameters for each block have been found by trial and error approach for better performance.

In this simulation, an Angle of Attack maneuver is performed using the non-linear model of the F-16. The parameters of the aircraft are same as mention in aircraft model. An angle of attack maneuver is to be performed. Thus, the desired performance is to track the angle of attack while other states remain unchanged. The angle of attack maneuver changes the speed and altitude, so these states are

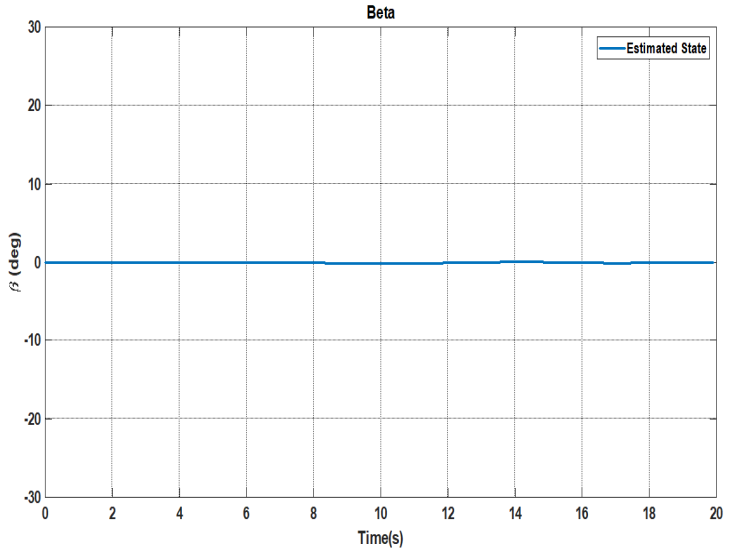

FIGURE 5.7: Side-Slip Angle Response using NN-based MPC

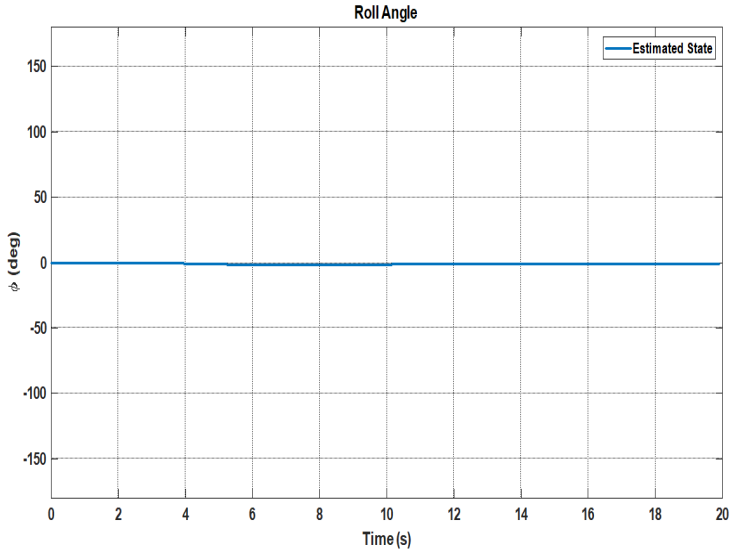

FIGURE 5.8: Roll Angle Response using NN-based MPC

not expected to be unchanged. The main control surfaces involve in this maneuver are throttle and elevator deflection. The controller is implemented in CasADi with an NN-based non-linear model of the aircraft. NN only provides the function approximation of the aerodynamic coefficients. Figure 5.5 shows the controller performance for tracking of non-linear controllers. The Aircraft is getting the desired Angle of Attack within a suitable time. The rise time t_r and settling time t_s of the closed-loop system are quite adequate. There is a little overshoot M_p that can be further improved. The performance parameters of the closed-loop

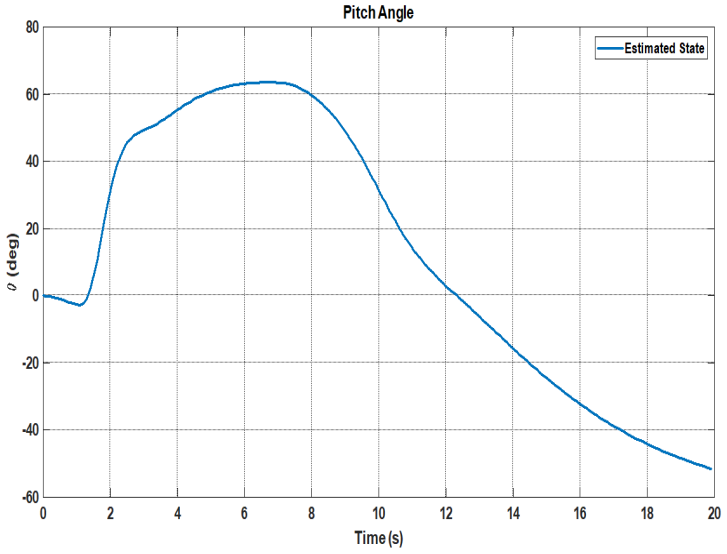

FIGURE 5.9: Pitch Angle Response using NN-based MPC

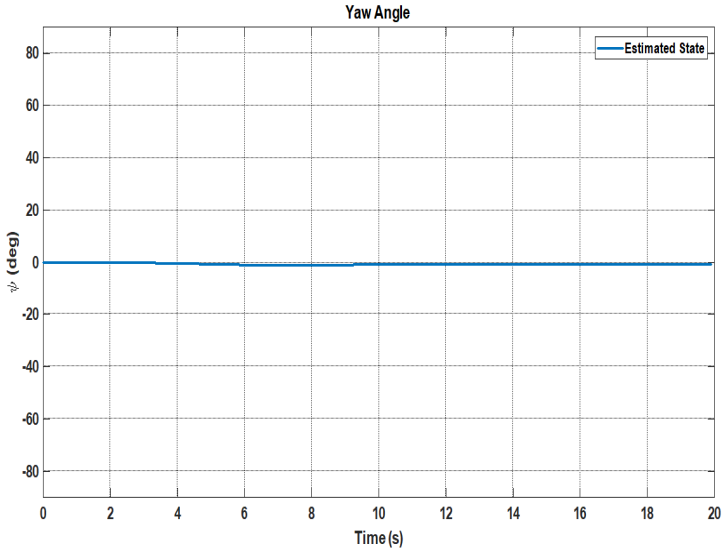

FIGURE 5.10: Yaw Angle Response using NN-based MPC

simulations for non-linear controllers are given as

$$\begin{aligned} t_r &= 1.00s \\ t_s &= 2.50s \\ M_p &= 8\% \end{aligned}$$

These results are compared to the maneuver control of fighter aircraft for the linear controller as in Chapter 3. In a linear controller, a linear model of the system and a linear controller is used. However, the latter results are for a non-linear model

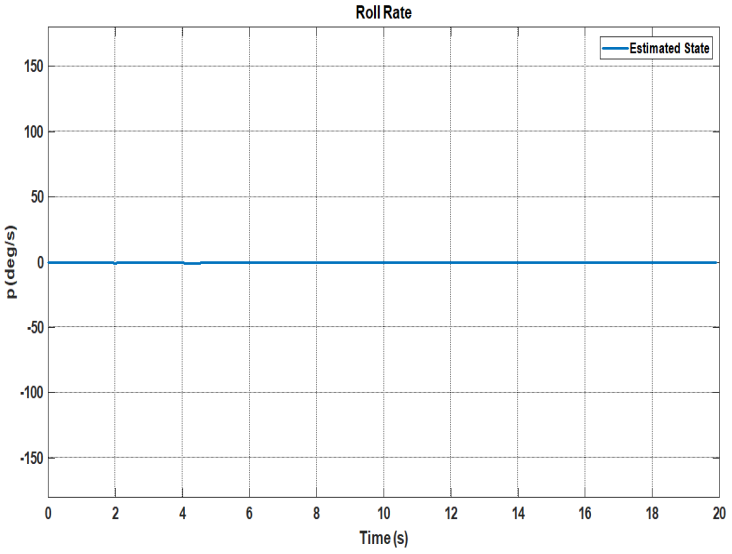

FIGURE 5.11: Roll Rate Response using NN-based MPC

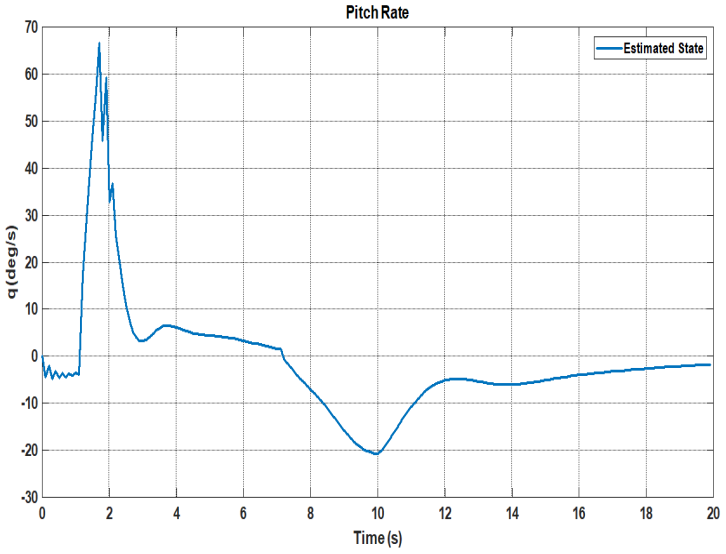

FIGURE 5.12: Pitch Rate Response using NN-based MPC

and a non-linear controller. The performance parameters are

$$\begin{aligned} t_r &= 1.30s \\ t_s &= 3.00s \\ M_p &= 8.3\% \end{aligned}$$

The performance comparison of the two methods clearly shows that the new technique performs better than the available technique. Also, this new technique is comparable to the linear MPC concerning computational complexity.

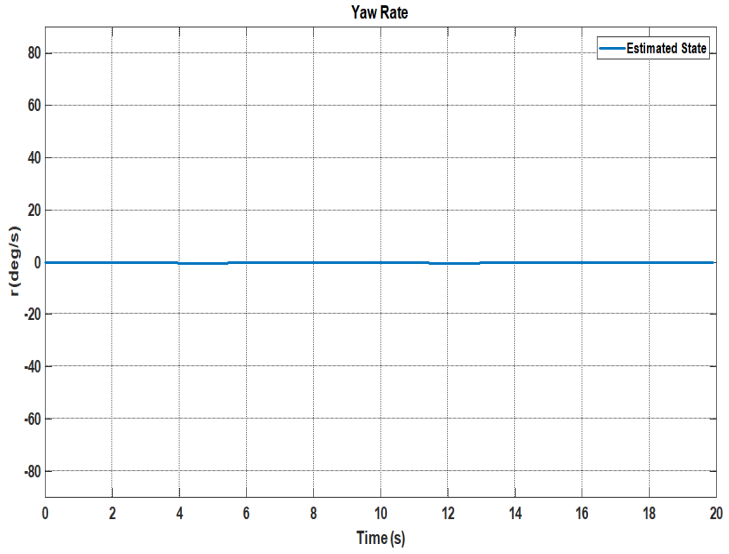

FIGURE 5.13: Yaw Rate Response using NN-based MPC

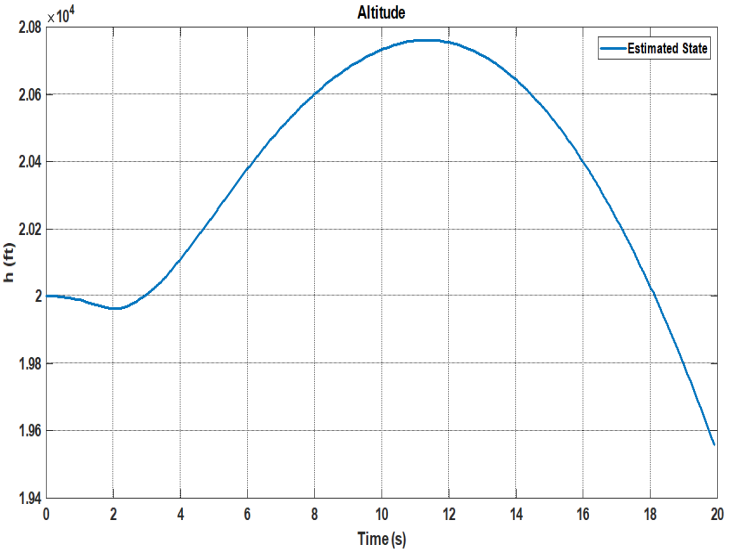

FIGURE 5.14: Altitude Response using NN-based MPC

The speed of the aircraft is shown in Figure 5.6, which is as expected from the desired maneuver. The speed goes down while performing the maneuver, and goes up when the maneuver is over. All other states from Figure 5.7 to 5.10 remain almost unchanged for lateral motion but change accordingly for longitudinal motion. These variables change slightly but are negligible. The altitude of the aircraft is shown in Figure 5.14, which indicates an increase during the maneuver and a decrease at the end of the maneuver. Figure 5.15 shows the longitudinal control surface deflection. It can be seen that the control limits reach their maximum

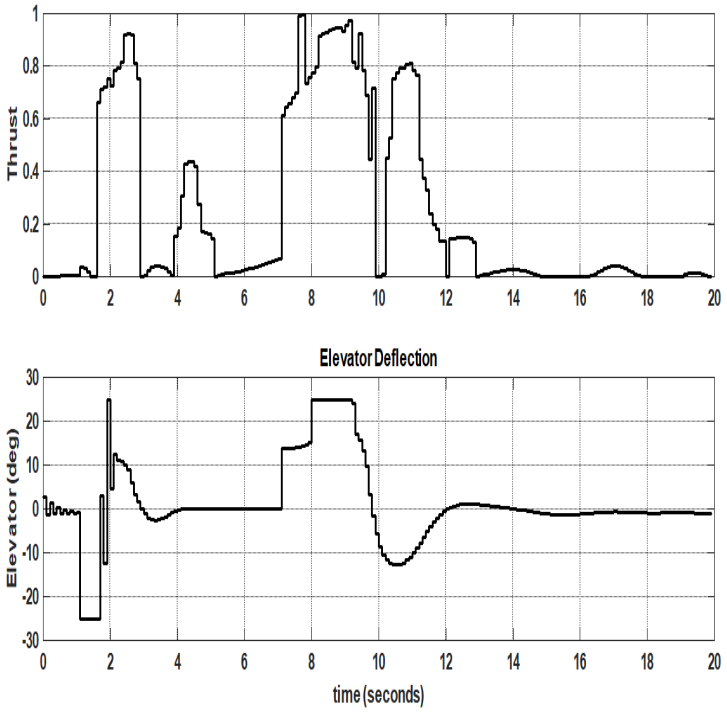

FIGURE 5.15: Longitudinal Input Controls using NN-based MPC

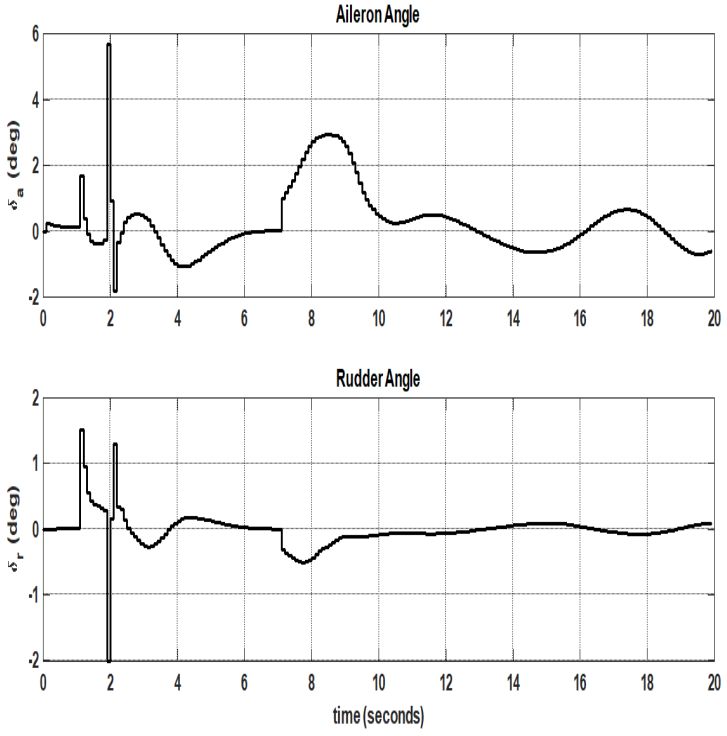

FIGURE 5.16: Lateral Input Controls using NN-based MPC

limits while performing the maneuver. However, the controller doesn't allow the surface above its physical limits. Hence, the control input is constrained. Figure 5.16 shows the lateral control surface deflection used to control the lateral motion of the aircraft while performing a maneuver.

5.8 Stability Analysis

To check the stability of the proposed control method, a Lyapunov-based stability analysis is performed. The Lyapunov-based stability analysis has been explained in [102]. The basic idea behind Lyapunov direct stability analysis is to choose an energy-like function for states x, and observe if its decay over time. A decreasing Lyapunov function indicates that the system is stable. For non-linear systems there's no general rule to select a Lyapunov function, and the presence of states and input constraints makes it almost impossible to check the global stability of the system. However, local stability can be proved by using a linear model of the system around a closed region of equilibrium. A usual approach is to select the Lyapunov function to be a positive definite function as in the equation below

$$V(\boldsymbol{x}) = \boldsymbol{x}^T Q \boldsymbol{x} \tag{5.48}$$

which has the property of a positive definite function, i.e., positive except when $\boldsymbol{x}$ is zero. The second property, which is the Lyapunov function, shows that it is monotonically decreasing.

$$V(\boldsymbol{x}) > 0 \quad \forall \boldsymbol{x} \neq 0, V(0) = 0 \tag{5.49}$$

$$\dot{V}(\boldsymbol{x}) \leq 0 \tag{5.50}$$

In NMPC, the Lyapunov candidate is chosen as the optimal cost-to-go value function, as defined

$$V(\boldsymbol{x}) = l \tag{5.51}$$

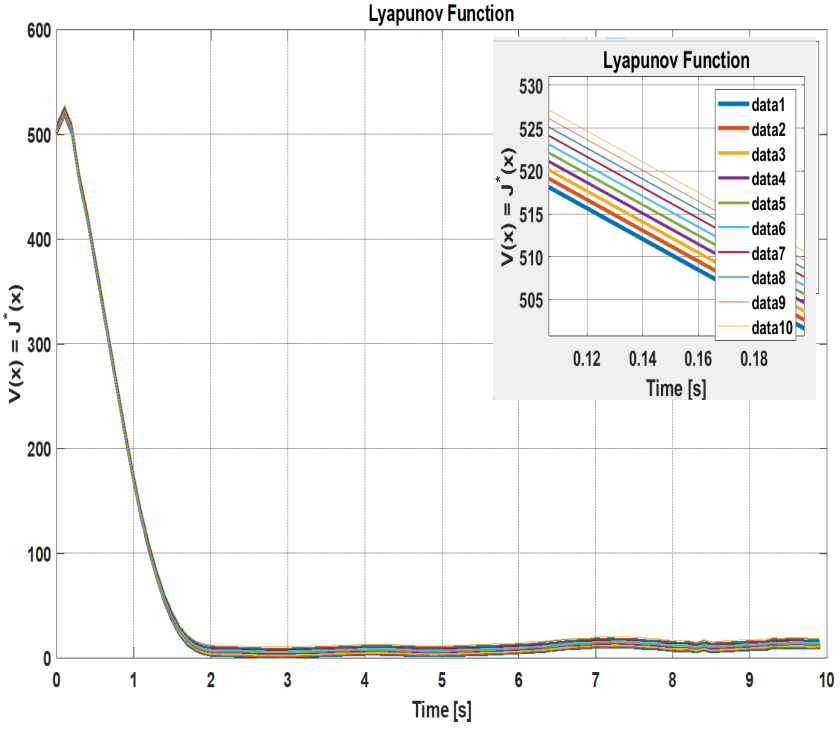

FIGURE 5.17: Lyapunov Stability Analysis

where l is defined in equation 5.15 and is given as

$$\min_{\boldsymbol{U}} \; l = \boldsymbol{E}^T Q \boldsymbol{E} + \boldsymbol{U}^T R \boldsymbol{U} \tag{5.52}$$

Which is the cost function in case of nonlinear MPC. The goal is to find out that the Lyapunov candidate decays over time.

$$V(\boldsymbol{x}_{k+1}) - V(\boldsymbol{x}_k) < 0 \tag{5.53}$$

A simulation-based Lyapunov stability analysis is performed by assuming the Lyapunov candidate is the same as the cost function. To check the robust stability of the system, several simulations have been performed with random initial conditions of the states around the equilibrium point. The results are shown in Figure 5.17, which proves the local stability of the controller. The Lyapunov function is slightly increasing in some moments that is because of the NN approximation of the aircraft. More careful approximation and more tuning of the controller will solve this issue.

5.9 Conclusion

The proposed method overcomes the drawback of the computational complexity associated with non-linear MPC to control the fast dynamic systems while remaining within physical control limits. NMPC may produce steady-state error in some cases. The combination of Offset-free MPC and NMPC mitigates the steady-state error. The only problem with NMPC is the high computation cost due to solving non-linear optimal control problems on each sampling instant. The proposed scheme addresses this problem with an offline simplification of NMPC using deep learning and symbolic approximation which would work in the CasADi toolbox for non-linear controller optimization and implementation. The integration of an unscented Kalman filter and NLP solver further improves robustness while mitigating the steady-state error.

Simulation results show the efficient performance of the proposed scheme for performing a high-angle of attack maneuver for fighter aircraft while showcasing the good rise and settling time. The computational complexity is reduced by solving a major part of the problem offline and finding the optimal solution online by using the simplified model. The computational complexity is calculated by performing simulations for a simulation time of 10 sec conducted in Matlab 2024 on a Dell Laptop with a 13th Gen Core i7-1355U processor (1.7GHz) and 16GB of RAM. The results are compared with those of the linear MPC controller in Chapter 3. The linear MPC controller takes on average 540sec, however, this proposed controller takes on average 214sec. The computational efficiency of the proposed controller is much better than the linear MPC.

A limitation of the proposed method is that it can't operate the system at its maximum limit, i.e., $\alpha = 45°$. This method can only cater to uncertainties and disturbances to some extent, resulting in performance reduction or system instability. A further improvement to this method could be to use a variation of the NMPC to remove the effects of these uncertainties and disturbances effectively. A fault-tolerant model can also be considered in further research.

The proposed method can reduce the computational complexity to make real-time implementation possible for fast dynamic systems. Most of the problems‘ can be

solved offline, but the online solution of NMPC in each iteration under constraints is still computationally challenging. A computationally efficient solution to this online optimization could broaden the applicability of the proposed technique.
The CasADi platform works well with the hardware. So a future direction would be to perform the Hardware-in-loop-based simulations for validation of the controller in the actual environment. This research has the potential for implementation for other fast dynamic systems i.e., spacecraft and UAVs etc.

Chapter 6

NN-based Offset-free Scenario-based MPC

6.1 Introduction

The previous chapter's NN-based NMPC provides good tracking while reducing the computational complexity of MPC [103]. This technique also mitigates uncertainty to some extent. However, for applications like fighter aircraft, uncertainty and external disturbances are not limited and must be efficiently catered to by the controller.

While performing complex maneuvers, the aircraft suffers from high aerodynamic forces [82] and the control surfaces can't reach their maximum limits. This is called an actuator fault. Actuator faults can be categorized as (i) permanent faults and (ii) temporary faults. Permanent faults are the ones in which the actuator fixes on a certain position and is unable to move. These faults can only be covered by some other actuator. Temporary faults are those in which the actuator is unable to get full deflection while performing complex maneuvers for a short period. After the maneuver, the actuator regains its original limits. The actuator's maximum deflection limit doesn't remain fixed throughout the maneuver but changes with time and aerodynamic force. These time-varying faults have been dealt with in [82] by using fault detection and MPC. However, only linear MPC and a linear model of the aircraft have been considered in that research. This chapter will

consider non-linear MPC and non-linear aircraft models.

The time-varying faults or the environmental effects pose uncertainty and disturbances in the model and should be catered to in the controller model. A special type of controller that handles uncertainty and considers constraints in its design is the scenario-based MPC (SMPC) [104]. SMPC solves non-linear MPC while considering different levels of uncertainty at each time step of the horizon, hence making an uncertainty tree. This uncertainty tree may contain fixed levels of uncertainty or vary at each level. This research considers only fixed uncertainty levels.

SMPC may also give an offset at the steady-state point. This problem can be solved by using offset-free MPC which solves another optimization problem at each sampling point to find the secondary reference [86]-[91]. Then SMPC follows the secondary reference instead of the original reference to remove the steady-state error. The use of SMPC with different levels of uncertainty at each time step, as well as solving another optimization problem along with MPC, makes the controller computationally complex.

The above-mentioned control scheme is ideal for constraint handling and optimal performance, but suffers from the computational burden. Because of this computational burden, this scheme is not useful for fast dynamic systems i.e., fighter aircraft. This research proposes an NN-based symbolic solution to the MPC problem in an offline manner for computational reduction. This offline solution solves the complex matrices offline and provides an algebraic expression. This algebraic expression can be used for online optimization based on the current scenario. The symbolic simplification of the SMPC problem can be achieved by a tool CasADi [96]. Thus, as in the previous chapter, the sMPC problem is then solved symbolically instead of numerically at the start, and a solution is obtained offline that contains algebraic expressions only. Then, in each iteration of optimization, it finds the solution numerically by just solving the algebraic expression instead of solving whole matrices. Hence, it reduces complexity to a large extent when solving complex problems, i.e., MPC. This research considers the use of CasADi with MATLAB only for controller design and application.

As in the previous chapter, the use of CasADi needs a model of the system that contains either constants or variables to be optimized. The problem where the model contains the state and input dependent variables (available in the form of lookup tables), the CasADi is also unable to tackle them. This type of problem is encountered in controlling the complex model, i.e., the aircraft model. The aircraft model contains the aerodynamic coefficients which are available in lookup tables and change the value with states and inputs. This type of problem can be solved using the function approximation of the lookup table for each aerodynamic coefficient, and then the problem converts to optimization variables only. Here again, Neural Networks (NN) is used for approximating the aerodynamic coefficients [97]-[98]. Also, this research uses NN as the function approximator for the aerodynamic coefficients only.

The use of offset-free MPC requires the value of the estimated states in the presence of external disturbances and noise because these values deteriorate the performance. Among many available estimators, the Kalman filter is assumed to be the optimal estimator. As for the complex system, i.e., fighter aircraft Unscented Kalman Filter (UKF) provides the best performance in the presence of external disturbances. That's why this research uses UKF for state estimation [92]-[93].

The proposed scheme in this chapter takes advantage of SMPC and NN-based offset-free NMPC in the previous chapter. This scheme transforms the aircraft control problem into symbolic formation and an NN-based approximated function in an offline fashion. Then it solves the NLP problem for offset-free reference and solves SMPC under different levels of uncertainties online. UKF helps estimate the system states in the presence of uncertainties.

The rest of the chapter follows the following pattern. Section 6.2 describes the specifications and constraints involved for the fighter aircraft used. It also contains the description and requirements for the Herbst-type maneuver. Section 6.3 includes the MPC problem, which is followed by scenario-based MPC and offset-free variation. A proposed solution for computationally efficient implementation of the chosen controller is discussed in 6.4, which describes the use of the CasADi toolbox and Neural Network (NN). This is further followed by UKF for the non-linear

state estimation. Section 6.5 tells about the time-varying faults while performing maneuvers. Closed-loop simulation and simulation results are shown in section 6.6. Computational complexity analysis is performed in section 6.7. Section 6.7 and Section 6.8 gives the conclusion and future directions for the current work.

6.2 Fighter Aircraft Specifications

Fighter aircraft are complex systems from a control point of view, with multiple states and multiple inputs. The model used for the current research is the F-16 model [100]. F-16 is one of the modern fighter aircraft and is being used widely in research because its data is easily available. The model contains 13 states and 4 inputs. 2 of the states are uncontrollable and, hence, not considered in this research. The current model contains 11 states and 4 inputs. The 11 states and 4 inputs in vector form are as

$$\boldsymbol{x}^T = [V_t \;\; \alpha \;\; \beta \;\; \phi \;\; \theta \;\; \psi \;\; p \;\; q \;\; r \;\; h \;\; P_a]$$

$$\boldsymbol{u}^T = \begin{bmatrix} th & \delta_e & \delta_a & \delta_r \end{bmatrix}$$

where V_t is the true velocity. α and β are the angle of attack and side slip angle. ϕ, θ, ψ are the roll angle, pitch angle, and yaw angle. p, q, and r are the roll rate, pitch rate, and yaw rate. h is the height and P_a is the engine power. The control vector contains 4 inputs namely thrust th, elevator deflection δ_e, aileron deflection δ_a, and rudder deflection δ_r respectively.

The aircraft's general equation of motion has not been shown here. However, some specific parameters related to F-16 aircraft are shown in Table 6.1. Aircraft have certain limits on states and control inputs as in Table 6.2.

For testing the controller's efficiency, a Herbst-type maneuver is performed. Herbst-type maneuver is one of the complex maneuvers and is essential for air superiority in a dogfight. This maneuver was named after Dr. W.B. Herbst as he proposed the maneuver. The complexity of this maneuver can be realized by the fact that it involves a strong coupling between longitudinal and lateral motion. In control

TABLE 6.1: Aircraft Parameters

Symbol	Name	Value	Unit
	Moment of Inertia		
M_{xx}	x-axis	9496	$slug - ft^2$
M_{yy}	y-axis	55814	$slug - ft^2$
M_{zz}	z-axis	63100	$slug - ft^2$
M_{xz}	Product of xz	982	$slug - ft^2$
	Geometry		
W	Weight	20500	lb
B	Span	30	ft
S	Area	300	ft^2
$\bar{c}$	Mean Aerodynamics Chord	11.32	ft
H_g	Engine Angular Momentum	160	$slug - ft^2/s$
α_T	Thrust Vector Angle	0°	
x_{CGR}	Ref cg in c	0.35	

TABLE 6.2: Aircraft Constraints

Symbol	Name	Constraint
	States	
V_T	Velocity	$390 \leq v_t \leq 900\, ft/s$
h	Altitude	$0 \leq h \leq 40000\, ft$
P_a	Engine Power	$0 \leq P_a \leq 100\,\%$
	Inputs	
	Longitudinal Control	
th	Thrust	$0 \leq th \leq 1$
δ_e	Elevator Angle	$-25^\circ \leq \delta_e \leq 25^\circ$
	Lateral Control	
δ_a	Aileron Angle	$-21.5^\circ \leq \delta_a \leq 21.5^\circ$
δ_r	Rudder Angle	$-30^\circ \leq \delta_r \leq 30^\circ$

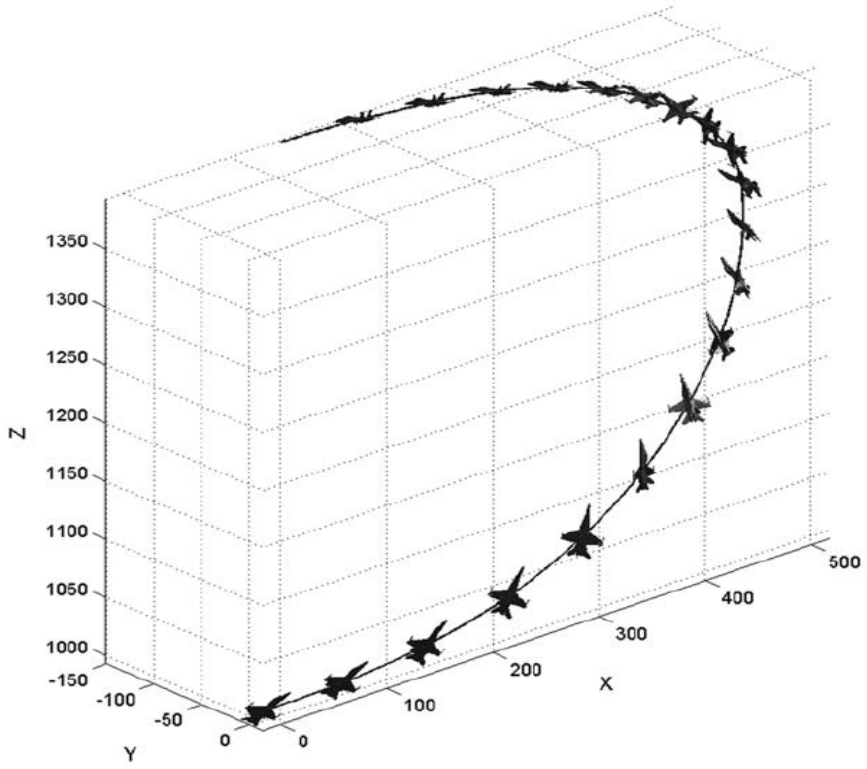

FIGURE 6.1: Herbst Maneuver

terms, it involves the tracking of AOA, velocity, and roll. A complete Herbst maneuver can be realized by Figure 6.1. In this maneuver, the aircraft goes at a high angle of attack at the start. Then, at this moment it rolls and changes the direction of velocity. After changing its direction it comes back to the equilibrium point. In effect, the aircraft goes at high altitude with its direction backward from the start of the maneuver.

In effect, the aircraft goes on a high angle of attack, thus reducing speed. Subsequently, it rolls and changes the direction of the velocity. After that, it reduces the angle of attack and maintains the speed in the reverse direction. From a control point of view, it involves the tracking of the angle of attack and roll angle while maintaining the side slip angle at zero. The strong coupling between the two modes of the aircraft makes the choice of the controller very critical. For achieving highly complex maneuvers, maximum control effort might be needed to achieve the best performance and quick response. Therefore, NMPC is the best choice for this type of maneuver control, which can give optimal performance while handling constraints.

6.3 NN-based Offset-Free Scenario-based MPC

NMPC is the one that solves optimization problems while using the non-linear prediction model of the system. One of the advantages of NMPC is that, unlike linear MPC, we don't need to change the model at each time step of the MPC. The NMPC problem is solved by finding the minimum cost (error) for a finite horizon of input i.e.,

$$J = \min_{\forall \boldsymbol{u}(j)} \left(\sum_{i=1}^{N} \boldsymbol{e}(i)^T q \, \boldsymbol{e}(i) + \sum_{j=1}^{N-1} \boldsymbol{u}(j)^T r \, \boldsymbol{u}(j) \right) \tag{6.1}$$

where N denotes the prediction horizon and

$$\boldsymbol{e}(i) = \boldsymbol{x}(i) - \boldsymbol{R}_{ref} \tag{6.2}$$

which denotes the state error in time step i from the current step with $\boldsymbol{x}(i)$ as the state at that time step and $\boldsymbol{R}_{ref}$ as the desired state. $\boldsymbol{u}(j)$ is the control input at time step j from the current input. The values of the future states can be found by using the non-linear model of the system

$$\boldsymbol{x}(i+1) = f(\boldsymbol{x}(i), \boldsymbol{u}(i)) \tag{6.3}$$

$$\boldsymbol{y}(i) = g(\boldsymbol{x}(i), \boldsymbol{u}(i)) \tag{6.4}$$

f and g are the non-linear functions of the states and inputs. The states and inputs have to fulfill the following inequality constraints

$$\boldsymbol{x}^{min} \leq \boldsymbol{x}(i) \leq \boldsymbol{x}^{max} \tag{6.5}$$

$$\boldsymbol{u}^{min} \leq \boldsymbol{u}(j) \leq \boldsymbol{u}^{max} \tag{6.6}$$

in which $\boldsymbol{x}^{min}$ is the lowest value of the state and $\boldsymbol{x}^{max}$ is the state's highest value. Similarly, $\boldsymbol{u}^{min}$ and $\boldsymbol{u}^{max}$ are the lowest and highest values of the inputs, respectively. The matrices q and r can be defined as:

$$q = \text{diag}(n) \tag{6.7}$$

$$r = \text{diag}(m) \tag{6.8}$$

Which shows the diagonal matrices for n number of states and m number of inputs. The NMPC solves the control problem efficiently. However, this controller cannot cater to environmental disturbances and uncertainties. The application, i.e., fighter aircraft, suffers from unavoidable disturbances and, thus, should be catered to in the controller design for better performance. Model-based controllers require the model to be perfect. However, environmental disturbances and modeling uncertainties may pose serious issues in controller design and applications. For applications like fighter aircraft, these issues may lead to disastrous effects.

Scenario-based MPC resolves this issue by finding the optimal value for each different uncertainty at each time step of the horizon. In NMPC, the prediction model gives the future states for a current state estimate, which is further used to find the optimal input for the next step. However, in SMPC, the prediction model predicts the future states based on the current state and input. It also predicts the future states based on possible uncertainty in control input and states. Then, using all of these predicted values, finds the optimal by minimizing the expected value of the cost function instead of the simple cost function.

A scenario tree for horizon length $N = 3$ and the number of scenarios $s = 2$ levels is shown in Figure 6.2. The scenario tree is constructed as follows. When $N = 1$, based on the current state $\boldsymbol{x}_0$, the model predicts the future state $\boldsymbol{x}_1$. Because of considering two scenarios (i.e., $s = 2$), if finds two values $\boldsymbol{x}_1^{(1)}$ and $\boldsymbol{x}_1^{(2)}$ based on the input $\boldsymbol{u}_0$ and estimated uncertainty $\boldsymbol{w}_0^{(1)}$ and $\boldsymbol{w}_0^{(2)}$. In second horizon (i.e., $N = 2$), the states $\boldsymbol{x}_2^{(1)}$ and $\boldsymbol{x}_2^{(2)}$ are predicted based on the values of $\boldsymbol{x}_1^{(1)}$ and estimated uncertainties. Similarly, state $\boldsymbol{x}_1^{(2)}$ gives the values of the states $\boldsymbol{x}_2^{(3)}$ and $\boldsymbol{x}_2^{(4)}$. Then the same process will be repeated in the $3rd$ horizon, which will give 8 states out of 4. This process will go on with the horizon length. The scenario tree grows with the horizon length and the number of scenarios, hence increasing computational complexity. The SMPC then solves the optimization problem of

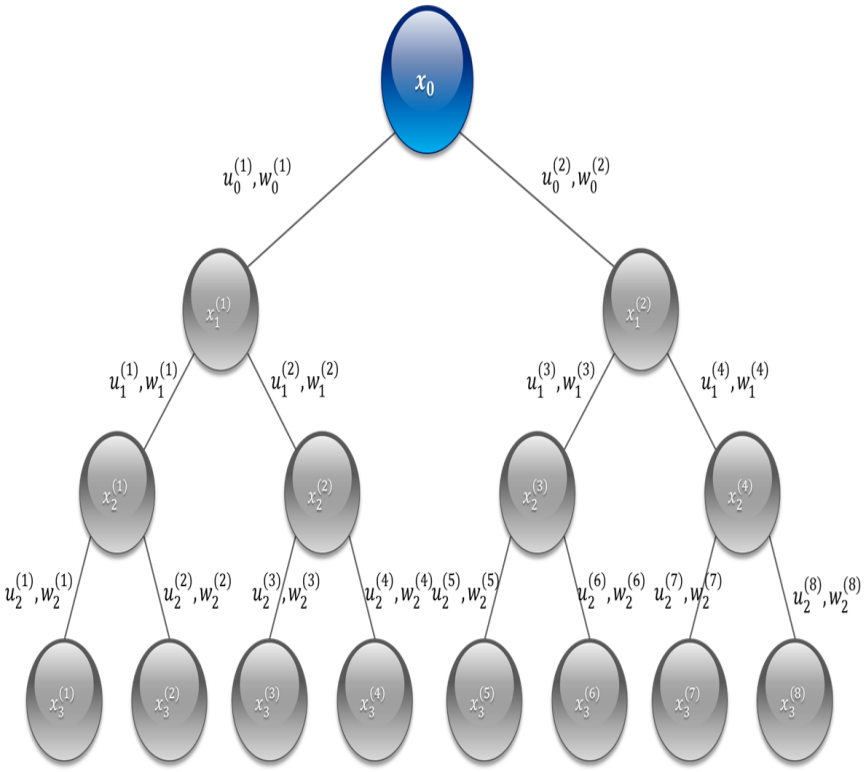

FIGURE 6.2: Scenario Tree

the form of equation

$$\begin{aligned}\min_{\forall u(i)} J \;=\; & \boldsymbol{e}(0)^T q\, \boldsymbol{e}(0) + \boldsymbol{u}(0)^T r\, \boldsymbol{u}(0) + \mathbb{E}\Big\{ \big(\boldsymbol{e}(1)^T q\, \boldsymbol{e}(1) + \boldsymbol{u}(1)^T r\, \boldsymbol{u}(1)\big) \\ & + \ldots + \mathbb{E}\Big\{ \big(\boldsymbol{e}(2)^T q\, \boldsymbol{e}(2) + \boldsymbol{u}(2)^T r\, \boldsymbol{u}(2)\big) + \mathbb{E}\Big\{ \big(\boldsymbol{e}(N-1)^T q\, \boldsymbol{e}(N-1) + \\ & \boldsymbol{u}(N-1)^T r\, \boldsymbol{u}(N-1) + \ldots + \mathbb{E}\Big\{ \boldsymbol{e}(N)^T q\, \boldsymbol{e}(N) \Big\}\Big\}\Big\}\Big\} \end{aligned} \tag{6.9}$$

where $\mathbb{E}$ denotes the conditional expectation with respect to uncertainty w and calculated by

$$\mathbb{E}\big\{\boldsymbol{e}(\boldsymbol{x}, \boldsymbol{u}, \boldsymbol{w})\big\} = \sum_{i=1}^{s} p_i \boldsymbol{e}(\boldsymbol{x}, \boldsymbol{u}, \boldsymbol{w}(i))) \tag{6.10}$$

s denotes the number of scenarios, and p_i denotes the weights for each uncertainty. The number of scenarios and their weighting numbers are not fixed. These values can be chosen based on the uncertainties available in the system, and on how important it is to remove the uncertainty. e represents the error at time step i

from the current time and is difference between state $x(i)$ and desired state x_{des} which is defined as

$$\boldsymbol{e}(i) = \boldsymbol{x}(i) - \boldsymbol{R}_{ref} \tag{6.11}$$

The state $\boldsymbol{x}(i)$ can be found by the non-linear function of the model

$$\boldsymbol{x}(i+1) = f\left(\boldsymbol{x}(i), \boldsymbol{u}(i)\right) + B_d\, \boldsymbol{w}(i) \tag{6.12}$$

where $\boldsymbol{w}$ is used for disturbance in the system at the current time step. This can be found by the disturbance model. The controller must fulfill the following constraints

$$\boldsymbol{x}^{min} \;\leq\; \boldsymbol{x}(i) \;\leq\; \boldsymbol{x}^{max} \tag{6.13}$$

$$\boldsymbol{u}^{min} \;\leq\; \boldsymbol{u}(j) \;\leq\; \boldsymbol{u}^{max} \tag{6.14}$$

SMPC is good for handling environmental disturbances and system uncertainties. However, in applications i.e., fighter aircraft it may give a steady-state error. The steady-state error can either be removed by inserting an integrator in the closed-loop model, but that means inserting another controller along with MPC. However, offset-free MPC resolves this issue by changing the controller design a little bit, and there's no need for an extra controller to remove steady-state error. The offset-free MPC finds the secondary reference for the steady state in the presence of system uncertainties and disturbances. This is done by solving the optimization problem at each sampling interval for the secondary reference.

$$l = \min_{\boldsymbol{x}_{ss}, \boldsymbol{u}_{ss}} \quad \boldsymbol{e}_{ss}^T q_{ss} \boldsymbol{e}_{ss} + \boldsymbol{u}_{ss}^T r_{ss} \boldsymbol{u}_{ss} \tag{6.15}$$

where

$$\boldsymbol{e}_{ss} = \boldsymbol{x}_{ss} - \boldsymbol{R}_{ref} \tag{6.16}$$

$$\boldsymbol{x}_{ss} = f(\boldsymbol{x}_{ss}, u) + B_d \boldsymbol{w}(i) \tag{6.17}$$

$$\boldsymbol{x}^{min} \leq \boldsymbol{x}_{ss} \leq \boldsymbol{x}^{max} \tag{6.18}$$

$$\boldsymbol{u}^{min} \leq \boldsymbol{u}_{ss} \leq \boldsymbol{u}^{max} \tag{6.19}$$

Now the SMPC is to solve

$$\begin{aligned}\min_{\forall \boldsymbol{u}(i)} l_e \;=\; & \boldsymbol{e}(0s)^T q\, \boldsymbol{e}(0s) + \boldsymbol{u}(0s)^T r\, \boldsymbol{u}(0s) + \mathbb{E}\Big\{ \big(\boldsymbol{e}(1s)^T q\, \boldsymbol{e}(1s) + \boldsymbol{u}(1s)^T r\, \boldsymbol{u}(1s)\big) \\ & + \mathbb{E}\Big\{ \big(\boldsymbol{e}(2s)^T q\, \boldsymbol{e}_(2s) + \boldsymbol{u}_(2s)^T r\, \boldsymbol{u}_(2s)\big) + ... + \\ & \mathbb{E}\Big\{ \big(\boldsymbol{e}((N-1)s)^T q\, \boldsymbol{e}((N-1)s) + \boldsymbol{u}((N-1)s)^T r\, \boldsymbol{u}((N-1)s)\big) + \\ & \mathbb{E}\Big\{ \boldsymbol{e}(Ns)^T q\, \boldsymbol{e}(Ns) \Big\} \Big\} \Big\} \Big\}\end{aligned} \tag{6.20}$$

$$\mathbb{E}\big\{\boldsymbol{e}(is)(\boldsymbol{x}, \boldsymbol{u}, \boldsymbol{w})\big\} = \sum_{j=1}^{s} p_j \boldsymbol{e}(is)(\boldsymbol{x}, \boldsymbol{u}, \boldsymbol{w}(j))) \tag{6.21}$$

e denotes the error in time step i from the secondary reference found above and is calculated as

$$\boldsymbol{e}(is) = \boldsymbol{x}(is) - \boldsymbol{x}_{ss} \tag{6.22}$$

With the system model represented by the non-linear equation

$$\boldsymbol{x}(i+1) = f\left(\boldsymbol{x}(i), u(i)\right) + B_d\, \boldsymbol{w}(is) \tag{6.23}$$

Under the constraints

$$\boldsymbol{x}^{min} \leq \boldsymbol{x}_{is} \leq \boldsymbol{x}^{max} \tag{6.24}$$

$$\boldsymbol{u}^{min} \leq \boldsymbol{u}_{is} \leq \boldsymbol{u}^{max} \tag{6.25}$$

The NMPC is itself computationally complex. The addition of uncertainty-based scenarios leads to finding more optimization variables. The increasing number

of optimization variables further increases the computational complexity. This problem will be resolved in the next section.

6.4 Computationally Efficient Implementation

The above-mentioned controller is quite efficient in tracking, but computationally inefficient. This inefficiency limits the controller's real-time implementation for fast dynamic systems, e.g., fighter aircraft. The fast dynamic systems require a response in milliseconds, and any delay in the controller response may lead to reduced performance. This chapter resolves this issue with a new NN-based solution that uses offline function approximation of the controller model and then online optimization of the control inputs based on that approximated function. A toolbox named CasADI also helps in the symbolic simplification of the function. This way of reducing computational complexity and making the MPC problem simple finds the use of CasADi for the control of complex systems i.e., OSMPC-based control of fighter aircraft. The non-linear solver used for CasADi is of the form:

$$\min_{\boldsymbol{x},\boldsymbol{u}} \quad f(\boldsymbol{x},\boldsymbol{u}) \tag{6.26}$$

where

$$\boldsymbol{x}^{min} \leq \boldsymbol{x} \leq \boldsymbol{x}^{max} \tag{6.27}$$

$$\boldsymbol{u}^{min} \leq \boldsymbol{u} \leq \boldsymbol{u}^{max} \tag{6.28}$$

$$g^{min} \leq g(\boldsymbol{x},\boldsymbol{u}) \leq g^{max} \tag{6.29}$$

Like in the previous chapter, the lookup tables for the aerodynamic coefficients have been approximated through Neural Network for use in the CasADi platform. The OSMPC needs information on disturbances and uncertainties to cater to them in design. Also, not all the system's states are measurable. There are various ways

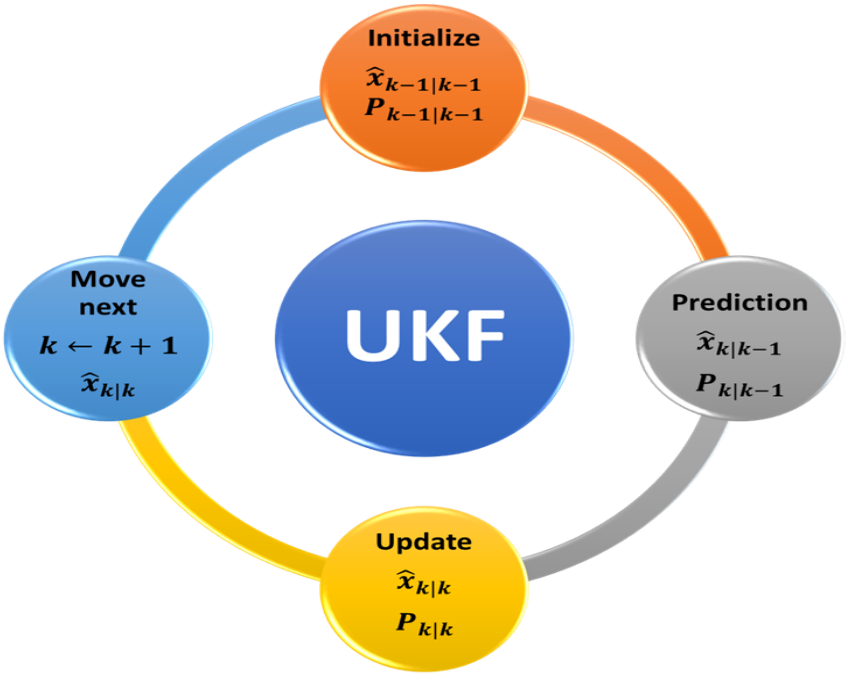

FIGURE 6.3: UKF Cycle

to estimate this disturbance. One of them is to estimate the system through a non-linear estimator and find the difference between the measured value and the estimated value as

$$\hat{\boldsymbol{w}} = \hat{\boldsymbol{x}} - \boldsymbol{x}_m \tag{6.30}$$

where

$$\hat{\boldsymbol{x}}(i+1) = f(\hat{\boldsymbol{x}}(i), \boldsymbol{u(i)}, \hat{\boldsymbol{w}}) \tag{6.31}$$

and the measured value is

$$\boldsymbol{x}(i+1) = f(\boldsymbol{x}(i), \boldsymbol{u}(i)) \tag{6.32}$$

For optimal estimation, UKF has been extensively used for non-linear state estimation in many applications involving full-state feedback. This research makes use of the UKF for non-linear state estimation of the aircraft. The UKF algorithm contains four steps involving initialization, finding sigma points, time update, and measurement update. The four steps can be viewed in Figure 6.3.

The structure of the proposed scheme can be summarized in Figure 6.4.

6.5 Time Varying Faults Tolerance

Environmental disturbances and aging effects in all the plants lead to performance degradation of the controller. The performance goes to the worse scenario and even leads to instability in several critical applications, i.e., fighter aircraft. Slight performance degradation may lead to drastic results in dog fights. Therefore, these effects must be catered to in controller design and applications.

For applications like fighter aircraft, the controller performance may suffer from two major issues, i.e., sensor fault and actuator fault. This work focuses on the actuator faults. Actuator faults are also of two types, i.e., Permanent and temporary. Permanent faults are the ones in which the actuator goes to a certain position and is unable to recover at any time. This is also known as a stuck fault, as the actuator is stuck in a certain position. No matter what the desired position suggested by the controller, the aircraft is unable to provide that position. This can be easily understood by Figure 6.5. The controller can only cater to these types of faults if there's an alternative to that control surface. In this research, this type of fault has not been considered.

Another type of fault, which is most prominent in aircraft maneuvers and is the focus of current research, is temporary jamming or temporary faults. In applications like fighter aircraft, this type of fault may also be referred to as stall load. In this type, the actuator is not damaged but suffers from high aerodynamic forces. Due to these aerodynamic forces, the actuator can't reach its extreme limits, which may lead to reduced performance or worse in certain scenarios. This type of fault is shown in Figure 6.6. In effect, the actuator can't go beyond the reduced limit, even if the controller requirement is more. However, this kind of fault is there for a short period, i.e., while performing maneuvers. Once the maneuver is over, the actuators regain their original limits. Since these types of faults may appear in any aircraft while performing maneuvers, therefore must be catered to in the controller design. Thus, the current research focuses on the control of these types of faults along with optimal tracking.

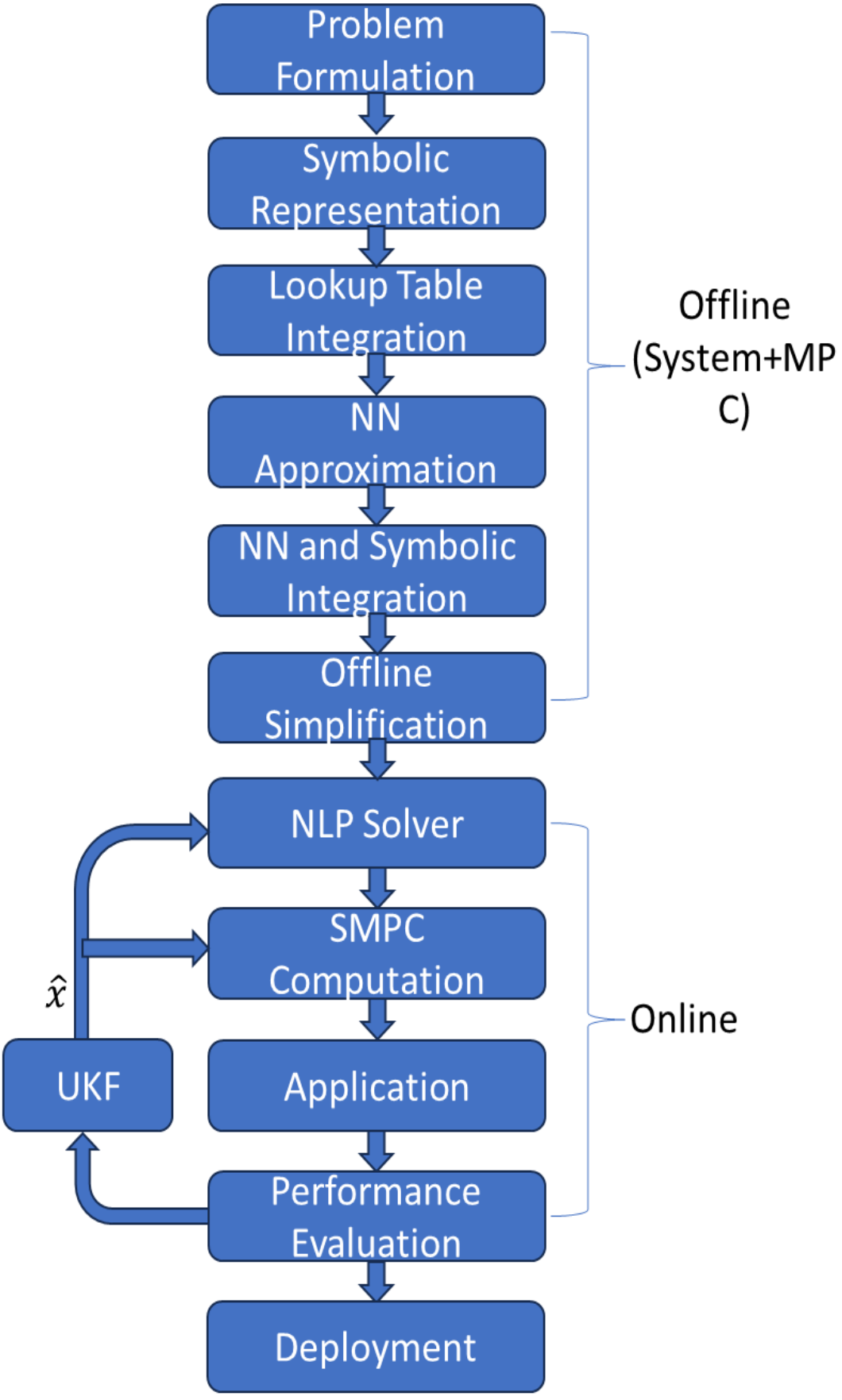

FIGURE 6.4: NN-based OSMPC Structure

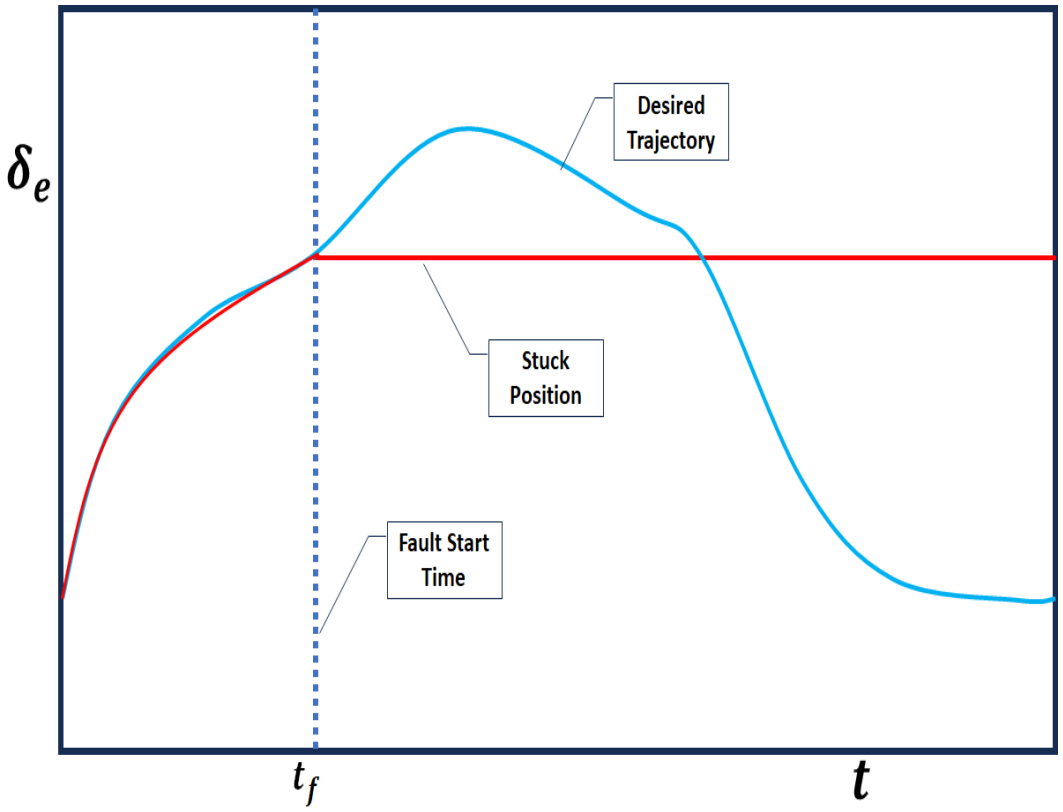

FIGURE 6.5: Stuck Fault

The aerodynamic forces, which are the main cause of temporary faults and are mostly encountered while performing high maneuvers in fighter aircraft, are not guaranteed to be fixed throughout the maneuver. The environmental effects may change in nanoseconds and may lead to a change in load on the actuator. These changing loads are reflected as time-varying faults in actuator performance. So, the actuator, which was assumed to be stuck to a certain fixed position during the fault period, doesn't stick to the fixed reduced limit. But this reduced limit may vary with time, which is termed as time-varying faults. This can be viewed as in Figure 6.7.

6.6 Numerical Example

The controller efficiency is tested in Matlab within the CasADi environment with the closed-loop model as in Figure 6.8. The aircraft data is available in non-linear form, and all necessary parameters are available in Table 6.1. $\boldsymbol{x}_{ref}$ is the desired

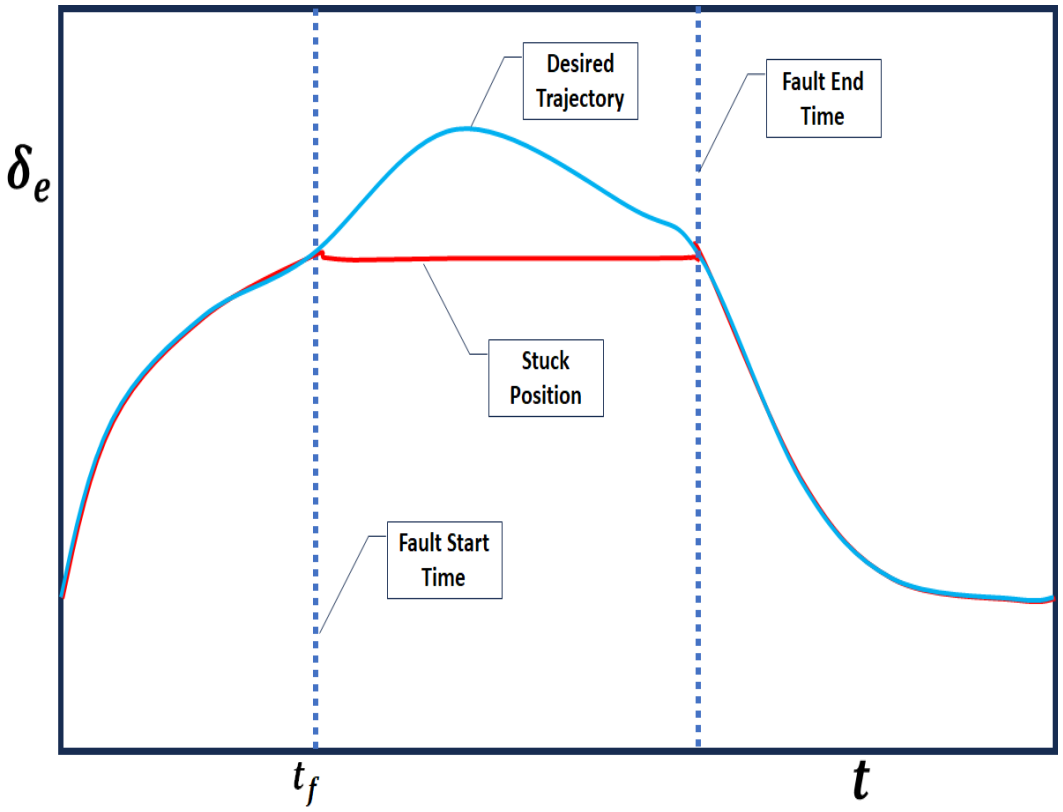

FIGURE 6.6: Stall Load

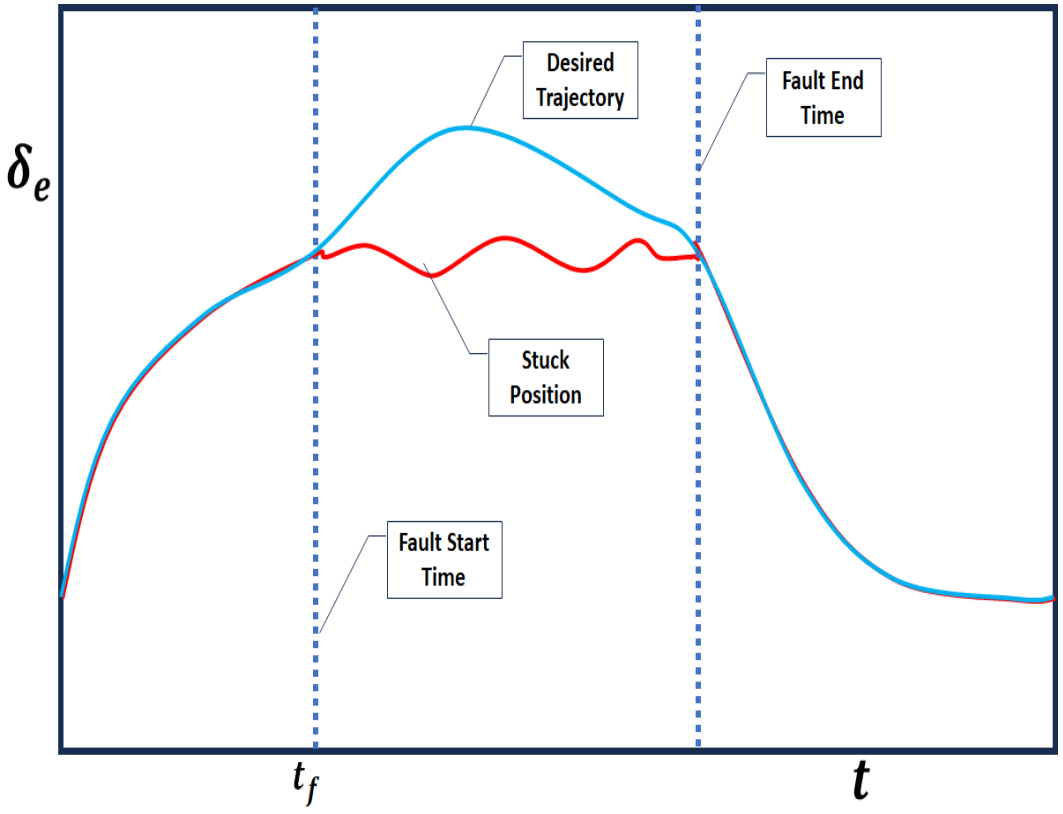

FIGURE 6.7: Time-Varying Stall Load

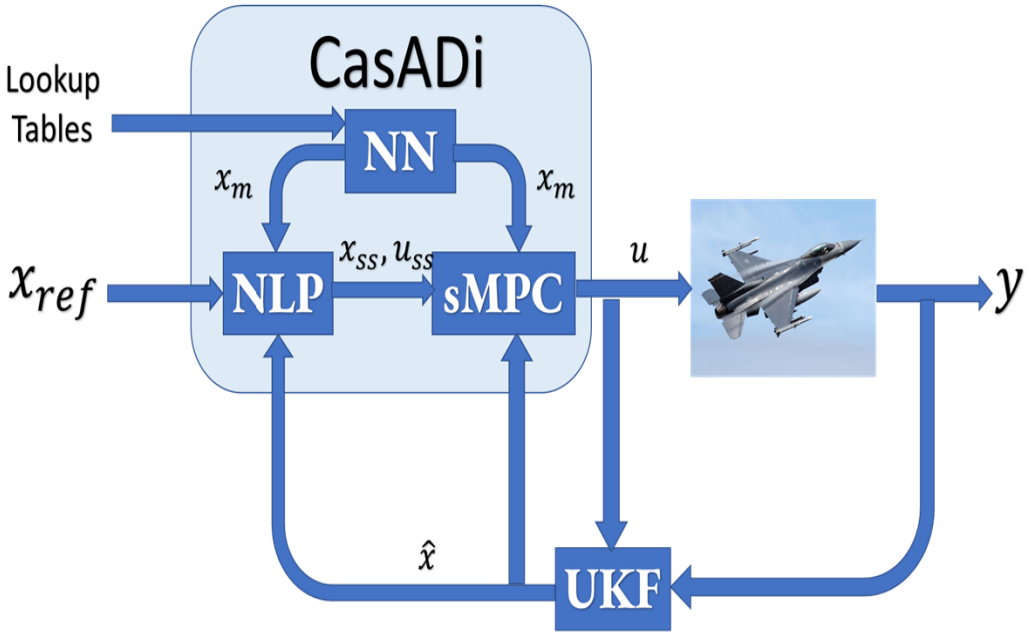

FIGURE 6.8: NN-based OSMPC Closed Loop Block Diagram

state with the specification obtained from the Herbst-type maneuver mentioned earlier. In effect, the reference is to track the angle of attack, roll angle, and side-slip angle.

The NN block is the neural network block. NN block approximates the function for different aerodynamic coefficients, which will be used for the non-linear model of the aircraft in the CasADi environment. This model will be further used in controller design.

NLP is the non-linear solver that is used to obtain a secondary reference for removing the steady-state error. The NLP block receives inputs from the NN block, reference input, and estimated value of state $\boldsymbol{x}$ from the estimator.

UKF is the Unscented Kalman Filter used to estimate the non-linear states of the system from the Aircraft's outputs and inputs. The estimated states have been fed to the NLP and SMPC block. SMPC is the heart of the controller, which is Scenario-based Model Predictive Control. The SMPC block gets the information of estimated state, secondary reference, and measured states from NN and based on that information, gives the input required by the aircraft to perform the maneuver. The block is also responsible for fault tolerance during complex maneuvers. This block forces the controller to work under reduced limits if necessary.

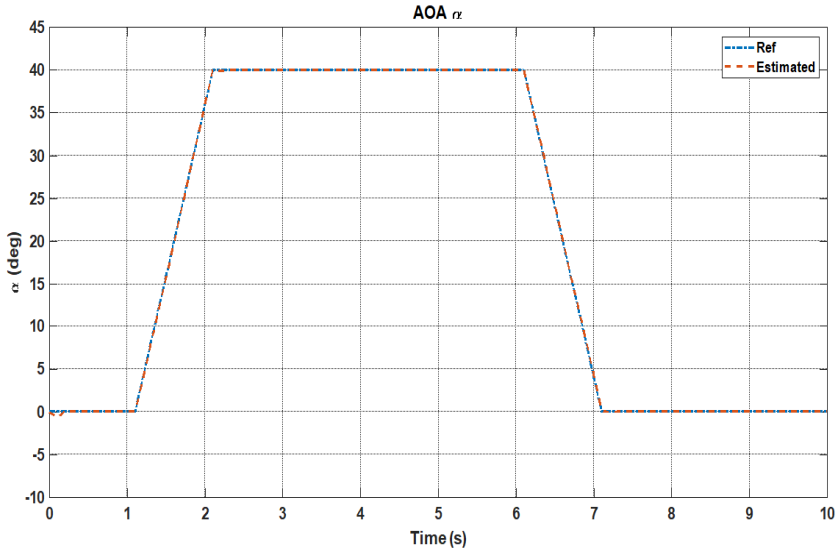

FIGURE 6.9: Angle of Attack tracking using OSMPC

In this chapter, a Herbst-type maneuver is performed. That means the aircraft performs longitudinal as well as lateral motion. An angle of attack tracking is shown in Figure 6.9. The results show that the aircraft follows the desired reference trajectory required to perform the desired maneuver quite efficiently. The tracking is quite smooth, which, in effect, gives minor overshoots and zero steady-state errors. The aircraft quickly follows the reference command with very small rise time, settling time, and overshoot as shown below.

$$\begin{aligned} t_r &= 0.10s \\ t_s &= 0.20s \\ M_p &= 0.00\% \end{aligned}$$

The lateral motion i.e., roll rate has also been controlled efficiently. There are some jerks in tracking lateral motion, but the overall performance is good. The tracking results for roll rate are shown in Figure 6.10. For this certain type of maneuver, the side-slip angle should be zero throughout the maneuver. The result for this controller is shown in Figure 6.11. This result indicates that the controller efficiently controls the side-slip motion. The controller is also able to control the speed of the aircraft as indicated by Figure 6.12. While performing the maneuver, it's been observed that a minor reduction in speed. The speed then increases at

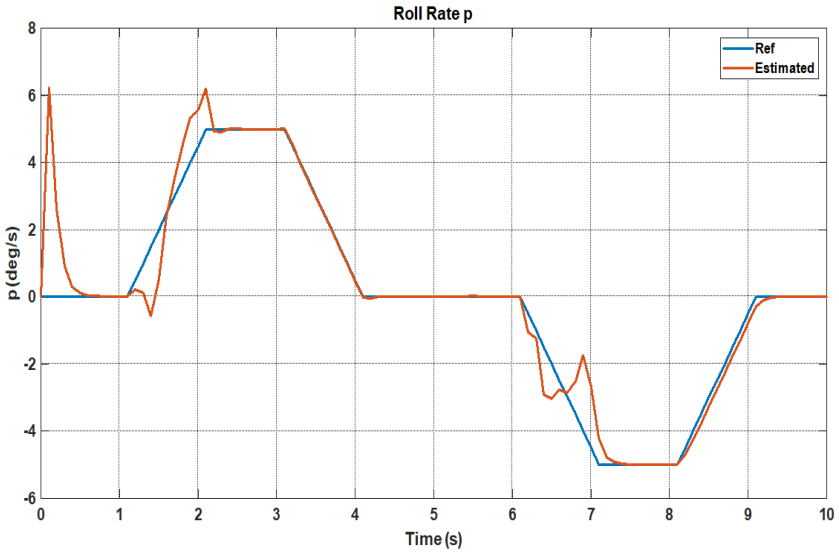

FIGURE 6.10: Roll Rate Response using OSMPC

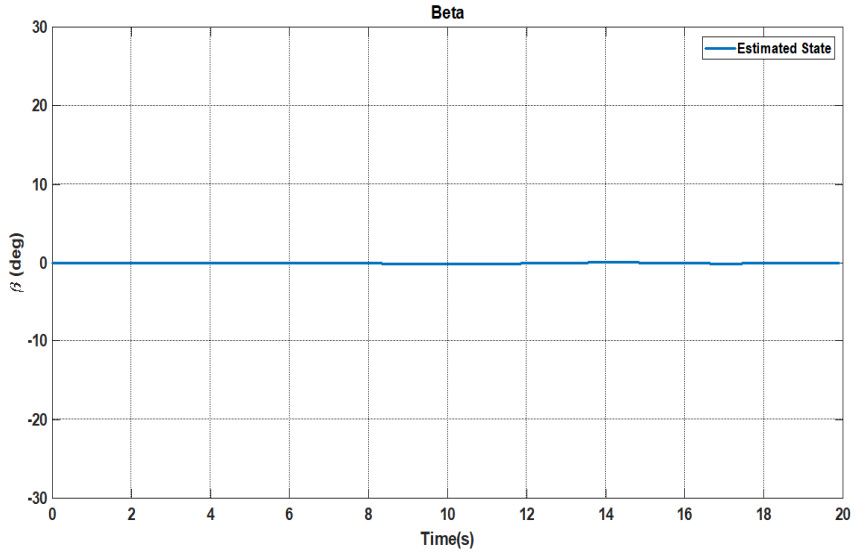

FIGURE 6.11: Side Slip Angle Response using OSMPC

the end of the maneuver. The pitch rate and yaw rate changes have been shown in Figure 6.13 and Figure 6.14, respectively.

The control inputs required to perform the desired maneuver have been shown in Figure 6.15 to Figure 6.18. All of the control surfaces remain in their constraints, yet give the best tracking in the presence of uncertainties.

As discussed earlier, the aircraft suffers from high aerodynamic forces while performing complex maneuvers. In these scenarios, the aircraft works under reduced control limits, and the controller has to redefine the control limits for efficient

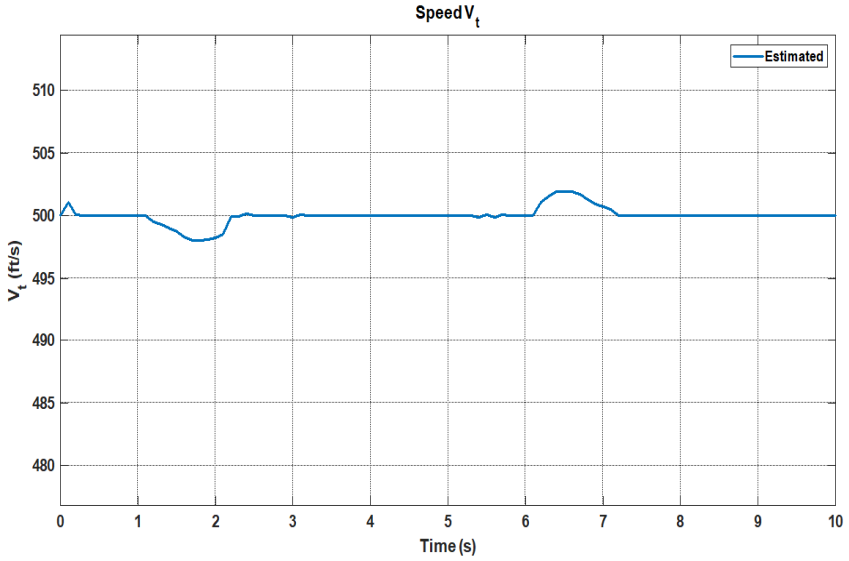

FIGURE 6.12: Speed Response using OSMPC

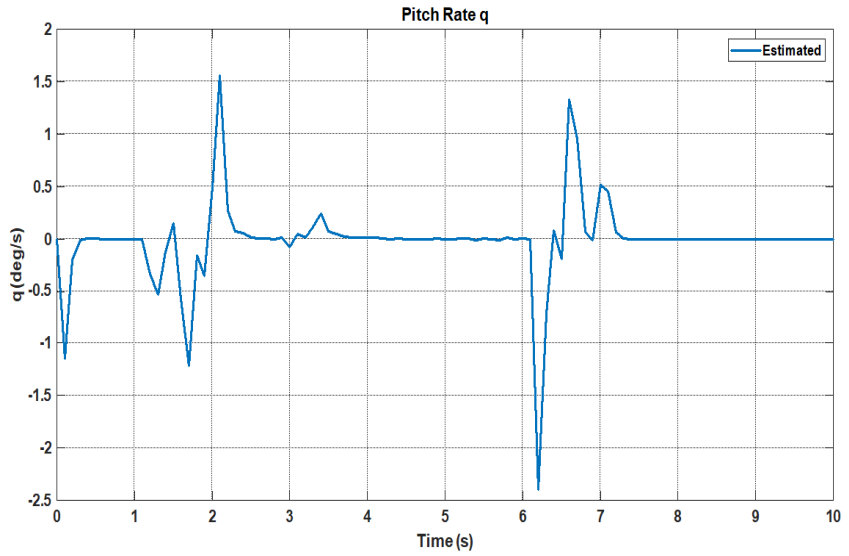

FIGURE 6.13: Pitch Rate Response using OSMPC

control. The controller has also been tested against faulty scenarios in which the controller has to work under reduced control limits because of high aerodynamic forces. In simulations, a saturation block is implemented on the elevator input, for which the saturation limits vary with time. In effect the control surface deflection is reduced by the saturation block. The controller adjusts the control limits seamlessly and operates under reduced limits in these scenarios, and the results remain unaffected. This behavior indicate that the controller shows robust behavior against faulty or degraded conditions.

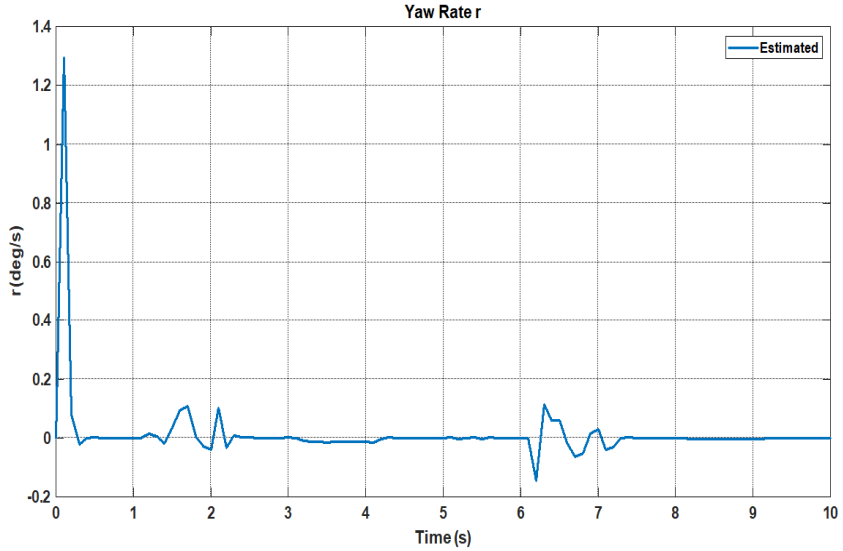

FIGURE 6.14: Yaw Rate Response using OSMPC

The results of the proposed controllers are compared with the results of [105]. In [105], a high angle of attack maneuver has been performed on an F-16 fighter aircraft with the thrust vectoring technique. In order to deal with non-linearities and uncertainties of the system, a Finite Time Extended State Observer has been used with Super Twisting Sliding Mode Control. A Herbst-type maneuver has been performed to validate the controller's efficiency. The simulation results indicate quite a good performance, giving performance parameters as

$$\begin{aligned} t_r &= 1.70s \\ t_s &= 2.50s \\ M_p &= 0.00\% \end{aligned}$$

Although thrust vectoring has been applied in the referenced work, our proposed scheme outperforms for the same type of maneuver. The performance metrics are much better in the proposed work.

6.7 Complexity Analysis

To compare the computation time of different control schemes. The computation time is obtained for the simulation time of 10 sec conducted in Matlab 2024 on

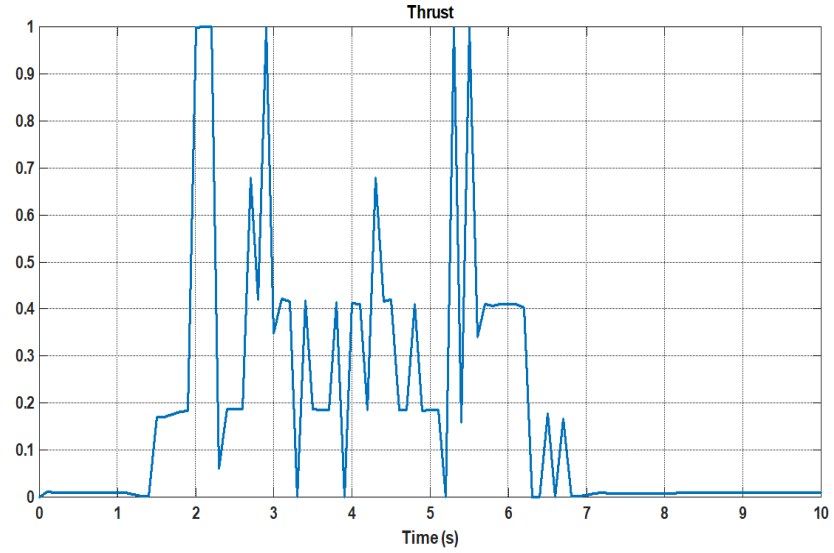

FIGURE 6.15: Thrust Input using OSMPC

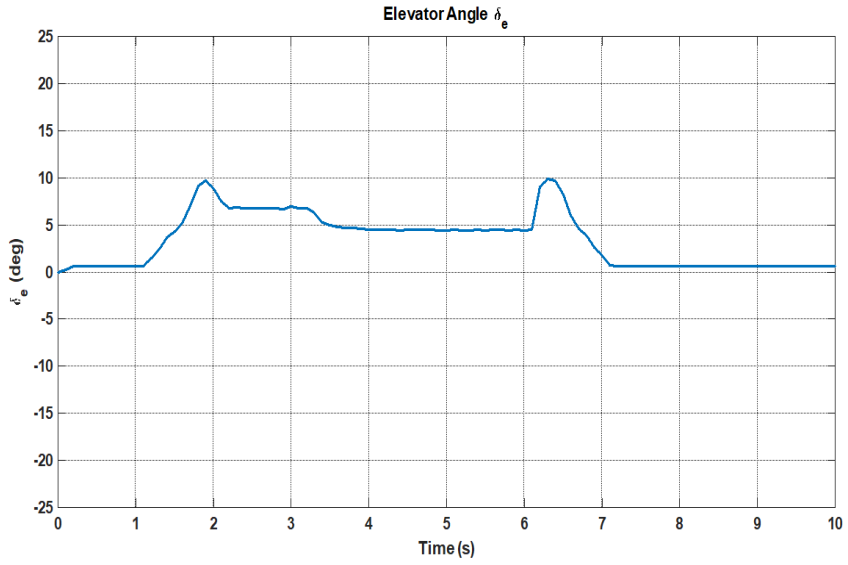

FIGURE 6.16: Elevator Angle using OSMPC

a Dell Laptop with a 13th Gen Core i7-1355U processor (1.7GHz) and 16GB of RAM. The computation time of SMPC, is higher than the computationally efficient NMPC (discussed in Chapter 5) but less than standard linear MPC. This increase on computations is mainly due to the fact, that in SMPC the number of states and number of inputs increase with the number of scenarios. Thus, giving a good performance at the cost of more computation time. The computation time of different MPC schemes is summarized in Figure 6.19 showing that LMPC takes more time than the computationally efficient NMPC and SMPC.

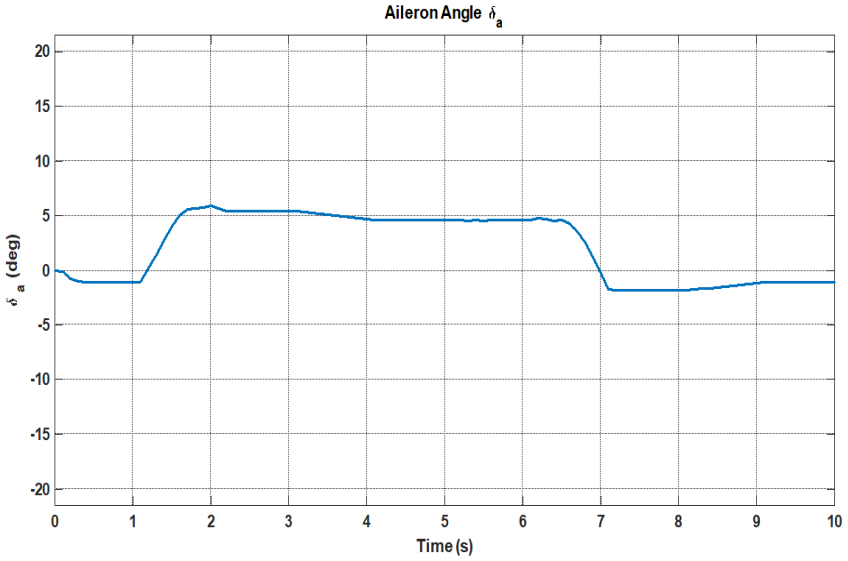

FIGURE 6.17: Aileron Angle using OSMPC

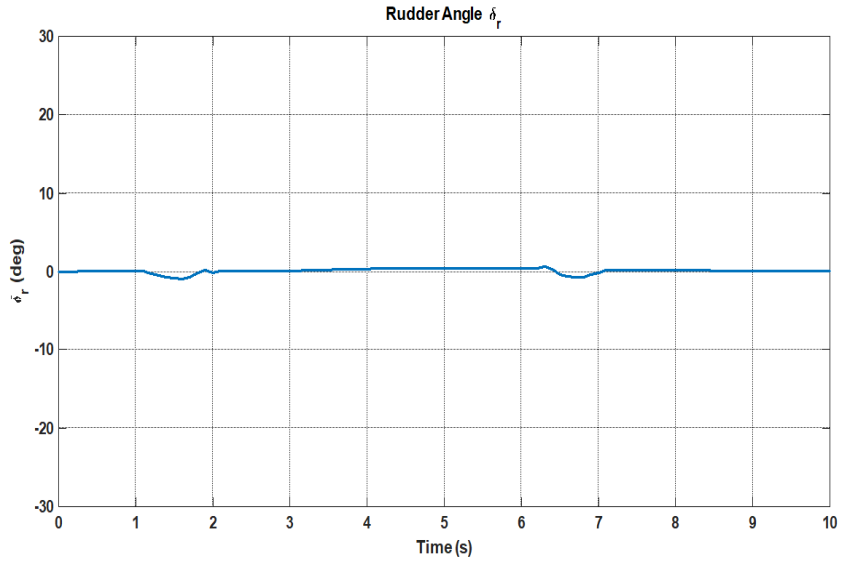

FIGURE 6.18: Rudder Angle using OSMPC

6.8 Conclusion

This chapter provides a computationally efficient implementation approach for OSMPC over fast dynamic systems i.e., fighter aircraft. The proposed approach also caters to uncertainties in the system and environmental disturbances. In effect, it caters to faults due to high aerodynamic forces on the control surfaces. The computational complexity is reduced by the NN-based system's approximation and then symbolic simplification of the controller formulation offline in CasADi. The

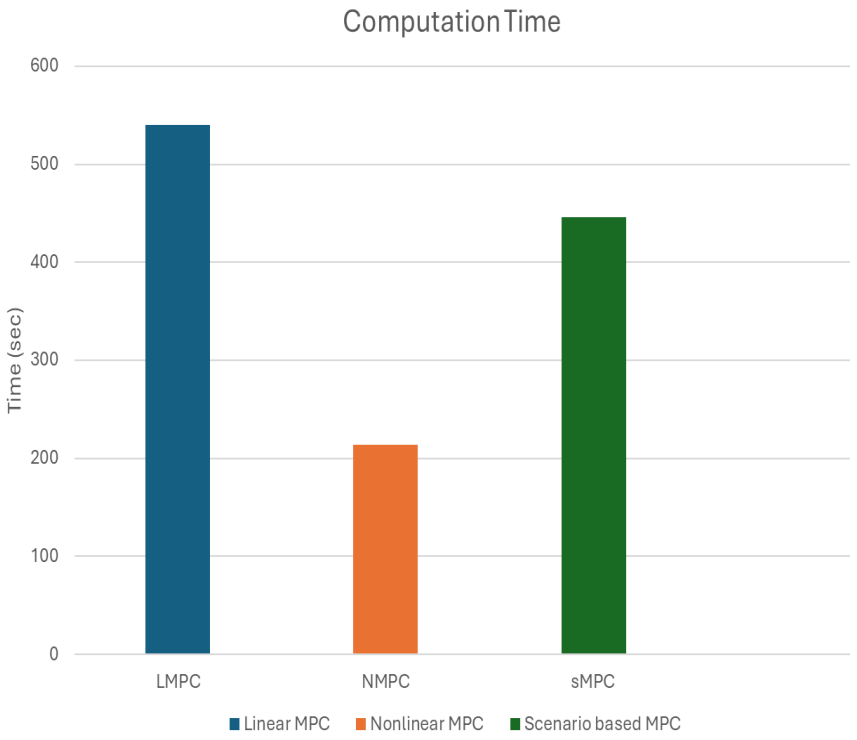

FIGURE 6.19: Computation Time

resulting simplified controller can then be optimized online for current estimates. A scenario-based MPC approach is used for controller implementation, which incorporates offset-free MPC for removing the steady-state error. A UKF-based state estimation provides optimal state estimation in the presence of uncertainties and disturbances.

This research considers only the uncertainty in approximating the aerodynamic coefficients, however, external disturbances and other uncertainties have not been considered in this research. One future direction is to test the controller under environmental disturbances.

This research considers only the Herbst-type maneuver for validation of the controller. This research assumes that if this type of maneuver can be performed efficiently by the controller, the other maneuver can also be performed efficiently. Thus those maneuvers can also be checked in the future by using this technique. The future direction for this could be the application of other fast dynamic systems i.e., UAVs, etc. Another direction could be the use of other methods for reducing the computational complexity of MPC along with this method.

Chapter 7

Conclusion and Future Work

7.1 Conclusion

This research is intended to find a computationally efficient solution for the control of modern fighter aircraft. The aim is to control fast maneuvers during dogfights. The reason for choosing the fast maneuvers for fighter aircraft is that it's hard to control. The other application won't be tough for a controller model that can control these types of maneuvers for fast aircraft. Considering this type of application, different controllers have been reviewed and tested for the fighter aircraft. The research starts with the LQR application for the fighter, which was followed by an H_∞ controller. The results of both of these controllers were compared with those of the MPC controller. The research shows that Model Predictive Control (MPC) is the best controller available. The MPC control model provides constraint satisfaction and excellent performance.

The aircraft chosen for this research is the F-16. F-16 is one of the modern fighter aircraft whose data is non-confidential and openly available. Thus the fast maneuver control of the F-16 using MPC is the main focus of the research. One of the challenges while performing the fast maneuvers is a stall load (temporary fault) in which the control surfaces can't achieve full deflection. Previous research in this area considers only fixed faults, i.e., the control limit reduces to a certain fixed limit. However, this research considers time-varying faults while performing fast maneuvers, and a new way of detecting and controlling these time-varying

faults through MPC has been discussed. The results of the new time-varying fault detection algorithm have been verified on a linear model of the F-16 for high angle of attack maneuver control. The results show that this new method improves performance as compared to the old techniques.

Considering MPC as the best controller, this research focuses on the computationally efficient ways of applying MPC for the fast maneuver control of modern fighter aircraft. The main problem with MPC is high computations, which is the reason for its application limitations. Further research is needed to find different ways of applying MPC with reduced computations. Among many available computation reduction techniques for MPC, one is move-blocking MPC. This technique fixes a certain number of inputs in a receding horizon fashion while finding the optimal control input using a blocking matrix. Among the different available blocking strategies for MPC, none gives optimal results. In Chapter 4, an LQR-based algorithm has been proposed for finding the optimal blocking matrix. This algorithm is based on the optimal behavior of the LQR controller. This method first finds the optimal controller using LQR and the optimal sequence offline. Based on this optimal sequence, it then fixes the number of inputs if the difference between two consecutive inputs is not too much. This method has been tested against Cessna aircraft. However, this method is not beneficial for fast maneuver control of fast aircraft. Besides, using linear MPC for these fast applications needs more computations because of the linearization of the model at each sampling interval (in mSec), other than solving large MPC problems.

The non-linear MPC performs better than linear MPC because it handles the nonlinearities of the system. However, non-linear MPC is more computationally complex than linear MPC. In the next research, a new way of implementing non-linear MPC has been shown for reduce computation time and better performance. This method involves solving the MPC problem offline using symbolic simplification by a toolbox named CasADi and NN where the approximation is needed. This method uses simplified functions instead of solving big matrices in the online computation of the MPC controller. This method has been modified using offset-free MPC to reduce steady-state error. This method has been tested for

TABLE 7.1: Performance Comparison of different MPC schemes

	LMPC	**NMPC**	**SMPC**	**STSMC[105]**
t_r sec	1.2	1.0	0.1	1.7
t_s sec	2.2	2.5	0.2	2.5
$\%OS$	3.3	8.0	0.0	0.0

high angle of attack maneuver control of the non-linear F-16 model. The result shows good performance for this scenario.

The next step is to improve performance in the presence of uncertainties and time-varying external disturbances while performing an even more complex Herbst-type maneuver. This maneuver uses both longitudinal as well as lateral motion control. In this part, the MPC model has been replaced by scenario-based MPC, which is further solved symbolically using CasADi and NN and modified by offset-free MPC. This method considers different levels of uncertainties in the input and state and solves the MPC problem using these scenarios. This method increases computational complexity but performs better in the presence of uncertainties and external disturbances. A Herbst-type maneuver has been performed quite efficiently for F-16 aircraft by using this new technique. The results validate the performance at the cost of increased computations.

The performance of different controllers, i.e., linear MPC, Non-linear MPC, and Scenario-based MPC, has been compared with STSMC. The performance summary is in Table 7.1. The performance comparison indicates that SMPC performs much better as compared to other controllers. The rise time, settling time, and overshoot are much better for SMPC as compared to LMPC, NMPC, and STSMC. The key findings of the research are summarized as:

- The design and implementation of the Time-varying fault detection and fault-tolerant algorithm using linear MPC.
- The design of the LQR-based optimal blocking strategy for reduced computational complexity and optimal performance in linear MPC.
- A NN-based computationally efficient offset-free nonlinear MPC.

- A computationally efficient NN-based offset-free scenario-based MPC.

7.2 Future Work

This research aims to find a computationally efficient solution of MPC for the control of fighter aircraft. However, hardware implementation is not within the scope of this research. One research direction is the HIL-based implementation of the designed controller by using some computationally efficient hardware. This could be done by implementing controller parts on hardware and aircraft models in MATLAB. However, controller selection and programming language play an important role in computational analysis. The controller must be able to perform each iteration of MPC within a specified sampling time. Otherwise, a reduction in performance can be observed.

Another way to implement this controller is to test these controllers on low-cost hardware applications, e.g., a quadcopter. Remotely operated machines, e.g., UAVs, can also be operated by this. However, the controller should be tuned for the particular application. Since MPC is a model-based controller, the controller design requirements might change slightly based on the application.

In addition to hardware implementations of this research, this research can be modified by other computationally efficient methods for MPC. One way could be using move-blocking MPC with the NN-based solution proposed in this research. The other way could be to use a different MPC solver for computational reduction. However, CasADi uses only a few solvers, so only those solvers that are compatible with CasADi can be used. Another approach can be the use of fast MPC.

This research can be further extended to include the use of the designed controller for other phases of aircraft flight, e.g., taxi, takeoff, and landing. However, the model specifications and flight conditions change accordingly.

This research targets the application of fighter aircraft. However, the scope of the proposed schemes in different chapters is not limited to the aircraft only. The proposed control scheme and implementation methods can be applied to any application. The future work includes the application of the proposed control schemes to different applications.

Bibliography

[1] Richard Russell. "Non-linear F-16 Simulation using Simulink and Matlab". 08 2003.

[2] Air Age Media. "Fly the Immelmann Turn". 6.03.2024. URL https://www.modelairplanenews.com/fly-the-immelmann-turn/.

[3] BigWorld Technology. "Mouse Control Masterclass: Split-S". 6.03.2024. URL https://worldofwarplanes.eu/news/how-to-split-s/.

[4] Wei Xu, Haowei Gu, and Fu Zhang. "Acceleration based iterative learning control for pugachev's cobra maneuver with quadrotor tailsitter vtol uavs". *work*, 7:12, 2019.

[5] Wikepedia. "Kulbit". 6.03.2024. URL https://en.wikipedia.org/wiki/Kulbit.

[6] Wei Xu, Haowei Gu, and Fu Zhang. "Acceleration based iterative learning control for pugachev's cobra maneuver with quadrotor tailsitter vtol uavs". *work*, 7:12, 2019.

[7] Tariq Samad. "A survey on industry impact and challenges thereof [technical activities]". *IEEE Control Systems Magazine*, 37(1):17–18, 2017.

[8] SN Deepa and G Sudha. "Longitudinal control of aircraft dynamics based on optimization of PID parameters". *Thermophysics and Aeromechanics*, 23:185–194, 2016.

[9] Pankaj Kumar. "PID Controller Design for Dynamic Motion of an Aircraft". 2019.

[10] Magdy AS Aboelela, Mohamed F Ahmed, and Hassen T Dorrah. "Design of aerospace control systems using fractional PID controller". *Journal of Advanced Research*, 3(3):225–232, 2012.

[11] Mohammad Salem, M Ali, and S Ashtiani. "Robust PID controller design for a modern type aircraft including handling quality evaluation". *American Journal of Aerospace Engineering*, 1(1):7, 2014.

[12] Mohammad Reza Mortazavi and Abolghasem Naghash. "Pitch and flight path controller design for F-16 aircraft using combination of LQR and EA techniques". *Proceedings of the Institution of Mechanical Engineers, Part G: Journal of Aerospace Engineering*, 232(10):1831–1843, 2018.

[13] Adnan Ashraf, Wu Mei, Liu Gaoyuan, Mian Muhammad Kamal, and Ali Mutahir. "Linear feedback and lqr controller design for aircraft pitch control". In *2018 IEEE 4th International Conference on Control Science and Systems Engineering (ICCSSE)*, pages 276–278. IEEE, 2018.

[14] Jyoti Ohri et al. "GA tuned LQR and PID controller for aircraft pitch control". In *2014 IEEE 6th India International Conference on Power Electronics (IICPE)*, pages 1–6. IEEE, 2014.

[15] Y Kanokmedhakul, N Pholdee, S Bureerat, and N Panagant. "LQR Aircraft pitch controller design for handling disturbance using differential evolution". *Journal of Research and Applications in Mechanical Engineering*, 7(2):145–153, 2019.

[16] J Shaji and BR Ashwin. "Pitch control of aircraft using LQR & LQG control". *International Journal of Advanced Research in Electrical, Electronics & Instrumentation Engineering*, 4(8):6981–6987, 2015.

[17] Clay Thompson, Edward Coleman, and James Blight. "Integral LQG controller design for a fighter aircraft". In *Guidance, Navigation and Control Conference*, page 2452, 1987.

[18] Labane Chrif and Zemalache Meguenni Kadda. "Aircraft control system using LQG and LQR controller with optimal estimation-Kalman filter design". *Procedia Engineering*, 80:245–257, 2014.

[19] Hitay Oezbay and Sanjay Garg. "Stable H (infinity) Controller Design for the Longitudinal Dynamics of an Aircraft". 1995.

[20] J Gadewadikar and FL Lewis. "Aircraft flight controller tracking design using H-infinity static output-feedback". *Transactions of the Institute of Measurement and Control*, 28(5):429–440, 2006.

[21] Jafar Zarei, Allahyar Montazeri, Mohmmad Reza Jahed Motlagh, and Javad Poshtan. "Design and comparison of LQG,LTR and H inf controllers for a VSTOL flight control system". *Journal of the Franklin Institute*, 344(5): 577–594, 2007.

[22] Fatima Shoaib, M Arsalan Khawaja, Hafiz Zeeshan Iqbal Khan, M Farooq Haydar, and Jamshed Riaz. "Optimal and robust solutions for longitudinal flight control of a canard-configured high performance aircraft". In *2019 16th International Bhurban Conference on Applied Sciences and Technology (IBCAST)*, pages 483–491. IEEE, 2019.

[23] Nurhana M Rouyan, Renuganth Varatharajoo, Samira Eshghi, Ermira Junita Abdullah, and Shinji Suzuki. "Aircraft pitch control tracking with sliding mode control". *International Journal of Engineering & Technology*, 7(4.13):62–65, 2018.

[24] Christian Tournes and Yuri Schtessel. "Aircraft control using sliding mode control". In *Guidance, Navigation, and Control Conference*, page 3692, 1996.

[25] Ying J Huang. "Sliding mode control design for aircraft control systems". In *Proceedings. The First IEEE Regional Conference on Aerospace Control Systems,*, pages 309–313. IEEE, 1993.

[26] Ekprasit Promtun and Sridhar Seshagiri. "Sliding mode control of pitch-rate of an F-16 aircraft". *IFAC Proceedings Volumes*, 41(2):1099–1104, 2008.

[27] Sridhar Seshagiri and Ekprasit Promtun. "Sliding mode control of F-16 longitudinal dynamics". In *2008 American Control Conference*, pages 1770–1775. IEEE, 2008.

[28] Erfan Shojaei Barjuei and Jesus Ortiz. "A comprehensive performance comparison of linear quadratic regulator (LQR) controller, model predictive controller (MPC), H inf loop shaping and u-synthesis on spatial compliant link-manipulators".

[29] Raktim Bhattacharya, Gary J Balas, M Alpay Kaya, and Andy Packard. "Nonlinear receding horizon control of an F-16 aircraft". *Journal of Guidance, Control, and Dynamics*, 25(5):924–931, 2002.

[30] Tamas Keviczky and Gary J Balas. "Receding horizon control of an F-16 aircraft: A comparative study". *Control Engineering Practice*, 14(9):1023–1033, 2006.

[31] Daniel Simon, Johan Löfberg, and Torkel Glad. "Reference tracking MPC using terminal set scaling". In *2012 IEEE 51st IEEE Conference on Decision and Control (CDC)*, pages 4543–4548. IEEE, 2012.

[32] Daniel Simon, Johan Löfberg, and Torkel Glad. "Reference tracking MPC using dynamic terminal set transformation". *IEEE Transactions on Automatic Control*, 59(10):2790–2795, 2014.

[33] James W Fuller, Aditya Kumar, and Richard C Millar. "Adaptive model based control of aircraft propulsion systems: status and outlook for naval aviation applications". In *Turbo Expo: Power for Land, Sea, and Air*, volume 42371, pages 507–513, 2006.

[34] Richard S Russell and F Non-Linear. "Simulation using Simulink and MATLAB". *University of Minnesota*, 1, 16.

[35] "Supercomputer". *Wikipedia,https://en.wikipedia.org/wiki/Supercomputer*, 07,07 2020. URL https://en.wikipedia.org/wiki/Supercomputer.

[36] Jianglin Lan. "Efficient model predictive control for nonlinear systems modelled by deep neural networks". *arXiv preprint arXiv:2405.10372*, 2024.

[37] Mohammad Alsalti, Manuel Barkey, Victor G Lopez, and Matthias A Müller. "Sample-and computationally efficient data-driven predictive control". In *2024 European Control Conference (ECC)*, pages 84–89. IEEE, 2024.

[38] Hendrik Alsmeier, Anton Savchenko, and Rolf Findeisen. "Neural Horizon Model Predictive Control-Increasing Computational Efficiency with Neural Networks". In *2024 American Control Conference (ACC)*, pages 1646–1651. IEEE, 2024.

[39] Alberto Leva, Simone Formentin, and Silvano Seva. "Overlapping-Horizon MPC: A Novel Approach to Computational Constraints in Real-Time Predictive Control". In *Third Workshop on Next Generation Real-Time Embedded Systems (NG-RES 2022)*. Schloss Dagstuhl-Leibniz-Zentrum für Informatik, 2022.

[40] Eivind Bøhn, Sebastien Gros, Signe Moe, and Tor Arne Johansen. "Optimization of the model predictive control meta-parameters through reinforcement learning". *Engineering Applications of Artificial Intelligence*, 123: 106211, 2023.

[41] Dakota Hamilton, Loraine Navarro, Dionysios Aliprantis, Steven Pekarek, and Greg Zweigle. "Linearized-Trajectory Model-Predictive Controller for Improving Microgrid Short-Term Stability with Real-Time Implementation". *IEEE Transactions on Industry Applications*, 2023.

[42] Daniel Tabas and Baosen Zhang. "Safe and efficient model predictive control using neural networks: An interior point approach". In *2022 IEEE 61st Conference on Decision and Control (CDC)*, pages 1142–1147. IEEE, 2022.

[43] Xia Pan, Xiaowei Chen, Qingyu Zhang, and Nannan Li. "Model Predictive Control: A Reinforcement Learning-based Approach". In *Journal of Physics: Conference Series*, volume 2203, page 012058. IOP Publishing, 2022.

[44] Fateme Bakhshande, Mark Spiller, Yen-Lin King, and Dirk Söffker. "Computationally efficient model predictive control for real time implementation experimentally applied on a hydraulic differential cylinder". *IFAC-PapersOnLine*, 53(2):8979–8984, 2020.

[45] Raphael Cagienard, Pascal Grieder, Eric C Kerrigan, and Manfred Morari. "Move blocking strategies in receding horizon control". *Journal of Process Control*, 17(6):563–570, 2007.

[46] Rohan C Shekhar and Jan M Maciejowski. "Robust variable horizon MPC with move blocking". *Systems & Control Letters*, 61(4):587–594, 2012.

[47] Rohan C Shekhar and Chris Manzie. "Optimal move blocking strategies for model predictive control". *Automatica*, 61:27–34, 2015.

[48] Tim Schwickart, Holger Voos, Mohamed Darouach, and Souad Bezzaoucha. "A flexible move blocking strategy to speed up model-predictive control while retaining a high tracking performance". In *2016 European Control Conference (ECC)*, pages 764–769. IEEE, 2016.

[49] Sang Hwan Son, Tae Hoon Oh, Jong Woo Kim, and Jong Min Lee. "Move blocked model predictive control with improved optimality using semi-explicit approach for applying time-varying blocking structure". *Journal of Process Control*, 92:50–61, 2020.

[50] Michael V Cook. *"Flight dynamics principles: a linear systems approach to aircraft stability and control"*. Butterworth-Heinemann, 2012.

[51] Brian L Stevens, Frank L Lewis, and Eric N Johnson. *"Aircraft control and simulation: dynamics, controls design, and autonomous systems"*. John Wiley & Sons, 2015.

[52] Junjie Liu, Zengqiang Chen, Mingwei Sun, and Qinglin Sun. "Practical coupling rejection control for Herbst maneuver with thrust vector". *Journal of Aircraft*, 56(4):1726–1734, 2019.

[53] Junjie Liu, Mingwei Sun, Zengqiang Chen, and Qinglin Sun. "Super-twisting sliding mode control for aircraft at high angle of attack based on finite-time extended state observer". *Nonlinear Dynamics*, 99(4):2785–2799, 2020.

[54] Jérôme Cieslak, David Henry, Ali Zolghadri, and Philippe Goupil. "Development of an active fault-tolerant flight control strategy". *Journal of guidance, control, and dynamics*, 31(1):135–147, 2008.

[55] Youmin Zhang and Jin Jiang. "Fault tolerant control system design with explicit consideration of performance degradation". *IEEE Transactions on Aerospace and Electronic Systems*, 39(3):838–848, 2003.

[56] Kumar Pakki Bharani Chandra, Lejun Chen, Halim Alwi, and Christopher Edwards. "Actuator faults and blow-down limit detection, and fault tolerant control for the RECONFIGURE benchmark problem". In *2016 IEEE Conference on Control Applications (CCA)*, pages 1544–1549. IEEE, 2016.

[57] Philippe Goupil, Josep Boada-Bauxell, Andres Marcos, Emmanuel Cortet, Murray Kerr, and Hugo Costa. "AIRBUS efforts towards advanced real-time fault diagnosis and fault tolerant control". *IFAC Proceedings Volumes*, 47 (3):3471–3476, 2014.

[58] Christopher Edwards, Thomas Lombaerts, Hafid Smaili, et al. "Fault tolerant flight control". *Lecture notes in control and information sciences*, 399: 1–560, 2010.

[59] Xiang Yu, Zhixiang Liu, and Youmin Zhang. "Fault-tolerant flight control design with finite-time adaptation under actuator stuck failures". *IEEE Transactions on Control Systems Technology*, 25(4):1431–1440, 2016.

[60] Jan M Maciejowski and Colin N Jones. "MPC fault-tolerant flight control case study: Flight 1862". *IFAC Proceedings Volumes*, 36(5):119–124, 2003.

[61] Kumar Pakki Bharani Chandra, Lejun Chen, Halim Alwi, and Christopher Edwards. "Actuator faults and blow-down limit detection, and fault tolerant control for the RECONFIGURE benchmark problem". In *2016 IEEE Conference on Control Applications (CCA)*, pages 1544–1549. IEEE, 2016.

[62] Fabio A De Almeida and Dirk Leißling. "Fault-tolerant model predictive control with flight-test results". *Journal of Guidance, Control, and Dynamics*, 33(2):363–375, 2010.

[63] MM Kale and AJ Chipperfield. "Stabilized MPC formulations for robust reconfigurable flight control". *Control Engineering Practice*, 13(6):771–788, 2005.

[64] Jiann-Shiun Lew. "Robust predictive control for structures under damage condition". *Journal of Guidance, Control, and Dynamics*, 36(6):1824–1829, 2013.

[65] Jan M Maciejowski. "The implicit daisy-chaining property of constrained predictive control". 1998.

[66] Florin Stoican and Sorin Olaru. *"Set-theoretic fault-tolerant control in multisensor systems"*. John Wiley & Sons, 2013.

[67] Alain Yetendje, Maria M Seron, and José A De Doná. "Robust multiactuator fault-tolerant MPC design for constrained systems". *International Journal of Robust and Nonlinear Control*, 23(16):1828–1845, 2013.

[68] Laura Ferranti, Yiming Wan, and Tamas Keviczky. "Fault-tolerant reference generation for model predictive control with active diagnosis of elevator jamming faults". *International Journal of Robust and Nonlinear Control*, 29(16): 5412–5428, 2019.

[69] Richard S Russell. "Non-linear F-16 simulation using Simulink and Matlab". *University of Minnesota*, 2003.

[70] Liuping Wang. *"Model predictive control system design and implementation using MATLAB®"*. Springer Science & Business Media, 2009.

[71] J. M. MacIejowski. *"Predictive Control With Constraints"*. Pearson education, 2002.

[72] Pankaj Kumar et al. "PID Controller Design for Dynamic Motion of an Aircraft". 2019.

[73] Mohammad Reza Mortazavi and Abolghasem Naghash. "Pitch and flight path controller design for F-16 aircraft using combination of LQR and EA techniques". *Proceedings of the Institution of Mechanical Engineers, Part G: Journal of Aerospace Engineering*, 232(10):1831–1843, 2018.

[74] Adnan Ashraf, Wu Mei, Liu Gaoyuan, Mian Muhammad Kamal, and Ali Mutahir. "Linear feedback and lqr controller design for aircraft pitch control". In *2018 IEEE 4th International Conference on Control Science and Systems Engineering (ICCSSE)*, pages 276–278. IEEE, 2018.

[75] J Shaji and RB Aswin. "Pitch control of aircraft using LQR & LQG control". *International Journal of Advanced Research in Electrical, Electronics and Instrumentation Engineering, IJAREEIE,(An ISO 3297: 2007 Certified Organisation)*, 4(8), 2015.

[76] J Gadewadikar and FL Lewis. "Aircraft flight controller tracking design using H-infinity static output-feedback". *Transactions of the Institute of Measurement and Control*, 28(5):429–440, 2006.

[77] Jafar Zarei, Allahyar Montazeri, Mohmmad Reza Jahed Motlagh, and Javad Poshtan. "Design and comparison of LQG/LTR and H controllers for a VSTOL flight control system". *Journal of the Franklin Institute*, 344(5): 577–594, 2007.

[78] Fatima Shoaib, M Arsalan Khawaja, Hafiz Zeeshan Iqbal Khan, M Farooq Haydar, and Jamshed Riaz. "Optimal and robust solutions for longitudinal flight control of a canard-configured high performance aircraft". In *2019 16th International Bhurban Conference on Applied Sciences and Technology (IBCAST)*, pages 483–491. IEEE, 2019.

[79] Nurhana M Rouyan, Renuganth Varatharajoo, Samira Eshghi, Ermira Junita Abdullah, and Shinji Suzuki. "Aircraft pitch control tracking with sliding mode control". *International Journal of Engineering & Technology*, 7(4.13):62–65, 2018.

[80] Erfan Shojaei Barjuei and Jesús Ortiz. "A comprehensive performance comparison of linear quadratic regulator (LQR) controller, model predictive controller (MPC), H loop shaping and μ-synthesis on spatial compliant link-manipulators". *International Journal of Dynamics and Control*, 9(1): 121–140, 2021.

[81] Tariq Samad. "A survey on industry impact and challenges thereof [technical activities]". *IEEE Control Systems Magazine*, 37(1):17–18, 2017.

[82] Muhammad Faheem Manzoor, Yasir Awais Butt, Fazal-ur Rehman, and Aamer Iqbal Bhatti. "Fault-Tolerant Control of Fighter Aircraft using MPC". In *2022 19th International Bhurban Conference on Applied Sciences and Technology (IBCAST)*, pages 598–602. IEEE, 2022.

[83] Tamás Keviczky and Gary J Balas. "Receding horizon control of an F-16 aircraft: A comparative study". *Control Engineering Practice*, 14(9):1023–1033, 2006.

[84] Runqi Chai, Antonios Tsourdos, Huijun Gao, Senchun Chai, and Yuanqing Xia. "Attitude tracking control for reentry vehicles using centralised robust model predictive control". *Automatica*, 145:110561, 2022.

[85] Runqi Chai, Antonios Tsourdos, Huijun Gao, Yuanqing Xia, and Senchun Chai. "Dual-loop tube-based robust model predictive attitude tracking control for spacecraft with system constraints and additive disturbances". *IEEE Transactions on Industrial Electronics*, 69(4):4022–4033, 2021.

[86] In David M. Prett and Carlos E. García, editors, *"Fundamental Process Control"*, Butterworths Series in Chemical Engineering, pages 233–241. Butterworth-Heinemann, 1988. ISBN 978-0-409-90082-8. doi:

https://doi.org/10.1016/B978-0-409-90082-8.50020-5. URL https://www.sciencedirect.com/science/article/pii/B9780409900828500205.

[87] Kenneth R Muske and Thomas A Badgwell. "Disturbance modeling for offset-free linear model predictive control". *Journal of Process Control*, 12 (5):617–632, 2002.

[88] Urban Maeder, Francesco Borrelli, and Manfred Morari. "Linear offset-free model predictive control". *Automatica*, 45(10):2214–2222, 2009.

[89] Giulio Betti, Marcello Farina, and Riccardo Scattolini. "An MPC algorithm for offset-free tracking of constant reference signals". In *2012 IEEE 51st IEEE conference on decision and control (CDC)*, pages 5182–5187. IEEE, 2012.

[90] Giulio Betti, Marcello Farina, and Riccardo Scattolini. "A robust MPC algorithm for offset-free tracking of constant reference signals". *IEEE Transactions on Automatic Control*, 58(9):2394–2400, 2013.

[91] Manfred Morari and Urban Maeder. "Nonlinear offset-free model predictive control". *Automatica*, 48(9):2059–2067, 2012.

[92] Corey Montella. "The Kalman filter and related algorithms: A literature review". *Res. Gate*, pages 1–17, 2011.

[93] Eric A Wan and Rudolph Van Der Merwe. "The unscented Kalman filter for nonlinear estimation". In *Proceedings of the IEEE 2000 Adaptive Systems for Signal Processing, Communications, and Control Symposium (Cat. No. 00EX373)*, pages 153–158. Ieee, 2000.

[94] Tamás Keviczky and Gary J Balas. "Receding horizon control of an F-16 aircraft: A comparative study". *Control Engineering Practice*, 14(9):1023–1033, 2006.

[95] Xia Pan, Xiaowei Chen, Qingyu Zhang, and Nannan Li. "Model Predictive Control: A Reinforcement Learning-based Approach". In *Journal of Physics: Conference Series*, volume 2203, page 012058. IOP Publishing, 2022.

[96] Joel AE Andersson, Joris Gillis, Greg Horn, James B Rawlings, and Moritz Diehl. "CasADi: a software framework for nonlinear optimization and optimal control". *Mathematical Programming Computation*, 11:1–36, 2019.

[97] Sidra Ghayour Bhatti and Aamer Iqbal Bhatti. "BiLSTM based phase modulation detection of radar emitters". In *2021 CIE International Conference on Radar (Radar)*, pages 3272–3276. IEEE, 2021.

[98] Sidra Ghayour Bhatti and Aamer Iqbal Bhatti. "Radar signals intrapulse modulation recognition using phase-based stft and bilstm". *IEEE Access*, 10:80184–80194, 2022.

[99] SN Deepa and G Sudha. "Longitudinal control of aircraft dynamics based on optimization of PID parameters". *Thermophysics and Aeromechanics*, 23:185–194, 2016.

[100] Brian L Stevens, Frank L Lewis, and Eric N Johnson. *"Aircraft control and simulation: dynamics, controls design, and autonomous systems"*. John Wiley & Sons, 2015.

[101] Shankar Sastry. *"Nonlinear systems: analysis, stability, and control"*, volume 10. Springer Science & Business Media, 2013.

[102] Jean-Jacques E Slotine, Weiping Li, et al. *"Applied nonlinear control"*, volume 199. Prentice hall Englewood Cliffs, NJ, 1991.

[103] Muhammad Faheem Manzoor, Joel A Paulson, Yasir Awais Butt, Fazal-ur Rehman, and Aamer Iqbal Bhatti. "Real-time implementation of nonlinear model predictive control for high angle of attack Maneuvers in fighter aircrafts using deep learning". *Systems Science & Control Engineering*, 12(1): 2342819, 2024.

[104] Angelo D Bonzanini, Joel A Paulson, and Ali Mesbah. "Safe learning-based model predictive control under state-and input-dependent uncertainty using scenario trees". In *2020 59th IEEE Conference on Decision and Control (CDC)*, pages 2448–2454. IEEE, 2020.

[105] Junjie Liu, Mingwei Sun, Zengqiang Chen, and Qinglin Sun. "Super-twisting sliding mode control for aircraft at high angle of attack based on finite-time extended state observer". *Nonlinear Dynamics*, 99:2785–2799, 2020.

Flying & Handling Qualities

Longitudinal Mode

Phugoid Mode Specifications

Table 5: Phugoid Mode

Level 1	$\zeta_p > 0.04$
Level 2	$\zeta_p > 0$
Level 3	$T_{2p} > 55s$

Short Period Mode Specifications

Table 6: Short Period Mode Damping Ratio

	Cat. A & C Flight Phases		Cat. B Flight Phases	
Level	Minimum	Maximum	Minimum	Maximum
1	0.35	1.30	0.30	2.00
2	0.25	2.00	0.20	2.00
3	0.15	no limit	0.15	no limit

Table 7: Short Period Mode ω_n

	Cat. A Phases		Cat. B Phases		Cat. C Phases	
Level	Minimum	Maximum	Minimum	Maximum	Minimum	Maximum
1	0.28	3.60	0.085	3.60	0.16	3.60
	$\omega_n \geq 1.0$				$\omega_n \geq 0.7$	
2	0.16	10	0.038	10	0.096	10
	$\omega_n \geq 0.6$				$\omega_n \geq 0.4$	
3	0.16	no limit	0.038	no limit	0.096	no limit

Lateral Mode

Roll Mode

Table 8: Roll Mode Max. Time Constant

Flight		Level		
Phase Category	Class	1	2	3
A	I,IV	1.0	1.4	no limit
	II,III	1.4	3.0	no limit
B	All	1.4	3.0	10
C	I,II-C,IV	1.0	1.4	no limit
	II-L,III	1.4	3.0	no limit

Spiral Mode

Table 9: Spiral Mode Min. Doubling Time Constant

Flight Phase Category	Level 1	Level 2	Level 3
A & C	12s	8s	4s
B	20s	8s	4s

Dutch Roll Mode

Table 10: Dutch Roll Mode

Level	Phase Category	Class	min ζ_d	min $\zeta_d\omega_{nd}$	min ω_{nd}
	A	I,IV	0.19	0.35	1.0
1		II,III	0.19	0.35	0.4
	B	All	0.08	0.15	0.4
	C	I,II-C,IV	0.08	0.15	1.0
		II-L,III	0.08	0.15	0.4
2	all	all	0.02	0.05	0.4
3	all	all	0.02	no limit	0.04