Nonlinear Model Predictive Control for Cooperative Control and Estimation

by

PENGKAI RU

Presented to the Faculty of the Graduate School of

The University of Texas at Arlington in Partial Fulfillment

of the Requirements

for the Degree of

DOCTOR OF PHILOSOPHY

THE UNIVERSITY OF TEXAS AT ARLINGTON

April, 12 2017

To my mom and dad

for always believing in me.

ACKNOWLEDGEMENTS

This dissertation would not be possible without help and support from many people. First and foremost, I would like to express my gratitude to my advisor, Dr. Kamesh Subbarao, for accepting me as his student, teaching me knowledge that is fundamental for completing my research, and guiding me through my Ph.D. He made me realize that taking a class could be a fun experience even when it is extremely difficult and even when you are constantly sleep-deprived. He rescued me when I was in a deep swamp of despair in research and gave me a direction I could go. I would always be indebted to him for the inspiration and support he has given me.

I would also like to thank members of my commitee: Dr. Atilla Dogan, Dr. Panayiotis Shiakolas, Dr. Andrej Korzeniowski, and Dr. Kent Lawrence. Dr. Atilla Dogan taught me the mathematical rigor I need as a Ph.D. student. And he always gives the best constructive criticism from a unique perspective as a control theoretician. Dr. Panayiotis Shiakolas taught me linear control and got me started as a controls researcher. Dr. Andrej Korzeniowski helped me gain a much deeper understanding of probability and stochastic processes. Dr. Kent Lawrence gave me some valuable feedback for my research after my dissertation proposal and I will never forget his encouragement.

My deepest thanks goes to the administrative staff at the Mechanical and Aerospace Engineering department, especially Debi (and Jef), Lanie, Sally, Flora, Ayesha and Kathy. Thank you for making this place feeling like home and thank you for making this foreign experience wonderful and amazing. To Debi and Jef, thank you for always offering me a couch to sleep and I would always, always remember the days I had when I was there.

My sincerest appreciation goes to my labmates and schoolmates. You made this journey fun and enjoyable. Special thanks to Gus, Alok, Pavan, Laura, Carlos, Ameya, Paul, Shashank, Tracie, Ziad, Roop, Mo, Denish and Kelvin. To Alok and Ameya, thank you for taking care of me when I was sick. To Mo, thank you for helping me with C++ and mock interviews. To Nanda and Sid, thank you for being my friend for the last 5 years.

Finally, I would like to thank my Mom for the last two decades and for all the love and support that she has given me. This degree is as much hers as it is mine.

ABSTRACT

Nonlinear Model Predictive Control for Cooperative Control and Estimation

Pengkai Ru, Ph.D.

The University of Texas at Arlington, 2017

Supervising Professor: Kamesh Subbarao

Recent advances in computational power have made it possible to do expensive online computations for control systems. It is becoming more realistic to perform computationally intensive optimization schemes online on systems that are not intrinsically stable and/or have very small time constants. Being one of the most important optimization based control approaches, model predictive control (MPC) has attracted a lot of interest from the research community due to its natural ability to incorporate constraints into its control formulation.

Linear MPC has been well researched and its stability can be guaranteed in the majority of its application scenarios. However, one issue that still remains with linear MPC is that it completely ignores the system's inherent nonlinearities thus giving a sub-optimal solution. On the other hand, if achievable, nonlinear MPC, would naturally yield a globally optimal solution and take into account all the innate nonlinear characteristics. While an exact solution to a nonlinear MPC problem remains extremely computationally intensive, if not impossible, one might wonder if there is a middle ground between the two. We tried to strike a balance in this dissertation by employing a state representation technique, namely, the state dependent coefficient

(SDC) representation. This new technique would render an improved performance in terms of optimality compared to linear MPC while still keeping the problem tractable. In fact, the computational power required is bounded only by a constant factor of the completely linearized MPC.

The purpose of this research is to provide a theoretical framework for the design of a specific kind of nonlinear MPC controller and its extension into a general cooperative scheme. The controller is designed and implemented on quadcopter systems.

TABLE OF CONTENTS

LIST OF ILLUSTRATIONS

LIST OF TABLES

Executive Summary

Model Predictive Control (MPC) is becoming more and more popular due to its ability to explicitly incorporate constraints into the control formulation. With the advancement of faster and cheaper computers, computationally intensive tasks are no longer as formidable as they used to be. All of this has helped to foster the research community's interest in MPC, especially on its application to nonlinear and/or fast dynamic systems.

The purpose of this dissertation is to provide a theoretical framework for synthesis of a nonlinear MPC scheme for cooperative control and estimation. Research on Linear MPC has sufficiently matured over the years but its nonlinear counterpart still has severe challenges to meet. It is well known that guaranteeing stability for linear MPC can be easily achieved by adding additional constraints on the terminal state. In this dissertation, we showed that similar results can be achieved for nonlinear systems. The key idea is using a technique called State Dependent Coefficient formulation. This formulation transforms a nonlinear system into a pseudo-linear form by produce a system matrix that is dependent on the current state. By using this, we can both use the established results from linear MPC and potentially exploit the benefits of the inherent nonlinearities of the system.

For constrained nonlinear systems, feasibility is one of the main issues that needs to be addressed before anything else. To achieve this, we provide Linear Matrix Inequality (LMI) conditions to check if the control parameters selected are realistic for certain state and input constraints.

Furthermore, it is proved that as long as an appropriate sampling interval is chosen, a sampled-data implementation of aforementioned nonlinear MPC algorithm would be guaranteed stable.

We also extended the nonlinear MPC to a cooperative control framework. Paired with an established consensus algorithm, stability of the formation controller is also shown. Extra constraints were added to avoid collisions between different vehicles during the formation process. A cooperative estimation algorithm is also presented to complement the cooperative control task for scenarios of reconnaissance and surveillance.

All of the above mentioned algorithms are applied to quadcopters in simulation.

CHAPTER 1

Introduction and Dissertation Outline

1.1 Introduction

Model Predictive Control (MPC), also known as Receding Horizon Control (RHC), is a control technique in which the current control action is obtained by optimizing a prespecified objective function over a finite horizon, at each sampling instant, based on the current state and model of the system. The objective is a function of both system state and future inputs. The model is used for predicting the behavior of the system if certain control inputs were to be implemented. After the optimization problem is solved, only the first control in the acquired control action sequence would be applied to the plant. And the same process is repeated at the next sampling instant. The scheme is shown in Fig. (1.1).

1.1.1 Model Predictive Control of Linear and Nonlinear Systems

Linear Model Predictive Control (LMPC) is MPC applied to linear systems. The stability of such a controller is one of the main issues that researchers are most concerned about. To guarantee the closed-loop stability of such controller, many techniques have been proposed and successfully proven. In [1], it is shown that by adding a terminal equality constraint, which is the same as an infinite terminal weighting matrix, the closed-loop system using LMPC is stable. But terminal equality constraint is very restrictive and may render the problem intractable or infeasible. To further relax the constraint, [2, 3] proved that if the terminal weighting matrix satisfies a certain inequality condition, then the closed-loop system is stable. Also [4] showed

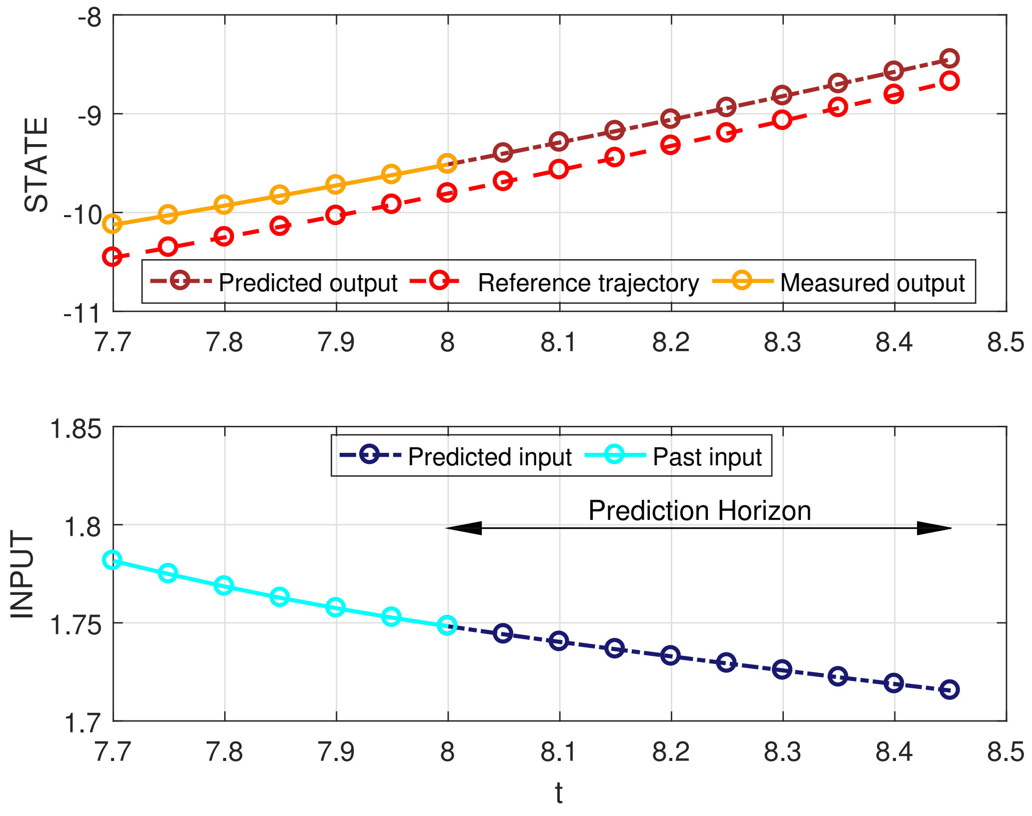


Figure 1.1. Model Predictive Control Scheme.

that such a terminal weighting matrix can be obtained by solving a Linear Matrix Inequality (LMI). But in most control systems, input variables are often limited in a certain range and state values are often bounded. LMPC can explicitly incorporate these constraints into the optimization. Similar to unconstrained systems, stability for constrained LMPC can be proved in the same way together with feasibility conditions. The same terminal equality condition also guarantees stability as long as the system is feasible. [4] proposed an invariant ellipsoid constraint and derived an explicit feasibility condition to guarantee stability. Other method like dual-mode receding horizon control was also proposed by switching control laws based on whether the state is inside or outside a certain ellipsoid [2].

In reality, dynamical systems are often represented by nonlinear differential equations. MPC applied to nonlinear systems is termed as Nonlinear Model Predictive Control (NMPC). NMPC offers a better solution than its linear counterpart in terms of optimality and stability. As with LMPC, numerous approaches have been proposed to guarantee the stability of NMPC controllers. By imposing a terminal equality constraint and using the value function as the Lyapunov function, it is shown that the corresponding NMPC controller is stable [2, 5]. [6] further relaxed the constraint by requiring the state to enter a neighborhood of the origin at the end of the horizon to guarantee closed-loop stability. Once the state entered that neighborhood, the control is switched to a local linear controller to drive the state to the origin. [7, 8] employed a different approach by using a quadratic terminal cost as well as a terminal inequality constraint. [9, 10] used an end point penalty which is the cost incurred if a locally stabilizing linear control law is applied at the end of the time horizon. The linear control law is exponentially stable locally near the origin. [11, 12] achieved a global stabilizing control law by finding a global Control Lyapunov Function (CLF) and including additional state constraints that require the derivative of the CLF along the receding horizon trajectory to be negative and also that the decrease in the value of the CLF be greater than that obtained using the controller derived from CLF. [13] proposed to utilize a special class of CLFs as terminal cost in the receding horizon scheme to guarantee stability. The benefit of this approach is that there is no need to impose terminal equality and/or inequality constraints and it would speed up computation significantly.

1.1.2 MPC Framework for Cooperative UAV Formations

Research on control of multivehicle systems performing cooperative tasks dates back to the late 1980s, initially beginning in the field of mobile robotics [14]. [15] offered a concise definition of "cooperative"as: a collection of N vehicles that are performing a shared task that are dependent on the relationship between the locations of the individual vehicles. And each vehicle is a dynamical system whose position is given by its location in 3-dimensional space. Cooperative control has been gaining interest in the research community because of its wide ranging applications such as formation flight, cooperative classification and surveillance, cooperative attack and rendezvous, distributed aperture observing and air traffic control.

In this dissertation, we only focus on formation control problems. Three general approaches have been proposed in the literature to solve the problem. 1. Consensus based approaches: both continuous [16, 17] and discrete [18–20] convergence algorithms have been proposed and proven to be stable. While the authors proved stability they did not address constraints on the states and inputs of the vehicles and their implication on the stability (See [21, 22] for a detailed survey on this topic); 2. Optimization based approaches: Model Predictive Control (MPC) can directly incorporate states and inputs constraints into the cooperative control problem. In [23], MPC is used for stabilizing three vehicle formations with input constrained dynamics on configuration space $SE(2)$ (See Ref. [24, chapter 4]). The same approach is extended to a more general framework in [25] to do task specification for multiple vehicles. [26] proposed a distributed version of MPC applied to multi-vehicle formation stabilization, by penalizing the deviation of vehicle trajectory from the its open-loop counterpart and enabling sufficiently fast MPC updates, the stability is guaranteed. Similar approach is implemented in [27] by using mixed-integer linear programs and it guarantees collision avoidance and constraint fulfillment. 3. Potential field based approaches: [28,29] used the concept of "virtual leaders" to guide the motion of other vehicles. Asymptotic stability is guaranteed for various schooling and flocking behaviors.

1.1.3 Cooperative Estimation and MPC Framework for Target Tracking

Tracking moving target(s) using sensor network(s) is an important application for military, law enforcement and defense systems, which require a good estimate of the target location. [30, 31] investigated mobile target estimation and tracking using in wireless ad hoc networks. But the sensor networks used are stationary and predetermined. Similar problems have been solved using a vision based approach in [32, 33]. The algorithm employed is based on multi-agent optimization technique to obtain an estimation of the pose of a 3D moving object from 2D vision data. Different variations of Kalman Filter are also used. [34] proposed a gradient-search based decentralized algorithm using mobile nodes as a sensor network to estimate the state of a dynamic target using range only measurements. [35] proposed three distributed Kalman filtering algorithms for sensor networks to estimate the state of a dynamic target while assuming its model is known. [36] employed an adaptive cooperative Kalman filtering technique to measure large-scale environmental fields. [37] used an information-theoretic approach while [38] combined Binary Bayesian Grid Filters (BBGF) with Rapidly-expanding Random Tree (RRT) planner to determine paths to be followed by the group vehicles to track the target.

Here we only consider a sub-problem of sensor network(s), namely cooperative estimation. More specifically, we consider the problem of estimating the location of a moving target in real-time using multiple UAVs and employing control laws to keep them close to the target.

1.2 List of Contributions

- Proposed a novel State Dependent Coefficient Based Model Predictive Control framework.
- Proved stability and convergence of errors for Sampled-Data NMPC based on SDC.
  - K. Subbarao, C. Tule, and P. Ru, Nonlinear model predictive control applied to trajectory tracking for unmanned aerial vehicles, in AIAA Atmospheric Flight Mechanics Conference, no. AIAA 2015-2857, Dallas, TX, June 2015 (Ref. [39]).
  - P. Ru and K. Subbarao, Nonlinear model predictive control for unmanned aerial vehicles, Journal of Aerospace Engineering. [Submitted] (Ref. [40]).
- Proposed a cooperative control formulation based on NMPC and proved its stability.
  - P. Ru, K. Subbarao, Cooperative Control of Unmanned Aerial Vehicles based on Nonlinear Model Predictive Control, AIAA Guidance, Navigation, and Control Conference, AIAA Science and Technology Forum and Exposition 2017, Grapevine, TX, USA, January 2017. (Ref. [41]).
- Proposed a cooperative estimation framework based on EKF and trilateration method.
  - P. Ru, K. Subbarao, Cooperative Estimation of Moving Target Position Using Unmanned Aerial Vehicles, AIAA Information Systems Infotech @ Aerospace, AIAA Science and Technology Forum and Exposition, Grapevine, TX, USA, January 2017. (Ref. [42]).

1.3 Dissertation Outline

This dissertation is organized as follows: In Chapter 2, some preliminary concepts are introduced including stability of dynamical systems, optimization, graph theory, and mathematical model of a quadrotor. Chapter 3 details Model Predictive Control applied to linear systems. Aspects about constraints incorporation and conditions for stability are provided. Chapter 4 introduces State Dependent Coefficient based Nonlinear Model Predictive Control. Aspects about constraints incorporation and conditions for stability are provided. Proof for convergence of errors for its sampled data implementation is also given. Both linear MPC and nonlinear MPC are evaluated in representative simulations and their results are discussed. In Chapter 5, a cooperative control framework based on the aforementioned nonlinear MPC is presented. Combined with an established consensus algorithm, conditions for stability are given. A new cooperative estimation algorithm is presented in Chapter 6 by combining extended Kalman filter and trilateration. Finally, in Chapter 7, concluding remarks are stated. Fig. (1.2) shows a graphical overview of this dissertation with shaded block representing the work covered.

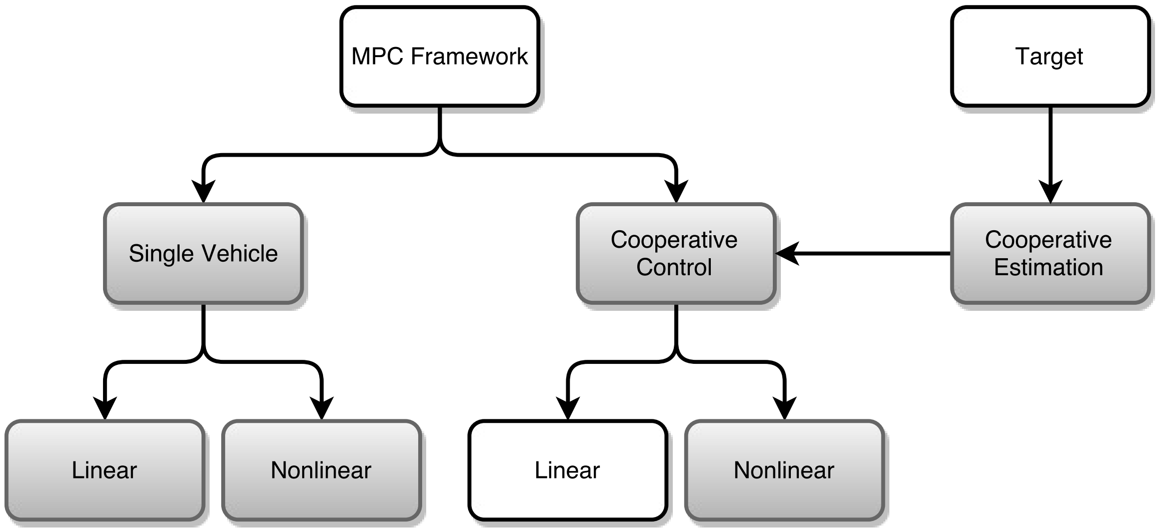


Figure 1.2. Model Predictive Control Overview.

CHAPTER 2

Preliminaries

2.1 Definitions

Define the 2-norm of a vector $\mathbf{x} \in \Re^n$ as:

$$\|\mathbf{x}\| = \sqrt{\mathbf{x}^T\mathbf{x}} \tag{2.1}$$

A matrix $\mathbf{Q} \in \Re^{m\times n}$ is positive definite ($\mathbf{Q} > 0$) if $\mathbf{x}^T\mathbf{Q}\mathbf{x} > 0$ for all $\mathbf{x} \neq \mathbf{0}$.

Suppose that $\mathbf{A} \in \Re^{m\times n}$ is full rank and $n \geq m$. Solutions $\mathbf{X}$ to the problem $\mathbf{AX} = \mathbf{Y}$ can be parameterized as $\mathbf{X} = \mathbf{A}^{-1}\mathbf{Y} + \mathbf{MV}$ for arbitrary matrices $\mathbf{V}$. $\mathbf{A}^{-1} = \mathbf{A}^T(\mathbf{AA}^T)^{-1}$ is the right inverse of $\mathbf{A}$ and $\mathbf{M}$ spans the null space of $\mathbf{A}$ with its columns orthogonal to each other. The numbers of rows of $\mathbf{V}$ is equal to the dimension of the null space of $\mathbf{A}$.

A continuous function $V : \Re^n \to \Re$ is a locally positive definite function if for some $\epsilon > 0$ and some continuous, strictly increasing function $\alpha : \Re_+ \to \Re$,

$$V(\mathbf{0}) = 0, \text{ and, } V(\mathbf{x}) \geq \alpha(||\mathbf{x}||) \quad \forall\, \mathbf{x} \in \mathbf{B}_\epsilon$$

A continuous function $V : \Re^n \to \Re$ is a positive definite function if for some $\epsilon > 0$ and some continuous, strictly increasing function $\alpha : \Re_+ \to \Re$,

$$V(\mathbf{0}) = 0, \text{ and, } V(\mathbf{x}) \geq \alpha(||\mathbf{x}||) \quad \forall\, \mathbf{x} \in \mathbf{B}_\epsilon$$

And $\alpha(p) \to \infty$ as $p \to \infty$.

A continuous function $V : \Re^n \to \Re$ is decrescent if for some $\epsilon > 0$ and some continuous, strictly increasing function $\beta : \Re_+ \to \Re$,

$$V(\mathbf{x}) \leq \beta(||\mathbf{x}||) \quad \forall\, \mathbf{x} \in B_\epsilon$$

Nonlinear (convex) inequalities can be converted to Linear Matrix Inequality (LMI) form using Schur complements. The basic idea is as follows: the LMI

$$\begin{bmatrix} \mathbf{Q}(\mathbf{x}) & \mathbf{S}(\mathbf{x}) \\ \mathbf{S}(\mathbf{x})^T & \mathbf{R}(\mathbf{x}) \end{bmatrix} > \mathbf{0} \tag{2.2}$$

where $\mathbf{Q}(\mathbf{x}) = \mathbf{Q}(\mathbf{x})^T$, $\mathbf{R}(\mathbf{x}) = \mathbf{R}(\mathbf{x})^T$, and $\mathbf{S}(\mathbf{x})$ depends affinely on $\mathbf{x}$ ($\mathbf{x} \in \Re^n$) , is equivalent to

$$\mathbf{R}(\mathbf{x}) > \mathbf{0}, \quad \mathbf{Q}(\mathbf{x}) - \mathbf{S}(\mathbf{x})\mathbf{R}(\mathbf{x})^{-1}\mathbf{S}(\mathbf{x})^T > \mathbf{0}. \tag{2.3}$$

In other words, the set of nonlinear inequalities (2.3) can be represented as the LMI (2.2).

2.2 Stability of Dynamical Systems

In this dissertation, we are only concerned with stability of equilibrium points of a dynamical system in the sense of Lyapunov. For a more comprehensive coverage of stability theory, please refer to [43, 44]. An equilibrium point is stable if all solutions starting at nearby points stay nearby; otherwise, it is unstable. It is asymptotically stable if all solutions starting at nearby points not only stay nearby, but also tend to the equilibrium point as time approaches infinity.

Consider the autonomous system

$$\dot{\mathbf{x}} = f(\mathbf{x}) \tag{2.4}$$

where $f : D \rightarrow \Re^n$ is locally Lipschitz map from a domain $D \subset \Re^n$ into $\Re^n$. Suppose $\mathbf{x} = \mathbf{0}$ is an equilibrium point of system (2.4); that is , $f(\mathbf{0}) = \mathbf{0}$.

The equilibrium point $\mathbf{x} = 0$ of (2.4) is:

- stable (in the sense of Lyapunov) if, for each $\epsilon > 0$, there is $\delta = \delta(\epsilon) > 0$ such that

$$||\mathbf{x}(0)|| < \delta \Rightarrow ||\mathbf{x}(t)|| < \epsilon, \quad \forall\, t \geq 0$$

- unstable if it is not stable.
- asymptotically stable if it is stable and δ can be chosen such that

$$||\mathbf{x}(0)|| < \delta \Rightarrow \lim_{t\to\infty} \mathbf{x}(t) = \mathbf{0}$$

For system (2.4), let $V(\mathbf{x})$ be non-negative function with derivative $\dot{V}(\mathbf{x})$ along the trajectories of the system.

- If $V(\mathbf{x})$ is locally positive definite and $\dot{V}(\mathbf{x}) \leq 0$ locally in $\mathbf{x}$, then the origin of the system is locally stable (in the sense of Lyapunov).
- If $V(\mathbf{x})$ is locally positive definite and decrescent, and $\dot{V}(\mathbf{x}) \leq 0$ locally in $\mathbf{x}$, then the origin of the system is uniformly locally stable (in the sense of Lyapunov).
- If $V(\mathbf{x})$ is locally positive definite and decrescent, and $-\dot{V}(\mathbf{x})$ locally positive definite, then the origin of the system is locally uniformly asymptotically stable.
- If $V(\mathbf{x})$ is positive definite and decrescent, and $-\dot{V}(\mathbf{x})$ locally positive definite, then the origin of the system is globally uniformly asymptotically stable.

2.3 Optimization

Given $\mathbf{x}_0$, assume a discrete system model:

$$\mathbf{x}_{k+1} = \mathbf{A}\mathbf{x}_k + \mathbf{B}\mathbf{u}_k, \tag{2.5}$$

where $\mathbf{A} \in \Re^{n\times n}$, $\mathbf{B} \in \Re^{n\times m}$, $\mathbf{x}_k \in \Re^n$, and $\mathbf{u}_k \in \Re^m$ are the system matrix, input matrix, states, and inputs, respectively.

The following optimization problems are relevant to this work.

2.3.1 Unconstrained Optimization

2.3.1.1 Linear Quadratic Regulator (LQR)

We start with the simplest case: Linear Quadratic Regulator (LQR). The performance objective for LQR is of the form:

$$J(\mathbf{x}_k, \mathbf{u}_k) = \frac{1}{2}\sum_{i=0}^{\infty}(\mathbf{x}_{k+i}^T\mathbf{Q}\mathbf{x}_{k+i} + \mathbf{u}_{k+i}^T\mathbf{R}\mathbf{u}_{k+i}) \tag{2.6}$$

where $\mathbf{Q} \in \Re^{n\times n}$ is the state weighting matrix and $\mathbf{Q} > 0$. $\mathbf{R} \in \Re^{m\times m}$ is the input weighting matrix and $\mathbf{R} > 0$. The control input that would minimize the objective function can be shown as:

$$\mathbf{u}_k = -(\mathbf{R} + \mathbf{B}^T\mathbf{P}\mathbf{B})^{-1}\mathbf{B}^T\mathbf{P}\mathbf{A}\mathbf{x}_k \tag{2.7}$$

in which $\mathbf{P}$ is the solution of the discrete algebraic Riccati Equation:

$$\mathbf{A}^T\mathbf{P}\mathbf{A} - \mathbf{P} + \mathbf{Q} - \mathbf{A}^T\mathbf{P}\mathbf{B}(\mathbf{R} + \mathbf{B}^T\mathbf{P}\mathbf{B})^{-1}\mathbf{B}^T\mathbf{P}\mathbf{A} = 0$$

2.3.1.2 Finite Horizon Quadratic Cost with Terminal Equality Constraint Regulator

Given horizon N, the performance objective is chosen as:

$$J(\mathbf{x}_k, \mathbf{u}_k) = \frac{1}{2}\sum_{i=0}^{N-1}(\mathbf{x}_{k+i}^T\mathbf{Q}\mathbf{x}_{k+i} + \mathbf{u}_{k+i}^T\mathbf{R}\mathbf{u}_{k+i}) \tag{2.8}$$

where $\mathbf{Q} \in \Re^{n\times n}$, $\mathbf{R} \in \Re^{m\times m}$ and $\mathbf{Q} > 0$, $\mathbf{R} > 0$. A terminal equality constraint is to be enforced so the system state goes to the origin at the end of the horizon:

$$\mathbf{x}_{k+N} = \mathbf{0} \tag{2.9}$$

The control inputs that minimize the above objective function is:

$$\mathbf{u}_k = -\mathbf{R}^{-1}\mathbf{B}^T\mathbf{P}_{k+1}\mathbf{A}\mathbf{x}_k \tag{2.10}$$

where

$$\mathbf{P}_k = \mathbf{A}^{-1}(\mathbf{I} + \mathbf{P}_{k+1}\mathbf{A}^{-T}\mathbf{Q}\mathbf{A}^{-1})^{-1}\mathbf{P}_{k+1}\mathbf{A} + \mathbf{B}\mathbf{R}^{-1}\mathbf{B}^T \tag{2.11}$$

with boundary condition $\mathbf{P}_{k+N} = \mathbf{B}\mathbf{R}^{-1}\mathbf{B}^T$.

2.3.1.3 Finite Horizon Quadratic Cost with Free Terminal State Regulator

Given horizon N, the performance objective is chosen as:

$$J(\mathbf{x}_k, \mathbf{u}_k) = \frac{1}{2}\sum_{i=0}^{N-1}(\mathbf{x}_{k+i}^T\mathbf{Q}\mathbf{x}_{k+i} + \mathbf{u}_{k+i}^T\mathbf{R}\mathbf{u}_{k+i}) + \frac{1}{2}\mathbf{x}_{k+N}^T\mathbf{Q}_f\mathbf{x}_{k+N} \tag{2.12}$$

where $\mathbf{Q}, \mathbf{Q}_f \in \Re^{n\times n}$, $\mathbf{R} \in \Re^{m\times m}$ and $\mathbf{Q}, \mathbf{Q}_f > 0$, $\mathbf{R} > 0$. $\mathbf{Q}_f$ is termed as the terminal weighting matrix. No constraint is enforced on the terminal state. The control inputs that minimize the objective function is:

$$\mathbf{u}_k = -\mathbf{R}^{-1}\mathbf{B}^T(\mathbf{I} + \mathbf{K}_{k+1}\mathbf{B}\mathbf{R}^{-1}\mathbf{B}^T)^{-1}\mathbf{K}_{k+1}\mathbf{A}\mathbf{x}_k \tag{2.13}$$

where

$$\mathbf{K}_k = \mathbf{A}^T\mathbf{K}_{k+1}\mathbf{A} - \mathbf{A}^T\mathbf{K}_{k+1}\mathbf{B}(\mathbf{R} + \mathbf{B}^T\mathbf{K}_{k+1}\mathbf{B})^{-1}\mathbf{B}^T\mathbf{K}_{k+1}\mathbf{A} + \mathbf{Q} \tag{2.14}$$

with boundary condition $\mathbf{K}_{k+N} = \mathbf{Q}_f$. This is an initial value problem and $\mathbf{K}_k$ is obtained by solving Eq. (2.14) backwards.

2.3.2 Constrained Optimization

The system of Eq. (2.5) is represented in the following predictive form (see Ref. [40]):

$$\mathbf{x}_{k+i+1} = \mathbf{A}\mathbf{x}_{k+i} + \mathbf{B}\mathbf{u}_{k+i} \tag{2.15}$$

with input and state constraints:

$$\begin{aligned} \mathbf{u}_{lb} \leq \quad \mathbf{u}_{k+j} \quad &\leq \mathbf{u}_{ub}, \quad j = 0, 1, \cdots, N-1 \\ \mathbf{z}_{lb} \leq \mathbf{C}_{\mathbf{z}}\mathbf{x}_{k+j} &\leq \mathbf{z}_{ub}, \quad j = 0, 1, \cdots, N \end{aligned} \tag{2.16}$$

States on $[k, k+N]$ can be formulated in batch form as:

$$\mathbf{X}_k = \mathbf{F}\mathbf{x}_k + \mathbf{H}\mathbf{U}_k \tag{2.17}$$

where $\mathbf{X}_k$, $\mathbf{F}$, $\mathbf{H}$, $\mathbf{U}_k$ are given as:

$$\mathbf{X}_k = \begin{bmatrix} \mathbf{x}_k \\ \mathbf{x}_{k+1} \\ \vdots \\ \mathbf{x}_{k+N-1} \end{bmatrix}, \quad \mathbf{U}_k = \begin{bmatrix} \mathbf{u}_k \\ \mathbf{u}_{k+1} \\ \vdots \\ \mathbf{u}_{k+N-1} \end{bmatrix}, \quad \mathbf{F} = \begin{bmatrix} \mathbf{I} \\ \mathbf{A} \\ \vdots \\ \mathbf{A}^{N-1} \end{bmatrix}$$

$$\mathbf{H} = \begin{bmatrix} \mathbf{0} & & & & \\ \mathbf{B} & \mathbf{0} & & & \\ \mathbf{AB} & \mathbf{B} & \mathbf{0} & & \\ \vdots & \vdots & \ddots & \ddots & \\ \mathbf{A}^{N-2}\mathbf{B} & \mathbf{A}^{N-3}\mathbf{B} & \dots & \mathbf{B} & \mathbf{0} \end{bmatrix}$$

And terminal state

$$\mathbf{x}_{k+N} = \mathbf{A}^N \mathbf{x}_k + \bar{\mathbf{B}} \mathbf{U}_k \tag{2.18}$$

where

$$\bar{\mathbf{B}} = \begin{bmatrix} \mathbf{A}^{N-1}\mathbf{B} & \mathbf{A}^{N-2}\mathbf{B} & \cdots & \mathbf{AB} & \mathbf{B} \end{bmatrix}$$

Constraints in Eq. (2.16) can be re-written as:

$$\begin{bmatrix} \mathbf{u}_{lb} \\ \mathbf{u}_{lb} \\ \vdots \\ \mathbf{u}_{lb} \end{bmatrix} \leq \mathbf{U}_k \leq \begin{bmatrix} \mathbf{u}_{ub} \\ \mathbf{u}_{ub} \\ \vdots \\ \mathbf{u}_{ub} \end{bmatrix}, \quad \begin{bmatrix} \mathbf{z}_{lb} \\ \mathbf{z}_{lb} \\ \vdots \\ \mathbf{z}_{lb} \end{bmatrix} \leq \mathcal{C}_z \mathbf{X}_k \leq \begin{bmatrix} \mathbf{z}_{ub} \\ \mathbf{z}_{ub} \\ \vdots \\ \mathbf{z}_{ub} \end{bmatrix} \tag{2.19}$$

where

$$\mathcal{C}_z = \begin{bmatrix} \mathbf{C_z} & & & \\ & \mathbf{C_z} & & \\ & & \mathbf{C_z} & \\ & & & \mathbf{C_z} \end{bmatrix}$$

2.3.2.1 Finite Horizon Quadratic Cost with Terminal Equality Constraint Regulator

Again, consider the following performance objective function:

$$J(\mathbf{x}_k, \mathbf{u}_k) = \frac{1}{2}\sum_{i=0}^{N-1}(\mathbf{x}_{k+i}^T\mathbf{Q}\mathbf{x}_{k+i} + \mathbf{u}_{k+i}^T\mathbf{R}\mathbf{u}_{k+i}) \tag{2.20}$$

where $\mathbf{Q} \in \Re^{n\times n}$, $\mathbf{R} \in \Re^{m\times m}$ and $\mathbf{Q} > 0$, $\mathbf{R} > 0$. A terminal equality constraint is to be enforced so the system state goes to the origin at the end of the horizon:

$$\mathbf{x}_{k+N} = \mathbf{0} \tag{2.21}$$

which means $\mathbf{A}^N\mathbf{x}_k + \bar{\mathbf{B}}\mathbf{U}_k = \mathbf{0}$. We can parameterize $\mathbf{U}_k$ based on this as shown in Eq. (2.22).

$$\mathbf{U}_k = -\bar{\mathbf{B}}^{-1}\mathbf{A}^N\mathbf{x}_k + \mathbf{M}\hat{\mathbf{U}}_k \tag{2.22}$$

where $\hat{\mathbf{U}}_k$ is a matrix containing the independent variables. The constraints (2.19) need to be reformulated as:

$$\begin{bmatrix} \mathbf{u}_{lb} \\ \mathbf{u}_{lb} \\ \vdots \\ \mathbf{u}_{lb} \end{bmatrix} + \bar{\mathbf{B}}^{-1}\mathbf{A}^N\mathbf{x}_k \;\le\; \mathbf{M}\hat{\mathbf{U}}_k \;\le\; \begin{bmatrix} \mathbf{u}_{ub} \\ \mathbf{u}_{ub} \\ \vdots \\ \mathbf{u}_{ub} \end{bmatrix} + \bar{\mathbf{B}}^{-1}\mathbf{A}^N\mathbf{x}_k \tag{2.23}$$

If there is a feasible solution for the system of Eq. (2.15) and the objective function Eq. (2.20) with constraints specified by Eq. (2.16) at the initial time, then the next solution is guaranteed to exist. The optimization problem for the above case can be summarized to the following Semi-Definite Programming (SDP) problem:

$$\min_{\hat{\mathbf{U}}_k} \gamma_1 \tag{2.24}$$

$$\text{subj. to: } \begin{bmatrix} \gamma_1 - \mathcal{V}_1 - (2(\mathbf{A}^N\mathbf{x}_k)^T\bar{\mathbf{B}}^{-T}\mathbf{W} - 2\mathbf{H}^T\bar{\mathbf{Q}}\mathbf{F}\mathbf{x}_k)\mathbf{M}\hat{\mathbf{U}}_k & -2\hat{\mathbf{U}}_k^T\sqrt{\mathcal{V}_2} \\ -2\sqrt{\mathcal{V}_2}\hat{\mathbf{U}}_k & \mathbf{I} \end{bmatrix} \le 0$$

(2.19)

where

$$\mathcal{V}_1 = (\mathbf{A}^N\mathbf{x}_k)^T\bar{\mathbf{B}}^{-T}\mathbf{W}\bar{\mathbf{B}}^{-1}(\mathbf{A}^N\mathbf{x}_k) + (\mathbf{F}\mathbf{x}_k)^T\bar{\mathbf{Q}}(\mathbf{F}\mathbf{x}_k) - 2\mathbf{H}^T\bar{\mathbf{Q}}\mathbf{F}\mathbf{x}_k\bar{\mathbf{B}}^{-1}\mathbf{A}^N\mathbf{x}_k$$

$$\mathcal{V}_2 = \mathbf{M}^T(\mathbf{H}^T\bar{\mathbf{Q}}\mathbf{H} + \bar{\mathbf{R}})\mathbf{M}$$

and

$$\bar{\mathbf{Q}} = \begin{bmatrix} \mathbf{Q} & & & \\ & \mathbf{Q} & & \\ & & \ddots & \\ & & & \mathbf{Q} \end{bmatrix}, \quad \bar{\mathbf{R}} = \begin{bmatrix} \mathbf{R} & & & \\ & \mathbf{R} & & \\ & & \ddots & \\ & & & \mathbf{R} \end{bmatrix}$$

And after obtaining the optimal $\hat{\mathbf{U}}_k^*$ from (2.24), substitute the value back to Eq. (2.22) to get $\mathbf{U}_k$.

$$\mathbf{U}_k = -\bar{\mathbf{B}}^{-1}\mathbf{A}^N\mathbf{x}_k + \mathbf{M}\hat{\mathbf{U}}_k^*$$

2.3.2.2 Finite Horizon Quadratic Cost with Free Terminal Cost Regulator

Again, the performance objective function is chosen as:

$$J(\mathbf{x}_k, \mathbf{u}_k) = \frac{1}{2}\sum_{i=0}^{N-1}\left(\mathbf{x}_{k+i}^T\mathbf{Q}\mathbf{x}_{k+i} + \mathbf{u}_{k+i}^T\mathbf{R}\mathbf{u}_{k+i}\right) + \frac{1}{2}\mathbf{x}_{k+N}^T\mathbf{Q}_f\mathbf{x}_{k+N} \tag{2.25}$$

where $\mathbf{Q}, \mathbf{Q}_f \in \Re^{n\times n}$, $\mathbf{R} \in \Re^{m\times m}$ and $\mathbf{Q}, \mathbf{Q}_f > 0$, $\mathbf{R} > 0$. No constraint is enforced on the terminal state. The same constraints as in Eq. (2.19) applies here as well.

$$\begin{bmatrix} \mathbf{u}_{lb} \\ \mathbf{u}_{lb} \\ \vdots \\ \mathbf{u}_{lb} \end{bmatrix} \leq \mathbf{U}_k \leq \begin{bmatrix} \mathbf{u}_{ub} \\ \mathbf{u}_{ub} \\ \vdots \\ \mathbf{u}_{ub} \end{bmatrix}, \quad \begin{bmatrix} \mathbf{z}_{lb} \\ \mathbf{z}_{lb} \\ \vdots \\ \mathbf{z}_{lb} \end{bmatrix} \leq \mathcal{C}_z\mathbf{X}_k \leq \begin{bmatrix} \mathbf{z}_{ub} \\ \mathbf{z}_{ub} \\ \vdots \\ \mathbf{z}_{ub} \end{bmatrix} \tag{2.26}$$

The solution to this problem can be transformed into the following SDP problem:

$$\min_{\mathbf{U}_k}(\gamma_1 + \gamma_2) \tag{2.27}$$

$$\text{subj. to:} \begin{bmatrix} \gamma_1 - 2\mathbf{x}_k^T\mathbf{F}^T\bar{\mathbf{Q}}\mathbf{H}\mathbf{U}_k - \mathbf{x}_k^T\mathbf{F}_k^T\bar{\mathbf{Q}}\mathbf{F}_k\mathbf{x}_k & \mathbf{U}_k^T \\ \mathbf{U}_k & (\mathbf{H}^T\bar{\mathbf{Q}}\mathbf{H} + \mathbf{R})^{-1} \end{bmatrix} \geq 0$$

$$\begin{bmatrix} \gamma_2 & [\mathbf{A}^N\mathbf{x}_k + \bar{\mathbf{B}}\mathbf{U}_k]^T \\ \mathbf{A}^N\mathbf{x}_k + \bar{\mathbf{B}}\mathbf{U}_k & \mathbf{Q}_f^{-1} \end{bmatrix} \geq 0$$

(2.19)

and the optimal control $\mathbf{U}_k^*$ can be obtained from solving Eq. (2.27).

2.3.3 Linear Matrix Inequality and Semi-Definite Programming

A brief introduction to Semi-Definite Programming (SDP) is given here. SDP can effectively solve many optimization problems involving Linear Matrix Inequalities (LMI). For a more detailed explanation, please refer to [45, 46].

Consider the problem of minimizing a linear function of a variable $\mathbf{x} \in \Re^m$ subject to a matrix inequality:

$$\begin{aligned} \text{minimize} \quad & \mathbf{c}^T\mathbf{x} \\ \text{subject to } & \mathbf{F}(\mathbf{x}) \geq \mathbf{0} \end{aligned} \tag{2.28}$$

where

$$\mathbf{F}(\mathbf{x}) \triangleq \mathbf{F}_0 + \sum_{i=1}^{m} \mathbf{x}_i\mathbf{F}_i$$

And vector $\mathbf{c} \in \Re^m$ and $m+1$ symmetric matrices $\mathbf{F}_0, \ldots, \mathbf{F}_m \in \Re^{n\times n}$. The inequality sign in $\mathbf{F}(\mathbf{x}) \geq \mathbf{0}$ means that $\mathbf{F}(\mathbf{x})$ is positive definite, i.e., $\mathbf{z}^T\mathbf{F}(\mathbf{x})\mathbf{z} \geq 0$ for all $\mathbf{z} \in \Re^n$. Inequality $\mathbf{F}(\mathbf{x}) \geq \mathbf{0}$ is called an linear matrix inequality and the problem (2.28) is called a semidefinite program. A semidefinite program is a convex optimization problem because its objective and constraint are convex: for $\mathbf{x}$, $\mathbf{y} \in D \subset \Re^m$, if $\mathbf{F}(\mathbf{x}) \geq \mathbf{0}$ and $\mathbf{F}(\mathbf{y}) \geq \mathbf{0}$, then for any $\lambda \in [0, 1]$,

$$\mathbf{F}(\lambda\mathbf{x} + (1-\lambda)\mathbf{y}) = \lambda\mathbf{F}(\mathbf{x}) + (1-\lambda)\mathbf{F}(\mathbf{y}) \geq \mathbf{0}$$

SDP can represent many important optimization problems. Consider a linear programming problem:

$$\begin{aligned} \text{minimize} \quad & \mathbf{c}^T\mathbf{x} \\ \text{subject to } & \mathbf{A}\mathbf{x} + \mathbf{b} \geq \mathbf{0} \end{aligned} \tag{2.29}$$

in which the inequality denotes component-wise inequality. And $\mathbf{x}$, $\mathbf{c} \in \Re^m$, $\mathbf{A} \in \Re^{n\times m}$, $\mathbf{b} \in \Re^n$. A vector $\mathbf{v} \geq \mathbf{0}$ if only if the matrix diag($\mathbf{v}$) is positive semidefinite, we can reformulate (2.29) as a SDP with $\mathbf{F}(\mathbf{x}) = \text{diag}(\mathbf{A}\mathbf{x} + \mathbf{b})$, where

$$\mathbf{F}_0 = \text{diag}(\mathbf{b}), \quad \mathbf{F}_i = \text{diag}(\mathbf{a}_i), \quad i = 1, \ldots, m$$

and $\mathbf{a}_i \in \Re^n$, $\mathbf{A} = [\mathbf{a}_1 \ \ldots \ \mathbf{a}_m]$. The notation diag denotes:

$$\text{diag}(\mathbf{v}) = \begin{bmatrix} \mathbf{v}_1 & 0 & \cdots & 0 \\ 0 & \mathbf{v}_2 & \cdots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ 0 & \cdots & \cdots & \mathbf{v}_n \end{bmatrix}$$

Also consider a quadratic programming (QP) problem, which will be used extensively later in this dissertation:

$$\begin{aligned} \text{minimize} \ & \frac{1}{2}\mathbf{x}^T\mathbf{P}\mathbf{x} + \mathbf{q}^T\mathbf{x} + r \\ \text{subject to} \quad & \mathbf{A}\mathbf{x} + \mathbf{b} \geq \mathbf{0} \end{aligned} \tag{2.30}$$

where $\mathbf{P} \in \Re^{m\times m}$, $\mathbf{P} > 0$, $\mathbf{q} \in \Re^n$, $r \in \Re$, $\mathbf{A} \in \Re^{n\times m}$, and $\mathbf{b} \in \Re^n$. This QP problem can be expressed in SDP form as:

$$\begin{aligned} &\text{minimize } t \\ &\text{subject to } \begin{bmatrix} t - r - \mathbf{q}^T\mathbf{x} & \mathbf{x}^T \\ \mathbf{x} & 2\mathbf{P}^{-1} \end{bmatrix} \geq \mathbf{0} \\ &\qquad \text{diag}(\mathbf{b}) + \sum_{i=1}^{m} \mathbf{x}_i \, \text{diag}(\mathbf{a}_i) \geq \mathbf{0} \end{aligned} \tag{2.31}$$

where $[\mathbf{a}_1 \ \dots \ \mathbf{a}_m] = \mathbf{A}$.

In particular, SDP can also represent a quadratically constrained quadratic programming (QCQP) problem. For example, a convex quadratic constraint like

$$(\mathbf{Ax} + \mathbf{b})^T(\mathbf{Ax} + \mathbf{b}) - \mathbf{c}^T\mathbf{x} - \mathbf{d} \leq 0$$

can be written as:

$$\begin{bmatrix} \mathbf{I} & \mathbf{Ax} + \mathbf{b} \\ (\mathbf{Ax} + \mathbf{b})^T & \mathbf{c}^T\mathbf{x} + \mathbf{d} \end{bmatrix} \geq \mathbf{0}$$

The left-hand side depends affinely on $\mathbf{x}$: it can be expressed as

$$\mathbf{F}(\mathbf{x}) = \mathbf{F}_0 + \mathbf{x}_1\mathbf{F}_1 + \cdots + \mathbf{x}_k\mathbf{F}_k \geq \mathbf{0}$$

with

$$\mathbf{F}_0 = \begin{bmatrix} \mathbf{I} & \mathbf{b} \\ \mathbf{b}^T & \mathbf{d} \end{bmatrix}, \quad \mathbf{F}_i = \begin{bmatrix} \mathbf{0} & \mathbf{a}_i \\ \mathbf{a}_i^T & \mathbf{c}_i \end{bmatrix}, \quad i = 1, \dots, L$$

where $\mathbf{A} = [\mathbf{a}_1 \dots \mathbf{a}_k]$.

2.4 Graph Theory

A brief tutorial on graph theory is given here. Information exchange between vehicles are often modeled by directed or undirected graphs. A directed graph is a pair $(\mathcal{N}, \mathcal{E})$, where $\mathcal{N} = \{1, \ldots, n\}$ is a finite nonempty node set and $\mathcal{E} \subset \mathcal{N} \times \mathcal{N}$ is an edge set of ordered pairs of nodes, called edges. The edge $(i, j) \in \mathcal{E}$ denotes vehicle j can obtain information from vehicle i. And i is called the parent node and i is the child node. Self edges $(i, i) \in \mathcal{E}$ are allowed.

A directed graph is strongly connected if there is a directed path from every node to every other node. A rooted directed tree is a directed graph in which every node has exactly one parent except for one node, called the root, which has no parent and which has a directed path to every other node.

A subgraph $(\mathcal{N}_1, \mathcal{E}_1)$ of $(\mathcal{N}, \mathcal{E})$ is a graph such that $\mathcal{N}_1 \subset \mathcal{N}$ and $\mathcal{E}_1 \subset (\mathcal{E} \bigcap \mathcal{N}_1 \times \mathcal{N}_1)$. The graph $(\mathcal{N}, \mathcal{E})$ has or contains a rooted directed spanning tree if a rooted directed spanning tree is a subgraph of $(\mathcal{N}, \mathcal{E})$. The existence of a rooted directed spanning tree is a weaker condition than being strongly connected.

2.5 Quadcopter Platform

2.5.1 Introduction to Quadcopter

A Quadcopter helicopter platform (often just called Quadcopter), is an under-actuated helicopter with two pairs of rotors in a cross configuration capable of spinning at different angular velocities in order to achieve translational and rotational motion. Rotor pair (1, 3) spins in one direction while the pair (2, 4) spins in the opposite (see Figure 2.1). Quadcopters can achieve different motions by imbalance of different pairs of rotor speed or simultaneous changes of all 4 rotors.

Quadcopters have been gaining popularity as research platforms because of their simplicity of design, their low cost of manufacturing compared to other unmanned aerial vehicles. Because they are challenging vehicles to control, wherever operated in an indoor environment or in the open field, they make a great platform for re-

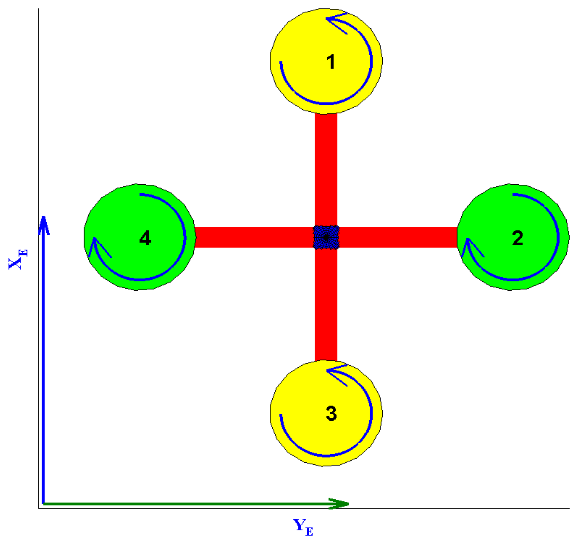


Figure 2.1. Diagram of a quadcopter top view.

search and development. Quadcopters have applications in both military and the civil sectors, some of which include, surveillance and reconnaissance, search and rescue, communications, logistics missions, fire fighting, agriculture, wildlife monitoring, terrain mapping, cave exploration, atmospheric monitoring, advertising, sports and entertainment, and law enforcement [47–51].

There is extensive literature on attitude stabilization, trajectory tracking, and formation stabilization of quadcopters. Virtually every possible control technique, linear control such as PID and linear quadratic methods, robust linear control, and nonlinear control [52, 53] techniques such as nonlinear model predictive control [39], adaptive control [54], iterative learning control [55], neural networks [56], backstepping [57], and sliding mode control among others have been tried and tested in simulations.

2.5.2 Quadcopter Model: Governing Equations of Motion

As shown in Fig. (2.2), let $O_E X_E Y_E Z_E$ denote an Earth-fixed Inertial frame and $O_B X_B Y_B Z_B$ a body-fixed frame whose origin O_B is at the center of mass of the quadrotor. The inertial position of the quadrotor is defined by $\mathbf{p} = (x, y, z)^T$ and the attitude by three Euler angles: roll, pitch, and yaw ($\mathbf{\Theta} = [\phi\ \theta\ \psi]^T$). $\mathbf{R} \in SO(3)$ is the orthogonal rotation matrix to orient the quadrotor using the Euler angles following the 3-2-1 sequence of rotations. $\mathbf{V}_B = (V_x, V_y, V_z)^T$ and $\mathbf{\Omega} = (p, q, r)^T$ represents for the inertial velocity in the body fixed frame and body angular velocities, respectively.

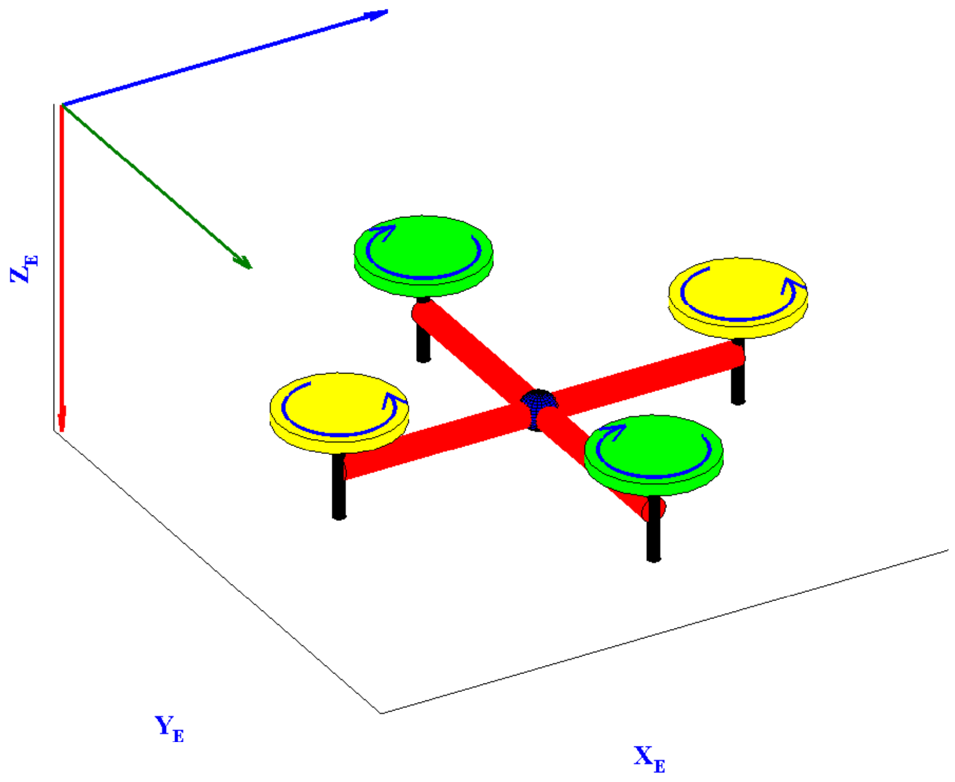


Figure 2.2. Quadcopter.

Neglecting the aerodynamic and gyroscopic effects, the quadrotor model can be shown as (Ref. [58]):

$$\begin{cases} \dot{\mathbf{p}} = \mathbf{R}_{BI}^T \mathbf{V}_B \\ \dot{\mathbf{V}}_{\mathbf{B}} = -\boldsymbol{\Omega} \times \mathbf{V}_B + \mathbf{R}_{BI}(g\hat{\mathbf{e}}_3) + \frac{T}{m}\hat{\mathbf{e}}_{\mathbf{3}} \\ \dot{\boldsymbol{\Theta}} = \mathbf{W}(\phi,\ \theta,\ \psi)\boldsymbol{\Omega} \\ \dot{\boldsymbol{\Omega}} = \mathbf{J}^{-1}(-\boldsymbol{\Omega} \times \mathbf{J}\boldsymbol{\Omega} + \boldsymbol{\tau}) \end{cases} \tag{2.32}$$

Where $\hat{\mathbf{e}}_{\mathbf{3}} = [0\ 0\ 1]^T$, $\boldsymbol{\tau} = [\tau_1\ \tau_2\ \tau_3]^T$ and

$$\mathbf{R}_{BI}(\phi,\ \theta,\ \psi) = \begin{bmatrix} C_\theta C_\psi & C_\theta S_\psi & -S_\theta \\ S_\theta C_\psi S_\phi - S_\psi C_\phi & S_\theta S_\psi S_\phi + C_\psi C_\phi & C_\theta S_\phi \\ S_\theta C_\psi C_\phi + S_\psi S_\phi & S_\theta S_\psi C_\phi - C_\psi S_\phi & C_\theta C_\phi \end{bmatrix}$$

$$\mathbf{W}(\phi,\ \theta,\ \psi) = \begin{bmatrix} 1 & \sin\phi\tan\theta & \cos\phi\tan\theta \\ 0 & \cos\phi & -\sin\phi \\ 0 & \sin\phi\sec\theta & \cos\phi\sec\theta \end{bmatrix} \quad \mathbf{J} = \begin{bmatrix} J_x & & \\ & J_y & \\ & & J_z \end{bmatrix} \tag{2.33}$$

and $C.$ and $S.$ denote the trigonometric functions $\sin\cdot$ and $\cos\cdot$, respectively.

The constants $g, m, \mathbf{J}$ in the equation denote the standard gravity acceleration, the mass and moment of inertia of the quadcopter, respectively. The variables T and $\boldsymbol{\tau}$ represent the total thrust and torques about the body axes of the quadcopter.

In addition, the relations between the total thrust, torque and lifting forces provided by each rotor can be expressed as:

$$\begin{bmatrix} T \\ \tau_1 \\ \tau_2 \\ \tau_3 \end{bmatrix} = \begin{bmatrix} -1 & -1 & -1 & -1 \\ 0 & -L & 0 & L \\ L & 0 & -L & 0 \\ -c & c & -c & c \end{bmatrix} \begin{bmatrix} F_1 \\ F_2 \\ F_3 \\ F_4 \end{bmatrix} \tag{2.34}$$

in which L is the distance from the rotor to the center of gravity (CG) of the quadcopter, c relates the rotor angular moment to the rotor lift (normal force). Combining Eq. (2.32) and (2.34), a more comprehensive model can be written as:

$$\begin{bmatrix} \dot{\mathbf{p}} \\ \dot{\mathbf{V}}_{\mathbf{B}} \\ \dot{\mathbf{\Theta}} \\ \dot{\mathbf{\Omega}} \end{bmatrix} = \begin{bmatrix} \mathbf{R}_{BI}^T \mathbf{V}_{\mathbf{B}} \\ -\mathbf{\Omega} \mathbf{V}_B + \mathbf{R}_{BI}(g\hat{\mathbf{e}}_{\mathbf{3}}) \\ \mathbf{W}(\phi,\ \theta,\ \psi)\mathbf{\Omega} \\ \mathbf{J}^{-1}(-\mathbf{\Omega} \times \mathbf{J}\mathbf{\Omega}) \end{bmatrix} + \begin{bmatrix} 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ -\frac{1}{m} & -\frac{1}{m} & -\frac{1}{m} & -\frac{1}{m} \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & -J_x^{-1}L & 0 & J_x^{-1}L \\ J_y^{-1}L & 0 & -J_y^{-1}L & 0 \\ -J_z^{-1}c & J_z^{-1}c & -J_z^{-1}c & J_z^{-1}c \end{bmatrix} \begin{bmatrix} F_1 \\ F_2 \\ F_3 \\ F_4 \end{bmatrix} \tag{2.35}$$

Let $\mathbf{x} = [\mathbf{p}^T\ \mathbf{V_B}^T\ \mathbf{\Theta}^T\ \mathbf{\Omega}^T]^T$ and $\mathbf{u} = [F_1\ F_2\ F_3\ F_4]^T$. Eq. (2.35) can be compactly written as:

$$\dot{\mathbf{x}} = \mathbf{f}(\mathbf{x}) + \mathbf{B_c}\mathbf{u} \tag{2.36}$$

Given the nature of the quadcopter, we restrict the range of operation of the vehicle as shown below. For e.g., at a pitch angle of $\theta = \pm\frac{\pi}{2}$, the matrix $\mathbf{W}(\phi,\ \theta,\ \psi)$ in Eq. (2.33) becomes singular.

$$\begin{array}{ccccc} -\pi & \leq & \phi & \leq & \pi \\ -\frac{\pi}{2} & < & \theta & < & \frac{\pi}{2} \\ -\pi & < & \psi & < & \pi \end{array} \tag{2.37}$$

Each rotor is assumed to behave approximately as a first order system (See Ref. [59]):

$$\dot{F}_i = \lambda_F(F_i^c - F_i), \quad i = 1, 2, 3, 4 \tag{2.38}$$

where F_i^c denotes the commanded value for rotor lift (thrust) and λ_F denotes first order actuator time constant (assumed same for all the rotors). Also we constrain the force each rotor is able to exert onto the body:

$$0 \leq F_i \leq \kappa_i mg, \ \kappa_i \in (\frac{1}{4}, 1] \tag{2.39}$$

CHAPTER 3

Linear Model Predictive Control

3.1 System Linearization

Given a nonlinear system of the form:

$$\dot{\mathbf{x}} = \mathbf{f}(\mathbf{x}(t), \mathbf{u}(t)) \tag{3.1}$$

where $\mathbf{x}(t) \in \Re^n$ are the system states and $\mathbf{u}(t) \in \Re^m$ are the system outputs as functions of time t. In general $m < n$ and $\mathbf{f}$ is vector valued function in $\mathcal{C}^2$. To apply linear model predictive control, we need to have dynamic system represented by linear form. This is often achieved by linearization. System 3.1 can be linearized about a desired equilibrium operating point. The values of the states and inputs for such an operating point will be denoted as the ordered pair, $\{\mathbf{x}_T, \mathbf{u}_T\}$. The linearized system can be expressed as:

$$\Delta\dot{\mathbf{x}} = \mathbf{A}_c \Delta\mathbf{x} + \mathbf{B}_c \Delta\mathbf{u} \tag{3.2}$$

where $\Delta\mathbf{x} = \mathbf{x} - \mathbf{x}_T$, $\Delta\mathbf{u} = \mathbf{u} - \mathbf{u}_T$, and

$$\mathbf{A}_c = \left.\frac{\partial \mathbf{f}}{\partial \mathbf{x}}\right|_{\mathbf{x}=\mathbf{x}_T, \mathbf{u}=\mathbf{u}_T}, \quad \mathbf{B}_c = \left.\frac{\partial \mathbf{f}}{\partial \mathbf{u}}\right|_{\mathbf{x}=\mathbf{x}_T, \mathbf{u}=\mathbf{u}_T}$$

and the subscript c denotes it is a continuous system. Assuming a sampling interval T_s, the equivalent discretized system can be formulated as:

$$\Delta\mathbf{x}_{k+i+1} = \mathbf{A}\Delta\mathbf{x}_{k+i} + \mathbf{B}\mathbf{u}_{k+i} \tag{3.3}$$

where k is the current sample and $\{\mathbf{A}, \mathbf{B}\}$ is the discrete equivalent of $\{\mathbf{A}_c, \mathbf{B}_c\}$. $\mathbf{A} \in \Re^{n\times n}$ is the state matrix and $\mathbf{B} \in \Re^{n\times m}$ is the input matrix.

Based on the model of (3.3), the controller predicts the future states progression as a function of current states and future inputs. Finite horizon MPC means only a certain number of steps are predicted. The number of steps N is termed as the prediction horizon. The prediction equations bears the same form as (2.17):

$$\Delta \mathbf{X}_k = \mathbf{F}\Delta \mathbf{x}_k + \mathbf{H}\Delta \mathbf{U}_k \tag{3.4}$$

$$\Delta \mathbf{x}_{k+N} = \mathbf{A}^N \Delta \mathbf{x}_k + \bar{\mathbf{B}}\Delta \mathbf{U}_k \tag{3.5}$$

where

$$\Delta \mathbf{X}_k = \begin{bmatrix} \Delta \mathbf{x}_k \\ \Delta \mathbf{x}_{k+1} \\ \Delta \mathbf{x}_{k+2} \\ \vdots \\ \Delta \mathbf{x}_{k+N-1} \end{bmatrix}, \quad \Delta \mathbf{U}_k = \begin{bmatrix} \Delta \mathbf{u}_k \\ \Delta \mathbf{u}_{k+1} \\ \Delta \mathbf{u}_{k+2} \\ \vdots \\ \Delta \mathbf{u}_{k+N-1} \end{bmatrix}$$

and $\mathbf{F}$, $\mathbf{H}$, and $\bar{\mathbf{B}}$ are defined the same as in (2.17).

3.2 Controller Design

The controller is designed to track a desired trajectory or a desired state while at the same time it should minimize the controller effort required. A penalty function is chosen to penalize the deviation of current states and the desired trajectory and the norm of actuator inputs. It is of the form:

$$\begin{aligned} J(\Delta \mathbf{x}_k, \Delta \mathbf{U}_k) = {} & (\Delta \mathbf{X}_k - \Delta \mathbf{X}_k^r)^T \bar{\mathbf{Q}} (\Delta \mathbf{X}_k - \Delta \mathbf{X}_k^r) + \Delta \mathbf{U}_k^T \bar{\mathbf{R}} \mathbf{U}_k \\ & + (\Delta \mathbf{x}_{k+N} - \Delta \mathbf{x}_{k+N}^r)^T \mathbf{Q}_f (\Delta \mathbf{x}_{k+N} - \Delta \mathbf{x}_{k+N}^r) \end{aligned} \tag{3.6}$$

The reference trajectory $\Delta\mathbf{x}^r$ is often known in advance. And $\Delta\mathbf{X}_k^r$ is defined as:

$$\Delta\mathbf{X}_k^r = \begin{bmatrix} \Delta\mathbf{x}_k^r \\ \Delta\mathbf{x}_{k+1}^r \\ \vdots \\ \Delta\mathbf{x}_{k+N-1}^r \end{bmatrix} \tag{3.7}$$

This implies that the controller is able to predict a series of adequate inputs that will drive the system towards the desired goal. The term $\bar{\mathbf{Q}}$ and $\bar{\mathbf{R}}$ are defined as:

$$\bar{\mathbf{Q}} = \begin{bmatrix} \mathbf{Q} & & & \\ & \mathbf{Q} & & \\ & & \ddots & \\ & & & \mathbf{Q} \end{bmatrix}, \quad \bar{\mathbf{R}} = \begin{bmatrix} \mathbf{R} & & & \\ & \mathbf{R} & & \\ & & \ddots & \\ & & & \mathbf{R} \end{bmatrix}$$

where $\mathbf{Q}, \mathbf{Q}_f \in \Re^{n\times n}$ and $\mathbf{R} \in \Re^{m\times m}$ are all diagonal and they satisfy $\mathbf{Q} \geq \mathbf{0}$, $\mathbf{Q}_f \geq \mathbf{0}$ and $\mathbf{R} \geq \mathbf{0}$. $\mathbf{Q}_f$ is chosen based on the solution to the discrete algebraic Riccati equation using

$$\mathbf{A}^T\mathbf{P}\mathbf{A} - \mathbf{P} + \mathbf{Q} - \mathbf{A}^T\mathbf{P}\mathbf{B}(\mathbf{R} + \mathbf{B}^T\mathbf{P}\mathbf{B})^{-1}\mathbf{B}^T\mathbf{P}\mathbf{A} = 0$$

$\mathbf{Q}_f$ is set equal to $\mathbf{P}$. With proper substitution, it follows that,

$$\begin{aligned} J(\Delta\mathbf{x}_k, \Delta\mathbf{U}_k) &= \Delta\mathbf{U}_k^T \left[\mathbf{H}^T\bar{\mathbf{Q}}\mathbf{H} + \bar{\mathbf{R}} + \bar{\mathbf{B}}^T\mathbf{Q}_f\bar{\mathbf{B}}\right] \Delta\mathbf{U}_k \\ &+ 2\left((\mathbf{F}\Delta\mathbf{x}_k - \Delta\mathbf{X}_k^r)^T \bar{\mathbf{Q}}\mathbf{H} + \left(\mathbf{A}^N\Delta\mathbf{x}_k - \Delta\mathbf{x}_{k+N}^r\right)^T \mathbf{Q}_f\bar{\mathbf{B}}\right) \Delta\mathbf{U}_k \\ &+ (\mathbf{F}\Delta\mathbf{x}_k - \Delta\mathbf{X}_k^r)^T \bar{\mathbf{Q}} (\mathbf{F}\Delta\mathbf{x}_k - \Delta\mathbf{X}_k^r) \\ &+ \left(\mathbf{A}^N\Delta\mathbf{x}_k - \Delta\mathbf{x}_{k+N}^r\right)^T \mathbf{Q}_f \left(\mathbf{A}^N\Delta\mathbf{x}_k - \Delta\mathbf{x}_{k+N}^r\right) \end{aligned} \tag{3.8}$$

3.3 Constraints

There are often two kinds of constraints present on a dynamical system, input constraints and state constraints. In many control systems, input variables cannot be arbitrarily large and may have some limitations, such as magnitude limits. Also state values must be bounded in many cases often for safety reasons.

3.3.1 Input Constraints

Input constraints often follow a general form

$$\mathbf{u}_{lb} \leq \mathbf{u}_{k+i} \leq \mathbf{u}_{ub}, \quad i = 1, \ldots, N-1$$

where $\mathbf{u}_{lb}$ is called the lower bound and $\mathbf{u}_{ub}$ is called the upper bound. However, the MPC problem is solved to obtain the perturbation controls since the model employed is the linearized dynamics about the equilibrium point. Since, $\mathbf{u}_{k+i} = \mathbf{u}_T + \Delta\mathbf{u}_{k+i}$, it can be seen that $\mathbf{u}_{lb} \leq \mathbf{u}_T + \Delta\mathbf{u}_{k+i} \leq \mathbf{u}_{ub}$. Alternately, $\mathbf{u}_{lb} - \mathbf{u}_T \leq \Delta\mathbf{u}_{k+i} \leq \mathbf{u}_{ub} - \mathbf{u}_T$. These constraints can be expressed in matrix form as follows:

$$\begin{bmatrix} \mathbf{I}_{m\times m} \\ -\mathbf{I}_{m\times m} \end{bmatrix} \Delta\mathbf{u}_{k+i} \leq \begin{bmatrix} (\mathbf{u}_{ub} - \mathbf{u}_T) \\ -(\mathbf{u}_{lb} - \mathbf{u}_T) \end{bmatrix} \tag{3.9}$$

where $\mathbf{I}_{m\times m}$ is an $m \times m$ identity matrix. After proper arrangement, the input constraint can be represented as:

$$\mathbf{M}_\mathbf{U} \Delta\mathbf{U}_k \leq \Delta\mathbf{U}_b \tag{3.10}$$

where

$$\mathbf{M}_\mathbf{U} = \begin{bmatrix} \begin{bmatrix} \mathbf{I}_{m\times m} \\ -\mathbf{I}_{m\times m} \end{bmatrix} & & & \\ & \begin{bmatrix} \mathbf{I}_{m\times m} \\ -\mathbf{I}_{m\times m} \end{bmatrix} & & \\ & & \ddots & \\ & & & \begin{bmatrix} \mathbf{I}_{m\times m} \\ -\mathbf{I}_{m\times m} \end{bmatrix} \end{bmatrix}$$

$$\Delta \mathbf{U}_b = \begin{bmatrix} \begin{bmatrix} (\mathbf{u}_{ub} - \mathbf{u}_T) \\ -(\mathbf{u}_{lb} - \mathbf{u}_T) \end{bmatrix} \\ \\ \begin{bmatrix} (\mathbf{u}_{ub} - \mathbf{u}_T) \\ -(\mathbf{u}_{lb} - \mathbf{u}_T) \end{bmatrix} \\ \vdots \\ \begin{bmatrix} (\mathbf{u}_{ub} - \mathbf{u}_T) \\ -(\mathbf{u}_{lb} - \mathbf{u}_T) \end{bmatrix} \end{bmatrix}$$

3.3.2 Actuator Rate Constraints

It is also necessary to constrain the rate of change of actuator inputs since the quadcopter motors often function similar to a first order system and abrupt changes cannot be implemented in reality (Ref. [60]). Let $\dot{\mathbf{u}}_{max}$ denote the maximum admissible rate of change, the constraints can be expressed as:

$$-\dot{\mathbf{u}}_{max} T_s \leq \Delta \mathbf{u}_{k+n} - \Delta \mathbf{u}_{k+n-1} \leq \dot{\mathbf{u}}_{max} T_s, \quad n = 1, \ldots, N-1. \tag{3.11}$$

After proper arrangement, the above constraint can be expressed in terms of input constraints in the form of:

$$-\dot{\mathbf{U}}_{max} T_s \leq (\mathbf{M_{U1}} - \mathbf{M_{U2}}) \Delta \mathbf{U}_k \leq \dot{\mathbf{U}}_{max} T_s \tag{3.12}$$

where

$$\mathbf{M_{U1}} = [\mathbf{0}_{(N-1)m \times m} \;\; \mathbf{I}_{(N-1)m \times (N-1)m}], \quad \mathbf{M_{U2}} = [\mathbf{I}_{(N-1)m \times (N-1)m} \;\; \mathbf{0}_{(N-1)m \times m}]$$

$$\dot{\mathbf{U}}_{max} = \begin{bmatrix} \dot{\mathbf{u}}_{max} \\ \dot{\mathbf{u}}_{max} \\ \vdots \\ \dot{\mathbf{u}}_{max} \end{bmatrix}$$

Thus summarized as:

$$\mathbf{M}_{\dot{\mathbf{U}}}\Delta\mathbf{U}_k \leq \dot{\mathbf{U}}_b$$

where

$$\mathbf{M}_{\dot{\mathbf{U}}} = \begin{bmatrix} \mathbf{M}_{\mathbf{U1}} - \mathbf{M}_{\mathbf{U2}} \\ \mathbf{M}_{\mathbf{U2}} - \mathbf{M}_{\mathbf{U1}} \end{bmatrix}, \quad \dot{\mathbf{U}}_b = \begin{bmatrix} \dot{\mathbf{U}}_{max}T_s \\ \dot{\mathbf{U}}_{max}T_s \end{bmatrix}$$

3.3.3 State and Output Constraints

Output constraints are categorized here as well because they are often a combinations of some or all of the states. States can also be considered as a special case of output when output matrix $\mathbf{C} = \mathbf{I}$. If a particular output is defined as $\Delta\mathbf{z} = \mathbf{C}_{\mathbf{z}}\Delta\mathbf{x}_k$, the constraints are represented as $\mathbf{z}_{lb} \leq \mathbf{C}_{\mathbf{z}}\mathbf{x}_{k+i} \leq \mathbf{z}_{ub}$ for $i = 0, 1, 2, \ldots, N-1$, where $\mathbf{z}_{lb}$ and $\mathbf{z}_{ub}$ are the upper and lower bounds for the output variables. Since $\mathbf{x}_{k+i} = \mathbf{x}_T + \Delta\mathbf{x}_{k+i}$, $\mathbf{z}_{lb} \leq \mathbf{C}_{\mathbf{z}}\left(\mathbf{x}_T + \Delta\mathbf{x}_{k+i}\right) \leq \mathbf{z}_{ub}$. Thus, $\mathbf{z}_{lb} - \mathbf{C}_{\mathbf{z}}\mathbf{x}_T \leq \mathbf{C}_{\mathbf{z}}\Delta\mathbf{x}_{k+i} \leq \mathbf{z}_{ub} - \mathbf{C}_{\mathbf{z}}\mathbf{x}_T$. It can be represented in matrix form as:

$$\begin{bmatrix} \mathbf{C}_{\mathbf{z}} \\ -\mathbf{C}_{\mathbf{z}} \end{bmatrix} \Delta\mathbf{x}_{k+i} \leq \begin{bmatrix} \left(\mathbf{z}_{ub} - \mathbf{C}_{\mathbf{z}}\mathbf{x}_T\right) \\ -\left(\mathbf{z}_{lb} - \mathbf{C}_{\mathbf{z}}\mathbf{x}_T\right) \end{bmatrix} \tag{3.13}$$

With proper arrangement and substitution of $\Delta\mathbf{X}_k$ from (3.5), the constraints can be expressed in terms of $\Delta\mathbf{U}_k$ as:

$$\mathcal{C}_{\mathbf{z}}\left(\mathbf{F}\Delta\mathbf{x}_k + \mathbf{H}\Delta\mathbf{U}_k\right) \leq \Delta\mathbf{Z}_b \tag{3.14}$$

where

$$\mathcal{C}_{\mathbf{z}} = \begin{bmatrix} \begin{bmatrix} \mathbf{C}_{\mathbf{z}} \\ -\mathbf{C}_{\mathbf{z}} \end{bmatrix} & & & \\ & \begin{bmatrix} \mathbf{C}_{\mathbf{z}} \\ -\mathbf{C}_{\mathbf{z}} \end{bmatrix} & & \\ & & \ddots & \\ & & & \begin{bmatrix} \mathbf{C}_{\mathbf{z}} \\ -\mathbf{C}_{\mathbf{z}} \end{bmatrix} \end{bmatrix}$$

$$\Delta\mathbf{Z}_b = \begin{bmatrix} \begin{bmatrix} (\mathbf{z}_{ub} - \mathbf{C}_{\mathbf{z}}\mathbf{x}_T) \\ -(\mathbf{z}_{lb} - \mathbf{C}_{\mathbf{z}}\mathbf{x}_T) \end{bmatrix} \\ \\ \begin{bmatrix} (\mathbf{z}_{ub} - \mathbf{C}_{\mathbf{z}}\mathbf{x}_T) \\ -(\mathbf{z}_{lb} - \mathbf{C}_{\mathbf{z}}\mathbf{x}_T) \end{bmatrix} \\ \vdots \\ \begin{bmatrix} (\mathbf{z}_{ub} - \mathbf{C}_{\mathbf{z}}\mathbf{x}_T) \\ -(\mathbf{z}_{lb} - \mathbf{C}_{\mathbf{z}}\mathbf{x}_T) \end{bmatrix} \end{bmatrix}$$

3.3.4 Combined Input and State Constraints

Both the input and output variable constraints can be integrated into one equation as shown below.

$$\begin{aligned} & & \mathbf{M}_{\mathbf{U}}\Delta\mathbf{U}_k &\leq \Delta\mathbf{U}_b & & \text{from Eq. (3.10)} \\ & & \mathcal{C}_{\mathbf{z}}\left(\mathbf{F}\Delta\mathbf{x}_k + \mathbf{H}\Delta\mathbf{U}_k\right) &\leq \Delta\mathbf{Z}_b & & \text{from Eq. (3.14)} \\ \text{or} & & \mathcal{C}_{\mathbf{z}}\mathbf{H}\Delta\mathbf{U}_k &\leq \Delta\mathbf{Z}_b - \mathcal{C}_{\mathbf{z}}\mathbf{F}\Delta\mathbf{x}_k & & \\ \therefore & & \mathbf{\Gamma}\Delta\mathbf{U}_k &\leq \mathbf{\Upsilon} & & \qquad (3.15) \end{aligned}$$

where $\boldsymbol{\Upsilon}$ is a matrix containing both input and output variable constraints and

$$\boldsymbol{\Gamma} = \begin{bmatrix} \mathbf{M}_{\mathbf{U}} \\ \mathbf{M}_{\dot{\mathbf{U}}} \\ \mathcal{C}_{\mathbf{z}}\mathbf{H} \end{bmatrix}, \quad \boldsymbol{\Upsilon} = \begin{bmatrix} \Delta\mathbf{U}_b \\ \dot{\mathbf{U}}_b \\ \Delta\mathbf{Z}_b - \mathcal{C}_{\mathbf{z}}\mathbf{F}\Delta\mathbf{x}_k \end{bmatrix}$$

3.4 Stability

See Appendix section (9.1).

3.5 Simulation Setup

The above algorithm is implemented in simulation on a quadcopter. The 6-degree of freedom equations of motion were linearized at a certain hover position denoted as

$$\mathbf{x}_T = [\mathbf{p}^T\ \mathbf{0}_{1\times3}\ \mathbf{0}_{1\times3}\ \mathbf{0}_{1\times3}]^T, \quad \mathbf{u}_T = [\frac{mg}{4}\ \frac{mg}{4}\ \frac{mg}{4}\ \frac{mg}{4}]^T$$

where $\mathbf{p} \in \Re^3$ is any constant position. Here, $\mathbf{p} = \mathbf{0}_{3\times1}$ is used. $\mathbf{u}_T$ is the thrust values needed to maintain hover. And the corresponding matrices $\mathbf{A}_c$ and $\mathbf{B}_c$ are:

$$\mathbf{A_c} = \begin{bmatrix} \mathbf{0}_{3\times3} & \mathbf{I}_{3\times3} & \mathbf{0}_{3\times3} & \mathbf{0}_{3\times3} \\ \mathbf{0}_{3\times3} & \mathbf{0}_{3\times3} & \mathbf{A}_{\mathbf{c},\boldsymbol{\Theta}} & \mathbf{0}_{3\times3} \\ \mathbf{0}_{3\times3} & \mathbf{0}_{3\times3} & \mathbf{0}_{3\times3} & \mathbf{I}_{3\times3} \\ \mathbf{0}_{3\times3} & \mathbf{0}_{3\times3} & \mathbf{0}_{3\times3} & \mathbf{0}_{3\times3} \end{bmatrix}$$

where

$$\mathbf{A}_{\mathbf{c},\boldsymbol{\Theta}} = \begin{bmatrix} 0 & -g & 0 \\ g & 0 & 0 \\ 0 & 0 & 0 \end{bmatrix}$$

and $\mathbf{B_c}$ matrix is the same as in Eq. (2.35).

Simulation results are combined with NMPC and shown in section (4.6).

CHAPTER 4

Nonlinear Model Predictive Control

4.1 State Dependent Coefficient based Pseudo-Linear Quadratic Control Formulation

In this section, the State Dependent Riccati Equation (SDRE) formulation is summarized. The essential idea is to utilize the State Dependent Coefficient (SDC) factorization (Ref. [61]) of the nonlinear dynamics. The SDC based control technique provides a sub-optimal solution to the control problem, but without imposing any state and/or input constraints. A state space representation of the quadcopter is obtained, where each of its system matrices are now expressed as functions of the current state (Ref. [62]). This method uses the quadratic form performance index to solve for an infinite horizon optimal problem, for which its solution is locally stable (Ref. [63]). This can be considered to be an improvement over the Linear Quadratic Control formulation for the linear time invariant (LTI) systems (see Ref. [64]). The developments in this section are key to the NMPC solution discussed in the next section. Consider a dynamic system as in (3.1), The SDRE formulation involves transforming the nonlinear system into the following state space form,

$$\dot{\mathbf{x}} \;=\; \mathbf{A_c}(\mathbf{x})\mathbf{x} + \mathbf{B_c}(\mathbf{x})\mathbf{u} \tag{4.1}$$

Where $\mathbf{x} \in \Re^n$ is the state vector, $\mathbf{u} \in \Re^m$ is the input vector, and $\mathbf{A_c}(\mathbf{x}) \in \Re^{n\times n}$ and $\mathbf{B_c}(\mathbf{x}) \in \Re^{n\times m}$ are the pseudo-linear system matrices in the SDC form and the pairs $(\mathbf{A_c}(\mathbf{x}), \mathbf{B_c}(\mathbf{x}))$ are stabilizable $\forall \mathbf{x} \in \Re^n$. It is important to note that this representation of $\mathbf{A_c}(\mathbf{x})$ and $\mathbf{B_c}(\mathbf{x})$ is not unique in general, except for a scalar system. Different state dependent coefficient matrices can be obtained from the equations of motion, and a solution to the optimization problem may or may not exist. *However, we derive a particular factorization (shown later) and the existence of the sub-optimal controller is assumed.* The SDC LQT (linear quadratic tracker) formulation minimizes the infinite horizon objective function:

$$\min\ J(\mathbf{x},\mathbf{u}) = \int_0^\infty \left[(\mathbf{x}-\mathbf{x}^r)^T\mathbf{Q}(\mathbf{x})(\mathbf{x}-\mathbf{x}^r) + \mathbf{u}^T\mathbf{R}(\mathbf{x})\mathbf{u}\right] dt \tag{4.2}$$

Where $\mathbf{Q}(\mathbf{x}) \geq 0$ is a positive semidefinite matrix and $\mathbf{R}(\mathbf{x}) > 0$ is a positive definite matrix with appropriate dimensions. The discrete-time equivalent of Eq. (4.1) is obtained by using a zero-order-hold (zoh) with a specified sample time. Let the discrete-time equivalent of the system be of the following form:

$$\mathbf{x}_{k+1} \ = \ \mathbf{A}(\mathbf{x}_k)\mathbf{x}_k + \mathbf{B}(\mathbf{x}_k)\mathbf{u}_k \tag{4.3}$$

Where, $\mathbf{A}(\mathbf{x}_k)$ and $\mathbf{B}(\mathbf{x}_k)$ are discrete approximations of the continuous $\mathbf{A_c}(\mathbf{x})$ and $\mathbf{B_c}(\mathbf{x})$ respectively.

From the description of the quadcopter, we note that $\mathbf{B}(\mathbf{x}_k) = \mathbf{B}$ (a constant matrix). Since Eq. (4.3) is of the pseudo-linear form, the system matrices can be considered to be constants for each sampling interval i.e.,$[t_k,\ t_{k+1})$ with $t_{k+1} - t_k = \Delta t$ (Ref. [65]). The SDC LQT optimization problem then requires the solution for the Discrete-time Algebraic Riccati Equation (DARE) which is of the form:

$$\begin{aligned}&\mathbf{Q}(\mathbf{x_k}) + \mathbf{A}(\mathbf{x}_k)^T\mathbf{P}(\mathbf{x_k})\mathbf{A}(\mathbf{x}_k) - \mathbf{P}(\mathbf{x_k})\\ &-\mathbf{A}(\mathbf{x}_k)^T\mathbf{P}(\mathbf{x_k})\mathbf{B}(\mathbf{x}_k)\left(\mathbf{B}(\mathbf{x}_k)^T\mathbf{P}(\mathbf{x_k})\mathbf{B}(\mathbf{x}_k) + \mathbf{R}(\mathbf{x_k})\right)^{-1}\mathbf{B}(\mathbf{x}_k)^T\mathbf{P}(\mathbf{x_k})\mathbf{A}(\mathbf{x}_k) = 0\end{aligned}$$

where $\mathbf{Q}_k$ and $\mathbf{R}_k$ are discrete equivalents of the weight matrices. The solution $\mathbf{P}(\mathbf{x}_k)$ from Eq. (4.4) is then used to compute the Kalman gain sequence as shown in Ref. [66].

$$\mathbf{K}(\mathbf{x_k}) = \left(\mathbf{B}(\mathbf{x}_k)^T\mathbf{P}(\mathbf{x_k})\mathbf{B}(\mathbf{x}_k) + \mathbf{R}(\mathbf{x_k})\right)^{-1}\mathbf{B}(\mathbf{x}_k)^T\mathbf{P}(\mathbf{x_k})\mathbf{A}(\mathbf{x}_k)$$

So that

$$\mathbf{u}_k = -\mathbf{K}(\mathbf{x_k})(\mathbf{x}_k - \mathbf{x}_k^r) \tag{4.4}$$

is the solution to the SDC LQT control problem.

4.2 SDC Representation of the Quadcopter Dynamics

It should be noted that in Eq. (4.1), the first term $\mathbf{A_c}(\mathbf{x})\mathbf{x}$ vanishes if $\mathbf{x} = \mathbf{0}$. For the quadcopter case (2.36), $f(\mathbf{x})$ doesn't completely vanish when $\mathbf{x} = \mathbf{0}$. Thus a slight change is needed in Eq. (4.1) to account for this. The quadcopter system will be transformed into the form:

$$\dot{\mathbf{x}} \;=\; \mathbf{A_c}(\mathbf{x})\mathbf{x} + \mathbf{B_c}(\mathbf{x})\delta\mathbf{u} \tag{4.5}$$

where $\delta\mathbf{u} = \mathbf{u} - \mathbf{u}_T$ and $\mathbf{u}_T$ represents the constant force term that is required to balance the weight of the quadcopter. It is assumed that the quadcopter starts from an initial equilibrium state. The new control design then focuses on synthesis of $\delta\mathbf{u}$. One possible way to factorize the equations of motion into the SDC form is presented in Eq. (4.6).

$$\mathbf{A_c}(\mathbf{x}) = \begin{bmatrix} \mathbf{0}_{3\times3} & \mathbf{R}_{BI}^T & \mathbf{0}_{3\times3} & \mathbf{0}_{3\times3} \\ \mathbf{0}_{3\times3} & \mathbf{A}_{22} & \mathbf{A}_{23} & \mathbf{A}_{24} \\ \mathbf{0}_{3\times3} & \mathbf{0}_{3\times3} & \mathbf{0}_{3\times3} & \mathbf{W} \\ \mathbf{0}_{3\times3} & \mathbf{0}_{3\times3} & \mathbf{0}_{3\times3} & \mathbf{A}_{44} \end{bmatrix} \tag{4.6}$$

Where

$$\mathbf{A}_{22} = \begin{bmatrix} 0 & \frac{r}{2} & -\frac{q}{2} \\ -\frac{r}{2} & 0 & \frac{p}{2} \\ \frac{q}{2} & -\frac{p}{2} & 0 \end{bmatrix}, \quad \mathbf{A}_{23} = \begin{bmatrix} 0 & -g\frac{\sin\theta}{\theta} & 0 \\ g\frac{\cos\theta\sin\phi}{\phi} & 0 & 0 \\ g\frac{(\cos\theta+1)(\cos\phi-1)}{2\phi} & g\frac{(\cos\phi+1)(\cos\theta-1)}{2\theta} & 0 \end{bmatrix}$$

$$\mathbf{A}_{24} = \begin{bmatrix} 0 & -\frac{w}{2} & \frac{v}{2} \\ \frac{w}{2} & 0 & -\frac{u}{2} \\ -\frac{v}{2} & \frac{u}{2} & 0 \end{bmatrix}, \quad \mathbf{A}_{44} = \begin{bmatrix} 0 & \frac{(J_y - J_z)r}{2J_x} & \frac{(J_y - J_z)q}{2J_x} \\ \frac{(J_z - J_x)r}{2J_y} & 0 & \frac{(J_z - J_x)p}{2J_y} \\ \frac{(J_x - J_y)q}{2J_z} & \frac{(J_x - J_y)p}{2J_z} & 0 \end{bmatrix}$$

and $\mathbf{B_c}$ is the same as in Eq. (2.35). But one issue that arises from using this form is that some terms in matrix $\mathbf{A}_{23}$ become infinite when the denominators become 0. To prevent this from happening, the first three terms of Taylor series expansions of $\sin\theta$, $\sin\phi$, $(\cos\theta - 1)$, and $(\cos\phi - 1)$ are used here to provide a close approximation (In reality, the approximations only differ from their true values 4% at most in the range of $(-\frac{\pi}{2},\ \frac{\pi}{2})$). By doing this, $\mathbf{A}_{23}$ can be expressed as shown in (4.7).

$$\mathbf{A}_{23} = \begin{bmatrix} 0 & -g(1 - \frac{\theta^2}{3!} + \frac{\theta^4}{5!}) & 0 \\ g\cos\theta(1 - \frac{\phi^2}{3!} + \frac{\phi^4}{5!}) & 0 & 0 \\ g\frac{\cos\theta + 1}{2}(-\frac{\phi}{2} + \frac{\phi^3}{4!} - \frac{\phi^5}{6!}) & g\frac{\cos\phi + 1}{2}(-\frac{\theta}{2} + \frac{\theta^3}{4!} - \frac{\theta^5}{6!}) & 0 \end{bmatrix} \tag{4.7}$$

Considering the specific aspects of the quadcopter as detailed above, the discretized system for the derivation of NMPC is summarized below:

$$\mathbf{x}_{k+1} = \mathbf{A}(\mathbf{x_k})\mathbf{x}_k + \mathbf{B}\,\delta\mathbf{u_k} \tag{4.8}$$

Where we have used the fact that for this system $\mathbf{B}(\mathbf{x}_k) = \mathbf{B}$ (a constant matrix). Following the same approach as outlined for the linear MPC, we arrive at the N step state prediction equations (similar to Eq. (3.5)):

$$\mathbf{X}_k = \mathbf{F}(\mathbf{x_k})\mathbf{x}_k + \mathbf{H}(\mathbf{x_k})\Delta\mathbf{U}_k \tag{4.9}$$

Where,

$$\mathbf{X}_k = \begin{bmatrix} \mathbf{x}_k \\ \mathbf{x}_{k+1} \\ \mathbf{x}_{k+2} \\ \vdots \\ \mathbf{x}_{k+N-1} \end{bmatrix}, \quad \Delta\mathbf{U}_k = \begin{bmatrix} \delta\mathbf{u}_k \\ \delta\mathbf{u}_{k+1} \\ \delta\mathbf{u}_{k+2} \\ \vdots \\ \delta\mathbf{u}_{k+N-1} \end{bmatrix}, \quad \mathbf{F}(\mathbf{x_k}) = \begin{bmatrix} \mathbf{I} \\ \mathbf{A}(\mathbf{x_k}) \\ \mathbf{A}(\mathbf{x_k})^2 \\ \vdots \\ \mathbf{A}(\mathbf{x_k})^{N-1} \end{bmatrix}$$

$$\mathbf{H}(\mathbf{x_k}) = \begin{bmatrix} \mathbf{0} & & & & \\ \mathbf{B} & \mathbf{0} & & & \\ \mathbf{A}(\mathbf{x_k})\mathbf{B} & \mathbf{B} & \mathbf{0} & & \\ \vdots & \vdots & \vdots & \ddots & \\ \mathbf{A}(\mathbf{x_k})^{N-2}\mathbf{B} & \mathbf{A}(\mathbf{x_k})^{N-3}\mathbf{B} & \cdots & \mathbf{B} & \mathbf{0} \end{bmatrix}$$

Similarly, the terminal state is given by:

$$\mathbf{x}_{k+N} = \mathbf{A}(\mathbf{x_k})^N \mathbf{x}_k + \bar{\mathbf{B}}(\mathbf{x_k})\Delta\mathbf{U}_k$$

where

$$\bar{\mathbf{B}}(\mathbf{x_k}) \;=\; \left[\mathbf{A}(\mathbf{x_k})^{N-1}\mathbf{B}\ \mathbf{A}(\mathbf{x_k})^{N-2}\mathbf{B}\ \cdots\ \mathbf{A}(\mathbf{x_k})\mathbf{B}\ \ \mathbf{B}\right]$$

4.3 Controller Design

The nonlinear MPC uses the same algorithm as the linear MPC to calculate the solution to the minimization of the cost function outlined in Eq. (3.6). The main difference between the linear and nonlinear version of the MPC is that now, the objective function depends on the current states of the system and needs to be calculated at the start of each sample interval. The objective function Eq. (3.6) then becomes:

$$\begin{aligned} J(\mathbf{x}_k, \mathbf{U}_k) &= \Delta\mathbf{U}_k^T \left(\mathbf{H}(\mathbf{x_k})^T\bar{\mathbf{Q}}\mathbf{H}(\mathbf{x_k}) + \bar{\mathbf{R}} + \bar{\mathbf{B}}(\mathbf{x_k})^T\mathbf{Q}_f\bar{\mathbf{B}}(\mathbf{x_k})\right)\Delta\mathbf{U}_k \\ &+ 2\left[(\mathbf{F}(\mathbf{x_k})\mathbf{x_k} - \mathbf{X}_k^r)^T\,\bar{\mathbf{Q}}\mathbf{H}(\mathbf{x_k}) + (\mathbf{A}(\mathbf{x_k})^N\mathbf{x}_k - \mathbf{x}_{k+N}^r)^T\mathbf{Q}_f\bar{\mathbf{B}}(\mathbf{x_k})\right]\Delta\mathbf{U}_k \\ &+ (\mathbf{F}(\mathbf{x_k})\mathbf{x}_k - \mathbf{X}_k^r)^T\bar{\mathbf{Q}}(\mathbf{F}(\mathbf{x_k})\mathbf{x}_k - \mathbf{X}_k^r) \\ &+ (\mathbf{A}(\mathbf{x_k})^N\mathbf{x}_k - \mathbf{x}_{k+N}^r)^T\mathbf{Q}_f(\mathbf{A}(\mathbf{x_k})^N\mathbf{x}_k - \mathbf{x}_{k+N}^r) \end{aligned} \tag{4.10}$$

The cost function seen to be quasi-quadratic and the regular quadratic programming method is used to solve for the control (Eq. (4.10)).

4.4 Constraints

Similar to Eq. (3.15), the input and output variable constraints are integrated into one equation of the form

$$\mathbf{\Gamma}(\mathbf{x_k})\Delta\mathbf{U}_k \leq \mathbf{\Upsilon}(\mathbf{x_k}) \tag{4.11}$$

where $\mathbf{\Upsilon}(\mathbf{x_k})$ is a matrix containing both input and output variable constraints and

$$\mathbf{\Gamma}(\mathbf{x_k}) = \begin{bmatrix} \mathbf{M}_{\mathbf{U}} \\ \mathbf{M}_{\dot{\mathbf{U}}} \\ \mathcal{C}_{\mathbf{z}}(\mathbf{x_k})\mathbf{H}(\mathbf{x_k}) \end{bmatrix}, \quad \mathbf{\Upsilon}(\mathbf{x_k}) = \begin{bmatrix} \Delta\mathbf{U}_b \\ \dot{\mathbf{U}}_b \\ \mathbf{Z}_b - \mathcal{C}_{\mathbf{z}}\,(\mathbf{x_k})(\mathbf{F}(\mathbf{x_k})\mathbf{x}_k + \mathbf{H}(\mathbf{x}_k)\mathbf{U}_T) \end{bmatrix}$$

Where,

$$\Delta\mathbf{U}_b = \begin{bmatrix} \begin{bmatrix} (\mathbf{u}_{ub} - \mathbf{u}_T) \\ -(\mathbf{u}_{lb} - \mathbf{u}_T) \end{bmatrix} \\ \\ \begin{bmatrix} (\mathbf{u}_{ub} - \mathbf{u}_T) \\ -(\mathbf{u}_{lb} - \mathbf{u}_T) \end{bmatrix} \\ \vdots \\ \begin{bmatrix} (\mathbf{u}_{ub} - \mathbf{u}_T) \\ -(\mathbf{u}_{lb} - \mathbf{u}_T) \end{bmatrix} \end{bmatrix}, \qquad \mathbf{Z}_b = \begin{bmatrix} \begin{bmatrix} \mathbf{z}_{ub} \\ -\mathbf{z}_{lb} \end{bmatrix} \\ \\ \begin{bmatrix} \mathbf{z}_{ub} \\ -\mathbf{z}_{lb} \end{bmatrix} \\ \vdots \\ \begin{bmatrix} \mathbf{z}_{ub} \\ -\mathbf{z}_{lb} \end{bmatrix} \end{bmatrix}$$

4.5 Stability and State Convergence

See Appendix section (9.2).

4.6 Simulation Results

For both LMPC and NMPC cases, the system dynamics presented in (2.36) subject to constraints (2.37) and (2.39) is simulated. Three different reference trajectories are generated to evaluate the effectiveness of the control laws: a) Helical, b) Spherical Spiral, and c) Straight Line Segments as shown in Fig. (4.1), (4.2), (4.3), (4.4), (4.5) and (4.6).

In order to further assess the effectiveness of the NMPC scheme, simulations were performed with simultaneous high frequency disturbance torques injected in the roll and pitch channels starting at 5s. The disturbances are two exponentially decaying sinusoidal functions during $t \in (5, 10)(s)$:

$$\delta\boldsymbol{\tau} = e^{-0.1t} \left[\sin(10t)\ \cos(10t)\ 0\right]^T$$

The disturbance is used to approximate the effects of wind during typical outdoor flight conditions. The same reference trajectories were being tracked and the results are shown in Fig. (4.7), (4.8), (4.9), (4.10), (4.11), and (4.12).

Varying prediction horizons for linear and nonlinear MPC are performed and their performances are summarized in the Table (4.1) and (4.2). The average RMS is calculated by using:

$$\text{Average RMS} = \sqrt{\frac{1}{N_t}\sum_{i=1}^{N_t} \|\mathbf{p}_i - \mathbf{p}_i^r\|_2^2}$$

where N_t is the size of the entire simulation horizon. The average control effort is calculated by using:

$$\text{Average Control Effort} = \frac{1}{N_t}\sum_{i=1}^{N_t} \Delta\mathbf{u}_i^T \Delta\mathbf{u}_i$$

The data shows that nonlinear MPC yields better position tracking and uses less control effort.

Table 4.1. RMS of Position Tracking Errors

Trajectory	Nonlinear/Linear	N = 10	N = 15	N = 25	N = 35	N = 45
Helical	Nonlinear	0.5663	0.5824	0.5831	0.5757	0.5832
Helical	Linear	0.8075	0.8348	0.8484	0.8364	0.8334
Spherical Spiral	Nonlinear	1.4739	1.4725	1.4725	1.4720	1.4726
Spherical Spiral	Linear	1.4984	1.5000	1.5016	1.5009	1.5009
Straight Line	Nonlinear	0.3112	0.3020	0.3169	0.3160	0.3155
Straight Line	Linear	0.3722	0.3575	0.3633	0.3612	0.3603

4.7 Concluding Remarks

Fig. (4.1) and (4.2) shows the reference trajectory and the tracking errors for both the linear and nonlinear MPC cases tracking a helical trajectory. It can be

Table 4.2. Average Control Effort

Trajectory	Nonlinear/Linear	N = 10	N = 15	N = 25	N = 35	N = 45
Helical	Nonlinear	0.3838	0.3724	0.3652	0.3665	0.3674
Helical	Linear	0.4205	0.4107	0.4057	0.4084	0.4114
Spherical Spiral	Nonlinear	0.1687	0.1689	0.1700	0.1707	0.1710
Spherical Spiral	Linear	0.1910	0.1913	0.1914	0.1913	0.1913
Straight Line	Nonlinear	0.2125	0.1433	0.1282	0.1280	0.1296
Straight Line	Linear	0.2245	0.1593	0.1388	0.1379	0.1367

seen that both controllers achieve good tracking, however the transient response of the nonlinear MPC is better. Clearly the speed of response with regards to position tracking errors is much better the nonlinear MPC.

Fig. (4.3) and (4.4) shows the results of tracking a spherical spiral trajectory. As with previous case, both the controllers achieve successful tracking.

Fig. (4.5) and (4.6) shows the results of tracking a straight line trajectory segments between two waypoints. As with the previous cases, successful tracking are achieved for both controllers. And the superior transient performance of the nonlinear MPC is clearly seen.

Figures (4.7), (4.8), (4.9), (4.10), (4.11), and (4.12) showed the performance of linear and nonlinear MPC subject to disturbances. Under disturbances, linear MPC and nonlinear MPC both performed well with acceptable amount of tracking errors and kept the input within bounds. The controllers provide a feedforward cancellation of the input disturbance. It can be seen from the figures that the nonlinear MPC significantly outperforms the linear MPC regards to transient performance.

4.8 Comparison with a Feedback Linearization Controller

In [67], a feedback-linearization based nonlinear controller is used to control a quadcopter. A comparison of feedback-linearization and MPC is made here to showcase the effectiveness of MPC in handling constraints.

The quadcopter equations of motion can be represented in the alternative form as below:

$$\dot{\mathbf{x}} = \begin{bmatrix} x_4 \\ x_5 \\ x_6 \\ (S_\theta C_\psi C_\phi + S_\psi S_\phi)\frac{T}{m} \\ (S_\theta S_\psi C_\phi - C_\psi S_\phi)\frac{T}{m} \\ g + C_\theta C_\phi \frac{T}{m} \\ x_7 \\ x_8 \\ x_9 \\ x_8 x_9 \frac{J_y - J_z}{J_x} + \frac{\tau_1}{J_x} \\ x_7 x_9 \frac{J_z - J_x}{J_y} + \frac{\tau_2}{J_y} \\ x_7 x_8 \frac{J_x - J_y}{J_z} + \frac{\tau_3}{J_z} \end{bmatrix} \tag{4.12}$$

where

$$\mathbf{x} = \begin{bmatrix} x_1 \\ x_2 \\ x_3 \\ x_4 \\ x_5 \\ x_6 \\ x_7 \\ x_8 \\ x_9 \\ x_{10} \\ x_{11} \\ x_{12} \end{bmatrix} = \begin{bmatrix} x \\ y \\ z \\ \dot{x} \\ \dot{y} \\ \dot{z} \\ \phi \\ \theta \\ \psi \\ \dot{\phi} \\ \dot{\theta} \\ \dot{\psi} \end{bmatrix}$$

A two loop control system structure is used here with the position loop as the outer loop and the attitude loop as the inner loop. The outer loop generates attitude command for inner loop to track. Detailed stability proof can be found in [67]. The control command can be generated by (assuming $\psi^d = 0$):

$$\ddot{x}_1 = \tilde{u}_1 = -k_{1,i}(x_1 - x_1^d) - k_{1,d}(x_4 - x_4^d) \tag{4.13}$$

$$\ddot{x}_2 = \tilde{u}_2 = -k_{2,i}(x_2 - x_2^d) - k_{2,d}(x_5 - x_5^d) \tag{4.14}$$

$$\ddot{x}_3 = \tilde{u}_3 = -k_{3,i}(x_3 - x_3^d) - k_{3,d}(x_6 - x_6^d) \tag{4.15}$$

$$T^d = \sqrt{(\tilde{u}_3 - g)^2 + \tilde{u}_1^2 + \tilde{u}_2^2} \tag{4.16}$$

$$\theta^d = \arctan(\frac{\tilde{u}_1}{\tilde{u}_3 - g}) \tag{4.17}$$

$$\phi^d = \arcsin(\frac{\tilde{u}_2 m}{T^d}) \tag{4.18}$$

The command ϕ^d, θ^d, ψ^d can be used for inner loop tracking.

$$\tau_1 = J_x \left(-x_8 x_9 \frac{J_y - J_z}{J_x} - k_{7,i}(x_7 - x_7^d) - k_{7,d} x_{10} \right) \tag{4.19}$$

$$\tau_2 = J_y \left(-x_7 x_9 \frac{J_z - J_x}{J_y} - k_{8,i}(x_8 - x_8^d) - k_{8,d} x_{11} \right) \tag{4.20}$$

$$\tau_3 = J_z \left(-x_7 x_8 \frac{J_x - J_y}{J_z} - k_{9,i}(x_9 - x_9^d) - k_{9,d} x_{12} \right) \tag{4.21}$$

The gains used for feedback linearization is fine tuned so that it has comparable performance compared to MPC as shown in Fig. (4.16) (Going from $\mathbf{p} = [1, 1, 1] \rightarrow \mathbf{p} = [0, 0, 0]$) and (4.18) (Going from $\mathbf{p} = [2, 2, 2] \rightarrow \mathbf{p} = [0, 0, 0]$). Good tracking performance is observed in both controllers with feedback linearization with slightly faster response and mild overshoot.

The same condition is simulated under a different set of actuator constraints $0 \leq F_i \leq 2.45$ with other setting as before. Results are shown in Fig. (4.20) and (4.22). It can be observed from Fig. (4.25) and (4.23) that feedback linearization controller's input saturated much more than LMPC.

When the starting position shifted to $\mathbf{p} = [3, 3, 3]$, the feedback linearization base controller becomes unstable while LMPC continues to track properly as shown in Fig. (4.24). This is due to the input and state violation that were committed by the feedback linearization controller.

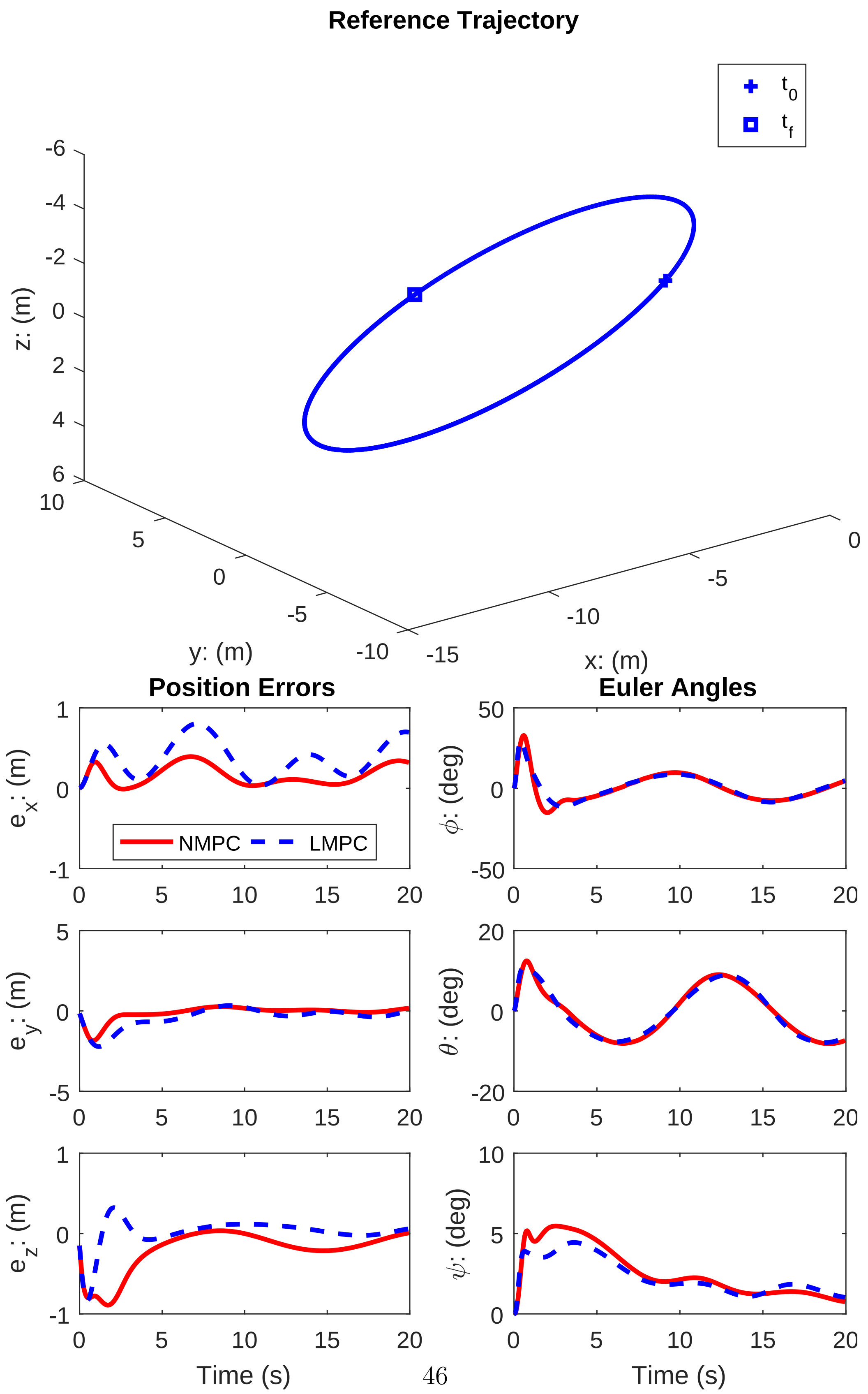


Figure 4.1. Trajectory 1: Helical Trajectory.

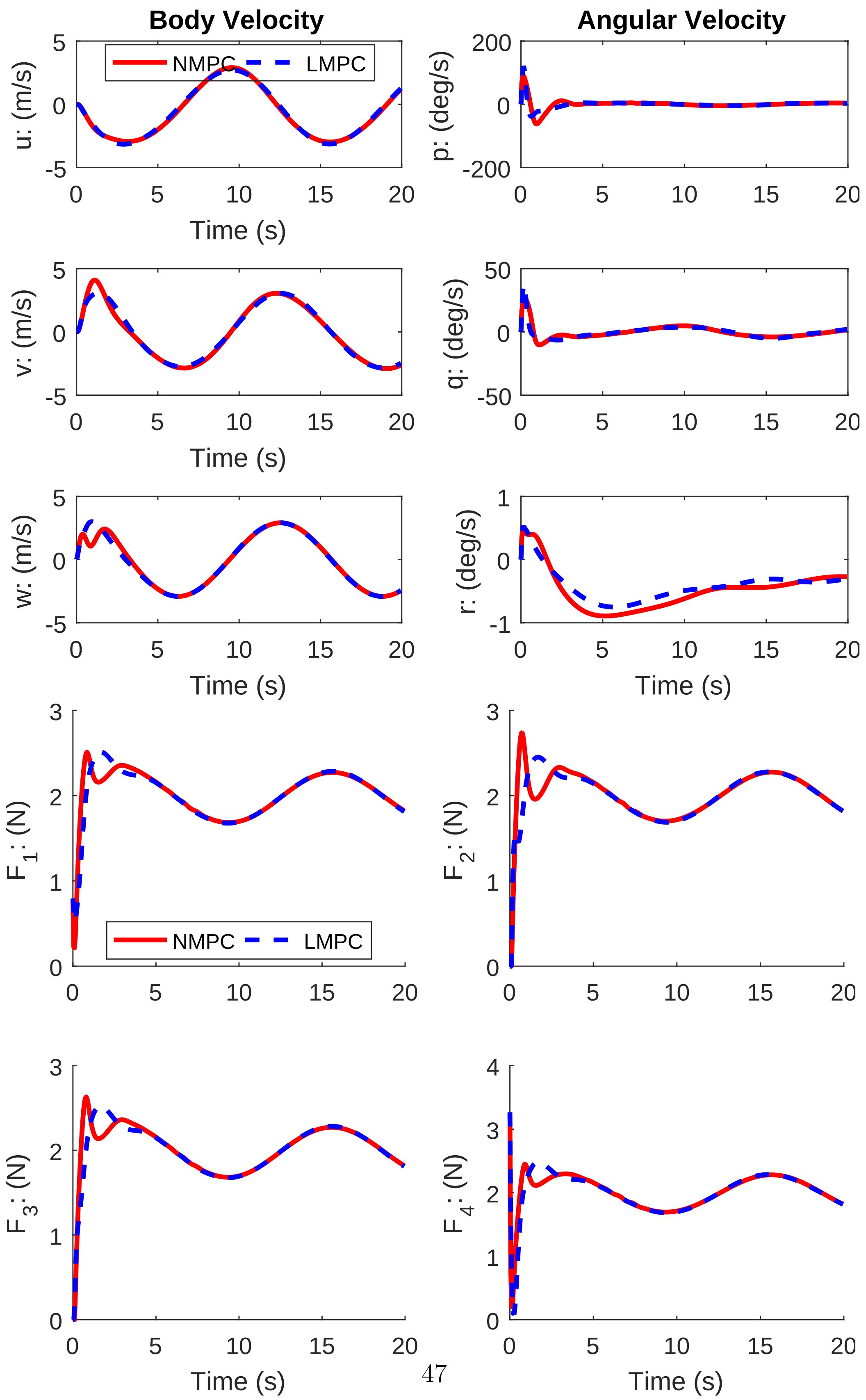


Figure 4.2. Trajectory 1: Helical Trajectory.

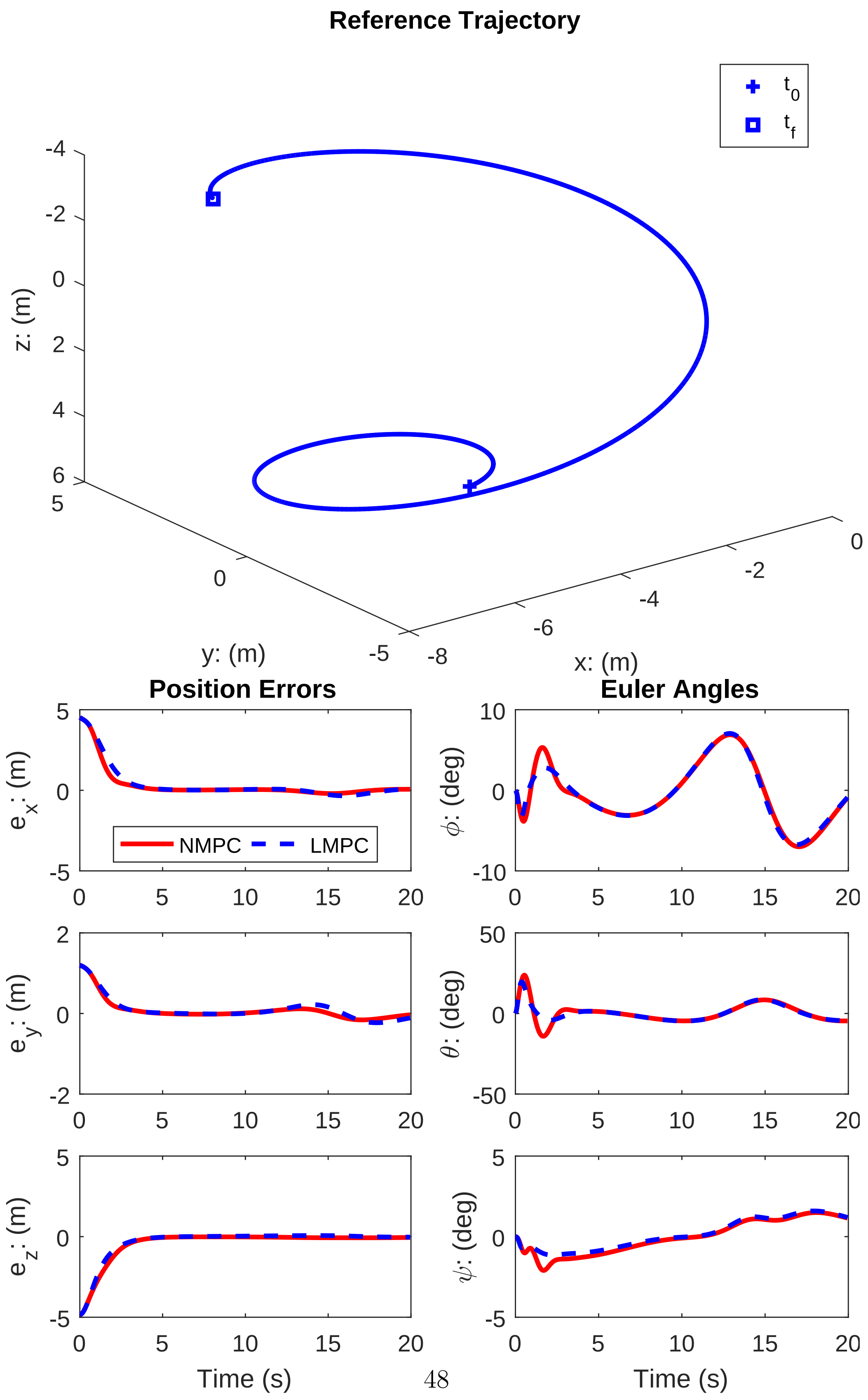


Figure 4.3. Trajectory 2: Spherical Spiral Trajectory.

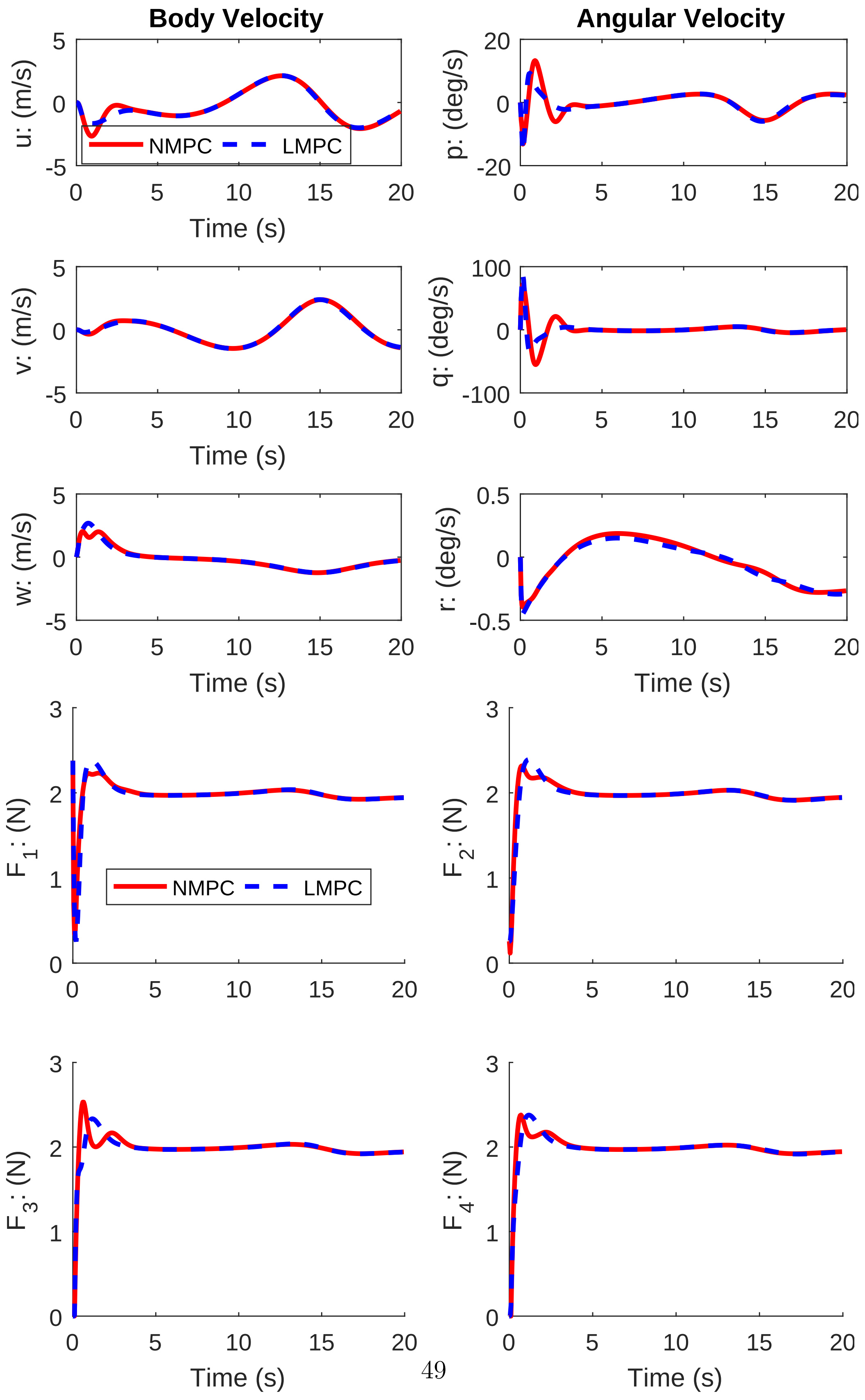


Figure 4.4. Trajectory 2: Spherical Spiral Trajectory.

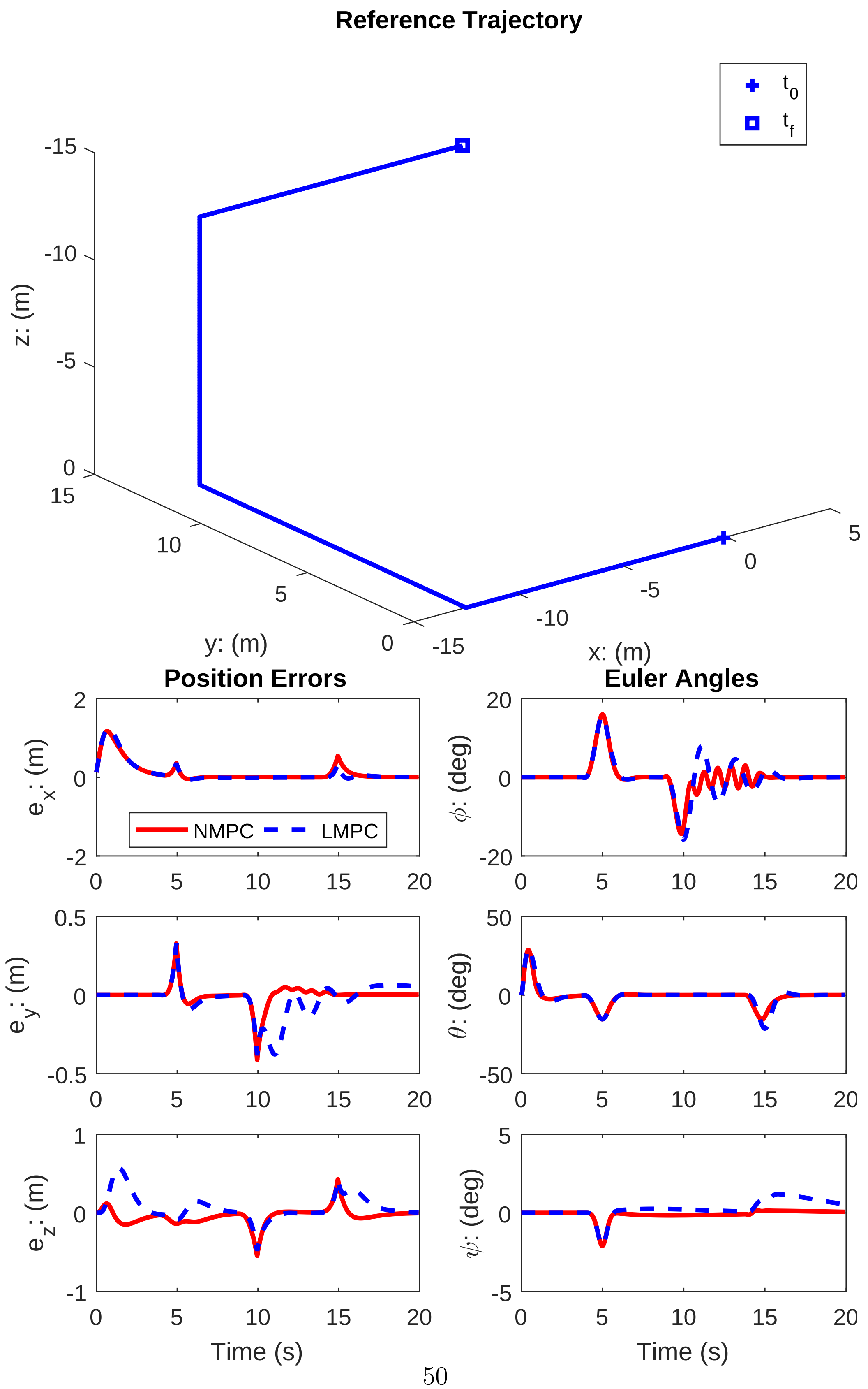


Figure 4.5. Trajectory 3: Straight Line Trajectory.

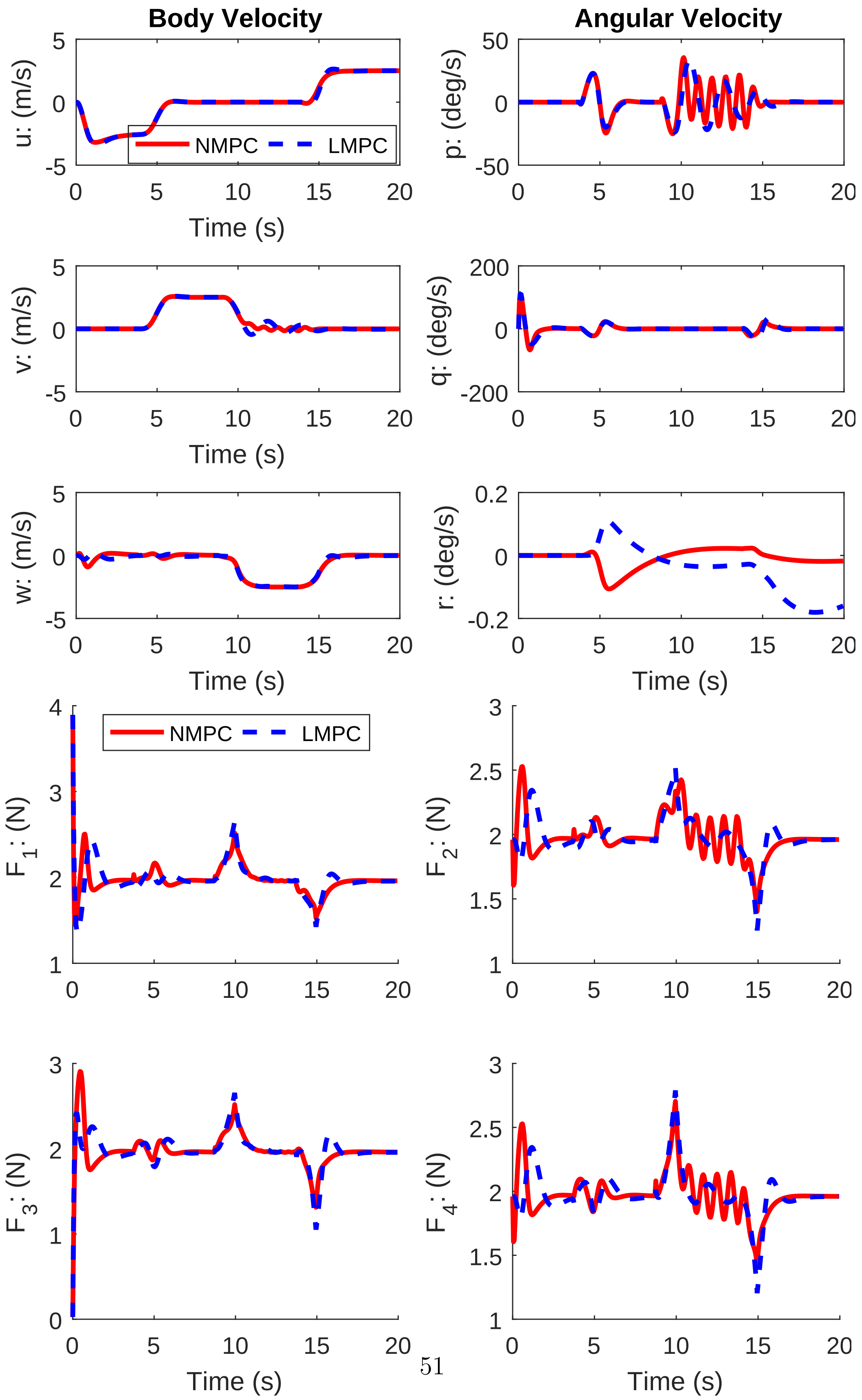


Figure 4.6. Trajectory 3: Straight Line Trajectory.

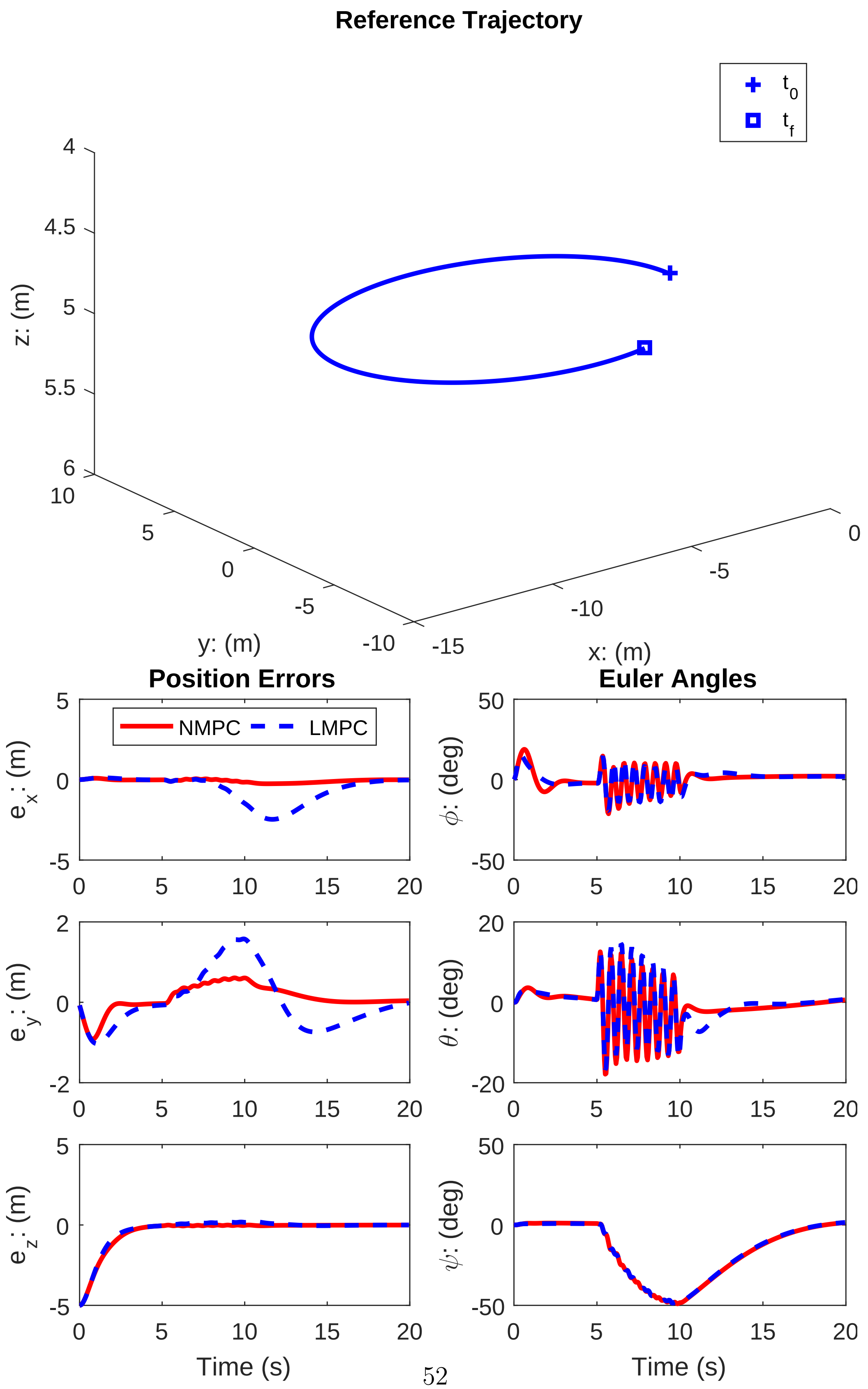


Figure 4.7. Trajectory 1: Helical Trajectory (With Disturbance).

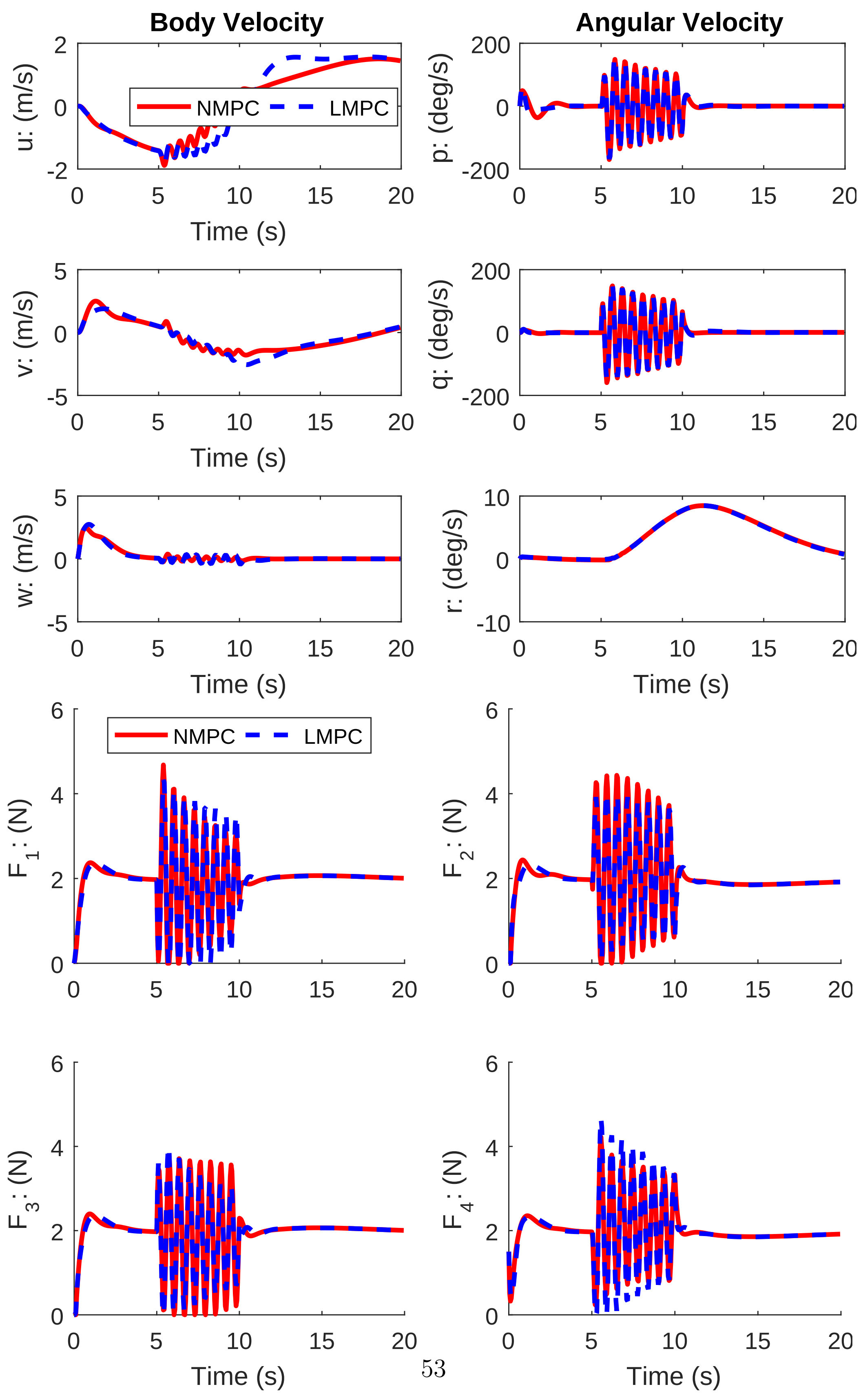


Figure 4.8. Trajectory 1: Helical Trajectory (With Disturbance).

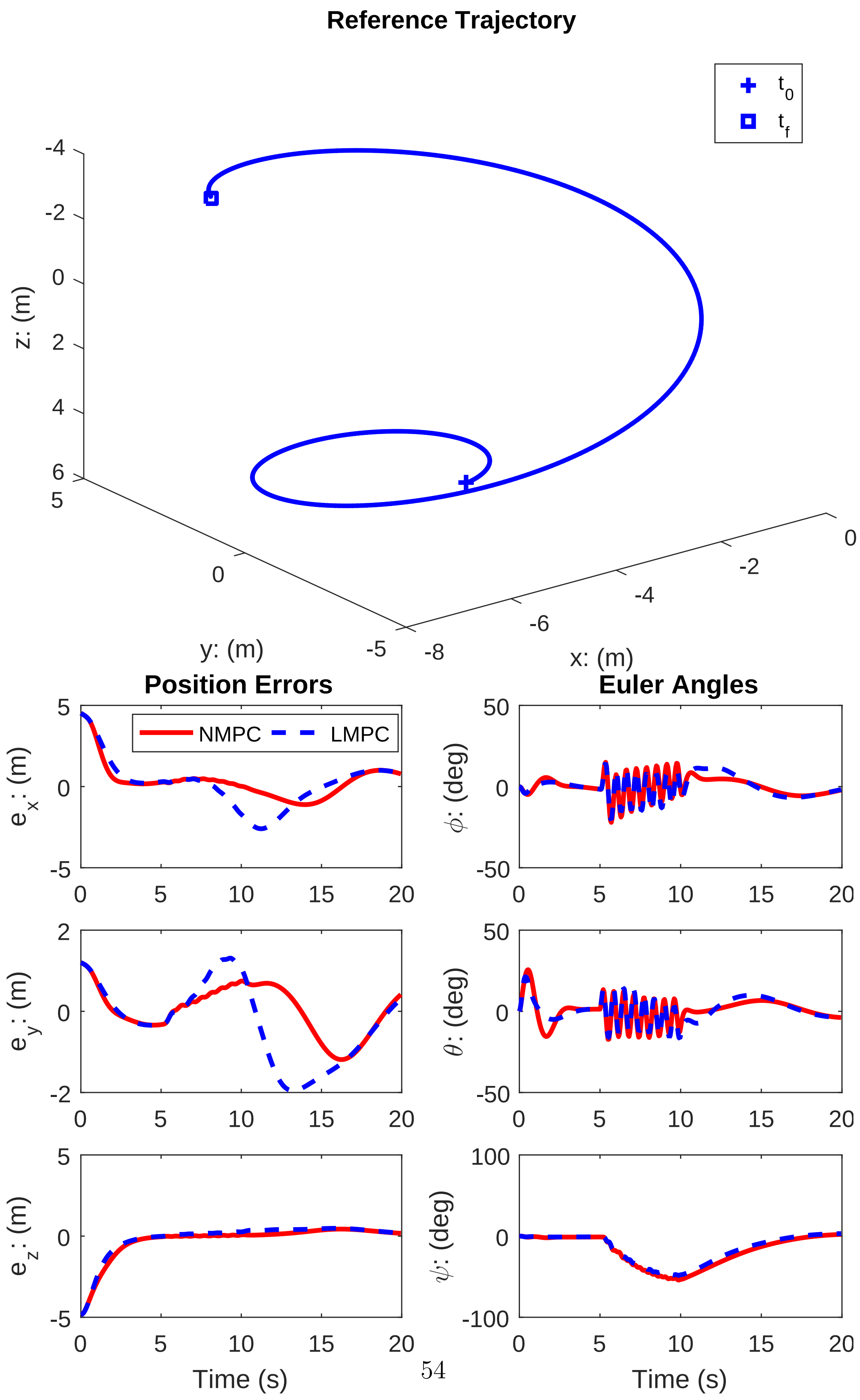


Figure 4.9. Trajectory 2: Spherical Spiral Trajectory (With Disturbance).

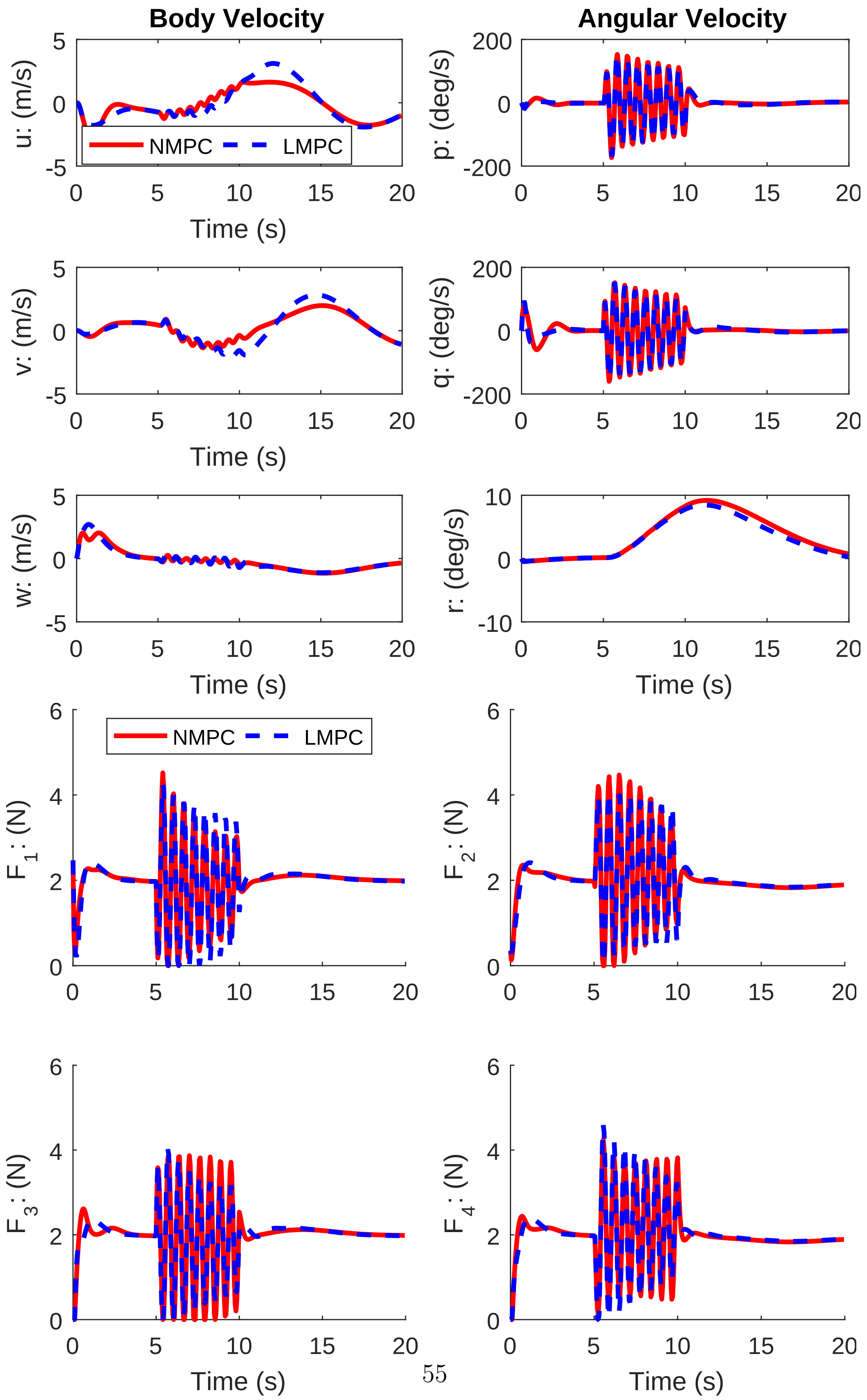


Figure 4.10. Trajectory 2: Spherical Spiral Trajectory (With Disturbance).

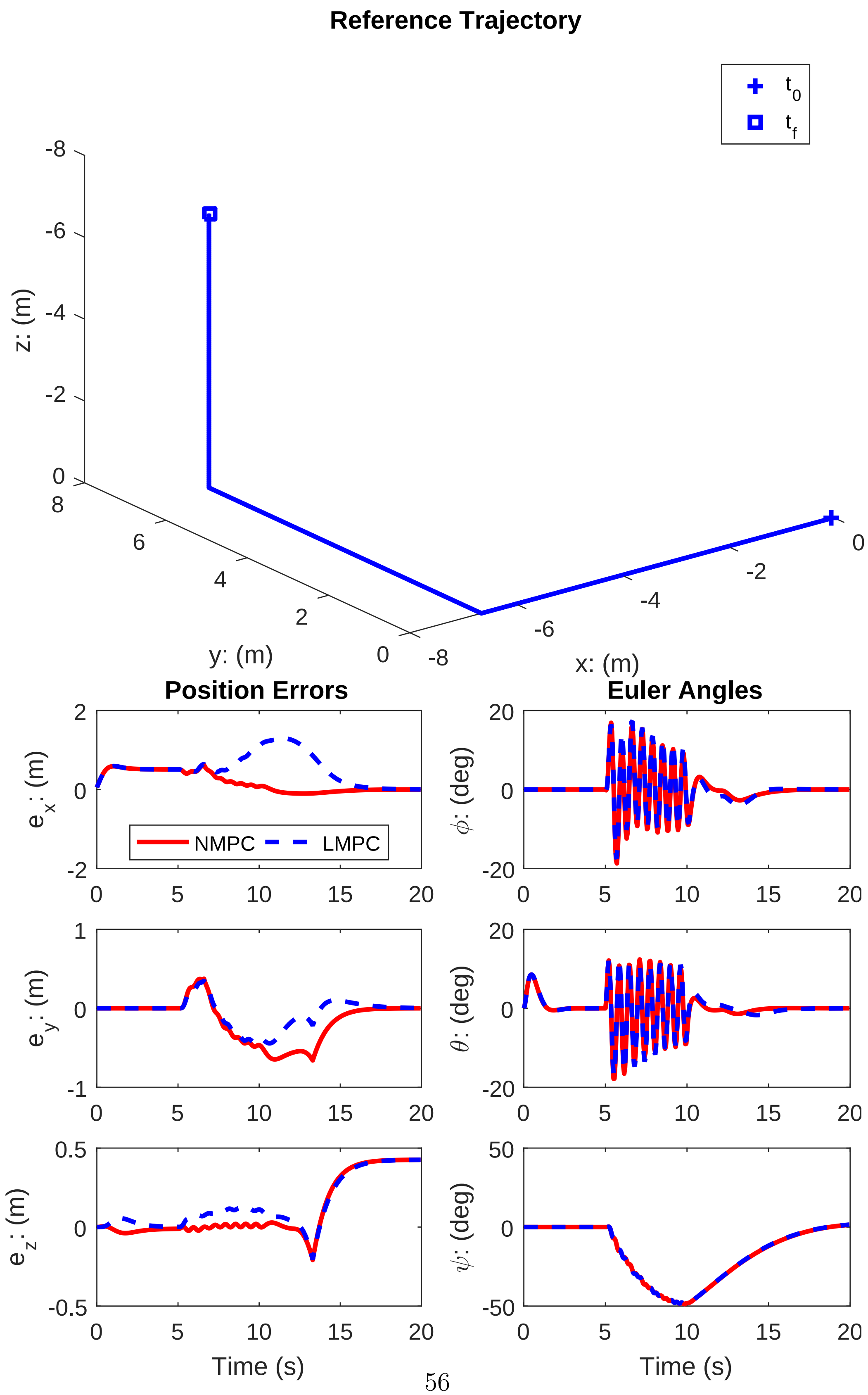


Figure 4.11. Trajectory 3: Straight Line Trajectory (With Disturbance).

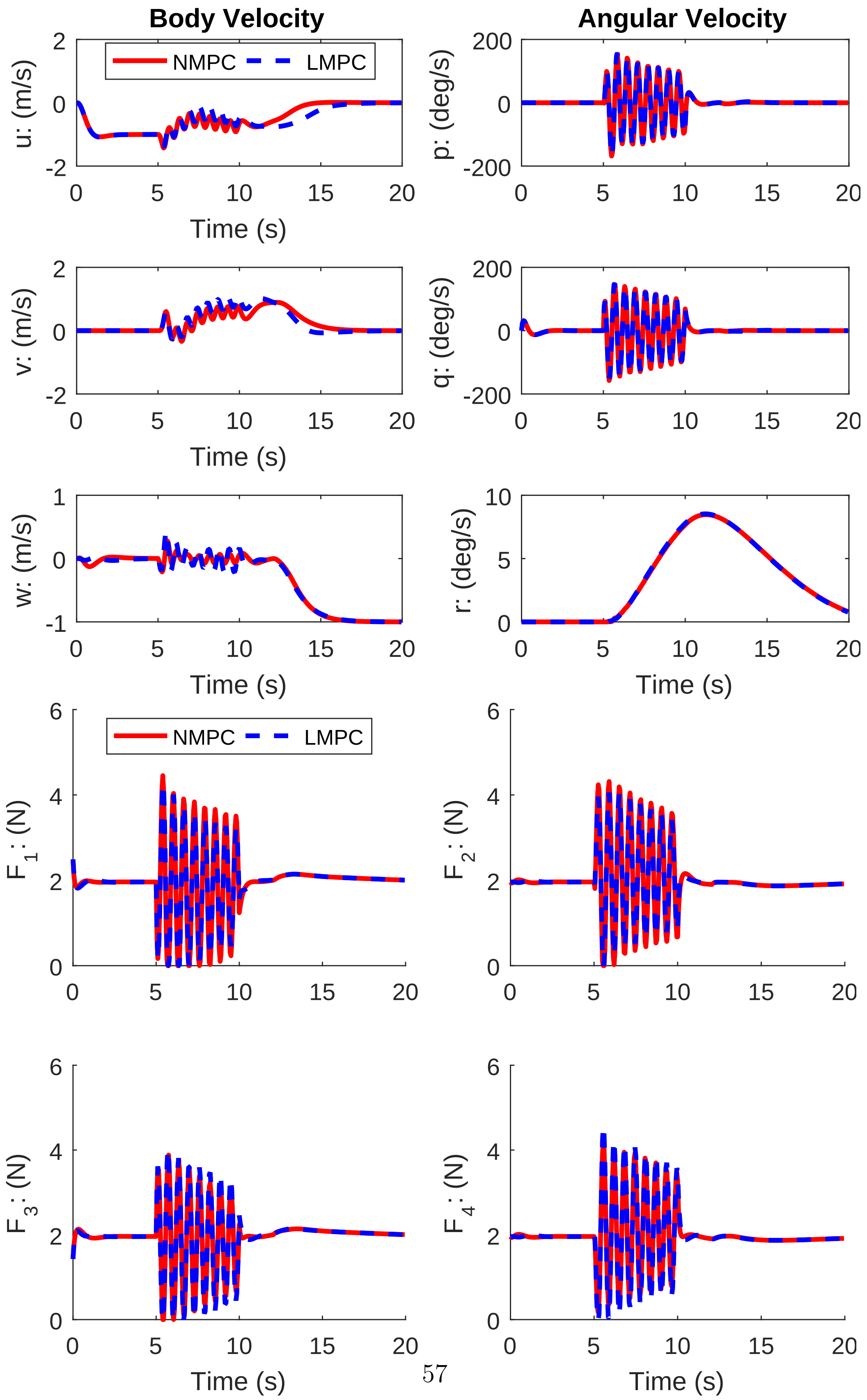


Figure 4.12. Trajectory 3: Straight Line Trajectory (With Disturbance).

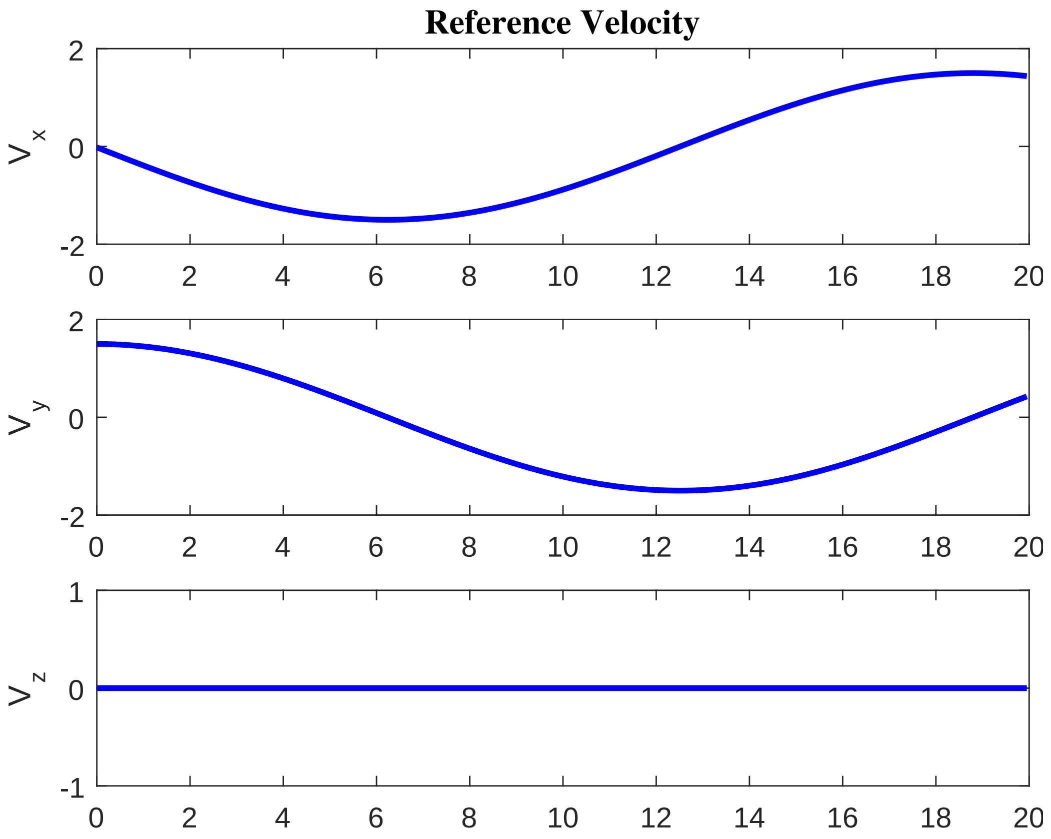


Figure 4.13. Reference Velocity for Helical Trajectory.

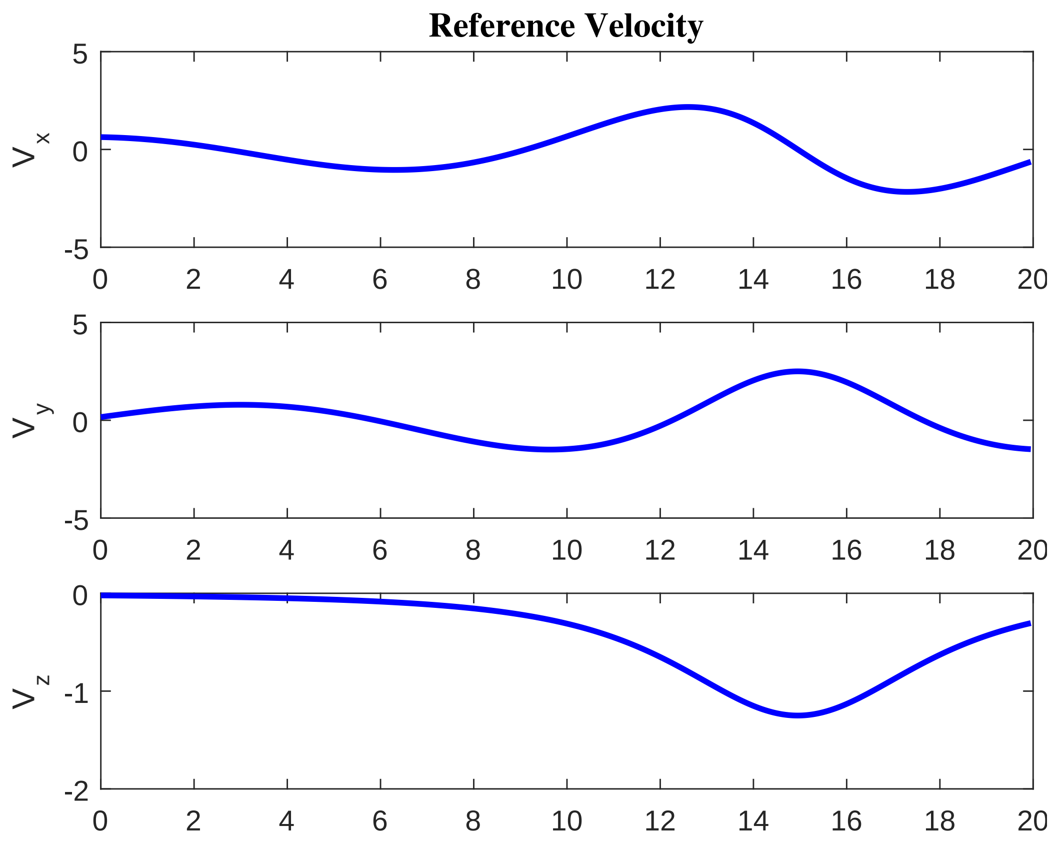


Figure 4.14. Reference Velocity for Spherical Spiral Trajectory.

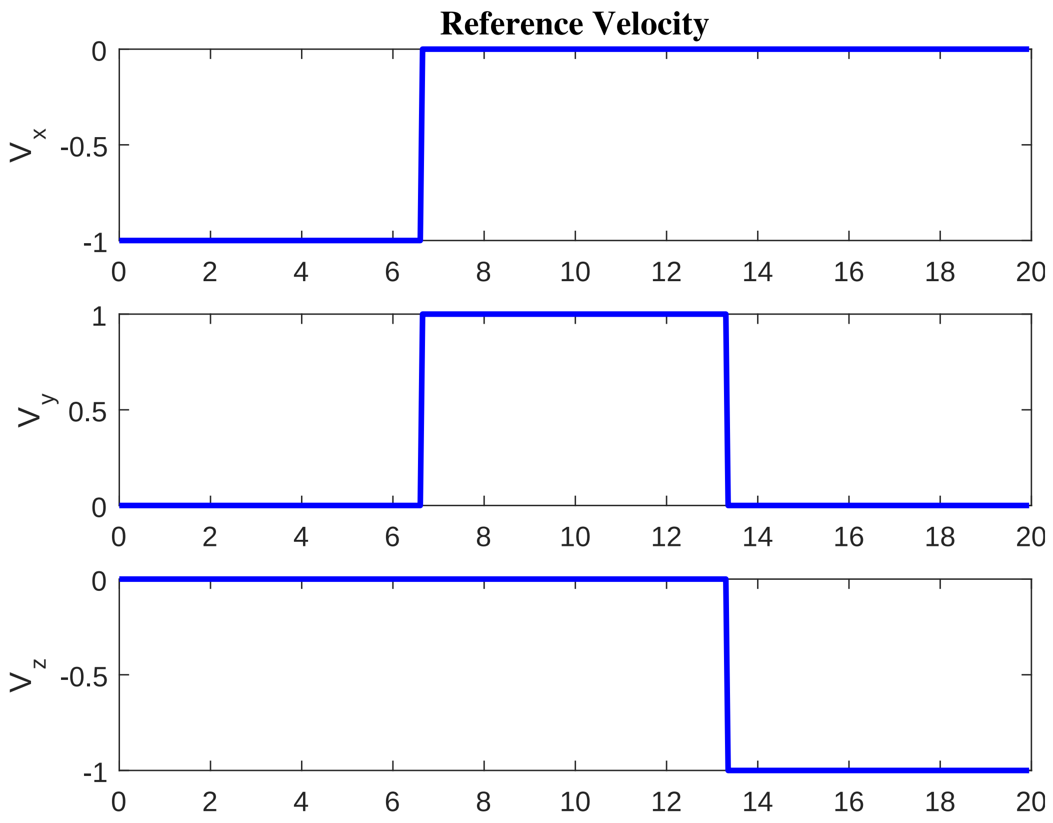


Figure 4.15. Reference Velocity for Straight Line Segments Trajectory.

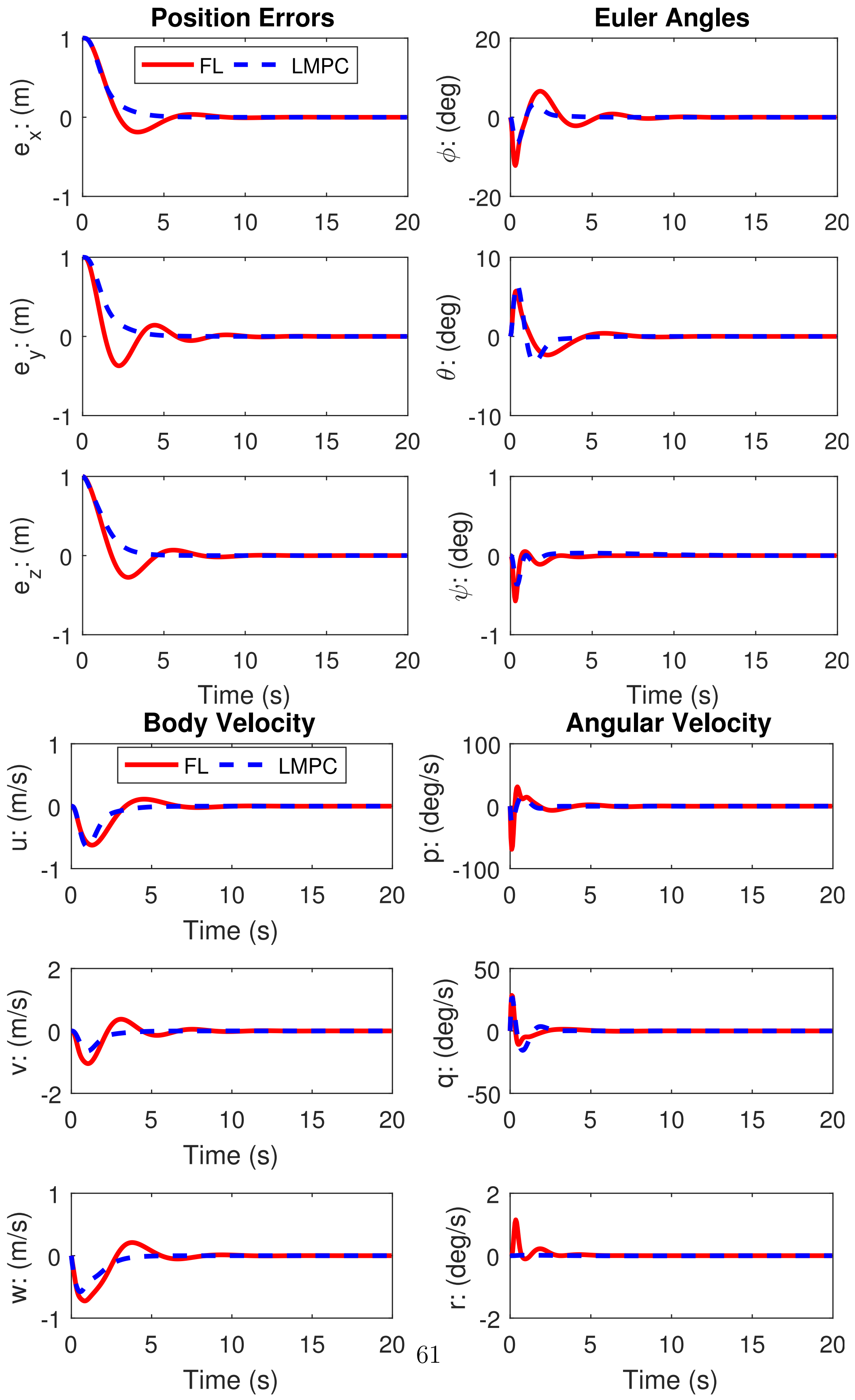


Figure 4.16. Regulator case 1: feedback linearization vs. LMPC.

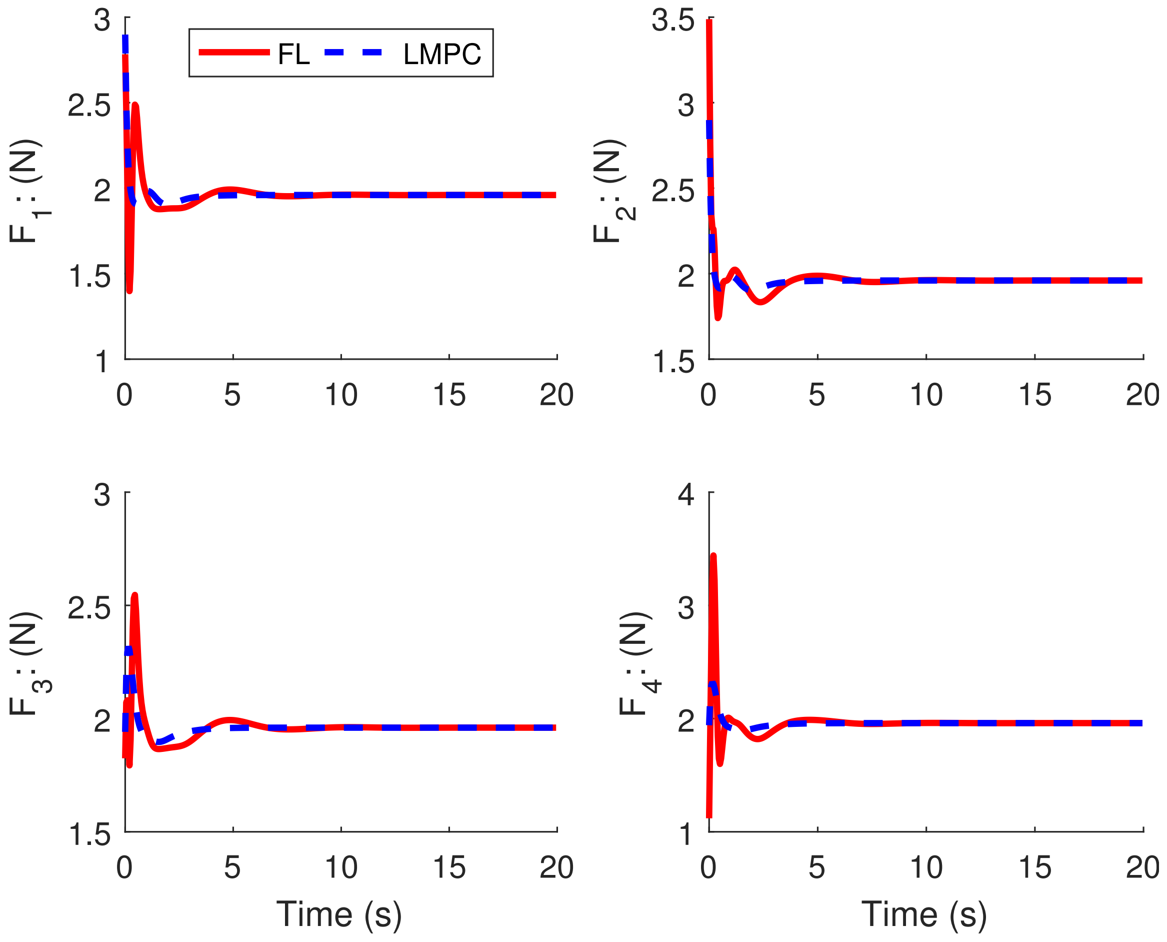


Figure 4.17. Regulator case 1 input: feedback linearization vs. LMPC.

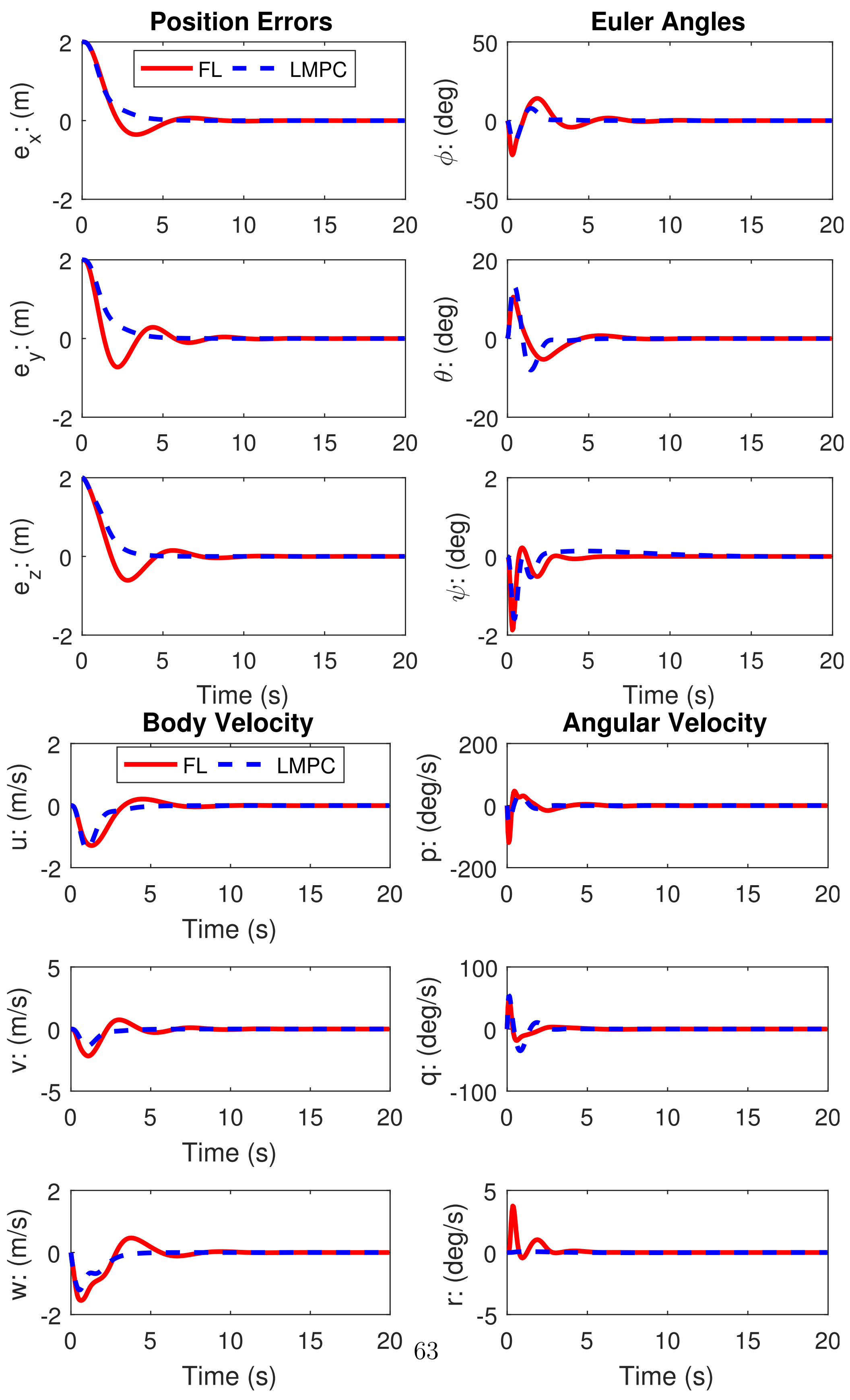


Figure 4.18. Regulator case 2: feedback linearization vs. LMPC.

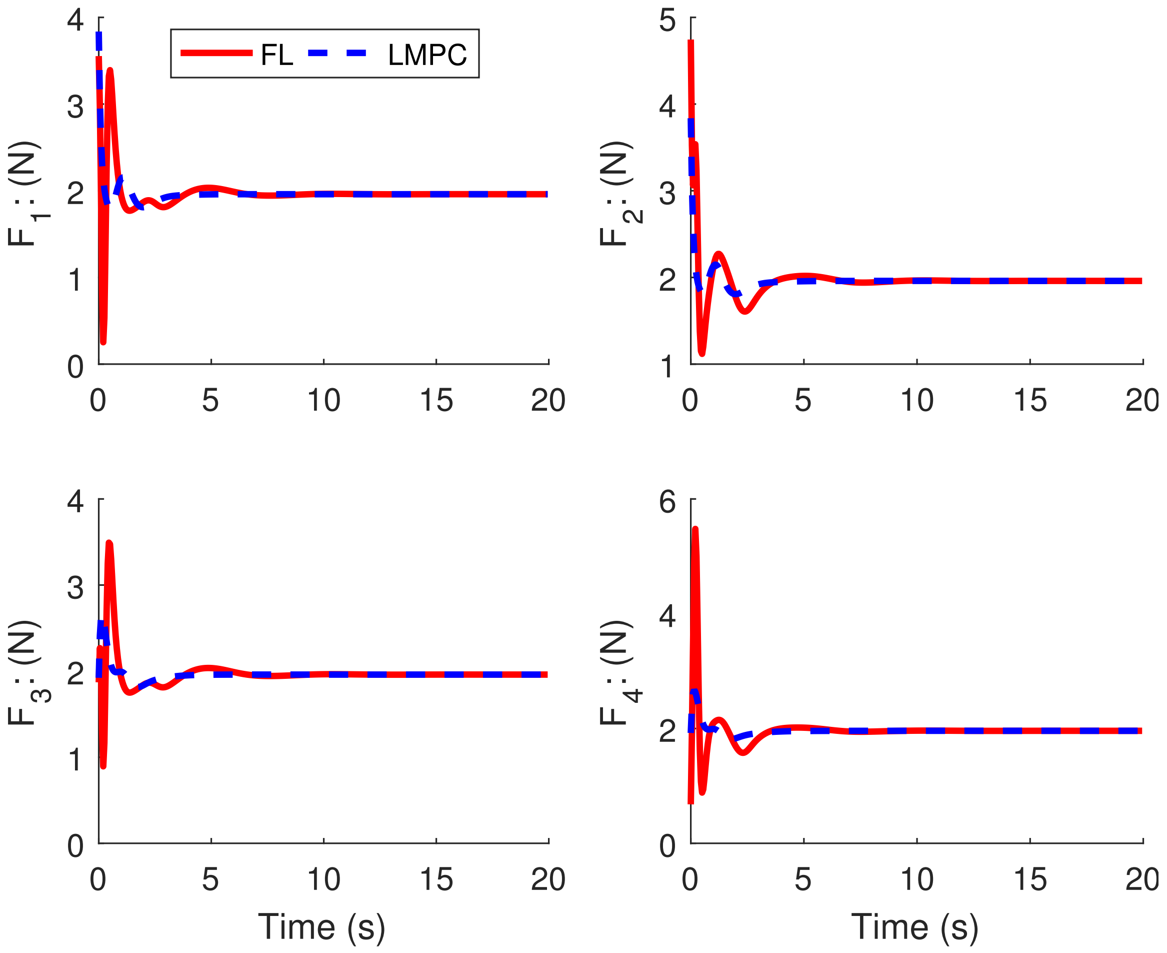


Figure 4.19. Regulator case 2 input: feedback linearization vs. LMPC.

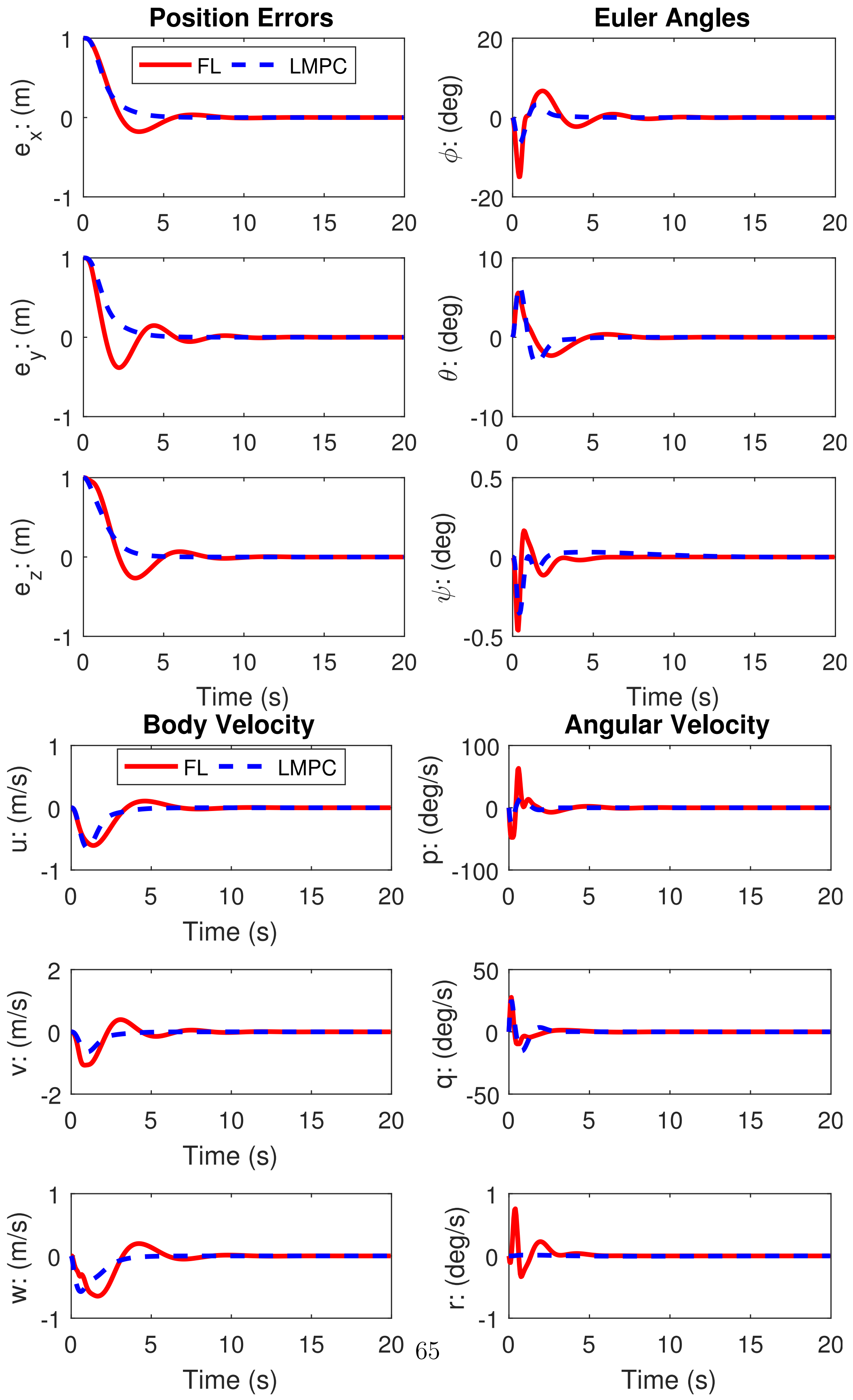


Figure 4.20. Regulator case 1 ($0 \leq F_i \leq 2.45$): feedback linearization vs. LMPC.

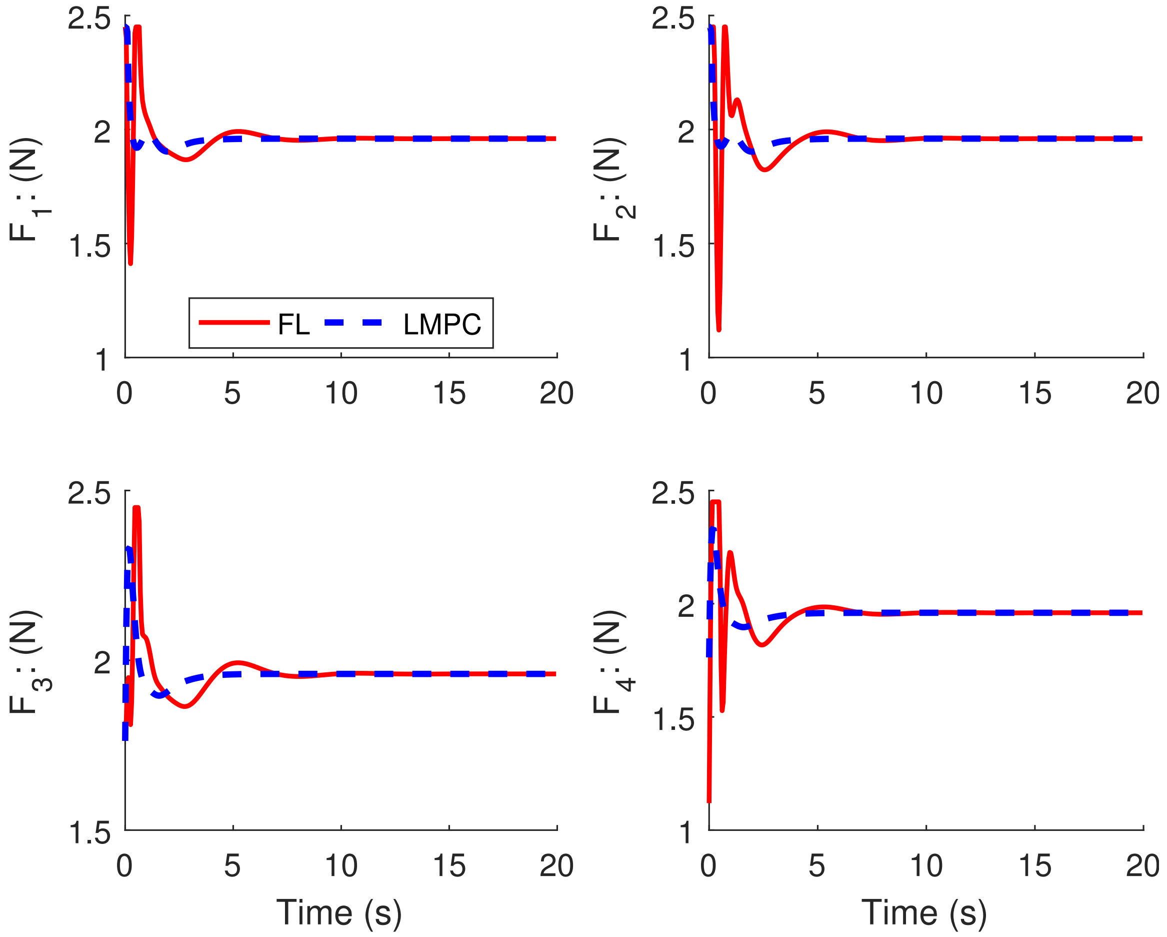


Figure 4.21. Regulator case 1 input ($0 \leq F_i \leq 2.45$): feedback linearization vs. LMPC.

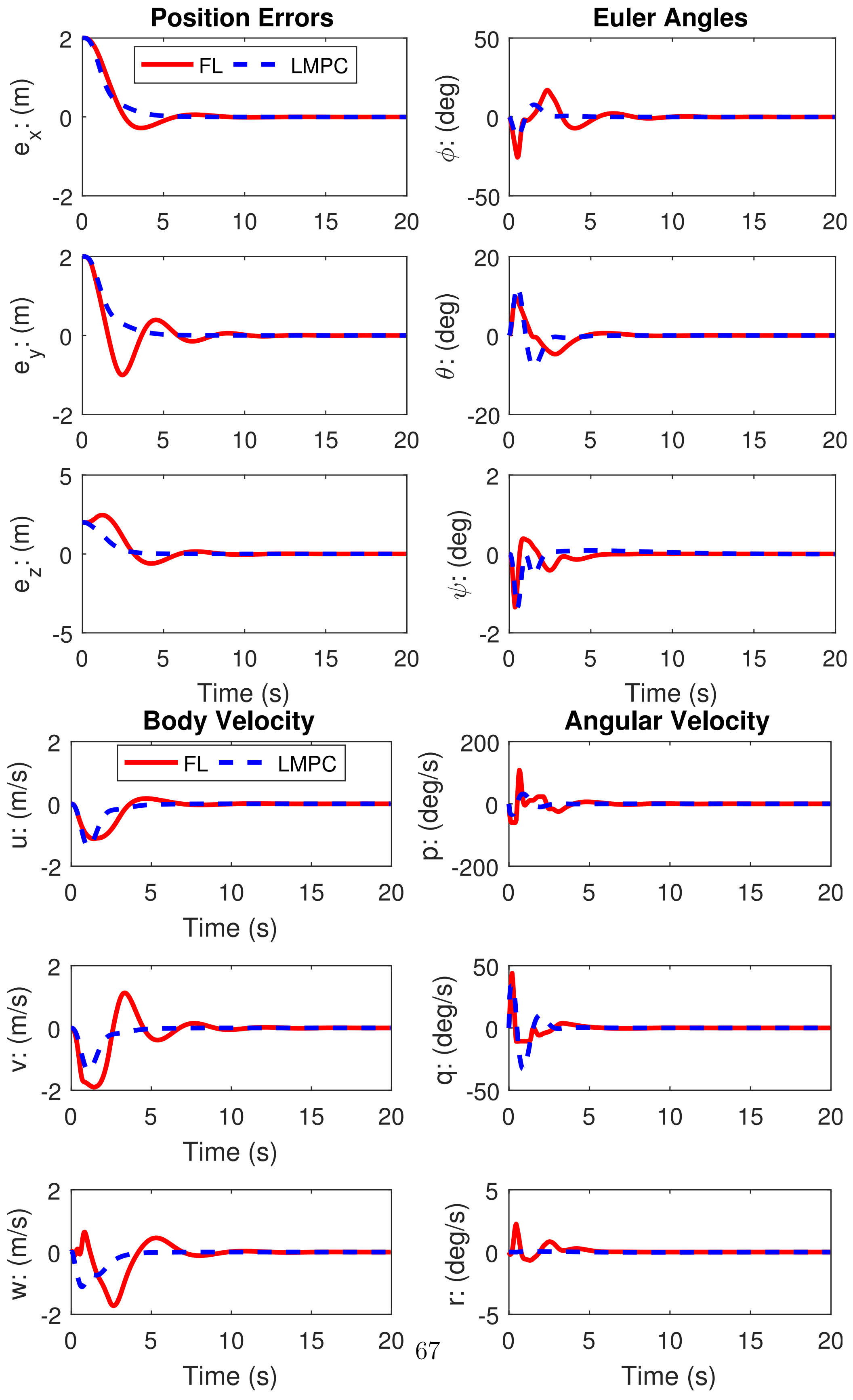


Figure 4.22. Regulator case 2 ($0 \leq F_i \leq 2.45$): feedback linearization vs. LMPC.

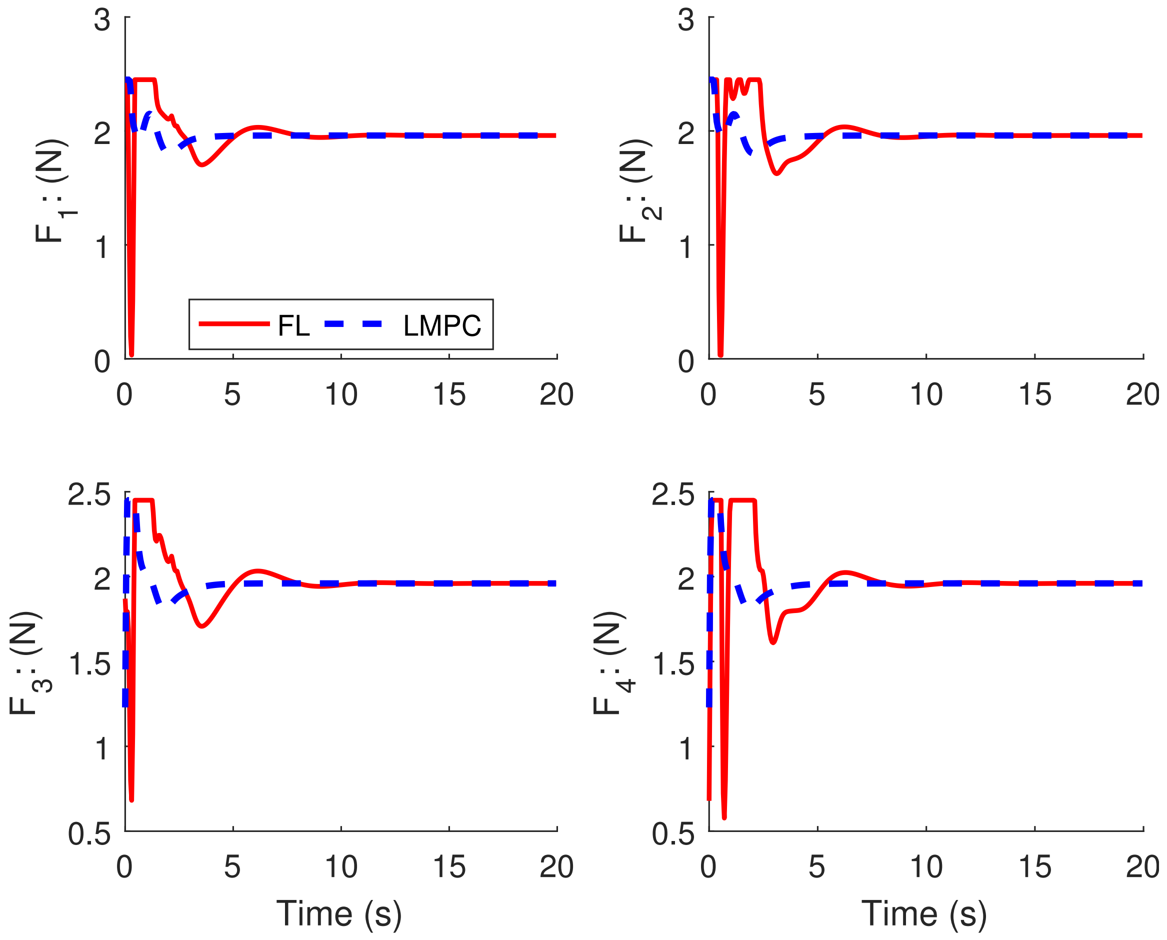


Figure 4.23. Regulator case 2 input ($0 \le F_i \le 2.45$): feedback linearization vs. LMPC.

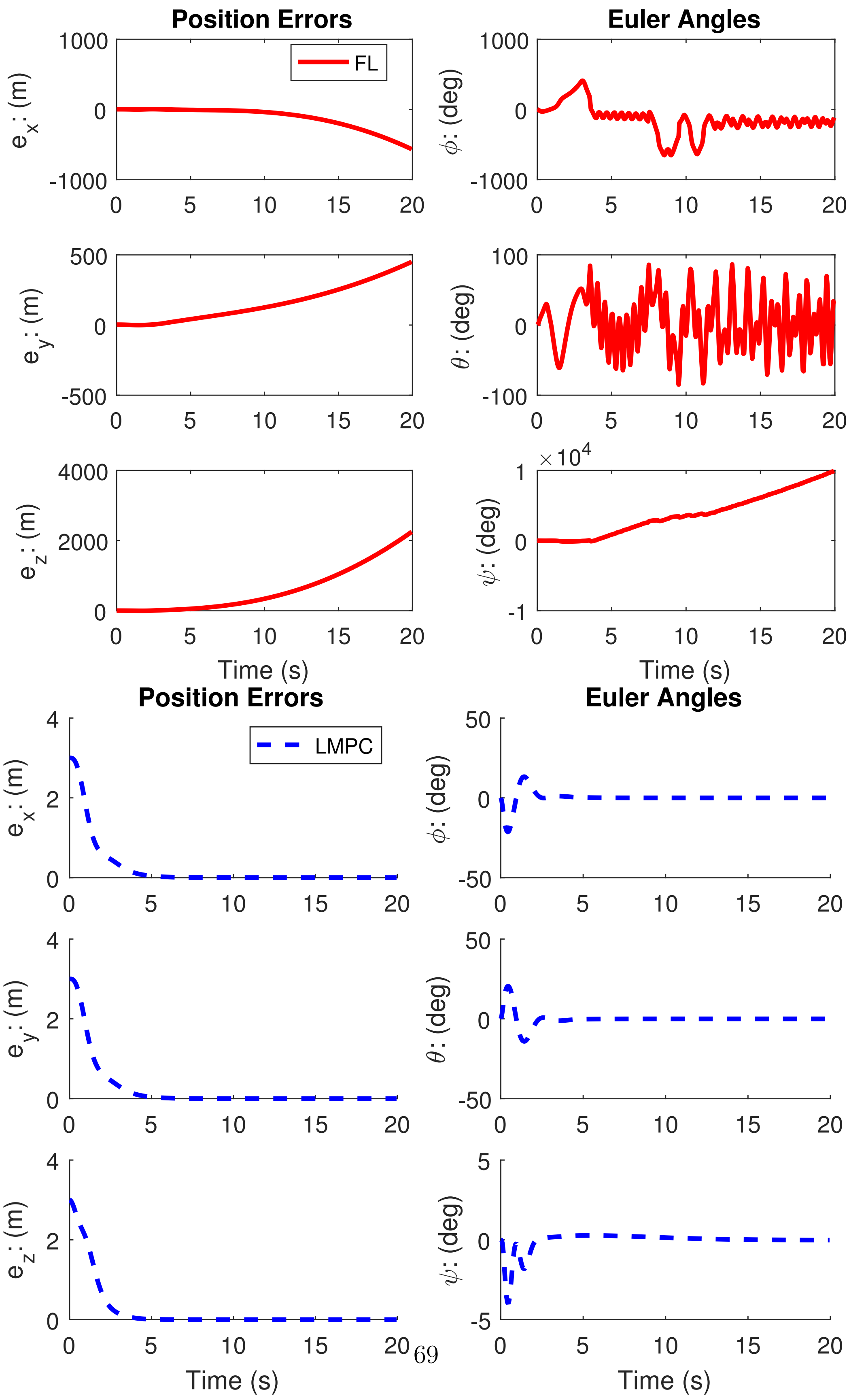


Figure 4.24. Regulator case 3: feedback linearization vs. LMPC.

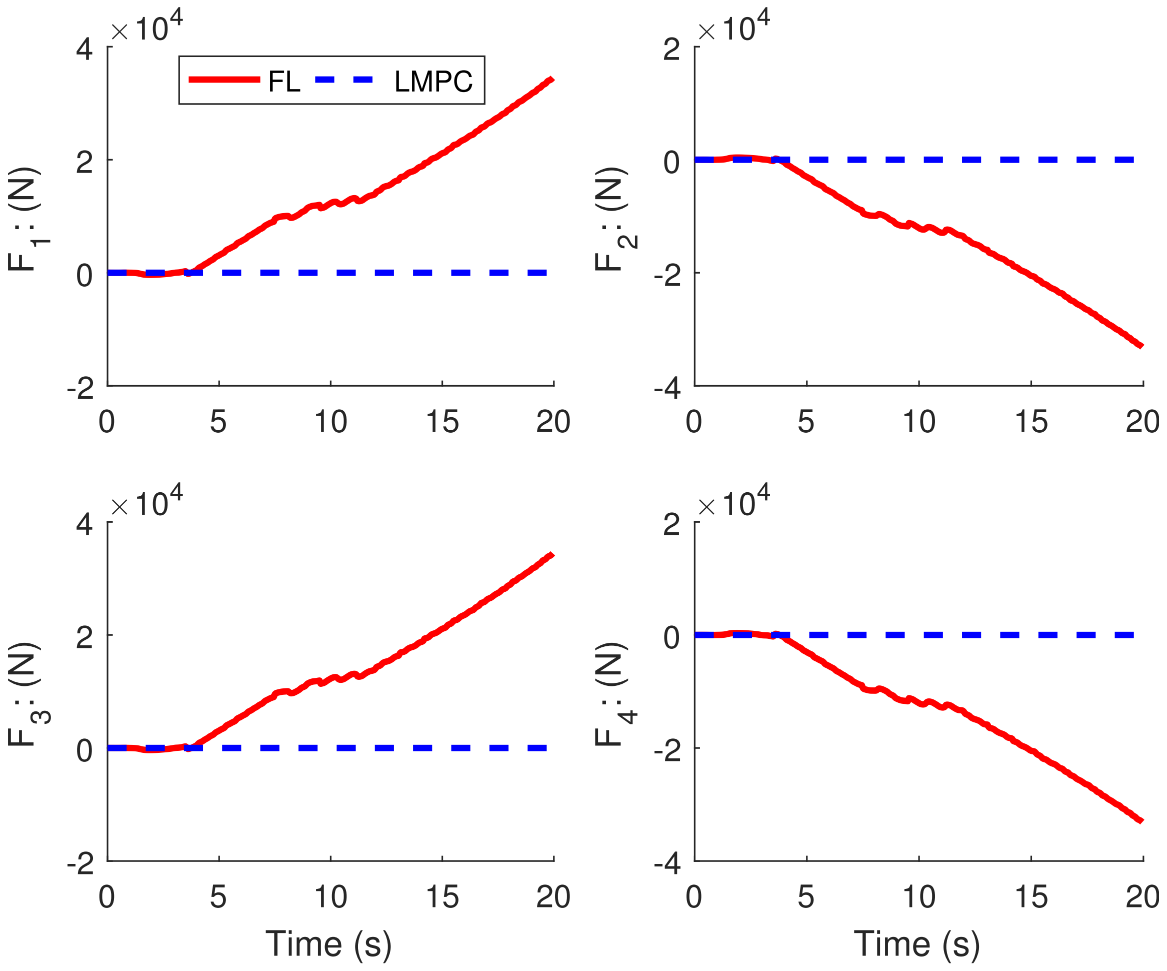


Figure 4.25. Regulator case 3 input: feedback linearization vs. LMPC.

CHAPTER 5

Cooperative Control based on Nonlinear Model Predictive Control

For the sake of simplicity, we will drop the bracketed term $(\mathbf{x_k})$ representing dependency and use the subscript k instead: for example, $\mathbf{A}(\mathbf{x_k})$ would simply be written as $\mathbf{A}_k$ and so on.

The same N step future state:

$$\begin{aligned} \mathbf{X}_k &= \mathbf{F}_k \mathbf{x}_k + \mathbf{H}_k \Delta \mathbf{U}_k \\ \mathbf{x}_{k+N} &= \mathbf{A}_k^N \mathbf{x}_k + \bar{\mathbf{B}}_k \Delta \mathbf{U}_k \end{aligned} \tag{5.1}$$

5.1 Formation Graph and Information Consensus

Consider a set of N_v quadcopters, the i-th vehicle being described by:

$$\mathbf{x}_{k+1}^i = \mathbf{A}_k^i \mathbf{x}_k^i + \mathbf{B} \Delta \mathbf{u}_k^i \tag{5.2}$$

where $\mathbf{x}_k^i, \Delta \mathbf{u}_k^i$ are states and inputs of the i-th vehicle and all of quadcopters follow the same model as described in Eq. (4.8). Similarly, we arrive at the N step prediction equations for the i-th vehicle as Eq. (5.1):

$$\mathbf{X}_k^i = \mathbf{F}_k^i \mathbf{x}_k^i + \mathbf{H}_k^i \Delta \mathbf{U}_k^i \tag{5.3}$$

And the terminal state:

$$\mathbf{x}_{k+N}^i = (\mathbf{A}_k^i)^N \mathbf{x}_k^i + \bar{\mathbf{B}}_k^i \Delta \mathbf{U}_k^i$$

We use a directed graph pair $(\mathcal{N}, \mathcal{E})$ (shown in Fig. (5.1)) to model the information exchange among vehicles, where $\mathcal{N} = \{1, ..., N_v\}$ is the node set representing each vehicle in the formation and $\mathcal{E} \subset \mathcal{N} \times \mathcal{N}$ is the edge set of ordered pairs of nodes, called edges. The edge $(i, j) \in \mathcal{E}$ denotes that vehicle j can access the current state of vehicle i (i.e. in Fig. (5.1), the edge $(3, 5)$ means vehicle 5 can access the information from vehicle 3). Self edges $(i, i) \in \mathcal{E}$ are allowed (i.e. in Fig. (5.1) , vehicle 2 can access its own location). We are using a leader-follower scheme in which only one of the vehicles, the leader, would have access to the external reference trajectory and the rest act as followers and maintain a formation relative to the leader (shown in Fig. (5.1), vehicle 1 is the leader and vehicles 2 - 6 are followers). Using $\mathbf{x}_{k+n}^{i,r}$ $(n = 1, \ldots, N)$ to denote reference trajectory for i-th vehicle for the horizon $[k+1, k+N]$, the reference trajectories for each vehicle are determined as follows. For the leader, $\mathbf{x}_{k+n}^{i,r}$ is given by the external reference trajectory. For the followers, the following proven stable consensus scheme from Ref. [19] is used here for generating reference trajectories.

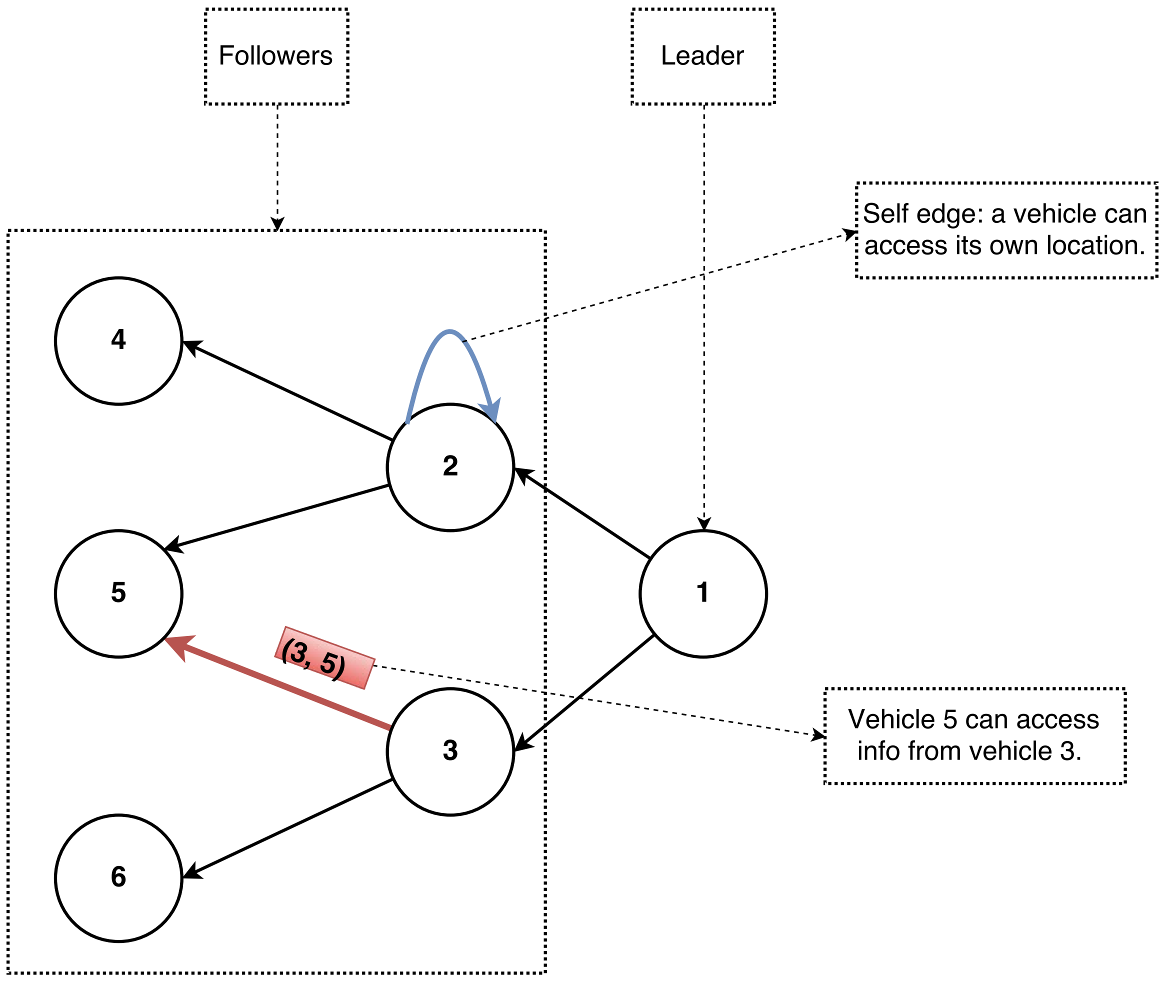

Followers
Leader
Self edge: a vehicle can access its own location.
4
2
5
1
(3, 5)
3
Vehicle 5 can access info from vehicle 3.
6

Figure 5.1. A directed graph example.

$$\mathbf{x}_{k+n}^{i,r} = \frac{1}{\sum_{j=1}^{N_v} \mathbf{G}_{ij}} \left(\mathbf{G}_{ii}\mathbf{x}_{k+n-1}^{i,r} + \sum_{\substack{(i,j)\in\mathcal{E} \\ j\neq i}} \mathbf{G}_{ij}(\mathbf{x}_k^j + \mathbf{d}_{ij}) \right) \quad n = 1,\ldots,N. \tag{5.4}$$

where $\mathbf{d}_{ij}$ denotes the relative position between vehicle i and j desired in the formation. And $\mathbf{G}$ is a matrix associated with the graph defined as: $\mathbf{G}_{ii} = 1,\ \forall i \in \mathcal{N}$ and $\mathbf{G}_{ij} = 1,\ \forall (j,i) \in \mathcal{E}$.

5.2 Controller Design

With reference trajectories determined, consider the following objective function for vehicle i:

$$
\begin{aligned}
J(\mathbf{x}_k^i) = {} & \left[\mathbf{X}_k^i - \mathbf{X}_k^{i,r}\right]^T \bar{\mathbf{Q}} \left[\mathbf{X}_k^i - \mathbf{X}_k^{i,r}\right] + (\Delta\mathbf{U}_k^i)^T \bar{\mathbf{R}} \Delta\mathbf{U}_k^i \\
& + \left[\mathbf{x}_{k+N}^i - \mathbf{x}_{k+N}^{i,r}\right]^T \mathbf{Q}_f \left[\mathbf{x}_{k+N}^i - \mathbf{x}_{k+N}^{i,r}\right] \qquad (5.5)
\end{aligned}
$$

where

$$
\bar{\mathbf{Q}} = \begin{bmatrix} \mathbf{Q} & & & \\ & \mathbf{Q} & & \\ & & \ddots & \\ & & & \mathbf{Q} \end{bmatrix}, \quad \bar{\mathbf{R}} = \begin{bmatrix} \mathbf{R} & & & \\ & \mathbf{R} & & \\ & & \ddots & \\ & & & \mathbf{R} \end{bmatrix}, \quad \mathbf{X}_k^{i,r} = \begin{bmatrix} \mathbf{x}_k^{i,r} \\ \mathbf{x}_{k+1}^{i,r} \\ \mathbf{x}_{k+2}^{i,r} \\ \vdots \\ \mathbf{x}_{k+N-1}^{i,r} \end{bmatrix}
$$

and $\mathbf{Q}, \mathbf{Q}_f \in \Re^{n\times n}$ and $\mathbf{R} \in \Re^{m\times m}$ and they satisfy $\mathbf{Q} \geq 0$, $\mathbf{Q}_f \geq 0$ and $\mathbf{R} > 0$.

5.3 Constraints

5.3.1 Combined Input and State Constraints

The same combined constraints as Eq. (4.11) in previous chapter are summarized here as:

$$
\mathbf{\Gamma}(\mathbf{x_k})\Delta\mathbf{U}_k \quad \leq \quad \mathbf{\Upsilon}(\mathbf{x_k}) \qquad (5.6)
$$

5.3.2 State Constraints - Collision Avoidance

The second part of state constraints stems from collision avoidance. In order to ensure the vehicles do not collide with each other, a predetermined safe distance d_s is to be enforced between them. The distance between any two vehicles $\mathbf{x}_k^i$ and $\mathbf{x}_k^j$ should always be greater than d_s.

$$
\forall (i,j) \in \mathcal{E}, \quad \|\mathbf{C_p}(\mathbf{x}_{k+n}^i - \mathbf{x}_k^j)\| \leq d_s, \quad n = 1, \ldots, N. \qquad (5.7)
$$

Where $\mathbf{C_p}$ is the output matrix corresponding to position $\mathbf{p}$ and $\mathbf{C_p} = [\mathbf{I}_{3\times3}\,\mathbf{0}_{3\times3}\,\mathbf{0}_{3\times3}\,\mathbf{0}_{3\times3}]$. For i-th vehicle, the constraints on its states can be expressed as:

$$\begin{aligned} \mathcal{C}_{\mathbf{z}}\left(\mathbf{F}\mathbf{x}_k^i + \mathbf{H}\Delta\mathbf{U}_k^i\right) &\leq \Delta\mathbf{Z}_b^i \\ \forall(i,j)\in\mathcal{E}, \quad \|\mathbf{C_p}(\mathbf{x}_{k+n}^i - \mathbf{x}_k^j)\| &\leq d_s, \quad n = 1,\ldots,N. \end{aligned} \tag{5.8}$$

5.4 Stability

It is proved in Ref. [19] that if all the associated interaction topologies contain a spanning tree, the discrete update scheme described in Eq. (5.4) achieves consensus asymptotically. As shown in Fig. (5.2), (5.3), (5.4), and (5.5), in all scenarios, vehicle 1 is the leader, which is also the root node of all spanning trees. Let $\mathcal{E}_k$ represent the communication topology during the time interval (t_k, t_{k+1}), if each $\mathcal{E}_k$ $(k \in \mathbb{N})$ contains a spanning tree, the reference trajectory of each vehicle given by Eq. (5.4) would eventually converge to its designated location relative to the leader as shown in Eq. (5.9).

$$\|\mathbf{x}_{k+n}^{i,r} - (\mathbf{x}_k^1 + \mathbf{d}_{i1})\| \to 0, \quad i = 2, 3, \cdots, N_v. \tag{5.9}$$

as $n \to \infty$. All reference trajectories converge asymptotically. Each agent employs an MPC controller to track its respective reference trajectory by minimizing its cost function described by Eq. (5.5), which has been proven stable in Ref. [39]. By combining the consensus strategy and the MPC tracking controller, the entire system achieves consensus asymptotically as $t \to \infty$.

The communication topology spanning tree constraint can be further relaxed to repeating occurrences of a set of topologies by which only their union needs to contain a spanning tree for consensus to be achieved. If there exists an infinite sequence of uniformly bounded, nonoverlapping time intervals $[k_jT_s, (k_j + l_j)T_s)$, $j = 1, 2, \ldots$, starting at $k_1 = 0$, with the property that each interval $[(k_j + l_j)T_s, k_{j+1}T_s)$ is uni-

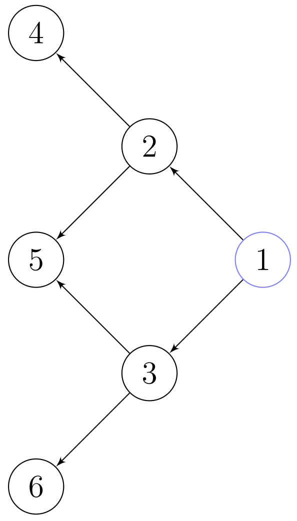


Figure 5.2. 1. Triangular formation: weakly connected overall.

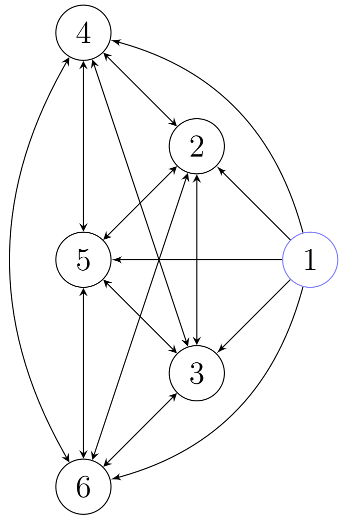


Figure 5.3. 2. Triangular formation: weakly connected overall, followers strongly connected.

formly bounded and the union of the graphs across each interval $[(k_j + l_j)T_s, k_{j+1}T_s)$ has a spanning tree, the discrete update scheme described in Eq. (5.4) achieves consensus asymptotically (See Ref. [19] Theorem 3.10). Combining this requirement with the MPC controller for tracking would also guarantee the stability of the formation controller.

5.5 Simulation Results

In this section simulation results are presented for formation tracking for a group of vehicles using the method detailed in the preceding sections. Two formations and two connectivity conditions are considered here as shown in Fig. (5.2), (5.3), (5.4), and (5.5). Safe distance to avoid collision is set as $d_s = 0.4(m)$. In scenarios 1 and 3, all the vehicles are weakly connected with vehicle 1 being the root node. Every follower vehicle is either directly or indirectly connected to the root node. While in scenarios 2 and 4, besides the overall weakly connected condition with vehicle 1

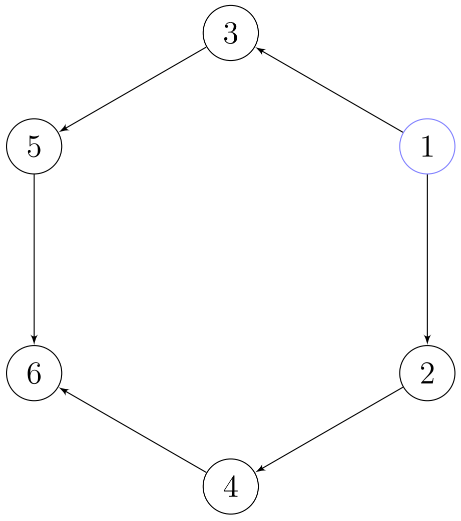


Figure 5.4. 3. Hexagonal formation: weakly connected overall.

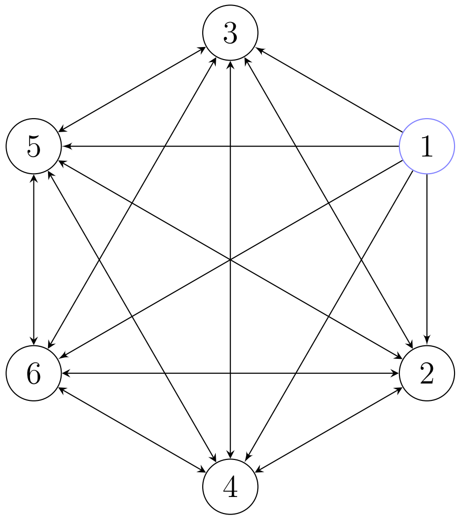


Figure 5.5. 4. Hexagonal formation: weakly connected overall, followers strongly connected.

being the root node, the follower nodes are strongly connected. Each follower vehicle is able to access information from any other follower vehicles. In all scenarios, root node vehicle 1 is the leader with access to the external reference trajectory.

Fig. (5.6) and (5.7) showed the formation history of 6 vehicles with weakly connected graph tracking a triangular shape formation starting from a straight line formation. Fig. (5.6) showed the formation history when looked in the X-Y plane and X-Z plane. The first plot in Fig. (5.7) showed its trajectory in 3D space and the second one showed its reference at different stages of its trajectory.

Fig. (5.8) and (5.9) showed the the same desired formation with its followers strongly connected. In comparison, it can be seen that strongly connected followers graph significantly reduces z position divergence of the vehicles without loss of performance in x and y directions.

The same phenomenon can be observed from Fig. (5.10), (5.11), (5.12), and (5.13), which showed the formation history of 6 vehicles tracking a hexagon formation with connectivity conditions shown in Fig. (5.4) and (5.5).

5.6 Concluding Remarks

A new strategy for formation stabilization is proposed by combining an existing consensus strategy and a nonlinear model predictive control law derived from a state dependent coefficient formulation that employs a novel parameterization to avoid singularities. The consensus strategy is used here to generate reference trajectories for the individual NMPC controller to track. By doing this, consensus is achieved while input, input rate, state, and output constraints are explicitly incorporated at the same time. Simulations are performed to show the effectiveness of the control strategy and it shows the beneficial effects of having followers strongly connected. The control law ensures stable formation convergence as long as certain interaction topology conditions are met. Those conditions were discussed as well.

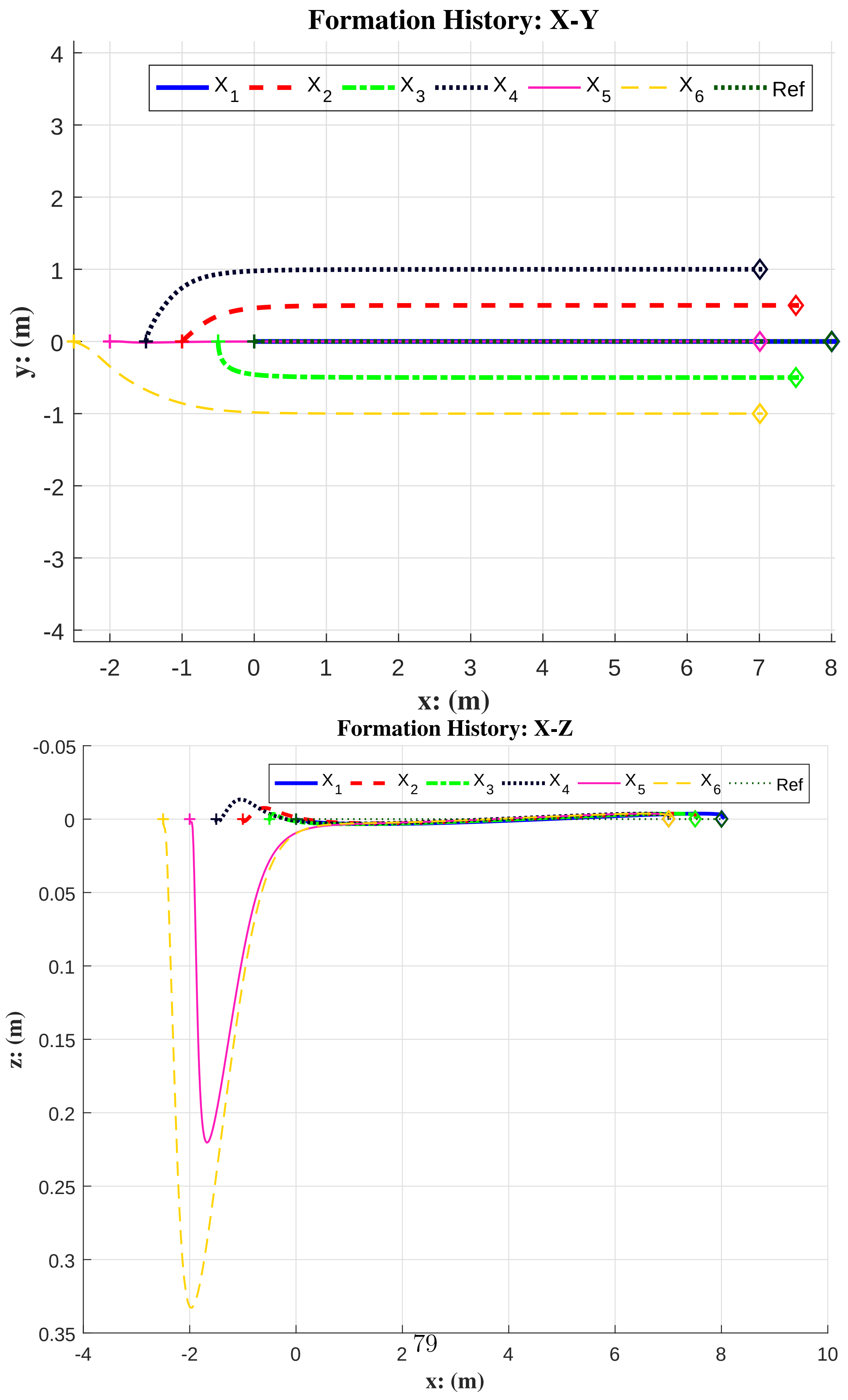
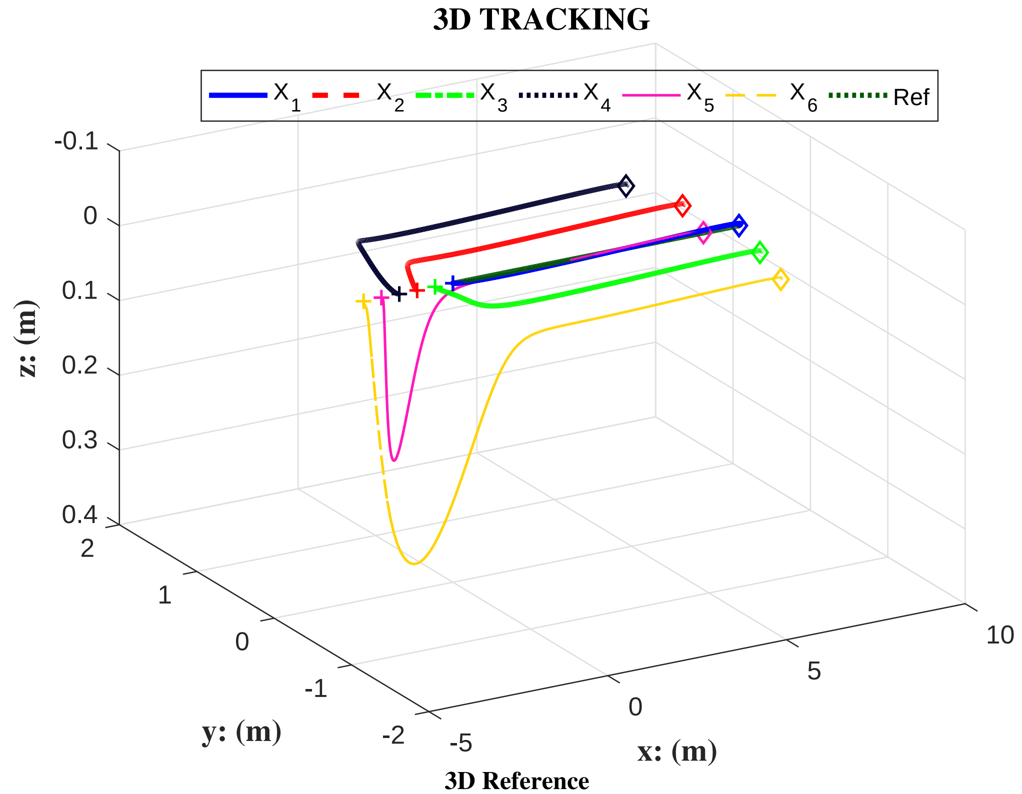


Figure 5.6. 1. Triangular 6-vehicle formation, weakly connected overall.



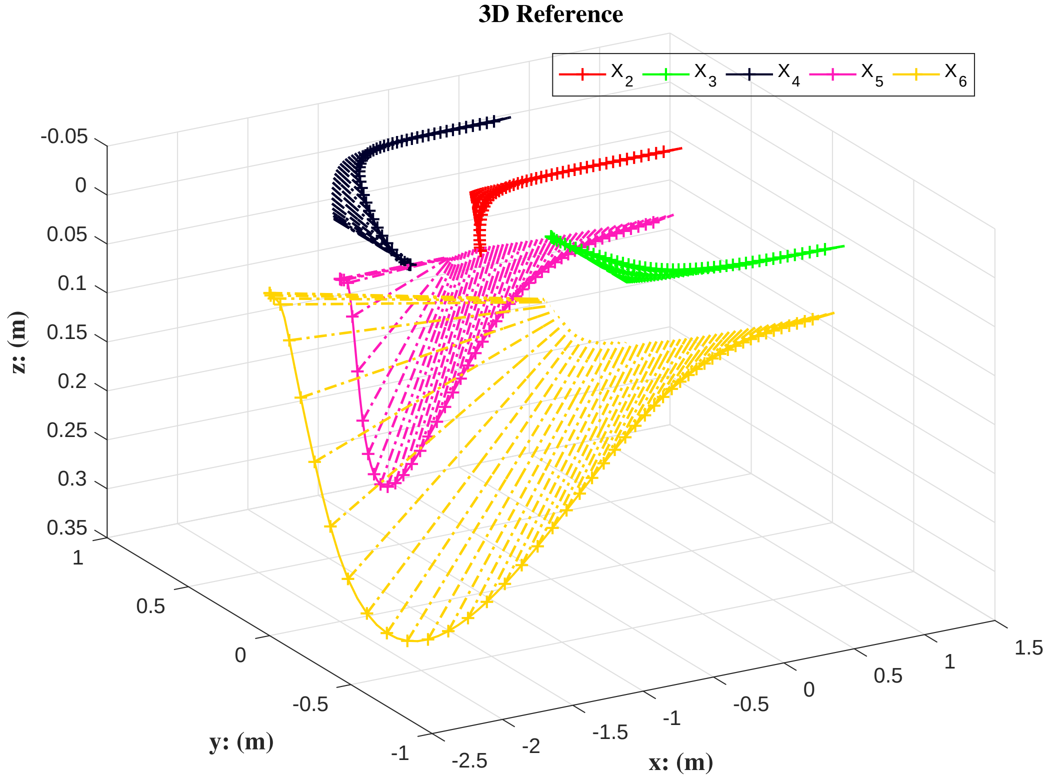


Figure 5.7. 1. Triangular 6-vehicle formation, weakly connected overall.

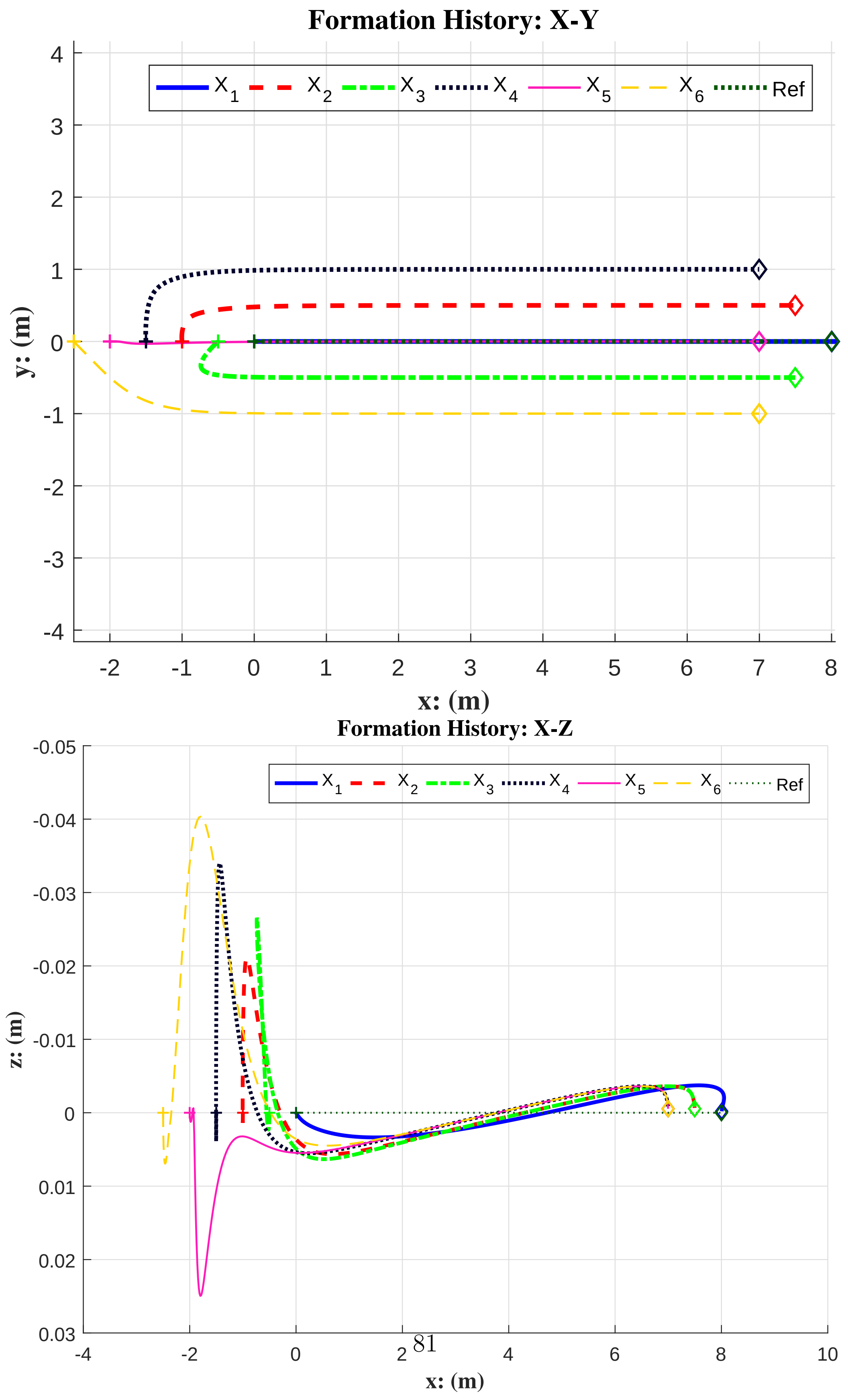
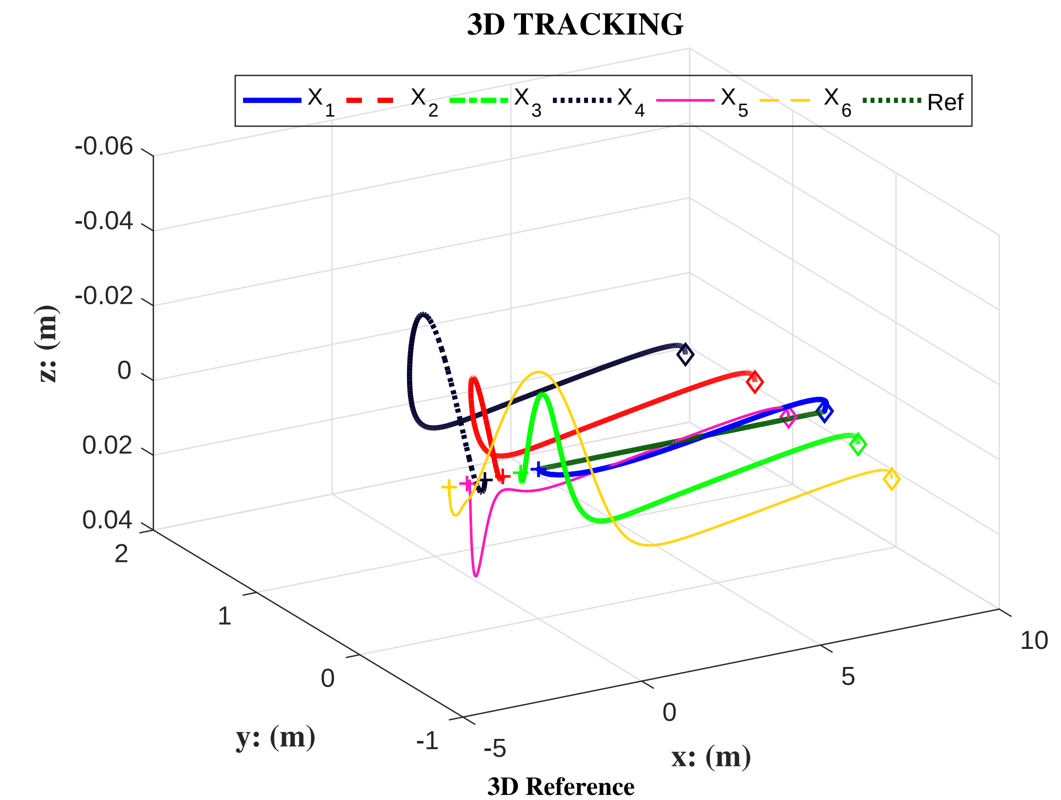


Figure 5.8. 2. Triangular 6-vehicle formation, weakly connected overall, followers strongly connected.



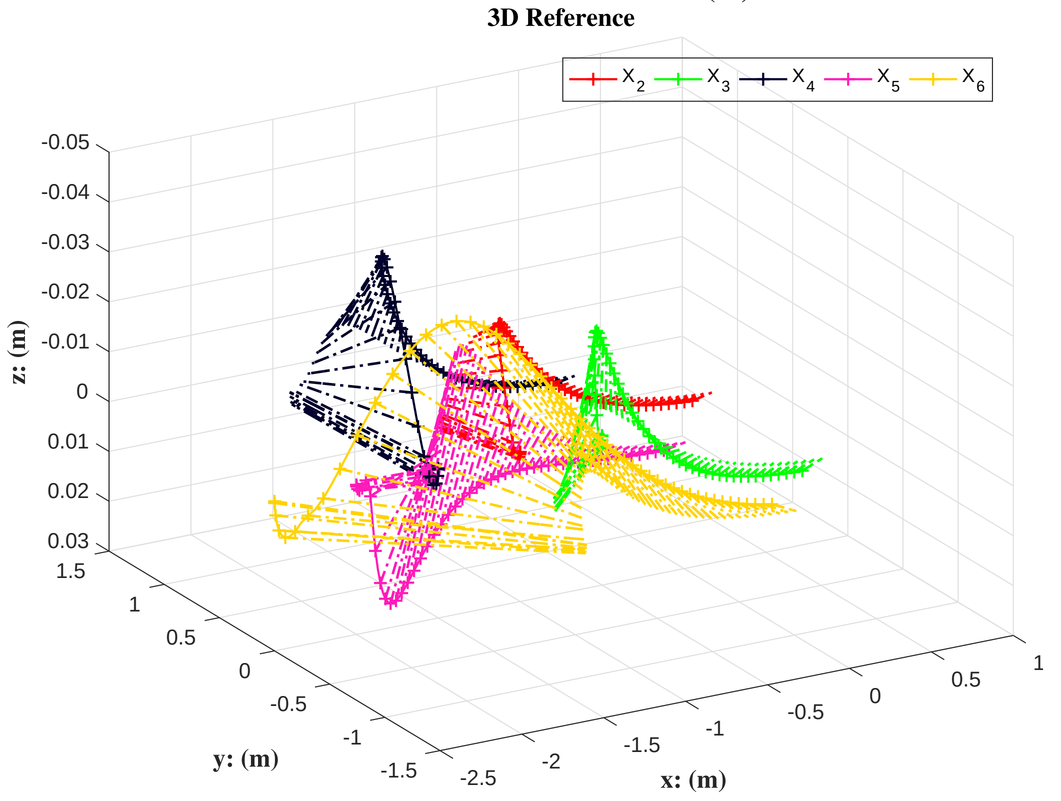


Figure 5.9. 2. Triangular 6-vehicle formation, weakly connected overall, followers strongly connected.

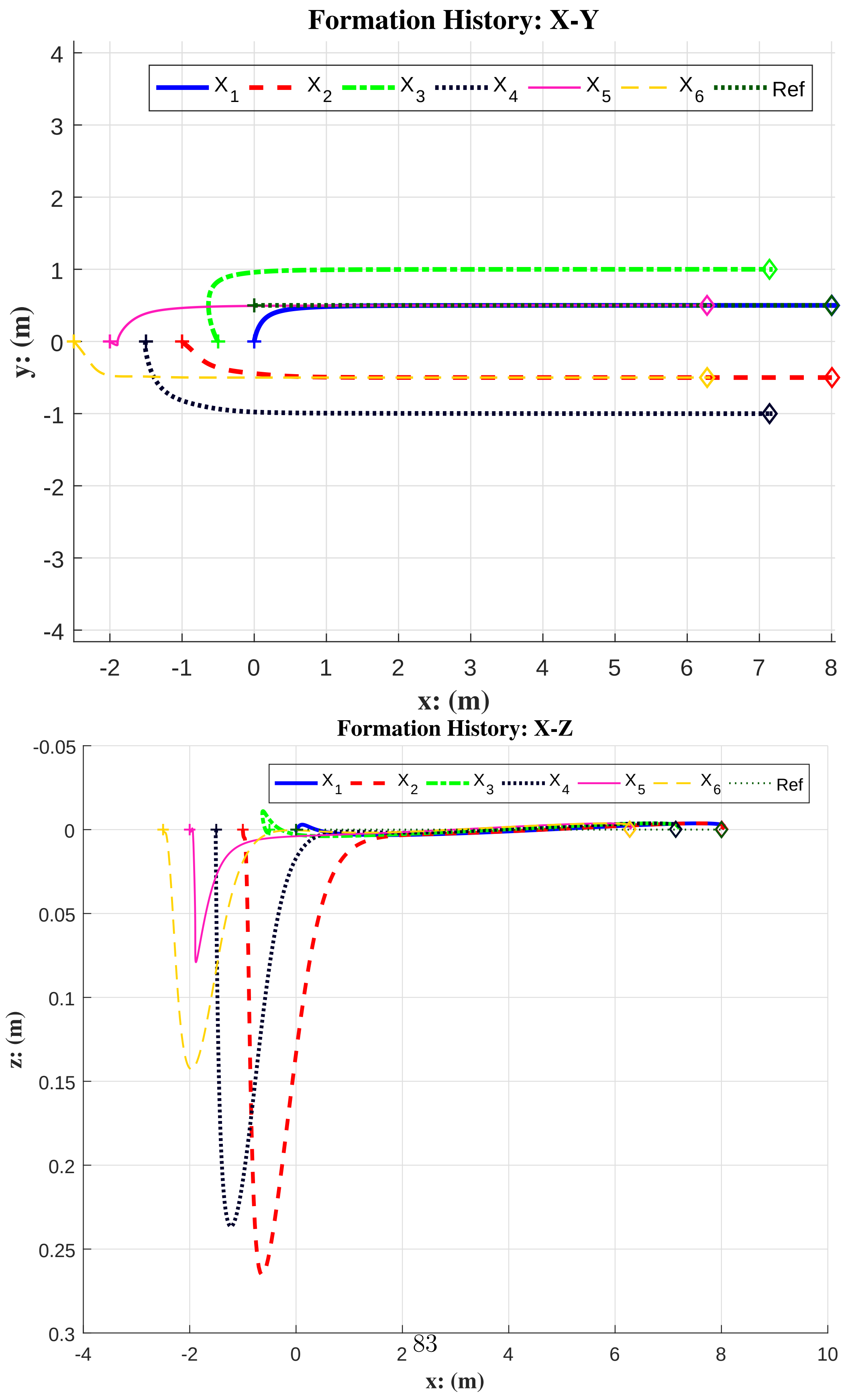
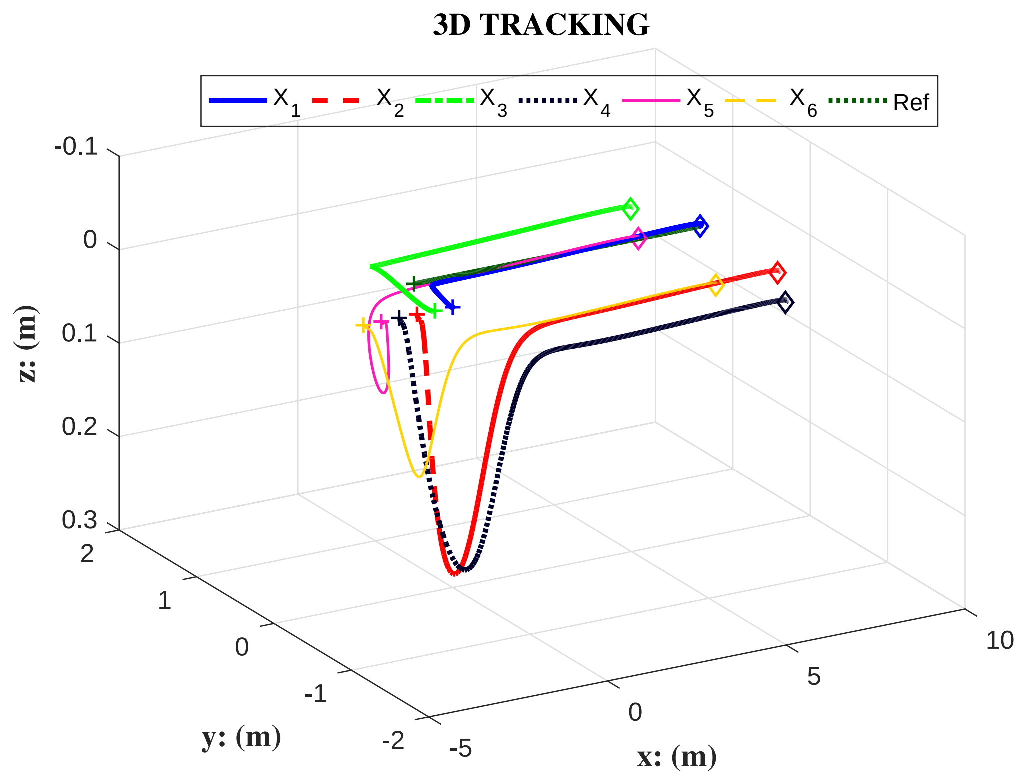


Figure 5.10. 3. Hexagonal 6-vehicle Formation, weakly connected overall.



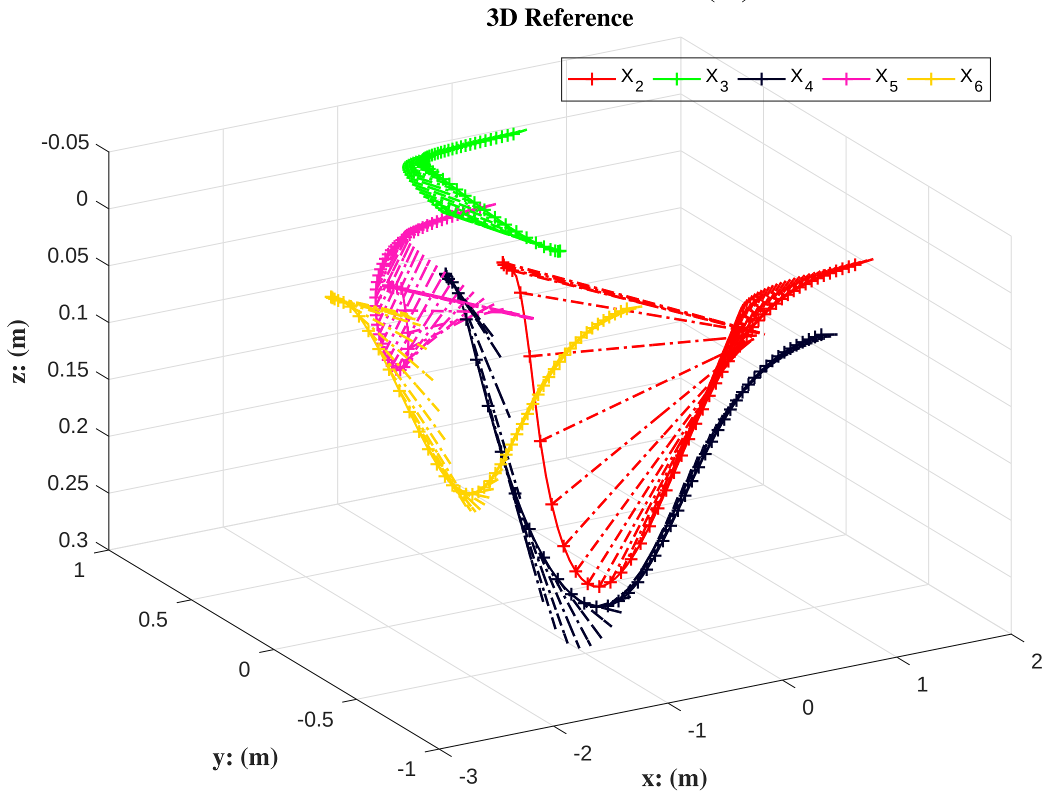


Figure 5.11. 3. Hexagonal 6-vehicle formation, weakly connected overall.

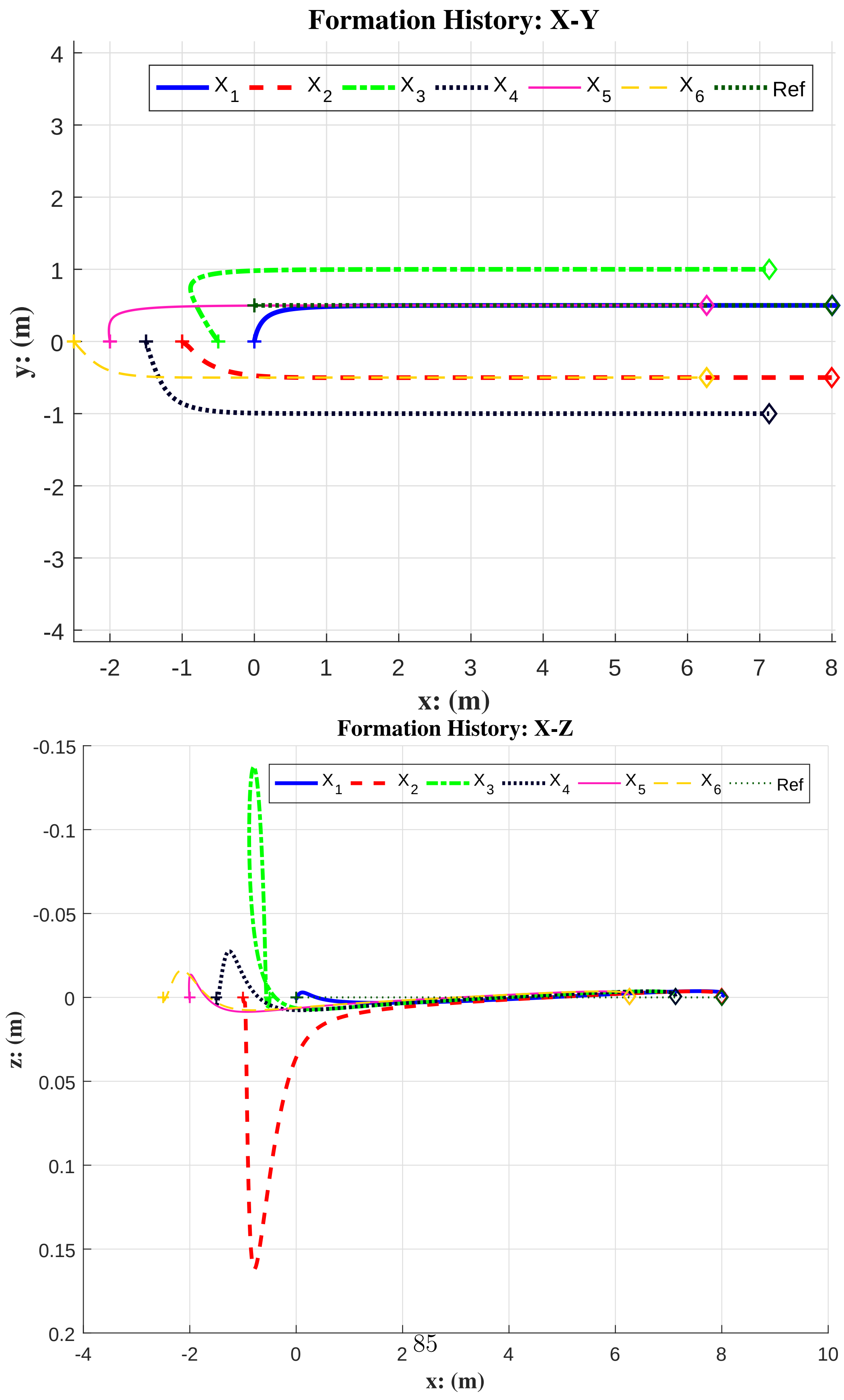


Figure 5.12. 4. Hexagonal 6-vehicle formation, weakly connected overall, followers strongly connected.

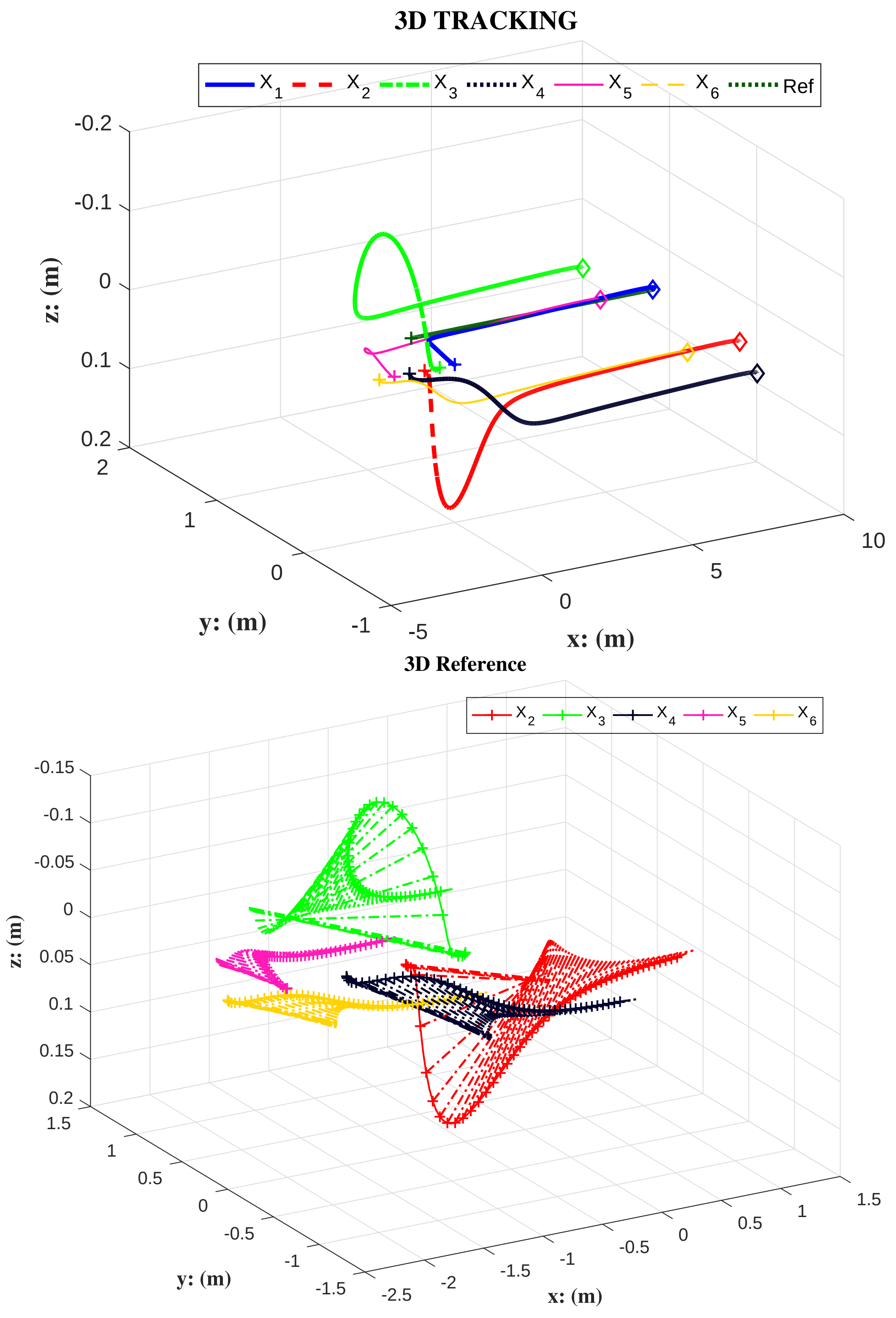


Figure 5.13. 4. Hexagonal 6-vehicle formation, weakly connected overall, followers strongly connected.

CHAPTER 6

Cooperative Estimation

Here, the cooperative estimation problem is solved with a dynamically varying inter-vehicle communication topology using limited measurements of a moving target. The conditions for a successful estimation problem are provided and the algorithm is evaluated in representative simulations. A covariance intersection based fusion with multiple estimates is performed that de-emphasizes target position estimates associated with large covariances, which typically happens when a vehicle in the team doesn't receive sufficient measurements for extended periods of time. Cooperative control is not the primary focus of this chapter, so long as this vehicle is in communication with some others that do obtain direct target measurements, it could be steered towards the target and stays close to the team.

6.1 Problem Formulation

Consider a set of N_v vehicles, each equipped with range sensors, that is able to detect the absolute distance between itself and the target. At any time instant, t_k, let $\overrightarrow{r}_i = [x_i \ y_i \ z_i]^T, (i = 1, \cdots, N_v)$ represent the Inertial position of the i-th vehicle. At this instant, let $\overrightarrow{r}_t = [x_t \ y_t \ z_t]^T$ represent the Inertial position of the target. The distance between any two vehicles is denoted as $d_{ij} = \|\overrightarrow{r}_j - \overrightarrow{r}_i\|$. Note, $\|\overrightarrow{r}_j - \overrightarrow{r}_i\| = \sqrt{(\overrightarrow{r}_j - \overrightarrow{r}_i) \cdot (\overrightarrow{r}_j - \overrightarrow{r}_i)}$. The operator '$\cdot$' in $\overrightarrow{p} \cdot \overrightarrow{q}$ represents the dot product between the two vectors $\overrightarrow{p}$ and $\overrightarrow{q}$.

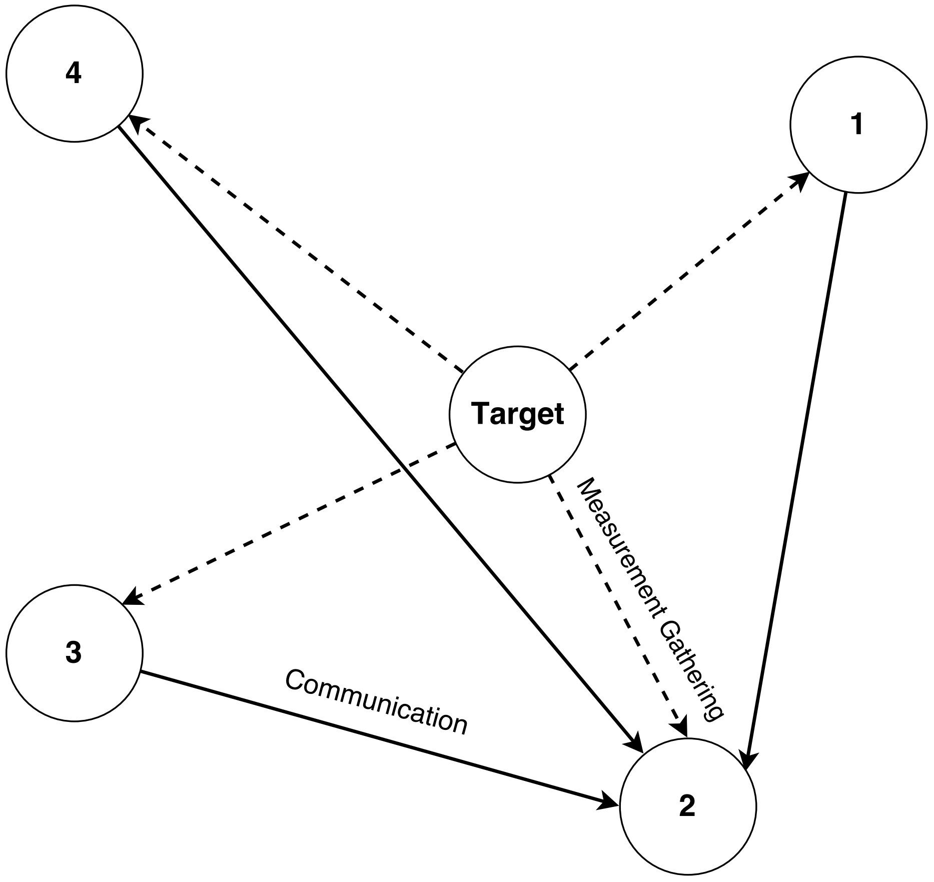


Figure 6.1. Example scenario: single target and multiple unmanned aerial vehicles.

Fig. (6.1) shows a typical formation scenario for the cooperative estimation problem as solved in this paper (Graph is in 2D). All agent vehicles are procuring measurements from the target. The measurements can be range and angles. At the same time, they also have inter-vehicle communications with other neighboring vehicles, i.e., in Fig. (6.1), vehicle 2 also gets information from vehicle 1,3, and 4. This allows vehicle 2 to make a better estimate of the location of the vehicle.

Define the operator '$\wedge$' to denote the enumerated list of vehicles in communication with any specific vehicle (including communication with itself) '**and**' can transmit target range information. For example, as per Fig. (6.1), $< 1 \wedge \{t, 1, 2, 3, 4\} > \ = \{t, 1\}$, $< 2 \ \wedge \ \{t, 1, 2, 3, 4\} > \ = \ \{t, 1, 2, 3, 4\}$. Also define $\lambda_i = \lambda(i, \{\cdots\}) \in \mathbb{Z}^+$ to denote the size of this enumerated list. Thus, $\lambda_1 = \lambda(1, \{t, 1, 2, 3, 4\}) = 2$ and $\lambda_2 = \lambda(2, \{t, 1, 2, 3, 4\}) = 5$. The trilateration based estimation problem is posed for all i when $\lambda_i \geq 4$ and is explained in the next section. When $\lambda_i < 4$, the target state estimate is constructed using an extended Kalman filter (EKF) utilizing an appropriate target model.

Problem Statement: Given N_v vehicles in a group with some inter-vehicle connection topology at time instant t_k, with all or a subset of the group receiving measurements from a moving target, the objective is to estimate the inertial position of the target such that the estimation errors $\| [\tilde{x}_t \ \tilde{y}_t \ \tilde{z}_t]_k^T \|$ are minimized. $[\tilde{x}_t \ \tilde{y}_t \ \tilde{z}_t]_k^T = [\hat{x}_t \ \hat{y}_t \ \hat{z}_t]_k^T - [x_t \ y_t \ z_t]_k^T$, where $[\hat{x}_t \ \hat{y}_t \ \hat{z}_t]_k^T$ is the fused estimate of the target true position $[x_t \ y_t \ z_t]_k^T$ at time t_k.

6.2 Solution Methodology

The solution methodology for the problem outlined in the previous section is presented here. The algorithm for implementing the cooperative estimation scheme is illustrated. The details of the steps involved in the estimation process are outlined in the next sections.

At any given time instant t_k,

```
**for** $i = 1 : N_v$ **do**
    **if** $\lambda_i \geq 4$ **then**
        $\rightarrow$ solve *trilateration* using nonlinear least squares.
        $\rightarrow$ Store ${}^i\hat{\vec{r}}_{t,k}$ (Target Position Estimate), ${}^i\boldsymbol{P}_{t,k}$ (Covariance)
        $\rightarrow$ Fuse all $\left({}^i\hat{\vec{r}}_{t,k},\ {}^i\boldsymbol{P}_{t,k}\right)$
    else
        → Implement EKF
        → Store ${}^{i}\hat{\vec{r}}_{t,k}$ (Target Position Estimate), ${}^{i}\boldsymbol{P}_{t,k}$ (Covariance)
    end if
end for
```

The algorithm is repeated for all $k \rightarrow \infty$.

6.2.1 Constant Velocity Target Model with Only Range Measurements

6.2.1.1 Trilateration based Estimation

The trilateration based estimation procedure is outlined in this section. If vehicle i can access the information from at least three other vehicles apart from getting the target position directly, i.e., $\lambda_i \geq 4$, the target dynamics is not required and a nonlinear least squares algorithm can be solved to determine the target's Inertial position. Assume the two neighboring vehicles that i can access information from are j_1, j_2, and j_3 the relations between the target location and locations of the vehicles at time t_k can be represented as:

$$\begin{aligned}
(x_i - x_t)^2 + (y_i - y_t)^2 + (z_i - z_t)^2 &= d_{ti}^2 \\
(x_{j_1} - x_t)^2 + (y_{j_1} - y_t)^2 + (z_{j_1} - z_t)^2 &= d_{tj_1}^2 \\
(x_{j_2} - x_t)^2 + (y_{j_2} - y_t)^2 + (z_{j_2} - z_t)^2 &= d_{tj_2}^2 \\
(x_{j_3} - x_t)^2 + (y_{j_3} - y_t)^2 + (z_{j_3} - z_t)^2 &= d_{tj_3}^2
\end{aligned} \tag{6.1}$$

Note, in Eq. (6.1), x_i, y_i, z_i, x_{j_1}, y_{j_1}, z_{j_1}, x_{j_2}, y_{j_2}, z_{j_2}, x_{j_3}, y_{j_3}, z_{j_3} are known from the individual position data that is exchanged. Additionally, d_{ti}, d_{tj_1}, d_{tj_2}, and d_{tj_3} are also exchanged. The only unknowns are x_t, y_t, and z_t that can be solved using a nonlinear least squares algorithm.

Eq. (6.1) can also be written as, $\boldsymbol{h}(x_t, y_t, z_t, \boldsymbol{p}) = \boldsymbol{d}$, where

$$\boldsymbol{p} = [x_i \; y_i \; z_i \; x_{j_1} \; y_{j_1} \; z_{j_1} \; x_{j_2} \; y_{j_2} \; z_{j_2} \; x_{j_3} \; y_{j_3} \; z_{j_3}]^T$$

and $\boldsymbol{d} = \left[d_{ti}^2 \; d_{tj_1}^2 \; d_{tj_2}^2 \; d_{tj_3}^2\right]^T$.

The Jacobian of $\boldsymbol{h}(x_t, y_t, z_t, \boldsymbol{p})$ with respect to the target state at time t_k is represented as $\boldsymbol{H}_k(x_t, y_t, z_t, \boldsymbol{p}) = \dfrac{\partial \boldsymbol{h}}{\partial \boldsymbol{r}_t}\Big|_{t_k}$ and can be expressed as,

$$\boldsymbol{H}_k(x_t, y_t, z_t, \boldsymbol{p}) = -2 \begin{bmatrix} x_i - x_t & y_i - y_t & z_i - z_t \\ x_{j_1} - x_t & y_{j_1} - y_t & z_{j_1} - z_t \\ x_{j_2} - x_t & y_{j_2} - y_t & z_{j_2} - z_t \\ x_{j_3} - x_t & y_{j_3} - y_t & z_{j_3} - z_t \end{bmatrix}$$

The covariance of the target's Inertial position estimate can be obtained as ${}^{i}\boldsymbol{P}_{t,k} = \left[\boldsymbol{H}_k(x_t, y_t, z_t, \boldsymbol{p})^T \boldsymbol{H}_k(x_t, y_t, z_t, \boldsymbol{p})\right]^{-1}$

6.2.1.2 Target Model based Extended Kalman Filter Estimation

If vehicle i can only access one other neighboring vehicle's information or none, the method detailed in previous subsection would be insufficient to determine the location of the target. To help better estimate the target position under similar circumstances, we introduce a target model and use it for target state propagation. Assume the model of the target can be shown as Eq. (6.2).

$$\dot{\mathbf{r}}_t = \mathbf{f}(\mathbf{r}_t, \mathbf{u}) + \mathbf{G}\mathbf{w} \tag{6.2}$$

The output of the system is defined as range measurements the vehicle is able to access:

$$\tilde{\mathbf{y}}_k = \mathbf{h}(\mathbf{r}_t) + \mathbf{v}_k = \begin{bmatrix} (\mathbf{r}_i - \mathbf{r}_t)^T(\mathbf{r}_i - \mathbf{r}_t) \\ (\mathbf{r}_{j_1} - \mathbf{r}_t)^T(\mathbf{r}_{j_1} - \mathbf{r}_t) \\ \vdots \end{bmatrix} + \mathbf{v}_k \tag{6.3}$$

Where $\mathbf{w} \sim N(\mathbf{0}, \mathbf{Q(t)})$ and $\mathbf{v}_k \sim N(\mathbf{0}, \mathbf{R})$ represent the process noise and measurement noise respectively and are assumed to be uncorrelated. Based on the model and the output, a continuous-discrete extended Kalman filter (EKF) can be used to estimate the target location. The EKF formulation can be summarized as follows:

$$\begin{array}{lrll}
\text{Initialization:} & \hat{\mathbf{r}}_t(t_0) & = \hat{\mathbf{r}}_{t0} & \\
 & \mathbf{P}_0 & = E\{\tilde{\mathbf{r}}_{t0}(\hat{\mathbf{r}}_{t0})^T\} & \\
\text{Gain Update:} & \mathbf{K}_k^- & = \mathbf{P}_k^-\mathbf{H}_k^T(\hat{\mathbf{r}}_{tk}^-)[\mathbf{H}_k(\hat{\mathbf{r}}_{tk}^-)\mathbf{P}_k^-\mathbf{H}_k^T(\hat{\mathbf{r}}_{tk}^-)] & (6.4) \\
 & \mathbf{H}_k(\hat{\mathbf{r}}_{tk}^-) & = \left.\dfrac{\partial \mathbf{h}}{\partial \mathbf{r}_t}\right|_{\hat{\mathbf{r}}_{tk}^-} & \\
\text{State \& Covariance} & \hat{\mathbf{r}}_{tk}^+ & = \hat{\mathbf{r}}_{tk}^- + \mathbf{K}_k[\tilde{\mathbf{y}}_k - \mathbf{h}(\hat{\mathbf{r}}_{\mathbf{tk}}^-)] & (6.5) \\
\text{Update:} & \mathbf{P}_k^+ & = [\mathbf{I} - \mathbf{K}_k\mathbf{H}_k(\hat{\mathbf{r}}_k^-)]\mathbf{P}_k^- & (6.6) \\
\text{State \& Covariance} & \dot{\hat{\mathbf{r}}}_t & = \mathbf{f}(\hat{\mathbf{r}}_t, \mathbf{u}) & (6.7) \\
\text{Propagation:} & \dot{\mathbf{P}}(t) & = \mathbf{F}(t)\mathbf{P}(t) + \mathbf{P}(t)\mathbf{F}(t)^T + \mathbf{G}(t)\mathbf{Q}(t)\mathbf{G}(t)^T & (6.8) \\
 & \mathbf{F}(t) & = \left.\dfrac{\partial \mathbf{f}}{\partial \hat{\mathbf{r}}_t}\right|_{\hat{\mathbf{r}}_t, \mathbf{u}} & (6.9)
\end{array}$$

Where

$$\left.\frac{\partial \mathbf{h}}{\partial \mathbf{r}_t}\right|_{\hat{\mathbf{r}}_{tk}^-} = 2\begin{bmatrix} (\hat{x}_t - x_i) & (\hat{y}_t - y_i) & (\hat{z}_t - z_i) \\ (\hat{x}_t - x_{j_1}) & (\hat{y}_t - y_{j_1}) & (\hat{z}_t - z_{j_1}) \\ & \vdots & \end{bmatrix}$$

6.2.2 Point Mass Aircraft Model with Range, Azimuth Angle, and Elevation Angle Measurements

In this case, vehicle i can receive measurements of range, azimuth angle, and elevation angle from UAV to Target. The continuous-time state model and the discrete-time measurements for reference UAV can be summarized as:

$$\begin{aligned}\dot{\mathbf{X}}_t &= \mathbf{f}_2(\mathbf{X}_t, \mathbf{u}^c, t) + \mathbf{G}_2\mathbf{w}_2 \\ \tilde{\mathbf{y}}_k &= \mathbf{h}_2(\mathbf{X}_k) + \mathbf{v}_k \end{aligned} \tag{6.10}$$

where $\mathbf{X}_t = [x_t,\ y_t,\ z_t,\ V_t,\ \gamma,\ \chi]^T$ is the state vector; $\tilde{\mathbf{y}}_k = [r_k,\ \phi_k,\ \theta_k]^T$ is the measurement vector; $\mathbf{u}^c = [V^c,\ a_v^c,\ a_h^c]^T$ is the input vector.

And the target model $\mathbf{f}_2$ used here is:

$$\begin{aligned} \dot{x}_t &= V_t \cos\chi\cos\gamma & \dot{V}_t &= V^c - V_t \\ \dot{y}_t &= V_t \sin\chi\cos\gamma & \dot{\gamma} &= \frac{a_v^c}{V_t} \\ \dot{z}_t &= V_t \sin\gamma & \dot{\chi} &= \frac{a_h^c}{V_t\cos\gamma} \end{aligned}$$

Notice here target's (V^c, a_v^c, a_h^c) are not available to agents.

The measurement function is :

$$\mathbf{h}_2(\mathbf{X}_k) = \begin{bmatrix} \sqrt{(x_t - x_i)^2 + (y_t - y_i)^2 + (z_t - z_i)^2} \\ \tan^{-1}(\frac{y_t - y_i}{x_t - x_i}) \\ \sin^{-1}(\frac{z_t - z_i}{\sqrt{(x_t-x_i)^2+(y_t-y_i)^2+(z_t-z_i)^2}}) \end{bmatrix} + \begin{bmatrix} v_r \\ v_\phi \\ v_\theta \end{bmatrix} \tag{6.11}$$

A few matrices that are needed for the EKF process are defined here as well:

$$\mathbf{F} = \left.\frac{\partial \mathbf{f}_2}{\partial \hat{\mathbf{X}}_t}\right|_{\hat{\mathbf{X}}_t} = \begin{bmatrix} \mathbf{0}_{3\times 3} & F_{12} \\ \mathbf{0}_{3\times 3} & F_{22} \end{bmatrix} \tag{6.12}$$

$$\mathbf{H} = \left.\frac{\partial \mathbf{h}_2}{\partial \hat{\mathbf{X}}_t}\right|_{\hat{\mathbf{X}}_t} = \begin{bmatrix} H_1 & H_2 \end{bmatrix} \tag{6.13}$$

where

$$F_{12} = \begin{bmatrix} \cos\hat{\gamma}\cos\hat{\chi} & -\hat{V}_t\sin\hat{\gamma}\cos\hat{\chi} & -\hat{V}_t\cos\hat{\gamma}\sin\hat{\chi} \\ \cos\hat{\gamma}\sin\hat{\chi} & -\hat{V}_t\sin\hat{\gamma}\sin\hat{\chi} & \hat{V}_t\cos\hat{\gamma}\cos\hat{\chi} \\ \sin\hat{\gamma} & \hat{V}_t\cos\hat{\gamma} & 0 \end{bmatrix}, \quad F_{22} = \begin{bmatrix} -1 & 0 & 0 \\ 0 & 0 & 0 \\ 0 & 0 & 0 \end{bmatrix}$$

$$H_1 = \begin{bmatrix} \frac{\hat{x}_t - x_i}{\hat{r}} & \frac{\hat{y}_t - y_i}{\hat{r}} & \frac{\hat{z}_t - z_i}{\hat{r}} \\ \frac{y_i - \hat{y}_t}{(\hat{x}_t - x_i)^2 + (\hat{y}_t - y_i)^2} & \frac{\hat{x}_t - x_i}{(\hat{x}_t - x_i)^2 + (\hat{y}_t - y_i)^2} & \\ -\frac{(\hat{x}_t - x_i)(\hat{z}_t - z_i)}{\hat{r}^2\sqrt{(\hat{x}_t - x_i)^2 + (\hat{y}_t - y_i)^2}} & -\frac{(\hat{y}_t - y_i)(\hat{z}_t - z_i)}{\hat{r}^2\sqrt{(\hat{x}_t - x_i)^2 + (\hat{y}_t - y_i)^2}} & \frac{\sqrt{(\hat{x}_t - x_i)^2 + (\hat{y}_t - y_i)^2}}{\hat{r}^2} \end{bmatrix}$$

and $H_2 = \mathbf{0}_{3\times3}$, $\hat{r} = \sqrt{(\hat{x}_t - x_i)^2 + (\hat{y}_t - y_i)^2 + (\hat{z}_t - z_i)^2}$.

In the above formulation, $\mathbf{h}_2(\mathbf{X}_k)$ and $\mathbf{H}$ can be expanded to accommodate measurements received from other vehicles.

6.3 Simulation Results

Simulations are performed to show the effectiveness of the algorithm as shown in Fig. (6.2 - 6.5).

Fig. (6.6) showed the true target position progression w.r.t time for the point mass aircraft model.

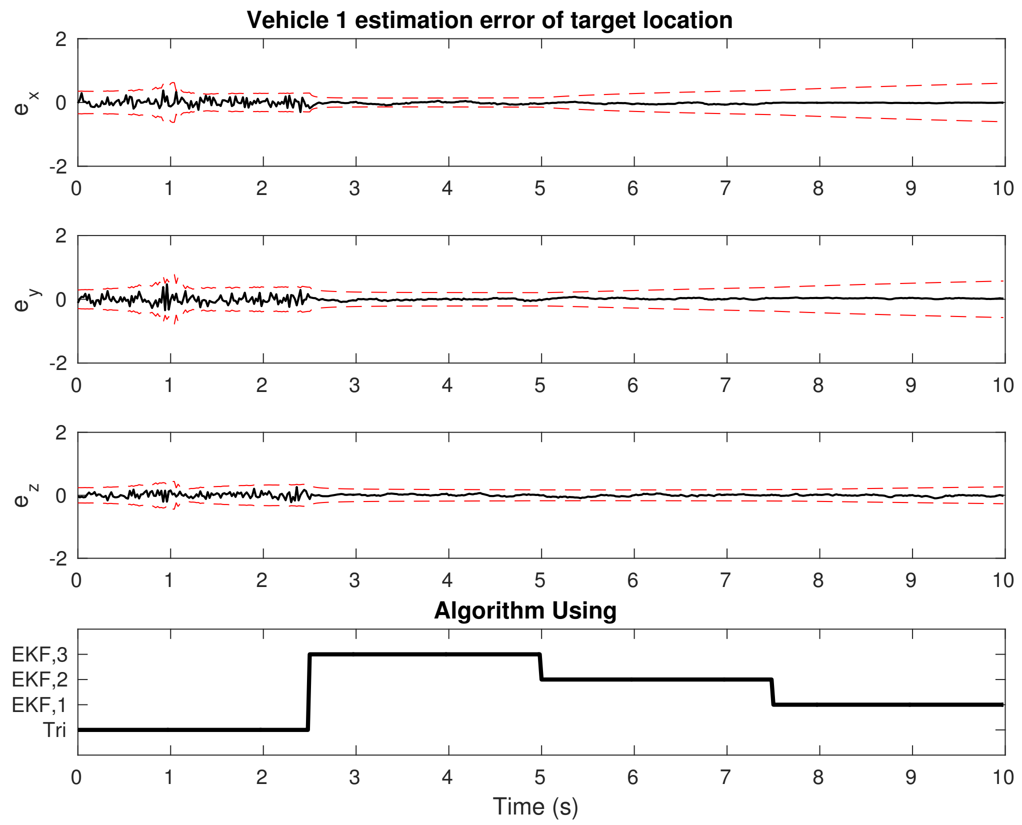
6.4 Concluding Remarks

In Fig. (6.2 - 6.5), the ylabel of the 4th subplot of each figure indicates which algorithm the vehicle is using. (EKF,1), (EKF,2), (EKF,3), and (EKF,4) means EKF is used with information from 1 vehicle, 2 vehicle, 3 vehicles, and 4 vehicles, respectively.

In the first plot of Fig. (6.2), at first, the vehicle is able to receive information from 3 other vehicles and trilateration base method is used. It can be seen that it estimates the target location reasonably well. Starting from $t = 2.5s$, the vehicle loses communication from one of the vehicles and starts using (EKF,3). The estimation becomes smoother since a approximate model of the target is introduced. At $t = 5s$, it lost one more accessible vehicle, and started using (EKF,2). The decrease of number of accessible vehicles also caused an increase of estimation covariance. Second plot of Fig. (6.2) showed the estimation error when its interaction topology is reversed.

In Fig. (6.3), the vehicles are using (EKF,3) and (EKF,2) throughout the simulation.

Fig. (6.4) and (6.5) showed the tracking performance with similar topology changes with with range, azimuth angle, and elevation angle measurements for a point mass aircraft model.



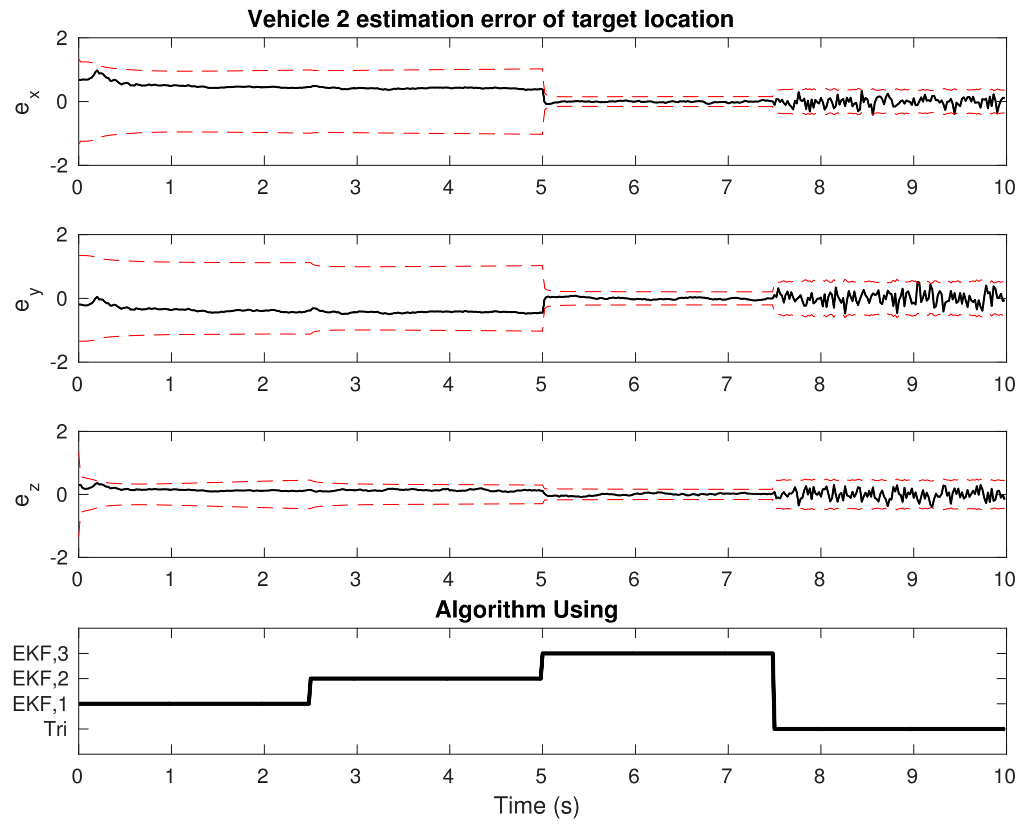


Figure 6.2. Constant velocity model, target position estimation from vehicle 1 and 2.

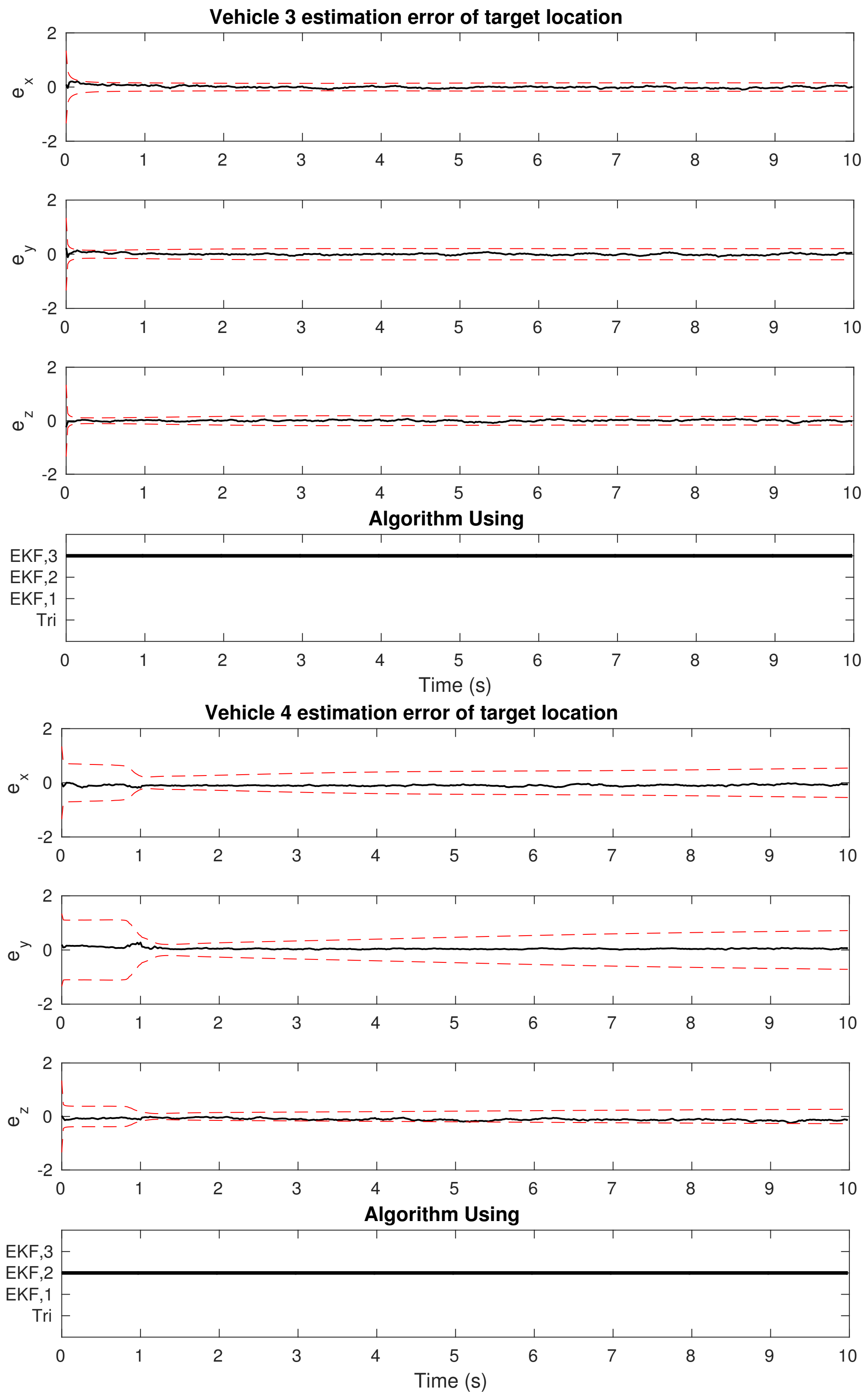


Figure 6.3. Constant velocity model, target position estimation from vehicle 3 and 4.

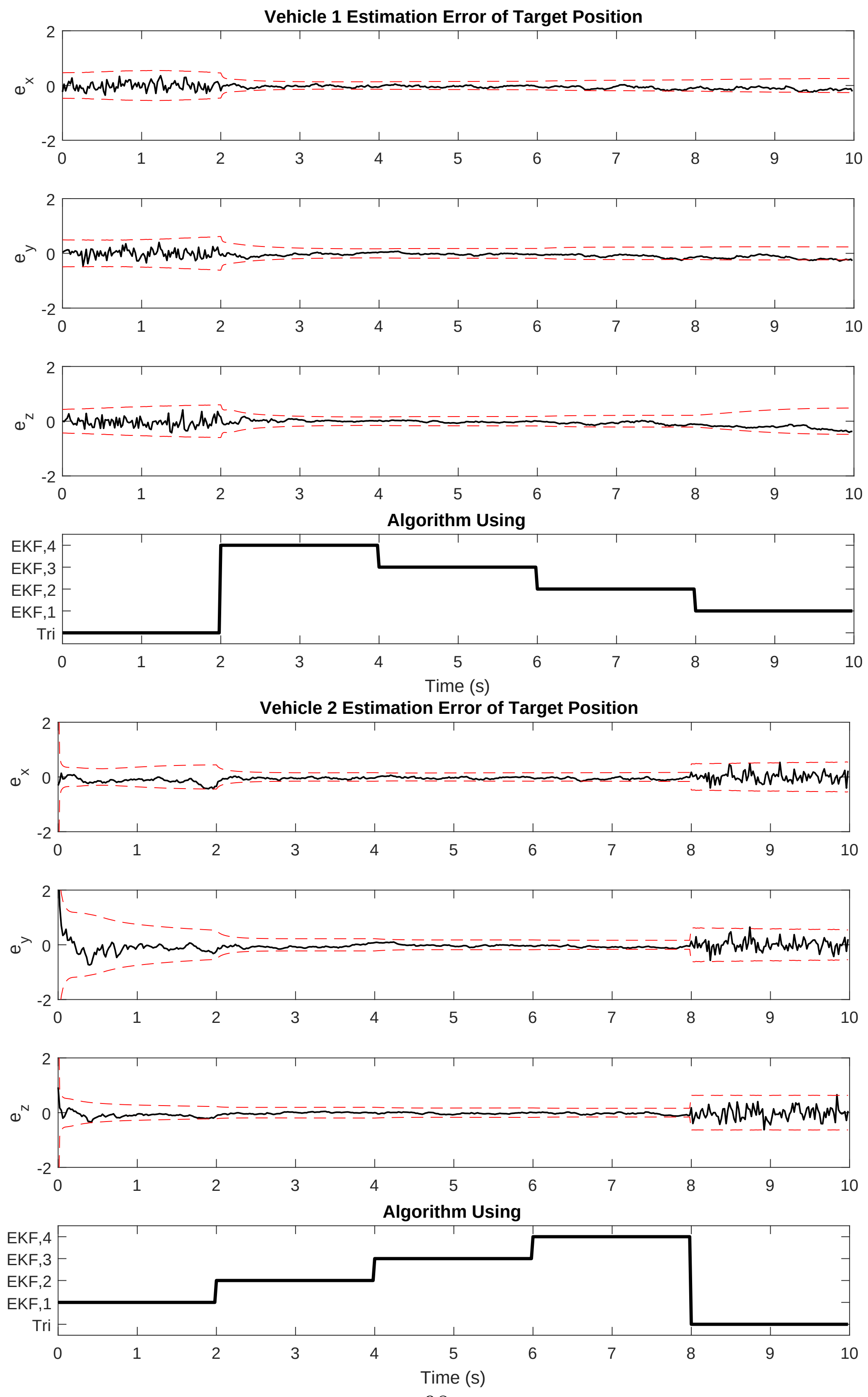
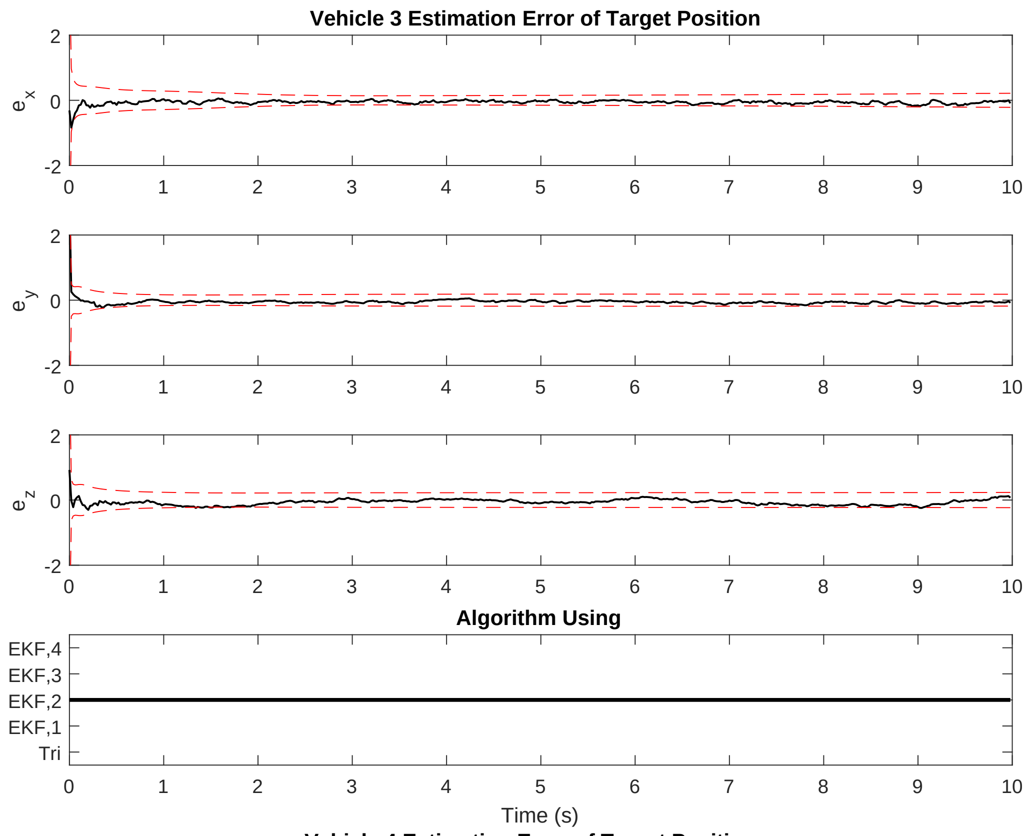


Figure 6.4. Point mass aircraft model, target position estimation from vehicle 1 and 2.



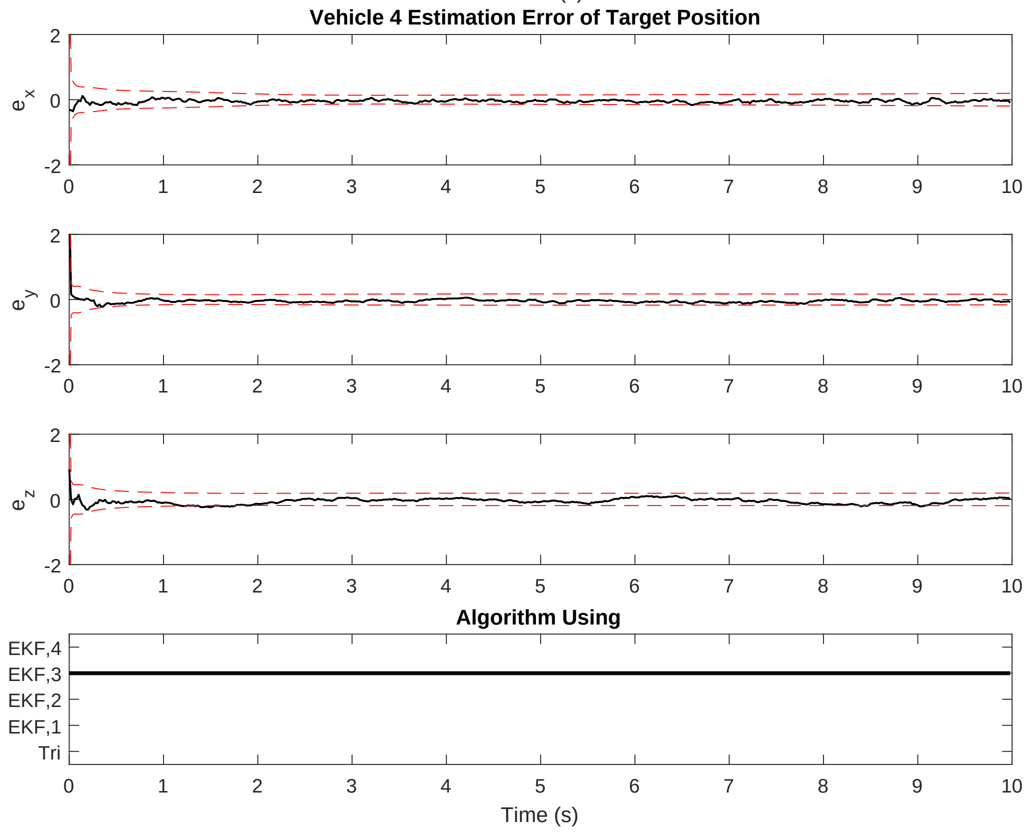


Figure 6.5. Point mass aircraft model, target position estimation from vehicle 3 and 4.

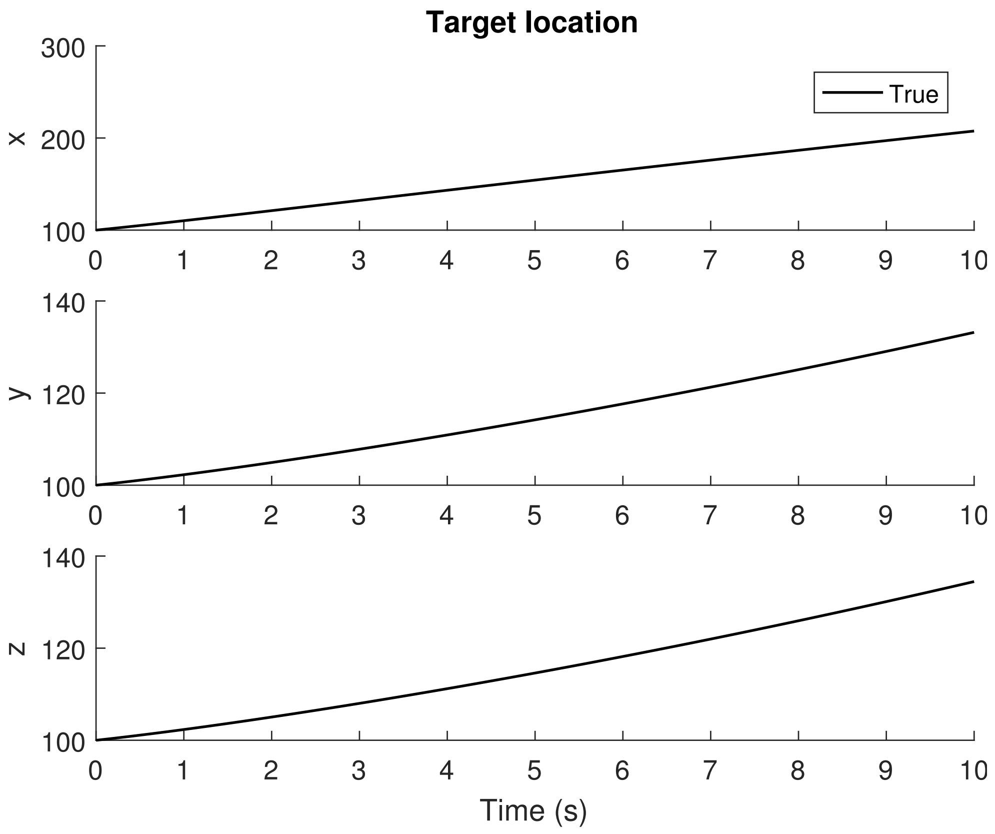


Figure 6.6. Point mass target model, true target position.

CHAPTER 7

Cooperative Control and Estimation

In this chapter, the cooperative control and estimation algorithms mentioned in preceding chapters are combined. Each vehicle generates an estimate of the target based on the measurements it made and the information it received from its accessible neighbors. The target becomes the leader here and the estimates each vehicle generates is used to track the target. In the cooperative control scheme, the agents have two objectives. First, maintain a desired relative distance to the target. Second, maintain a desired formation with the other agents. The two objectives are weighted based on their importance, which is characterized by their covariance. In simple terms, if target estimation is good, stay closer to target estimates. If target estimation is bad, stay closer to formation.

7.1 Solution Methodology

Fig. (7.1) shows the overall cooperation scheme. The process works as follows:

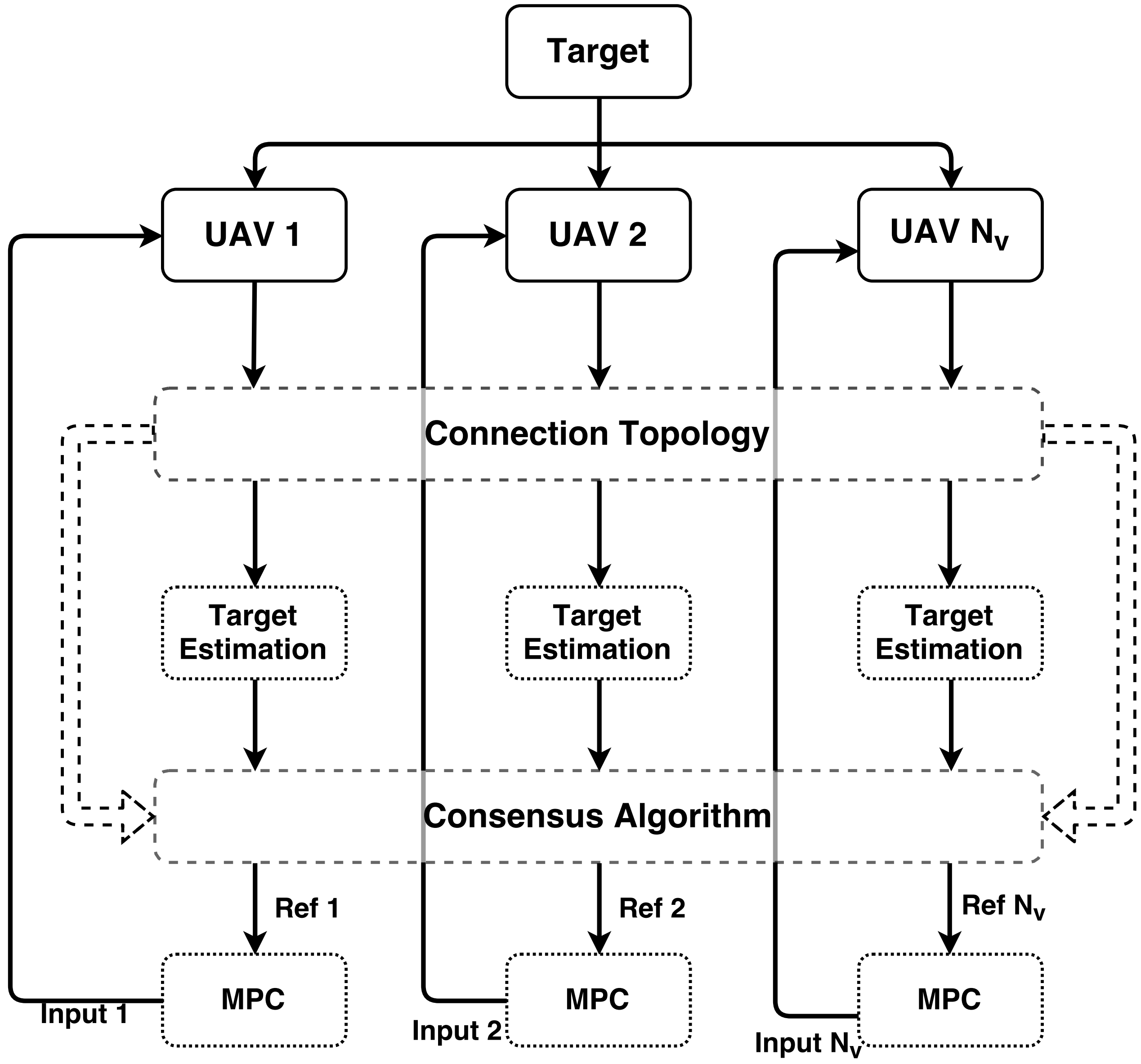


Figure 7.1. Cooperative control and estimation.

1. UAVs obtain measurements regarding the target, respectively.
2. For each vehicle, based on its own measurements and information it received from its communicating neighbours (described in ‘Connection Topology’block.), it generates a target estimation of its own.

3. There are two algorithms for target estimation for each vehicle:
   (a) Trilateration (when measurements are sufficient enough).
   (b) Extended Kalman filter (when measurements are not sufficient).
4. Target estimation and location information from communicating neighbours are fed into a consensus algorithm to generate reference trajectories for vehicles. This process is broken down into two steps, shown in Fig. (7.2).
   (a) Neighbour consensus: a desired location for vehicle 1 is generated based its neighbouring agents' location.

   A reference generation equation similar to Eq. (5.4) is used here:

$$\mathbf{x}_{k+n,f}^{i,r} = \frac{1}{\sum_{j=1}^{N_v} \mathbf{G}_{ij}} \left(\mathbf{G}_{ii}\mathbf{x}_{k+n-1}^{i,r} + \sum_{\substack{(i,j)\in\mathcal{E} \\ j\neq i}} \mathbf{G}_{ij}(\mathbf{x}_k^j + \mathbf{d}_{ij}) \right) \tag{7.1}$$

   where $n = 1, \cdots, N$.
   (b) Target estimation fusion: the desired location for vehicle 1 from previous step is then weighted and averaged with the desired location calculated from target estimate.

$$\mathbf{x}_{k+n}^{i,r} = (\mathbf{P}_{f_i}^{-1} + \mathbf{P}_{t_i}^{-1})^{-1} \left(\mathbf{P}_{f_i}^{-1}\mathbf{x}_{k+n,f}^{i,r} + \mathbf{P}_{t_i}^{-1}(\hat{\mathbf{x}}_t^i + \mathbf{d}_{it}) \right) \tag{7.2}$$

   where $\mathbf{x}_{k+n,f}^{i,r}$ is from Eq. (7.1), $\mathbf{d}_{it}$ is the relative position between vehicle i and target. And $\mathbf{P}_{f_i}$ is covariance value artificially assigned to generated desired location from accessible neighboring agents.
5. An MPC controller is used to compute control for each vehicle to track their respective reference trajectory.
6. That control inputs are then fed back to respective vehicles for execution.

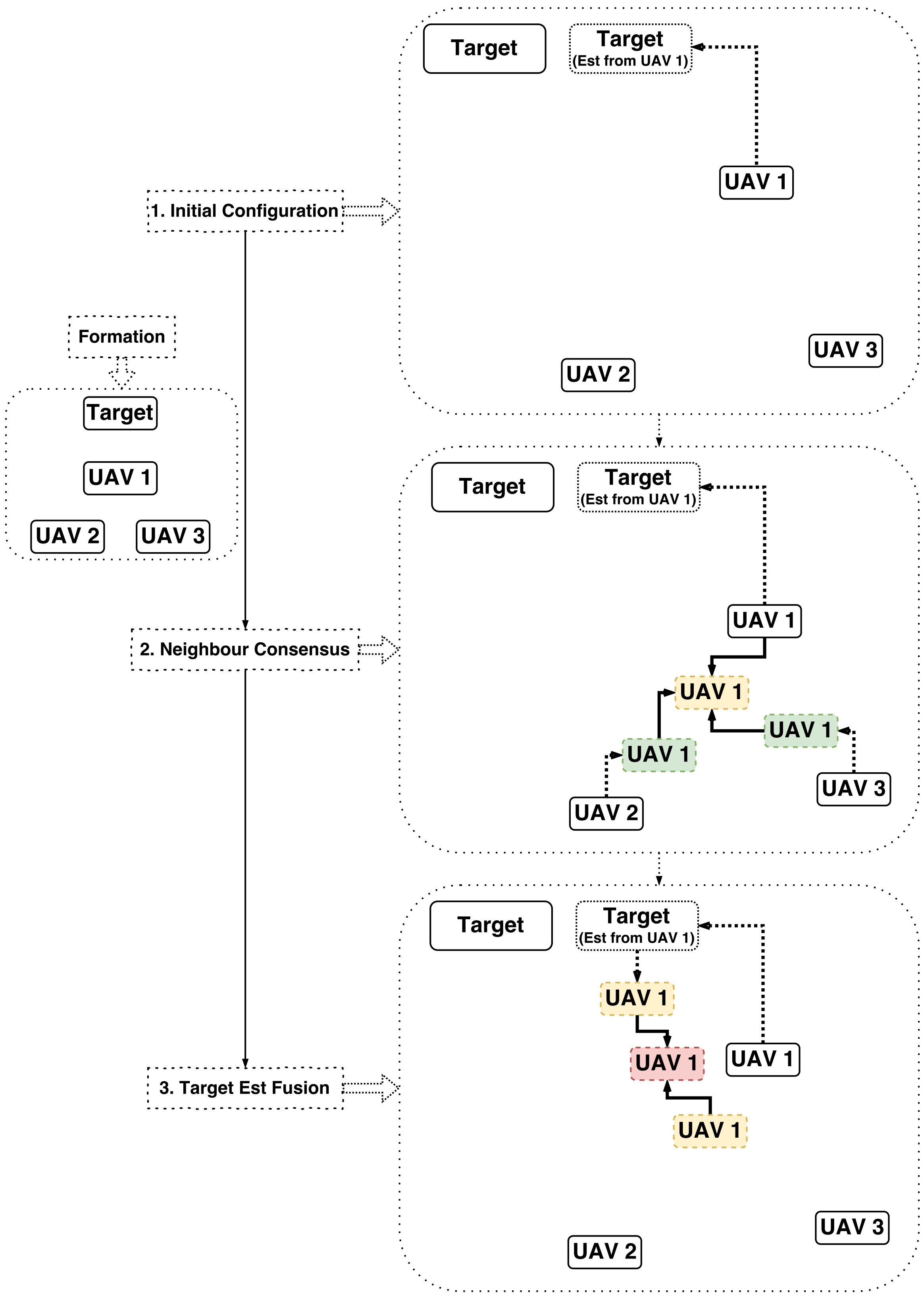


Figure 7.2. Reference generation for MPC.

7.2 Simulation Setup and Results

The desired formation is shown in Fig. (7.3). This formation here is slightly different than the one used in cooperative control to help better estimate z position of the target. Using the method detailed in the preceding section and setting $\mathbf{P}_{f_i} = \mathrm{diag}(0.1, 0.1, 0.1)$. Two simulations with different connection topology conditions are simulated and their results are shown in Fig. (7.4 - 7.9).

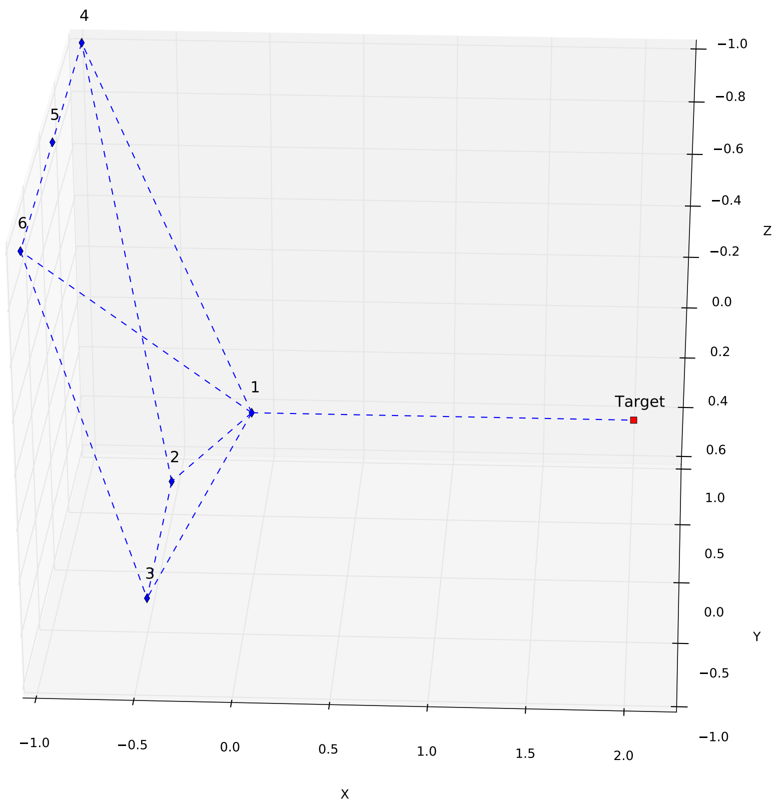
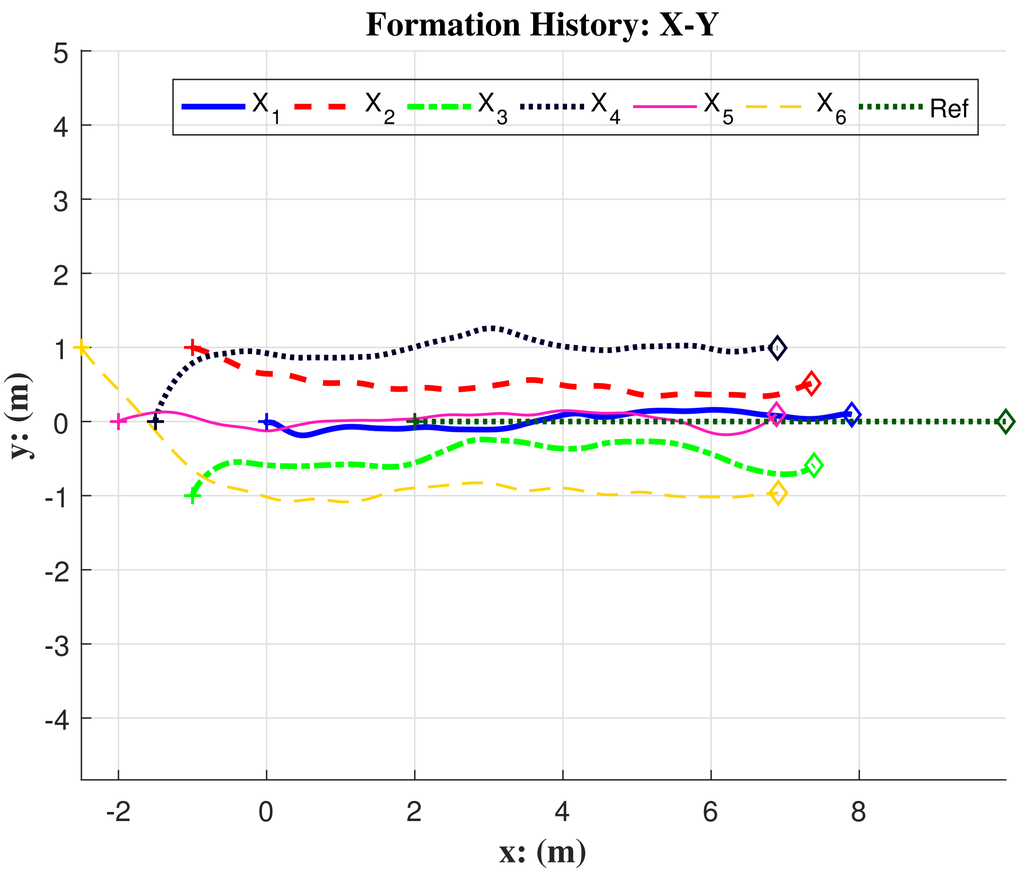


Figure 7.3. Desired formation for cooperative control and estimation.



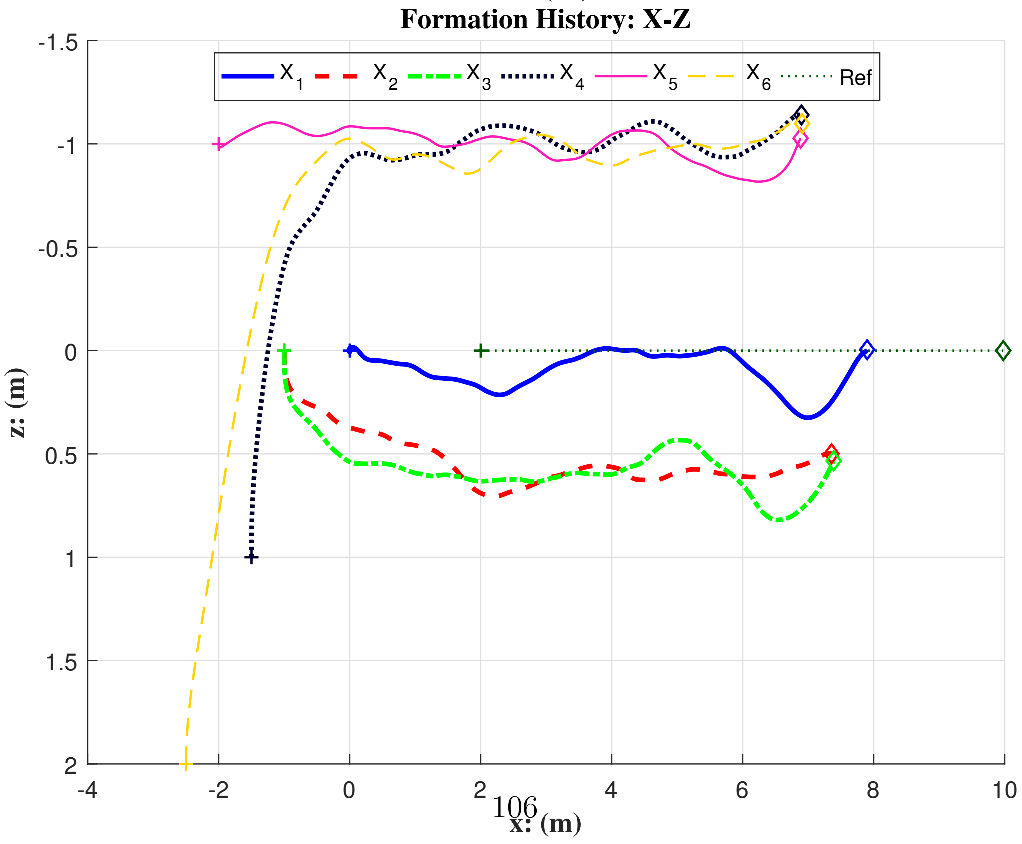
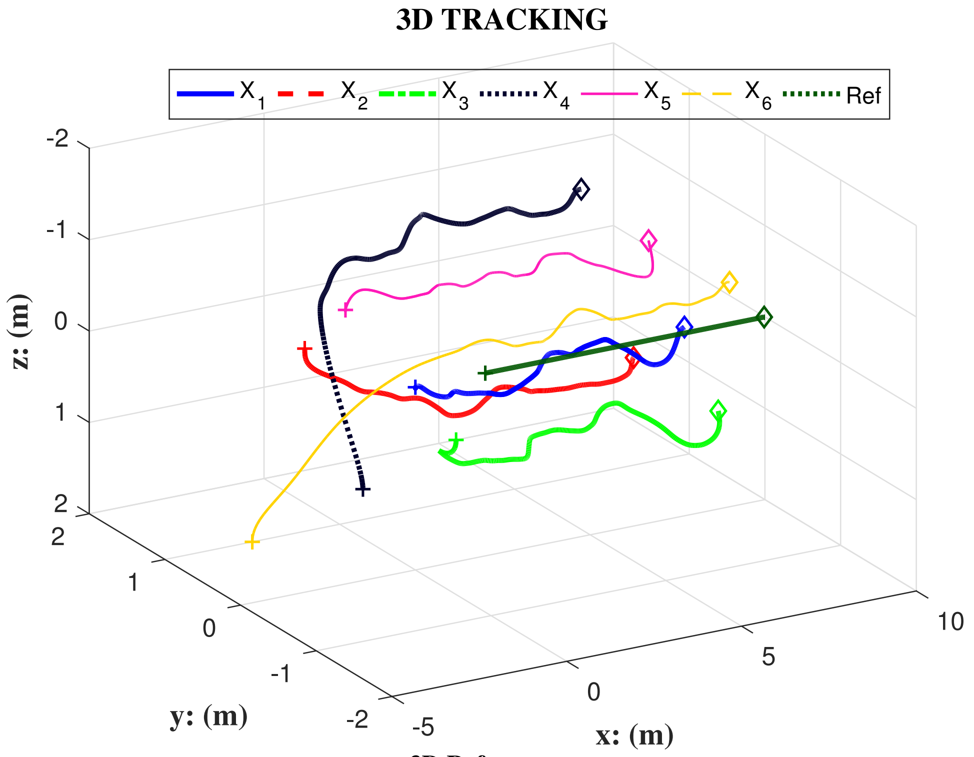


Figure 7.4. Case 1: XY and XZ of cooperative control and estimation.



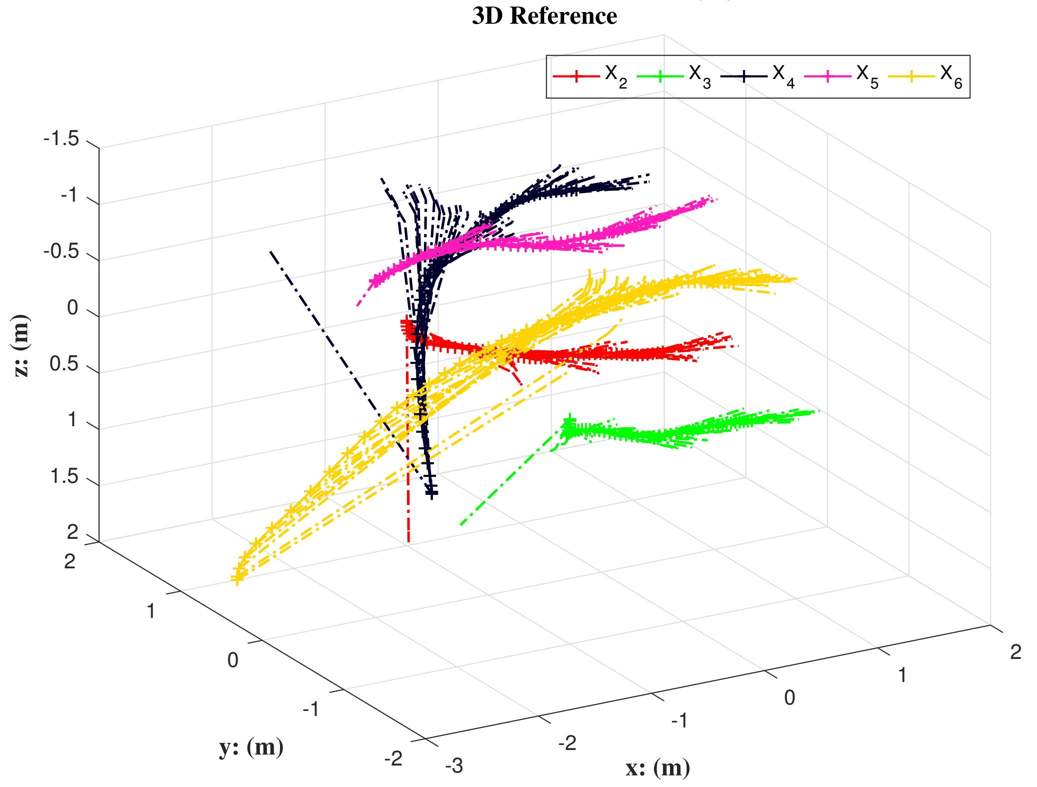


Figure 7.5. Case 1: 3D and references of cooperative control and estimation.

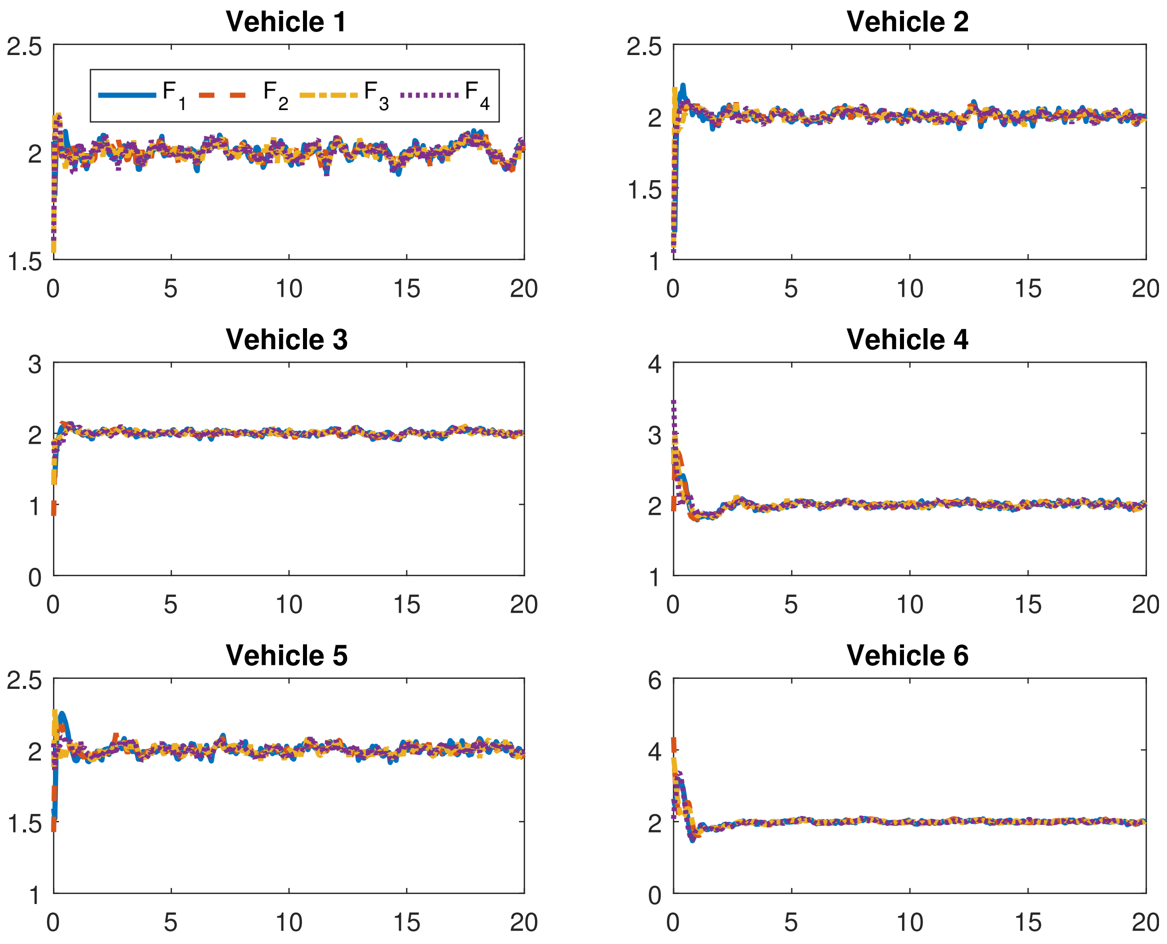
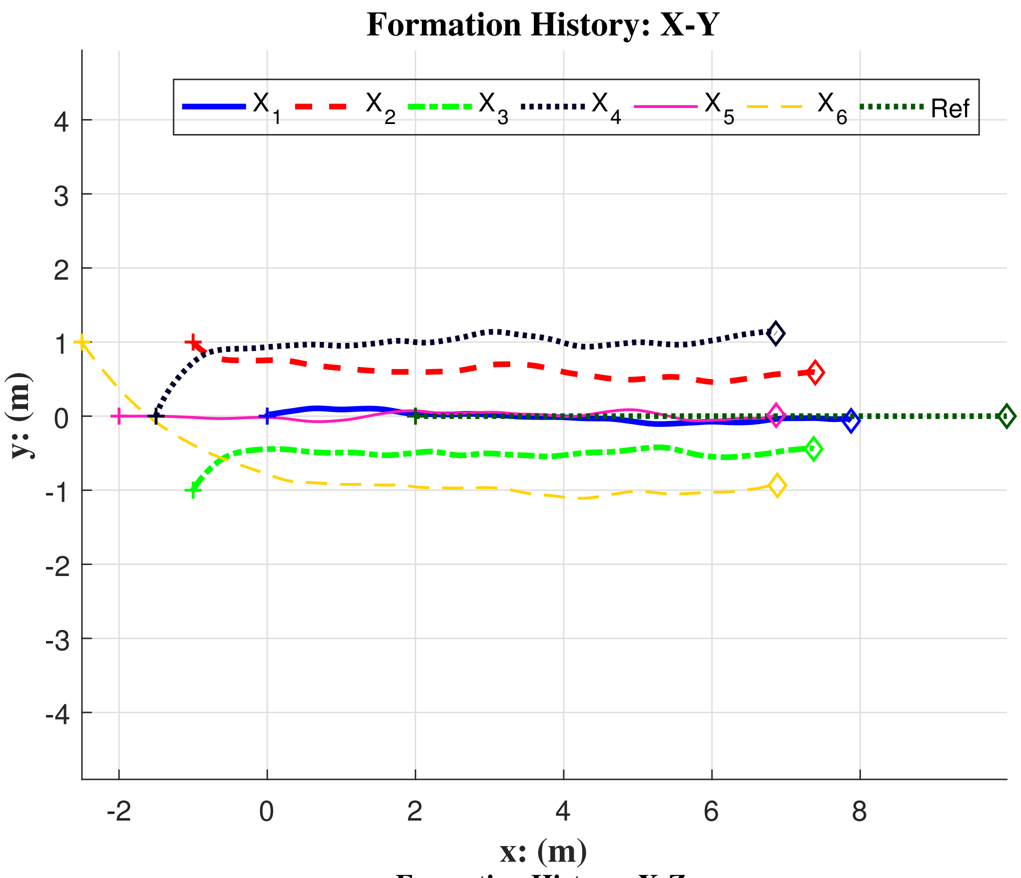


Figure 7.6. Case 1: Inputs of each vehicle of cooperative control and estimation.



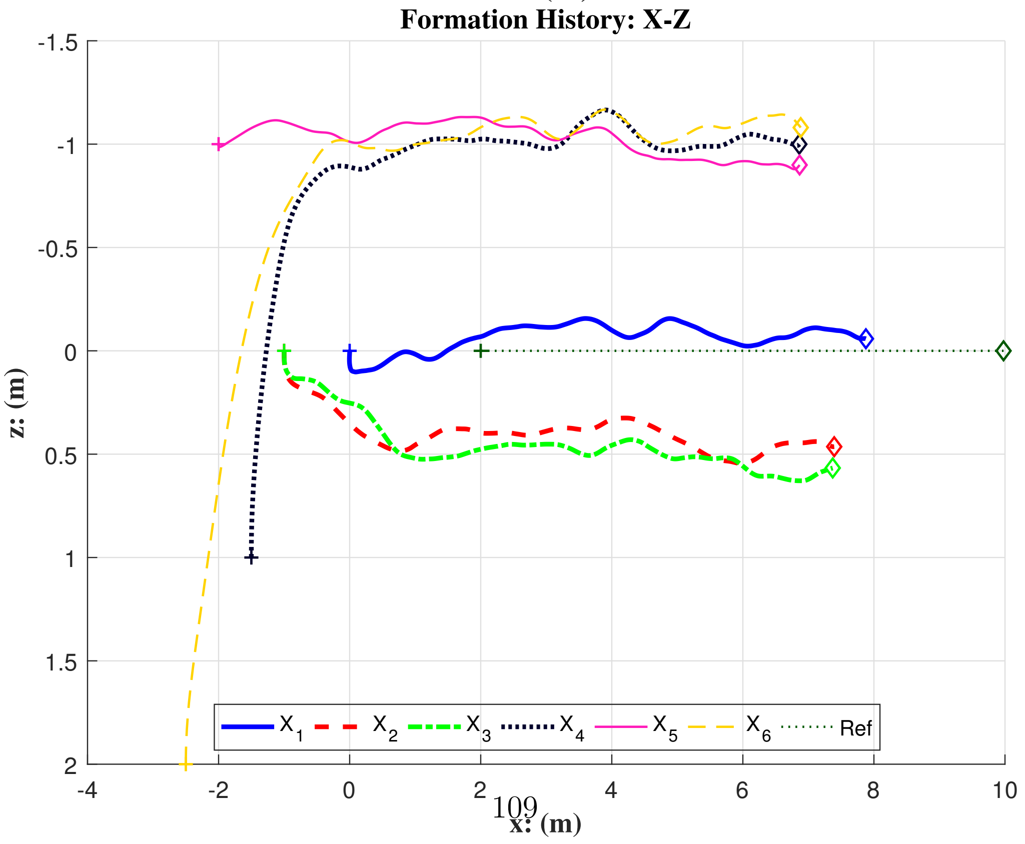
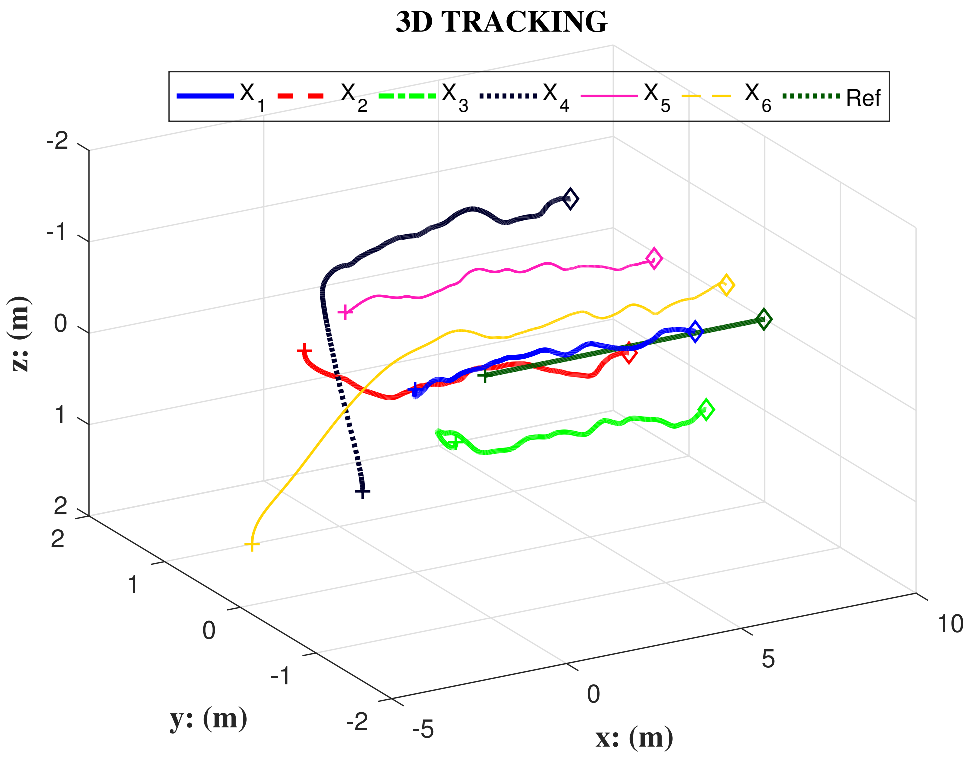


Figure 7.7. Case 2: XY and XZ of cooperative control and estimation.



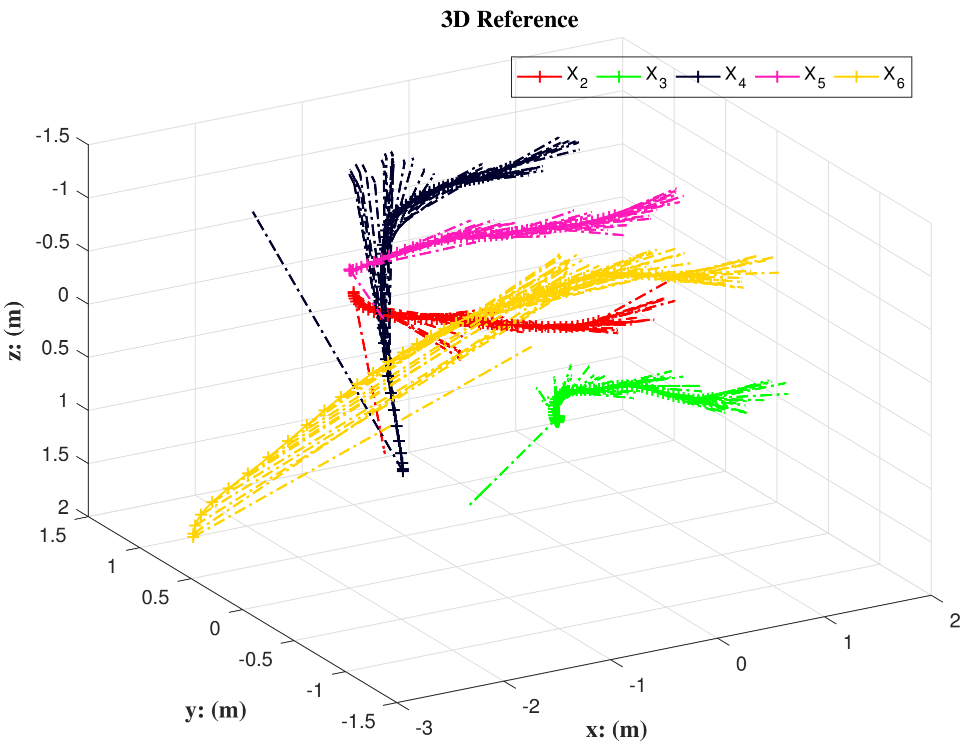


Figure 7.8. Case 2: 3D and references of cooperative control and estimation.

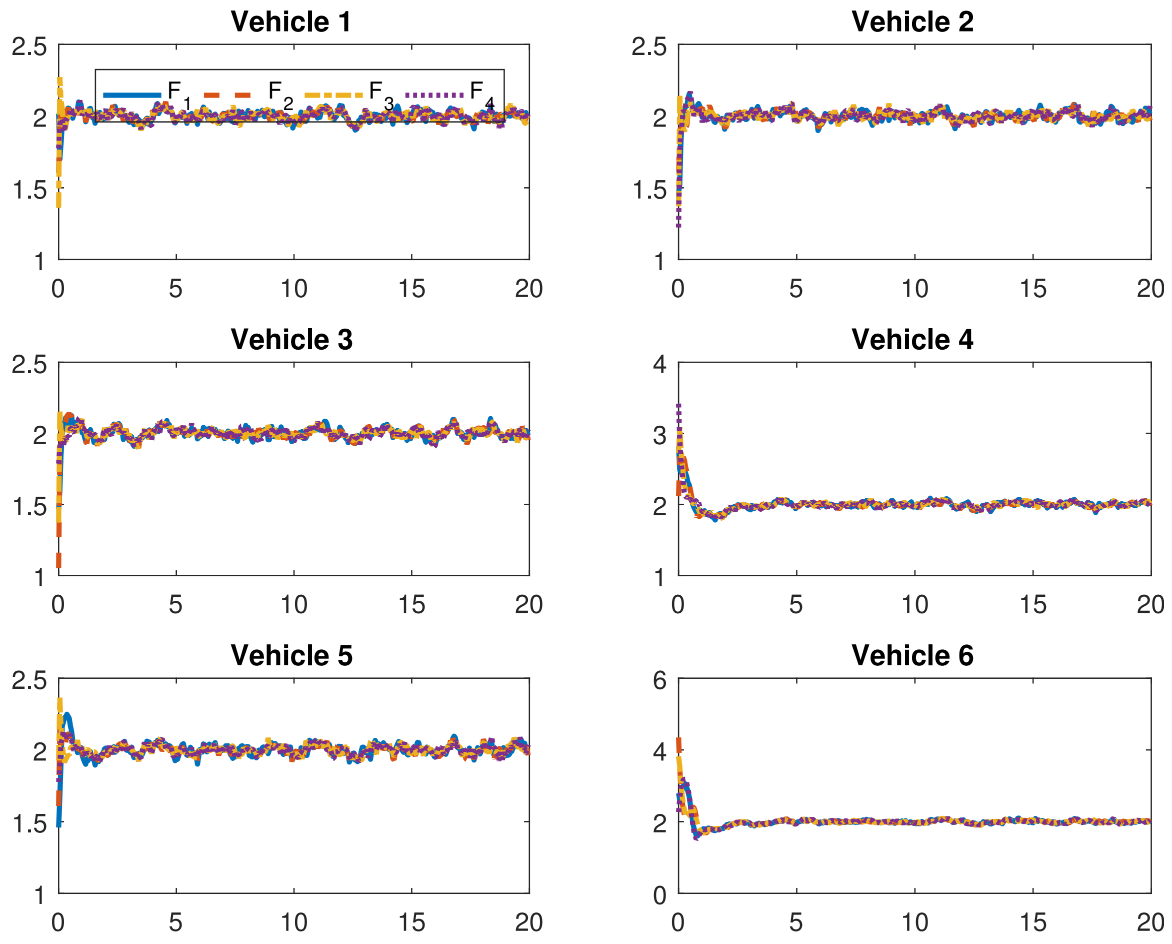


Figure 7.9. Case 2: Inputs of each vehicle of cooperative control and estimation.

CHAPTER 8

Summary, Conclusions and Future Work

8.1 Summary and Conclusions

The purpose of this dissertation is to develop a Model Predictive Control (also known as receding horizon control or moving horizon control) framework that is suitable for control of unmanned aerial vehicles. The developed scheme is then extended to perform cooperative control and estimation for multiple unmanned aerial vehicles.

The main idea behind Model Predictive Control is to solve successive sequences of open-loop finite horizon optimal control problems in real time. The control inputs obtained from this formulation is then applied to the system but only for a fraction of the horizon length. The one property that makes MPC so attractive is that it is able incorporate constraints explicitly into the problem formulation. Linear constrained MPC has been extensively studied while nonlinear constrained cases remains relatively barren.

Two main issues arise from the nonlinear MPC problem. The first is optimization for a general nonlinear problem over even a small horizon is very computationally intensive, if not impossible. The second is stability conditions for nonlinear MPC controllers are often too complicated or too restrictive, if not impossible to check against.

We developed a novel nonlinear MPC algorithm based on State Dependent Coefficient (SDC) formulation. SDC transforms a nonlinear system into a pseudo-linear form by employing a system matrix $\mathbf{A}$ that is dependent on the current state $\mathbf{x}$. It makes use of the existing mature linear MPC framework as well as exploit the inherent nonlinearities of the system itself. It was shown that by choosing an appropriate sampling interval, a sampled-data implementation of the proposed nonlinear MPC algorithm leads to a stable system. And for more specific constrained cases, a LMI condition is provided to guarantee feasibility and stability for constrained nonlinear systems.

The algorithm is then applied to an unmanned aerial vehicle, known as a quadcopter. Its linear counterpart is also implemented in simulation as control experiments. Three representative trajectories were tracked and the performances of linear MPC and nonlinear MPC are compared and discussed. It is shown from the simulation that nonlinear MPC proved to be superior in trajectory tracking and disturbance rejection.

In the next step, the control scheme is extended to control multiple vehicles in a distributed fashion to perform a shared task. Combined with an established consensus algorithm, we showed that as long as the underlying communication topology among the vehicles contains a spanning tree, the stability of the formation controller is guaranteed. The topologies constraint can be further relaxed to the union of the underlying graphs of infinite nonoverlapping time intervals having a spanning tree.

In Chapter 6, the topic of cooperative estimation is discussed to complement the aforementioned cooperative control scheme. Two methods for target location estimation are discussed and simulated.

In Chapter 7, the cooperative estimation and cooperative control are combined. The estimates obtained from cooperative estimation are used as external tracking commands for cooperative control. Cooperative control enables vehicles that are not in direct communication with the target to be able to stay close as long as they are in contact with as least one neighboring vehicles that are indirectly connected to the target.

8.2 Future Work

8.2.1 Experimental Validation

The next natural step of this research would be to implement the above algorithm on actual hardware. Currently, we are actively working on bringing this algorithm onto the ASL quadcopter platform. The biggest obstacle in this would be timely computation of control inputs. MATLAB is great but is not available for an outdoor environment. Transforming the MATLAB code into embedded C/C++ code would be necessary. A few functions that are available in MATLAB but not in C++ would need to be rewritten.

8.2.2 Optimal SDC Representation

The SDC representaion for quadcopter used in this dissertation is better than its linear counterpart, but is probably not optimal. A investigation could be done to find a better function $\mathbf{A}(\mathbf{x})$ that would require less control effort and better tracking and/or better transient response. In fact, it is not clear whether there exists a globally optimal SDC representation for the MPC method employed. That would need to be investigated as well.

8.2.3 MPC for Formation Consensus

The formation convergence in this dissertation is achieved through a consensus scheme. The benefits of this method is it lowers the computation load on each vehicle. But if a central workstation is available in certain scenarios, MPC can be used for reference trajectory generation for each vehicle.

CHAPTER 9

Appendix

9.1 Stability of Linear MPC

Stability of Unconstrained Linear MPC

Without loss of generality, the proof of stability will be given only for the regulator case. For trajectory tracking problems, we can always construct an error system with the new state being $\mathbf{e} = \mathbf{x} - \mathbf{x}^r$. With the proper feedforward input, the new system would be of the same form as the regulator. The cost function in Eq. (3.8) can be simplified respectively as:

$$\begin{aligned} J(\Delta\mathbf{x}_k, \Delta\mathbf{U}_k) &= \Delta\mathbf{U}_k^T \left(\mathbf{H}^T\bar{\mathbf{Q}}\mathbf{H} + \bar{\mathbf{R}} + \bar{\mathbf{B}}^T\mathbf{Q}_f\bar{\mathbf{B}}\right)\Delta\mathbf{U}_k \\ &+ 2\left(\left(\mathbf{F}\Delta\mathbf{x}_k\right)^T\bar{\mathbf{Q}}\mathbf{H} + \left(\mathbf{A}^N\Delta\mathbf{x}_k\right)^T\mathbf{Q}_f\bar{\mathbf{B}}\right)\Delta\mathbf{U}_k \\ &+ \left(\mathbf{F}\Delta\mathbf{x}_k\right)^T\bar{\mathbf{Q}}\left(\mathbf{F}\Delta\mathbf{x}_k\right) + \left(\mathbf{A}^N\Delta\mathbf{x}_k\right)^T\mathbf{Q}_f\left(\mathbf{A}^N\Delta\mathbf{x}_k\right) \end{aligned} \qquad (9.1)$$

Assuming there are no constraints present on the inputs or states, the optimal control $\Delta\mathbf{U}_k^*$ can be determined by setting $\dfrac{\partial J\left(\Delta\mathbf{x}_k, \Delta\mathbf{U}_k\right)}{\partial\Delta\mathbf{U}_k} = \mathbf{0}$, i.e.,

$$\Delta\mathbf{U}_k^* = -\left(\mathbf{H}^T\bar{\mathbf{Q}}\mathbf{H} + \bar{\mathbf{R}} + \bar{\mathbf{B}}^T\mathbf{Q}_f\bar{\mathbf{B}}\right)^{-1}\left[\mathbf{H}^T\bar{\mathbf{Q}}\mathbf{F} + \bar{\mathbf{B}}^T\mathbf{Q}_f\mathbf{A}^N\right]\Delta\mathbf{x}_k \qquad (9.2)$$

And the control input $\Delta\mathbf{u}_k^*$ can be extracted as follows.

$$\Delta\mathbf{u}_k^* = [\mathbf{I}_{m\times m}\ \mathbf{0}_{m\times m}\ \cdots\ \mathbf{0}_{m\times m}]\Delta\mathbf{U}_k^* \qquad (9.3)$$

Combining Eq. (9.2) and (9.3), $\Delta\mathbf{u}_k$ can be simplified.

$$\Delta\mathbf{u}_k^* = -\mathbf{K}_k\Delta\mathbf{x}_k \qquad (9.4)$$

where

$$\mathbf{K}_k = [\mathbf{I}_{m\times m}\ \mathbf{0}_{m\times m}\ \cdots\ \mathbf{0}_{m\times m}] \left(\mathbf{H}^T\bar{\mathbf{Q}}\mathbf{H} + \bar{\mathbf{R}} + \bar{\mathbf{B}}^T\mathbf{Q}_f\bar{\mathbf{B}}\right)^{-1} \left[\mathbf{H}^T\bar{\mathbf{Q}}\mathbf{F} + \bar{\mathbf{B}}^T\mathbf{Q}_f\mathbf{A}^N\right] \tag{9.5}$$

According to Ref. [2], if $\mathbf{Q}_f$ in Eq. (9.1) satisfies the following inequality:

$$\mathbf{Q}_f \geq \mathbf{Q} + \mathbf{K}_k^T\mathbf{R}\mathbf{K}_k + (\mathbf{A} - \mathbf{B}\mathbf{K_k})^T\mathbf{Q}_f(\mathbf{A} - \mathbf{B}\mathbf{K_k}) \tag{9.6}$$

for some $\mathbf{K}_k \in \Re^{m\times n}$, then the system of Eq. (3.3) driven by control $\Delta\mathbf{u}_k^*$ of Eq. (9.3) is stable.

Proof: The optimal cost $J(\Delta\mathbf{x}_k, \Delta\mathbf{U}_k^*)$ satisfies the following monotonicity condition:

$$J(\Delta\mathbf{x}_k, \Delta\mathbf{U}_k^{1*}, N+1) \leq J(\Delta\mathbf{x}_k, \Delta\mathbf{U}_k^{2*}, N) \tag{9.7}$$

where N denotes the prediction horizon and $\Delta\mathbf{U}_k^{1*}$, $\Delta\mathbf{U}_k^{2*}$ represent the optimal control derived from minimizing their respective cost functions. Imagine a new control input $\Delta\mathbf{U}_k^1$ by using $\Delta\mathbf{U}_k^{2*}$ up to time $k+N-1$ and $\Delta\mathbf{u}_{k+N} = -\mathbf{K}_k\Delta\mathbf{x}_{k+N}$, the cost for this control would be:

$$\begin{aligned} J(\Delta\mathbf{x}_k, \Delta\mathbf{U}_k^1, N+1) = &\sum_{i=k}^{k+N-1}(\Delta\mathbf{x}_i^T\mathbf{Q}\Delta\mathbf{x}_i + \Delta\mathbf{u}_i^T\mathbf{R}\Delta\mathbf{u}_i) + \Delta\mathbf{x}_{k+N}^T\mathbf{Q}\Delta\mathbf{x}_{k+N} \\ &+ \Delta\mathbf{x}_{k+N}\mathbf{K_k}^T\mathbf{R}\mathbf{K_k}\Delta\mathbf{x}_{k+N} \\ &+ \Delta\mathbf{x}_{k+N}^T(\mathbf{A} - \mathbf{B}\mathbf{K_k})^T\mathbf{Q}_f(\mathbf{A} - \mathbf{B}\mathbf{K_k})\Delta\mathbf{x}_{k+N} \end{aligned} \tag{9.8}$$

Since $J(\Delta\mathbf{x}_k, \Delta\mathbf{U}_k^{1*}, N+1)$ is optimal and denoting $\Delta J = J(\Delta\mathbf{x}_k, \Delta\mathbf{U}_k^{1*}, N+1) - J(\Delta\mathbf{x}_k, \Delta\mathbf{U}_k^{2*}, N)$, the following applies:

$$\begin{aligned} \Delta J &\leq J(\Delta\mathbf{x}_k, \Delta\mathbf{U}_k^1, N+1) - J(\Delta\mathbf{x}_k, \Delta\mathbf{U}_k^{2*}, N) \\ &\leq \Delta\mathbf{x}_{k+N}^T\mathbf{Q}\Delta\mathbf{x}_{k+N} + \Delta\mathbf{x}_{k+N}\mathbf{K}^T\mathbf{R}\mathbf{K}\Delta\mathbf{x}_{k+N} \\ &\quad + \Delta\mathbf{x}_{k+N}^T(\mathbf{A} - \mathbf{B}\mathbf{K_k})^T\mathbf{Q}_f(\mathbf{A} - \mathbf{B}\mathbf{K_k})\Delta\mathbf{x}_{k+N} - \Delta\mathbf{x}_{k+N}^T\mathbf{Q}_f\Delta\mathbf{x}_{k+N} \\ &\leq \Delta\mathbf{x}_{k+N}^T[\mathbf{Q} + \mathbf{K_k}^T\mathbf{R}\mathbf{K_k} + (\mathbf{A} - \mathbf{B}\mathbf{K_k})^T\mathbf{Q}_f(\mathbf{A} - \mathbf{B}\mathbf{K_k}) - \mathbf{Q}_f]\Delta\mathbf{x}_{k+N} \\ &\leq 0 \end{aligned} \tag{9.9}$$

thus proving the cost monotonicity condition. From the cost monotonicity condition, the following non-increasing sequence can be obtained:

$$\begin{aligned} J(\Delta\mathbf{x}_k, \Delta\mathbf{U}_k^*, N) &= \Delta\mathbf{x}_k^T\mathbf{Q}\Delta\mathbf{x}_k + \Delta\mathbf{u}_k^T\mathbf{R}\Delta\mathbf{u}_k + J(\Delta\mathbf{x}_{k+1}, \Delta\mathbf{U}_{k+1}^*, N-1) \\ &\geq \Delta\mathbf{x}_k^T\mathbf{Q}\Delta\mathbf{x}_k + \Delta\mathbf{u}_k^T\mathbf{R}\Delta\mathbf{u}_k + J(\Delta\mathbf{x}_{k+1}, \Delta\mathbf{U}_{k+1}^*, N) \\ &\geq J(\Delta\mathbf{x}_{k+1}, \Delta\mathbf{U}_{k+1}^*, N) \end{aligned} \tag{9.10}$$

Since a non-increasing sequence bounded from below converges to a constant, and $J(\Delta\mathbf{x}_k, \Delta\mathbf{U}_k^*, N) \geq 0$, and hence, $J(\Delta\mathbf{x}_k, \Delta\mathbf{U}_k^*, N) \to$ a nonnegative constant as $k \to \infty$. Thus,

$$\sum_{k=i}^{i+j} \left(\Delta\mathbf{x}_k^T\mathbf{Q}\Delta\mathbf{x}_k + \Delta\mathbf{u}_k^T\mathbf{R}\Delta\mathbf{u}_k\right) \to 0,\ j = 0, 1, 2, \cdots \tag{9.11}$$

which leads to the following equation:

$$\sum_{k=i}^{i+j} \Delta\mathbf{x}_k^T[(\mathbf{A}-\mathbf{B}\mathbf{K}_k)^j]^T(\mathbf{Q}+\mathbf{K}_k^T\mathbf{R}\mathbf{K}_k)[(\mathbf{A}-\mathbf{B}\mathbf{K}_k)^j]\Delta\mathbf{x}_k \to 0, \quad j = 0, 1, 2, \cdots \tag{9.12}$$

It is proved in Ref. [66] that if $(\mathbf{A}, \mathbf{C})$ is observable, then

$$\left((\mathbf{A} - \mathbf{B}\mathbf{K_k}), \begin{bmatrix} \sqrt{\mathbf{Q}} \\ \sqrt{\mathbf{R}}\mathbf{K}_k \end{bmatrix} \right)$$

is observable for any $\mathbf{K}_k$. Thus, the only solution that could guarantee this is the trivial solution $\Delta\mathbf{x}_k = \mathbf{0}$. Hence system (3.3), driven by control $\Delta\mathbf{u}_k^*$ of Eq. (9.3) by minimizing the quadratic cost (9.1), is stable.

Stability of Constrained Linear MPC

Again without loss of generality, we will only consider stability for the regulator case. To guarantee the same monotonicity condition as Eq. (9.7), it must be guaranteed that the system is feasible. The following Linear Matrix Inequalities (LMI) are constructed for this purpose. There must exist a set of $(\mathbf{Q}_f, \mathbf{Q}, \mathbf{R}, \mathbf{K}_k, \Delta\mathbf{U}_k)$ that satisfy the following conditions:

$$\gamma < \infty \tag{9.13}$$

$$\begin{bmatrix} \gamma - 2\Delta\mathbf{x}_k^T \left[\mathbf{F}\bar{\mathbf{Q}}\mathbf{H} + \left(\mathbf{A}^N\right)^T \mathbf{Q}_f\bar{\mathbf{B}}\right]\Delta\mathbf{U}_k & \Delta\mathbf{U}_k^T \\ \Delta\mathbf{U}_k & \left(\mathbf{H}^T\bar{\mathbf{Q}}\mathbf{H} + \bar{\mathbf{R}} + \bar{\mathbf{B}}^T\mathbf{Q}_f\bar{\mathbf{B}}\right)^{-1} \end{bmatrix} \geq \mathbf{0} \tag{9.14}$$

$$\mathbf{\Gamma}\Delta\mathbf{U}_k \leq \mathbf{\Upsilon} \tag{9.15}$$

$$\begin{bmatrix} \mathbf{Q}_f^{-1} & \mathbf{Q}_f^{-1}(\mathbf{A}-\mathbf{B}\mathbf{K_k})^T & \mathbf{Q}_f^{-1}\sqrt{\mathbf{Q}} & \mathbf{Q}_f^{-1}\mathbf{K_k}^T\sqrt{\mathbf{R}} \\ (\mathbf{A}-\mathbf{B}\mathbf{K_k})\mathbf{Q}_f^{-1} & \mathbf{Q}_f^{-1} & \mathbf{0}_{n\times n} & \mathbf{0}_{n\times n} \\ \sqrt{\mathbf{Q}}\mathbf{Q}_f^{-1} & \mathbf{0}_{n\times n} & \mathbf{I}_{n\times n} & \mathbf{0}_{n\times n} \\ \sqrt{\mathbf{R}}\mathbf{K}_k(\mathbf{Q}_f)^{-1} & \mathbf{0}_{n\times n} & \mathbf{0}_{n\times n} & \mathbf{I}_{n\times n} \end{bmatrix} \geq \mathbf{0} \tag{9.16}$$

$$\begin{bmatrix} \mathbf{I}_{m\times m} \\ -\mathbf{I}_{m\times m} \end{bmatrix} \mathbf{K}_k\left(\mathbf{A}^N\Delta\mathbf{x}_k + \bar{\mathbf{B}}\Delta\mathbf{U}_k\right) \leq \begin{bmatrix} (\mathbf{u}_{ub} - \mathbf{u}_T) \\ -(\mathbf{u}_{lb} - \mathbf{u}_T) \end{bmatrix} \tag{9.17}$$

$$\begin{bmatrix} \mathbf{C_z} \\ -\mathbf{C_z} \end{bmatrix} (\mathbf{A}-\mathbf{B}\mathbf{K}_k)\left(\mathbf{A}^N\Delta\mathbf{x}_k + \bar{\mathbf{B}}\Delta\mathbf{U}_k\right) \leq \begin{bmatrix} (\mathbf{z}_{ub} - \mathbf{C_z}\mathbf{x}_T) \\ -(\mathbf{z}_{lb} - \mathbf{C_z}\mathbf{x}_T) \end{bmatrix} \tag{9.18}$$

Since the constraints are constant, this condition only needs to be checked once at the beginning. After obtaining a suitable $(\mathbf{Q}_f, \mathbf{Q}, \mathbf{R})$ from above LMI, $\Delta\mathbf{U}_k^*$ is obtained by solving the following optimization problem:

$$\Delta\mathbf{U}_k^* = \arg\min_{\Delta\mathbf{U}_k} \gamma$$

subject to Eqs. (9.13), (9.14), (9.15), (9.16), (9.17) and (9.18).

Solving the above optimization problem, yields $\Delta \mathbf{u}_k^*$,

$$\Delta \mathbf{u}_k^* = [\mathbf{I}_{m\times m}\ \mathbf{0}_{m\times m}\ \cdots\ \mathbf{0}_{m\times m}]\Delta \mathbf{U}_k^* \tag{9.19}$$

The same proof of asymptotic stability from unconstrained case can be used here. Thus, system (3.3) driven by control (9.19) subject to constraints (3.15) is stable.

9.2 Stability of Nonlinear MPC

Stability of Unconstrained Sampled-Data NMPC based on SDC formulation

Assumptions:

- $(\mathbf{A_c}(\mathbf{x}),\ \mathbf{B_c}(\mathbf{x}))$ is point-wise controllable. Thus, $\forall\ \mathbf{x} \in \Re^n$, i.e., $\exists\ \mathbf{K_c}(\mathbf{x}) \in \Re^{m\times n}$ such that $(\mathbf{A_c}(\mathbf{x}) - \mathbf{B_c}(\mathbf{x})\mathbf{K_c}(\mathbf{x}))$ is point-wise Hurwitz.
- $\mathbf{K_c}(\mathbf{x})$ is obtained as a solution to the SDRE outlined in section 4.1. The control law is thus expressed as,

$$\mathbf{u_c} = -\mathbf{K_c}(\mathbf{x})(\mathbf{x} - \mathbf{x}^r) \tag{9.20}$$

- $\mathbf{x}^r(t)$ is an admissible reference state trajectory, i.e. $\mathbf{x}^r(t)$ satisfies the governing equations of motion as well as the state constraints.
- The reference control inputs $\mathbf{u}^r(t)$ obtained from Eq. (3.1) together with $\mathbf{x}^r(t)$, and $\dot{\mathbf{x}}^r(t)$ satisfy the control constraints discussed previously.

In this section we will first show that for an appropriate sample time Δt the ZOH control computed as

$$\mathbf{u}_c(k\Delta t) = -\mathbf{K_c}(\mathbf{x}(k\Delta t))\,(\mathbf{x}(k\Delta t) - \mathbf{x}^r(k\Delta t))$$

when applied to the nonlinear system in Eq. (3.1) results in bounded trajectory tracking errors, i.e. $\|\mathbf{x}(k\Delta t) - \mathbf{x}^r(k\Delta t)\| < \varepsilon$, $\varepsilon > 0$ for $k > k_N$, where $k_N \in \mathcal{Z}^+$. For the rest of the discussion, we will simply write $\mathbf{x}(k\Delta t)$ and $\mathbf{u}_c(k\Delta t)$ as $\mathbf{x}_k$ and $\mathbf{u}_k$.

For $t \in [k\Delta t,\ (k+1)\Delta t)$, the frozen in time SDC representation for the system is considered, thus the states evolve as,

$$\dot{\tilde{\mathbf{x}}} = \mathbf{A_c}(\hat{\mathbf{x}}_k)\tilde{\mathbf{x}} + \mathbf{B_c}(\hat{\mathbf{x}}_k)\mathbf{u}_k, \quad \tilde{\mathbf{x}}(k\Delta t) = \mathbf{x}_k, \quad t \in [k\Delta t,\ (k+1)\Delta t)] \tag{9.21}$$

At the next sampling interval $[(k+1)\Delta t,\ (k+2)\Delta t)$, $\tilde{\mathbf{x}}((k+1)\Delta t)$ is replaced with $\mathbf{x}_{k+1}$ measured from the original system Eq. (4.1), and the same process starts over again. To avoid potential ambiguities, the solution of differential Eq. (9.21) at the end of the interval, namely, the value of $\tilde{\mathbf{x}}$ as $t \to (k+1)\Delta t$, is denoted as $\tilde{\mathbf{x}}'_{k+1}$. Note, the control input $\mathbf{u}_c$ is constant during the interval $[k\Delta t, (k+1)\Delta t)]$. Under the assumptions stated earlier, the control for Eq.(9.21) will stabilize the original system Eq. (4.1) provided the control law given by Eq. (9.20) will achieve Uniformly Globally Asymptotically Stability (UGAS) for system (3.1). It should be noted here, this algorithm is merely a sampled-data implementation of Eq. (9.20). Ref. [68] proved the stability of sampled-data control based on SDC provided that $\mathbf{P}_k$ obtained from Eq. (4.4) converges to a constant matrix, in other words, $\lim_{k\to\infty} \mathbf{P}_k$ exists. However, the convergence of $\mathbf{P}_k$ is hard, if not impossible to guarantee. Here we offer an alternative proof with a different, albeit under restrictive assumptions, hoping to shed some light on this issue.

Without loss of generality, we would only prove the stability of the system for the regulator case, meaning $\mathbf{x}^r = \mathbf{0}$ at all time. Firstly, $\mathbf{x_k} \neq \mathbf{0}$ since $\mathbf{x_k} = \mathbf{0}$ dictates $\mathbf{u_k} = \mathbf{0}$, which makes it trivial.

With a constant input $\mathbf{u_k}$ during time interval $t \in [k\Delta t,\ (k+1)\Delta t]$, we obtain $\mathbf{x}_{k+1}$ from the following:

$$\mathbf{x}_{k+1} = \mathbf{x}_k + \int_{k\Delta t}^{(k+1)\Delta t} \big[\mathbf{f}(\mathbf{x}(\tau)) + \mathbf{g}(\mathbf{x}(\tau))\mathbf{u_k}\big] d\tau \tag{9.22}$$

and obtain $\tilde{\mathbf{x}}'_{k+1}$ from :

$$\tilde{\mathbf{x}}'_{k+1} = e^{\mathbf{A}(\mathbf{x}_k)\Delta t}\mathbf{x_k} + \int_{k\Delta t}^{(k+1)\Delta t} \big(e^{\mathbf{A}(\mathbf{x}_k)(s-\tau)}\mathbf{B}(\mathbf{x}_k)\mathbf{u}_k\big)\, d\tau \tag{9.23}$$

The control input $\mathbf{u}_k$ is designed based on (9.23) such that:

$$\|\tilde{\mathbf{x}}'_{k+1}\| < \|\mathbf{x}_k\|$$

It should be noted that such an input $\mathbf{u}_k$ will always exist since the system in Eq. (4.1) is assumed point wise controllable. The local truncation error (between the true nonlinear system and the discretized pseudo-linear system subject to the ZOH control law) at the end of the time interval is defined as:

$$\mathbf{e}_{k+1} = \tilde{\mathbf{x}}'_{k+1} - \mathbf{x}_{k+1} \tag{9.24}$$

Define a term $\lambda(\mathbf{x}_k, \Delta t)$ as:

$$\lambda(\mathbf{x}_k, \Delta t) = \frac{\|\mathbf{e}_{k+1}\|}{\|\tilde{\mathbf{x}}'_{k+1}\| - \|\mathbf{x}_k\|}$$

By taking the limit of $\lambda(\cdot, \Delta t)$ when $\Delta t \to 0$ we obtain:

$$\begin{aligned}
\lim_{\Delta t \to 0} \lambda(\mathbf{x}_k, \Delta t) &= \lim_{\Delta t \to 0} \frac{\|\tilde{\mathbf{x}}'_{k+1} - \mathbf{x}_{k+1}\|}{\|\tilde{\mathbf{x}}'_{k+1}\| - \|\mathbf{x}_k\|} \\
&= \lim_{\Delta t \to 0} \frac{\|\tilde{\mathbf{x}}'_{k+1} - \mathbf{x}_{k+1}\|}{\Delta t} \frac{1}{\frac{\|\tilde{\mathbf{x}}'_{k+1}\| - \|\mathbf{x}_k\|}{\Delta t}}
\end{aligned}$$

Let

$$\lambda_F(\mathbf{x}_k, \Delta t) = \frac{\|\tilde{\mathbf{x}}'_{k+1} - \mathbf{x}_{k+1}\|}{\Delta t}, \quad \lambda_G(\mathbf{x}_k, \Delta t) = \frac{\|\tilde{\mathbf{x}}'_{k+1}\| - \|\mathbf{x}_k\|}{\Delta t}$$

One can then obtain:

$$\begin{aligned}
\lim_{\Delta t \to 0} \lambda_F(\mathbf{x}_k, \Delta t) &= \lim_{\Delta t \to 0} \frac{\|\tilde{\mathbf{x}}'_{k+1} - \mathbf{x}_{k+1}\|}{\Delta t} \\
&= \lim_{\Delta t \to 0} \frac{\|(\tilde{\mathbf{x}}'_{k+1} - \mathbf{x}_k) - (\mathbf{x}_{k+1} - \mathbf{x}_k)\|}{\Delta t} \to 0 \\
\lim_{\Delta t \to 0} \lambda_G(\mathbf{x}_k, \Delta t) &= \lim_{\Delta t \to 0} \frac{\|\tilde{\mathbf{x}}'_{k+1}\| - \|\mathbf{x}_k\|}{\Delta t} = \frac{\mathbf{x}_k^T \big(\mathbf{f}(\mathbf{x}_k) + \mathbf{g}(\mathbf{x}_k)\mathbf{u}_k\big)}{\|\mathbf{x}_k\|}
\end{aligned}$$

Also we can deduct $\mathbf{x}_k \neq \mathbf{0} \Rightarrow \mathbf{f}(\mathbf{x}_k) + \mathbf{g}(\mathbf{x}_k)\mathbf{u}_k \neq \mathbf{0}$. A stabilizing $\mathbf{u}_k$ dictates $\mathbf{x}_k^T \big(\mathbf{f}(\mathbf{x}_k) + \mathbf{g}(\mathbf{x}_k)\mathbf{u}_k\big) \neq 0$. From all of this it can be concluded that:

$$\lim_{\Delta t \to 0} \lambda(x_k, \Delta t) = \frac{\lim_{\Delta t \to 0} \lambda_F(\mathbf{x}_k, \Delta t)}{\lim_{\Delta t \to 0} \lambda_G(\mathbf{x}_k, \Delta t)} \to \frac{0}{\frac{\mathbf{x}_k^T \left(\mathbf{f}(\mathbf{x}_k) + \mathbf{g}(\mathbf{x}_k)\mathbf{u}_k\right)}{\|\mathbf{x}_k\|}} = 0$$

Thus, $\exists\ \Delta t \in (0, \infty)$ that ensures the corresponding $\lambda(\mathbf{x}_k, \Delta t) \in (0, 1)$. It leads to:

$$\|\mathbf{e}_{k+1}\| = \lambda(\mathbf{x}_k, \Delta t) \left| \|\tilde{\mathbf{x}}'_{k+1}\| - \|\mathbf{x}_k\| \right| < \|\mathbf{x}_k\| - \|\tilde{\mathbf{x}}'_{k+1}\|$$

Combined with Eq. (9.24) one can then easily deduce that $\|\mathbf{x}_{k+1}\| < \|\mathbf{x}_k\|$ which means that the system would progress successively as, $\|\mathbf{x}_{i+1}\| < \|\mathbf{x}_i\| \quad i \to 0, 1, 2, \ldots$. Therefore, one can conclude that $\exists\, \Delta t \in [0, \infty)$ that guarantees a control law designed based on Eq. (9.21) will stabilize the original system described by Eq. (3.1). Thus, it is shown that the sampled-data ZOH controller based on the pseudo-linear representation in Eq. (4.1) also ensures that the states of the true nonlinear system stay close to the states evolving based on Eq. (4.1). The next section focuses on the constrained sampled-data NMPC based on the SDC system.

Stability of Constrained Sampled-Data NMPC based on SDC System

The only difference between the constrained sampled-data NMPC and constrained linear MPC is that the feasibility of the system needs to be checked at the start of every sampling interval. For simplicity, the brackets will be replaced as subscript to indicate its dependence on $\mathbf{x}_k$, $(\cdot)(\mathbf{x}_k)$ would be written as $(\cdot)_k$. Similarly for each $\mathbf{x}_k$, there must exist a set of $(\mathbf{Q}_{fk}, \mathbf{Q}_k, \mathbf{R}_k, \mathbf{K}_k, \Delta\mathbf{U}_k)$ that satisfy the following conditions:

$$\gamma_k < \infty \tag{9.25}$$

$$\begin{bmatrix} \gamma_k - 2\left[(\mathbf{F}_k\mathbf{x}_k)^T\bar{\mathbf{Q}}_{Nk}\mathbf{H}_\mathbf{k} + \left(\mathbf{A}_k^N\mathbf{x}_k\right)^T\mathbf{Q}_{fk}\bar{\mathbf{B}}_\mathbf{k}\right]\Delta\mathbf{U}_k & \Delta\mathbf{U}_k^T \\ \Delta\mathbf{U}_k & \left(\mathbf{H}_k^T\bar{\mathbf{Q}}_{Nk}\mathbf{H}_k + \bar{\mathbf{R}}_{Nk} + \bar{\mathbf{B}}_k^T\mathbf{Q}_{fk}\bar{\mathbf{B}}_k\right)^{-1} \end{bmatrix} \geq \mathbf{0} \tag{9.26}$$

$$\mathbf{\Gamma}(\mathbf{x_k})\Delta\mathbf{U}_k \leq \mathbf{\Upsilon}(\mathbf{x_k}) \tag{9.27}$$

$$\begin{bmatrix} \mathbf{Q}_{fk}^{-1} & \mathbf{Q}_{fk}^{-1}(\mathbf{A}_k - \mathbf{B}_k\mathbf{K}_k)^T & \mathbf{Q}_{fk}^{-1}\sqrt{\mathbf{Q}_k} & (\mathbf{Q}_{fk})^{-1}\mathbf{K}_k^T\sqrt{\mathbf{R}_k} \\ (\mathbf{A}_k - \mathbf{B_k}\mathbf{K_k})(\mathbf{Q}_{fk})^{-1} & (\mathbf{Q}_{fk})^{-1} & \mathbf{0}_{n\times n} & \mathbf{0}_{n\times n} \\ \sqrt{\mathbf{Q}_k}\mathbf{Q}_{fk}^{-1} & \mathbf{0}_{n\times n} & \mathbf{I}_{n\times n} & \mathbf{0}_{n\times n} \\ \sqrt{\mathbf{R}_k}\mathbf{K}_k\mathbf{Q}_{fk}^{-1} & \mathbf{0}_{n\times n} & \mathbf{0}_{n\times n} & \mathbf{I}_{n\times n} \end{bmatrix} \geq \mathbf{0} \tag{9.28}$$

$$\begin{bmatrix} \mathbf{I}_{m\times m} \\ -\mathbf{I}_{m\times m} \end{bmatrix} \mathbf{K}_k(\mathbf{A}_k^N \mathbf{x}_k + \bar{\mathbf{B}}_k \Delta \mathbf{U}_k) \leq \begin{bmatrix} (\mathbf{u}_{ub} - \mathbf{u}_T) \\ -(\mathbf{u}_{lb} - \mathbf{u}_T) \end{bmatrix} \tag{9.29}$$

$$\begin{bmatrix} \mathbf{C}_{zk} \\ -\mathbf{C}_{zk} \end{bmatrix} (\mathbf{A}_k - \mathbf{B}_k \mathbf{K}_k)[\mathbf{A}_k^N \mathbf{x}_k + \bar{\mathbf{B}}_k(\mathbf{U}_T + \Delta \mathbf{U}_k)] \leq \begin{bmatrix} \mathbf{z}_{ub} \\ -\mathbf{z}_{lb} \end{bmatrix} \tag{9.30}$$

After obtaining a suitable $(\mathbf{Q}_{fk}, \mathbf{Q}_k, \mathbf{R}_k)$ from above LMI, $\Delta \mathbf{U}_k^*$ is obtained by doing the following optimization:

$$\Delta \mathbf{U}_k^* = \arg \min_{\Delta \mathbf{U}_k} \gamma_k$$

subject to the conditions specified in Eqs. (9.25), (9.26), (9.27), (9.28), (9.29), and (9.30). Upon solving the optimization problem, $\Delta \mathbf{u}_k^*$ can be obtained by

$$\Delta \mathbf{u}_k^* = [\mathbf{I}_{m\times m}\ \mathbf{0}_{m\times m}\ \cdots\ \mathbf{0}_{m\times m}] \Delta \mathbf{U}_k^* \tag{9.31}$$

9.3 Derivation for Solving the Trilateration based Estimation

$$\mathbf{q}_k = \begin{bmatrix} d_i^2 - (x_k^i)^2 - (y_k^i)^2 - (z_k^i)^2 - [d_{j_1}^2 - (x_k^{j_1})^2 - (y_k^{j_1})^2 - (z_k^{j_1})^2] \\ d_{j_1}^2 - (x_k^{j_1})^2 - (y_k^{j_1})^2 - (z_k^{j_1})^2 - [d_{j_2}^2 - (x_k^{j_2})^2 - (y_k^{j_2})^2 - (z_k^{j_2})^2] \\ d_{j_2}^2 - (x_k^{j_2})^2 - (y_k^{j_2})^2 - (z_k^{j_2})^2 - [d_{j_3}^2 - (x_k^{j_3})^2 - (y_k^{j_3})^2 - (z_k^{j_3})^2] \end{bmatrix}$$

Using Linear Least Squares, the location of the target $\mathbf{p}_t$ can be expressed as:

$$[x_t \ y_t \ z_t]^T = (\mathbf{H}_k^T \mathbf{H}_k)^{-1} \mathbf{H}_k^T \mathbf{q}_k$$

This method can be used to provide a good initial estimation for nonlinear least squares algorithm.

9.4 Simulation Parameters for Linear/Nonlinear MPC

The following parameters were used for all the simulations. The parameters for the control law synthesis were the same regardless of the reference trajectories and the linear/nonlinear control themes.

- Mass: $m = 0.8$ (kg).
- Moment of inertia: $\mathbf{J} = \begin{bmatrix} 0.0224 & & \\ & 0.0224 & \\ & & 0.0436 \end{bmatrix} (kg \cdot m^2)$.
- Distance from the center of the rotor to CG of the quadcopter: $L = 0.165$ (m).
- Ratio of rotor angular momentum to rotor lift: $c = 0.002167$ (m).
- Gravitational acceleration: $g = 9.8$ (m/s^2).
- Rotor actuator time constant: $\lambda_F = 50$.
- Control horizon for linear and nonlinear MPC: $N = 25$.
- Prediction horizon for linear and nonlinear MPC: $M = 25$.
- Control horizon for nonlinear MPC used in cooperative control: $N = 5$.
- Sample time: $\Delta t = 0.05$ (s).
- State weighting matrix: $\mathbf{Q} = \text{diag}\,[1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1]$.
- Input weighting matrix: $\mathbf{R} = \text{diag}\,[1, 1, 1, 1]$.
- Input constraint coefficient: $\kappa_i = 1$ $(i = 1, 2, 3, 4)$.

REFERENCES

[1] W. Kwon and A. Pearson, "A modified quadratic cost problem and feedback stabilization of a linear system," *IEEE Transactions on Automatic Control*, vol. 22, no. 5, pp. 838–842, Oct 1977.

[2] W. H. Kwon and S. H. Han, *Receding horizon control: model predictive control for state models*. Springer Science & Business Media, 2006.

[3] W. H. Kwon and K. B. Kim, "On stabilizing receding horizon controls for linear continuous time-invariant systems," *IEEE Transactions on Automatic Control*, vol. 45, no. 7, pp. 1329–1334, Jul 2000.

[4] J. W. Lee, W. H. Kwon, and J. Choi, "On stability of constrained receding horizon control with finite terminal weighting matrix," in *Control Conference (ECC), 1997 European*, July 1997, pp. 313–318.

[5] S. S. Keerthi and E. G. Gilbert, "Optimal infinite-horizon feedback laws for a general class of constrained discrete-time systems: Stability and moving-horizon approximations," *Journal of Optimization Theory and Applications*, vol. 57, no. 2, pp. 265–293, 1988.

[6] H. Michalska and D. Q. Mayne, "Robust receding horizon control of constrained nonlinear systems," *IEEE Transactions on Automatic Control*, vol. 38, no. 11, pp. 1623–1633, Nov 1993.

[7] T. Parisini and R. Zoppoli, "A receding-horizon regulator for nonlinear systems and a neural approximation," *Automatica*, vol. 31, no. 10, pp. 1443 – 1451, 1995.

[8] H. Chen and F. Allower, “A quasi-infinite horizon nonlinear model predictive control scheme with guaranteed stability,” *Automatica*, vol. 34, no. 10, pp. 1205 – 1217, 1998.

[9] L. Magni and R. Sepulchre, “Stability margins of nonlinear receding-horizon control via inverse optimality,” *Systems and Control Letters*, vol. 32, no. 4, pp. 241 – 245, 1997.

[10] G. D. Nicolao, L. Magni, and R. Scattolini, “Stabilizing receding horizon control of nonlinear time varying systems,” *IEEE Transactions on Automatic Control*, vol. 43, no. 7, pp. 1030–1036, Jul 1998.

[11] J. A. Primbs, “Nonlinear optimal control: a receding horizon approach,” Ph.D. dissertation, California Institute of Technology, 1999.

[12] J. A. Primbs, V. Nevistic, and J. C. Doyle, “A receding horizon generalization of pointwise min-norm controllers,” *IEEE Transactions on Automatic Control*, vol. 45, no. 5, pp. 898–909, May 2000.

[13] A. Jadbabaie, “Receding horizon control of nonlinear systems: A control lyapunov function appoach,” Ph.D. dissertation, California Institute of Technology, 2000.

[14] L. E. Parker, *Current State of the Art in Distributed Autonomous Mobile Robotics.* Tokyo: Springer Japan, 2000, pp. 3–12. [Online]. Available: http://dx.doi.org/10.1007/978-4-431-67919-6_1

[15] R. M. Murray, “Recent research in cooperative control of multivehicle systems,” *Journal of Dynamic Systems, Measurement, and Control*, vol. 129, no. 5, pp. 571–583, 2007.

[16] R. Olfati-Saber and R. M. Murray, “Consensus problems in networks of agents with switching topology and time-delays,” *IEEE Transactions on Automatic Control*, vol. 49, no. 9, pp. 1520–1533, Sept 2004.

[17] J. A. Fax and R. M. Murray, “Information flow and cooperative control of vehicle formations,” *IEEE Transactions on Automatic Control*, vol. 49, no. 9, pp. 1465–1476, Sept 2004.

[18] A. Jadbabaie, J. Lin, and A. S. Morse, “Coordination of groups of mobile autonomous agents using nearest neighbor rules,” *IEEE Transactions on Automatic Control*, vol. 48, no. 6, pp. 988–1001, June 2003.

[19] W. Ren and R. W. Beard, “Consensus seeking in multiagent systems under dynamically changing interaction topologies,” *IEEE Transactions on automatic control*, vol. 50, no. 5, pp. 655–661, 2005.

[20] L. Moreau, “Stability of multiagent systems with time-dependent communication links,” *IEEE Transactions on Automatic Control*, vol. 50, no. 2, pp. 169–182, Feb 2005.

[21] W. Ren, R. W. Beard, and E. M. Atkins, “A survey of consensus problems in multi-agent coordination,” in *Proceedings of the 2005, American Control Conference, 2005.*, June 2005, pp. 1859–1864 vol. 3.

[22] ——, “Information consensus in multivehicle cooperative control,” *IEEE Control Systems*, vol. 27, no. 2, pp. 71–82, April 2007.

[23] W. B. Dunbar and R. M. Murray, “Model predictive control of coordinated multi-vehicle formations,” in *Decision and Control, 2002, Proceedings of the 41st IEEE Conference on*, vol. 4, Dec 2002, pp. 4631–4636.

[24] S. M. LaValle, *Planning algorithms.* Cambridge university press, 2006.

[25] R. O. Saber, W. B. Dunbar, and R. M. Murray, “Cooperative control of multi-vehicle systems using cost graphs and optimization,” in *American Control Conference, 2003. Proceedings of the 2003*, vol. 3, June 2003, pp. 2217–2222.

[26] W. B. Dunbar and R. M. Murray, “Distributed receding horizon control for multi-vehicle formation stabilization,” *Automatica*, vol. 42, no. 4, pp. 549 –

558, 2006. [Online]. Available: http://www.sciencedirect.com/science/article/pii/S0005109806000136

[27] F. Borrelli, T. Keviczky, and G. J. Balas, "Collision-free uav formation flight using decentralized optimization and invariant sets," in *Decision and Control, 2004. CDC. 43rd IEEE Conference on*, vol. 1, Dec 2004, pp. 1099–1104.

[28] N. E. Leonard and E. Fiorelli, "Virtual leaders, artificial potentials and coordinated control of groups," in *Decision and Control, 2001. Proceedings of the 40th IEEE Conference on*, vol. 3, 2001, pp. 2968–2973 vol.3.

[29] P. Ogren, E. Fiorelli, and N. E. Leonard, "Cooperative control of mobile sensor networks:adaptive gradient climbing in a distributed environment," *IEEE Transactions on Automatic Control*, vol. 49, no. 8, pp. 1292–1302, Aug 2004.

[30] C.-Y. Chong, F. Zhao, S. Mori, and S. Kumar, "Distributed tracking in wireless ad hoc sensor networks," in *Information Fusion, 2003. Proceedings of the Sixth International Conference of*, vol. 1, July 2003, pp. 431–438.

[31] F. Zhao, J. Shin, and J. Reich, "Information-driven dynamic sensor collaboration," *IEEE Signal Processing Magazine*, vol. 19, no. 2, pp. 61–72, Mar 2002.

[32] T. Hatanaka, M. Fujita, and F. Bullo, "Vision-based cooperative estimation via multi-agent optimization," in *49th IEEE Conference on Decision and Control (CDC)*, Dec 2010, pp. 2492–2497.

[33] T. Hatanaka and M. Fujita, "Cooperative estimation of averaged 3-d moving target poses via networked visual motion observer," *IEEE Transactions on Automatic Control*, vol. 58, no. 3, pp. 623–638, March 2013.

[34] T. H. Chung, V. Gupta, J. W. Burdick, and R. M. Murray, "On a decentralized active sensing strategy using mobile sensor platforms in a network," in *Decision and Control, 2004. CDC. 43rd IEEE Conference on*, vol. 2, Dec 2004, pp. 1914–1919 Vol.2.

[35] R. Olfati-Saber, "Distributed kalman filtering for sensor networks," in *Decision and Control, 2007 46th IEEE Conference on*, Dec 2007, pp. 5492–5498.

[36] F. Zhang and N. E. Leonard, "Cooperative filters and control for cooperative exploration," *IEEE Transactions on Automatic Control*, vol. 55, no. 3, pp. 650–663, March 2010.

[37] M. Stachura and E. W. Frew, "Cooperative target localization with a communication-aware unmanned aircraft system," *Journal of Guidance, Control, and Dynamics*, vol. 34, no. 5, pp. 1352–1362, 2011.

[38] P. Scerri, T. Von Gonten, G. Fudge, S. Owens, and K. Sycara, "Transitioning multiagent technology to uav applications," in *Proceedings of the 7th International Joint Conference on Autonomous Agents and Multiagent Systems: Industrial Track*, ser. AAMAS '08. Richland, SC: International Foundation for Autonomous Agents and Multiagent Systems, 2008, pp. 89–96. [Online]. Available: http://dl.acm.org/citation.cfm?id=1402795.1402812

[39] K. Subbarao, C. Tule, and P. Ru, "Nonlinear model predictive control applied to trajectory tracking for unmanned aerial vehicles," in *AIAA Atmospheric Flight Mechanics Conference*, no. AIAA 2015-2857, Dallas, TX, June 2015.

[40] P. Ru and K. Subbarao, "Nonlinear model predictive control for unmanned aerial vehicles," Journal of Aerospace Engineering, May 2017, to be published.

[41] ——, "Cooperative control of unmanned aerial vehicles based on nonlinear model predictive control," in *AIAA Guidance, Navigation, and Control Conference, AIAA Science and Technology Forum and Exposition 2017.*, Grapevine, TX, USA, January 2017.

[42] ——, "Cooperative estimation of moving target position using unmanned aerial vehicles," in *AIAA Information Systems-AIAA Infotech @ Aerospace, AIAA Sci-*

ence and Technology Forum and Exposition 2017, Grapevine, TX, USA, January 2017.

[43] H. K. Khalil and J. Grizzle, *Nonlinear systems.* Prentice hall New Jersey, 1996, vol. 3.

[44] R. M. Murray, Z. Li, S. S. Sastry, and S. S. Sastry, *A mathematical introduction to robotic manipulation.* CRC press, 1994.

[45] S. P. Boyd, L. El Ghaoui, E. Feron, and V. Balakrishnan, *Linear matrix inequalities in system and control theory.* SIAM, 1994, vol. 15.

[46] L. Vandenberghe and S. Boyd, "Semidefinite programming," *SIAM review*, vol. 38, no. 1, pp. 49–95, 1996.

[47] R. Materna, "Highlights from aerospace industry report 2012: Facts, figures and outlook for the aviation and aerospace manufacturing industry," p. 258, 2012.

[48] F. Augugliaro, S. Lupashin, M. Hamer, C. Male, M. Hehn, M. W. Mueller, J. S. Willmann, F. Gramazio, M. Kohler, and R. D'Andrea, "The flight assembled architecture installation: Cooperative construction with flying machines," *IEEE Control Systems*, vol. 34, no. 4, pp. 46–64, August 2014.

[49] S. Lupashin, M. Hehn, M. W. Mueller, A. P. Schoellig, M. Sherback, and R. D. Andrea, "A platform for aerial robotics research and demonstration: the flying machine arena," *Mechatronics*, pp. 41–54, 2014.

[50] Q. Lindsey, D. Mellinger, and V. Kumar, "Construction with quadrotor teams," *Autonomous Robots*, vol. 33, pp. 323–336, 2012.

[51] E. deVries and K. Subbarao, "Cooperative control of swarms of unmanned aerial vehicles," in *49th AIAA Aerospace Sciences Meeting and Conference*, no. AIAA 2011-78, Orlando, FL, January 2011.

[52] A. Das, K. Subbarao, and F. L. Lewis, "Dynamic inversion with zero-dynamics stabilization for quadrotor control," *IET Control Theory & Applications (for-*

merly IEE Proceedings Control Theory & Applications), vol. 3, no. 3, pp. 303–314, March 2009.

[53] E. deVries and K. Subbarao, “Backstepping based nested multi-loop control laws for a quadrotor,” in *11th International Conference on Control, Automation, Robotics and Vision*, Singapore, December 2010.

[54] Z. Zuo and P. Ru, “Augmented l1 adaptive tracking control of quad-rotor unmanned aircrafts,” *IEEE Transactions on Aerospace and Electronic Systems*, vol. 50, no. 4, pp. 3090–3101, October 2014.

[55] A. P. Schoellig, F. L. Mueller, and R. D’Andrea, “Optimization-based iterative learning for precise quadrocopter trajectory tracking,” *Autonomous Robots*, vol. 33, no. 1, pp. 103–127, 2012.

[56] A. Das, F. L. Lewis, and K. Subbarao, “Backstepping approach for controlling a quadrotor using neural networks,” *Journal of Intelligent & Robotic Systems, Springer*, vol. 56, pp. 127–151, September 2009.

[57] Z. Zuo, “Trajectory tracking control design with command-filtered compensation for a quadrotor,” *IET Control Theory Applications*, vol. 4, no. 11, pp. 2343–2355, November 2010.

[58] B. N. Pamadi, *Performance, Stability, Dynamics, and Control of Airplanes.* AIAA, 2004.

[59] S. Lupashin, A. Schöllig, M. Sherback, and R. D’Andrea, “A simple learning strategy for high-speed quadrocopter multi-flips,” in *Robotics and Automation (ICRA), 2010 IEEE International Conference on.* IEEE, 2010, pp. 1642–1648.

[60] O. Härkegård, “Dynamic control allocation using constrained quadratic programming,” *Journal of Guidance, Control, and Dynamics*, vol. 27, no. 6, pp. 1028–1034, 2004.

[61] T. Cimen, “State-dependent riccati equation (sdre) control: a survey,” in *Proceedings of the 17th World Congress of the International Federation of Automatic Control (IFAC)*, vol. 17, no. 1, July, 2008, pp. 3761–3775.

[62] A. Bogdanov, M. Carlsson, G. Harvey, J. Hunt, D. Kieburtz, R. V. D. Merwe, and E. Wan, “State-dependent riccati equation control of a small unmanned helicopter,” in *In Proceedings of the AIAA Guidance Navigation and Control Conference*, 2003.

[63] A. S. Dutka, A. W. Ordys, and M. J. Grimble, “Optimized discrete-time state dependent riccati equation regulator,” in *American Control Conference, 2005. Proceedings of the 2005*, vol. 4, June 2005, pp. 2293 – 2298.

[64] A. A. Bogdanov, , E. A. Wan, M. Carlsson, Y. Zhang, R. Kieburtz, and A. Baptista, “Model predictive neural control of a high-fidelity helicopter model,” in *In Proceedings of the AIAA Guidance Navigation and Control Conference*, 2001, pp. 6–9.

[65] N. M. Singh, J. Dubey, and G. Laddha, “Control of pendulum on a cart with state dependent riccati equations,” in *Proceedings of World Academy of Science, Engineering and Technology*, vol. 31, 2008.

[66] F. L. Lewis and V. L. Syrmos, *Optimal Control.* Wiley-Interscience, 1995.

[67] H. Voos, “Nonlinear control of a quadrotor micro-uav using feedback-linearization,” in *2009 IEEE International Conference on Mechatronics*, April 2009, pp. 1–6.

[68] K. Hammett, “Control of nonlinear systems via state feedback state-dependent riccati equation techniques,” Ph.D. dissertation, AFIT, Wright Patterson AFB, OH, 1997.

BIOGRAPHICAL STATEMENT

Pengkai Ru obtained his bachelor's degree in Mechanical Engineering from Beihang University in Beijing, China. After graduation, he began his doctoral studies in Aerospace Engineering at the University of Texas at Arlington in 2012. His main research focus is nonlinear model predictive control applied to unmanned aerial vehicles.