TRAJECTORY GENERATION AND CONSTRAINED CONTROL OF QUADROTORS

by

CARLOS ALBERTO TULE

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

MASTER OF SCIENCE IN AEROSPACE ENGINEERING

THE UNIVERSITY OF TEXAS AT ARLINGTON

December 2012

Copyright © by CARLOS ALBERTO TULE 2012

All Rights Reserved

To my parents, Laura Olivares and Carlos Tule.

ACKNOWLEDGEMENTS

I would like to express my gratitude to my research supervisor Dr. Kamesh Subbarao for his guidance, support and advice during the course of my Master's degree. I would also like to thank Dr. Ben Harris and Dr. Panayiotis Shiakolas for taking time out of their busy schedule to be part of my defense committee.

In addition, I want to thank my fellow members of the Aerospace Systems Lab for their support and ideas for my thesis.

Finally, I would also like to thank my family for all their endless support they have provided me until now. It is thanks to them and their motivation that I was able to come to a different country to pursue a master's degree.

November 27, 2012

ABSTRACT

TRAJECTORY GENERATION AND CONSTRAINED CONTROL
OF QUADROTORS

CARLOS ALBERTO TULE, M.S.

The University of Texas at Arlington, 2012

Supervising Professor: Kamesh Subbarao

Unmanned Aerial Systems, although still in early development, are expected to grow in both the military and civil sectors. As part of the UAV sector, the Quadrotor helicopter platform has been receiving a lot of interest from various academic and research institutions because of their simplistic design and low cost to manufacture, yet remaining a challenging platform to control.

Four different controllers were derived for the trajectory generation and constrained control of a quadrotor platform. The first approach involves the linear version of the Model Predictive Control (MPC) algorithm to solve the state constrained optimization problem. The second approach uses the State Dependent Coefficient (SDC) form to capture the system non linearities into a pseudo-linear system matrix, which is used to derive the State Dependent Riccati Equation (SDRE) based optimal control. For the third approach, the SDC form is exploited for obtaining a nonlinear equivalent of the model predictive control. Lastly, a combination of the nonlinear MPC and SDRE optimal control algorithms is used to explore the feasibility of a near real-time nonlinear optimization technique.

TABLE OF CONTENTS

LIST OF ILLUSTRATIONS

LIST OF TABLES

CHAPTER 1

INTRODUCTION

1.1 The Quadrotor Helicopter platform

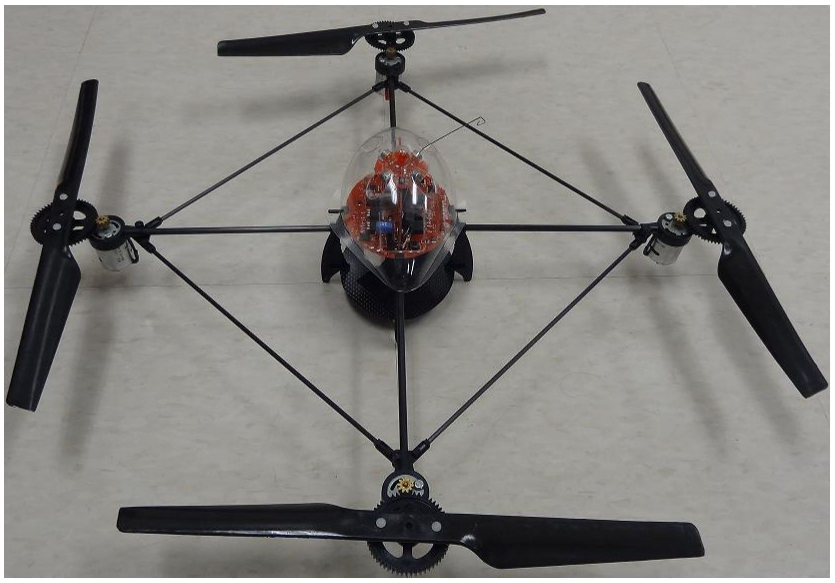


Figure 1.1. Draganflyer V Quadrotor.

A Quadrotor helicopter platform (often just called Quadrotor), is an under-actuated helicopter with two pairs of rotors in cross configuration capable of spinning at different angular velocities in order to achieve motion. Rotor pair (1,3) spins in one direction while the pair (2,4) spins in the opposite (see Figure 1.2). The different motions the quadrotor can perform are

- Vertical motion: Achieved by simultaneous change in rotor speed
- Roll motion: Achieved by imbalance in the rotor speed of pair (2,4).
- Pitch motion: Achieved by imbalance in the rotor speed of pair (1,3).

- Yaw motion: Achieved by rotor speed imbalance between all rotors.

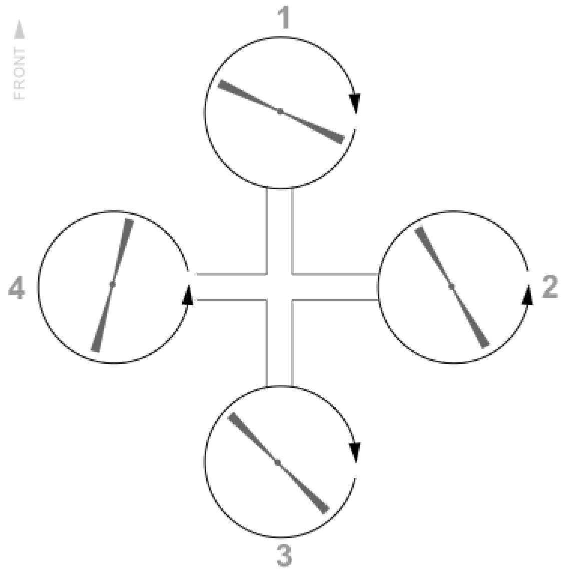


Figure 1.2. Diagram of a Quadrotor top view, image courtesy of https://pixhawk.ethz.ch/.

Quadrotors 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 research and development. They also possess many advantages over standard helicopters in terms of safety and efficiency at small sizes [1], and unlike normal helicopters, the rotor pitch is fixed as they rotate.

There are many applications for a quadrotor platform, both in the military and the civil sectors. Some of their possible applications are [2]:

Military applications

- Surveillance and Reconnaissance
- Search and Rescue
- Communications
- Logistics missions

Civil applications:

- Search and Rescue
- Surveillance
- Wildlife monitoring
- Terrain mapping
- Cave exploring
- Network and Communications
- Research and Development
- Entertainment

Currently, unmanned aerial vehicles applications are mostly in the military field. [3] However, due to having a very promising application in research an development, the quadrotor makes a good platform for testing new technological advances. One of the most recent experiments on quadrotors involved forty nine quadrotors in formation flight from ARS Electronica performing a choreographed dance [4] (Figure 1.3).

1.2 Literature review

There has been some research performed in the area of trajectory generation and constrained control of unmanned vehicles. A. de Luca [5, 6] has worked on generating trajectories for robots, where he discusses a spline interpolation method to solve a minimum time optimization problem, while staying under torque and velocity constraints. Y. Bouktir [7] also used a spline interpolation method to generate optimal-time trajectories and applied it to a micro quadrotor.

Daniel Mellinger and Nathan Michael [8] from the University of Pennsylvania have been working with quadrotors. They have rewritten the equations of motion as algebraic functions of a flat output: outputs that can express the states and inputs of

Figure 1.3. Quadrotor formation flight, image courtesy of http://www.aec.at.

the system in terms of its outputs and their derivatives [9]. This approach facilitates the automatic trajectory generation for the system. They also included position constraints and generated trajectories that require large, feasible accelerations for the quadrotor. In [10] they generate trajectories by designing a sequence of controllers to drive the system to a desired goal state G.

H. Huang in [11, 12] discusses the design of safe, aggressive maneuvers and control for a back flip trajectory. Hoffmann in [13] constructed a dynamically feasible, desired speed profile for a given sequence of waypoints. M. Hehn and R. D'Andrea [14, 15] from ETH Zurich have worked on trajectory generation for quadrotors. They implemented dynamic constraints to an optimal control method and verified the existence of optimal trajectories.

I. Palunko, R. Fierro, and P. Cruz [16] address the problem of quadrotor trajectory generation and tracking while carrying a suspended payload. They solved this by developing an optimal controller based on the dynamic programming technique: breaking down a problem into several subproblems. H. Kim, D. Shim and S. Sastry [17] looked into nonlinear model predictive tracking control (NMPTC), and applied it generate trajectories to unmanned rotorcraft while staying under input and state constraints.

1.3 Thesis outline

This thesis derives the equations of motion for the quadrotor, the linearization of the system, and the use of an optimal control algorithm to constrain the input and outputs of the system and develop a feasible trajectory. Chapter 3 focuses on the development of the Model Predictive Control (MPC) formulation to derive an optimal controller and feasible trajectories. Chapter 4 addresses the State Dependent Coefficient (SDC) form which allows the representation of the system in a pseudo-linear form, as well as the use of a Ricatti controller. Chapter 5 deals with a nonlinear version of the MPC by using the SDC form from Chapter 4. Chapter 6 discusses the effect of combining the Riccati controller and the nonlinear MPC. Finally, chapter 7 addresses the overall results from all schemes and provides some final remarks on the research presented in this thesis.

CHAPTER 2

THE QUADROTOR MODEL

2.1 Reference frames

In order to formulate an appropriate mathematical model of the quadrotor, it is first required to define a set of coordinate systems for specifying the position, velocity, forces and moments acting on the vehicle [18]. Let the inertial reference frame be the surface of the earth, and the body frame to be fixed to the body of the quadrotor. For the body axes system the center of gravity is selected to be the origin of the frame, the positive **x** direction points toward rotor 1, positive **y** direction points toward its right or rotor 2, and the positive **z** axis direction of the frame points downward so as to complete the form of a right-hand system, as shown in Fig. (2.1).

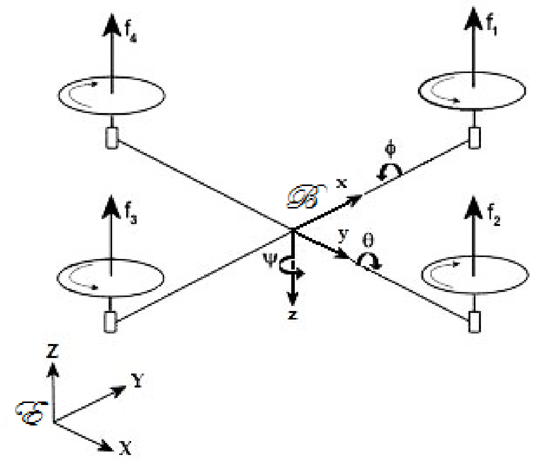


Figure 2.1. Quadrotor coordinate frames.

2.2 Rotational matrix

In the mathematical formulation of the vehicle, it is necessary to relate the body frame to the inertial frame. This can be achieved by using a rotation matrix that translates the body dynamics into another frame [19], which for the quadrotor case is some reference in the surface. Such orientation can be described by using the Euler angles method [20], where three consecutive rotations about the three Euler angles ϕ, θ and ψ make the inertial frame coincide with the Body axes frame. A 3-2-1 Euler angle sequence is used to construct the rotation matrix such that the rotation from inertial frame to the body frame is defined as

$$T_I^b = R(\phi)R(\theta)R(\psi) \tag{2.1}$$

where

$$R(\phi) = \begin{bmatrix} 1 & 0 & 0 \\ 0 & C(\phi) & S(\phi) \\ 0 & -S(\phi) & C(\phi) \end{bmatrix}, R(\theta) = \begin{bmatrix} C(\theta) & 0 & -S(\theta) \\ 0 & 1 & 0 \\ S(\theta) & 0 & C(\theta) \end{bmatrix}, R(\psi) = \begin{bmatrix} C(\psi) & S(\psi) & 0 \\ -S(\psi) & C(\psi) & 0 \\ 0 & 0 & 1 \end{bmatrix}$$

such that

$$T_I^b = \begin{bmatrix} C(\theta)C(\psi) & C(\theta)S(\psi) & -S(\theta) \\ S(\theta)S(\phi)C(\psi) - S(\psi)C(\phi) & S(\psi)S(\theta)S(\phi) + C(\psi)C(\phi) & S(\phi)C(\theta) \\ S(\theta)C(\phi)C(\psi) + S(\psi)S(\phi) & S(\psi)S(\theta)C(\phi) - C(\psi)S(\phi) & C(\phi)C(\theta) \end{bmatrix} \tag{2.2}$$

Where $C(\cdot)$ and $S(\cdot)$ denote $Cos(\cdot)$ and $Sin(\cdot)$.

In order to avoid ambiguities and prevent possible singularities in the equations of motion due to the Euler angles technique, it is necessary to restrict the range that each angle is able to take. Let the maximum and minimum value of each angle be

$$\begin{array}{ccccc} -\pi & \leq & \phi & \leq & \pi \\ -\frac{\pi}{2} & < & \theta & < & \frac{\pi}{2} \\ -\pi & < & \psi & < & \pi \end{array} \tag{2.3}$$

2.3 System Modeling

Several assumptions need to be made in order to derive the equations of motion governing the quadrotor helicopter. Such assumptions are:

- Center of gravity of the vehicle and the body frame origin coincide.
- Earth is flat and non rotating.
- Structure is rigid.
- Vehicle is symmetric.
- Vehicle has a fixed mass.

With these assumptions it is now possible to obtain the kinematics and dynamics equations of motion.

2.3.1 Quadrotor kinematics

2.3.1.1 Rotational kinematics

The rotational kinematics are the vehicle angular velocity components and are usually expressed in the body frame (p, q, r), so it is necessary to map them to the derivatives of the Euler angles by a series of rotations. Therefore:

$$\begin{bmatrix} p \\ q \\ r \end{bmatrix} = R(\phi)R(\theta) \begin{bmatrix} 0 \\ 0 \\ \dot{\psi} \end{bmatrix} + R(\phi) \begin{bmatrix} 0 \\ \dot{\theta} \\ 0 \end{bmatrix} + \begin{bmatrix} \dot{\phi} \\ 0 \\ 0 \end{bmatrix} \tag{2.4}$$

Substituting $R(\phi)$ and $R(\theta)$

$$\begin{bmatrix} p \\ q \\ r \end{bmatrix} = \begin{bmatrix} 1 & 0 & -S(\theta) \\ 0 & C(\phi) & S(\phi)C(\theta) \\ 0 & -S(\phi) & C(\phi)C(\theta) \end{bmatrix} \begin{bmatrix} \dot{\phi} \\ \dot{\theta} \\ \dot{\psi} \end{bmatrix} \tag{2.5}$$

$$\begin{bmatrix} \dot{\phi} \\ \dot{\theta} \\ \dot{\psi} \end{bmatrix} = \begin{bmatrix} 1 & 0 & -S(\theta) \\ 0 & C(\phi) & S(\phi)C(\theta) \\ 0 & -S(\phi) & C(\phi)C(\theta) \end{bmatrix}^{-1} \begin{bmatrix} p \\ q \\ r \end{bmatrix} \tag{2.6}$$

$$\begin{bmatrix} \dot{\phi} \\ \dot{\theta} \\ \dot{\psi} \end{bmatrix} = \begin{bmatrix} 1 & T(\theta)S(\phi) & T(\theta)C(\phi) \\ 0 & C(\phi) & -S(\theta) \\ 0 & \frac{-S(\phi)}{C(\theta)} & \frac{C(\phi)}{C(\theta)} \end{bmatrix} \begin{bmatrix} p \\ q \\ r \end{bmatrix} \tag{2.7}$$

2.3.1.2 Translational kinematics

Let V_I denote the velocity components in the inertial frame ($\dot{x}$,$\dot{y}$,$\dot{z}$) and V_B the velocity vector of the rigid body (u,v,w). It is possible to translate the motion in the Body frame to the Inertial frame by using the rotational matrix of equation (2.2), such that

$$V_I = [T_I^b]^T V_B \tag{2.8}$$

Therefore

$$\begin{bmatrix} \dot{x} \\ \dot{y} \\ \dot{z} \end{bmatrix} = \begin{bmatrix} C(\theta)C(\psi) & S(\theta)S(\phi)C(\psi) - C(\phi)S(\psi) & S(\theta)C(\phi)C(\psi) + S(\phi)S(\psi) \\ C(\theta)S(\psi) & S(\theta)S(\phi)S(\psi) + C(\phi)C(\psi) & S(\theta)C(\phi)S(\psi) - S(\phi)C(\psi) \\ -S(\theta) & C(\theta)S(\phi) & C(\theta)C(\phi) \end{bmatrix} \begin{bmatrix} u \\ v \\ w \end{bmatrix} \tag{2.9}$$

2.3.2 Quadrotor dynamics

2.3.2.1 Translational dynamics

The translational dynamics are based on Newton's second law of motion

$$F_I = m\left(\frac{dV_I}{dt}\right) \tag{2.10}$$

we can obtain the velocity component V_I in terms of the body frame velocity such that

$$[T_I^b]^T F_b = m\frac{d}{dt}([T_I^b]^T V_b) \tag{2.11}$$

taking its derivative with respect to time

$$m\left([\dot{T_I^b}]^T V_b + [T_I^b]^T \dot{V_b}\right) = [T_I^b]^T F_b \tag{2.12}$$

Premutiplying by $[T_I^b]$

$$m[T_I^b]\dot{[T_I^b]}^T V_b + m\dot{V_b} = F_b \tag{2.13}$$

Solving for $\dot{V_b}$

$$m\dot{V_b} = mS(\omega_B)V_b + F_b \tag{2.14}$$

where $S(\omega_B)$ is the skew symmetric form of the angular velocity vector in the body frame (p,q,r)

Expanding equation (2.14), the translational dynamics in the body frame can be defined as

$$\begin{bmatrix} \dot{u} \\ \dot{v} \\ \dot{w} \end{bmatrix} = \frac{1}{m}\begin{bmatrix} F_x \\ F_y \\ F_z \end{bmatrix} + [T_I^b]\begin{bmatrix} 0 \\ 0 \\ g \end{bmatrix} - \begin{bmatrix} q\cdot w - r\cdot v \\ r\cdot u - p\cdot w \\ p\cdot v - q\cdot u \end{bmatrix} \tag{2.15}$$

where m is the mass, g is the gravity, and F_x, F_y and F_z are the forces in the x, y and z axis respectively.

2.3.2.2 Rotational dynamics

Euler's equation for rotationa dynamics relates the change in angular momentum in a specific point to the sum of external moments acting around that point, such that

$$\frac{d}{dt}\left(I_I \omega_I\right) = M_I \tag{2.16}$$

where I_I is the inertia matrix given in the inertial frame, ω_I is the angular velocity in the inertial frame, and M_I is the representation of the external moment vector in the inertial frame.The rotation matrix $[T_I^b]$ can then be used to translate the Inertia matrix and angular velocity in the inertial frame to the body frame, allowing to write equation (2.16) as

$$\frac{d}{dt}\left([T_I^b]^T I_b [T_I^b][T_I^b]^T \omega_b\right) = [T_I^b]^T M_b \tag{2.17}$$

$$\frac{d}{dt}\left([T_I^b]^T I_b \omega_b\right) = [T_I^b]^T M_b \tag{2.18}$$

Taking its time derivative

$$\dot{[T_I^b]}^T I_b \omega_b + [T_I^b]^T I_b \dot{\omega}_b = [T_I^b]^T M_b \tag{2.19}$$

Premultiplying everything by $[T_I^b]$

$$[T_I^b]\dot{[T_I^b]}^T I_b \omega_b + I_b \dot{\omega}_b = M_b \tag{2.20}$$

and

$$[T_I^b]\dot{[T_I^b]}^T = -S(\omega_b) \tag{2.21}$$

Therefore

$$-S(\omega_b) I_b \omega_b + I_b \dot{\omega}_b = M_b \tag{2.22}$$

Solving for $\dot{\omega}_b$

$$\dot{\omega}_b = I_b^{-1}\left(M_b + S(\omega_b) I_b \omega_b\right) \tag{2.23}$$

Equation (2.23) can also be written as

$$\dot{\omega_b} = I_b^{-1} \left(M_b + \omega_b \times (I_b \omega_b) \right) \tag{2.24}$$

Where the Inertia matrix is given by

$$I_b = \begin{bmatrix} I_x & I_{xy} & I_{xz} \\ I_{yx} & I_y & I_{yz} \\ I_{zx} & I_{zy} & I_z \end{bmatrix} \tag{2.25}$$

and the moment vector is given by

$$M_b = \begin{bmatrix} L \\ M \\ N \end{bmatrix} \tag{2.26}$$

where L,M and N are the roll, pitch and yaw moments in the body frame around the vehicle center of gravity.

2.4 Simplification of the equations of motion

The equations of motion previously derived represent the general state of the quadrotor with respect to the surface of the Earth. However, since most control systems are derived based on a linear model, these governing equations need to be further simplified or linearized around some reference condition. One possible way to simplify the equations of motion is by using the Small Disturbance Theory. This method assumes that the general motion of the vehicle consists of small amplitude deviations or disturbances from a steady operating point. Therefore, the disturbed motion becomes

$$\begin{array}{lll} u = u_0 + \Delta u & v = v_0 + \Delta v & w = w_0 + \Delta w \\ p = p_0 + \Delta p & q = q_0 + \Delta q & r = r_0 + \Delta r \\ \phi = \phi_0 + \Delta \phi & \theta = \theta_0 + \Delta \theta & \psi = \psi_0 + \Delta \psi \\ x = x_0 + \Delta x & y = y_0 + \Delta y & z = z_0 + \Delta z \\ F_x = F_{x0} + \Delta F_x & F_y = F_{y0} + \Delta F_y & F_z = F_{z0} + \Delta F_z \\ L = L_0 + \Delta L & M = M_0 + \Delta M & N = N_0 + \Delta N \end{array}$$

Where the suffix $_0$ denotes the steady operating point.

For the purpose of this research, it is necessary to linearize the quadrotor around a trim condition. Such trim condition was chosen to be hover. While hovering, all the steady operating points are zero except for the vertical force F_z, which must compensate for the gravity effect. Therefore

$$\begin{array}{lll} u = \Delta u & v = \Delta v & w = \Delta w \\ p = \Delta p & q = \Delta q & r = \Delta r \\ \phi = \Delta \phi & \theta = \Delta \theta & \psi = \Delta \psi \\ x = \Delta x & y = \Delta y & z = \Delta z \\ F_x = \Delta F_x & F_y = \Delta F_y & F_z = m \cdot g + \Delta F_z \\ L = \Delta L & M = \Delta M & N = \Delta N \end{array}$$

The next step is to substitute those values into the equations of motion. By ignoring higher order terms and taking the sine of a small angle to be equal to itself, and the cosine of a small angle to be equal to one, the equations of motion then become

$$\begin{aligned}
\Delta\dot{u} &= -g\Delta\theta + \Delta F_x/m \\
\Delta\dot{v} &= g\Delta\phi + \Delta F_y/m \\
\Delta\dot{w} &= \Delta F_z \\
\Delta\dot{p} &= \Delta L/I_x \\
\Delta\dot{q} &= \Delta M/I_y \\
\Delta\dot{r} &= \Delta N/I_z \\
\Delta\dot{\phi} &= \Delta p \\
\Delta\dot{\theta} &= \Delta q \\
\Delta\dot{\psi} &= \Delta r \\
\Delta\dot{x} &= \Delta u \\
\Delta\dot{y} &= \Delta v \\
\Delta\dot{z} &= \Delta w
\end{aligned} \tag{2.27}$$

CHAPTER 3

THE MODEL PREDICTIVE CONTROL APPROACH

3.1 Introduction

Model predictive control (MPC) which can also be called Receding Horizon Control (RHC), is a technique in which a mathematical model of a system is used to solve a finite, moving horizon, closed loop optimal control problem [21] by using the current states of the system [22]. MPC is able to take into account the physical and mechanical limitations of the plant during the design process [23], and predict a number of future outputs of the system (called Prediction Horizon), in order to formulate an optimal controller effort to bring the system to a desired state given a reference trajectory.

This optimization problem is solved at each sampling interval but only the first part of the solution to the optimization problem is applied to the system until the next sampling interval. This routine is repeated for each subsequent time intervals [24].

3.2 Model Predictive Control Formulation

3.3 Plant Model and Prediction Horizon

Let a nonlinear system of the form

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

Where $\mathbf{x}(\mathbf{t}) \in \Re^n$ are the system states, and $\mathbf{u}(\mathbf{t}) \in \Re^m$ are the system inputs.

The system can then be modeled as a linear, discrete-time system form

$$\begin{aligned} x_{k+1} &= A_d x_k + B_d u_k \\ y_k &= C_d x_k + D_d u_k \end{aligned} \tag{3.2}$$

Where **k** is the current sample,$\mathbf{A_d} \in \Re^{nxn}$ is the state matrix, $\mathbf{B_d} \in \Re^{nxm}$ is the input matrix, $\mathbf{C_d} \in \Re^{pxn}$ is the output matrix, and $\mathbf{D_d} \in \Re^{pxm}$ is the feedforward matrix.

The objective is to drive the system towards a desired state. For this, the controller uses a Prediction Horizon N to predict a number of future states of the system. In order to estimate such outputs the current measurements of the system are employed within an estimator in order to predict its future behavior.

To predict such future states of the system, it is required to implement a state observer in the controller formulation. For this approach, a Linear Quadratic Estimator (Kalman filter) is used. It uses the current state to compute a Kalman gain for the discrete-time problem of equation (3.2) [25].

$$\begin{aligned} \hat{x}_{k+1+i} &= A_d \hat{x}_{k+i} + B_d u_{k+i} \\ \hat{y}_{k+i} &= C_d \hat{x}_{k+i} + D_d u_{k+i} \end{aligned} \tag{3.3}$$

where

$$i = 1, 2 \; ... \; N \tag{3.4}$$

By expanding equation (3.3), the predicted states and predicted outputs can be obtained based only on the initial state of the system and its future control input u_{k+j}

$$\begin{aligned} \hat{x}_{k+2} &= A_d \hat{x}_{k+1} + B_d u_{k+1} \\ \hat{x}_{k+3} &= A_d \hat{x}_{k+2} + B_d u_{k+2} \\ &= A_d(A_d \hat{x}_{k+1} + B_d u_{k+1}) + B_d u_{k+2} \\ &= A_d^2 \hat{x}_{k+1} + A_d B_d u_{k+1} + B_d u_{k+2} \\ &\vdots \\ \hat{x}_{k+N} &= A_d^{N-1} \hat{x}_{k+1} + \textstyle\sum_{j=1}^{N-1} A^{N-j-1} B_d u_{k+j} \end{aligned} \tag{3.5}$$

Once the state predictions have been derived, the output state equations can be derived in a similar way as equation (3.5).

$$\begin{aligned}
\hat{y}_{k+1} &= C_d\hat{x}_{k+1} + D_d u_{k+1} \\
\hat{y}_{k+2} &= C_d\hat{x}_{k+2} + D_d u_{k+2} \\
&= C_d(A_d\hat{x}_{k+1} + B_d u_{k+1}) + D_d u_{k+2} \\
&= C_d A_d\hat{x}_{k+1} + C_d B_d u_{k+1} + D_d u_{k+2} \\
&\vdots \\
\hat{y}_{k+N} &= C_d A_d^{N-1}\hat{x}_{k+1} + C_d(\textstyle\sum_{j=1}^{N-1} A_d^{N-j-1} B_d u_{k+j}) + D_d u_{k+N}
\end{aligned} \tag{3.6}$$

Thus, the standard prediction matrix is of the form

$$\begin{bmatrix} \hat{x}_{k+2} \\ \hat{x}_{k+3} \\ \hat{x}_{k+4} \\ \vdots \\ \hat{x}_{k+N+1} \end{bmatrix} = \begin{bmatrix} A \\ A^2 \\ A^3 \\ \vdots \\ A^N \end{bmatrix} \hat{x}_{k+1} + \begin{bmatrix} B & 0 & ... & 0 \\ AB & B & ... & 0 \\ A^2B & AB & ... & 0 \\ \vdots & \vdots & \ddots & \vdots \\ A^{N-1}B & A^{N-2} & ... & B \end{bmatrix} \begin{bmatrix} u_{k+1} \\ u_{k+2} \\ u_{k+3} \\ \vdots \\ u_{k+N} \end{bmatrix} \tag{3.7}$$

$$\begin{bmatrix} y_{k+1} \\ y_{k+2} \\ y_{k+3} \\ \vdots \\ y_{k+N} \end{bmatrix} = \begin{bmatrix} C \\ CA \\ CA^2 \\ \vdots \\ CA^{N-1} \end{bmatrix} \hat{x}_{k+1} + \begin{bmatrix} D & 0 & ... & 0 \\ CB & D & ... & 0 \\ CAB & CB & ... & 0 \\ \vdots & \vdots & \ddots & \vdots \\ CA^{N-2}B & CA^{N-3} & ... & D \end{bmatrix} \begin{bmatrix} u_{k+1} \\ u_{k+2} \\ u_{k+3} \\ \vdots \\ u_{k+N} \end{bmatrix} \tag{3.8}$$

Or, in its short form

$$\begin{aligned}
\bar{X}_k &= \bar{A}\hat{x}_{k+1} + \bar{B}\bar{U}_k \\
\bar{Y}_k &= \bar{C}\hat{x}_{k+1} + \bar{D}\bar{U}_k
\end{aligned} \tag{3.9}$$

3.4 Controller Design

The MPC algorithm requires the use of an objective function in its control formulation in order to calculate the optimal solution at each sampling interval. It must be chosen in a way such that the predicted outputs, derived from the prediction horizon N (equation 3.9) drive the system to a desired state or track a desired trajectory y_d, while at the same time, it should minimize the controller effort u_k required [26].

For the quadrotor scenario, the penalty function for the states in the objective function is chosen such that it penalizes the norm of the difference between the current output states and the desired trajectory $||y_k - y_d||^2$. In addition, we can choose a similar way to penalize quick changes in the actuator dynamics. This can be done by including the term $||u_k - u_{k-1}||^2$.

Therefore, the penalty function for the entire prediction horizon is of the form

$$J(\hat{x}_{k+1}, u_k) = \frac{1}{2} \sum_{j=1}^{N} \left(||\hat{y}_{k+j} - y_{d_{k+j}}||_Q^2 + ||u_{k+j} - u_{k+j-1}||_R^2 \right) \tag{3.10}$$

It is important to note that usually, the reference trajectory y_d is known in advance. This implies that the controller is able to react beforehand and predict a series of adequate inputs that will drive the system towards the desired goal.

In the quadratic form of equation (3.10) the term $||\hat{y}_{k+j} - y_{d_{k+j}}||_Q^2$ can be expanded into $(y_k - y_{d_k})^T Q (y_k - y_{d_k})$, where Q is a diagonal, positive definite matrix. Similarly we can expand the term $||u_{k+j} - u_{k+j-1}||_R^2$ into $(u_{k+j} - u_{k+j-1})^T R (u_{k+j} - u_{k+j-1})$, where R is also a diagonal, positive definite matrix.

Expanding the summation terms of equation (3.10)

$$J(\hat{x}_{k+1}, U_k) = \frac{1}{2} \left[(\bar{Y}_k - \bar{Y}_d)^T \bar{Q} (\bar{Y}_k - \bar{Y}_d) + (\bar{U}_k - u_{k-1})^T \bar{R} (\bar{U}_k - u_{k-1}) \right] \tag{3.11}$$

with $\bar{Q}$ and $\bar{R}$ being block diagonal matrices defined as

$$\bar{Q} = \begin{bmatrix} Q & & & \\ & Q & & \\ & & \ddots & \\ & & & Q \end{bmatrix} \tag{3.12}$$

$$\bar{R} = \begin{bmatrix} 2R & -R & & & \\ -R & 2R & -R & & \\ & & \ddots & & \\ & & & -R & R \end{bmatrix} \tag{3.13}$$

Substituting $\bar{Y}_k$ from equation (3.9) into equation (3.11) the objective function can be expressed as [27]:

$$J(\hat{x}_{k+1}, U_k) \;=\; \tfrac{1}{2} U_k^T H U_k + U_t^T f \tag{3.14}$$

where

$$H \;=\; \bar{D}^T \bar{Q} \bar{D} + \bar{R} \tag{3.15}$$

$$f \;=\; \left[\bar{D}^T \bar{Q} \bar{C} - \bar{D}^T \bar{Q}\right] \begin{bmatrix} \hat{x}_{k+1} \\ \bar{Y}_d \end{bmatrix} - \begin{bmatrix} Ru_k \\ 0 \\ \vdots \\ 0 \end{bmatrix} \tag{3.16}$$

3.5 Quadratic Programming

Since the cost function from equation (3.14) is of quadratic form, a Linear Quadratic Programming method can be used to solve the optimization problem. The Linear Quadratic Programming method solves optimization problems of the form [28]:

$$\begin{aligned} f(x) = &\ \tfrac{1}{2}x^T H x + f^T x \\ s.t. \qquad &\ Ax \leq b \\ &\ A_{eq}x = B_{eq} \end{aligned} \tag{3.17}$$

The idea behind QP is to minimize the quadratic function of equation (3.14) by looking for a feasible search direction

3.5.1 Input and State Constraint handling

Special attention can be given to the constraint handling capabilities of the MPC problem formulation now that the objective function has been specified (equation 3.14).In the case of the quadrotor, it is needed to constrain both the total thrust force of each rotor and restrict the magnitude of the angles in order to stay within the limits allowed by the Euler angles formulation discussed in Chapter 2.

3.5.2 Input Constraints

It is needed to constrain the maximum force each rotor is able to deliver in our mathematical model in order to make it perform in such a way that it resembles a realistic physical model. The forces need to be within some lower bound l_b and some upper bound u_b

$$l_b \ \leq F_i \leq \ u_b \tag{3.18}$$

These constraints can be expressed in matrix form as follows

$$\begin{bmatrix} I \\ -I \end{bmatrix} U_k \ \leq \ \begin{bmatrix} u_b \\ l_b \end{bmatrix} \tag{3.19}$$

Where u_b and l_b denote upper bound and lower bound respectively, and I is an identity matrix.

3.5.3 Output State Constraints

It is also of high importance to limit the angles so that no singularities are encountered due to the limitations in the model description.In the case of the quadrotor, the angles need to be within

$$\begin{aligned} -\pi &\leq \phi \leq \pi \\ -\tfrac{\pi}{2} &\leq \theta \leq \tfrac{\pi}{2} \\ -\pi &\leq \psi \leq \pi \end{aligned} \tag{3.20}$$

Angular constraints can be interpreted as output variable constraints [29] of the form

$$\begin{bmatrix} -\bar{D} \\ \bar{D} \end{bmatrix} U_k \leq \begin{bmatrix} \bar{C}\hat{x}_{k+1} - \bar{Y}_k - \alpha_u \\ \bar{C}\hat{x}_{k+1} - \bar{Y}_k + \alpha_l \end{bmatrix} \tag{3.21}$$

Where α_u and α_l are the upper and lower constraints for the output variables.

3.5.4 Combining Input and Output State Constraints

Both the input and output variable constrains can be integrated into one only equation of the form

$$\Gamma\bar{Y} = \Gamma(\bar{C}\hat{x}_{k+1} + \bar{D}U_k) \leq \delta \tag{3.22}$$

where δ is a matrix containing both input and output variable constrains, and Γ is a matrix that relates the outputs with its constraints.

The constraints of equation (3.22) can be expressed in terms of U_k as follows

$$\begin{bmatrix} I \\ -I \\ \Gamma\bar{D} \end{bmatrix} U_k \leq \begin{bmatrix} b_u \\ -b_l \\ \alpha - \Gamma(\bar{C}\hat{x}_{k+1} + \bar{Y}_k) \end{bmatrix} \tag{3.23}$$

In summary, the Model Predictive Control algorithm can be represented with the following flow chart

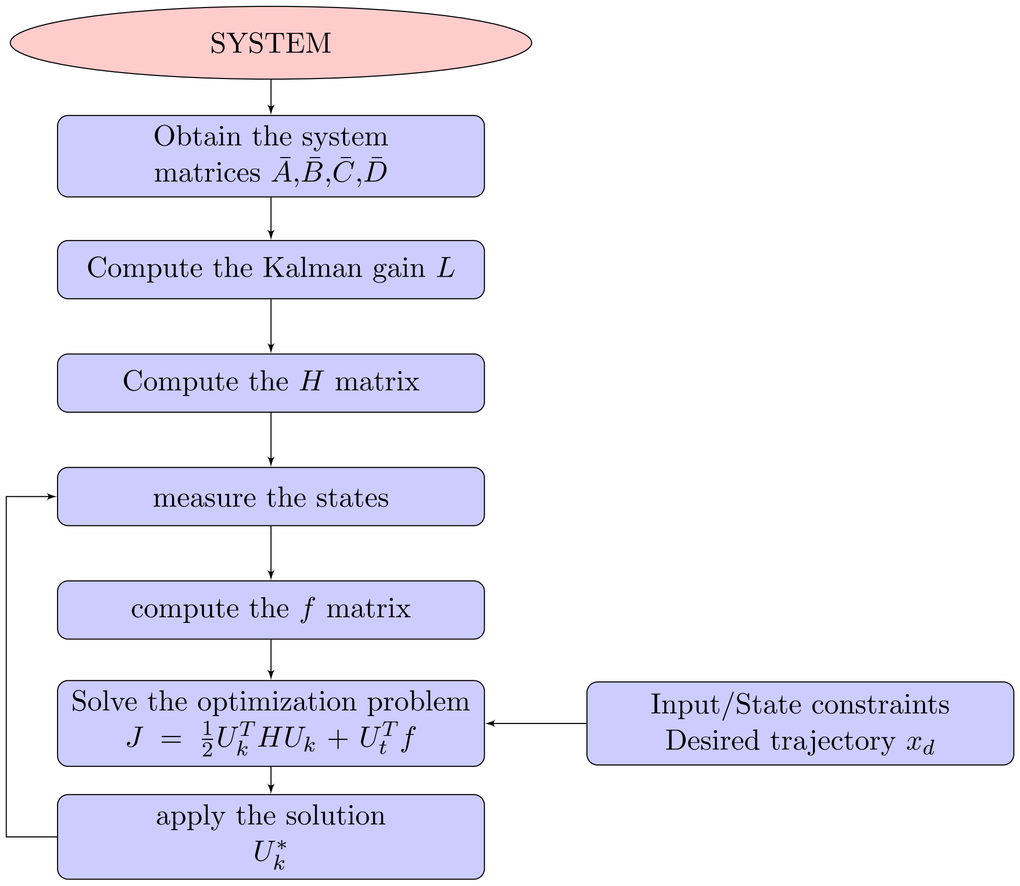

SYSTEM
Obtain the system matrices $\bar{A}$,$\bar{B}$,$\bar{C}$,$\bar{D}$
Compute the Kalman gain L
Compute the H matrix
measure the states
compute the f matrix
Solve the optimization problem $J = \frac{1}{2}U_k^T H U_k + U_t^T f$
Input/State constraints Desired trajectory x_d
apply the solution U_k^*

Figure 3.1. MPC Algorithm Flow Chart.

3.6 Results

3.6.1 Choice of **N** and **M**

One of the key parameters in the Model Predictive Control optimization problem is choosing an appropriate Prediction Horizon (**N**) and Control Horizon (**M**). Their values will directly impact the performance of the controller; choosing a high Prediction Horizon will improve the performance, but will increase the computational time, while the value of the Control Horizon will determine how fast the system will react.

It is desired to select an appropriate Prediction and Control Horizon. For this reason, a ten second simulation of different cases of **N** and **M** for a smooth reference trajectory was run, and the Root Mean Square (**RMS**) of the error on the position was obtained.

The **RMS** of X,Y and Z axis position error for the benchmark is shown by the following graphs

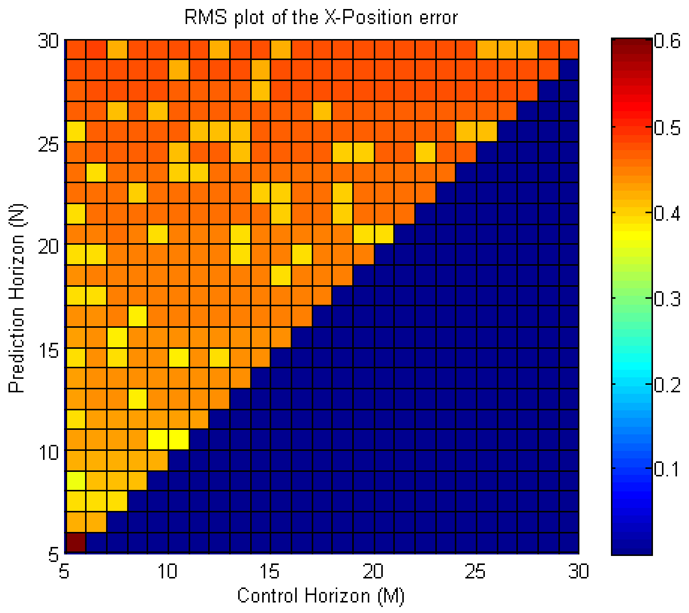


Figure 3.2. MPC RMS of the X position error benchmark.

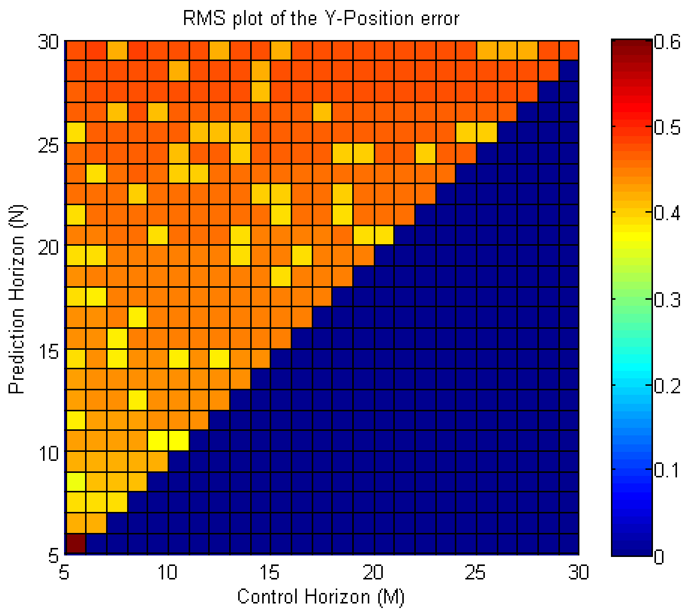


Figure 3.3. MPC RMS of the Y position error benchmark.

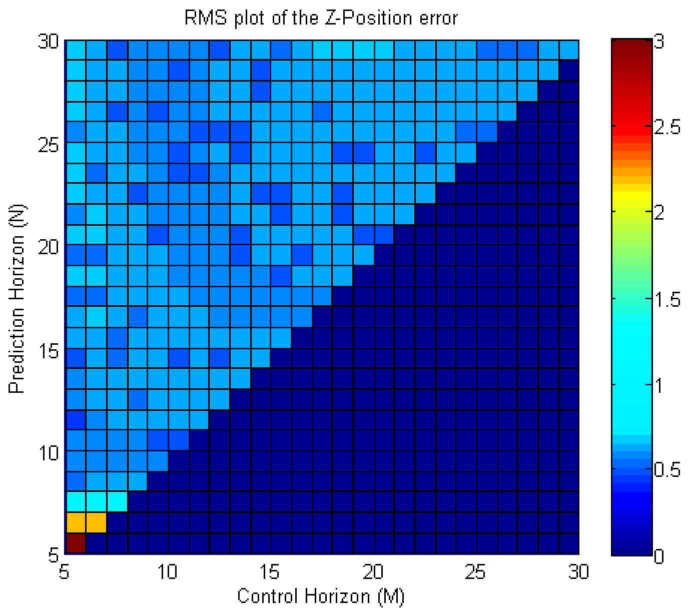


Figure 3.4. MPC RMS of the Z position error benchmark.

It is also desired to know the total computation time for each **N** and **M** scenario. This is illustrated by Figure 3.5.

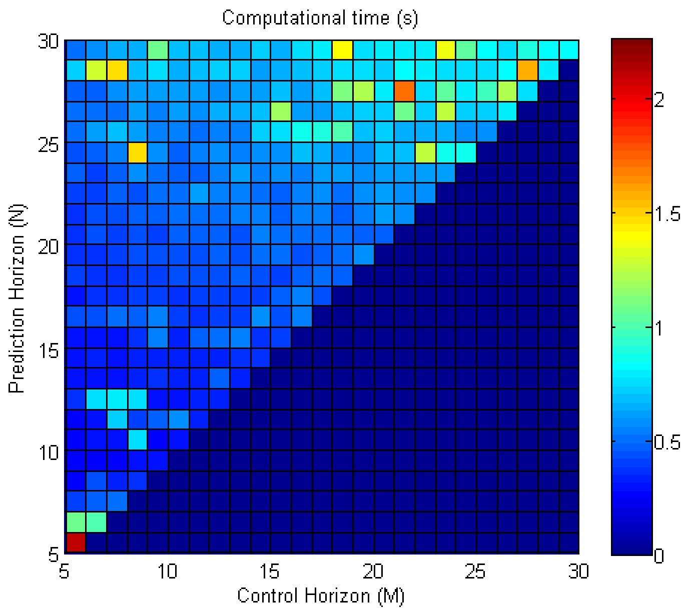


Figure 3.5. MPC simulation time benchmark.

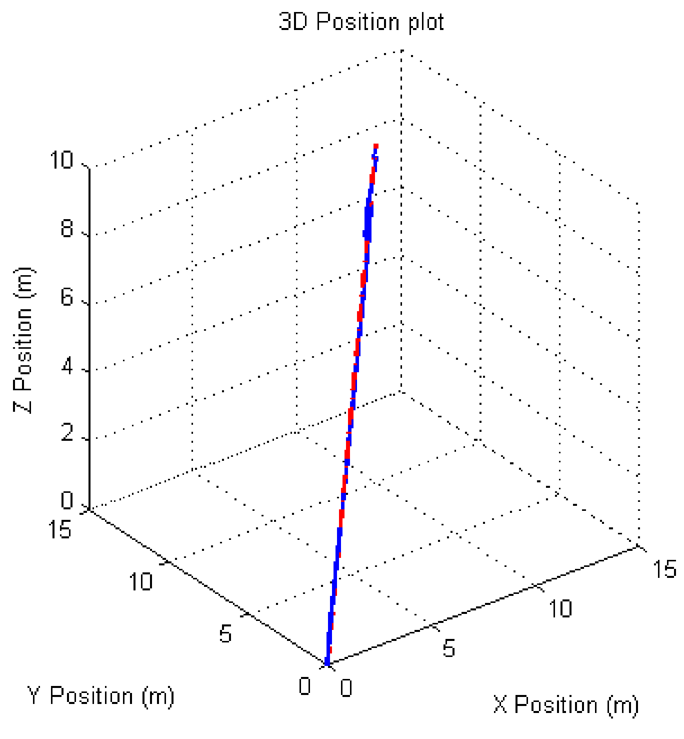
These results show that for values on **N** > 20 the computational time increases significantly, but the **RMS** of the position error only improves slightly. A good choice of **N** and **M** would be **N** $= 11$, **M** $= 5$. This combination has a small **RMS** and an overall fast computational time. However, for the purpose of this thesis, **N** was chosen to be 40, and **M** to be 25.

3.6.2 Simulation Results

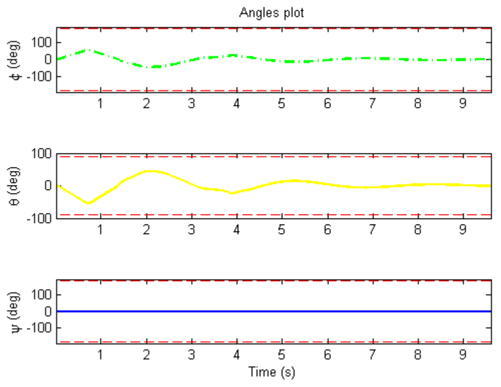
To illustrate the results of the MPC algorithm, four scenarios will be considered. Each scenario involves different types of reference trajectories and uses the previously established values of $\mathbf{N}$ and $\mathbf{M}$.The output state constraints are the ones defined in section (4.5), while the force constraint on each motor are set to be as $\frac{1}{2}mg \approx 2.5[N]$. The four scenarios are the following

- Scenario 1: Smooth trajectory
- Scenario 2: Helicoidal trajectory
- Scenario 3: Straight lines trajectory
- Scenario 4: Spherical Spiral trajectory

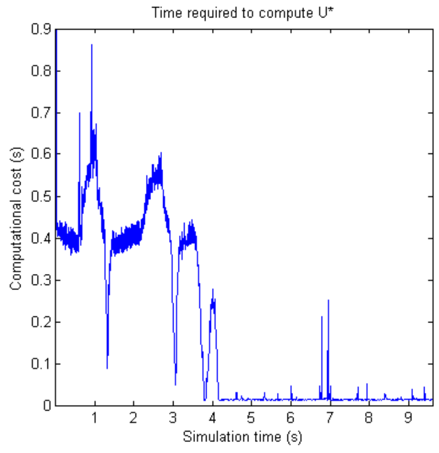
The results of the four scenarios are presented below



(a) 3D position

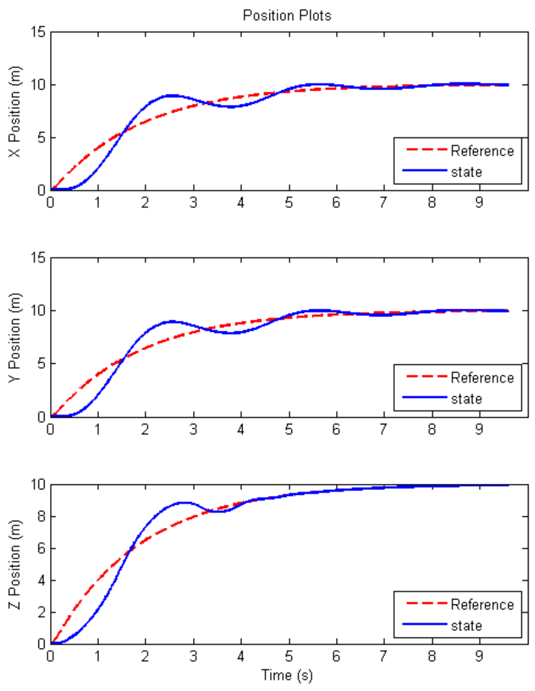


(b) Angular position

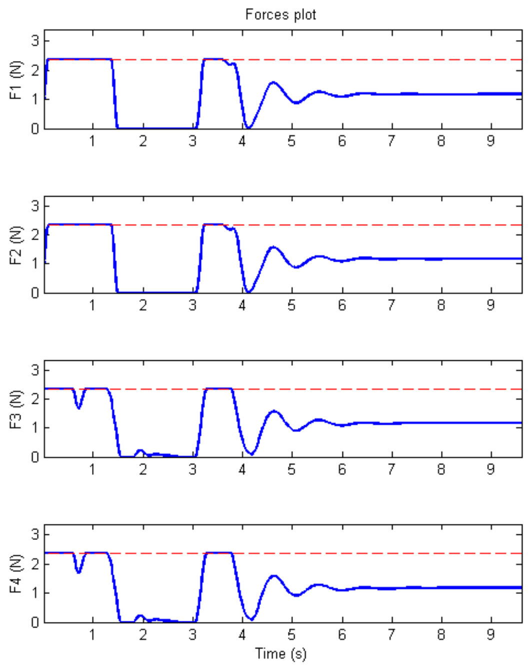


(c) Computational time

Figure 3.6. MPC Scenario 1: Smooth trajectory.

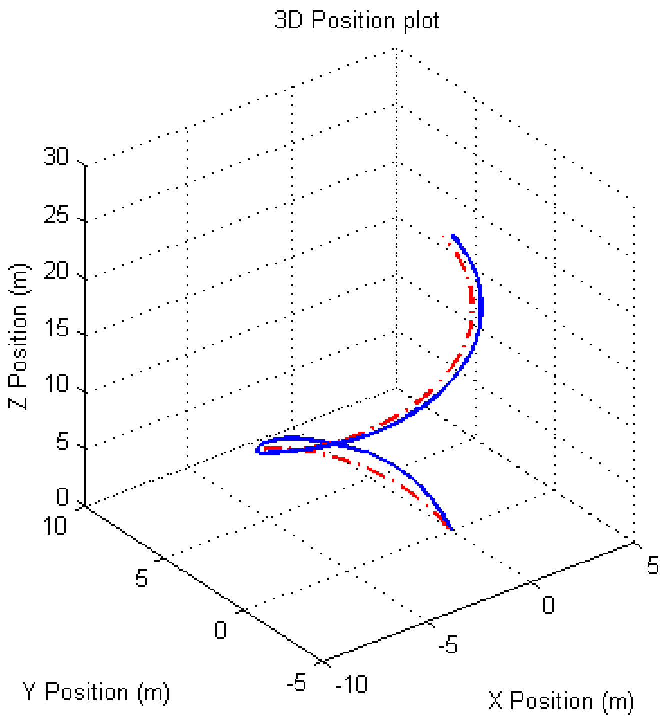


(a) Position

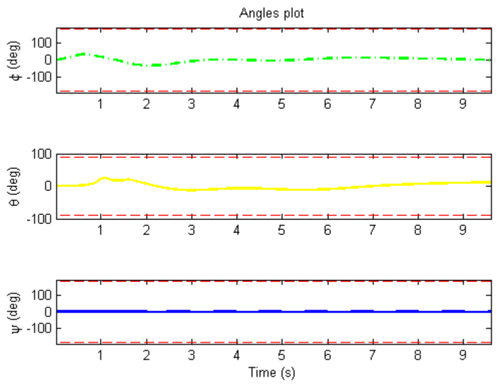


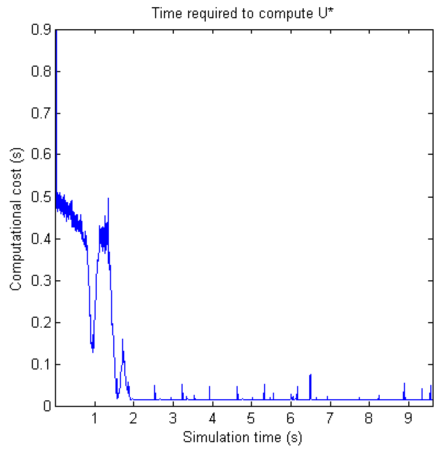
(b) Forces

Figure 3.7. MPC Scenario 1: Position and Forces.



(a) 3D position

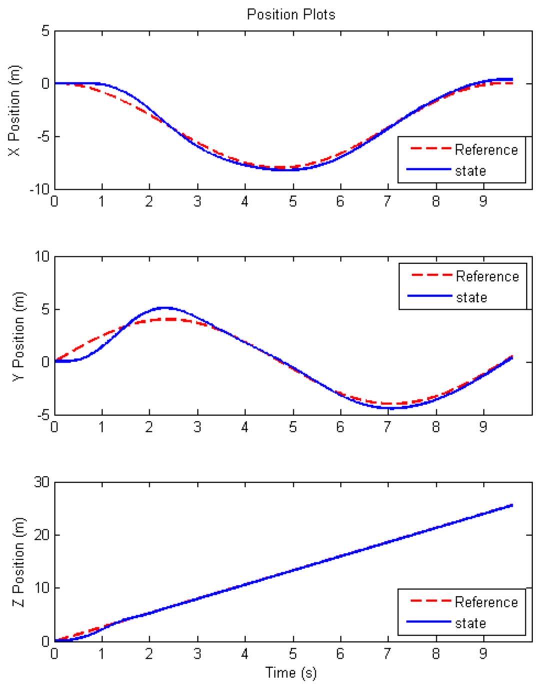




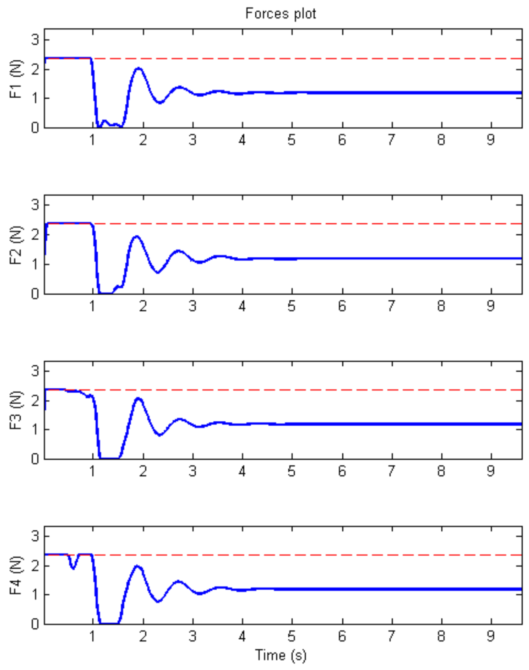
(b) Angular position

(c) Computational time

Figure 3.8. MPC Scenario 2: trajectory.

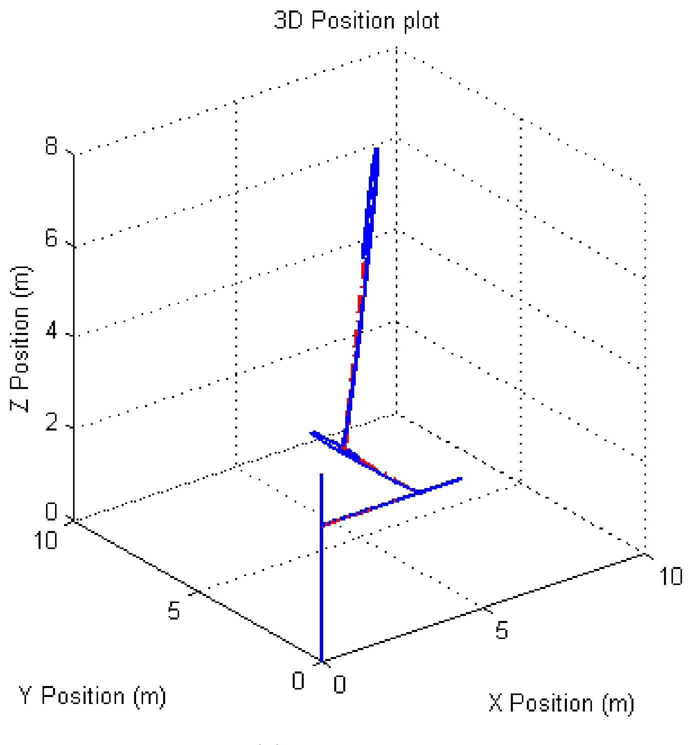


(a) Position

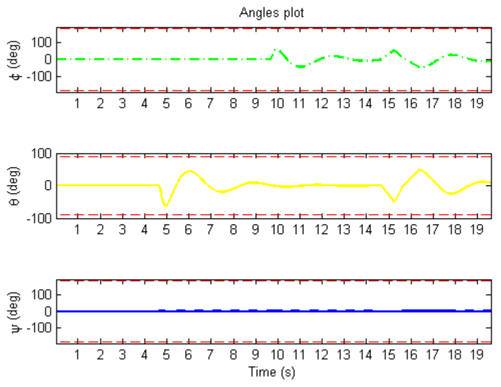


(b) Forces

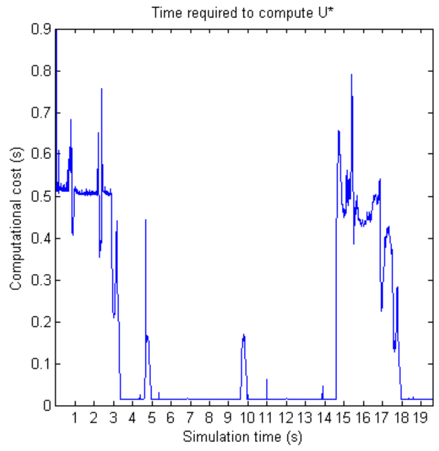
Figure 3.9. MPC Scenario 2: Position and Forces.



(a) 3D position

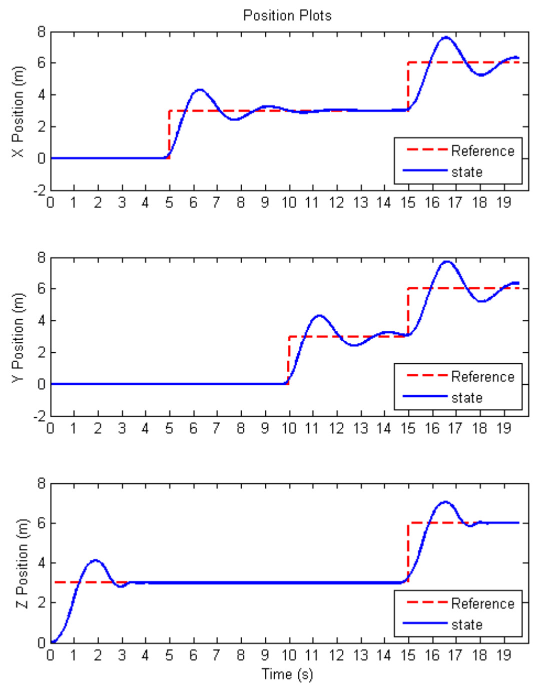


(b) Angular position

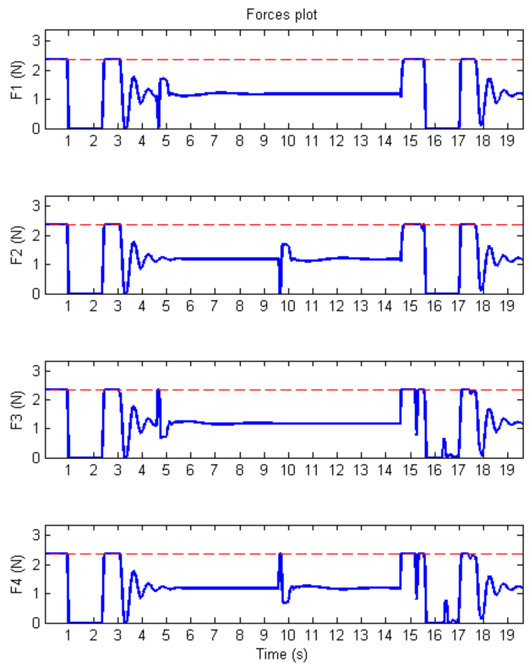


(c) Computational time

Figure 3.10. MPC Scenario 3: Straight lines trajectory.

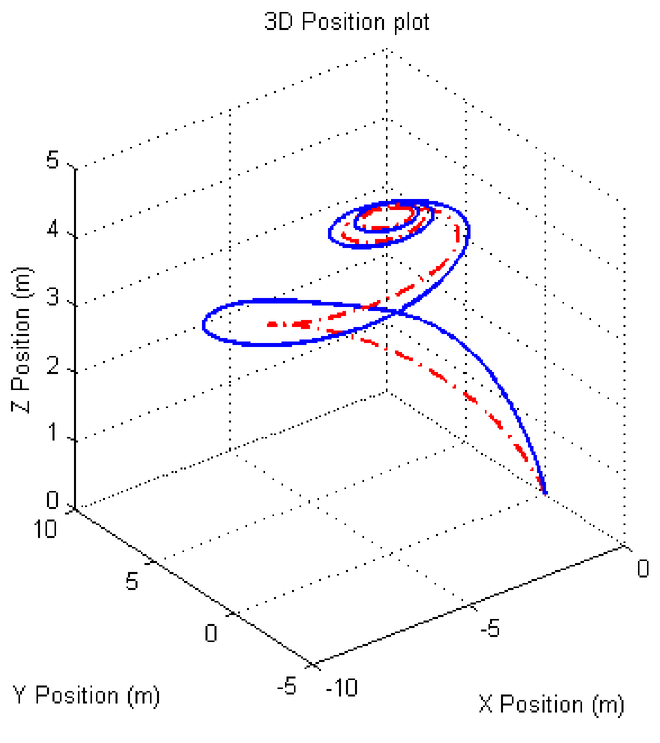


(a) Position

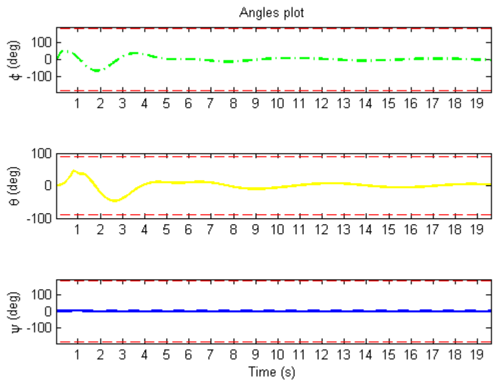


(b) Forces

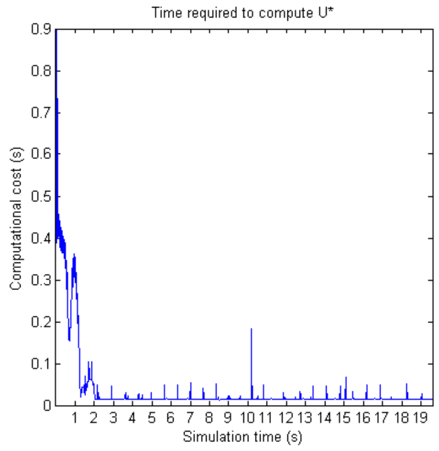
Figure 3.11. MPC Scenario 3: Position and Forces.



(a) 3D position

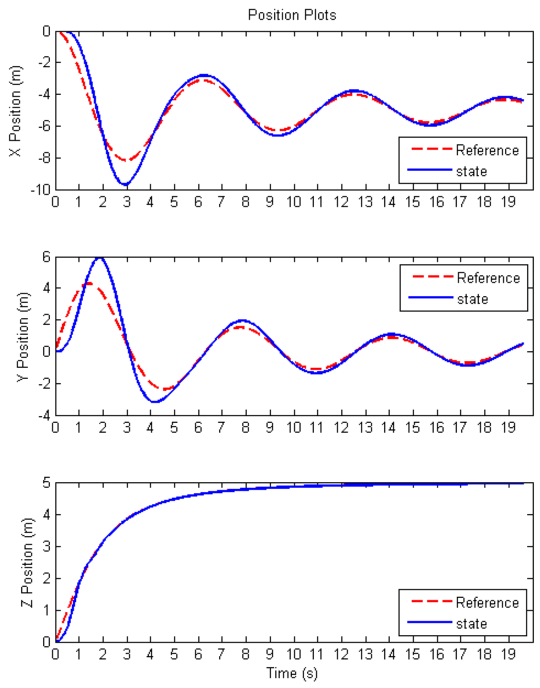


(b) Angular position

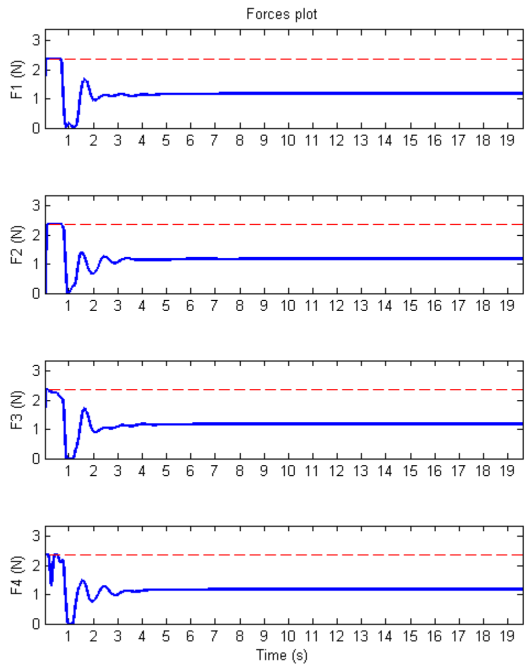


(c) Computational time

Figure 3.12. MPC Scenario 4: Spherical spiral trajectory.



(a) Position



(b) Forces

Figure 3.13. MPC Scenario 4: Position and Forces.

CHAPTER 4

THE STATE DEPENDENT RICCATI EQUATION APPROACH

4.1 Introduction

The MPC formulation worked well as a linear approach for the quadrotor. However, it is now desired to expand the concept of trajectory generation and constrain handling to the nonlinear version of the system. This can be accomplished by using the State Dependent Riccati Equation (SDRE) formulation.

The SDRE technique requires the manipulation of the nonlinear equations of motion previously derived in Chapter 2. It uses a State Dependent Coefficient (SDC) form to factorize the nonlinear equations of motion into matrix form and fully capture the model nonlinearities [30]. Thus, a state space representation of the quadrotor system can be created, where each of its system matrices are now expressed as functions of the current state [31].

This method uses the quadratic form performance index of equation (3.10) to solve for an infinite horizon optimal problem, for which its solution is locally stable [32]. It can also be considered to be an improvement over the Linear Quadratic Control formulation [33].

4.2 SDRE Method

Let a nonlinear system be of the following form

$$\dot{x} = f(x(t), u(t)) \tag{4.1}$$

The State Dependent Riccati Equation formulation involves transforming the nonlinear system of equation (4.1) into the following state space form

$$\dot{x} = A(x)x + B(x)u \tag{4.2}$$

Where $\mathbf{x} \in \Re^n$ is the state vector, $\mathbf{u} \in \Re^m$ is the input vector, and $A(x) \in \Re^{n\times n}$ and $B(x) \in \Re^{n\times m}$ are the pseudo-linear system matrices in the state dependent coefficient form. The SDC form is obtained by the factorization of the equations of motion.

It is important to note that this representation of $A(x)$ and $B(x)$ is not unique. Different state dependent matrices can be obtained from the equations of motion, and a solution to the optimization problem may or may not exist.

The State Dependent Riccati Equation formulation minimizes the infinite horizon objective function

$$min\ J(x,u) = \int_0^\infty (x^T Q(x)x + u^T R(x)U)dt \tag{4.3}$$

Where $Q(x) \geq 0$ is a positive semidefinite matrix and $R(x) > 0$ is a positive definite matrix.

For the quadrotor case, the discrete-time equivalent of equation (4.2) is required. This is obtained by evaluating both $A(x)$ and $B(x)$ matrices at each sample interval and then performing the discretization using a zero order hold.

Let the discrete-time equivalent of the system be of the following form

$$x_k = A_k(x_k)x_k + B_k(x_k)u_k \tag{4.4}$$

Since the equation (4.4) is of pseudo-linear form, the system matrices can be considered to be constants at each sampling interval[34]. This allows the SDRE optimization problem to solve for the Discrete-time Algebraic Riccati Equation (DARE) which is of the form

$$A_k^T P_k A_k - P_k - A_k^T P_k B(B^T P_k B + R)^{-1} B^T P_k A_k + Q = 0 \tag{4.5}$$

Where Q and R are weight matrices.

The solution P_k from equation (4.5) is then used to compute the Kalman gain sequence[35].

$$K_k = \left(B_k(x_k)^T P_k B_k(x_k) + R\right)^{-1} B_k(x_k)^T P_k A_k(x_k) \tag{4.6}$$

So that

$$u_k = -K_k x_k \tag{4.7}$$

is the optimal solution to the SDRE problem formulation.

The State Dependent Riccati Equation algorithm can be summarized as following

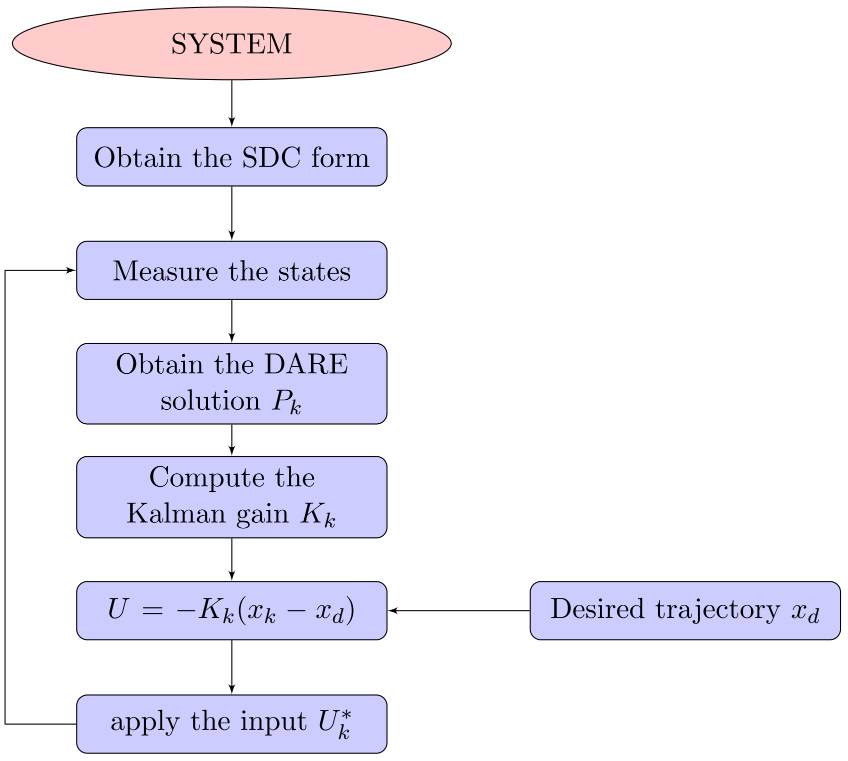


Figure 4.1. SDRE Algorithm Flow Chart.

4.3 Results

For the quadrotor case, one possible way to factorize the equations of motion into SDC form is presented below. However, this choice of $A(x)$ and $B(x)$ is not an unique representation of such equations of motion. Different choices of $A(x)$ and $B(x)$ can impact the performance and stability of the system.

$$A(x) = \begin{bmatrix} 0 & r & -q & 0 & 0 & 0 & 0 & \frac{-gS(\theta)}{\theta} & 0\ 0\ 0\ 0 \\ -r & 0 & p & 0 & 0 & 0 & \frac{gC(\theta)S(\phi)}{\phi} & 0 & 0\ 0\ 0\ 0 \\ q & -p & 0 & 0 & 0 & 0 & 0 & 0 & 0\ 0\ 0\ 0 \\ 0 & 0 & 0 & 0 & \frac{(I_z-I_y)r}{I_x} & 0 & 0 & 0 & 0\ 0\ 0\ 0 \\ 0 & 0 & 0 & 0 & 0 & \frac{-(I_x-I_z)p}{I_y} & 0 & 0 & 0\ 0\ 0\ 0 \\ 0 & 0 & 0 & \frac{(I_x-I_y)q}{I_z} & 0 & 0 & 0 & 0 & 0\ 0\ 0\ 0 \\ 0 & 0 & 0 & 1 & T(\theta)S(\phi) & T(\theta)C(\phi) & 0 & 0 & 0\ 0\ 0\ 0 \\ 0 & 0 & 0 & 0 & C(\phi) & -S(\phi) & 0 & 0 & 0\ 0\ 0\ 0 \\ 0 & 0 & 0 & 0 & \frac{S(\phi)}{C(\theta)} & \frac{C(\phi)}{C(\theta)} & 0 & 0 & 0\ 0\ 0\ 0 \\ a_1 & a_2 & a_3 & 0 & 0 & 0 & 0 & 0 & 0\ 0\ 0\ 0 \\ a_4 & a_5 & a_6 & 0 & 0 & 0 & 0 & 0 & 0\ 0\ 0\ 0 \\ -S(\theta) & S(\phi)C(\theta) & C(\phi)C(\theta) & 0 & 0 & 0 & 0 & 0 & 0\ 0\ 0\ 0 \end{bmatrix} \tag{4.8}$$

and

$$B(x) = \begin{bmatrix} 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ \frac{1}{m} & \frac{1}{m} & \frac{1}{m} & \frac{1}{m} \\ 0 & \frac{-d}{I_x} & 0 & \frac{d}{I_x} \\ \frac{d}{I_y} & 0 & \frac{-d}{I_y} & 0 \\ \frac{k}{I_z} & \frac{-k}{I_z} & \frac{k}{I_z} & \frac{-k}{I_z} \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \end{bmatrix} \tag{4.9}$$

Where $C(\cdot)$, $S(\cdot)$ and $T(\cdot)$ denote $Cos(\cdot)$, $Sin(\cdot)$ and $Tan(\cdot)$. $a_1 = C(\theta)C(\phi)$, $a_2 = -C(\phi)S(\psi) + S(\phi)S(\theta)C(\psi)$, $a_3 = S(\phi)S(\psi) + C(\phi)S(\theta)C(\psi)$, $a_4 = C(\theta)S(\psi)$, $a_5 = C(\phi)C(\psi) + S(\phi)S(\theta)S(\psi)$, $a_6 = -S(\phi)C(\psi) + C(\phi)S(\theta)S(\psi)$, g is the gravity, m is the mass of the system, I_x, I_y and I_z are the principal moments of inertia of the quadrotor, d is the distance from the CG of the quadrotor to the motors, and k is the torsional constant of the motors.

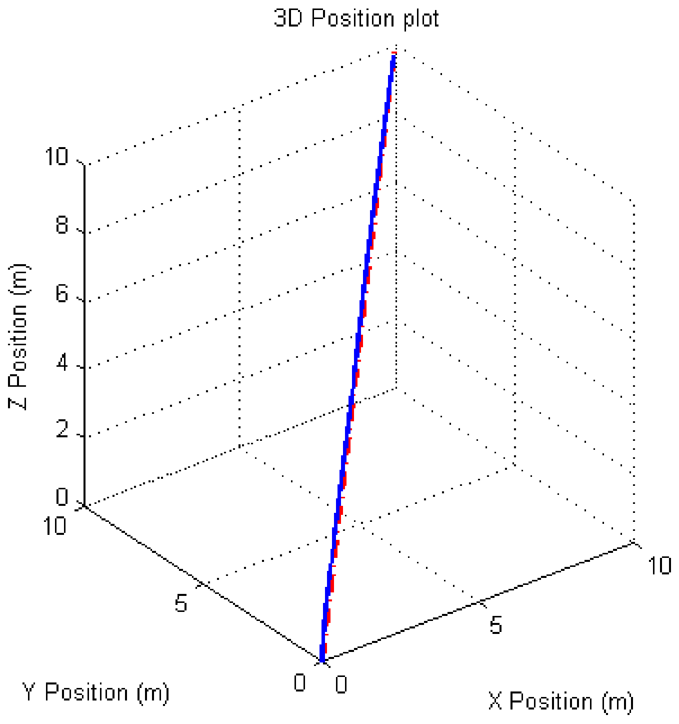
These SDC form matrices are used in the simulation of the four different scenarios described in Chapter 4.The simulation information for each scenario is presented in table 4.1 and the summary of results is given in table 4.2 and figures (4.2) to (4.9).

Table 4.1. SDRE Scenario information

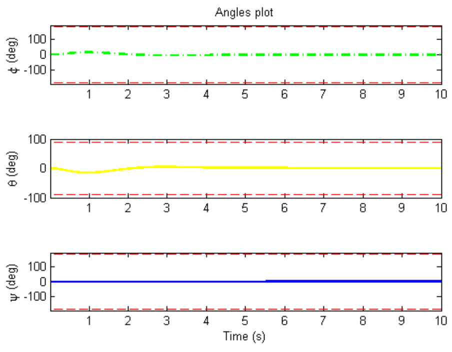
Scenario	Ts	simulation time
Smooth	0.01 s	10 s
Helicoidal	0.01 s	10 s
Straight Lines	0.01 s	20 s
Spherical Spiral	0.01 s	20 s

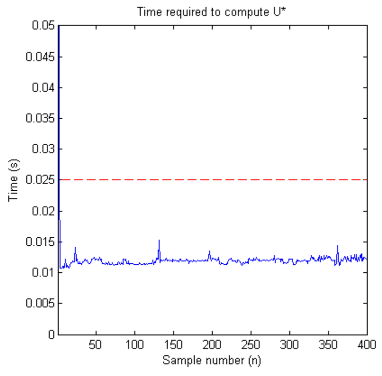
Table 4.2. SDRE RMS of the position error results

Scenario	X RMS	Y RMS	Z RMS	Average RMS
Smooth	1.8916 m	1.8898 m	1.4606 m	1.7473 m
Helicoidal	2.3423 m	2.1470 m	2.7304 m	2.4066 m
Straight Lines	0.6418 m	0.6441 m	0.4972 m	0.5944 m
Spherical Spiral	1.5530 m	1.3305 m	1.1539 m	1.3458 m



(a) 3D position

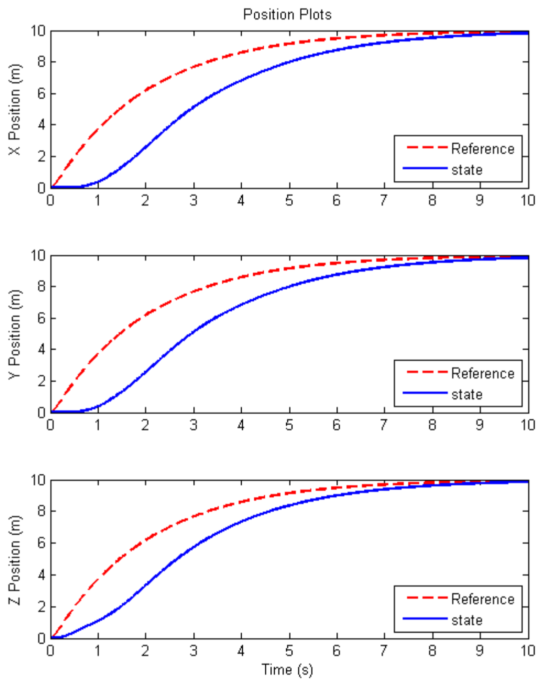




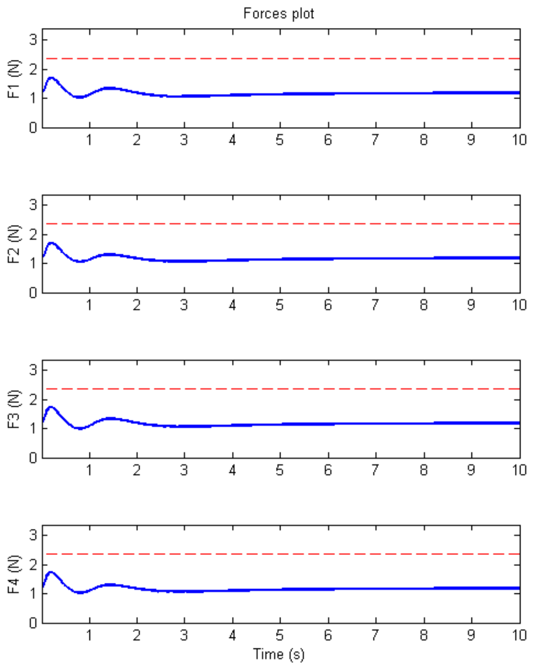
(b) Angular position

(c) Computational time

Figure 4.2. SDRE Scenario 1: Smooth trajectory.

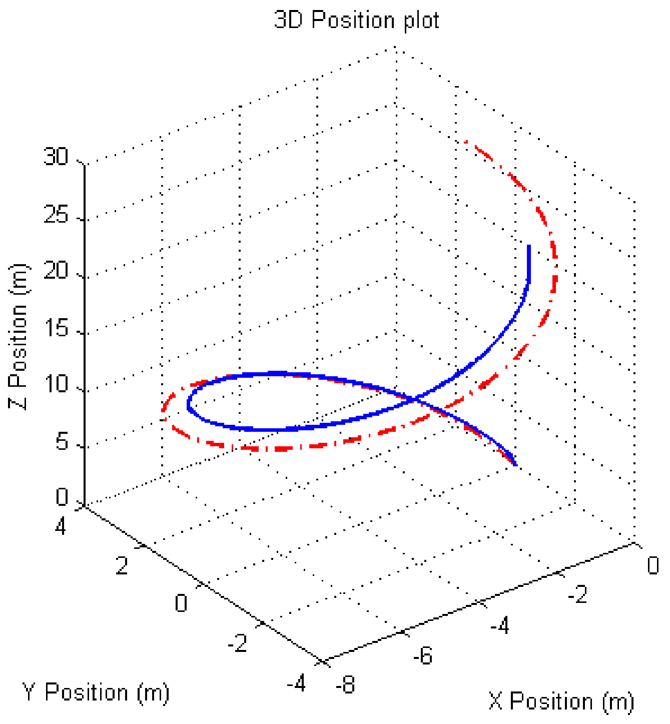


(a) Position

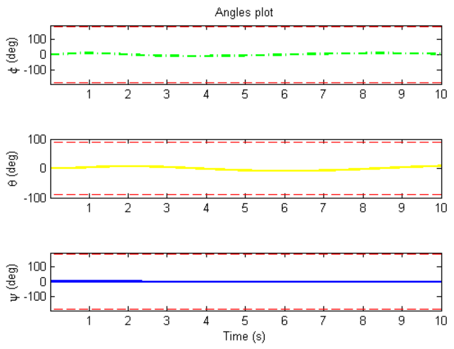


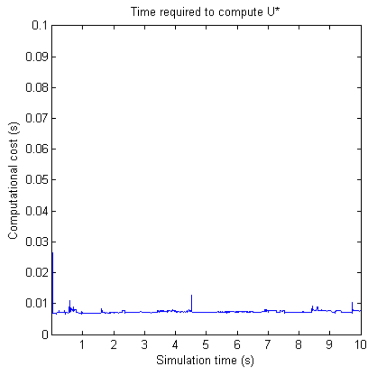
(b) Forces

Figure 4.3. SDRE Scenario 1: Position and Forces.



(a) 3D position

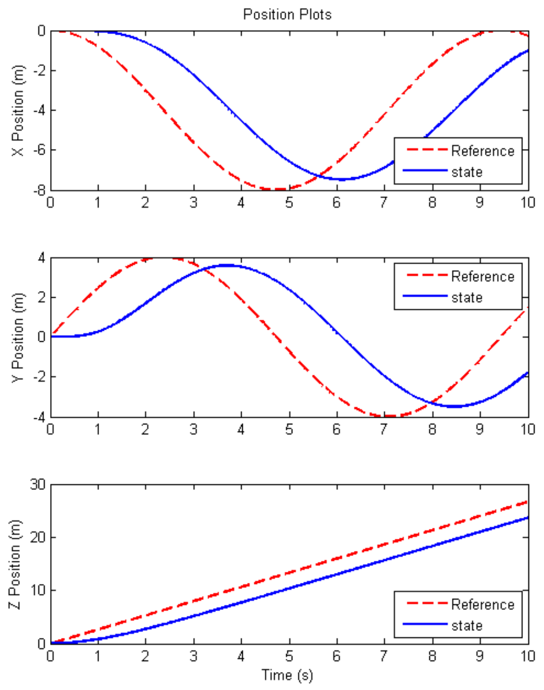




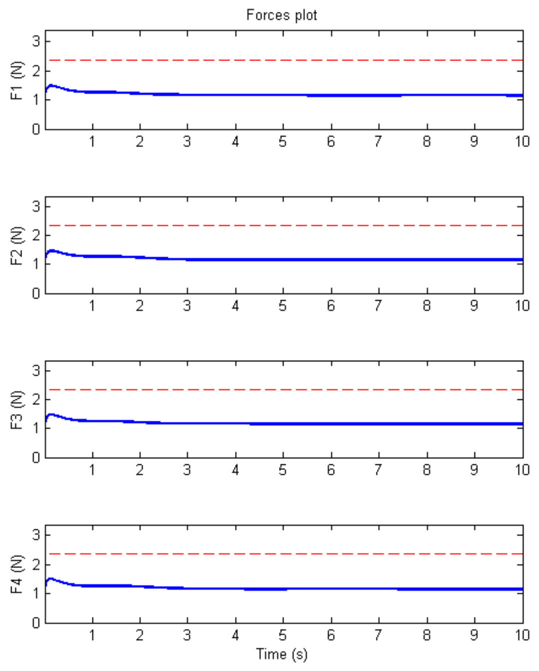
(b) Angular position

(c) Computational time

Figure 4.4. SDRE Scenario 2: Helicoidal trajectory.

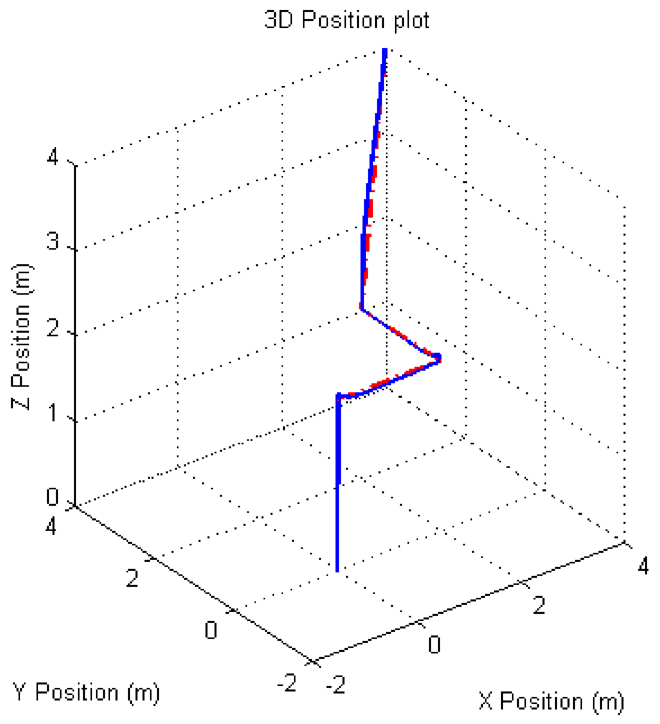


(a) Position

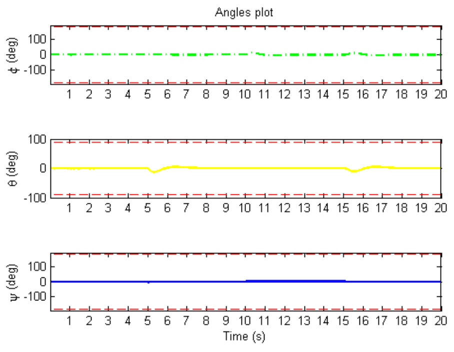


(b) Forces

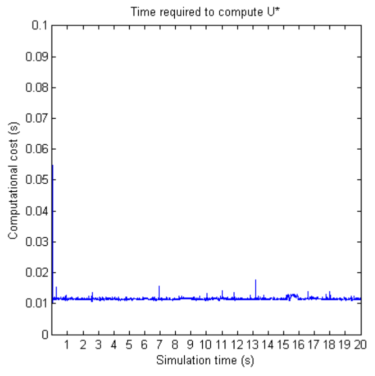
Figure 4.5. SDRE Scenario 2: Position and Forces.



(a) 3D position

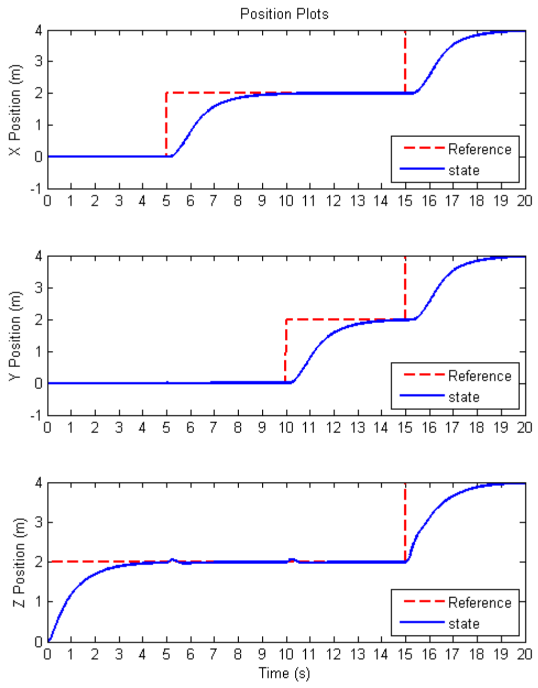


(b) Angular position

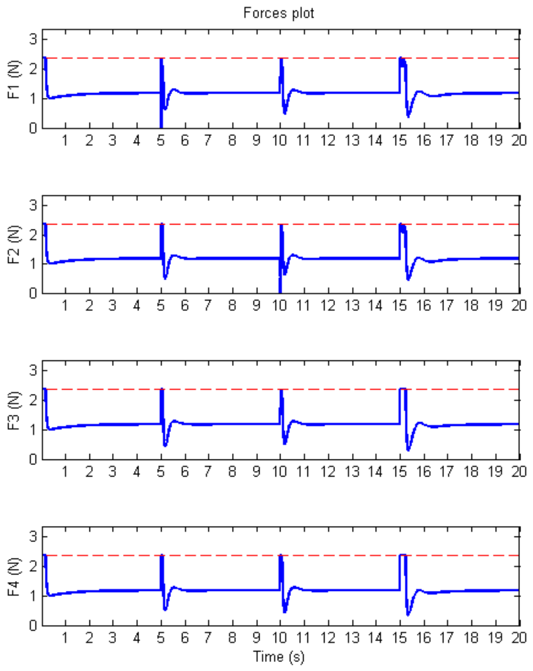


(c) Computational time

Figure 4.6. SDRE Scenario 3: Straight lines trajectory.

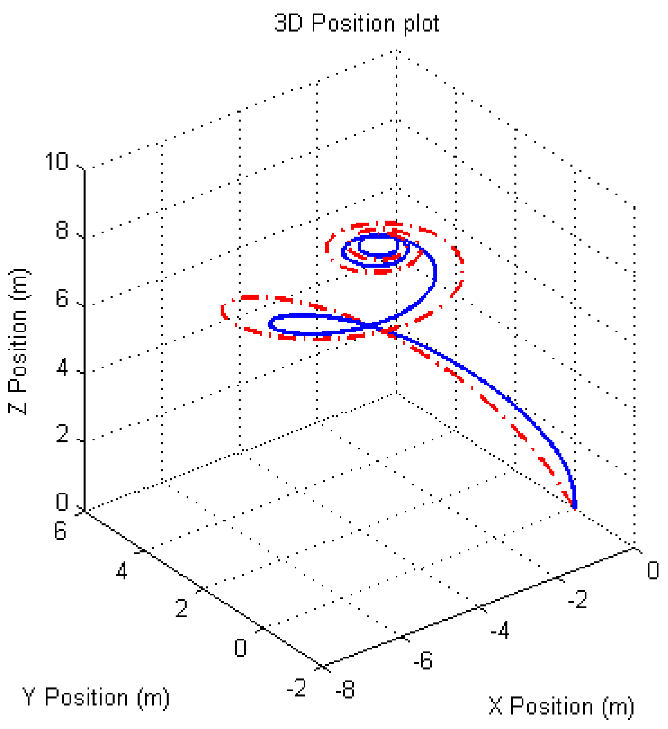


(a) Position

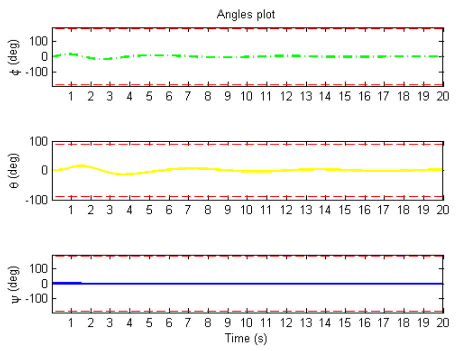


(b) Forces

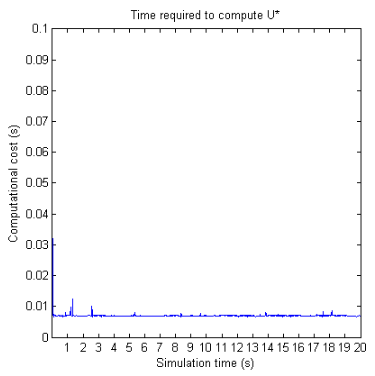
Figure 4.7. SDRE Scenario 3: Position and Forces.



(a) 3D position

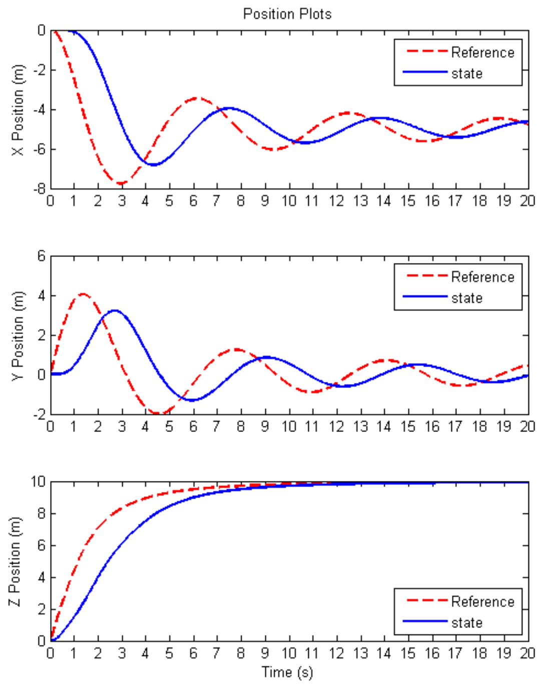


(b) Angular position

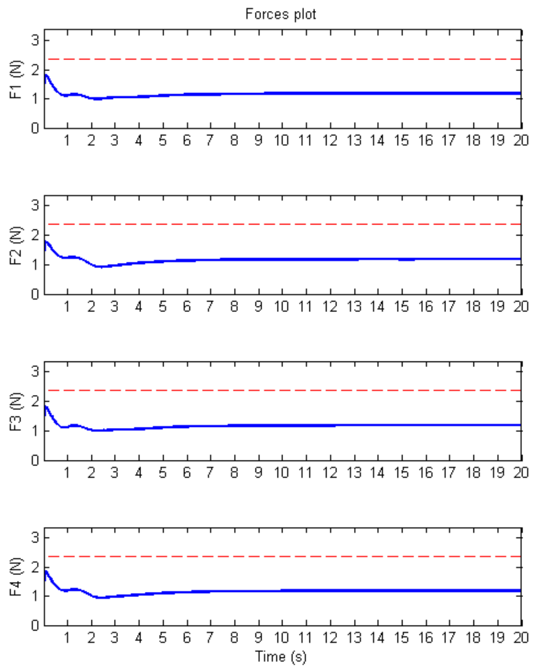


(c) Computational time

Figure 4.8. SDRE Scenario 4: Spherical spiral trajectory.



(a) Position



(b) Forces

Figure 4.9. SDRE Scenario 4: Position and Forces.

CHAPTER 5

The Nonlinear Model Predictive Control

5.1 Introduction

The idea behind creating a nonlinear controller is that, since many systems are inherently nonlinear, it should be possible to formulate a type of controller that takes into account such nonlinearities and outperforms a linear type of controller, which are sometimes inadequate to describe some dynamical processes properly [36].

5.2 NMPC Problem Formulation

Consider a nonlinear system of the form

$$\dot{x} = f(x(t), u(t)) \tag{5.1}$$

Where $\mathbf{x(t)} \in \Re^n$ are the system states and $\mathbf{u(t)} \in \Re^m$ are the system inputs.

The formulation for the Nonlinear Model Predictive Control is very similar to the linear formulation discussed previously on Chapter 4. The difference relies in that the $\mathbf{A}$ and $\mathbf{B}$ matrices of equation 3.2 are going to be expressed into the State Dependent Coefficient (SDC) form discussed in Chapter 5.

By using the SDC form, the $\mathbf{A(x)}$ and $\mathbf{B(x)}$ matrices are now calculated at each sampling interval and then used to compute a new $\bar{\mathbf{A}}(\mathbf{x})$ and $\bar{\mathbf{B}}(\mathbf{x})$, which now depend on the current states, allowing the representation of the system in the following pseudo-linear form

$$\bar{X}_k = \bar{A}(x)\hat{x}_{k+1} + \bar{B}(x)\bar{U}_k \tag{5.2}$$

$$\bar{Y}_k = \bar{C}(x)\hat{x}_{k+1} + \bar{D}(x)\bar{U}_k \tag{5.3}$$

Where

$$\bar{A}(x) = \begin{bmatrix} A(x) \\ A(x)^2 \\ A(x)^3 \\ \vdots \\ A(x)^N \end{bmatrix} \qquad \bar{B}(x) = \begin{bmatrix} B(x) & 0 & ... & 0 \\ A(x)B(x) & B(x) & ... & 0 \\ A(x)^2B(x) & A(x)B(x) & ... & 0 \\ \vdots & \vdots & \ddots & \vdots \\ A(x)^{N-1}B(x) & A(x)^{N-2}B(x) & ... & B(x) \end{bmatrix}$$

$$\bar{C}(x) = \begin{bmatrix} C \\ CA(x) \\ CA(x)^2 \\ \vdots \\ CA(x)^{N-1} \end{bmatrix} \qquad \bar{D}(x) = \begin{bmatrix} D & 0 & ... & 0 \\ CB(x) & D & ... & 0 \\ CA(x)B(x) & CB(x) & ... & 0 \\ \vdots & \vdots & \ddots & \vdots \\ CA(x)^{N-2}B(x) & CA(x)^{N-3}B(x) & ... & D \end{bmatrix}$$

5.3 Controller Design

The nonlinear MPC uses the same algorithm as the linear MPC to calculate the optimal solution to the optimization problem of equation (3.14). The main difference between the linear and nonlinear version of the MPC is that, now, our objective function depends on the current states of the system and needs to be calculated at each sample interval.

The objective function then becomes

$$J(\hat{x}_{k+1}, U_k) = \tfrac{1}{2}U_k^T H(x) U_k + U_t^T f(x) \tag{5.4}$$

$$H(x) = \bar{D}(x)^T \bar{Q} \bar{D}(x) + \bar{R} \tag{5.5}$$

$$f(x) = \left[\bar{D}(x)^T \bar{Q} \bar{C}(x) - \bar{D}(x)^T \bar{Q}\right] \begin{bmatrix} \hat{x}_{k+1} \\ \bar{Y}_d \end{bmatrix} - \begin{bmatrix} Ru_k \\ 0 \\ \vdots \\ 0 \end{bmatrix} \tag{5.6}$$

where $H(x)$ and $f(x)$ are now state dependent.

The cost function can then be considered to be quasi-linear, and the regular quadratic programming optimization method can be used to solve the optimization problem (equation 5.4).

In summary, the Nonlinear Model Predictive Control algorithm can be represented with the following flow chart

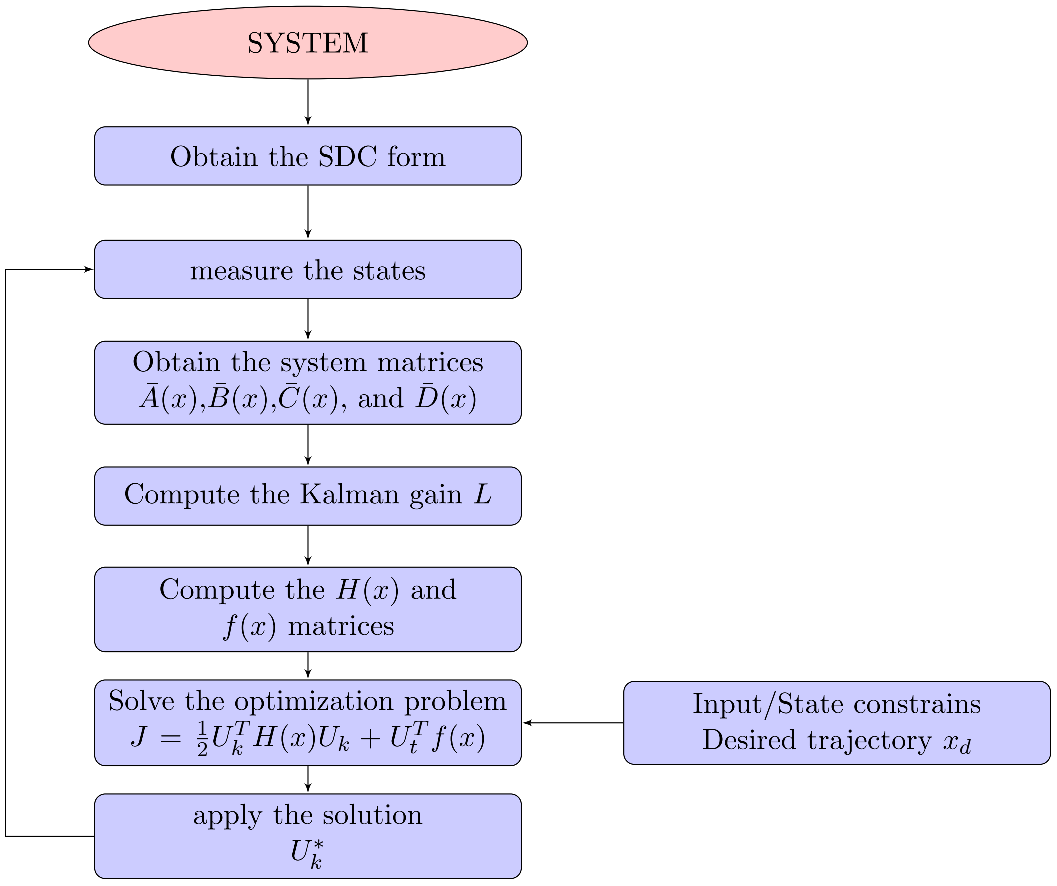

SYSTEM
Obtain the SDC form
measure the states
Obtain the system matrices $\bar{A}(x)$,$\bar{B}(x)$,$\bar{C}(x)$, and $\bar{D}(x)$
Compute the Kalman gain L
Compute the $H(x)$ and $f(x)$ matrices
Solve the optimization problem $J = \frac{1}{2}U_k^T H(x)U_k + U_t^T f(x)$
Input/State constrains Desired trajectory x_d
apply the solution U_k^*

Figure 5.1. NMPC Algorithm Flow Chart.

5.4 Results

Using the previously defined SDC form system matrices $\mathbf{A}(\mathbf{x})$ and $\mathbf{B}(\mathbf{x})$ from equation (4.8), the nonlinear MPC algorithm performance can be evaluated by using the four scenarios defined in Section 4 results. The simulation information for each scenario is the following

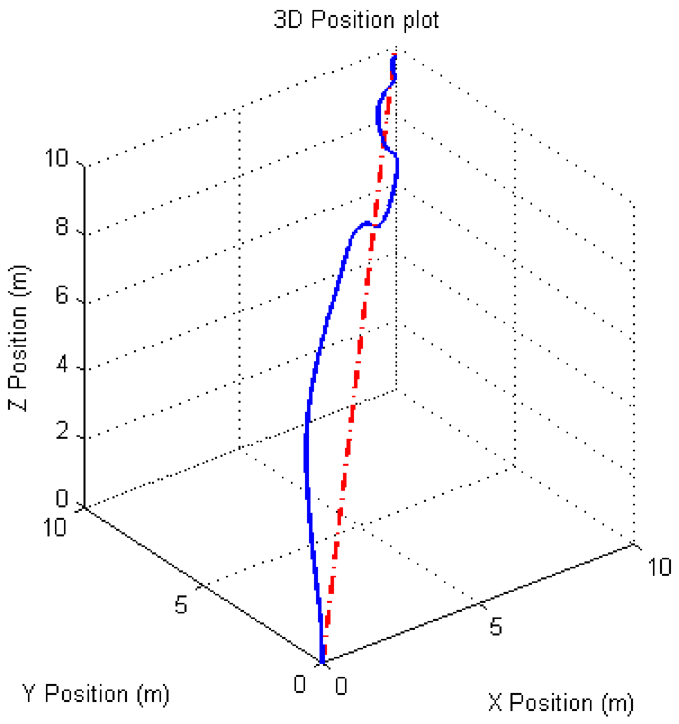
Table 5.1. NMPC Scenario information

Scenario	Ts	N	M	simulation time
Smooth	0.01 s	40	25	10 s
Helicoidal	0.01 s	40	25	10 s
Straight Lines	0.01 s	40	25	20 s
Spherical Spiral	0.01 s	40	25	20 s

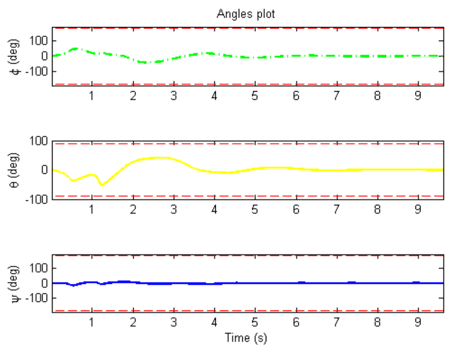
And the summary of results for each scenario:

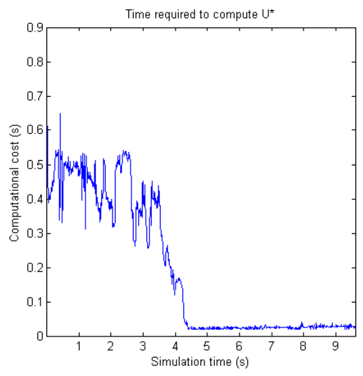
Table 5.2. NMPC RMS of the position error results

Scenario	X RMS	Y RMS	Z RMS	Average RMS
Smooth	1.0632 m	0.9317 m	0.4905 m	0.8285 m
Helicoidal	0.3296 m	0.5430 m	0.2768 m	0.3831 m
Straight Lines	0.4583 m	0.4500 m	0.3736 m	0.4273 m
Spherical Spiral	1.6234 m	0.8325 m	0.5090 m	0.9883 m



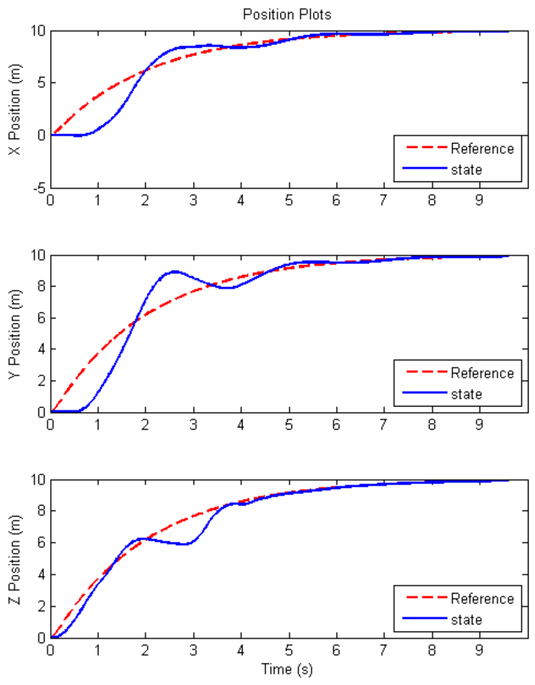
(a) 3D position



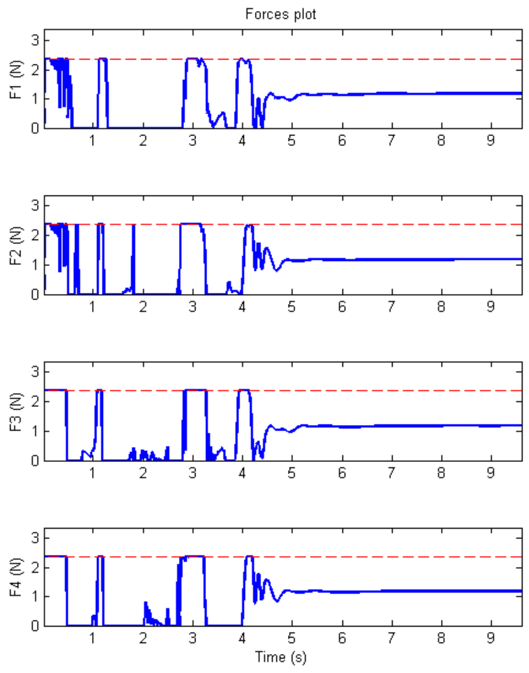


(b) Angular position (c) Computational time

Figure 5.2. NMPC Scenario 1: Smooth trajectory.

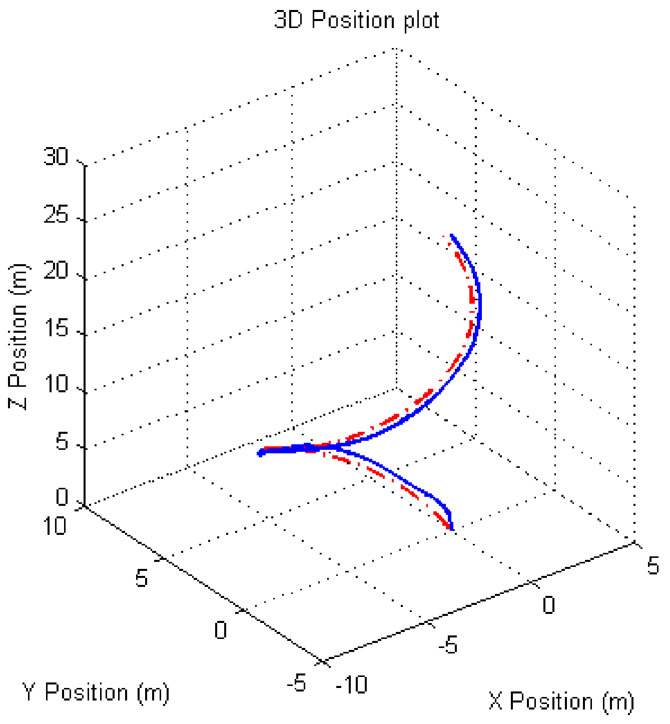


(a) Position

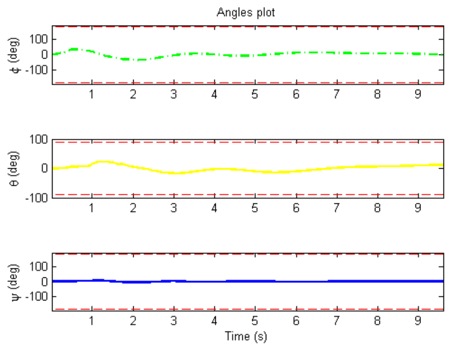


(b) Forces

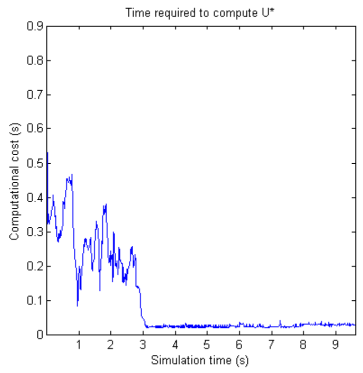
Figure 5.3. NMPC Scenario 1: Position and Forces.



(a) 3D position

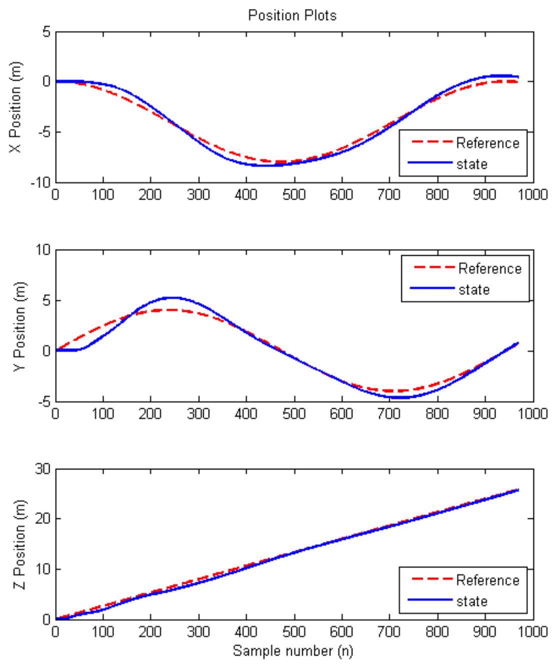


(b) Angular position

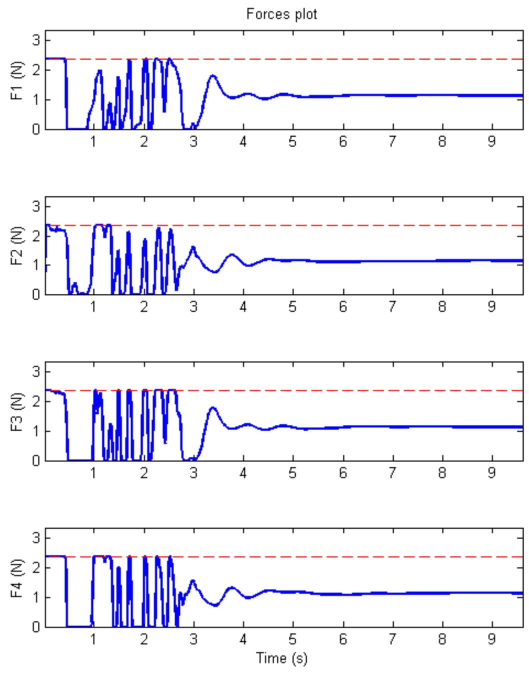


(c) Computational time

Figure 5.4. NMPC Scenario 2: Helicoidal trajectory.

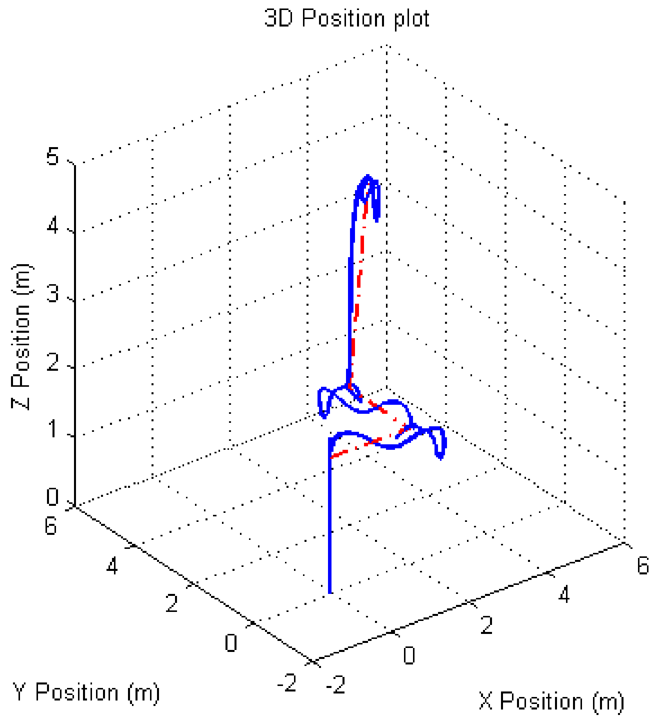


(a) Position

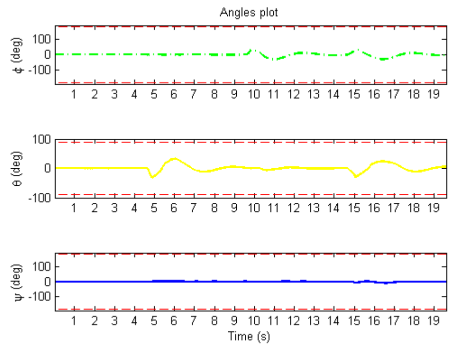


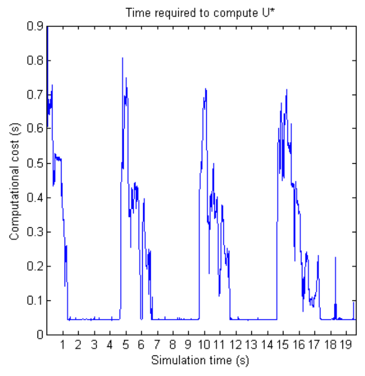
(b) Forces

Figure 5.5. NMPC Scenario 2: Position and Forces.



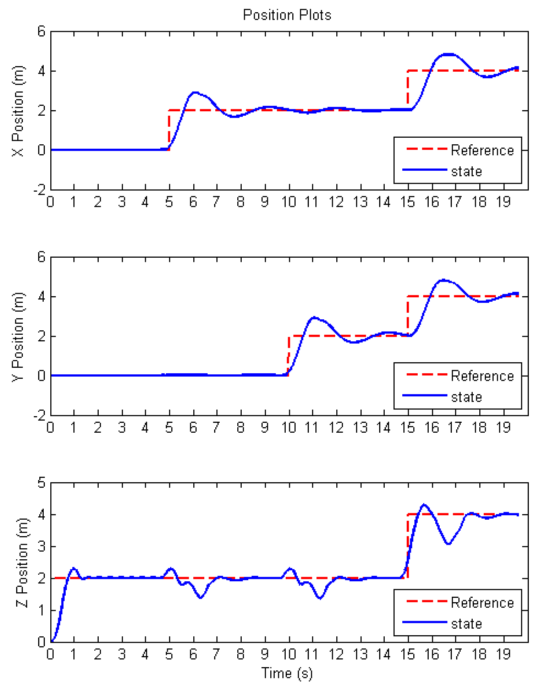
(a) 3D position



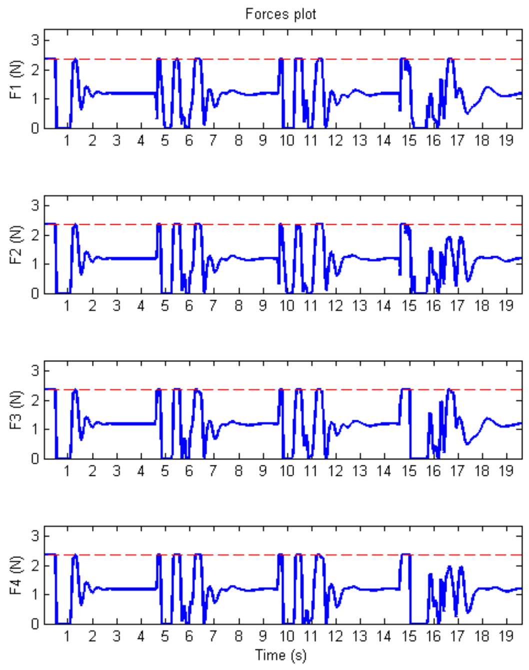


(b) Angular position (c) Computational time

Figure 5.6. NMPC Scenario 3: Straight lines trajectory.

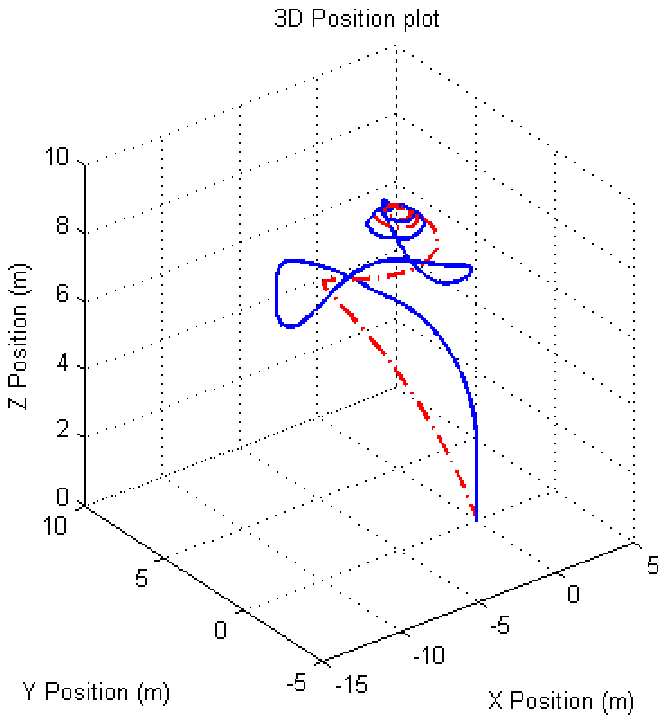


(a) Position

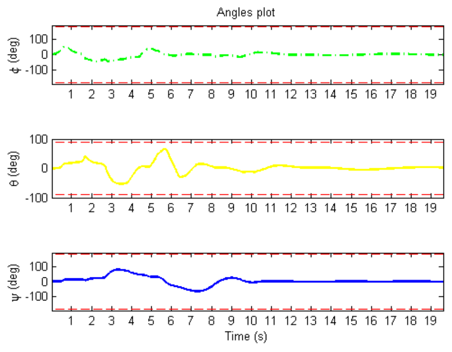


(b) Forces

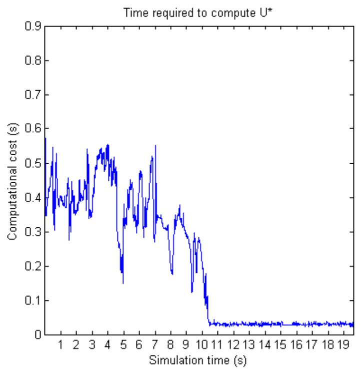
Figure 5.7. NMPC Scenario 3: Position and Forces.



(a) 3D position

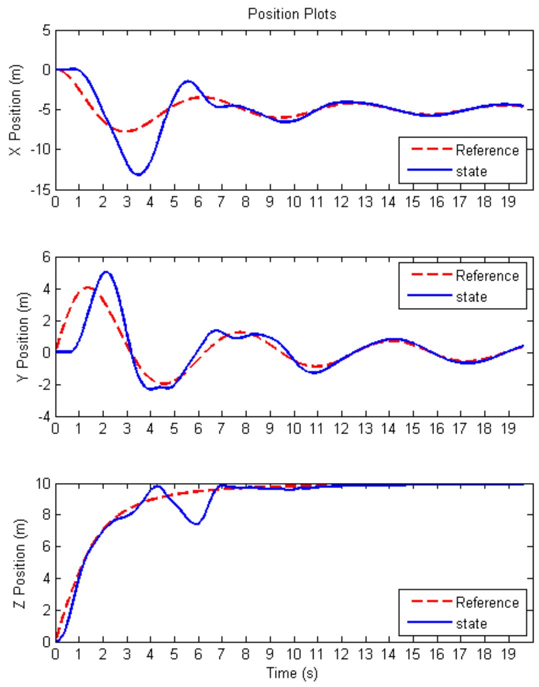


(b) Angular position

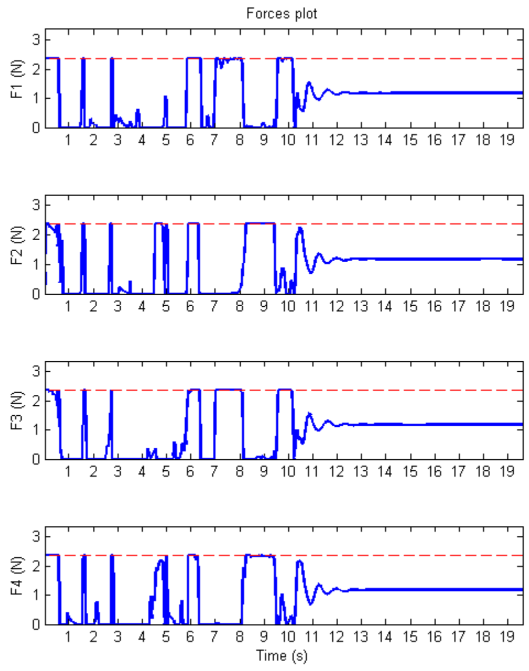


(c) Computational time

Figure 5.8. NMPC Scenario 4: Spherical Spiral.



(a) Position



(b) Forces

Figure 5.9. NMPC Scenario 4: Position and Forces.

CHAPTER 6

COMBINATION OF THE NMPC AND SDRE APPROACHES

6.1 Introduction

The goal of this approach was to combine both the State Dependent Riccati Equation (SDRE) and the Nonlinear Model Predictive Control (NMPC) in order to obtain a more robust [37], optimal control response of the system. The SDRE approach would provide a faster and robust response of the system, while the MPC would provide an optimal solution over a specified prediction horizon.

6.2 Formulation

The procedure consists on obtaining the system matrices $\mathbf{A}(x)$,$\mathbf{B}(x)$, $\mathbf{C}(x)$ and $\mathbf{D}(x)$ in the SDC form from the nonlinear set of equations, and use them to compute the solution to the minimization problem of the NMPC of equation (5.4), and the solution of equation (4.7) from the SDRE problem formulation.

Both solutions must then be combined into one single solution. This is done by multiplying the optimal solution from the NMPC by α, and the solution to the SDRE problem by a factor of $(1-\alpha)$, so that

$$U^* = \alpha U^*_{NMPC} + (1-\alpha)U^*_{SDRE} \tag{6.1}$$

Where U^*_{NMPC} is the optimal solution to the NMPC optimization problem, U^*_{SDRE} is the optimal solution to the SDRE, and U^* is the total force to be applied to the system.

The algorithm to compute this procedure can be represented with the following flow chart

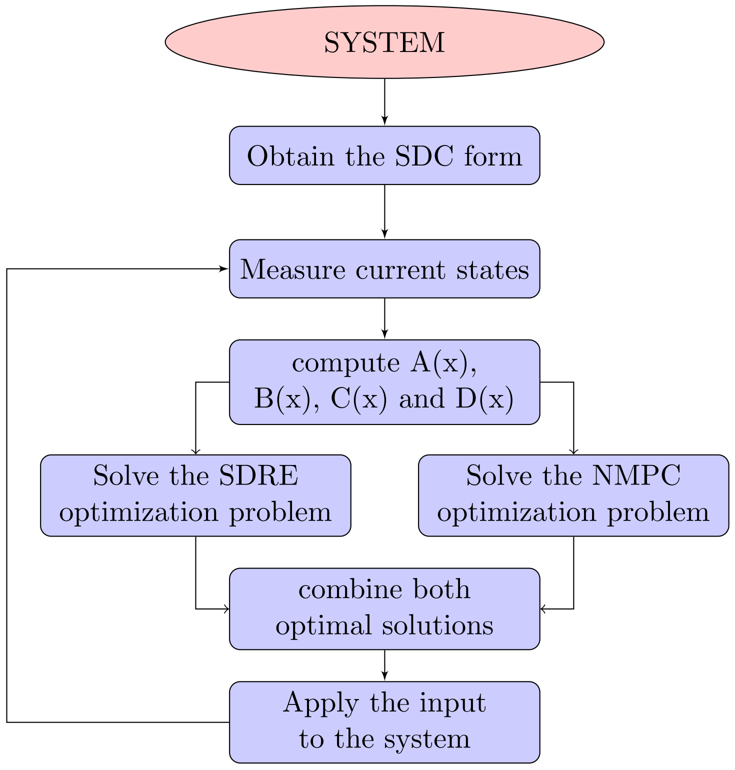

SYSTEM
Obtain the SDC form
Measure current states
compute A(x), B(x), C(x) and D(x)
Solve the SDRE optimization problem
Solve the NMPC optimization problem
combine both optimal solutions
Apply the input to the system

Figure 6.1. SDRE and NMPC combination Flow Chart.

And the block diagram representation of the system is depicted in figure (6.2).

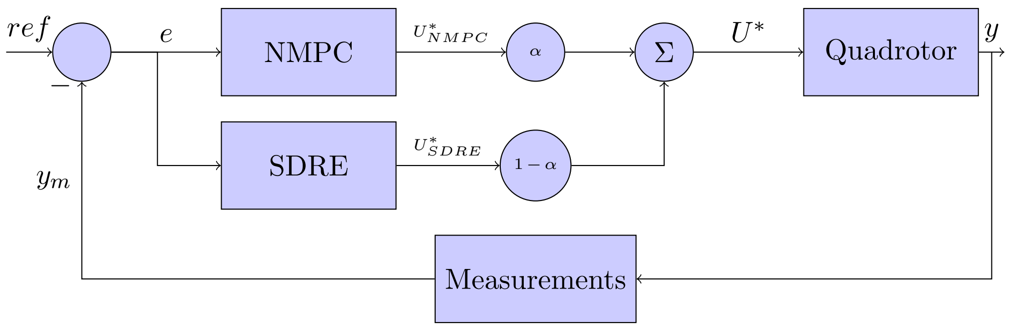


Figure 6.2. SDRE and NMPC block diagram.

6.3 Results

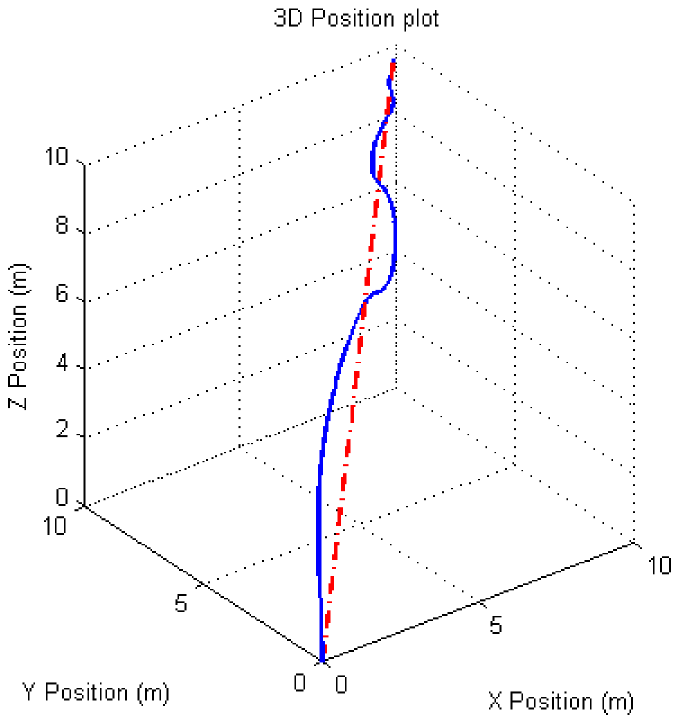
Using again the previously defined SDC form system matrices $\mathbf{A}(\mathbf{x})$ and $\mathbf{B}(\mathbf{x})$ from equation 4.8, it is now desired to evaluate the performance of the combination of both the SDRE and the NMPC algorithms. The same four scenarios are considered again in order to provide a comparison between the combined system, and each one working independently. As it will later be shown, the designed spherical spiral trajectory still continues to be a challenge for the quadrotor. Table 6.1 provides the simulation information and the simulation results are presented in table 6.2. It can be seen that the designed spherical spiral trajectory still continues to be a challenge for the quadrotor.

Table 6.1. Combined control scenario information

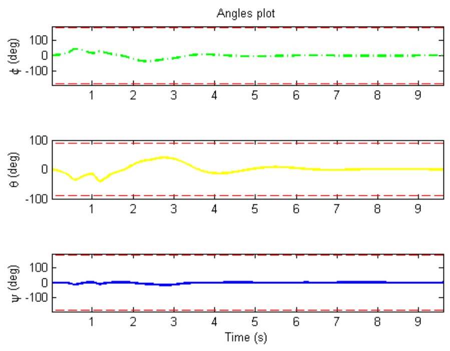
Scenario	Ts	N	M	α	simulation time
Smooth	0.01 s	40	25	0.90	10 s
Helicoidal	0.01 s	40	25	0.75	10 s
Straight Lines	0.01 s	40	25	0.90	20 s
Spherical Spiral	0.01 s	40	25	0.95	20 s

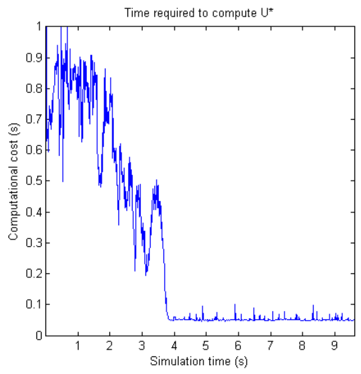
Table 6.2. Combined control RMS of the position error results

Scenario	X RMS	Y RMS	Z RMS	Average RMS
Smooth	0.8677 m	0.7090 m	0.4798 m	0.6855 m
Helicoidal	0.3417 m	0.5544 m	0.2998 m	0.3986 m
Straight Lines	0.4475 m	0.4345 m	0.3429 m	0.4083 m
Spherical Spiral	1.6785 m	1.4725 m	0.5334 m	1.2281 m



(a) 3D position

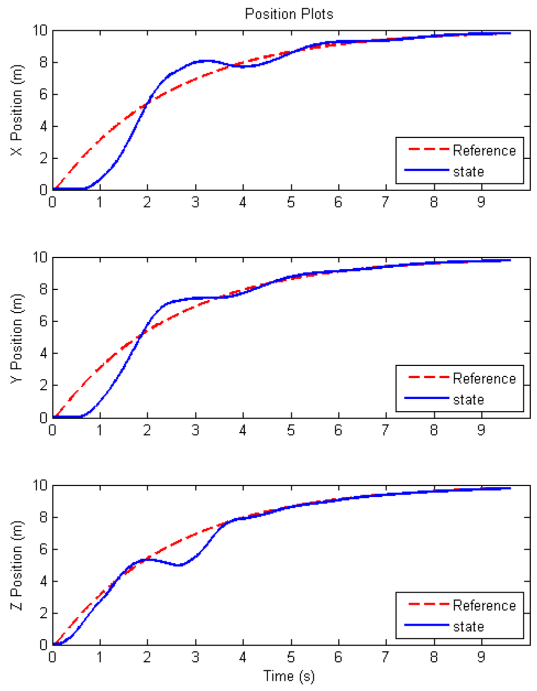




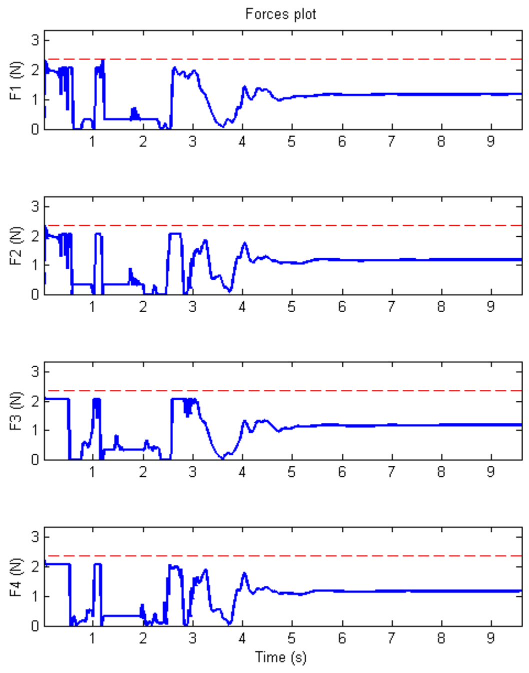
(b) Angular position

(c) Computational time

Figure 6.3. Combined Scenario 1: Smooth trajectory.

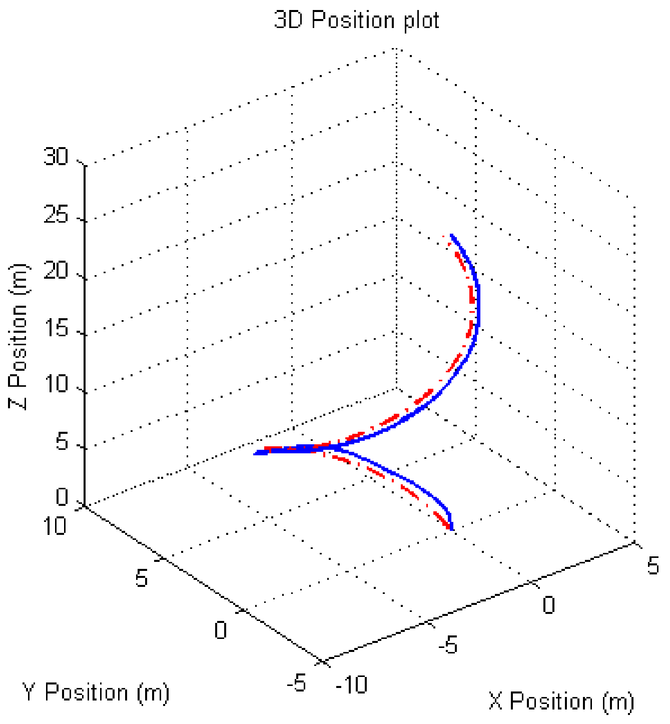


(a) Position

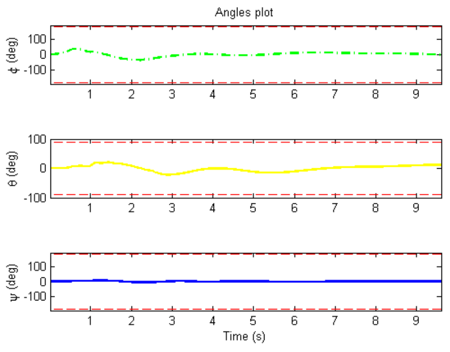


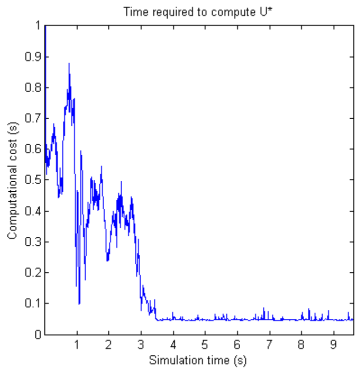
(b) Forces

Figure 6.4. Combined Scenario 1: Position and Forces.



(a) 3D position

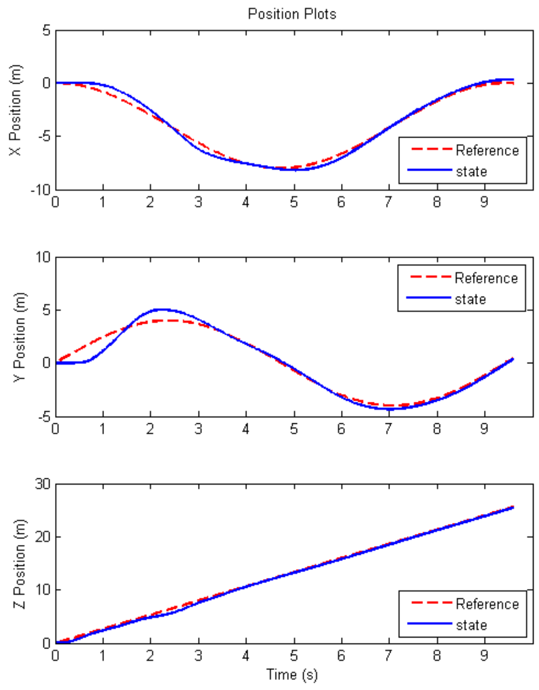




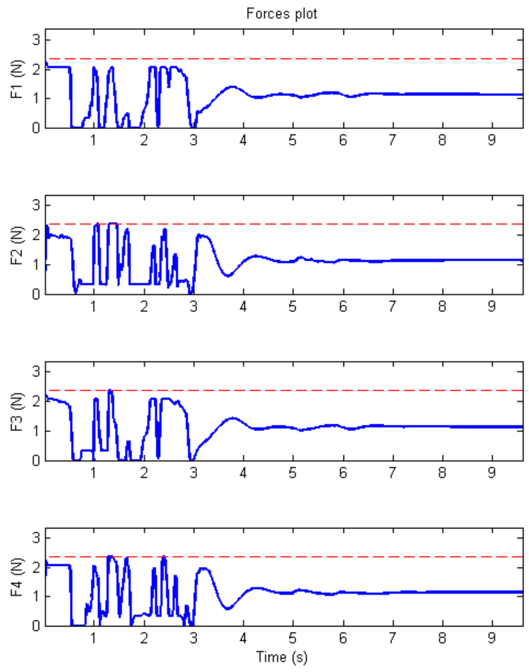
(b) Angular position

(c) Computational time

Figure 6.5. Combined Scenario 2: Helicoidal trajectory.

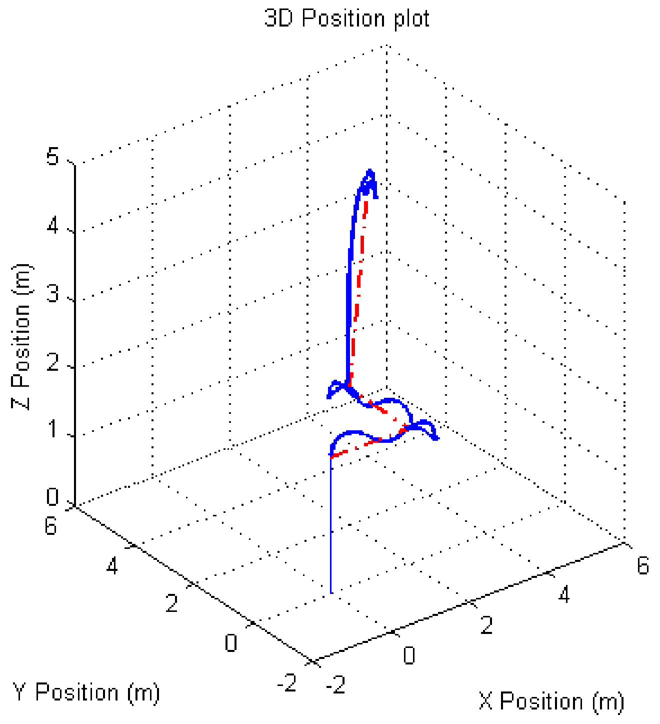


(a) Position

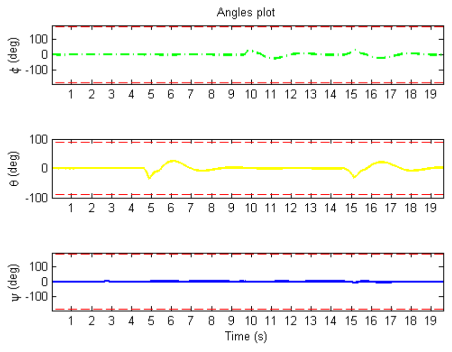


(b) Forces

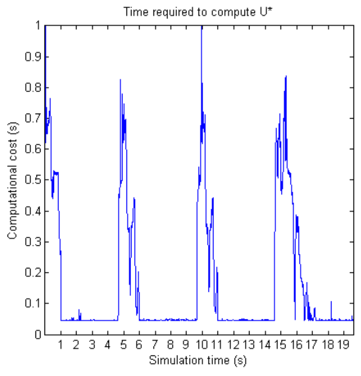
Figure 6.6. Combined Scenario 2: Position and Forces.



(a) 3D position

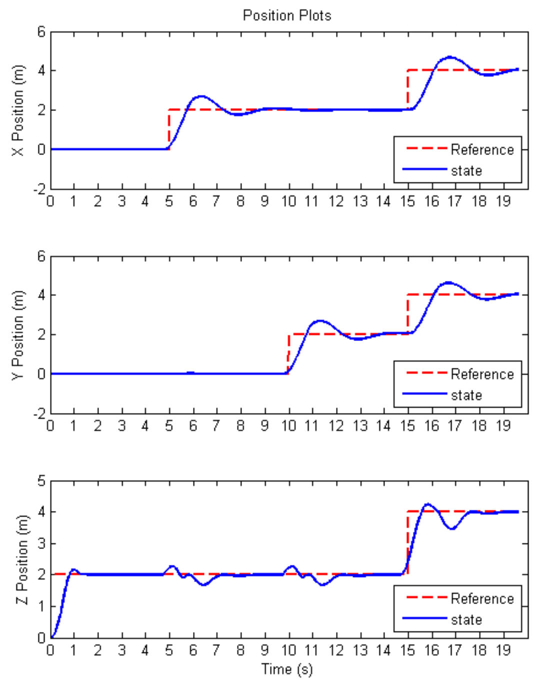


(b) Angular position

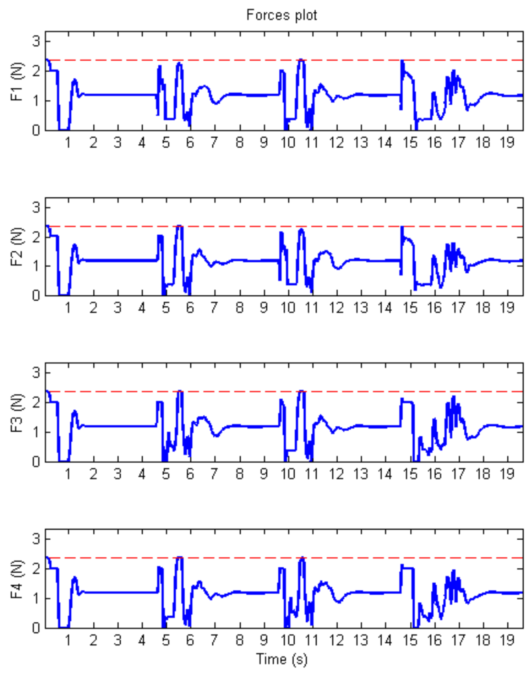


(c) Computational time

Figure 6.7. Combined Scenario 3: Straight lines trajectory.

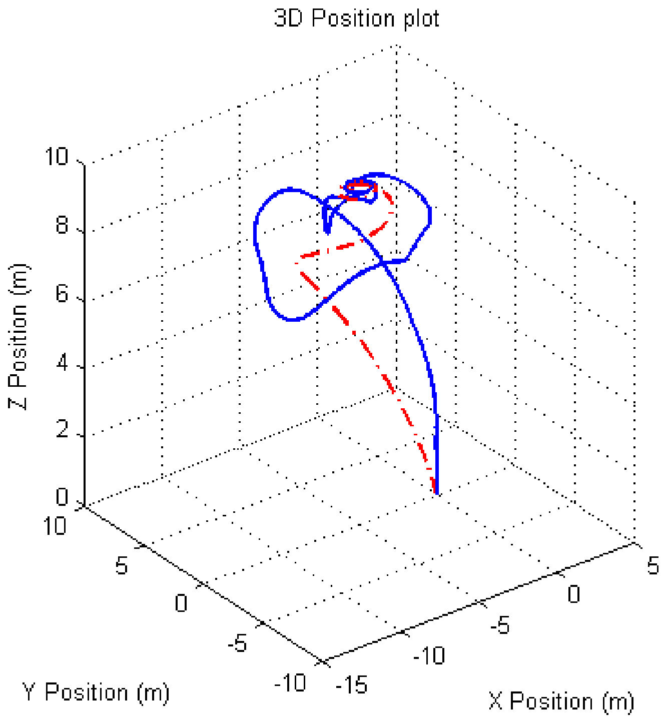


(a) Position

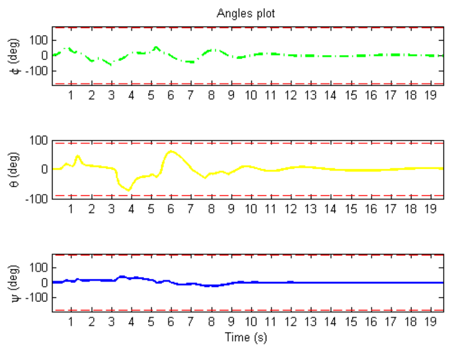


(b) Forces

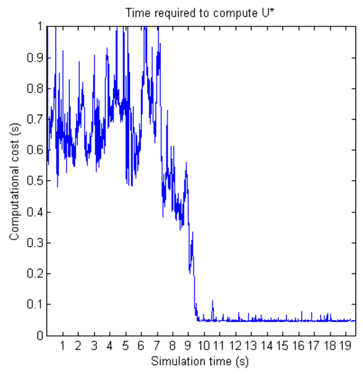
Figure 6.8. Combined Scenario 3: Position and Forces.



(a) 3D position

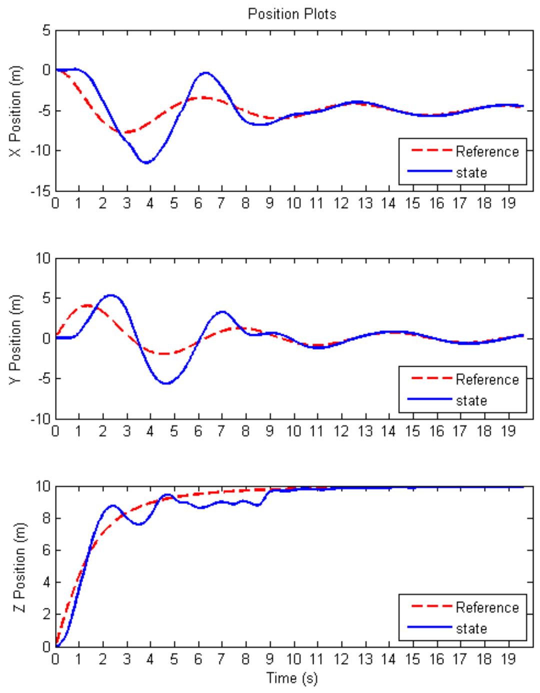


(b) Angular position

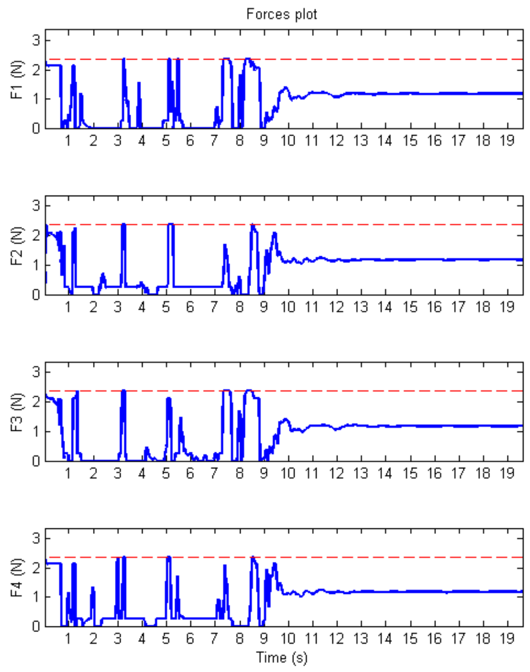


(c) Computational time

Figure 6.9. Combined Scenario 4: Spherical spiral trajectory.



(a) Position



(b) Forces

Figure 6.10. Combined Scenario 4: Position and Forces.

CHAPTER 7

CONCLUDING REMARKS

7.1 Summary of Results

This research discussed different methods of generating trajectories for a quadrotor helicopter platform. For the linear case, the six degree of freedom equations of motion were linearized and implemented into the MPC algorithm, which optimized the actuator effort based on the constrains imposed. For the nonlinear case, the SDC form was used to capture the system nonlinearities into a pseudo-linear system matrix. This allows the use of nonlinear techniques like SDRE, and even allows to implement them in the MPC algorithm to create a nonlinear version of it.

the quadrotor parameters are given in table 7.1, while the various parameters used in the simulation are presented in table 7.2. Q and R are the weight matrices, N and M are the prediction and control horizon, and Ts is the sampling time.

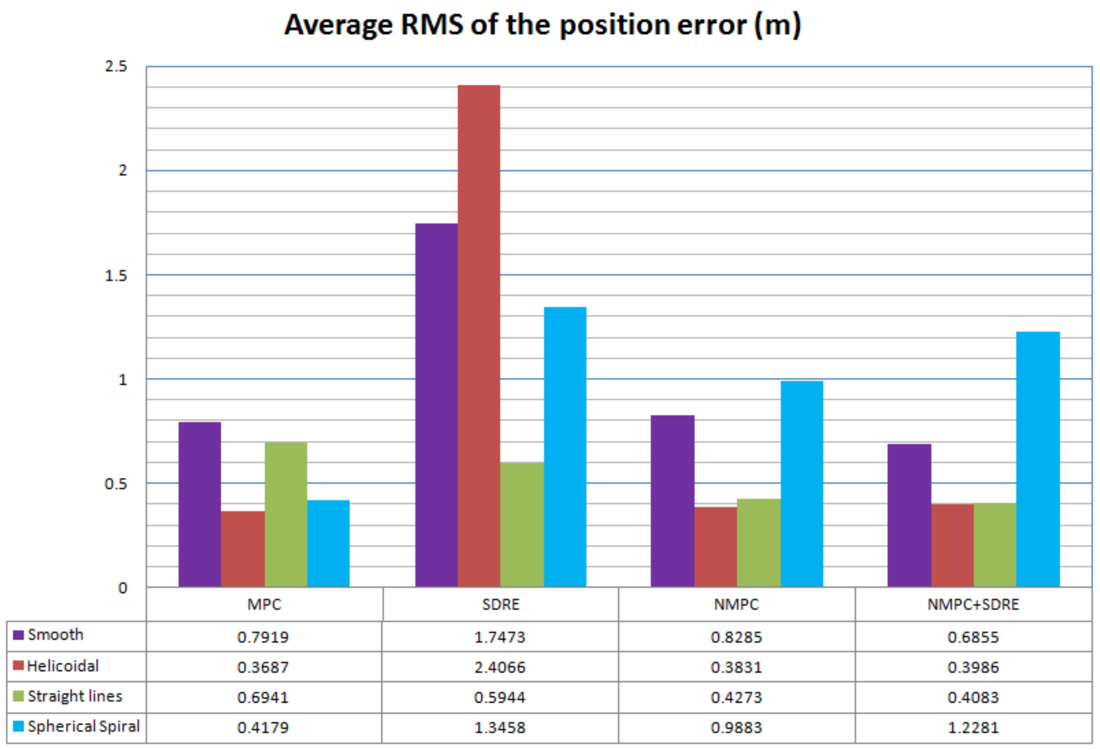
Table 7.1. Quadrotor Parameters

Mass (kg)	0.482
Ix (kgm^2)	0.0066
Iy (kgm^2)	0.0066
Iz (kgm^2)	0.0131
Length (m)	0.53
Width (m)	0.53
Height (m)	0.15

Table 7.2. Simulation Information

		Q	R	N	M	Ts (s)	Sim Length (s)
MPC	Smooth	1	0.05	40	25	0.01	10
	Helicoidal	1	0.05	40	25	0.01	10
	Straight Lines	1	0.01	40	25	0.01	20
	Spherical Spiral	1	0.05	40	25	0.01	20
SDRE	Smooth	30	1	-	-	0.01	10
	Helicoidal	0.35	0.01	-	-	0.01	10
	Straight Lines	100	3	-	-	0.01	20
	Spherical Spiral	100	0.1	-	-	0.01	20
NMPC	Smooth	100	0.1	40	25	0.01	10
	Helicoidal	100	0.5	40	25	0.01	10
	Straight Lines	100	1	40	25	0.01	20
	Spherical Spiral	20	0.1	40	25	0.01	20
NMPC+SDRE	Smooth	100	0.1	40	25	0.01	10
	Helicoidal	100	0.5	40	25	0.01	10
	Straight Lines	100	1	40	25	0.01	20
	Spherical Spiral	20	0.1	40	25	0.01	20

The average root mean square (RMS) of the position error of the four control algorithms are shown in Figure (7.1). Even though the RMS of the position error are within acceptable limits, it can be seen that, with exception for the SDRE controller, the MPC, NMPC and NMPC and SDRE combination performed similarly for a fixed *N* and *M*. It could also be possible to improve the performance of the MPC and NMPC algorithm by tailoring the *N* and *M* accordingly for a specific scenario, like figures (3.2) to (3.5). By doing so it is possible to reduce the computational time required to run the optimization algorithm and have a similar, or better tracking response.



	MPC	SDRE	NMPC	NMPC+SDRE
🟪 Smooth	0.7919	1.7473	0.8285	0.6855
🟥 Helicoidal	0.3687	2.4066	0.3831	0.3986
🟩 Straight lines	0.6941	0.5944	0.4273	0.4083
🟦 Spherical Spiral	0.4179	1.3458	0.9883	1.2281

Figure 7.1. Average RMS of the position error.

Overall, the SDC proved to be an useful way to capture the model nonlinearities and derive an optimal controller. It can be seen from Figure (7.1) that out of the nonlinear techniques, the SDRE is the one with the biggest RMS. This is because it tries to keep the actuator effort in its minimum. The combination of the NMPC and SDRE seems to be a very feasible controller. It combined the fast reaction from the NMPC and the smooth response of the SDRE to provide overall better tracking, with the exception of the spiral trajectory. Furthermore, by reducing M, it is possible to make the NMPC respond more aggressively and provide a faster response, thus increasing the performance of the control algorithm.

7.2 Future Work

There are some aspects that could benefit from additional work. Including some rotor dynamics could improve the overall performance of the system. Currently, the simulation is for the case of an indoor flight, but in case it is desired to fly the quadrotor in an outside environment a wind/gust model should be implemented.

Another interesting approach could be the use of differential flatness to model the system dynamics and develop feasible trajectories. In addition, the time to compute the nonlinear MPC is higher than the required sampling time; another method could be researched to develop a nonlinear MPC and see how it behaves compared to the SDC approach used in this research.

REFERENCES

[1] H. Huang, G. Hoffmann, S. Waslander, and C. Tomlin, "Aerodynamics and control of autonomous quadrotor helicopters in aggressive maneuvering," in *Robotics and Automation, 2009. ICRA '09. IEEE International Conference on*, may 2009, pp. 3277 –3282.

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

[3] I. Dikmen, A. Arisoy, and H. Temeltas, "Attitude control of a quadrotor," in *Recent Advances in Space Technologies, 2009. RAST '09. 4th International Conference on*, june 2009, pp. 722 –727.

[4] (2012, September) Spaxels / klangwolke - quadrocopter. [Online]. Available: http://www.aec.at/futurelab/de/referenzen/kategorie/kunst-am-bau/spaxels-klangwolken-quadrocopter/

[5] A. de Luca, L. Lanari, and G. Oriolo, "A sensitivity approach to optimal spline robot trajectories," *Automatica*, vol. 27, no. 3, pp. 535 – 539, 1991. [Online]. Available: http://www.sciencedirect.com/science/article/pii/000510989190111E

[6] A. de Luca and G. Oriolo, "Trajectory planning and control for planar robots with passive last joint," *The International Journal of Robotics Research*, vol. 21, no. 5-6, pp. 575–590, 2002.

[7] Y. Bouktir, M. Haddad, and T. Chettibi, "Trajectory planning for a quadrotor helicopter," in *Control and Automation, 2008 16th Mediterranean Conference on*, june 2008, pp. 1258 –1263.

[8] D. Mellinger and V. Kumar, "Minimum snap trajectory generation and control for quadrotors," in *Proceedings of the IEEE International Conference on Robotics and Automation (ICRA)*, May 2011.

[9] M. van Nieuwstadt and R. M. Murray, "Real time trajectory generation for differentially flat systems," *Int. Journal of Robust and Nonlinear Control*, vol. 8, pp. 995–1020, 1996.

[10] D. Mellinger, N. Michael, and V. Kumar, "Trajectory generation and control for precise aggressive maneuvers with quadrotors," in *Proceedings of the International Symposium on Experimental Robotics*, Dec 2010.

[11] H. Huang, G. Hoffmann, S. Waslander, and C. Tomlin, "Aerodynamics and control of autonomous quadrotor helicopters in aggressive maneuvering," in *Robotics and Automation, 2009. ICRA '09. IEEE International Conference on*, may 2009, pp. 3277 –3282.

[12] J. Gillula, H. Huang, M. Vitus, and C. Tomlin, "Design of guaranteed safe maneuvers using reachable sets: Autonomous quadrotor aerobatics in theory and practice," in *Robotics and Automation (ICRA), 2010 IEEE International Conference on*, may 2010, pp. 1649 –1654.

[13] G. M. Hoffmann, S. L. Wasl, and C. J. Tomlin, "Quadrotor helicopter trajectory tracking control," in *In Proc. AIAA Guidance, Navigation, and Control Conf*, 2008.

[14] 2011.

[15] M. Hehn, R. Ritz, and R. D'Andrea, "Performance benchmarking of quadrotor systems using time-optimal control," *Autonomous Robots*, vol. 33, pp. 69–88, 2012. [Online]. Available: http://dx.doi.org/10.1007/s10514-012-9282-3

[16] I. Palunko, R. Fierro, and P. Cruz, "Trajectory generation for swing-free maneuvers of a quadrotor with suspended payload: A dynamic programming approach," in *ICRA*, 2012, pp. 2691–2697.

[17] H. Kim, D. Shim, and S. Sastry, "Nonlinear model predictive tracking control for rotorcraft-based unmanned aerial vehicles," in *American Control Conference, 2002. Proceedings of the 2002*, vol. 5, 2002, pp. 3576 – 3581 vol.5.

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

[19] J. M. B. Domingues, "Quadrotor prototype," Master's thesis, Universidade Técnica de Lisboa.

[20] W. F. Phillips, *Mechanics of flight.* Wiley, 2010.

[21] *Model Predictive Control applied to tracking and attitude stabilization of a VTOL quadrotor aircraft*, COBEM. 21st International Congress of Mechanical Engineering, october 2011.

[22] D. Mayne, J. Rawlings, C. Rao, and P. Scokaert, "Constrained model predictive control: Stability and optimality," *Automatica*, vol. 36, no. 6, pp. 789 – 814, 2000.

[23] K. Alexis, G. Nikolakopoulos, and A. Tzes, "Model predictive control scheme for the autonomous flight of an unmanned quadrotor," in *Industrial Electronics (ISIE), 2011 IEEE International Symposium on*, june 2011, pp. 2243 –2248.

[24] R. Findeisen, T. Raff, and F. Allgower, "Sampled-data model predictive control for constrained continuous time systems," Advanced Strategies in Control Systems with Input and Output Constraints, pp. 207–235, 2007.

[25] A. Wills and W. P. Heath, "Application of barrier function model predictive control to an edible oil refining process," *Journal of Process Control*, vol. 15, no. 2, pp. 183–200, mar 2005.

[26] E. F. Camacho and C. Bordons, *Model Predictive Control*. Springer, 1999.

[27] A. Wills, "Technical report ee04025 - notes on linear model predictive control," Department of Electrical and Computer Engineering, University of Newcastle, Australia, Tech. Rep. EE04025, jan 2004.

[28] L. V. Stephen Boyd, *Convex Optimization*. Cambridge University Press, 2004.

[29] C. E. Garcia, J. H. Lee, D. Prett, and M. Moravi, *Model Predictive Control*. Prentice Hall PTR, 2004.

[30] T. Çimen, "State-Dependent Riccati Equation (SDRE) Control: A Survey," in *Proceedings of the 17th IFAC World Congress*, 2008, pp. 3761–3775. [Online]. Available: http://www.ifac-papersonline.net/Detailed/36352.html

[31] A. B. Magnus, E. 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," 2003.

[32] A. Dutka, A. Ordys, and M. Grimble, "Optimized discrete-time state dependent riccati equation regulator," in *American Control Conference, 2005. Proceedings of the 2005*, june 2005, pp. 2293 – 2298 vol. 4.

[33] A. A. Bogdanov, E. 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.

[34] 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.

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

[36] F. Allgower, R. Findeisen, and C. Ebenbauer, "Nonlinear model predictive control," in *Encyclopedia of Life Support Systems (EOLSS), Developed under the Auspices of the UNESCO, Eolss Publishers, Oxford ,UK*, vol. XI.

[37] E. A. Wan and A. A. Bogdanov, "Combining state-dependent riccati equation (sdre) methods with model predictive neural control (mpnc)."

BIOGRAPHICAL STATEMENT

Carlos Tule was born in H. Matamoros, Tamaulipas, Mexico in 1986. He obtained his Bachelor's degree in Mechatronics Engineering from the Instituto Tecnológico y de Estudios Superiores de Monterrey, Monterrey, Mexico in 2009. In 2010 he came to the United States to pursue a M.Sc. degree program in Aerospace Engineering at The University of Texas at Arlington.