ISTANBUL TECHNICAL UNIVERSITY ★ GRADUATE SCHOOL

TRAJECTORY TRACKING CONTROL OF QUADROTOR UAV WITH LPV APPROACH MPC CONTROLLER

M.Sc. THESIS

Mücahit DEMİRCİOĞLU

Department of Aeronautical and Astronautical Engineering

Aeronautical and Astronautical Engineering Programme

JANUARY 2023

ISTANBUL TECHNICAL UNIVERSITY ★ GRADUATE SCHOOL

TRAJECTORY TRACKING CONTROL OF QUADROTORUAV WITH LPV APPROACH MPC CONTROLLER

M.Sc. THESIS

Mücahit DEMİRCİOĞLU
(511191212)

Department of Aeronautical and Astronautical Engineering

Aeronautical and Astronautical Engineering Programme

Thesis Advisor: Prof. Dr. Elbrus M. JAFEROV

JANUARY 2023

İSTANBUL TEKNİK ÜNİVERSİTESİ ★ LİSANSÜSTÜ EĞİTİM ENSTİTÜSÜ

DÖRT KANATLI BİR İHA'NIN LPV YAKLAŞIM MPC KONTROL İLE YÖRÜNGE TAKİP KONTROLÜ

YÜKSEK LİSANS TEZİ

Mücahit DEMİRCİOĞLU
(511191212)

Uçak ve Uzay Mühendisliği Anabilim Dalı

Uçak ve Uzay Mühendisliği Programı

Tez Danışmanı: Prof. Dr. Elbrus M. JAFEROV

HAZİRAN 2023

Mücahit Demircioğlu, a M.Sc. student of İTU Graduate School student ID 511191212, successfully defended the thesis entitled “TRAJECTORY TRACKING CONTROL OF QUADROTOR UAV WITH LPV APPROACH MPC CONTROLLER”, which he prepared after fulfilling the requirements specified in the associated legislations, before the jury whose signatures are below.

Thesis Advisor:	**Prof. Dr. Elbrus M. JAFEROV** Istanbul Technical University	**……...**
Jury Members:	**Assoc. Dr. Emre KOYUNCU** Istanbul Technical University	**……...**
	Asst. Prof. Cengiz ÖZBEK Istanbul Beykent University	**……...**

Date of Submission : May 25 2023

Date of Defense : June 26 2023

To my family and friends,

FOREWORD

I would like to express my sincere gratitude to all my teachers who supported me especially my thesis supervisor Prof. Dr. Elbrus M. JAFEROV for his support through this study. He always supported me either my lesson phase or thesis phase.

I am also wanted to express my endless gratitude to my family. They supported me through my education. They are always encouraging through my life.

Lastly I am thankful for my great friends. They are always there when I needed them so I am very thankful for them.

June 2023 Mücahit DEMİRCİOĞLU

TABLE OF CONTENTS

Page

ABBREVIATIONS

LPV **:** Linear Parameter-Varying

MPC **:** Model Predictive Control

MATLAB **:** Matrix Laboratory

UAV **:** Unmanned Aerial Vehicle

VTOL **:** Vertical Take-Off

LQR **:** Linear Quadratic Regulator

SYMBOLS

$\boldsymbol{\varphi}$	**:** Pitch Angle
$\boldsymbol{\theta}$	**:** Roll Angle
$\boldsymbol{\psi}$	**:** Yaw Angle
$\boldsymbol{\Omega}$	**:** Motor Rotational Speed
$\boldsymbol{b}$	**:** Thrust Coefficient
$\boldsymbol{l}$	**:** Length Between Propellers to Quadrotor's Center of Mass
$\boldsymbol{U}$	**:** System Inputs
$\boldsymbol{R}$	**:** Rotational Transfer Matrix
$\boldsymbol{T}$	**:** Angular Transfer Matrix
$\boldsymbol{u}$	**:** Body Frame x-direction Velocity
$\boldsymbol{v}$	**:** Body Frame y-direction Velocity
$\boldsymbol{w}$	**:** Body Frame z-direction Velocity
$\boldsymbol{p}$	**:** Body Frame φ-direction Angular Velocity
$\boldsymbol{q}$	**:** Body Frame θ-direction Angular Velocity
$\boldsymbol{r}$	**:** Body Frame ψ-direction Angular Velocity
$\boldsymbol{V}$	**:** Linear Velocity Matrix
$\boldsymbol{W}$	**:** Angular Velocity Matrix
$\boldsymbol{F}$	**:** Force
$\boldsymbol{M}$	**:** Moment
$\boldsymbol{\tau}$	**:** Torque
$\boldsymbol{I}$	**:** Moment of Inertıa
$\boldsymbol{g_a}$	**:** Actuators Gyroscopic Moment
$\boldsymbol{m}$	**:** Mass
$\boldsymbol{g}$	**:** Gravitational Acceleration
$\boldsymbol{J_{tp}}$	**:** Motors Moment of Inertia
$\boldsymbol{T_s}$	**:** Time Sample
$\boldsymbol{J}$	**:** Cost Function

LIST OF TABLES

Page

LIST OF FIGURES

Page

TRAJECTORY TRACKING CONTROL OF QUADROTOR UAV WITH LPV APPROACH MPC CONTROLLER

ABSTRACT

For the last couple of decades, quadrotor control is one of the trending research topics since it has wide areas in use. On the other hand, quadrotors are relatively complex systems since they have six degrees of freedom and only four inputs. Three inputs are pitch, roll, and yaw angle, and one input is the thrust force of the motors. From that point, two control loops have been used for most research about quadrotor control. Also in this research trajectory tracking control of the quadrotor is obtained by two control loops. The position control has been obtained by feedback control, which is responsible for reaching the desired x,y, and z coordinates and speed of the quadrotor from each direction relative to the earth frame. The MPC (Model Predictive Control) achieves attitude control with LPV (Linear Parameter Varying) approach. LPV method is obtained by collecting all of the nonlinear parameters on the state-space equations of the quadrotor to the A matrix. Since MPC control works on the discrete-time from each time sample, we can assume all the variables on the matrix A as constants, and the control algorithm can be handled as a linear system. But the quadrotor model remains nonlinear. For testing this approach on the quadrotor MATLAB model is designed and tested on different trajectories. From each trajectory, the quadrotor catches the trajectory and follows it smoothly. From this research perspective, the LPV-MPC method has good results on trajectory tracking but it needs to be tested on the real-time system to observe CPU usage on the quadrotor.

Key Words: Trajectory Tracking, Quadrotor Control, LPV-MPC, Linear Parameter Varying

DÖRT KANATLI BİR İHA'NIN LPV YAKLAŞIM MPC KONTROL İLE YÖRÜNGE TAKİP KONTROLÜ

ÖZET

Gelişmekte olan teknoloji ile birlikte son birkaç on yılda quadrotorlar ya da daha genel kullanılan ismi ile dronlar oldukça yaygın bir şekilde kullanılmaya başlandı. Bunun başlıca sebebi zaman içerisinde düşen maliyetin ulaşılabilirliği arttırması olarak gösterilebilir. Düşen maliyetlerle birlikte, dronların birçok yeni kullanım alanları ortaya çıkmaya başlamıştır. Bunlardan en sık en etkili kullanım alanı ise sinema dünyası ve savunma teknolojileridir. Her ne kadar savunma teknolojilerinde kullanılan insansız hava araçlarına da drone denede burada bahsi geçen dronlar türlerine göre farklılık göstermektedir. Temelde drone olarak hitap edilen insansız hava araçları üçe ayrılmaktadır. Sabit kanatlı, döner kanatlı ve hibtit insansız hava aracı bu üç türü temsil etmektedir. Bu yapılan tez çalışması temelde döner kanatlı insansız hava aracı olan türü ele almaktadır. Döner kanatlı insansız hava araçları pervane sayısına göre sınıflandırılmaktadır. Burada ele alınan araç dört pervaneli quadrotor olmuştur.

Qaudrotorların düşen maliyet, yazılım ve elektronik alanında kat edilen gelişmeler özellikle kontrol mühendislerini quadrotorlar üzerine çalışmak konusunda teşvik edici olmuştur. Kolay ulaşabilirlik ve kolay test edilebilmeleri başlıca araştırmacılara sağladığı büyük avantajlardan en önemlileridir. Bu nedenle geçen son yıllarda quadrotorlar üzerine bir çok çalışma yapılmıştır. Her ne kadar test etmek kolay olsa da bir quadrotorun kontrolü görece karmaşık bir yapıya sahiptir. Bunun en temek nedeni bir quadrotorun altı serbestlik derecesine sahip olmasına karşın, bunlar x, y, z koordinatları ile yuvarlanma, yunulama ve yalpalama açılarıdır, dört girişe sahiptir, bunlar itki ile yuvarlanma, yunulama ve yalpalama açılarıdır. Bu nedenle yeterince iyi modellenmemiş bir kontrolcü ile quadrotor kontrol etmek olasılık dışıdır.

Quadrotorların bu karmaşık yapısı sebebi ile araştırmacılar geçmişte doğru bir matematiksel model çıkartma konusunda oldukça fazla çalışmalar yapmışlardır. Bu çalışmaların sonucunda literatürde şu anda gerçeğe oldukça yakın pek çok quadrotor matematiksel modeli bulunmaktadır. Bu tez çalışmasında en temelden bir quadrotor

matematiksel denklemlerinin çıkartılışı anlatılmış olsa da bu çıkarımlar literatür araştırmaları sonucunda ve bazı varsayımlar sonucunda elde edilmiştir.

Oluşturulan bu matematiksel model oldukça girift bir yapıya sahiptir. Bu nedenle quadrotor üzerinde kontrol çalışması yapan araştırmacılar olabildiğince tek bir kontrolcü ile tüm sistemin kontrol edilmesinden kaçınırlar. Bunun temel sebebi oluşan bu girift yapı kontrolcünün cevap süresini uzatması ve birbirleriyle bütünleşmiş çalışan sistemler kullanılan aynı tip ve güçteki kontrolcülerin yeterince iyi cevap verememesidir.

Litereatürde tasarlanan quadrotor kontrolcülerinin büyük çoğunluğu temel iki kontrolcü şaması üzerinden yapılmaktadır. Bunlar yükseklik ve tutum kontrolcüleridir. Tutum kontrolcüsü quadrotorun açılarının tayin edilmesinde ve yükseklik kontrolcüsü de yüksekliğin ayarlanmasından sorumludur. Quadrotorun koordinat düzlemindeki x ve y eksenlerinden yapacağı hareketler bunlara bağlı olan yuvarlanama yunulama ve yalpalama açıları aracılığı ile sağlanmaktadır.

Yapılan bu tez çalışmasında da temelde iki adet kontrolcü kullanılarak bir quadrotorun yörünge takip kontrolcüsü tasarlanmıştır. Yükseklik kontrolü bir geri besleme kontrolcüsü ile sağlanırken quadrotorun asıl hareketini belirleyen tutum kontrolcüsü ise Model Öngörülü Kontrol (MPC) ile sağlanmıştır. Model Öngörülü Kontrol temelde çevrimiçi bir optimizasyon kontrolcüsü olması sebebiyle oldukça güçlü bir kontrolcüdür. Fakat doğrusal Model Öngörülü Kontrol matematiksel modeldeki değişimlerin kontrolcü üzerinden adapte edilmemesi sebebiyle hatalı sonuç verebilmektedir. Bu nedenle bu çalışmada Model Öngörülü Kontrol ile tasarlanan kontrolcüye Doğrusal Değişken Parametre(LPV) metodu uygulanmıştır.

Doğrusal Değişken Parametre (LPV) metodu temelde bir yaklaşıma dayanmaktadır. Bu yaklaşım da kontrol edilen sistemdeki tüm doğrusal olmayan denklemlerinin bir matrise toplanacak şekilde durum-uzay modelinin çıkartılması ve bu doğrusal olmayan parametrelerin ayrık zamanda her bir zaman örneklemi arasında doğrusal ve sabit olduğu düşünülmesi bu nedenle de sisteme doğrusal sistem yaklaşımlarının uygulanmasının uygun hale gelmesini dayanmaktadır. Bu yaklaşım oldukça başarılı olmakla birlikte yapılan bu tez çalışmasında simülasyon sonuçlarının mükemmele yakın olduğu görülmüştür.

Model Öngörülü Kontrol temelde iki adet matematiksel model ile çalışmaktadır. Klasik bir kontrol siteminde kontrolcüden elde edilen cevaplar matematiksel modele veya gereç zamanlı bir sisteme iletilir ve modelsen ve ya sistem sensörlerinden alınan cevaplar yeniden kontrolcüye iletilerek bir döngü oluşturulur. Ardından kontrolcü geri beslemelere göre yeni bir giriş sinyali sisteme gönderir ve döngü devam eder. Model Öngörülü Kontrolde ise modelden gelen durum cevapları kontrolcüye iletilir ve kontrolcü kontrol ufku boyunca öngörüde bulunur. Bunu da kendi içerisinde bulunan matematiksel model üzerinde test eder ve elde ettiği sonuçları optimal bir şekilde dengeler. Ardından bu optimal sonuçlardan ilk ayrık zamandakini gerçek sisteme iletir ve döngü devam eder. Her zamana aralığında öngörüde bulunularak en optimal sistem girdisini elde edilmesi oldukça göz alıcı gözükse de beraberinde bazı sorular getirmektedir. Bunların en temeli yüksek işlem gücü gereksinin ve buna bağlı olarak geç sistem cevabıdır

Her ne kadar Doğrusal Değişken Parametre (LPV) metodu ciddi bir avantaj sağlamış olsa da Model Öngörülü Kontrol ’ün (MPC) hantal bir kontrolcü olması sistemlerin yavaş çalışmasına neden olmaktadır. Bu nedenle bu yapılan çalışmada simülasyon sonuçları her ne kadar oldukça iyi gözükse de gerçek bir sistemde elde edilecek sonuçlar buradakilerden ciddi farklılıklar gösterebilir. Bu nedenle bu çalınmanın ileri açası olarak buradaki kontrolcü tasarımının gerçek bir quadrotor üzerinden test edilmesi gerekmektedir.

Anahtar Kelimeler: Yörünge Takibi, Quadrotor Kontrolü, LPV-MPC, Lineer Değiken Parametre, Model Öngörülü Kontrol

1. INTRODUCTION

This thesis's aim is that control a quadrotor UAV for trajectory-tracking purposes. For this purpose, the controller of the quadrotor is split into two parts like other researcher applications in the literature: the position controller and the angular control; also, it can be called position and attitude control. [1,2] In this thesis, the feedback controller achieves the position controller, and the MPC controller executes the angular or attitude control. The main focus of this thesis is to design an attitude control that can be strong enough to follow the desired trajectory as closely as possible. The main reason for choosing the MPC controller is to make a nonlinear control that is both efficient and applicable for real-time application. [3,4] That's why, besides using the MPC controller, the LPV approach is also used on the attitude controller. The LPV approach makes the system more stable for different trajectories, and even though it makes the controller nonlinear, it is easy to apply to the controller. [5,6]

In the literature, there are some examples of LPV-MPC controllers of quadrotors. Still, they mainly focus on the simulation results, which are relatively hard to calculate on a computer. Another focus of this work is making the controller as simple as possible to apply any quadrotor in the future. Even though this controller is not applied to the real quadrotor, all the parameters are used from a real quadrotor. This work will be planned to be applied to an actual commercial quadrotor or a custom quadrotor.

All the simulations in this thesis are made in the MATLAB environment, and some MATLAB libraries are used for driving and optimization problems. For example, for driving the equation "*ode45"* method is used, and for the minimization of the cost function "*quadprog"* method has been used. [7]

The main outline of the thesis can be summarized as; the first section of the thesis introduction of the work. Section 1.1. is focused on general information about UAVs. Section 1.2 is the comparison of each type of UAV. Section 1.3 is focused on the different control strategies for quadrotor control. Section 1.4 is about Quadrotor control inputs and their brief definition; more detailed descriptions will be given in Section 2 and Section 3. Section 2 is focused on the equations of the motions and their

deriving these equations. Section 3 is about controller types used in the quadrotor and optimization strategies. Finally, sections 4 and 5 are simulations, results, and the last comments about this work.

1.1. General Knowledge of UAV

The recent development in control, robotics, and computer technologies led to unmanned air vehicles (UAV) popularity and incrementation. In the last couple of decades, with the increasing popularity of UAV researchers, it has started to interest more and more in the control of the UAV. Another reason for the uprising interest is that the controller of the UAV is easy to test, thanks to MATLAB and other software that allow researchers to experiment. The other reason is that easy to arrange an experiment environment in real life. It is easy to apply the controller on a quadrotor or any other rotary wing UAV, and it is easy to test in real life since they are relatively cheap. Moreover, they are applicable for close-room experiments because most have small structures with slow and precise movements.

There are mainly three types of UAVs exists. These are rotary-wing UAVs, fixed-wing UAVs, and hybrid UAVs. [8] Each has a different kind of application in fields. Therefore, they have some advantages and some disadvantages of each other.

1.1.1. Rotary Wing UAVs

They are the most common UVA types of all. It is also known as drones. They are called rotary wings, but in general look, they don't have any wings. They have motors that generate thrust with rotational movements, called rotary wings. They are also called multi-rotors. The most commonly used ones are four-motored quadrotors, but others have three, four, six, or eight rotors. [9]

Rotary-wing UAVs are common because they are relatively cheap and have precise movements. That's why they are used in many fields. [9,14] They can be used in agriculture, electrical cable failure inspections, delivery, scouting small areas harder for humans to enter, cinematography, medical support, search and rescue, and many more. Rotary-wing UAVs are used in some fields; in others, they are planned to be used. One example of the Rotary-wing UAV is Parrot Mambo Mini drone that is given in Figure 1.1

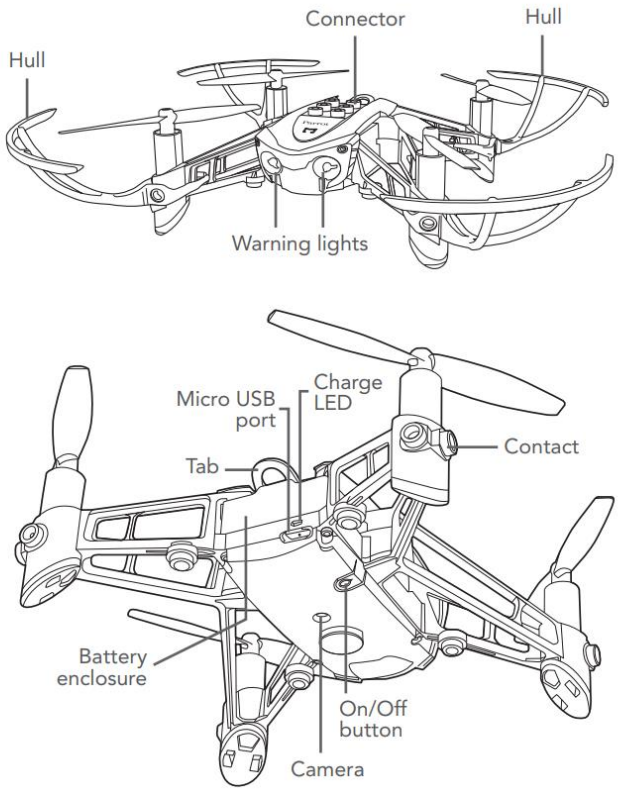

Figure 1.1 Parrot Mambo Mini Drone

The central aspect of rotary-wing UAVs is that they can take off vertically. So that they can take off almost from any place. Also, they are small structures, and precise movement allows them to scout tight areas that are hard to enter. In the literature, vertical take-off is represented as VTOL.

Their precise movements and hovering ability are very crucial for cinematographic applications. In recent movies, directors have been using drones very much.

Despite all the advantages of drones, they have some significant disadvantages. Some of them are;

- They are not very capable of carrying the load. Their payload is minimal.
- Their energy consumption is too high for comparing the other type of UAVs
- Range and endurance of the rotary-wing UAVs are small.
- They have complex mechanisms, so their controller design is also complex.
- They have limited speed to control.

1.1.2. Fixed-Wing UAVs

The fixed-wing UAVs are the second most common UAVs that exist. They have similar aspects as the aircraft. They produce lift force by the vehicle's speed and take off as runway take-off. In the literature, runway take-off is represented as CTOL.

The main differences between aircraft and fixed-wing UAVs are that fixed-wing UAVs are autonomously or remotely controlled. Fixed-wing UAVs are nowadays mainly used in warfare and cartography since both applications are dangerous for humans and take a long time to achieve the desired result. [11] In today's world, almost all Fixed-wing UAVs use a turboprop engine, electrical engine, or piston engine, but almost none use any turbojet motor. [12] But from the development of communication and control technology, more turbojet engine-powered Fixed-wing UAVs will be produced. In Figure 1.2 is one of most commonly known UAV in Turkey Baykar TB2 is given as an example.

Figure 1.2 Baykar TB2 Fixed-Wings UAV [13]

Fixed-wing UAVs are energy efficient and have a long-range endurance compared to rotary-wing UAVs. They are much faster, and they have a very high payload capacity. Also, they are mechanically simple, so they can be assumed to be controlled easily compared to rotary-wing UAVs. [9] But of course, there are some disadvantages to them:

- They need appropriate fields to take off since they cannot vertically take off.
- They cannot hover in the air.
- They are maneuver capability is not high as rotary-wing UAVs'.

1.1.3. Hybrid UAVs

Hybrid UAVs are getting more and more interest nowadays. From the understanding of its name, it is a mixture of fixed-wing and rotary-wing UAVs. They are developed to compensate for all the disadvantages of rotary and fixed-wing UAVs. They can vertically take off, and some can fly like fixed-wing UAVs. In Figure 1.3 there is an example of a hybrid UAV is given.

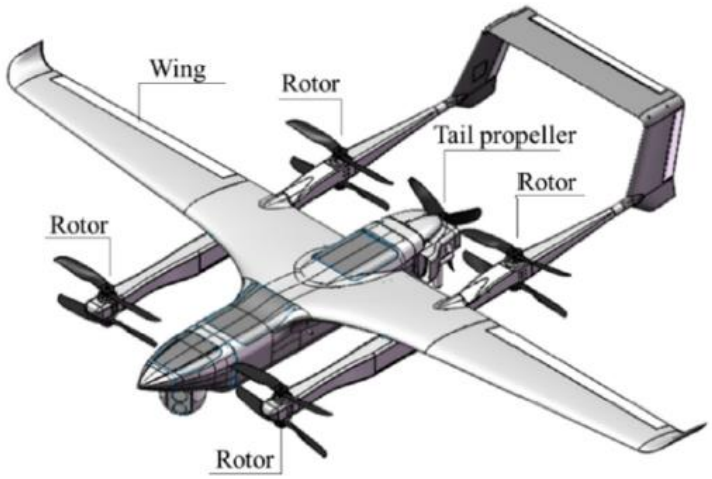

Figure 1.3 Hybrid VTOL UAV [15]

There are many different designs for hybrid UAVs exist. Some have mechanical parts attached to the rotor to rotate the motor after the VTOL; some have divided motors; some are responsible for take-off, and some for propulsion. [15]

Hybrid UAVs are a relatively new concept for aeronautical areas. They created to take advantage of both fixed-wing and rotary-wing UAVs, but they came up with some other disadvantages:

- They are mechanically complex systems.
- Since they are included both fixed and rotary wing UAV equipment, they have complex controller designs.
- Since they have different approaches to hybrid concepts, they have different types of controllers and equations of motion, making them create a custom controller design for each.

1.2. Comparisons of All UAV Types

This section collects all the advantages and disadvantages of all types of UAVs. Table 1.1 shows all the comparison results.

Table 1.1 Comparison Table for UAV Types

UAV Types	Advantageous	Disadvantageous
Fixed-Wing	• Energy efficient • High speed, range, and endurance • Mechanically simple	• Cannot vertically take-off • Low agility for small movements • Needs a proper field to take-off • Cannot hover on air
Rotary-Wing	• Hight agility for flight • Vertical take-off ability • Low cost for the test	• Mechanically complex • Very low energy efficiency • Complex controller design needed
Hybrid	• Can take-off vertically • Low energy consumption • High speed, range, endurance, and agility	• Highly complex for controller design • Mechanically more complex than fixed-wing UAVs.

The comparison table shows that all UAVs have some advantages and disadvantageous for each other. So, there is not the best choice for the selection of UAVs, but it depends on the application. On the other hand, hybrid UAVs have massive potential for future applications. From the point of those in this thesis one of the popular rotary-wing UAV Parrot Mambo Mini drone is chosen which is given in Figure 1.4.

Figure 1.4 Parrot Mambo Mini Drone [16]

1.3. Controller Types for Quadrotor Control

Many different types of controllers exist in the literature. Like all the concepts of the world, there is no best controller for all the applications. Some of them have a better solution for some applications than others but have disadvantages, like complexity or high-cost energy. So that there are optimal controller selections for each type of application. There are many different solutions for controller selection regarding quadrotor control. In Figure 1.5, some controller type has been given categorized ways. [17] Next section is dedicated to explaining the concepts for each controller given in Figure 1.5.

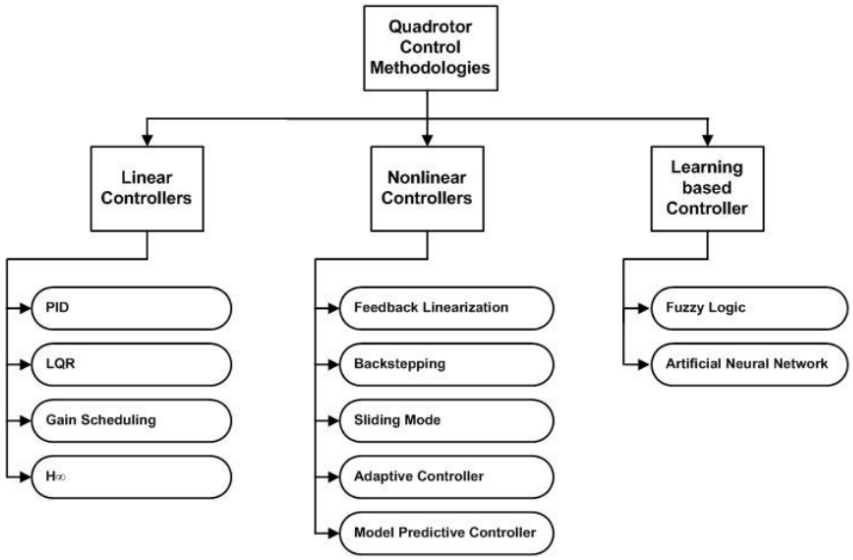

Figure 1.5 Controller Methodologies for Quadrotor Control [17]

Linear Controller is the most common controller type because it is easy to design and use. They are very straightforward when it comes to the design of them. The simplest way they can express as have plant, controller, and feedback. Figure 1.6 is a simple example of a feedback controller.

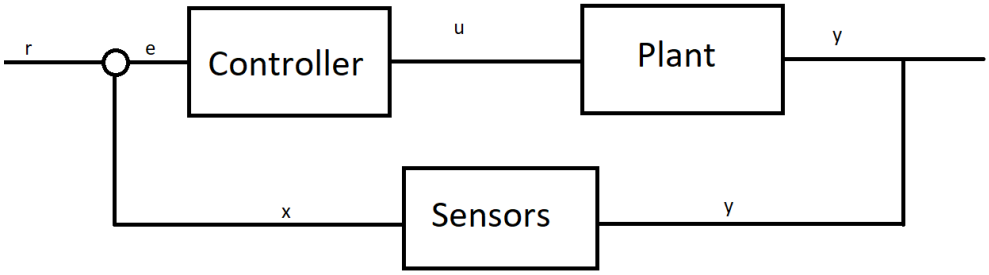

Figure 1.6 Feedback controller

The other controller methodologies mostly have the same concepts, but it changes inside of the controller.

1.3.1. Feedback Controllers

The linear controller technique is the most common technique for quadrotor control. They are simple and sufficient enough for most cases. [17]

PID Controller

PID controller is powerful in applications, and dynamics are unknown because trial-and-error methods can tune it after applying the PID controller. [17,18] It is time-consuming since finding appropriate gains is random. But some tools and techniques exist for finding appropriate control gain. MATLAB has some excellent tools for that purpose.

LQR (Linear Quadratic Regulator) Controller

LQR is one of the many types of optimal controllers. It is powerful due to its robust performance. [17] Some successful quadrotor applications exist in the literature, but some unsuccessful ones exist elsewhere. From this point of that, LQR mainly uses some other controller. [17]

Gain Scheduling

It is a powerful control technique since it is a linear but nonlinear model-based control. Some applications are used for quadrotor control. It is a stroll controller in quadrotor control because it is a model-based control algorithm. The model can be modified for each quadrotor.

H_∞ (H-infinity)

H_∞ is one of the most popular robust controllers in literature. It is powerful due to can be applied to unmodeled dynamical systems. [17] There are some applications for quadrotor control with H_∞ control—most of the works about path-following commands are in the literature.

1.3.2. Nonlinear Controllers

Nonlinear controllers are some of the robust controller designs because they are based on nonlinear model-based controllers. [17] Almost all systems are nonlinear in real life, so they apply to nearly every scenario. But on the other hand, even though some systems are nonlinear, they can be solved by a linear approach very easily, so using a

nonlinear controller for those systems makes the system complicated and energy inefficient.

Feedback Linearisation

Feedback Linearisation is based on converting the nonlinear dynamics of the system into a linear-based system after applying the linear control methods to the design and converting back to the nonlinear system with inverse conversion methods. This is the basic principle of the feedback linearisation control method. [17] Some examples of quadrotors' feedback linearisation controller design exist in the literature.

Backstepping

Backstepping control can be used for systems that have a high disturbance. It is suitable for compensating for the trouble, but on the other hand, it is not robust as the other system. Some examples of quadrotor control with backstepping control, but the hybrid controlling technique is more sufficient for quadrotor control. [17]

Sliding Mode

It is a nonlinear system control tool. Its robustness is better than any other control method in this thesis for quadrotor control. It has good outputs for quadrotor control with disturbance and uncertainties. [17,19] But it has a low quality for flight performance.

Adaptive Control

The adaptive controller is a robust control method that works well under uncertainties. Moreover, it has a good solution with a nonlinear model with unmodeled dynamics. [17]

Model Predictive Control

Model Predictive Control (MPC) is an online model-based optimization control algorithm. MPC uses the current and past control output to predict the future output and input. MPC can be categorized as nonlinear, adaptive, or linear control. Nonlinear MPC can be described by its name. It uses that nonlinear model for control; adaptive MPC uses the system model with some variables that are updated in time so that the system model adapts in time. [17,20,21,22,23]

MPC works on discrete time, predicting the future output and input until the prediction horizon. After the prediction, it optimizes the cost function, including weights on the creation, information, and future values. This allows the system to find an optimal solution for each reference trajectory time step. [17,20,21,22,23]

1.3.3. Learning-Based Controllers

Learning-based controllers are the most powerful controllers. They are robust, dynamic models of the system are not necessary, and the optimization is quite good comparing the other controllers. Still, their major disadvantage is that they must be adequately created to work fine. Moreover, they are much more complex than the other ones. [17] Most common learning-based controllers are fuzzy and neural-network control or adaptive neural network controls.

1.4. Inputs of The Quadrotor

Quadrotors are controlled by regulating the angular velocities of each motor. With this regulation, the quadrotor creates angular movement and thirst. Those angular movements are called pitch, roll, and yaw movements. Those movements and the thrust are called quadrotors inputs. [24, 25, 26]

Despite having four inputs, quadrotors have six outputs (x,y,z,φ,θ,ψ). This thesis represents inputs as U_1, U_2, U_3, and U_4. These inputs can mathematically be shown as follows;

$$\begin{aligned} U_1 &= b(\Omega_1^2 + \Omega_2^2 + \Omega_3^2 + \Omega_4^2) \\ U_2 &= bl(\Omega_1^2 - \Omega_2^2 - \Omega_3^2 + \Omega_4^2) \\ U_3 &= bl(-\Omega_1^2 - \Omega_2^2 + \Omega_3^2 + \Omega_4^2) \\ U_4 &= d(-\Omega_1^2 + \Omega_2^2 - \Omega_3^2 + \Omega_4^2) \end{aligned} \tag{1.1}$$

Where b is the thrust coefficient of the motors, l is the distance between the engines and the center of mass of the quadrotor, d is the drag coefficient of the motors, and $\boldsymbol{\Omega_i}$ is the angular speed of each i^{th} motor.

The inputs U_1, U_2, U_3, and U_4 are called thrust, pitch, roll, and yaw, respectively. Figure 1.7 to Figure 1.10 shows schematic concepts of each input.

1.4.1. Thrust Force - U_1 (N)

When four motors of the quadrotor rotate, they create thrust for force from the geometry of the propeller. Therefore, propellers design and arrange the quadrotor to develop a downward thrust as the reaction quadrotor moves upward. In the quadrotor design, opposite-sided two of the four motors pair with each other and rotate in the same orientation, although two others rotate on opposite sides. [24, 25, 26] Motor 1 and motor 3 rotate clockwise, and motor two and motor 4 rotates counterclockwise.

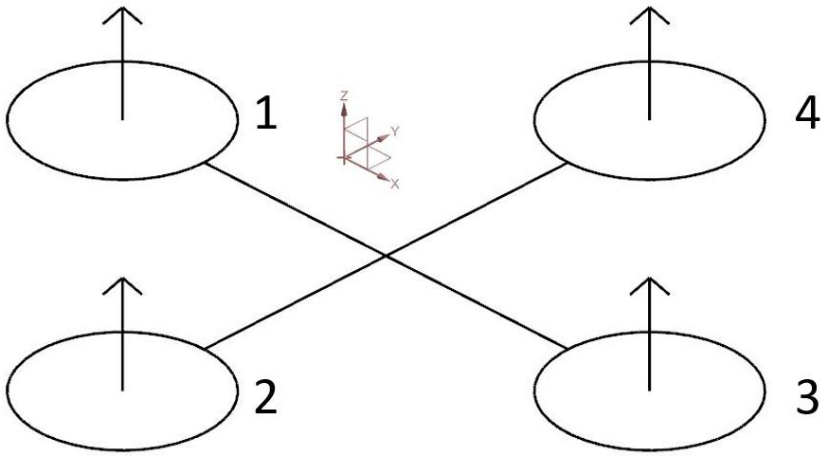

Figure 1.7 Thrust Force Motor Orientations

Two clockwise, two counterclockwise rotating propeller design allows the quadrotor to fight stability. If all four motors rotate in the same direction, the quadrotor moves upward but also turns in the same direction as the motors. All the motor causes this to push the air in the same direction. From Newton's third law, pushing air causes to create a force in another direction. Collectively they cause the quadrotor to take a moment in the same direction as the propellers. [24, 25, 26]

1.4.2. Roll Moment – U_2 (N.m)

Roll moment is made by keeping the front and the rear motor at the same speed, but the sides differ. This is because the quadrotor tends to roll toward the side, which has a lower motor speed. [24, 25, 26]. Figure 1.8 shows the motor speed and orientation; the roll moment to each side is shown as a schema.

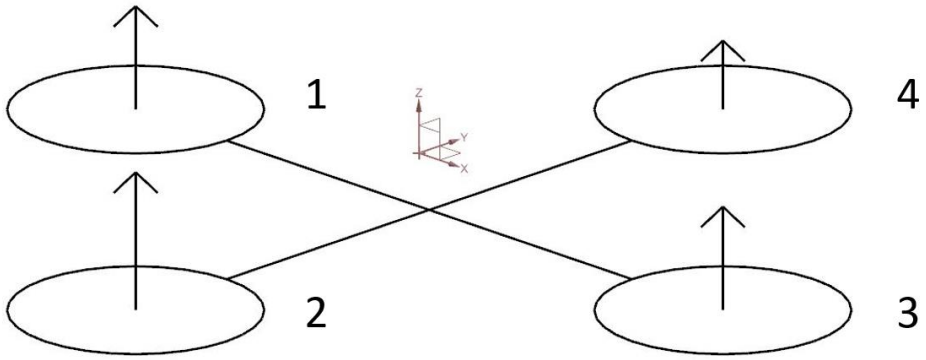

Figure 1.8 Roll Moment for Each Sides

1.4.3. Pitch Moment – U_3 (N.m)

A pitch moment occurs when two side motors have the same speed, the front motor is decreased speed, and the rear engine has a higher rate than the front motor. [24, 25, 26] When the pitch moment action happens, the controller needs to compensate for the thrust force not to cause losing altitude of the quadrotor. For example, in Figure 1.9, the pitch moment is shown.

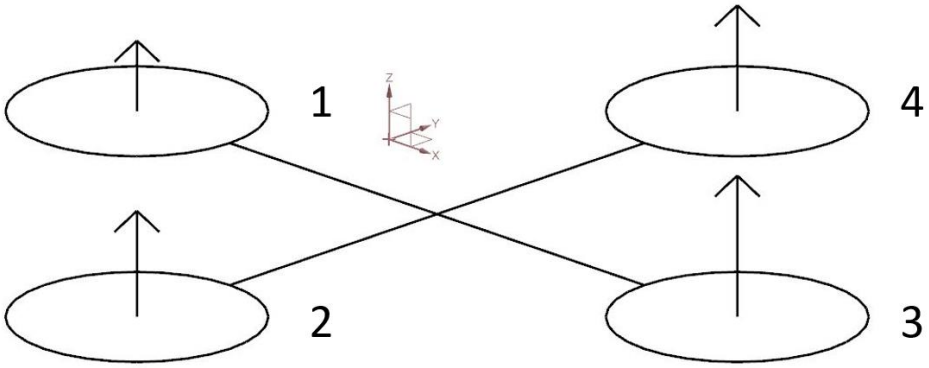

Figure 1.9 Pitch Moment

1.4.4. Yaw Moment – U4 (N.m)

Yaw moment is a little bit different from the other ones. It is made by the same principle as the other movement as adjusting the motor speed, but it occurs by changing the pair motors that have the same rate but are different to the other pair that causes the force mentioned on the third force part. Different pressure caused by pushing the air to the rotation direction causes the quadrotor to rotate. When this movement happens, the controller needs to adjust the motor's speed to a level so that the quadrotor does not lose altitude. This occurs by lowering a pair's speed but

increasing the other pair's speed until the quadrotor has the same thrust force. Figure 1.10 represents the yaw moment of the quadrotor. [24, 25, 26]

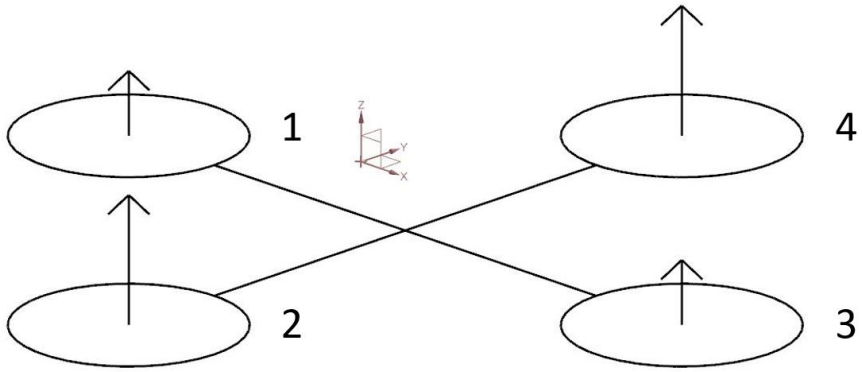

Figure 1.10 Yaw Moment

Those are the four input signals of the quadrotor. After controlling the inputs, real quadrotor control turns into the four motor speeds to seed directly to ESC (Electronic Speed Controller) or the controller in the quadrotor-embedded CPU unit. The states are collated with the quadrotors sensors after the control signal is sent to the systems. But in this thesis, no experiment was done, so the quadrotors model calculated the sensor values.

2. EQUATIONS OF MOTION (QUADROTOR MODEL)

In section 1 in the control methodologies part, some methods do not need a mathematical system model. But in this thesis, MPC controllers have been used, so a proper model is needed. Also, in this thesis, there are no experimental results, so validation is only made by simulation, and for proper simulation, a suitable mathematical mode is needed.

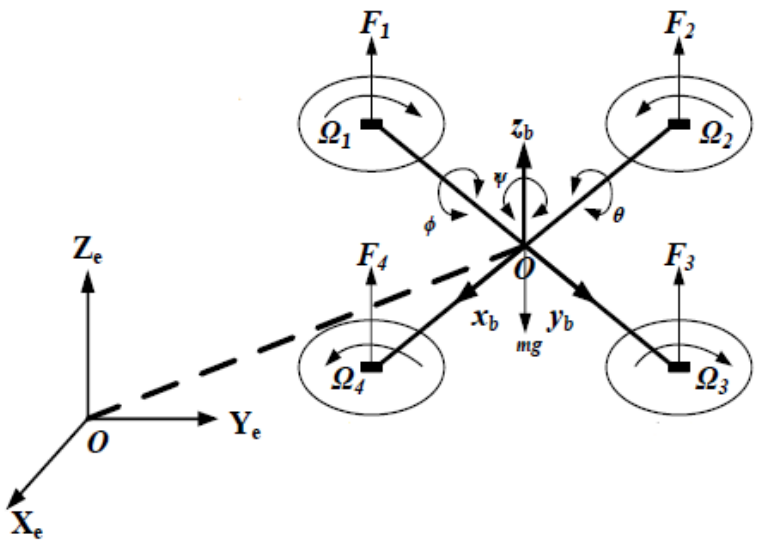

Figure 2.1 Body to Earth Coordinates Frame Representation [26]

A quadrotor is a moving object that cannot be controlled earth frame coordinate system. There should be a body frame coordinate system, but on the other hand, trajectory references of the quadrotor are sent by being defined in earth frame coordinates. For an example of the two different coordinates frames is given in Figure 2.1Two coordinate frames are defined for this approach to work, and a transformation matrix is defined.

2.1. Euler Angles

Transformation matrix derived for interchanging earth-to-body and body-to-earth coordinates for any input or output. Two different transformation matrices are derived.

One is called the rotation matrix, and the other is called the angular transformation matrix.

In this part coordinates transformation matrix is derived for roll, pitch, and yaw movement. Those rotations are called Euler Angles.

2.1.1. Roll

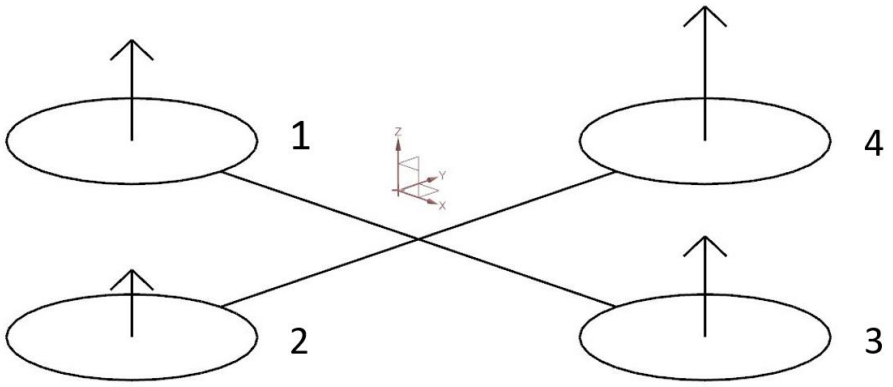

Figure 2.2 Roll Movement

Roll movements occur around the X_b axis as it is shown in Figure 2.2. Therefore, the Rx rotation matrix is derived considering no rotational change on the X_b axis.

$$X_b = X_e$$

$$Y_b = Y_e \cos(\phi) - Z_e \sin(\phi) \tag{2.1}$$

$$Z_b = Y_e \sin(\phi) + Z_e \cos(\phi)$$

Equation 2.1 shows the rotational transformation equations of the roll. As mentioned before, there is no change in body X axis to earth X axis. Therefore, these equations should be represented in matrix form to achieve the transformation matrix.

$$\begin{bmatrix} X_b \\ Y_b \\ Z_b \end{bmatrix} = \begin{bmatrix} 1 & 0 & 0 \\ 0 & \cos(\phi) & -\sin(\phi) \\ 0 & \sin(\phi) & \cos(\phi) \end{bmatrix} \begin{bmatrix} X_e \\ Y_e \\ Z_e \end{bmatrix}$$

$$R_x = \begin{bmatrix} 1 & 0 & 0 \\ 0 & \cos(\phi) & -\sin(\phi) \\ 0 & \sin(\phi) & \cos(\phi) \end{bmatrix} \tag{2.2}$$

Equation 2.2 shows the rotational equation in matrix form and the R_x matrix required for calculating the total transformation matrix.

2.1.2. Pitch

Pitch movement occurs around the Y_b axis. So that Y_b and Y_e are the same. There is no coefficient to convert them. All the equations for pitch movement are shown in Equation 2.3 and in Figure 2.3 it show that how the pitch movement occurs.

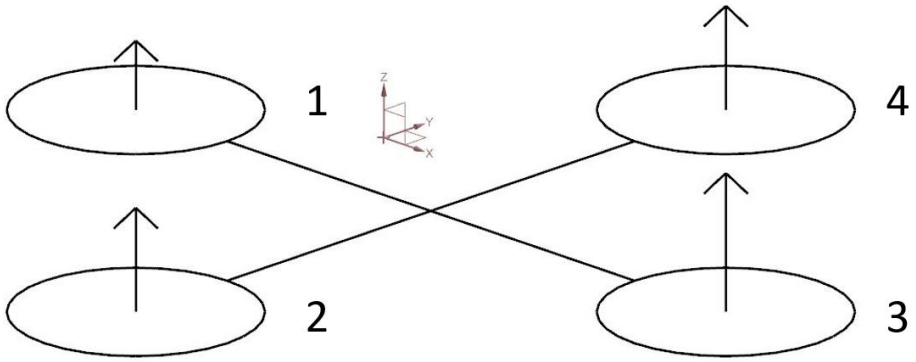

Figure 2.3 Pitch Movement

$$X_b = X_e \cos(\theta) + Z_e \sin(\theta)$$
$$Y_b = Y_e \tag{2.3}$$
$$Z_b = -X_e \sin(\theta) + Z_e \cos(\theta)$$

Equation 2.3 shows the rotational equations of pitch movement. Equation 2.4 is the matrix form representations of Equation 2.3 and the R_y matrix.

$$\begin{bmatrix} X_b \\ Y_b \\ Z_b \end{bmatrix} = \begin{bmatrix} \cos(\theta) & 0 & \sin(\theta) \\ 0 & 1 & 0 \\ -\sin(\theta) & 0 & \cos(\theta) \end{bmatrix} \begin{bmatrix} X_e \\ Y_e \\ Z_e \end{bmatrix}$$
$$R_y = \begin{bmatrix} \cos(\theta) & 0 & \sin(\theta) \\ 0 & 1 & 0 \\ -\sin(\theta) & 0 & \cos(\theta) \end{bmatrix} \tag{2.4}$$

2.1.3. Yaw

Yaw movement is rotation about Z_b axis. That's why Z_b axis to Z_e axis there is coefficient exists. They are directly converted. All the equations for yaw movement are given in Equation 2.5 and in Figure 2.4 it show that how the yaw movement occurs.

$$\begin{aligned} X_b &= X_e \cos(\psi) - Y_e \sin(\psi) \\ Y_b &= X_e \sin(\psi) + Y_e \cos(\psi) \\ Z_b &= Z_e \end{aligned} \tag{2.5}$$

The rotational matrix R_z is shown in Equation 2.6.

$$\begin{bmatrix} X_b \\ Y_b \\ Z_b \end{bmatrix} = \begin{bmatrix} \cos(\psi) & -\sin(\psi) & 0 \\ \sin(\psi) & \cos(\psi) & 0 \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} X_e \\ Y_e \\ Z_e \end{bmatrix}$$

$$R_z = \begin{bmatrix} \cos(\psi) & -\sin(\psi) & 0 \\ \sin(\psi) & \cos(\psi) & 0 \\ 0 & 0 & 1 \end{bmatrix} \tag{2.6}$$

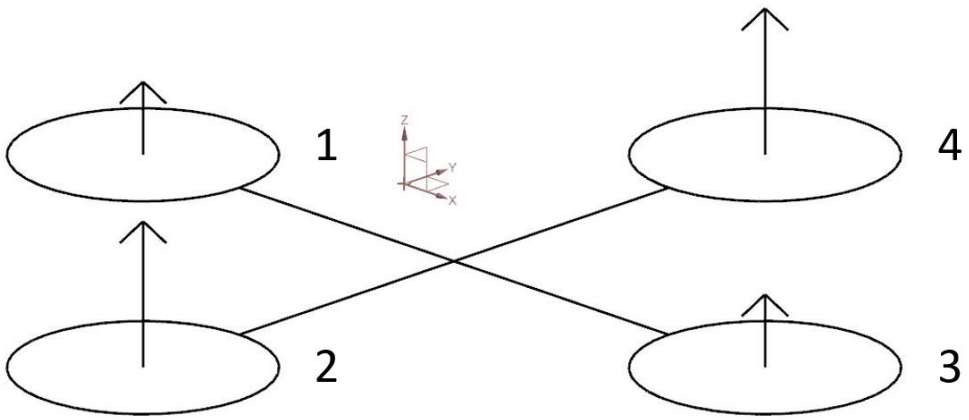

Figure 2.4 Yaw Movement

Rotational and angular rotation transformation matrices can be driven with all the Rx, Ry, and Rz rotational matrices.

2.2. Rotational Transformation Matrix

The orders of the three rotational matrices are essential for establishing the transformation matrices. However, it is not the same rotating the x, y, and z axes order as the z, y, and x-axes. [27] This thesis uses the "ZYX" order for the transformation matrices.

$$R_{zyx,be} = R_z R_y R_x$$

$$R_{zyx,eb} = \begin{bmatrix} \cos(\psi) & -\sin(\psi) & 0 \\ \sin(\psi) & \cos(\psi) & 0 \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} \cos(\theta) & 0 & \sin(\theta) \\ 0 & 1 & 0 \\ -\sin(\theta) & 0 & \cos(\theta) \end{bmatrix} \begin{bmatrix} 1 & 0 & 0 \\ 0 & \cos(\phi) & -\sin(\phi) \\ 0 & \sin(\phi) & \cos(\phi) \end{bmatrix} \tag{2.7}$$

$$R_{zyx,eb} = \begin{bmatrix} \mathrm{c}(\theta)\mathrm{c}(\psi) & \mathrm{s}(\phi)\,\mathrm{s}(\theta)\mathrm{c}(\psi) - \mathrm{c}(\phi)\,\mathrm{s}(\psi) & \mathrm{c}(\phi)\,\mathrm{s}(\theta)\,\mathrm{c}(\psi) + \mathrm{s}(\phi)\,\mathrm{s}(\psi) \\ c(\theta)s(\psi) & \mathrm{s}(\phi)\,\mathrm{s}(\theta)\mathrm{s}(\psi) + \mathrm{c}(\phi)\,\mathrm{c}(\psi) & \mathrm{c}(\phi)\,\mathrm{s}(\theta)\,\mathrm{c}(\psi) - \mathrm{s}(\phi)\,\mathrm{c}(\psi) \\ -s(\theta) & \mathrm{s}(\phi)\,\mathrm{c}(\theta) & c(\phi)c(\theta) \end{bmatrix}$$

Equation 2.7 shows the calculation of the rotational transformation matrix where c() stands for cos() and s() stands for sin() in short. This matrix is used for transforming earth coordinates to body coordinates. It can be reversed, as shown in Equation 2.8, by taking the matrix inverse.

$$R_{be} = (R_{zyx,eb})^{-1} \tag{2.8}$$

In the control of the quadrotor, the reference signals are sent based on the earth coordinates so that earth to body transformation matrix is needed.

2.3. Angular Transformation Matrix

As the rotational transformation mentions, the "ZYX" order is also used to establish the angular transformation matrix. The angular transformation matrix transforms earth-to-body and body-to-earth coordinate frames of angles, angular velocities, and acceleration.

$$T_{be} \begin{bmatrix} \dot{\phi} \\ \dot{\theta} \\ \dot{\psi} \end{bmatrix} = R_x R_y R_z \begin{bmatrix} 0 \\ 0 \\ \dot{\psi} \end{bmatrix} + R_x R_y \begin{bmatrix} 0 \\ \dot{\theta} \\ 0 \end{bmatrix} + R_x \begin{bmatrix} \dot{\phi} \\ 0 \\ 0 \end{bmatrix} \tag{2.9}$$

If the angular rate from Equation 2.9 can be extracted, it shows the angular transformation matrix as shown in Equation 2.10.

$$T_{be} = \begin{bmatrix} 1 & 0 & -\sin(\theta) \\ 0 & \cos(\phi) & \sin(\phi)\cos(\theta) \\ 0 & -\sin(\phi) & \cos(\phi)\cos(\theta) \end{bmatrix} \quad (2.10)$$

Equation 2.10 shows the angular transformation matrix of the body-to-earth. Due to earth references on the controller, the earth-to-body transformation matrix is needed.

$$T_{eb} = (T_{be})^{-1}$$

$$T_{eb} = \begin{bmatrix} 1 & \sin(\phi)\tan(\theta) & \cos(\phi)\tan(\theta) \\ 0 & \cos(\phi) & -\sin(\phi) \\ 0 & \sin(\phi)\sec(\theta) & \cos(\phi)\sec(\theta) \end{bmatrix} \quad (2.11)$$

To satisfy the need for an earth-to-body angular transformation matrix, Equation 2.11 is established. From now on, the rotational and the angular transformation matrices are represented as R and T, respectively.

2.4. Kinematic of Quadrotor

Quadrotors have six degrees of freedom (x,y,z,φ,θ,ψ). Those are the system outputs. Controlling the quadrotor, the simulation part calculates these outputs changing rates. These rates are converted into the simulation using transformation matrices established in sections 2.2 and 2.3 and the quadrotor body changing rates.

$$\begin{bmatrix} \dot{x} \\ \dot{y} \\ \dot{z} \end{bmatrix} = R \begin{bmatrix} u \\ v \\ w \end{bmatrix}$$

$$\begin{bmatrix} \dot{\phi} \\ \dot{\theta} \\ \dot{\psi} \end{bmatrix} = T \begin{bmatrix} p \\ q \\ r \end{bmatrix} \quad (2.12)$$

The transformation of body-to-earth coordinates of angular and positional velocities is shown in Equation 2.12. In Equation 2.12, there are terms u,v,w,p,q, and r are velocity in the x direction, velocity in the y direction, velocity in the z direction, angular velocity in φ direction, angular velocity in θ direction and angular velocity in ψ direction on the body frame respectively.

$$V = RV_b$$

$$W = TW_b \quad (2.13)$$

Equation 2.13 shows the short representation of Equation 2.12. In other parts, these representations are used.

$$\begin{aligned} V &= [x \quad y \quad z]^T \\ W &= [\phi \quad \theta \quad \psi]^T \\ V_b &= [u \quad v \quad w]^T \\ W_b &= [p \quad q \quad r]^T \end{aligned} \tag{2.14}$$

2.5. Forces and Moments

Several different forces and moments are acting on the quadrotor. Therefore, these forces and moments should be introduced appropriately to find the equations of motion of the quadrotor.

Any system's mathematical models have some assumption to control the system. So the quadrotors mathematical model is not too different from the other system. Some assumptions are made to find a proper model to control a quadrotor.

$$\begin{aligned} F_{body} &= F_{gravity} + F_{thrust} \\ F_b &= mgR\hat{k} - F_t\hat{k}_b \end{aligned} \tag{2.15}$$

In Equation 2.15, $\hat{k}$ and $\hat{k}_b$ are axis unit vectors on earth and body frame, respectively.

$$F_b = m\dot{V}_b + W_b x(mV_b) \tag{2.16}$$

Newton's law of motion shows the total force acting on the quadrotor, as shown in Equation 2.16.

$$\begin{aligned} M_{body} &= \tau_{body} - g_{actuator} \\ M_{body} &= M_b \qquad \tau_{body} = \tau_b \qquad g_{actuator} = g_a \end{aligned} \tag{2.17}$$

Moments acting on the quadrotor are shown in Equation 2.17.

$$M_b = I\dot{W}_b + W_b x(IW_b) \tag{2.18}$$

All the moments and the forces acting on the quadrotor are given in Equation 2.15 to Equation 2.18. That equation shows the resultant force and moments in two different

ways. One is theoretical, and the other one is the assumption and observations. In Equation 2.18 I is the inertia matrix which can be shown as;

$$I = \begin{bmatrix} I_x & 0 & 0 \\ 0 & I_y & 0 \\ 0 & 0 & I_z \end{bmatrix}$$

Equation 2.19 shows the results of substituting all the parameters into Equation 2.16, where $F_b = [F_x \quad F_y \quad F_z]^T$ forces action in each direction of the quadrotor body.

$$\begin{bmatrix} F_x \\ F_y \\ F_z \end{bmatrix} = m \begin{bmatrix} \dot{u} \\ \dot{v} \\ \dot{w} \end{bmatrix} + \begin{bmatrix} p \\ q \\ r \end{bmatrix} x \left(m \begin{bmatrix} u \\ v \\ w \end{bmatrix} \right) \tag{2.19}$$

$$\begin{bmatrix} F_x \\ F_y \\ F_z \end{bmatrix} = m \begin{bmatrix} u + q\dot{w} - rv \\ \dot{v} - pw + ru \\ \dot{w} + pv - qu \end{bmatrix}$$

Equation 2.20 shows the resultant moment matrix, substituted form of all the parameters into Equation 2.18.

$$\begin{bmatrix} M_x \\ M_y \\ M_z \end{bmatrix} = \begin{bmatrix} I_x & 0 & 0 \\ 0 & I_y & 0 \\ 0 & 0 & I_z \end{bmatrix} \begin{bmatrix} \dot{p} \\ \dot{q} \\ \dot{r} \end{bmatrix} + \begin{bmatrix} p \\ q \\ r \end{bmatrix} x \left(\begin{bmatrix} I_x & 0 & 0 \\ 0 & I_y & 0 \\ 0 & 0 & I_z \end{bmatrix} \begin{bmatrix} p \\ q \\ r \end{bmatrix} \right) \tag{2.20}$$

$$\begin{bmatrix} M_x \\ M_y \\ M_z \end{bmatrix} = \begin{bmatrix} \dot{p}I_x - qrI_y + qrI_z \\ \dot{q}I_y + prI_x - prI_z \\ \dot{r}I_z - pqI_x + pqI_y \end{bmatrix}$$

Equation 2.15 and Equation 2.19 are equal to each other since they are equal to F_b. So that substituting both equations with each other is resultant the equation 2.21.

$$\begin{aligned} -mg[\sin(\theta)] &= m(\dot{u} + qw - rv) \\ mg[\cos(\theta)\sin(\phi)] &= m(\dot{v} - pw + ru) \\ mg[\cos(\theta)\cos(\phi)] - F_t &= m(\dot{w} + pv - qu) \end{aligned} \tag{2.21}$$

Also, the same process was applied in Equations 2.17 and Equation 2.20; the results are shown in Equation 2.22.

$$\tau_x + g_{ax} = \dot{p}I_x - qrI_y + qrI_z$$

$$\tau_y + g_{ay} = \dot{q}I_y + pqI_x - prI_z \qquad (2.22)$$

$$\tau_z + g_{az} = \dot{r}I_z - pqI_x + pqI_y$$

Equation 2.21 and Equation 2.22 shows all the moments and forces acting on the quadrotor body. These equations are derived with the assumption given below;

- Center of mass is the center of the quadrotor's four motors. Therefore, all the motors have an equal distance to the center of mass.
- Quadrotor's origin and the center of mass coincide.
- Quadrotor's body is rigid, and there is no flexing through flight.
- No wind force acting on the quadrotor. If it was considered, the aerodynamic force part should be added to the total force.
- Quadrotor's propellers are rigid and generate thrust proportional to the square of the motor's rotational speed.
- No noise was measured between the sensors, motor,s and controller communication.

Equation 2.21 and Equation 2.22 are derived by considering these assumptions.

2.6. Actuator and Gyroscopic Moment

In a quadrotor, actors are four motors. Section 2.5 moments and the action of the forces on the quadrotor derives as torque and force. In this section, firstly, actuator dynamics was given, and then the gyroscopic moment $g_a$was explained. Finally, section 1.4 shows the inputs of the system. Those inputs are the torque and the thrust force action on the quadrotor.

$$F_t = b(\Omega_1^2 + \Omega_2^2 + \Omega_3^2 + \Omega_4^2) = U_1$$

$$\tau_x = bl(\Omega_1^2 - \Omega_2^2 - \Omega_3^2 + \Omega_4^2) = U_2 \qquad (2.23)$$

$$\tau_y = bl(-\Omega_1^2 - \Omega_2^2 + \Omega_3^2 + \Omega_4^2) = U_3$$

$$\tau_z = d(-\Omega_1^2 + \Omega_2^2 - \Omega_3^2 + \Omega_4^2) = U_4$$

The gyroscopic moments are compelling on the roll and the pitch moment, but their effect on the yaw moment is too small, which is neglected in this thesis. In Figure 2.5 how gyroscopic moment is occur is given as an example .

$$\Omega_t = \Omega_1 - \Omega_2 + \Omega_3 - \Omega_4 \tag{2.24}$$

Equation 2.24 is the total motor rotation in the order of their rotation orientations.

$$g_a = J_{tp}(W_b x \hat{k}_b)\Omega_t \tag{2.25}$$

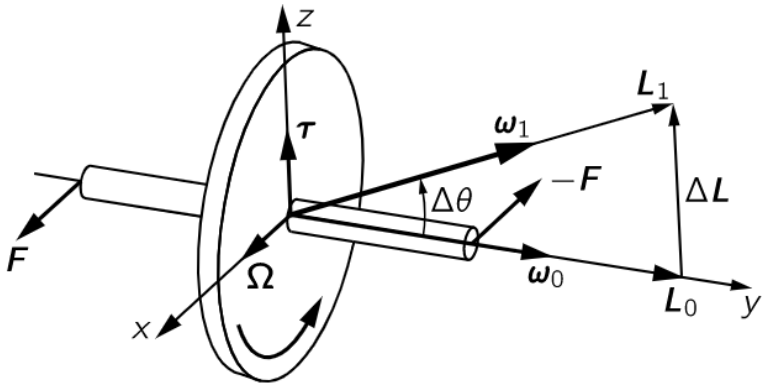

Figure 2.5 Gyroscopic Moments Example

Equation 2.25 is the gyroscopic moments on the actuators. The gyroscopic moments that are used for the finding quadrotors model are shown in Equation 2.26

$$g_{ax} = -\frac{J_{tp}}{I_x} q\Omega_t$$

$$g_{ay} = \frac{J_{tp}}{I_Y} P\Omega_t \tag{2.26}$$

2.7. Quadrotor Model and State-Space Equations

From the beginning of Section 2, all the equations for deriving the quadrotor's mathematical model (equations of motion). This section forms these equations to find the desired parameters to control a quadrotor. Those parameters change (x, y, z, u, v, w, φ, θ, ψ, p, q, r).

Controller design of the quadrotor made from state-space representation. State-space representation is used in many articles about quadrotor control in the literature. [24,28] In the thesis also, the state-space representation is used.

$$\dot{x}(t) = Ax(t) + Bu(t)$$
$$y(t) = Cx(t) + Du(t) \tag{2.27}$$

Equation 2.27 show the state-space model of any control system. This representation is based on continuous time; however, MPC is a discrete-time-based control. Finally, Equation 2.28 shows the discrete-time state-space model given.

$$\dot{x}(k+1) = Ax(k) + Bu(k)$$
$$y(k) = Cx(k) + Du(k) \tag{2.28}$$

State-space model of the quadrotor can be established with the derived equations. However, before starting the state-space model, the quadrotor's states should be decided to maintain stable control.

$$X = [\dot{x} \quad \dot{y} \quad \dot{z} \quad \dot{u} \quad \dot{v} \quad \dot{w} \quad \dot{\phi} \quad \dot{\theta} \quad \dot{\psi} \quad \dot{p} \quad \dot{q} \quad \dot{r}\,]^T \tag{2.29}$$

Quadrotors have twelve states. These are given in Equation 2.29. It is represented on the body frame coordinate system. These states can also be represented as Equation 2.30 in the Earth frame coordinate system.

$$X = [\dot{x} \quad \ddot{x} \quad \dot{y} \quad \ddot{y} \quad \dot{z} \quad \ddot{z} \quad \dot{\phi} \quad \ddot{\phi} \quad \dot{\theta} \quad \ddot{\theta} \quad \dot{\psi} \quad \ddot{\psi}\,]^T \tag{2.30}$$

Before establishing the state-space model, all the equations included in the state-space model need to be established.

$$\dot{u} = (vr - wq) + gsin(\theta)$$
$$\dot{v} = (wp - ur) - gcos(\theta)\sin(\phi)$$
$$\dot{w} = (uq - vp) - gcos(\theta)\cos(\phi) + \frac{U_1}{m}$$
$$\dot{p} = \left(\frac{I_y - I_z}{I_x}\right) qr - q\frac{J_{tp}}{I_x}\Omega_t + \frac{U_2}{I_x} \tag{2.31}$$
$$\dot{q} = \left(\frac{I_z - I_x}{I_y}\right) pr + p\frac{J_{tp}}{I_y}\Omega_t + \frac{U_3}{I_y}$$
$$\dot{r} = \left(\frac{I_x - I_y}{I_z}\right) pq + \frac{U_4}{I_z}$$

Equation 2.31 is the nonlinear model of the quadrotor based on the body frame coordinates system. However, references to the trajectory are given to the system as an earth coordinate system. That's why there needs to be earth coordinates system representation needed.

$$\ddot{x} = [s(\phi)s(\psi) + c(\phi)c(\psi)s(\theta)]\frac{U_1}{m}$$

$$\ddot{y} = [c(\phi)s(\psi)s(\theta) - c(\psi)s(\phi)]\frac{U_1}{m}$$

$$\ddot{z} = -g + [c(\phi)c(\theta)]\frac{U_1}{m}$$

$$\ddot{\phi} = \left(\frac{I_y - I_z}{I_x}\right)\dot{\theta}\dot{\psi} - \dot{\theta}\frac{J_{tp}}{I_x}\Omega_t + \frac{U_2}{I_x} \tag{2.32}$$

$$\ddot{\theta} = \left(\frac{I_z - I_x}{I_y}\right)\dot{\phi}\dot{\psi} + \dot{\phi}\frac{J_{tp}}{I_y}\Omega_t + \frac{U_3}{I_y}$$

$$\ddot{\psi} = \left(\frac{I_x - I_y}{I_z}\right)\dot{\phi}\dot{\theta} + \frac{U_4}{I_z}$$

State-space models can be derived from all the given terms in Equation 2.31 and Equation 2.32. Equations and conversion of the body frame and the earth frame coordinate are given in Equation 2.33.

$$\dot{x} = w[s(\phi)c(\psi) + c(\phi)s(\psi)s(\theta)] - v[c(\phi)s(\psi) - c(\psi)s(\phi)s(\theta)] + u[c(\psi)c(\theta)]$$
$$\dot{y} = v[c(\phi)c(\psi) + s(\phi)s(\psi)s(\theta)] - w[c(\psi)s(\phi) - c(\phi)s(\psi)s(\theta)] + u[c(\theta)s(\psi)]$$
$$\dot{z} = w[c(\phi)c(\theta)] - u[s(\theta)] + v[c(\theta)s(\phi)]$$
$$\dot{\phi} = p + [s(\phi)t(\theta)]q + [c(\phi)t(\theta)]r$$
$$\dot{\theta} = [c(\theta)]q - [s(\phi)]r$$
$$\dot{\psi} = \left[\frac{s(\phi)}{c(\theta)}\right]q + \left[\frac{c(\phi)}{c(\theta)}\right]r$$
$$\dot{p} = \left(\frac{I_y - I_z}{I_x}\right)qr - q\frac{J_{tp}}{I_x}\Omega_t + \frac{U_2}{I_x} \tag{2.33}$$
$$\dot{q} = \left(\frac{I_z - I_x}{I_y}\right)pr + p\frac{J_{tp}}{I_x}\Omega_t + \frac{U_3}{I_y}$$
$$\dot{r} = \left(\frac{I_x - I_y}{I_z}\right)pq + \frac{U_4}{I_z}$$
$$\dot{u} = rv - qw + g[s(\theta)]$$
$$\dot{v} = pw - rv + g[c(\theta)s(\phi)]$$
$$\dot{w} = qu - pv - g[c(\theta)c(\phi)] + \frac{U_1}{m}$$

Some terms in Equation 2.33 can be represented as;

$$\begin{bmatrix}\dot{x}\\ \dot{y}\\ \dot{z}\end{bmatrix} = R\begin{bmatrix}u\\ v\\ w\end{bmatrix}$$

$$\begin{bmatrix}\dot{\phi}\\ \dot{\theta}\\ \dot{\psi}\end{bmatrix} = T\begin{bmatrix}p\\ q\\ r\end{bmatrix}$$

Quadrotor's state-space representation can be established from all the equations given above. Therefore, equation 3.34 is the final state-space representation of the quadrotor model used in this thesis.

$$\begin{bmatrix}\dot{x}\\ \ddot{x}\\ \dot{y}\\ \ddot{y}\\ \dot{z}\\ \ddot{z}\\ \dot{\phi}\\ \ddot{\phi}\\ \dot{\theta}\\ \ddot{\theta}\\ \dot{\psi}\\ \ddot{\psi}\end{bmatrix} = \begin{bmatrix}0&1&0&0&0&0&0&0&0&0&0&0\\ 0&0&0&0&0&0&0&0&0&0&0&0\\ 0&0&0&1&0&0&0&0&0&0&0&0\\ 0&0&0&0&0&0&0&0&0&0&0&0\\ 0&0&0&0&0&1&0&0&0&0&0&0\\ 0&0&0&0&0&0&0&0&0&0&0&0\\ 0&0&0&0&0&0&0&1&0&0&0&0\\ 0&0&0&0&0&0&0&0&0&-\frac{J_{tp}}{I_x}\Omega_t&0&\dot{\theta}\left(\frac{I_y-I_z}{I_x}\right)\\ 0&0&0&0&0&0&0&0&0&1&0&0\\ 0&0&0&0&0&0&0&\frac{J_{tp}}{I_y}\Omega_t&0&0&0&\dot{\phi}\left(\frac{I_z-I_x}{I_y}\right)\\ 0&0&0&0&0&0&0&0&0&0&0&1\\ 0&0&0&0&0&0&0&\frac{\dot{\theta}}{2}\left(\frac{I_x-I_y}{I_z}\right)&0&\frac{\dot{\phi}}{2}\left(\frac{I_x-I_y}{I_z}\right)&0&0\end{bmatrix}\begin{bmatrix}x\\ \dot{x}\\ y\\ \dot{y}\\ z\\ \dot{z}\\ \phi\\ \dot{\phi}\\ \theta\\ \dot{\theta}\\ \psi\\ \dot{\psi}\end{bmatrix}$$
$$+\begin{bmatrix}0&0&0&0\\ [s(\phi)s(\psi)+c(\phi)c(\psi)s(\theta)]\frac{1}{m}&0&0&0\\ 0&0&0&0\\ [c(\phi)s(\psi)s(\theta)-c(\psi)s(\phi)]\frac{1}{m}&0&0&0\\ 0&0&0&0\\ -\frac{g}{U_1}+[c(\phi)c(\theta)]\frac{1}{m}&0&0&0\\ 0&0&0&0\\ 0&\frac{1}{I_x}&0&0\\ 0&0&0&0\\ 0&0&\frac{1}{I_y}&0\\ 0&0&0&0\\ 0&0&0&\frac{1}{I_z}\end{bmatrix}\begin{bmatrix}U_1\\ U_2\\ U_3\\ U_4\end{bmatrix} \quad (2.34)$$

Equation 3.34 is the first part of the state-space model. It gives the A and the B matrix of the model. In Equation 3.35 C, the D matrix is given.

$$\begin{bmatrix} x \\ y \\ z \\ \phi \\ \theta \\ \psi \end{bmatrix} = \begin{bmatrix} 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 \end{bmatrix} \begin{bmatrix} x \\ \dot{x} \\ y \\ \dot{y} \\ z \\ \dot{z} \\ \phi \\ \dot{\phi} \\ \theta \\ \dot{\theta} \\ \psi \\ \dot{\psi} \end{bmatrix} + \begin{bmatrix} 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \end{bmatrix} \begin{bmatrix} U_1 \\ U_2 \\ U_3 \\ U_4 \end{bmatrix} \quad (2.35)$$

Equation 2.35 shows the C and the D matrix of the state-space model. In this thesis, no disturbance is implemented on the system so that the D matrix is all zeros.

In the next section, where the quadrotor controller is explained, this state-space model is separated into two parts to create two controllers. One controller is responsible for position control, and the other for angular control. The state-space model given in Equation 3.34 and Equation 3.35 was made by considering this approach. All the nonlinear parameters are on a single matrix from the state space equation. That approach was made on purpose because of the LPV approach. In the next section, more information is given about the LPV approach.

3. CONTROLLER DESIGN

In this part of the thesis controller design of the quadrotor is handled. As mentioned before, the controller is discussed in two parts. Figure 3.1 shows that the whole control system. The position controller controls the x, y, and z directions and the U_1 input; the other is an angular controller, which controls φ, θ, ψ angles and U_2, U_3, and U_4 inputs.

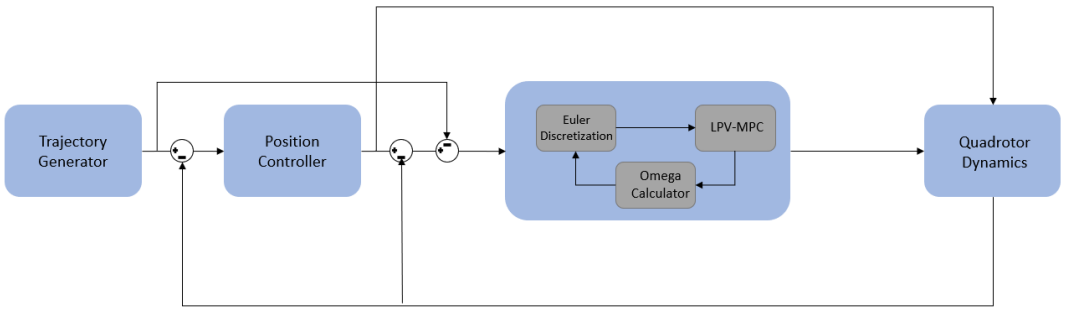

Figure 3.1 Block Diagram of the Control System of Quadrotor

PID controller is chosen for the position controller. It is a straightforward controller type, easy to design and optimize. Those are the main reasons to choose a PID controller. Also, it is a controller responsible for finding an appropriate U1 input shown in Equation 2.23. It is the thrust force of the quadrotor. It can be called relatively easy to handle, considering the other complexities of the quadrotor.

The main topic of this thesis is that the MPC controller is used for the angular controller of the quadrotor. Therefore, the angular controller is responsible for finding the desired U_2, U_3, and U_4 inputs. MPC controller is an online optimiser controller; that's why it is chosen for the angular controller. The angular change of the quadrotor controls the quadrotor's position. That is the main reason to choose a more robust controller for the angular controller of the quadrotor.

As mentioned before there are two different controllers have been used. Why not use two MPC controllers to find the position and the angular controller inputs? Even if MPC has better results for both applications, the energy consumption will be higher

than the simple PID controller. Furthermore, the position controller's PID controller seems enough to find desired reference angles, altitude and input in the literature. [1,29] Hence, using a robust controller on the position controller is unnecessary.

3.1. MPC Controller

MPC controller is used for angular control. Equation 2.34 and Equation 2.35 show the whole state-space model of the quadrotor. Since the controller is separated into two parts, that equation must also be separated. For the angular controller, it should include the angles and U_2, U_3, and U_4 inputs. From that knowledge, Equation 3.1 is derived.

$$\begin{bmatrix}\dot{\phi}\\ \ddot{\phi}\\ \dot{\theta}\\ \ddot{\theta}\\ \dot{\psi}\\ \ddot{\psi}\end{bmatrix}=\begin{bmatrix}0 & 1 & 0 & 0 & 0 & 0\\ 0 & 0 & 0 & -\frac{J_{tp}}{I_x}\Omega_t & 0 & \dot{\theta}\left(\frac{I_y-I_z}{I_x}\right)\\ 0 & 0 & 0 & 1 & 0 & 0\\ 0 & \frac{J_{tp}}{I_y}\Omega_t & 0 & 0 & 0 & \dot{\phi}\left(\frac{I_z-I_x}{I_y}\right)\\ 0 & 0 & 0 & 0 & 0 & 1\\ 0 & \frac{\dot{\theta}}{2}\left(\frac{I_x-I_y}{I_z}\right) & 0 & \frac{\dot{\phi}}{2}\left(\frac{I_x-I_y}{I_z}\right) & 0 & 0\end{bmatrix}\begin{bmatrix}\phi\\ \dot{\phi}\\ \theta\\ \dot{\theta}\\ \psi\\ \dot{\psi}\end{bmatrix}+\begin{bmatrix}0 & 0 & 0\\ \frac{1}{I_x} & 0 & 0\\ 0 & 0 & 0\\ 0 & \frac{1}{I_y} & 0\\ 0 & 0 & 0\\ 0 & 0 & \frac{1}{I_z}\end{bmatrix}\begin{bmatrix}U_2\\ U_3\\ U_4\end{bmatrix} \quad (3.1)$$

Equation 3.1 shows the states of the systems and includes the A and the B matrix of the system. C and the D of the system are shown in Equation 3.2.

$$\begin{bmatrix}\phi\\ \theta\\ \psi\end{bmatrix}=\begin{bmatrix}1 & 0 & 0 & 0 & 0 & 0\\ 0 & 0 & 1 & 0 & 0 & 0\\ 0 & 0 & 0 & 0 & 1 & 0\end{bmatrix}\begin{bmatrix}\phi\\ \dot{\phi}\\ \theta\\ \dot{\theta}\\ \psi\\ \dot{\psi}\end{bmatrix}+\begin{bmatrix}0 & 0 & 0\\ 0 & 0 & 0\\ 0 & 0 & 0\\ 0 & 0 & 0\\ 0 & 0 & 0\\ 0 & 0 & 0\end{bmatrix}\begin{bmatrix}U_2\\ U_3\\ U_4\end{bmatrix} \quad (3.2)$$

For the system's angular controller, all the needed state-space model is separated as in equation 3.1 and 3.2. But before starting the controller design, this state-space model must be on the discrete-time model since MPC is based on the discrete-time model. In that case, this thesis has used the discretization of the state-space model Euler Discretization method. [21, 22 ,23]

3.1.1. Euler Discretization Method

As mentioned before, MPC is based on the discrete-time system, so applying the MPC controller to the system state-space model of the quadrotor must be in discrete form. In that case, the Euler Discretization method has been used on the state-space model.

(3.3)

$$\dot{x}(k) = Ax(k) + Bu(k)$$

$$\dot{x}(k) = \frac{x(k+1) - x(k)}{T_s}$$

Equation 3.3 shows the Euler Discretization method that has been used. [21, 22 ,23, 30] If both equations in 3.3 are implemented to each other, they end up in Equation 3.4.

$$\frac{x(k+1) - x(k)}{T_s} = Ax(k) + Bu(k)$$
$$x(k+1) = T_s\big(Ax(k) + Bu(k)\big) + x(k) \tag{3.4}$$

When collecting the common terms together, it ends up like Equation 3.5.

$$x(k+1) = x(k)(AT_s + I) + u(k)(BT_s) \tag{3.5}$$

So discretization is done by that. $(AT_s + I)$ can be called as A_d and (BT_s) can be called B_d.In the output part, it is unrelated to the output's time derivation so that it can be directly discretized as $C_d = C$ and D_d=D. Collectively discrete-time state-space model can be represented as;

$$x(k+1) = x(k)A_d + u(k)B_d$$
$$y(k) = x(k)C_d + u(k)D_d \tag{3.6}$$

3.1.2. Augmented State-Space Model

The augmented state-space model has been derived from the discrete-time state-space model given in the previous section. An augmented state-space model uses the states and the system's input to control the whole system with one matrix. After this section, MPC prediction matrices are derived with this augmented matrix. The purpose of the augmentation will be more explicit in that section. Equation 3.7 shows the augmented state matrix of the system. [21, 22, 23, 31]

$$x_a(k) = \begin{bmatrix} \Delta x(k+1) \\ u(k) \end{bmatrix} \tag{3.7}$$

The state-space model can also be augmented with states and the system's outputs. The new stats of the system can be represented in Equation 3.8.

(3.8)

$$x_a(k) = \begin{bmatrix} \Delta x(k+1) \\ y(k+1) \end{bmatrix}$$

In this thesis, the system is augmented as Equation 3.7 since controlling a quadrotor requires controlling the input and the states.

$$\begin{bmatrix} \Delta x(k+1) \\ u(k) \end{bmatrix} = \begin{bmatrix} A & B \\ 0 & I \end{bmatrix} \begin{bmatrix} x(k) \\ u(k-1) \end{bmatrix} + \begin{bmatrix} x(k) \\ u(k-1) \end{bmatrix} \Delta u(k)$$
$$y(k) = \begin{bmatrix} C & 0 \end{bmatrix} \begin{bmatrix} x(k) \\ u(k-1) \end{bmatrix} \tag{3.9}$$

Equation 3.9 is the augmented form of the state-space model. The given matrices A, B, and C are the discrete forms of the quadrotor's state-space model A_d, $B_{d,}$ and C_d matrices in Equation 3.6, respectively. Therefore, equation 3.9 can be represented as a shot form as Equation 3.10.

$$\tilde{x}(k+1) = \tilde{A}\tilde{x}(k) + \tilde{B}\Delta u(k)$$
$$y(k) = \tilde{C}\tilde{x}(k) \tag{3.10}$$

3.1.3. Prediction Equations of State-Space Model

In this section, prediction equations are derived to find future input and states of the system. Future control action ad future states calculate prediction equations. Future states are found using instantaneous time sample k_i and prediction of the future control action; future states are found with these future control actions. [21, 22, 23]

$$\hat{u}(k|k) = \Delta\hat{u}(k|k) + u(k-1)$$
$$\hat{u}(k+1|k) = \Delta\hat{u}(k+1|k) + +\Delta\hat{u}(k|k) + u(k-1)$$
$$\vdots \tag{3.11}$$
$$\hat{u}(k+N_u-1|k) = \Delta\hat{u}(k+N_u-1|k) + \cdots + \Delta\hat{u}(k|k) + u(k-1)$$

Future control actions are calculated as Equation 3.11. N_u in Equation 3.11 represents the control horizon, which is the number of predicted control actions in the future.

$$\Delta u(k+i|k) = u(k+i|) - u(k+i-1|k) \tag{3.12}$$

In the literature for MPC, it is better to choose a control horizon smaller than the prediction horizon, which is the number of how many states of the future are predicted.

(3.13)

That is a problem for the prediction equations given in Equation 3.18. To solve the cause of the problem, Equation 3.13 is used for the control equation. [21, 22, 23]

$$u(k+i|k) = u(k+N_u-1), \qquad where\ N_u \le i \le N_p$$

Equation 3.13 allows us to solve the problem of unequal prediction and control horizon. State predictions needed to be calculated after finding the future control action equations.

$$\begin{aligned}
\hat{x}(k+1|k) &= Ax(k) + B[\Delta\hat{u}(k|k) + u(k-1)] \\
&= Ax(k) + B\Delta u(k|k) \\
\hat{x}(k+2|k) &= Ax(k+1) + B[\Delta\hat{u}(k+1|k) + \Delta\hat{u}(k|k) + u(k-1)] \\
&= A^2x(k) + AB\Delta u(k|k) + B\Delta u(k+i|k) \\
\hat{x}(k+3|k) &= Ax(k+2) \\
&\quad + B[\Delta\hat{u}(k+2|k) + \Delta\hat{u}(k+1|k) + \Delta\hat{u}(k|k) + u(k-1)] \\
&= A^3x(k) + A^2B\Delta u(k|k) + AB\Delta u(k+1|k) + B\Delta u(k+2|k) \\
&\vdots \\
\hat{x}\big(k+N_p\big|k\big) &= A^{N_p}x(k) + (A^{N_p-1} + \cdots + A + I)B\Delta\hat{u}(k|k) + \cdots \\
&\quad + (A^{N_p-N_c} + \cdots + A + I)B\Delta\hat{u}(k+N_c-1|k) + (A^{N_p-1} + \cdots + A \\
&\quad + I)Bu(k-1)
\end{aligned}$$

Equation 3.14 shows the prediction state equation. It is derived from previous input and states to future input. Now output equations must be derived to control the system.

$$\begin{aligned}
y(k+1|k) &= Cx(k+1|k) \\
y(k+2|k) &= Cx(k+2|k) \\
&\vdots \\
y\big(k+N_p\big|k\big) &= Cx\big(k+N_p\big|k\big)
\end{aligned} \tag{3.15}$$

Equation 3.15 shows the derivation process of the output of the system. Implementing states to the output equation ends up as Equation 3.16.

$$y(k+1|k) = CAx(k) + CB\Delta u(k|k) \tag{3.16}$$

$$y(k+2|k) = CA^2x(k) + CAB\Delta u(k|k) + CB\Delta u(k+1|k)$$

$$\vdots$$

$$y(k+N_p|k) = CA^{N_p}x(k) + CA^{N_p-1}B\Delta u(k|k) + CA^{N_p-2}B\Delta u(k+1|k) + \cdots + CA^{N_p-N_c}B\Delta u(k+N_c-1|k)$$

Equation 3.16 show the output equation of the system. From Equations 3.14 and 3.16 MPC optimization matrix can be established. Before that, these equations must mention that A, B, and C matrices are discrete-time state-space matrices.

$$\hat{y}(k) = Fx(k) + H\Delta u(k) \tag{3.17}$$

Equation 3.17 is the short version of the optimizer output.

$$\begin{bmatrix} y(k+1) \\ y(k+2) \\ y(k+3) \\ \vdots \\ y(k+N_p) \end{bmatrix} = \begin{bmatrix} CA \\ CA^2 \\ CA^3 \\ \vdots \\ CA^{N_p} \end{bmatrix} x(k) + \begin{bmatrix} CB & 0 & 0 & \dots & 0 \\ CAB & CB & 0 & \dots & 0 \\ CA^2B & CAB & CB & \dots & 0 \\ \vdots & \dots & \dots & \ddots & \vdots \\ CA^{N_p-1}B & CA^{N_p-2}B & CA^{N_p-3}B & \dots & CA^{N_p-N_c}B \end{bmatrix} \Delta u(k) \tag{3.18}$$

F and the H matrices mentioned in Equation 3.17 is derived in Equation 3.18. In the next section, an optimization equation is derived in that equation F and H matrices are used.

3.1.4. Optimization Equation

In this section, the optimization equation is derived based on the MPC optimization algorithms in the literature. [21, 22, 23] Optimization of the quadrotor's equations made by the cost function and minimizing the cost function by adding the weights on the system.

$$J = (Y - W)^T(Y - W) + \Delta U^T \lambda \Delta U \tag{3.19}$$

Equation 3.19 shows the fundamental cost function of the MPC problem. In literature, many books use the same cost function even though other more custom functions exist.

(3.20)

$$Y = Fx + H\Delta u$$

In Equation 3.17, the term W represents each time sample's reference trajectory.

$$W = \begin{bmatrix} r(k+1) \\ r(k+2) \\ r(k+3) \\ \vdots \\ r(k+N_p) \end{bmatrix} \tag{3.18}$$

W matrices cover the reference trajectories for the $N_p$time sample, but on the other hand, in this thesis reference, W is used as step input. That is made by taking the first reference trajectory and using the same trajectory for other time samples. This approach is made for only one step. In the next step, the following reference trajectory is used like the other one. From that approach, each time step approaches, the desired reference is targeted.

$$\begin{aligned} J &= (Fx + H\Delta u - W)^T (Fx + H\Delta u - W) + \Delta u^T \lambda \Delta u \\ &= x^T F^T Fx + x^T F^T H\Delta u - x^T F^T W + \Delta u^T H^T Fx + \Delta u^T H^T H\Delta u \\ &\quad - \Delta u^T H^T W - W^T Fx - W^T H\Delta u + W^T W + \Delta u^T \lambda \Delta u \\ &= \Delta u^T (H^T H + \lambda)\Delta u + 2x^T F^T H\Delta u - 2W^T H\Delta u + x^T F^T Fx - 2x^T F^T W \\ &\quad + W^T W \end{aligned} \tag{3.19}$$

Equation 3.19 is the open form of the cost function used on the quadrotor's optimization problem. Equation 3.19 can be represented as Equation 3.20, given below. [21, 22, 23]

$$J = \Delta u^T M \Delta u + 2f^T \Delta u + b \tag{3.20}$$

Equation 3.20 is the collective version of Equation 3.20. All the terms of Equation 3.21 are collected together through the M, f, and b terms. [21, 22, 23]

$$M = H^T H + \lambda \tag{3.21}$$

$$2f^T \Delta u = 2x^T F^T H\Delta u - 2W^T H\Delta u$$

$$b = x^T F^T Fx - 2x^T F^T W + W^T W$$

(3.22)

Equation 3.22 shows the M, f, and b terms in Equation 3.20. Optimization of the cost function, mathematically, taking the derivative of the given cost function. The quadrotor optimization problem is the minimization of the input action. That the cost function needs to be derived according to Δu and equation, it to zero so that the minimum Δu term can be found from the given equation. [21, 22, 23]

In the Equation 3.22, b term has non Δu related part. Since the cost function minimized to Δu in the derivation of the cost function b term will be zero. So that there is no need for the b term. [21, 22, 23]

$$J = \Delta u^T M \Delta u + 2 f^T \Delta u \tag{3.23}$$

At the end, the cost function is end up as Equation 3.23. [21, 22, 23]

3.1.5. Quadratic Programing on MATLAB (quadprog)

In this thesis cost function is minimized with the quadratic programing method. For that purpose, MATLAB's "quadprog" method has been used.

$$\min_x \left\{ \frac{1}{2} x^T H x + f^T x \right\} \tag{3.24}$$

MATLAB's "quadprog" method works based on the Equation 3.24. In the previous section cost function is derived for this form of the quadratic programing equation. [7]

$$\begin{gathered} \min_{\Delta u} \left\{ \frac{J}{2} = \frac{1}{2} \Delta u^T M \Delta u + \frac{2}{2} f^T \Delta u \right\} \\ \min_{\Delta u} \left\{ J = \frac{1}{2} \Delta u^T M \Delta u + f^T \Delta u \right\} \end{gathered} \tag{3.25}$$

For quadrotor control M and f terms must be calculated in order to minimize the cost function and find the optimized input for trajectory tracking. Equation 3.25 shows the MPC cost function that is used in MATLAB code.

3.2. Position Controller

In this section, position controller design process has been given. Position controller controls the U_1 control input and finds the ϕ and θ angles based on the speeds and the U_1 control input. Those angles after used in MPC controller. There are Two controller is tried for position controller. One is feedback linearization and the other is PID

(3.26)

controller. In this thesis, PID controller is given because of it is more straightforward method for a controller.

$$u = P + I + D$$

$$P = k_p e(t)$$

$$I = k_i \int e(t-1)dt$$

$$D = k_d \dot{e}(t)$$

Fundamental format of the PID controller is given in Equation 3.26 and 2.27. If Equation 3.27 is implemented into the Equation 3.26 it is end up with the Equation 3.28.

$$u(t) = k_p e(t) + k_i \int e(t-1)dt + k_d \dot{e}(t) \quad (3.28)$$

Equation 3.28 shows the total control input calculation equation for PID control. The term "e" in the Equation 3.28 is represent to error of the given control surface.

$$e(t) = x_d(t) - x(t) \quad (3.29)$$

Where;

$x_d(t)$: Desired state at t^{th} sample time

$x(t)$: Measured (Calculated) state at t^{th} sample time

All the equations need design the position controller is derived. Now position controller can be design.

Position controller controls the ϕ and θ angles and the U_1 (Thrust) of the quadrotor in this thesis. For that purpose, there are three controller algorithms needed to measure the control input and necessary angles.

$$u_1(t) = \frac{\ddot{z}m + mg}{\cos(\phi)\cos(\theta)} \quad (3.29)$$

Equation 3.29 is derived from the Equation 2.32. It needs to find the U_1 input of the quadrotor. If PID controller is applied on the Equation 3.29, it is end like Equation 3.30. [32, 33]

(3.30)

$$u_1(t) = \frac{m}{\cos(\phi)\cos(\theta)} \left(k_p e_{\ddot{z}}(t) + k_i \int_0^t e_{\ddot{z}}(t)dt + k_d \frac{e_{\ddot{z}}(t) - e_{\ddot{z}}(t-1)}{dt} + mg \right)$$

Equation 3.30 is the controller algorithm of U_1 control input. This is the first part of the controller; second part of the controller is as mentioned before measuring the desired ϕ and θ angles. For that purpose, there are two more equations derived.

$$\begin{aligned} \theta_d(t) &= k_p(x_d^e - x^e) + k_i \int_0^t (x_d^e - x^e)dt + k_d(\dot{x}_d^e - \dot{x}^e) \\ \phi_d(t) &= k_p(y_d^e - y^e) + k_i \int_0^t (y_d^e - y^e)dt + k_d(\dot{y}_d^e - \dot{y}^e) \end{aligned} \tag{3.31}$$

For measuring algorithm of the desired ϕ and θ angles are given in the Equation 3.31. [32, 33] In the Equation 3.31, the terms subscript "d" stands for the desired and the superscript "e" stands for earth frame coordinates.

After the Equation 3.31 all the needed equations are derived for the quadrotor controller. In the next section, simulation of the quadrotors trajectory tracking control methods will be given.

In the MATLAB code for position controller feedback linearization method has been used for simplicity.

$$\begin{aligned} k_1 &= -2 \\ k_2 &= -3 \end{aligned} \tag{3.32}$$

In feedback linearization implemented same as the PID controller but only differences the constant terms in Equation 3.32 is multiplied with the position errors.

4. SIMULATION OF THE QUADROTOR CONTROL

This section is about how the simulation of the trajectory tracking of the quadrotor handed. For the simulation quadrotor's control, MATLAB has been used. Typically controlling the quadrotor, there is one sensor dynamics, and some filters must be used for the appropriate solution. In this thesis, no experiment is made in real life, so those parts are not implemented in the system. But one extra part is implemented in the system to simulate the results. That is the nonlinear dynamics calculator of the quadrotor. All the essential equations are derived in Equation 2.33. In order to simulate the quadrotor, the system needs to calculate all twelve equations given in Equation 2.33. But there is an issue for that purpose. All the equations in Equation 2.33 are calculated with time derivatives. So that in the simulations, those terms are deviated in order to find the solution those terms needed to be integrated. But it is not easy for a computer to integrate a term. Likely there are some algorithms has been made to calculate the integration approximately. In MATLAB, "ode23, ode113, ode89, ode45, etc." more methods have been developed. [7] In this thesis, the ode45 method has been used.

4.1. Ode45

Ode45 is one of the unique methods based on the Runge-Kutta integration methods. As mentioned before, it used the Dormand-Prince method as a special Runge-Kutta method. [7] The Runge-Kutta method is a numerical method that splits the given differential equation into parts and finds the numerical slope of each point based on the previous point value. In that way, the integral is calculated at a low cost. [7, 34]

$$y_{n+1} = y_n + \frac{h}{6}(k_1 + 2k_2 + 2k_3 + k_4)$$
$$t_{n+1} = t_n + h \tag{4.1}$$

Where k values can be calculated as;

$$k_1 = f(t_n, y_n)$$

$$k_2 = f(t_n + \frac{1}{2}h, y_n + \frac{1}{2}k_1)$$

$$k_3 = f(t_n + \frac{1}{2}h, y_n + \frac{1}{2}k_2) \tag{4.2}$$

$$k_4 = f(t_n + h, y_n + k_3)$$

Equation 4.1 and 4.2 is shows the Runge-Kutta method for integral calculation. Dormand-Prince Methos is the similar method but more robust. [35, 36, 37]

$$y_{n+1} = y_n + \frac{35}{384}k_1 + \frac{500}{1113}k_2 + \frac{125}{192}k_4 - \frac{2187}{6784}k_5 + \frac{11}{84}k_6$$

$$t_{n+1} = t_n + h \tag{4.3}$$

Also, the k terms can be calculated as Equation 4.4.[35,36,37]

$$k_1 = fh(t_n, y_n)$$

$$k_2 = fh(t_n + \frac{1}{5}h, y_n + \frac{1}{5}k_1)$$

$$k_3 = fh(t_n + \frac{3}{10}h, y_n + \frac{3}{40}k_1 + \frac{9}{40}k_2)$$

$$k_4 = fh(t_n + \frac{4}{5}h, y_n + \frac{44}{45}k_1 - \frac{56}{15}k_2 + \frac{32}{9}k_3) \tag{4.4}$$

$$k_5 = fh(t_n + \frac{8}{9}h, y_n + \frac{19372}{6561}k_1 - \frac{25360}{2187}k_2 + \frac{64448}{6562}k_3 - \frac{212}{729}k_4)$$

$$k_6 = fh(t_n + h, y_n + \frac{9017}{3168}k_1 - \frac{355}{33}k_2 - \frac{46732}{5247}k_3 + \frac{49}{176}k_4 - \frac{5103}{18656}k_5)$$

In MATLAB when the ode45 method has been called Equation 4.3 and 4.4 is calculated if needed some other calculation has been made. [7]

4.2. Specifications of the Parrot Mambo Mini Drone

In the introduction part, it was mentioned that in the thesis, the parrot mambo mini drone's which given in Figure 4.1 specifications had been used. The main reason to use a real commercial drone specifies that it is palled to test the controller in the Parrot Mambo mini drone in the future. The feedback controller can be developed in different ways.

Figure 4.1 Parrot Mambo Mini Drone [2]

In the Table 4.1 Parrot Mambo Mini Drone's specifications has been given.

Table 4.1 Parrot Mambo Mini Drone Specifications [2]

Specification	Variable	Unit	Value
Mass	m	kg	0.063
Rotor distance to the center of mass	l	m	0.0624
Thrust coefficient	d	Ns^2	0.0107
Drag coefficient	b	Nms^2	$0.78264x10^{-3}$
Roll moment of inertia	I_{xx}	$kgms^2$	$0.582857x10^{-4}$
Pitch moment of inertia	I_{yy}	$kgms^2$	$0.716914x10^{-4}$
Yaw moment of inertia	I_{zz}	$kgms^2$	$0.1x10^{-3}$
Motors moment of inertia	J_p	$kgms^2$	$0.1021x10^{-6}$

In the simulation MPC controller 30 prediction horizon and 3 control horizons has been chosen.

4.3. Trajectory Generation

This thesis is about the trajectory tracking control of the quadrotor so that different trajectories are essential to test the controller of the quadrotor. For that purpose, some different kinds of trajectories have been used. However, some of these trajectories are used in the literature. Each trajectory that is used in the literature is referenced. In this section, one of the trajectories and the reference values that have been used for the controller are given. For each trajectory, the calculation is given in their results part.

Table 4.2 Helix Trajectory x, y, z Calculation Table

Helix Trajectory	
$\boldsymbol{x} =$	$\sin(0.1t)$
$\boldsymbol{y} =$	$\cos(0.1t)$
$\boldsymbol{x} =$	$0.6t$

In Table 4.2 a show the helix trajectory calculation parameters and Figure 4.2 shows the helix trajectory geometry. In the next chapter simulations results has been given.

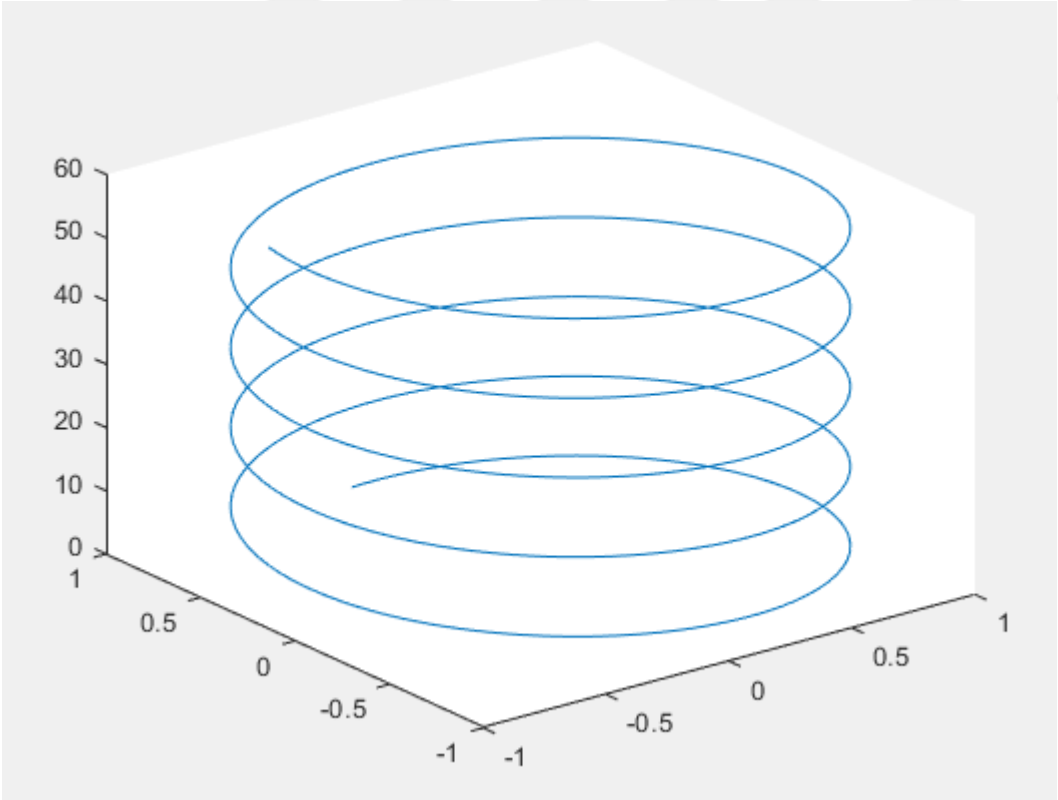

Figure 4.2 Helix Trajectory

5. SIMULATIONS RESULTS

In this section, trajectory tracking control of the quadrotor's results has been given.

5.1. Complex Helix Trajectory

This trajectory is generated with the x, y and the z equations that is given in the table 5.1. In Figure 5.1 is show the trajectory of the quadrotor and how the quadrotor follows it. Also Figure 5.2 is show the x,y, z coordinates changing in time for the quadrotor, Figure 5.3 linear velocity change in time, Figure 5.4 how the angles of the quadrotor changes in time, Figure 5.5 shows the angular velocities respect to time and the Figure 5.6 and Figure 5.7 shows the U1, U2 and U3, U4 system inpusts change in time respectively.

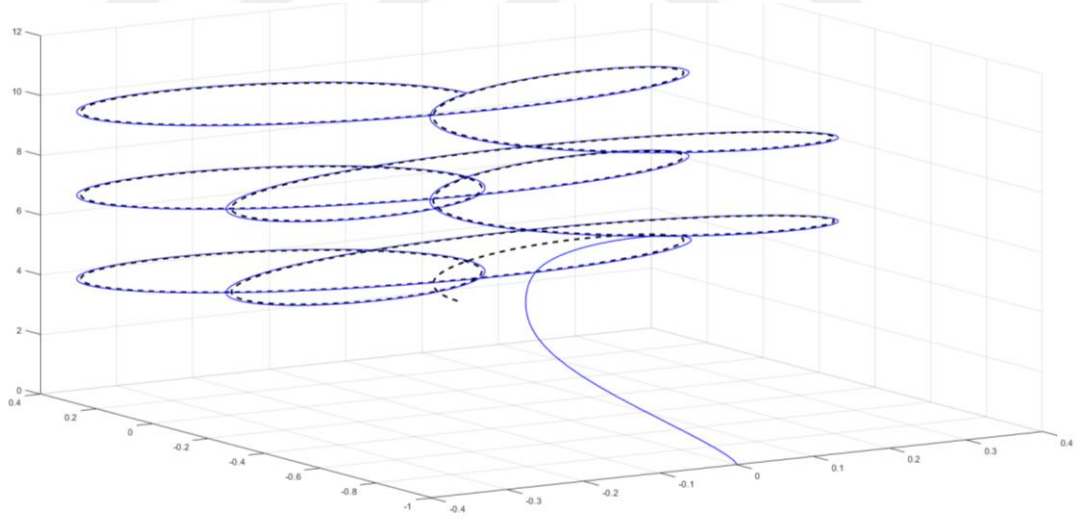

Figure 5.1 Complex Helix Trajectory

Table 5.1 Complex Helix Trajectory Generation Table [38]

Complex Helix Trajectory	
$\boldsymbol{x} =$	$\cos(0.5t) - cos^3(0.5t)$
$\boldsymbol{y} =$	$\sin(0.5t) - \sin^3(0.5t)$
$\boldsymbol{z} =$	$3 + \frac{7}{test\ time} * t$

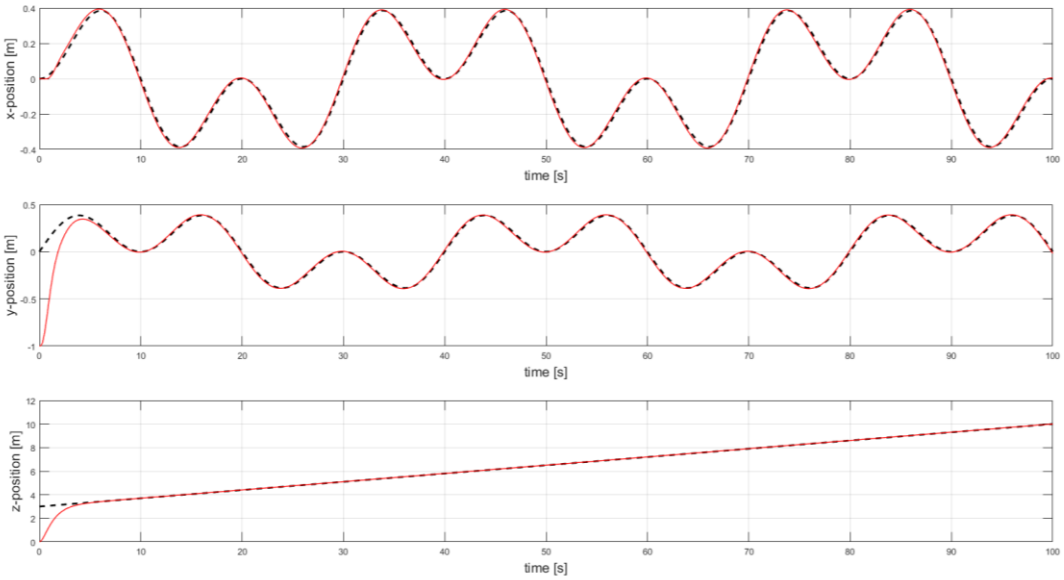

Figure 5.2 Position of the Quadrotor (Complex Helix Trajectory)

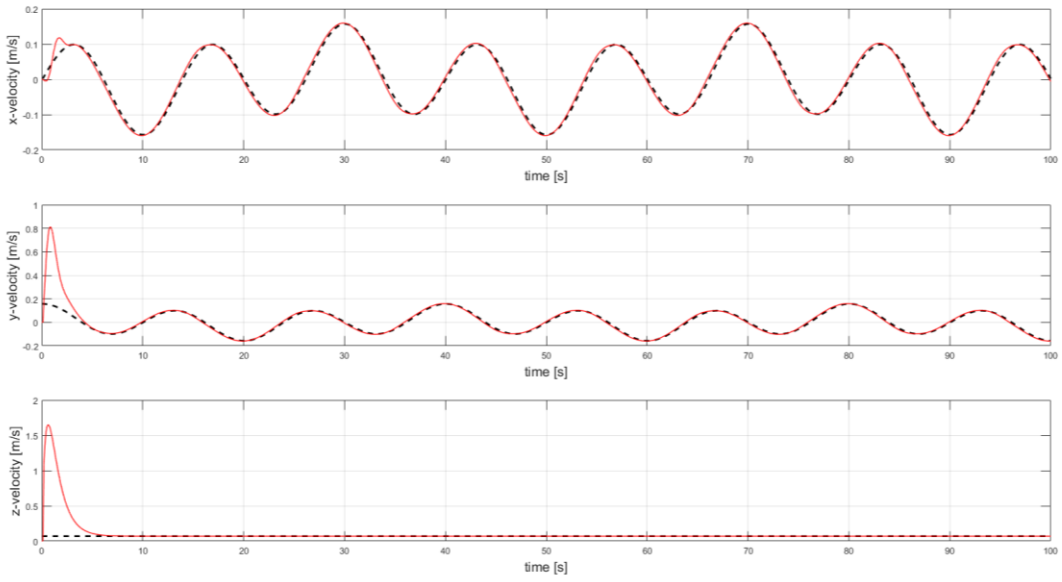

Figure 5.3 Linear Velocities of the Quadrotor (Complex Helix Trajectory)

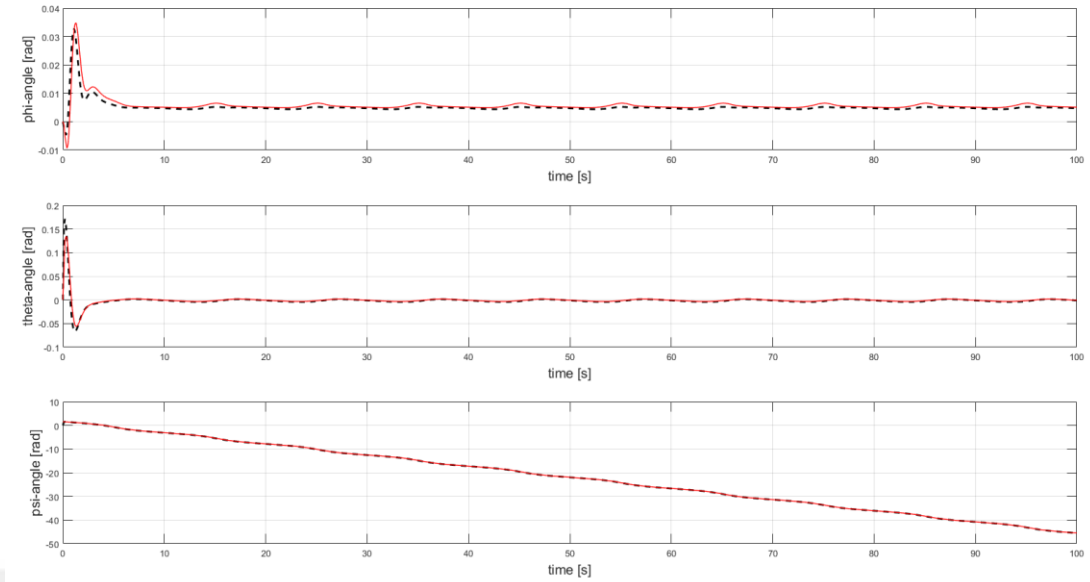

Figure 5.4 Angles of the Quadrotor (Complex Helix Trajectory)

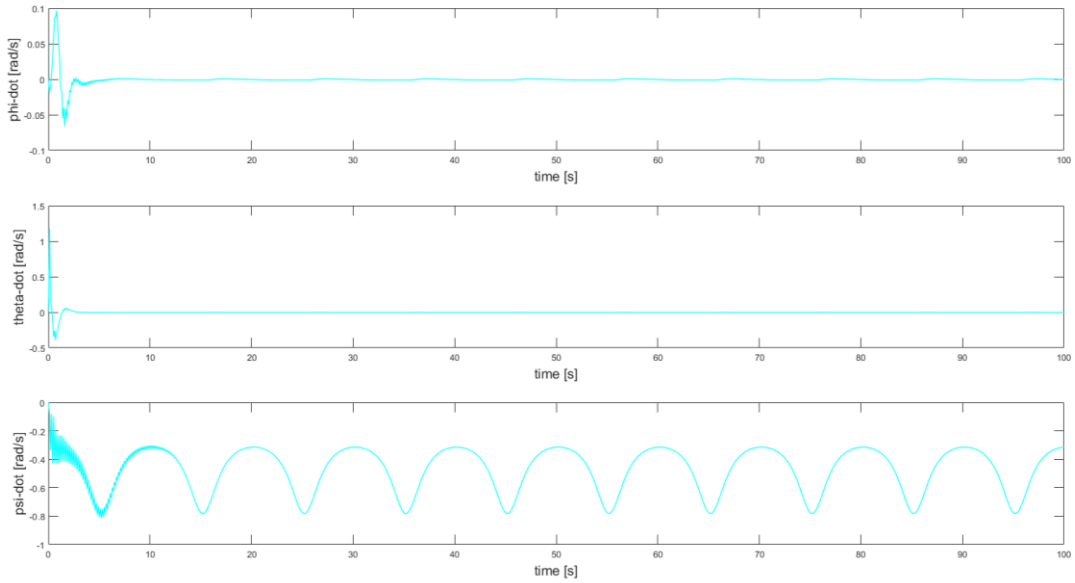

Figure 5.5 Angular Velocities of the Quadrotor (Complex Helix Trajectory)

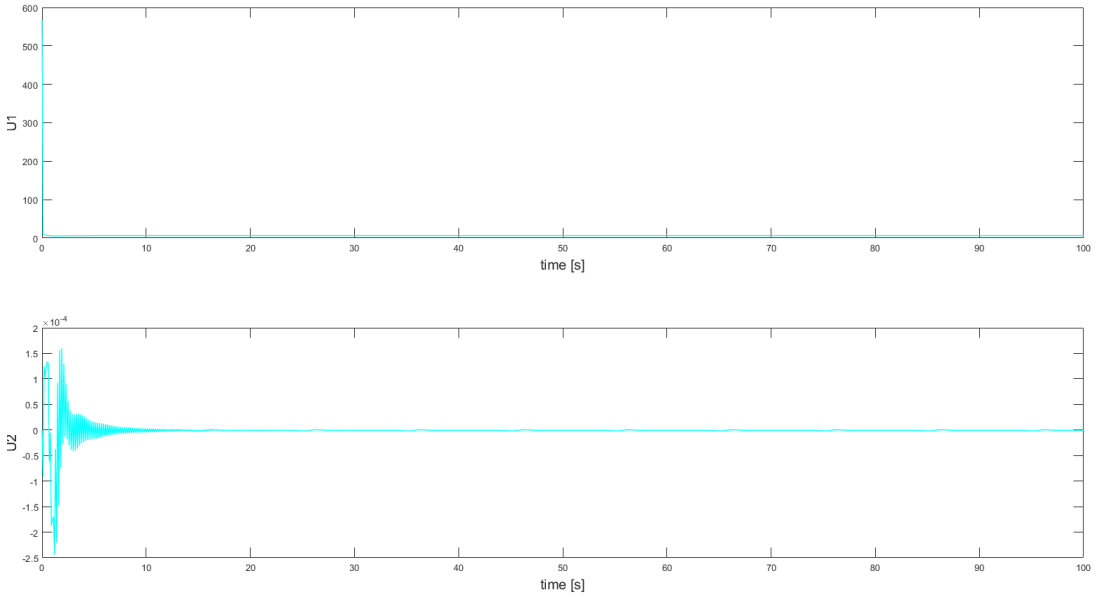

Figure 5.6 Inputs of the Quadrotor U1, U2 (Complex Helix Trajectory)

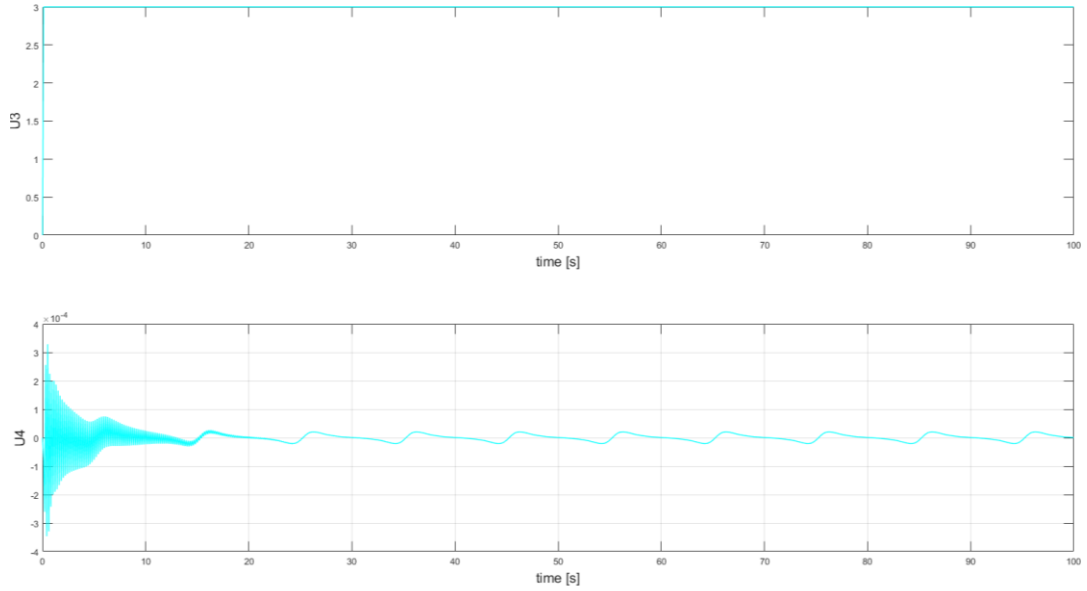

Figure 5.7 Inputs of the Quadrotor U3, U4 (Complex Helix Trajectory)

5.2. Broken Line Trajectory

Broken line trajectory is a two-line trajectory held together. The name is not on the literature it is a made-up name. In Figure 5.1 is show the trajectory of the quadrotor and how the quadrotor follows it and in Table 5.2 shows the how it created.

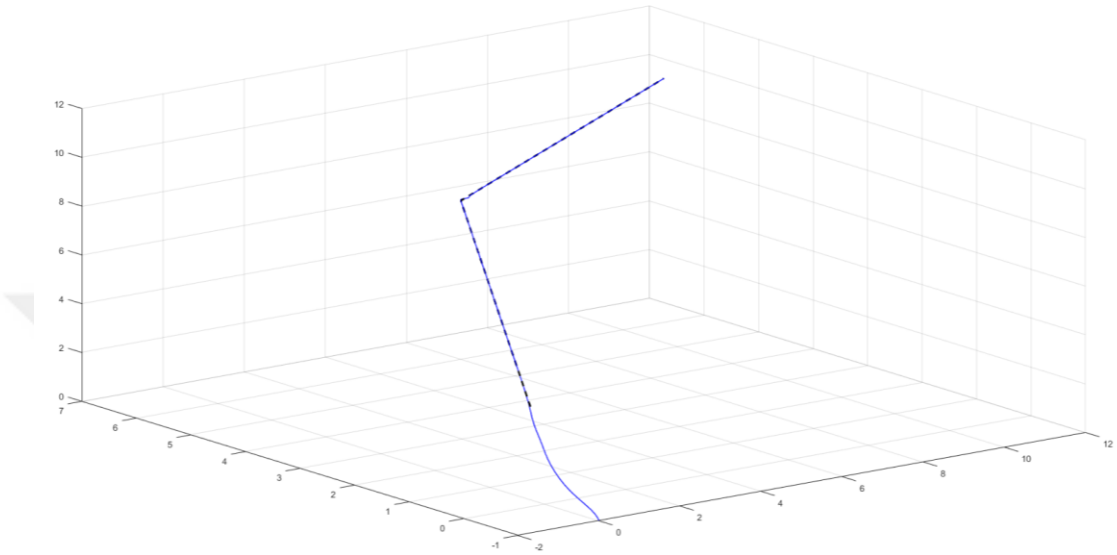

Figure 5.8 Broken Line Trajectory

Also same as the previous trajectory tracking results Figure 5.9 is show the x,y, z coordinates changing in time for the quadrotor, Figure 5.10 linear velocity change in time, Figure 5.11 how the angles of the quadrotor changes in time, Figure 5.12 shows the angular velocities respect to time and the Figure 5.13 and Figure 5.14 shows the U1, U2 and U3, U4 system inputs change in time respectively.

Table 5.2 Broken Line Trajectory Generation Table

Broken Line Trajectory	
$\boldsymbol{x} =$	$2\frac{\mathrm{t}}{20} + 1$
$\boldsymbol{y} =$	$\frac{2\mathrm{t}}{20} + 1, \quad 0 < t < \frac{test\ time}{2}$ $\frac{test\ time}{20} + 1, \quad t > \frac{test\ time}{2}$
$z =$	$3 + \frac{7}{test\ time} * t$

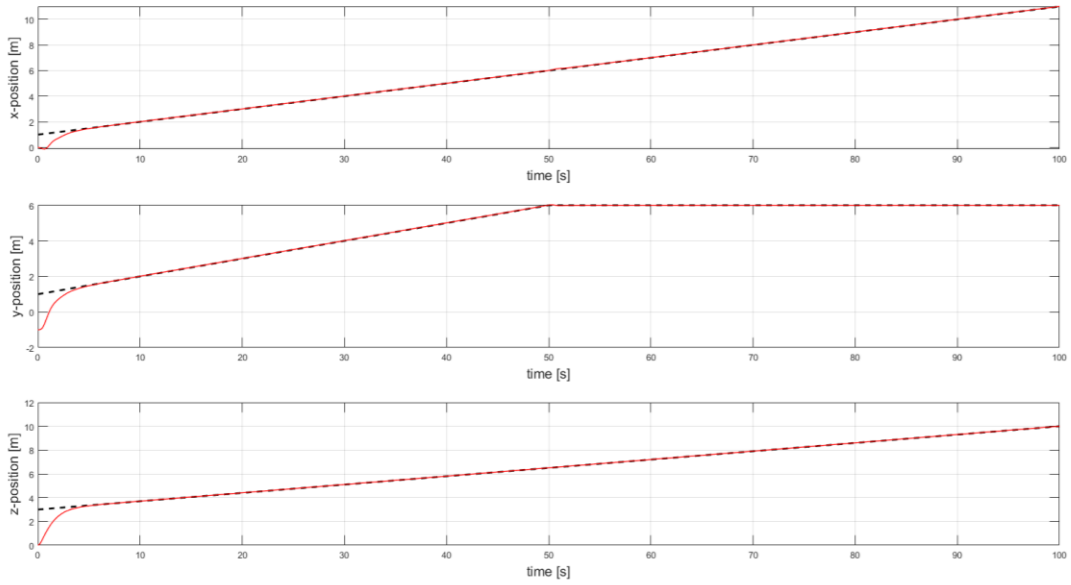

Figure 5.9 Position of the Quadrotor (Broken Line Trajectory)

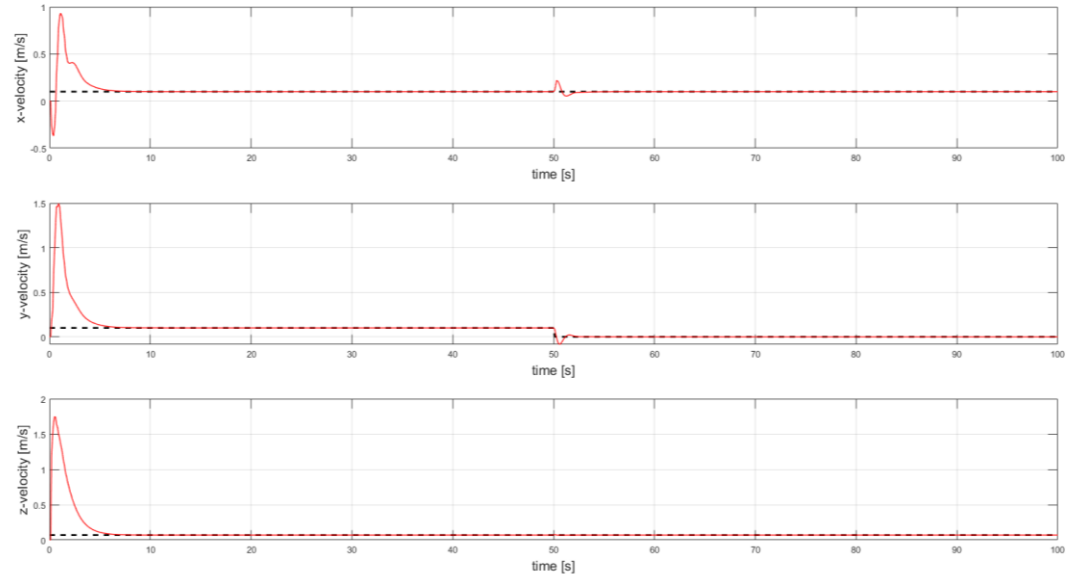

Figure 5.10 Linear Velocities of the Quadrotor (Broken Line Trajectory)

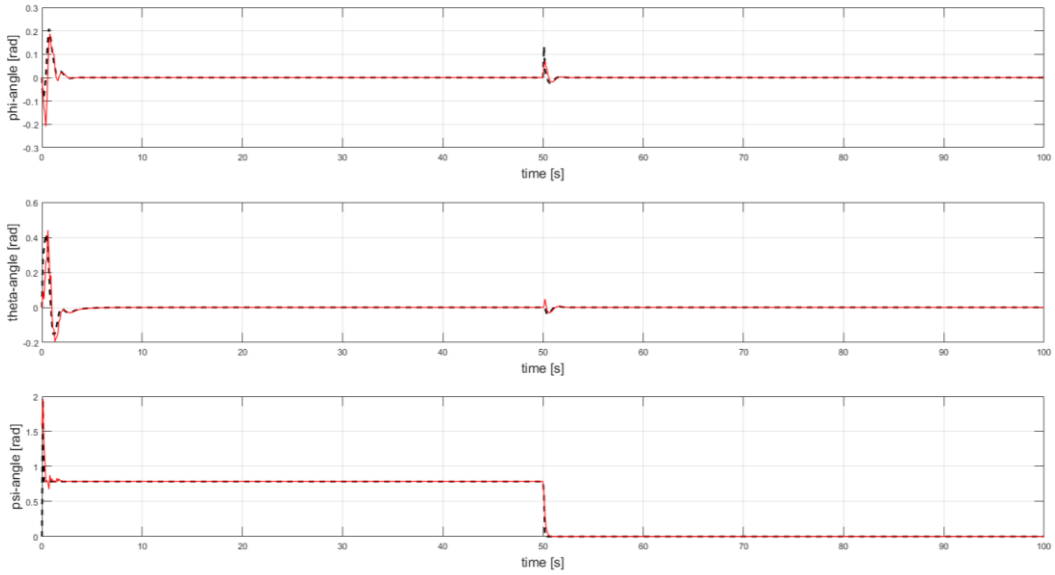

Figure 5.11 Angles of the Quadrotor (Broken Line Trajectory)

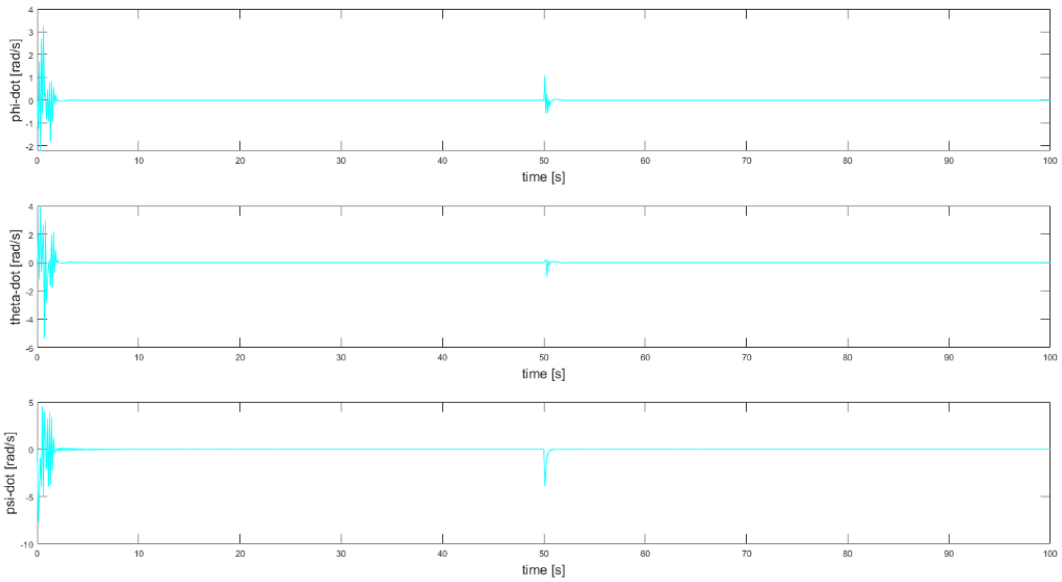

Figure 5.12 Angular Velocities of the Quadrotor (Broken Line Trajectory)

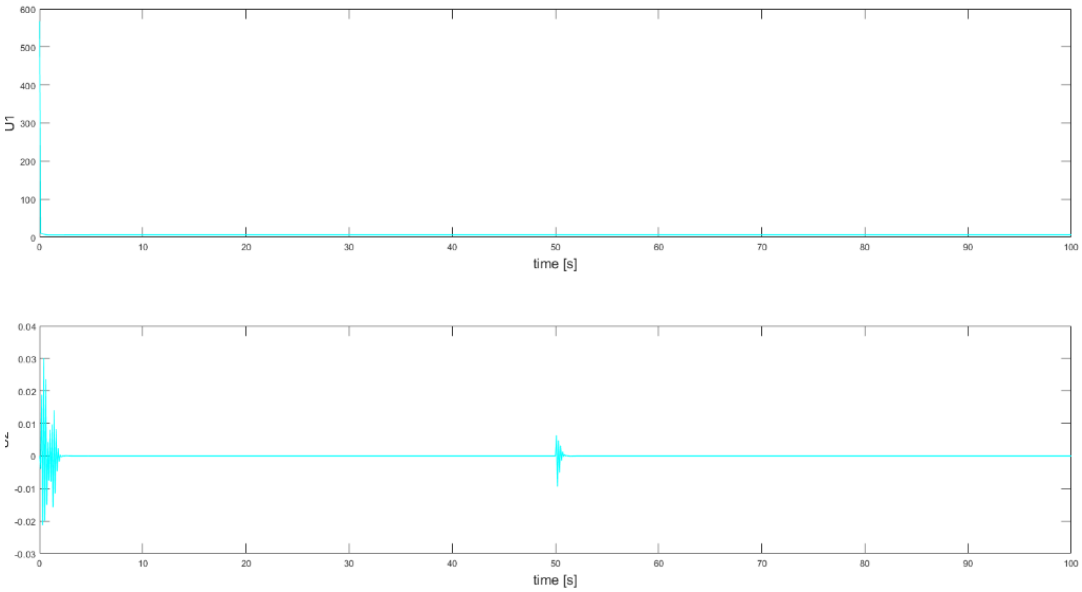

Figure 5.13 Inputs of the Quadrotor U1, U2 (Broken Line Trajectory)

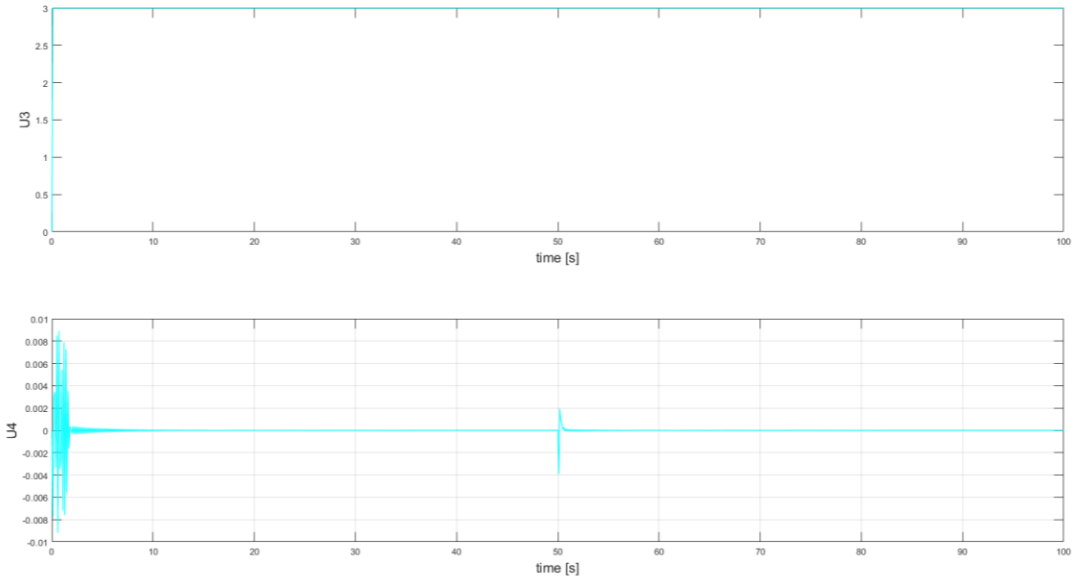

Figure 5.14 Inputs of the Quadrotor U3, U4 (Broken Line Trajectory)

6. CONCLUSION

In this works quadrotor's trajectory tracking performance is tested with the LPV-MPC controller. It seems that trajectory tracking performance is quite well after the quadrotor catches the trajectory for the first time. However, it took some time to catch the trajectory for the quadrotor. It may cause form the cost function of the MPC controller. One other cause for this problem is the position controller performance. Firstly, the PID controller is designed for the position controller, but for computational simplicity, it switches to the feedback linearization controller. It is simpler. Also, a slow rise time problem may cause that controller.

Model predictive control over all performance is quite good. It follows the trajectory almost perfectly. The controller tested with different trajectories to ensure performance, but almost all the trajectories had the same results. Also, the LPV method has been improving performance. So even if it is not seen comparatively in this thesis, it is tested. One problem for Theis thesis is that there is no real-time experiment made. So it may perform well on the computer, but it may be computationally complex for small embedded hardware on the quadrotor.

In future work, all the given controller pieces, like standard MPC, PID, and Feedback linearization, are competitively planned to be shown in graphs. Also, All these controllers will be experimentally validated to ensure performance.

REFERENCES

[1] **Kaplan, M. R., Eraslan, A., Beke, A., & Kumbasar, T**. (2019). Altitude and position control of Parrot Mambo Minidrone with PID and Fuzzy Pid Controllers. *2019 11th International Conference on Electrical and Electronics Engineering (ELECO)*. https://doi.org/10.23919/eleco47770.2019.8990445

[2] **Noordin, A., Basri, M. A., & Mohamed, Z.** (2020). Simulation and experimental study on PID control of a quadrotor MAV with perturbation. *Bulletin of Electrical Engineering and Informatics*, *9*(5), 1811–1818. https://doi.org/10.11591/eei.v9i5.2158

[3] **Yu, R., Guo, H., Sun, Z., & Chen, H.** (2015). MPC-based Regional Path Tracking Controller design for Autonomous Ground Vehicles. *2015 IEEE International Conference on Systems, Man, and Cybernetics*. https://doi.org/10.1109/smc.2015.439

[4] **Zhao, W., & Go, T. H.** (2014). Quadcopter Formation flight control combining MPC and robust feedback linearization. *Journal of the Franklin Institute*, *351*(3), 1335–1355. https://doi.org/10.1016/j.jfranklin.2013.10.021

[5] **López-Estrada, F. R., Ponsart, J.-C., Theilliol, D., Zhang, Y., & Astorga-Zaragoza, C.-M.** (2015). LPV model-based tracking control and robust sensor fault diagnosis for a quadrotor UAV. *Journal of Intelligent & Robotic Systems*, *84*(1–4), 163–177. https://doi.org/10.1007/s10846-015-0295-y

[6] **Alcalá, E., Puig, V., & Quevedo, J.** (2019). LPV-MPC control for Autonomous Vehicles. *IFAC-PapersOnLine*, *52*(28), 106–113. https://doi.org/10.1016/j.ifacol.2019.12.356

[7] *MATLAB Help*. Documentation - MATLAB & Simulink. (n.d.). https://www.mathworks.com/help/

[8] **Singhal, G., Bansod, B., & Mathew, L**. (2018). *Unmanned Aerial Vehicle Classification, Applications and Challenges: A Review*. https://doi.org/10.20944/preprints201811.0601.v1

[9**] James Rennie8 November 2016 in Articles, & Rennie, J.** (2022, May 9). *Drone types: Multi-rotor, fixed-wing, single rotor, hybrid vtol*. AUAV. https://www.auav.com.au/articles/drone-types/

[10] Parrot MambUser Guideo Mini Drone . (n.d.). https://www.parrot.com/assets/s3fs-public/2021-09/mambo_quick-start-guide_uk-fr-sp-de-it-nl-pt-ar_0.pdf

[11] **Boon, M. A., Drijfhout, A. P., & Tesfamichael, S.** (2017). Comparison of a fixed-wing and multi-rotor UAV for environmental mapping applications: A case study. *The International Archives of the Photogrammetry, Remote Sensing and Spatial Information Sciences*, *XLII-2/W6*, 47–54. https://doi.org/10.5194/isprs-archives-xlii-2-w6-47-2017

[12] **Zhang, B., Song, Z., Zhao, F., & Liu, C.** (2022). Overview of propulsion systems for Unmanned Aerial Vehicles. *Energies*, *15*(2), 455. https://doi.org/10.3390/en15020455

[13] **Wikimedia Foundation.** (2023, May 13). *Unmanned Combat Aerial Vehicle*. Wikipedia. https://en.wikipedia.org/wiki/Unmanned_combat_aerial_vehicle#/media/File:Bayraktar_TB2_Runway.jpg

[14] **Jalil, B., Leone, G. R., Martinelli, M., Moroni, D., Pascali, M. A., & Berton, A.** (2019). Fault detection in power equipment via an unmanned aerial system using multi modal data. *Sensors*, *19*(13), 3014. https://doi.org/10.3390/s19133014

[15] **Zong, J., Zhu, B., Hou, Z., Yang, X., & Zhai, J.** (2021). Evaluation and comparison of Hybrid Wing VTOL UAV with four different electric propulsion systems. *Aerospace*, *8*(9), 256. https://doi.org/10.3390/aerospace8090256

[16] *Figure of Parrot Mambo Mini Drone*. Parrot Mambo Mini Drone. (n.d.). https://www.alibaba.com/product-detail/100-Original-PARROT-Mambo-Mini-RC_60741405474.html

[17] **Amin, R., Aijun, L., & Shamshirband, S.** (2016). A review of Quadrotor UAV: Control Methodologies and Performance Evaluation. *International Journal of Automation and Control*, *10*(2), 87. https://doi.org/10.1504/ijaac.2016.076453

[18] **Meslouli, I., Mesli, R., Kahouadji, M., Choukchou-Braham, A., & Cherki, B.** (2018). Quadrotor design procedure and PID control for Outdoor Free Flight. *2018 International Conference on Electrical Sciences and Technologies in Maghreb (CISTEM)*. https://doi.org/10.1109/cistem.2018.8613602

[19] **Xu, R., & Ozguner, U.** (2006). Sliding mode control of a quadrotor helicopter. *Proceedings of the 45th IEEE Conference on Decision and Control*. https://doi.org/10.1109/cdc.2006.377588

[20] **Maciejowski, J. M.** (2008). *Predictive control: With constraints*. Prentice Hall.

[21] **Rossiter, J. A.** (2013). *Model-based predictive control a practical approach*. CRC Press.

[22] **Camacho, E. F., & Bordons, C.** (1995). *Model predictive control in the process industry*. Springer.

[23] Model Predictive Control System Design and implementation using MATLAB®. (2009). *Advances in Industrial Control*. https://doi.org/10.1007/978-1-84882-331-0

[24] **Bouabdallah, S., & Siegwart, R.** (n.d.). Backstepping and sliding-mode techniques applied to an indoor micro quadrotor. *Proceedings of the 2005 IEEE International Conference on Robotics and Automation*. https://doi.org/10.1109/robot.2005.1570447

[25] **Bouabdallah, S., Murrieri, P., & Siegwart, R.** (2004). Design and control of an indoor micro quadrotor. *IEEE International Conference on Robotics and Automation, 2004. Proceedings. ICRA '04. 2004*. https://doi.org/10.1109/robot.2004.1302409

[26] **Deveerasetty, K. K., Zhou, Y., Yang, Z., & Wu, Q.** (2018). Robust control design for the trajectory tracking of a quadrotor. *2018 IEEE International Conference on Cyborg and Bionic Systems (CBS)*. https://doi.org/10.1109/cbs.2018.8612221

[27] **Mokhtari, A., & Benallegue, A.** (2004). Dynamic feedback controller of Euler angles and wind parameters estimation for a quadrotor unmanned aerial vehicle. *IEEE International Conference on Robotics and Automation, 2004. Proceedings. ICRA '04. 2004*. https://doi.org/10.1109/robot.2004.1307414

[28] **Suthanthira Vanitha, N., Manivannan, L., Meenakshi, T., & Radhika, K.** (2020). Stability Analysis of quadrotor using state space mathematical modeling. *Materials Today: Proceedings*, *33*, 4040–4043. https://doi.org/10.1016/j.matpr.2020.06.428

[29] **Khan, H. S., & Kadri, M. B.** (2014). Position control of quadrotor by embedded PID control with hardware in loop simulation. *17th IEEE International Multi Topic Conference 2014*. https://doi.org/10.1109/inmic.2014.7097372

[30] **Zeigler, B. P., Muzy, A., & Kofman, E.** (2019). Modeling formalisms and their simulators. *Theory of Modeling and Simulation*, 43–91. https://doi.org/10.1016/b978-0-12-813370-5.00011-0

[31] **Pannocchia, G.** (2015). Offset-free tracking MPC: A tutorial review and comparison of different formulations. *2015 European Control Conference (ECC)*. https://doi.org/10.1109/ecc.2015.7330597

[32] **Li, J., & Li, Y.** (2011). Dynamic Analysis and PID control for a quadrotor. *2011 IEEE International Conference on Mechatronics and Automation*. https://doi.org/10.1109/icma.2011.5985724

[33] **Says:, Y., Says:, C., & says:, O. M.** (n.d.). *Quadrotor control system design - position, attitude, and Motor Control*. Wil Selby. https://wilselby.com/research/arducopter/controller-design/

[34] **Lopez, C.** (2014). *Matlab differential equations*. Apress.

[35] **Calvo, M., Montijano, J. I., & Randez, L.** (1990). A fifth-order interpolant for the Dormand and Prince Runge-Kutta method. *Journal of Computational and Applied Mathematics*, *29*(1), 91–100. https://doi.org/10.1016/0377-0427(90)90198-9

[36] Wikimedia Foundation. (2023a, April 24). *Runge–Kutta methods*. Wikipedia. https://en.wikipedia.org/wiki/Runge%E2%80%93Kutta_methods

[37] *Dormand-Prince Method*¶. Dormand-Prince Method - Numerary (latest) documentation in English. (n.d.). https://numerary.readthedocs.io/en/latest/dormand-prince-method.html

[38] **Islam, M., Okasha, M., & Idres, M. M.** (2017). Dynamics and control of Quadcopter using linear model predictive control approach. *IOP Conference Series: Materials Science and Engineering*, *270*, 012007. https://doi.org/10.1088/1757-899x/270/1/012007

APPENDICES

APPENDIX A: Helix Trajectory

APPENDIX B: Extending Helix Trajectory

APPENDIX C: Curly Line Trajectory

APPENDIX D: Circular Complex Movements Trajectory

APPENDIX A: Helix Trajectory

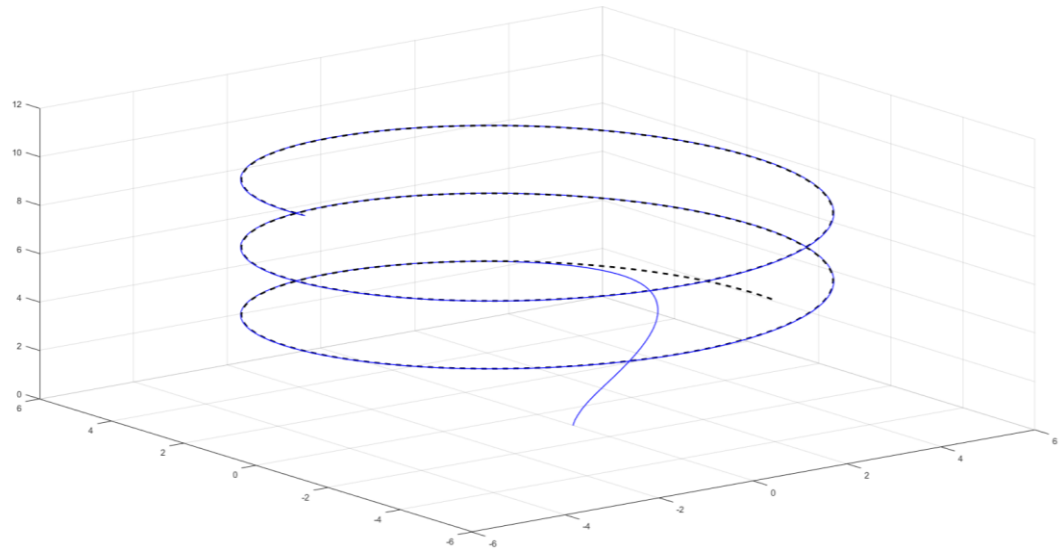

Figure A.1 Helix Trajectory

Table A.1 Helix Trajectory Generation Table [38]

Helix Trajectory	
$\boldsymbol{x} =$	$\sin(0.3t)$
$\boldsymbol{y} =$	$\cos(0.3t)$
$z =$	$3 + \frac{7}{test\ time} * t$

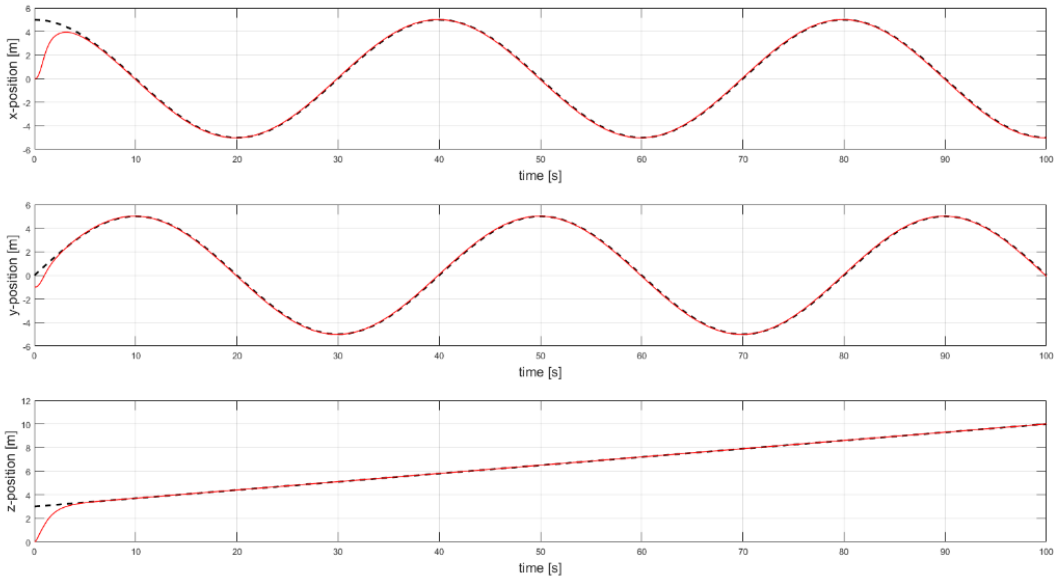

Figure A.2 Positions of the Quadrotor (Helix Trajectory)

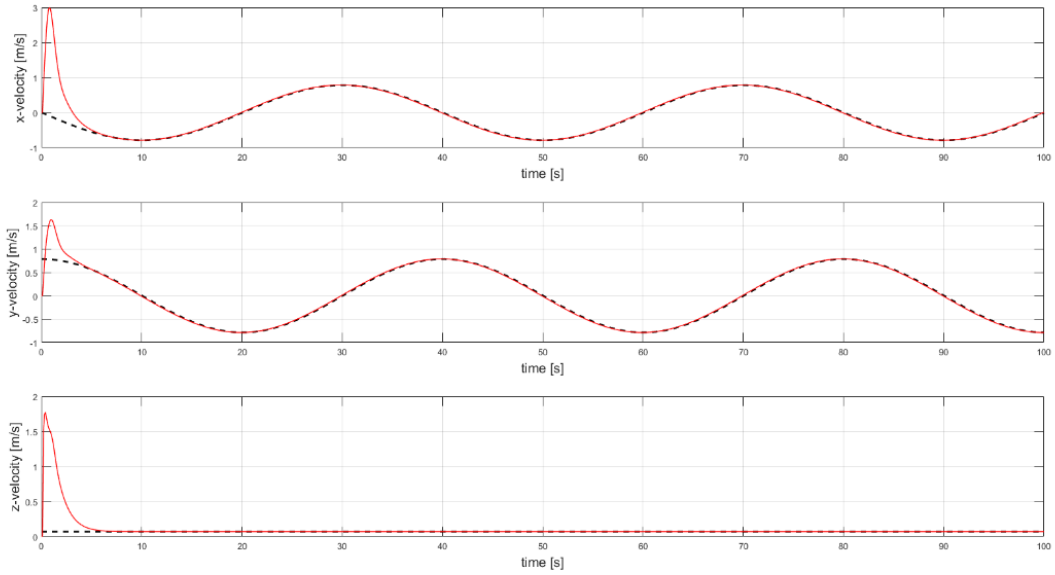

Figure A.3 Linear Velocities of the Quadrotor (Helix Trajectory)

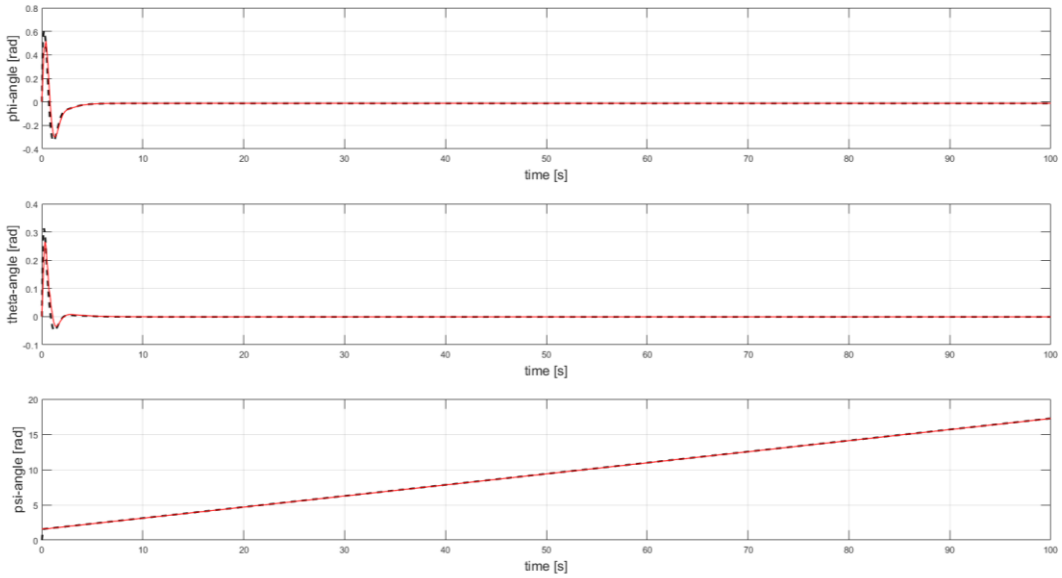

Figure A.4 Angles of the Quadrotor (Helix Trajectory)

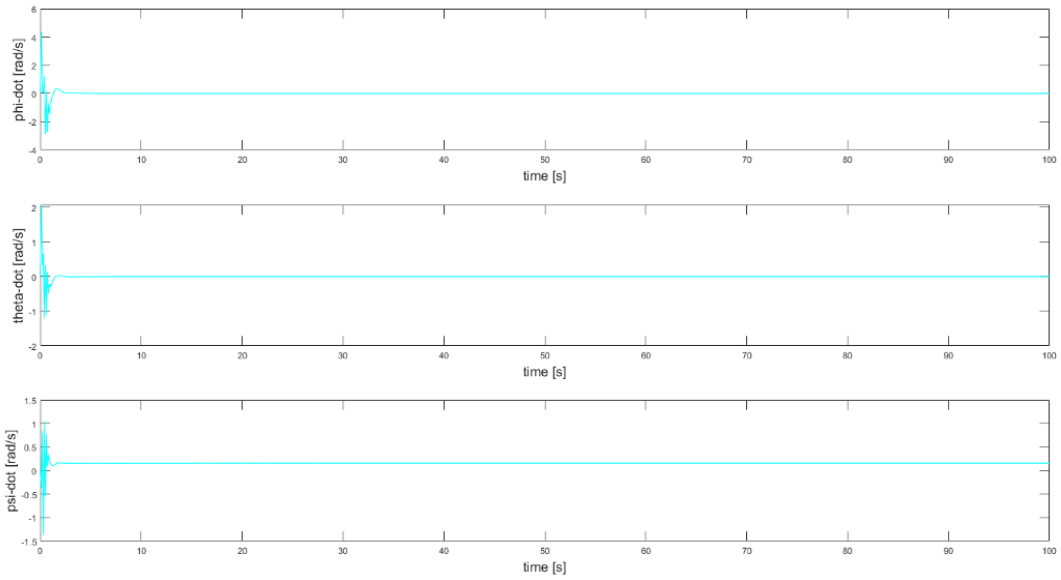

Figure A.5 Angular Velocities of the Quadrotor (Helix Trajectory)

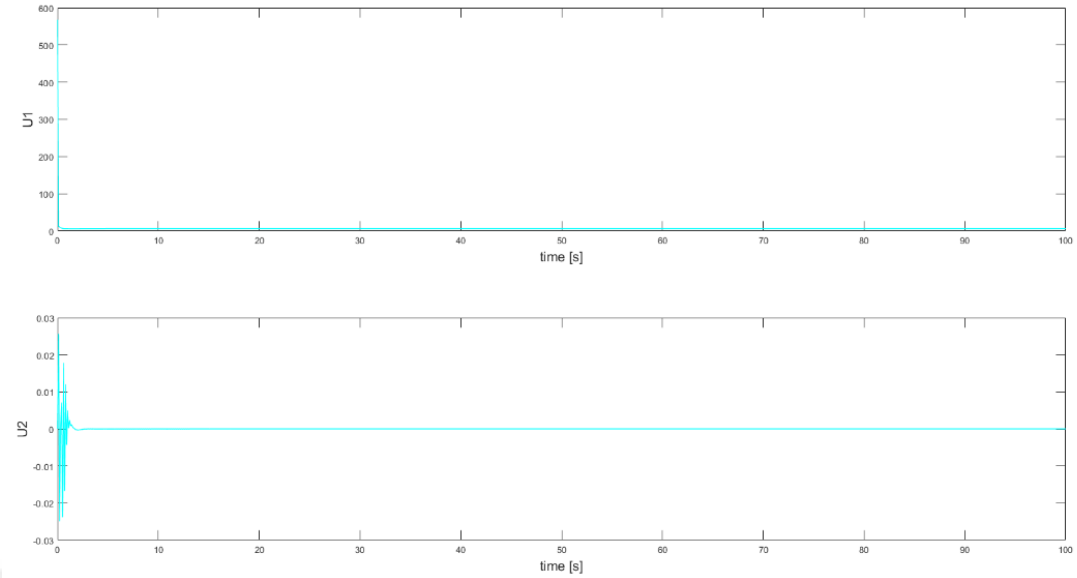

Figure A.6 Inputs of the Quadrotor U1, U2 (Helix Trajectory)

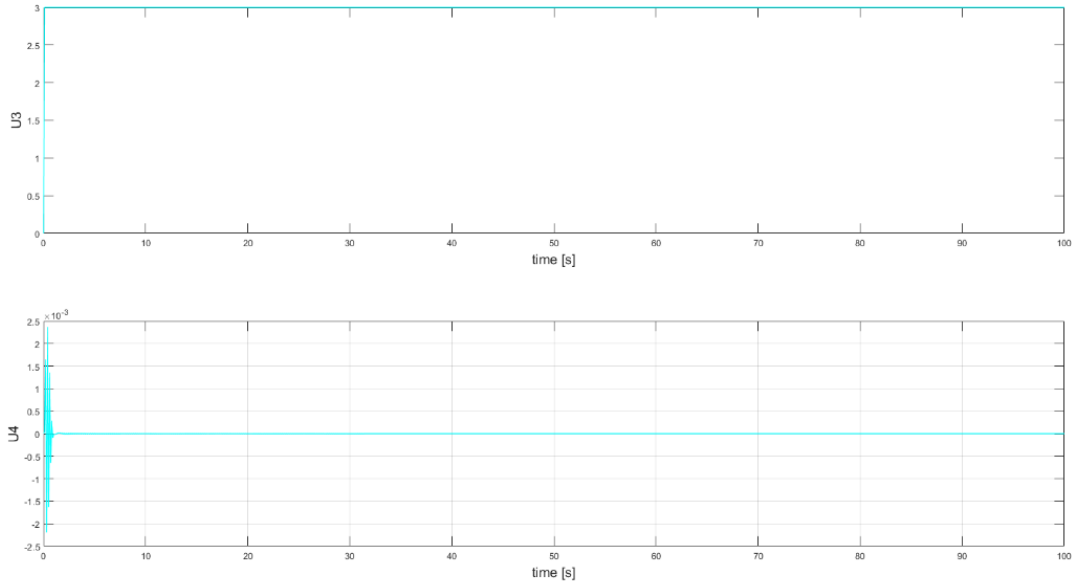

Figure A.7 Inputs of the Quadrotor U3, U4 (Helix Trajectory)

APPENDIX B: Extending Helix Trajectory

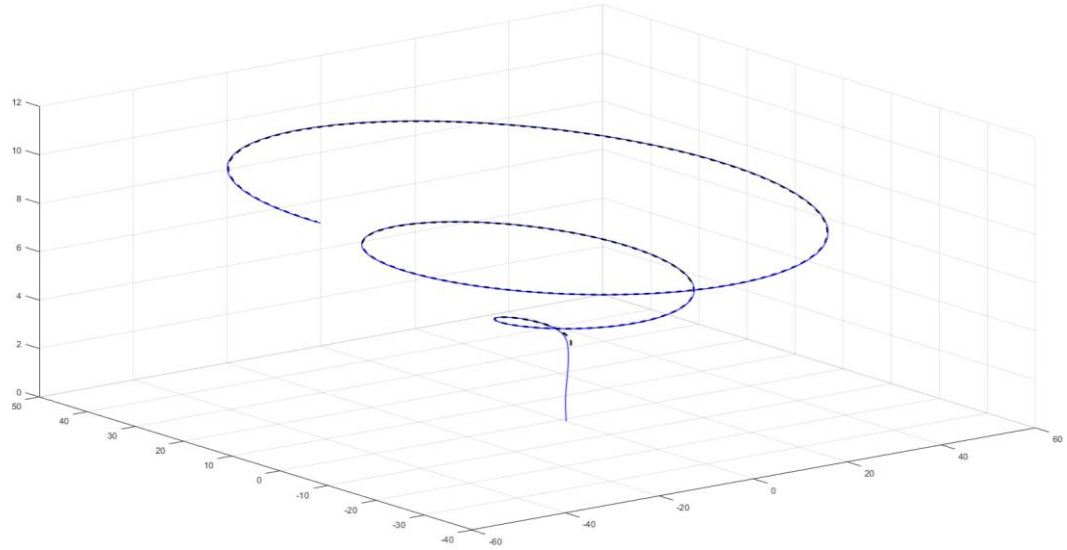

Figure B.1 Extending Helix Trajectory

Table B.1 Extending Helix Trajectory Generation Table

Extending Helix Trajectory	
$\boldsymbol{x} =$	$(0.5\mathrm{t} + 2)\cos(0.3t)$
$\boldsymbol{y} =$	$(0.5\mathrm{t} + 2)\sin(0.3t)$
$z =$	$3 + \frac{7}{test\ time} * t$

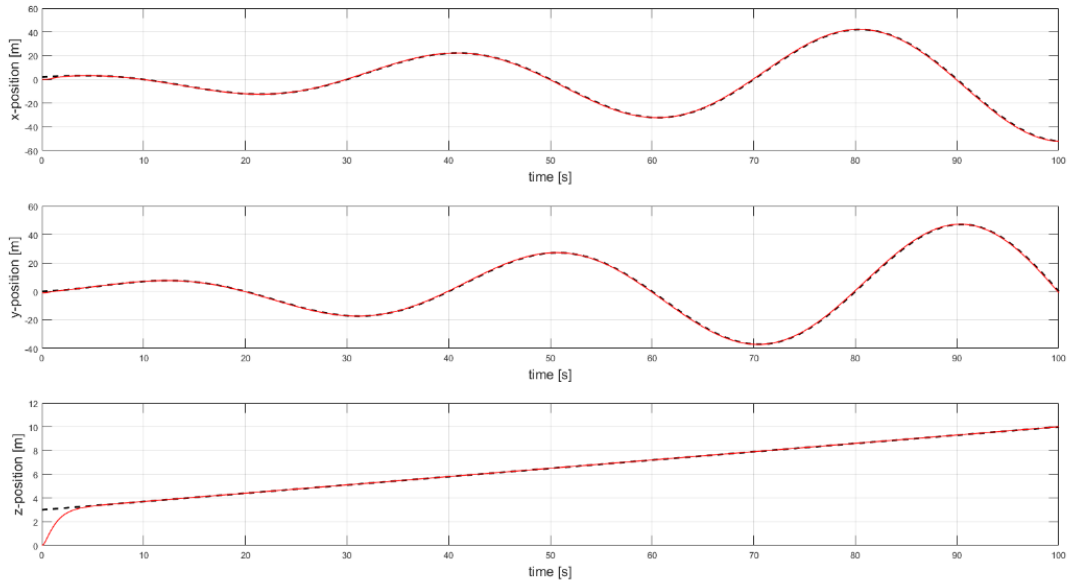

Figure B.2 Positions of the Quadrotor (Extending Helix Trajectory)

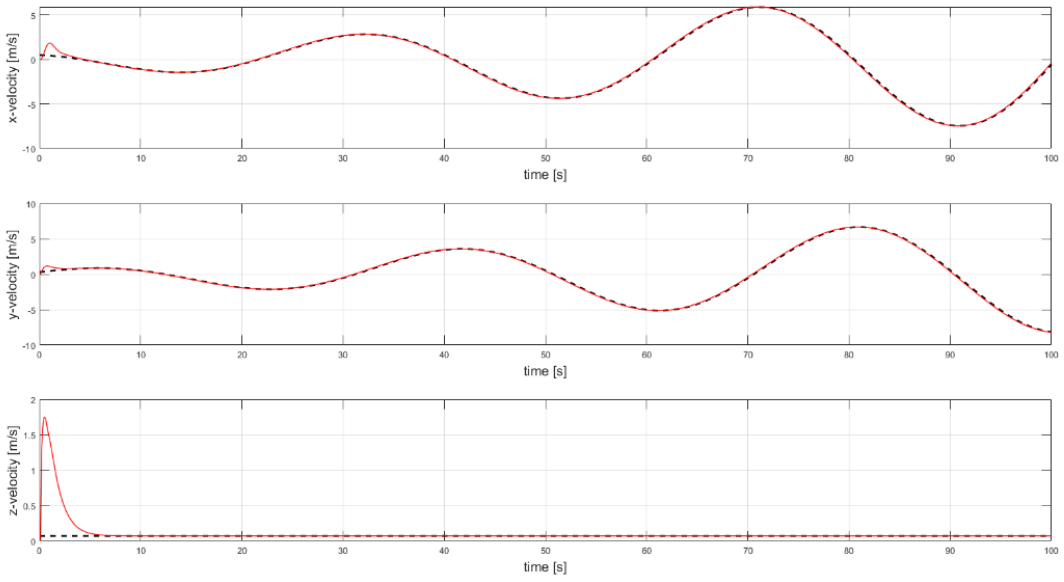

Figure B.3 Linear Velocities of the Quadrotor (Extending Helix Trajectory)

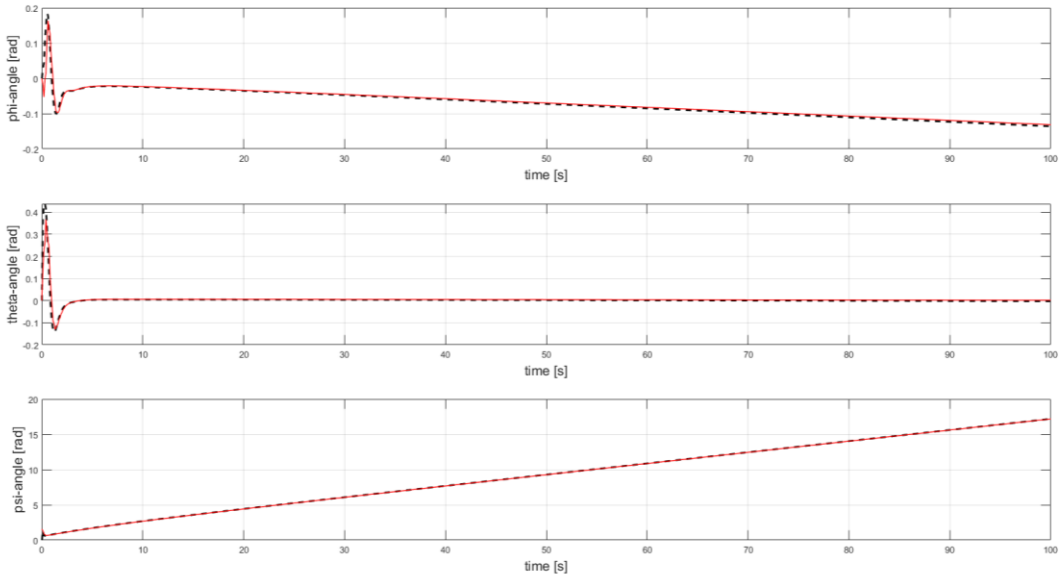

Figure B.4 Angles of the Quadrotor (Extending Helix Trajectory)

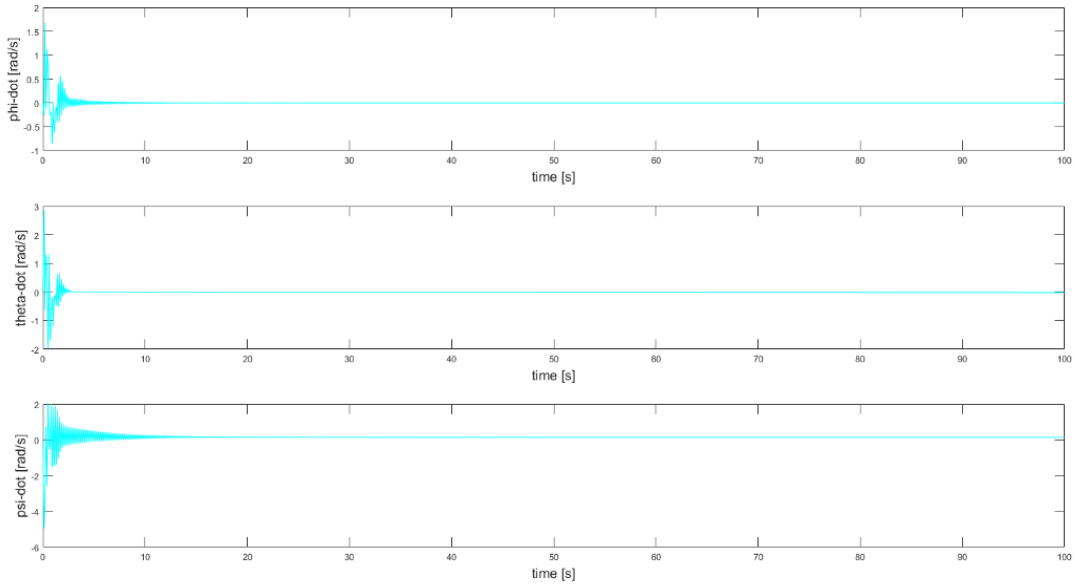

Figure B.5 Angular Velocities of the Quadrotor (Extending Helix Trajectory)

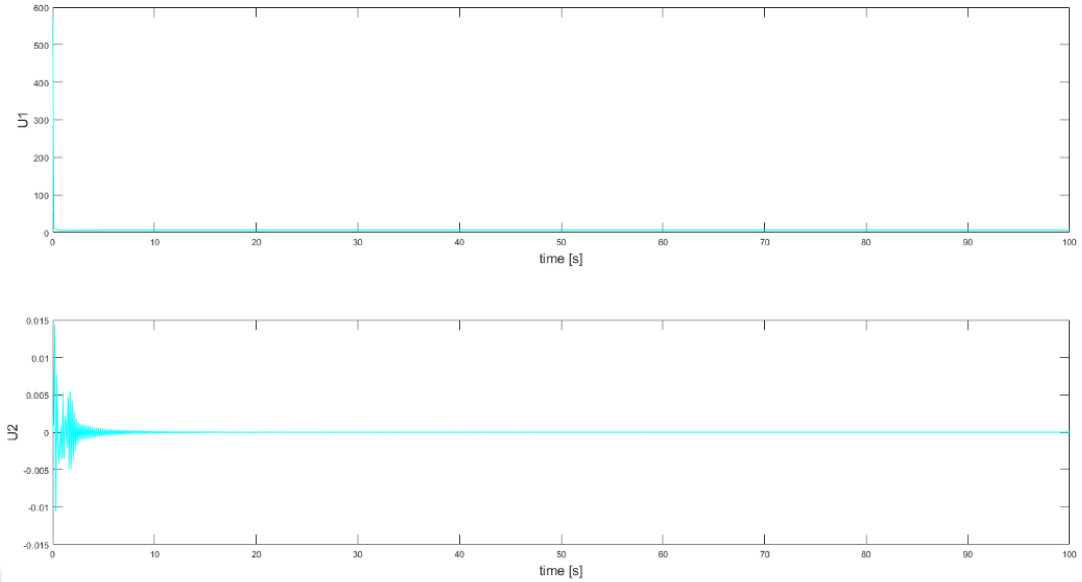

Figure B.6 Inputs of the Quadrotor U1, U2 (Extending Helix Trajectory)

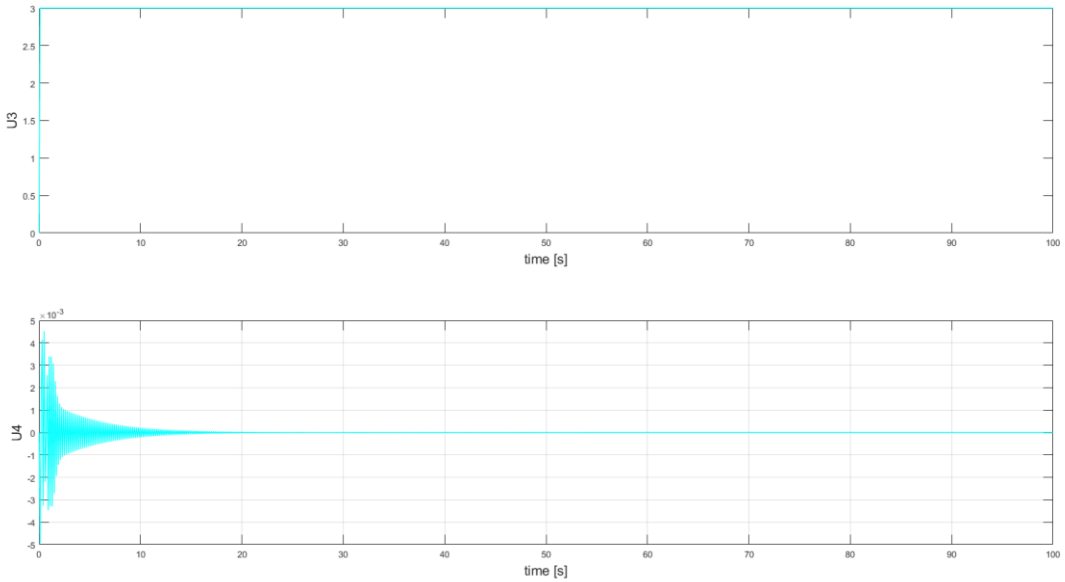

Figure B.7 Inputs of the Quadrotor U3, U4 (Extending Helix Trajectory)

APPENDIX C: Curly Line Trajectory

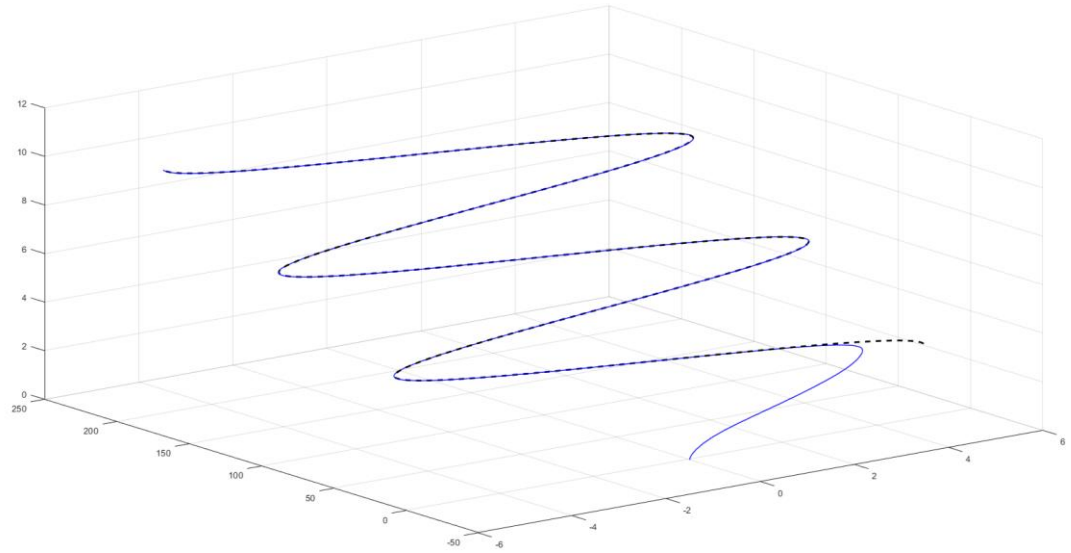

Figure C.1 Curly Line Trajectory

Table C.1 Curly Line Trajectory Generation Table

Curly Line Trajectory	
$\boldsymbol{x} =$	$\cos(0.3t)$
$\boldsymbol{y} =$	2t
$z =$	$3 + \frac{7}{test\ time} * t$

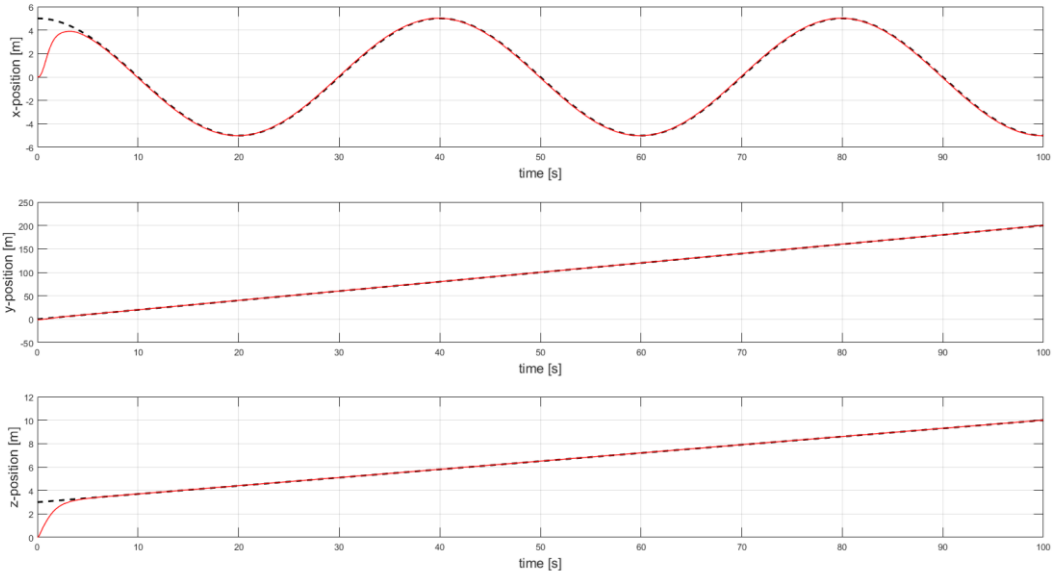

Figure C.2 Positions of the Quadrotor (Curly Line Trajectory)

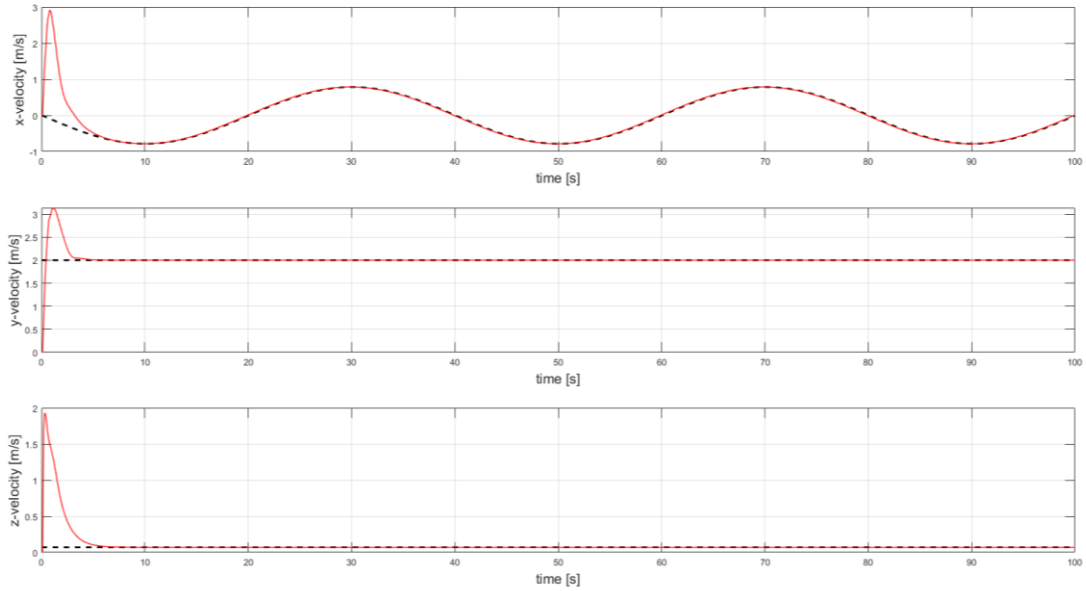

Figure C.3 Linear Velocities of the Quadrotor (Curly Line Trajectory)

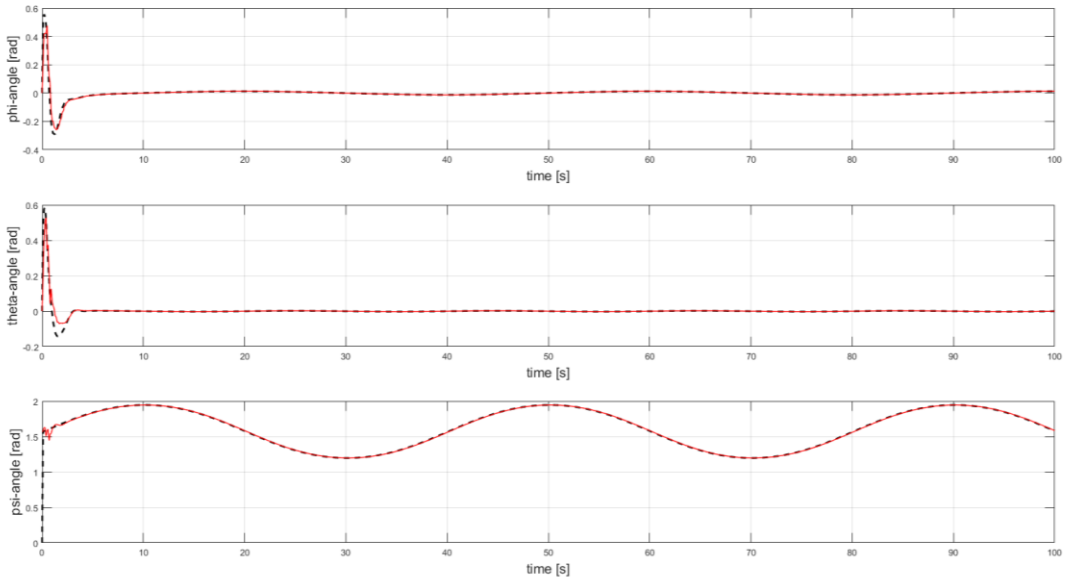

Figure C.4 Angles of the Quadrotor (Curly Line Trajectory)

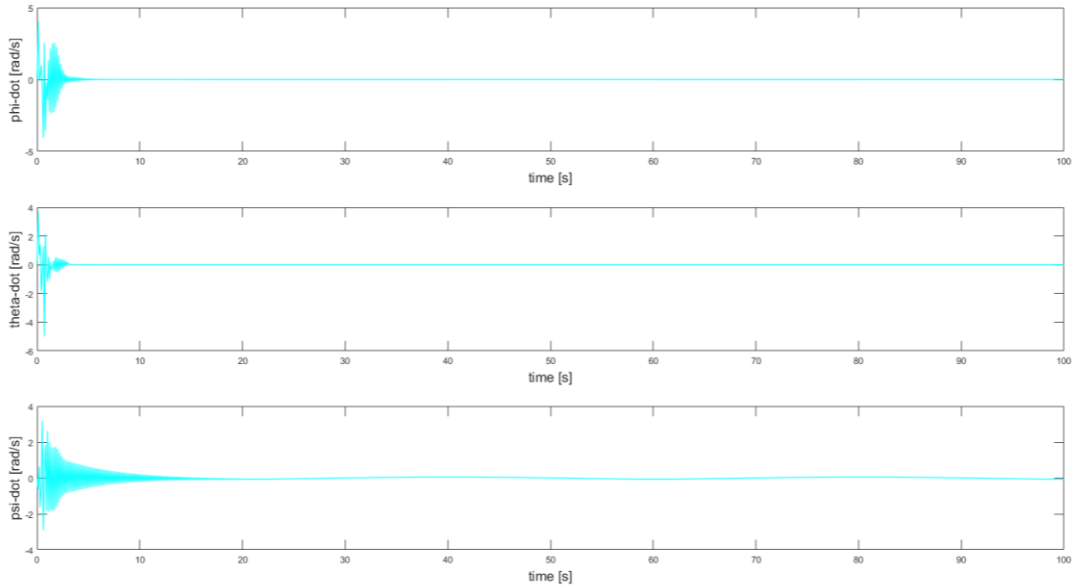

Figure C.5 Angular Speeds of the Quadrotor (Curly Line Trajectory)

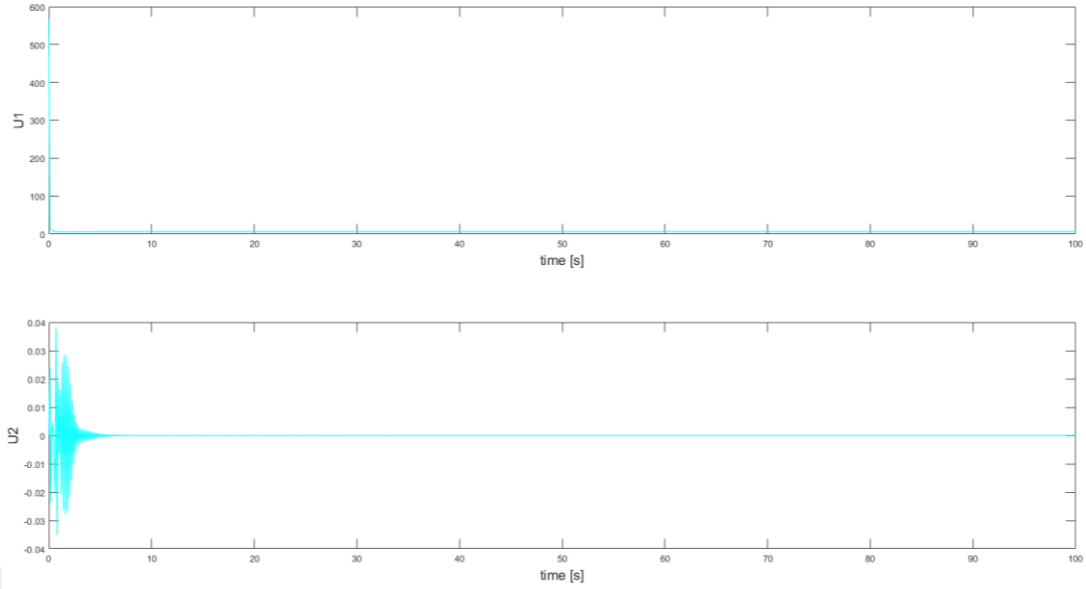

Figure C.8 Inputs of the Quadrotor U1, U2 (Curly Line Trajectory)

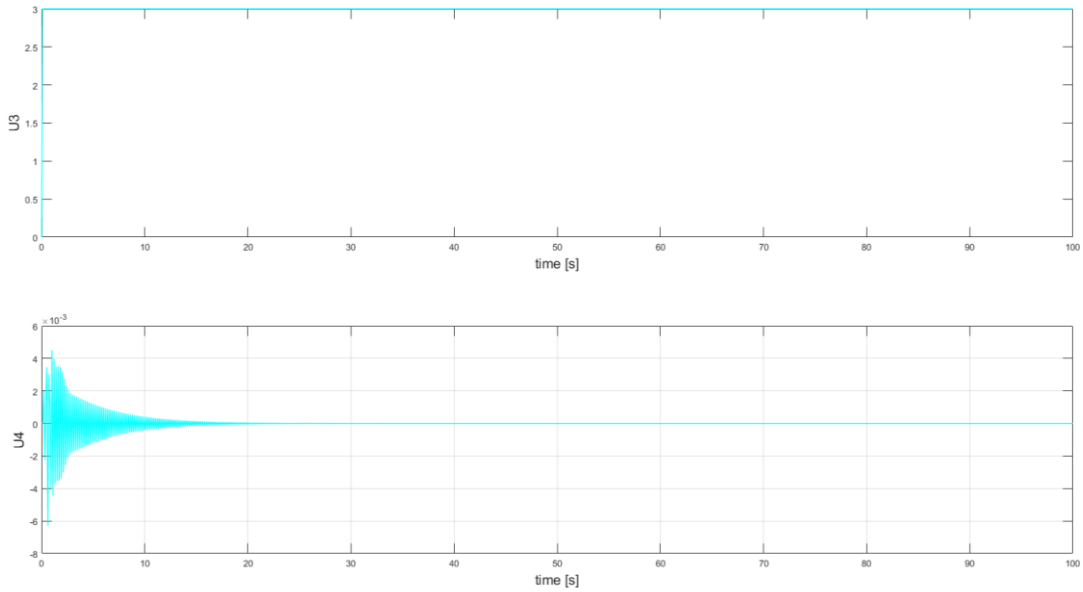

Figure C.7 Inputs of the Quadrotor U3, U4 (Curly Line Trajectory)

APPENDIX D: Circular Complex Movements Trajectory

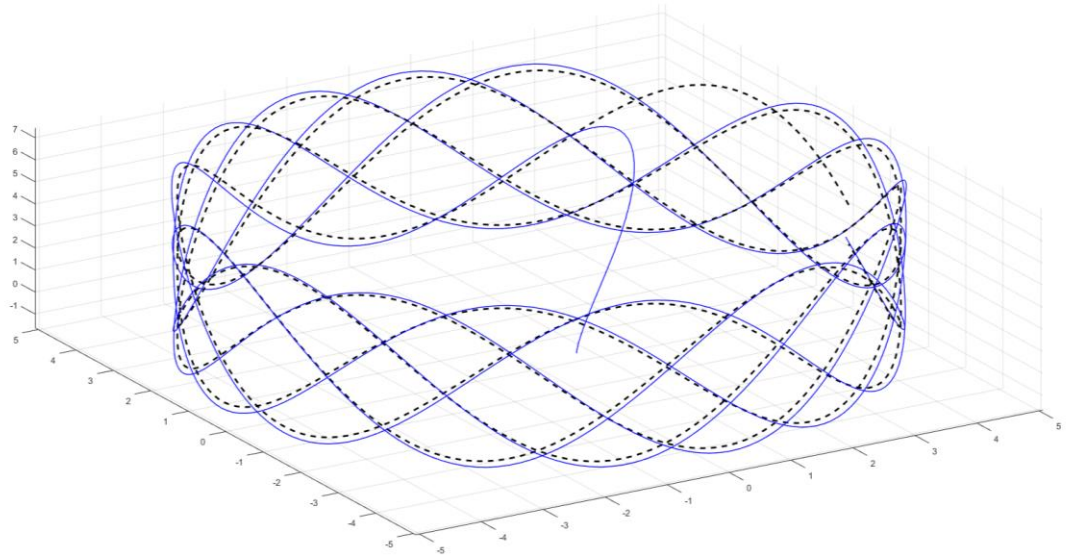

Figure D.1 Circular Complex Movements Trajectory

Table D.1 Circular Complex Movements Trajectory Generation Table [38]

Circular Complex Movements Trajectory	
$\boldsymbol{x} =$	$5\cos(0.3t)$
$\boldsymbol{y} =$	$5\sin(0.3t)$
$z =$	$3 + \frac{350}{test\ time} * \sin(t)$

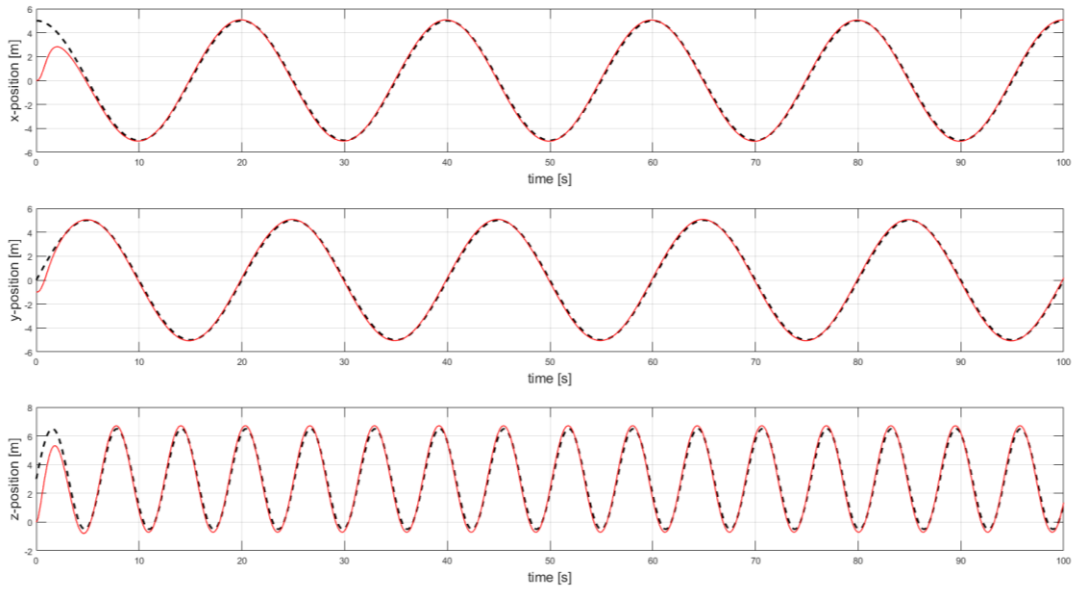

Figure D.2 Positions of the Quadrotor Circular (Complex Movements Trajectory)

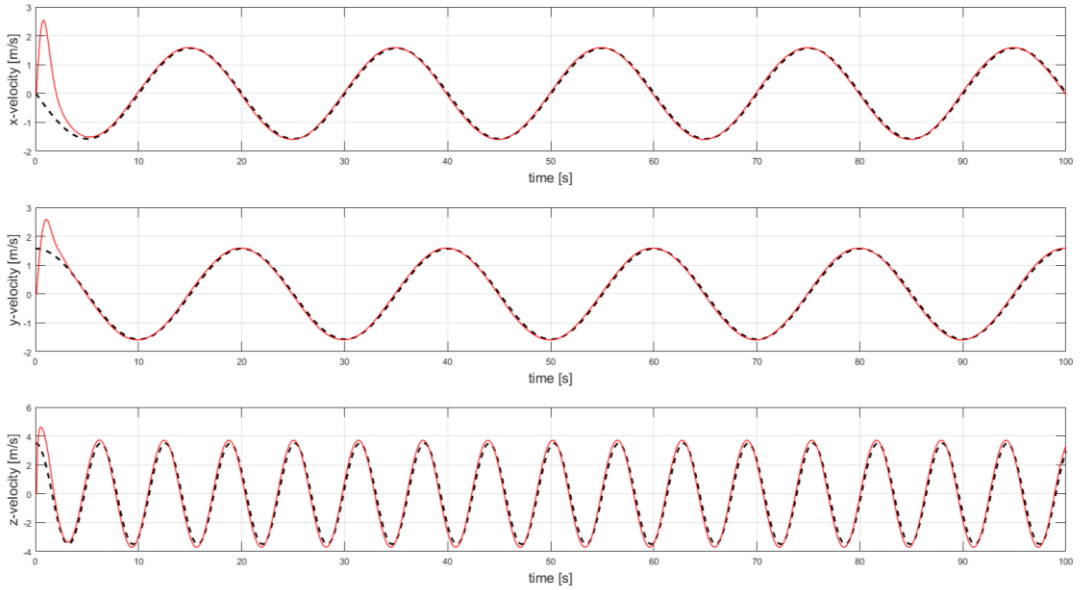

Figure D.3 Linear Velocities of the Quadrotor (Complex Movements Trajectory)

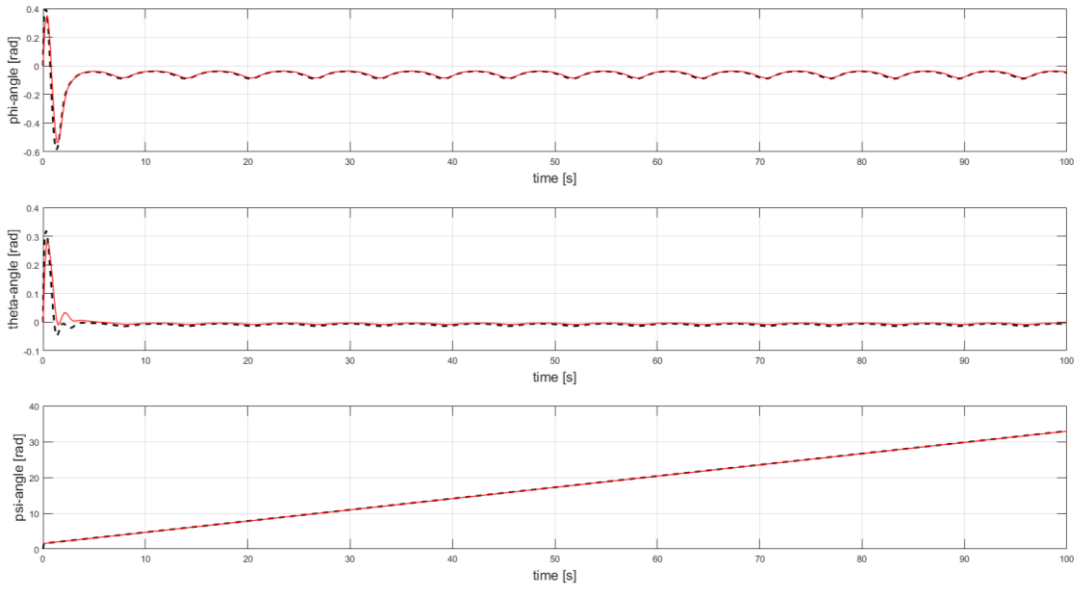

Figure D.4 Angles of the Quadrotor (Complex Movements Trajectory)

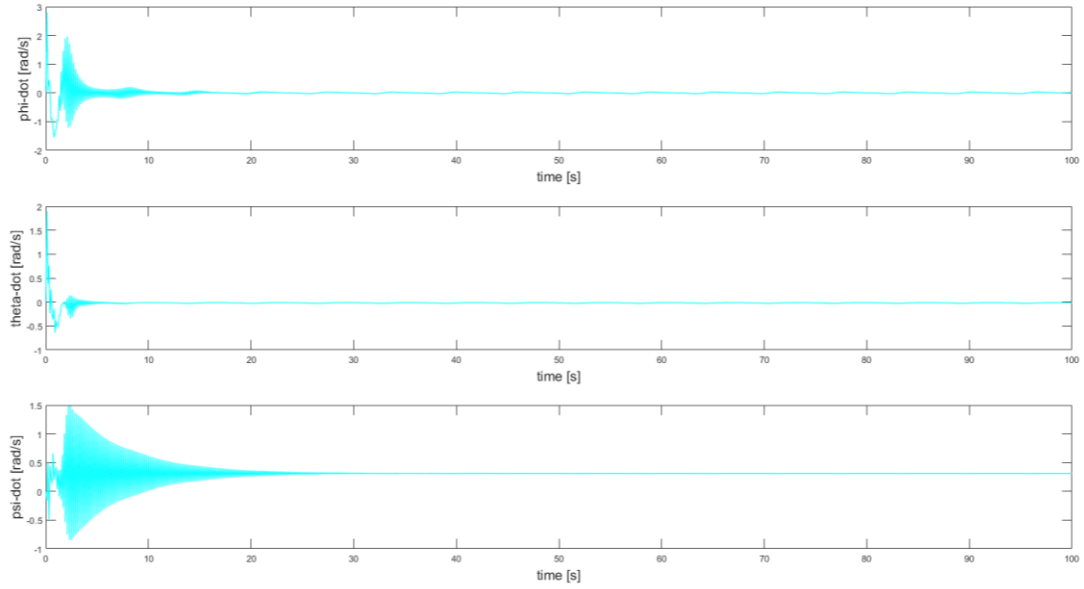

Figure D.5 Angular Velocities of the Quadrotor (Complex Movements Trajectory)

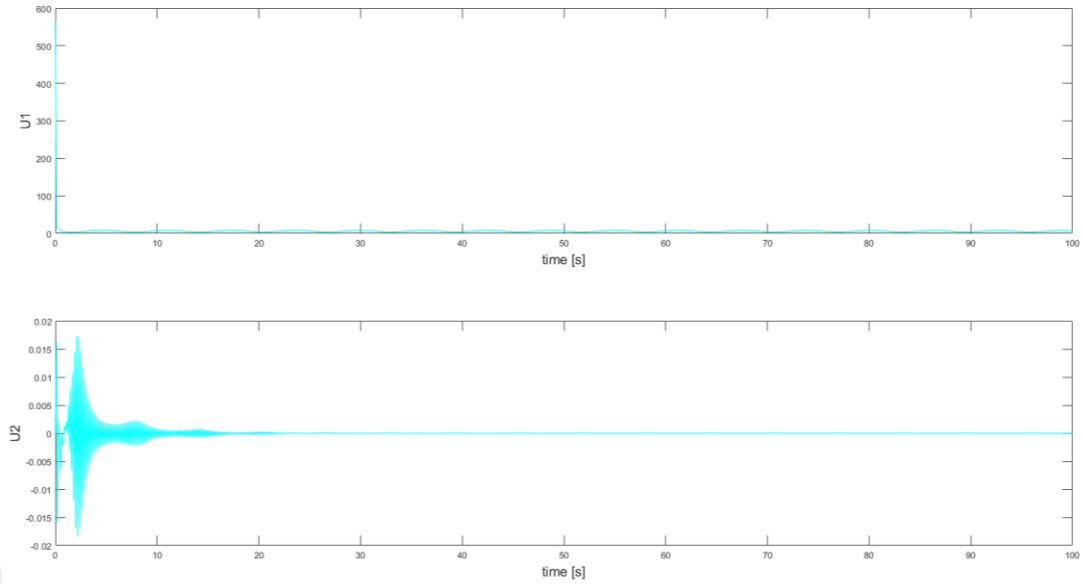

Figure D.6 Inputs of the Quadrotor U1, U2 (Complex Movements Trajectory)

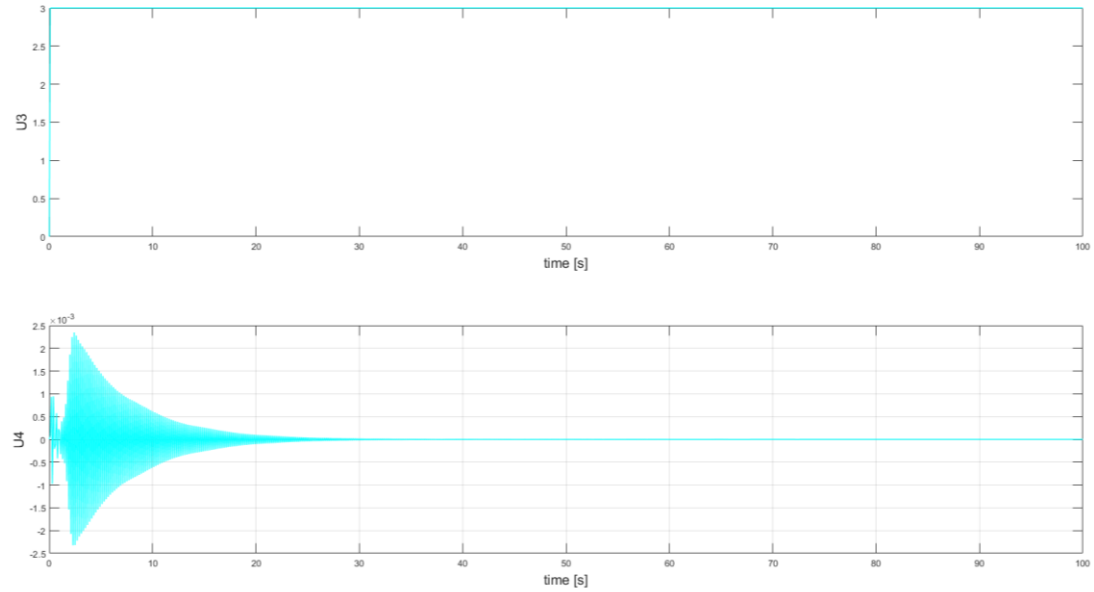

Figure D.7 Inputs of the Quadrotor U3, U4 (Complex Movements Trajectory)

CURRICULUM VITAE

Name Surname : Mücahit DEMİRCİOĞLU

EDUCATION :

- **B.Sc. :** 2019, Piri Reis University, Engineering Faculty, Mechanical Engineering
- **M.Sc. :** 2022, Istanbul Technical University, Graduate School, Aeronautical and Astronautical Engineering

PROFESSIONAL EXPERIENCE AND REWARDS:

- 2022 – Present, Research Assistant at Beykent University, Faculty of Engineering and Architecture, Mechanical Engineering Department

PUBLICATIONS, PRESENTATIONS AND PATENTS:

- **Demircioğlu M., Jafarov E.** (2023). Trajectory Tracking Control of UAV Quadrotor with Model Predictive Control via Linear Parametric Variable Approach. *Uluslararası Genç Araştırmacı Öğrenci Kongresi* (Özet Bildiri/Sözlü Sunum)(Yayın No:8383708)