Master's Thesis
석사 학위논문

Trajectory Tracking of Quadrotors Using Differential Flatness and Computed Torque Control

Kihoon Choi(최 기 훈 崔 基 訓)

Department of Information and Communication Engineering
정보통신융합공학전공

DGIST

2016

Master's Thesis
석사 학위논문

Trajectory Tracking of Quadrotors Using Differential Flatness and Computed Torque Control

Kihoon Choi(최 기 훈 崔 基 訓)

Department of Information and Communication Engineering
정보통신융합공학전공

DGIST

2016

Trajectory Tracking of Quadrotors Using Differential Flatness and Computed Torque Control

Advisor : Professor Yongsoon Eun

Co-advisor : Professor Dong Eui Chang

by

Kihoon Choi
Department of Information and Communication Engineering
DGIST

A thesis submitted to the faculty of DGIST in partial fulfillment of the requirements for the degree of Master of Science in the Department of Information and Communication Engineering. The study was conducted in accordance with Code of Research Ethics[1]

05(month). 23(day). 2016(year)

Approved by

Professor 은 용 순 (Signature)
(Advisor)

Professor 장 동 의 (Signature)
(Co-Advisor)

[1] Declaration of Ethical Conduct in Research: I, as a graduate student of DGIST, hereby declare that I have not committed any acts that may damage the credibility of my research. These include, but are not limited to: falsification, thesis written by someone else, distortion of research findings or plagiarism. I affirm that my thesis contains honest conclusions based on my own careful research under the guidance of my thesis advisor.

Trajectory Tracking of Quadrotors Using Differential Flatness and Computed Torque Control

Kihoon Choi

Accepted in partial fulfillment of the requirements
for the degree of Master of Science.

05(month). 23(day). 2016(year)

Head of Committee_____(인)
Prof. Yongsoon Eun
Committee Member_____(인)
Prof. Dong Eui Chang
Committee Member_____(인)
Prof. Kyung-Joon Park

MS/IC
201422019

최 기 훈. Kihoon Choi. Trajectory Tracking of Quadrotors Using Differential Flatness and Computed Torque Control. Department of Information and Communication Engineering. 2016. 56p. Prof. Eun, Yongsoon. Co-Advisors Prof. Chang, Dong Eui.

ABSTRACT

Unmanned Aerial Vehicles (UAV) has recently been receiving much attention because of a wide range of potentional applications such as environmental monitoring, disaster monitoring, reconnaissance and even deliveries for online shopping. For these applications, position and attitude control is an important task. However, the challenge of position and attitude control lies in that position of quadrotor is coupled with roll, pitch and yaw motions in non-linear manner. Motion planning is also important. Because reference trajectories inconsistent with feasible motion of the quadrotor make controller design difficult and result in poor tracking performance. The objective of this thesis is to design controller for quadrotor position and attitude motion tracking control. First, the quadrotor dynamics are modeled using reference frames, rotation matrix, force, moments, kinematics and dynamics by Euler-Newton Equation. Then, differential flatness-based motion planning is presented for reference trajectory generation. Finally, PID type controller and Computed Torque Method controller are designed for position and attitude control. Results are validated using MATLAB simulations.

Keywords : Flatness, Motion Planning, Trajectory Tracking Control, Quadrotor, UAV

Contents

List of Figures

List of Tables

List of Symbols

F^i The inertial frame $\{e_1, e_2, e_3\}$

F^v The vehicle frame $\{e_1, e_2, e_3\}$

F^{v_1} The vehicle-1 frame $\{e_1', e_2', e_3'\}$

F^{v_2} The vehicle-2 frame $\{E_1', E_2', E_3'\}$.

F^b The body frame $\{E_1,\ E_2,\ E_3\}$

e_1 X-axis in the inertial frame $[1 \quad 0 \quad 0]^T$

e_2 Y-axis in the inertial frame $[0 \quad 1 \quad 0]^T$

e_3 Z-axis in the inertial frame $[0 \quad 0 \quad 1]^T$

E_1 X-axis in the body frame $[1 \quad 0 \quad 0]^T$

E_2 Y-axis in the body frame $[0 \quad 1 \quad 0]^T$

E_3 Z-axis in the body frame $[0 \quad 0 \quad 1]^T$

x Position vector in the inertial frame

v Linear velocity vector in the inertial frame

$\dot{v}$ Linear acceleration vector in the inertial frame

R Rotation matrix

$SO(3)$ Orthonomal matrix

$\hat{\cdot}$ The Hat map

$so(3)$ 3×3 Skew symmetric matrices.

η Euler angle vector

ϕ Roll angle

θ Pitch angle

ψ Yaw angle

Ω Angular velocity in the body frame

$\dot{\Omega}$ Angular acceleration in the body frame

$\dot{\eta}$ Euler angle derivative component vector

C Transformation matrix from the angular velocity in the body frame to the Euler angle rates

J Quadrotor;s diagonal inertia matrix

J_x Area moment of inertia about E_1

J_y Area moment of inertia about E_2

J_z Area moment of inertia about E_3

τ Torque on the quadrotor expressed in the body frame

F_* Each motor force(front, left, back, right)

τ_* Each motor moment(front, left, back, right)

m Quadrotor's mass

g Gravitational acceleration

f Total force in the quadrotor

k_F Motor force constant

k_M Motor moment constant

ω_*^2 Angular speed squared of each motors

ω_f^2 Angular speed squared of the front motor

ω_l^2 Angular speed squared of the left motor

ω_b^2 Angular speed squared of the back motor

ω_r^2 Angular speed squared of the right motor

l Length from rotors to the center of the quadrotor

$y(t)$ The flat output

$X(t)$ The state vector

$U(t)$ The control input vector

k_D Derivative gain of attitude controller

k_P Proportional gain of attitude controller

k_{II} Integral gain of position controller

k_{DD} Derivative gain of position controller

k_{PP} Proportional gain of position controller

η_r Euler angle reference vector

x_r Position reference vector

Θ The joint variables vector

Θ_d Desired joint variables

$M(\Theta)$ The inertia term matrix of the robot manipulator

$V(\Theta, \dot{\Theta})$ The vector of centrifugal and Coriolis terms

$G(\Theta)$ the vector of gravity terms

E The error about desired joint variables and real joint variables

K_v The Computed Torque Method controller gain about $\dot{E}$

K_P The Computed Torque Method controller gain about E

α The inertia term matrix

β The centrifugal and Coriolis term matrix

Chapter 1

Introduction

Unmanned Aerial Vehicles (UAV) has recently been receiving much attention because of a wide range of potentional applications such as environmental monitoring, disaster monitoring, reconnaissance and even deliveries for only shopping. UAV also is used for patrols and military operations in Korea. It is described by Figure 1.1.

Figure 1.1: Patrol drone(left), military operation(right) in Korea

For these applications, controlling the position and attitude is an important task. The challenge of position and attitude control lies in that position of quadrotor is coupled with roll, pitch and yaw motions in non-linear manner. Motion planning is also needed. Because reference trajectories inconsistent with feasible motion of the quadrotor make controller design difficult and result in poor tracking performance. Quadrotors usually have a small capacity of the battery so flight time of quadrotors is usually 20 ~ 30 minutes. Therefore, there is a need for more appropriate motion trajectory generation. Because reference trajectories inconsistent with feasible motion trajectories of the

quadrotor reduce the life of motors. So motion trajectory generation methods are more important. To solve this problem, many researchers propose a variety of quadrotor controller and motion planning. After then, we introduce quadrotor control and motion planning method briefly.

1.1 Previous Work

There have been many papers on quadrotor control systems and motion planning. When we search the previous work for our research, we focus on quadrotor control and UAV motion planning. First, we introduce a variety of quadrotor controls.

Bouabdallah et al. proposed the usage of PID and LQ control techniques [1] to be applied on the quadrotor which was able to stabilize the quadrotor attitude of its hovering. After then, Bouabdallah and Siegwart proposed the use of back-stepping and sliding-mode nonlinear control methods [2] to control the quadrotor which gave good performance in the presence of disturbances. ChangSu et al. proposed a passivity-based adaptive back-stepping control for quadrotor with teleoperation system [3]. And Taeyoung proposed 2 types of attitude tracking controller which are smooth control and hybrid control scheme for a rigid body [4] and first method can guarantee almost semi-global exponential stability and second method verify global exponential stability. Tse-Huai et al. also proposed attitude tracking control using angular velocity observer when angular velocity measurements cannot be available [5].

In case of the differential flatness-based motion planning, Murray firstly proposed this method for aircraft [6]. Since then, many papers used its method [7-8]. Mellinger et al. proposed Mixed-Integer Quadratic Program (MIQP) motion planning for UAV [9]. Turpin et al. proposed Concurrent Assignment and Planning (CAPT) for multiple UAVs [10]. Another motion planning methods are RRT. Richter et al. recently proposed collision free motion planning using RRT for UAV [11].

This thesis also proposes the quadrotor position and attitude controller and flatness-based motion planning as previous work.

1.2 Motivation

The Motivation of this thesis is to design controller for quadrotor position and attitude motion tracking controller for our research scenarios which are attack scenarios using quadrotor. For attack scenarios, quadrotor motion planning is essential and we have quadrotor platforms – Hummingbird and AR.Drone 2.0.

Figure 1.2: Asctec Hummingbird(left), Parrot AR.Drone(right)

Because quadrotor platforms are expensive and friable so simulation validations are important and necessary before experiment validations. This thesis focuses on the quadrotor simulator. For this simulator, first, the detailed quadrotor dynamics are modeled using reference frames, rotation matrix, force, moments, kinematics and dynamics by Euler-Newton Equation. Then, differential flatness-based motion planning is presented for reference trajectory generation. Finally, PID type controller and Computed Torque Method controller are designed for position and attitude control. Results are validated using MATLAB simulations.

1.3 Thesis Structure

This thesis structure of this thesis is as follows. Chapter 2 presents the quadrotor mathematical modeling based on the Newton-Euler method including the reference frames,

rotation matrix, force, moments, kinematics and dynamics. Chapter 3 shows the flatness definition and the reference trajectory generation using differential flatness-based motion planning. Chapter 4 presents the structure of total quadrotor control. Chapter 5 shows the simulation results using Matlab Simulink. At last, Chapter 6 shows conclusion and future work for this thesis.

Chapter 2

Background

In this paper, we design quadrotor position and attitude tracking controller. Especially, attitude controller is designed using Computed Torque Method (CTM). To help readers understand attitude controller, this chapter is prepared.

2.1 Computed Torque Method

Computed Torque Method [12] is the application of robot manipulator control. This method usually used the rotational motion control of robot manipulator. In this subsection, Computed Torque Method is simply introduced.

The rigid body dynamics of robot manipulator have the form.

$$\tau = M(\Theta)\ddot{\Theta} + V\left(\Theta, \dot{\Theta}\right) + G(\Theta) \tag{2.1}$$

where $\Theta \in R^n$ is the joint variables vector, $M(\Theta) \in R^{n\times n}$ is the inertia term matrix of the robot manipulator, $V\left(\Theta, \dot{\Theta}\right) \in R^n$ is the vector of centrifugal and coriolis terms and $G(\Theta) \in R^n$ is the vector of gravity terms.

$M(\Theta)$, $V\left(\Theta, \dot{\Theta}\right)$, $G(\Theta)$ consist of Θ or $\dot{\Theta}$ and is so complicated. To control robot manipulator, choose the control input.

$$\tau = \alpha\tau' + \beta \tag{2.2}$$

where $\tau \in R^n$ is the vector of joint torques, $\alpha = M(\Theta)$ and $\beta = V\left(\Theta, \dot{\Theta}\right) + G(\Theta)$

We introduce new input τ' and it is given by

$$\tau' = \ddot{\Theta}_d + K_v\dot{E} + K_P E \tag{2.3}$$

where desired joint variables Θ_d, joint variable error $E = \Theta_d - \Theta$, K_v is the controller gain about $\dot{E}$ and K_P is the controller gain about E.

Using equation (2.2) and (2.3), the closed loop dynamics is characterized by error equation.

$$\ddot{E} + K_v\dot{E} + K_P E = 0 \tag{2.4}$$

According to the linear system theory, convergence of the tracking error to zero is guaranteed [12]. Figure 2.1 is illustrated by the overall structure of robot manipulator control.

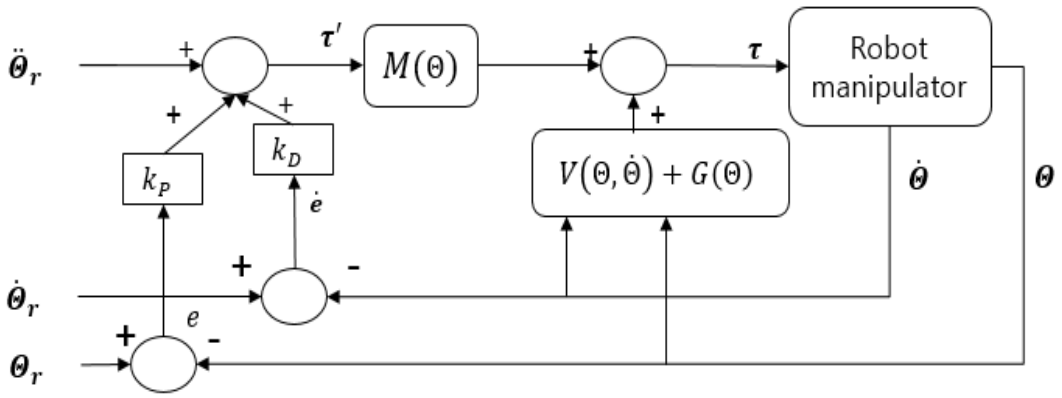

Figure 2.1: The structure of the robot manipulator control by CTM.

To help readers understand Computed Torque Method, mass spring damper system and 2-DOF robot manipulator examples are prepared.

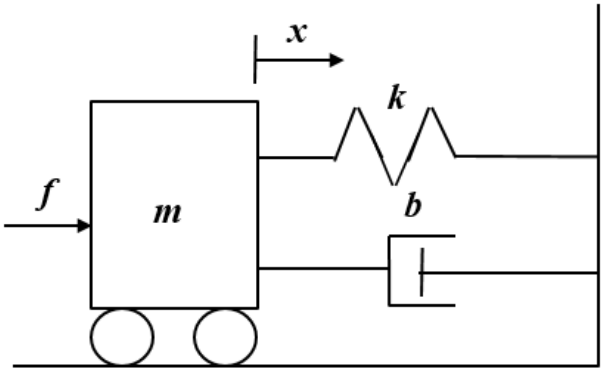

Figure 2.2: Mass spring damper system example.

First of all, we show an example of the linear system for understanding. Figure 2.2 shows mass spring damper system. Variable m is the mass. Variable b is the damping coefficient. Variable k is the spring constant. Variable x is position of the mass and Variable f is force which is control input of mass spring damper system. To describe mass spring damper system, we consider state space representation. It is given by

$$\begin{bmatrix}\dot{x}_1\\ \dot{x}_2\end{bmatrix} = \begin{bmatrix}0 & 1\\ -k/m & -b/m\end{bmatrix}\begin{bmatrix}x_1\\ x_2\end{bmatrix} + \begin{bmatrix}0\\ 1/m\end{bmatrix}u \tag{2.5}$$

$$y = [1 \quad 0]\begin{bmatrix}x_1\\ x_2\end{bmatrix}$$

where x_1 is position of the mass, x_2 is velocity of the mass.

To control mass spring damper system, we choose the control input f as follows:

$$f = \alpha f' + \beta \tag{2.6}$$

where $\alpha = m,\ \beta = b\dot{x} + kx.$

We can choose new control input f' as follows:

$$f' = \ddot{x}_r + k_D \dot{e}_x + k_P e_x \tag{2.7}$$

where e_x is position error and $\dot{e}_x$ is velocity error.

Figure 2.3 is the closed-loop control system of mass spring damper system using Computed Torque Method.

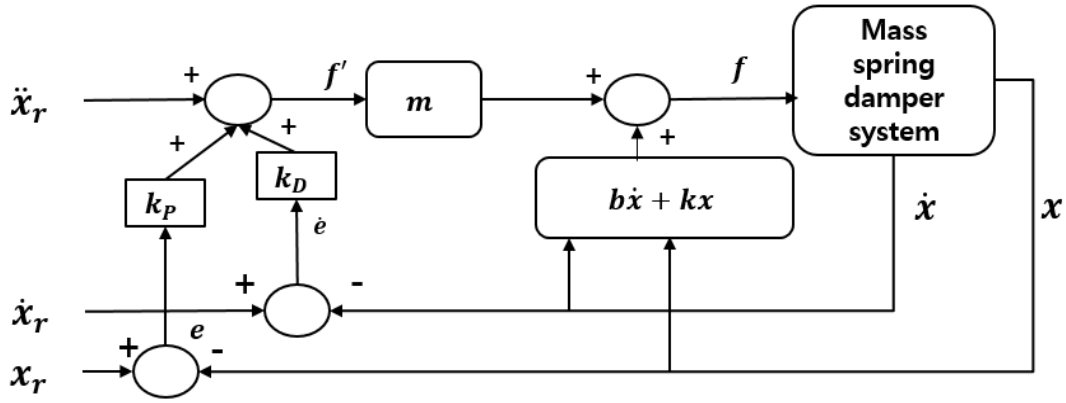

Figure 2.3: The structure of mass spring damper system control by CTM.

We validated mass spring damper system example using MATLAB simulation. Position reference is $x_r = 1$m. The result shows a stable dynamics. Result is illustrated in Figure 2.4.

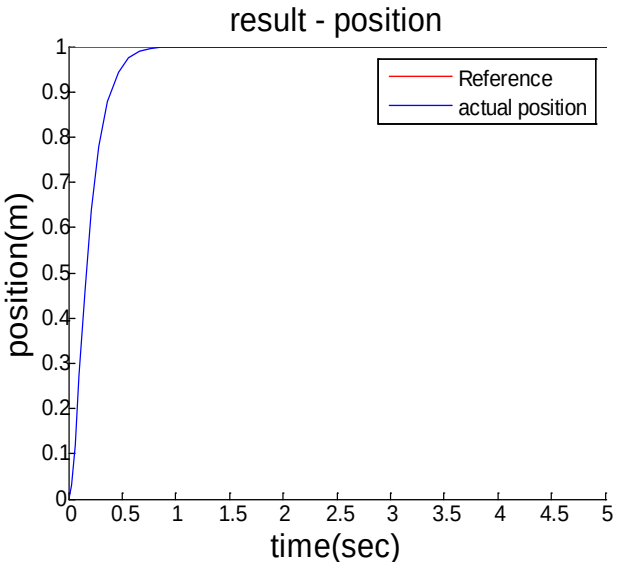

Figure 2.4: Result of mass spring damper system example.

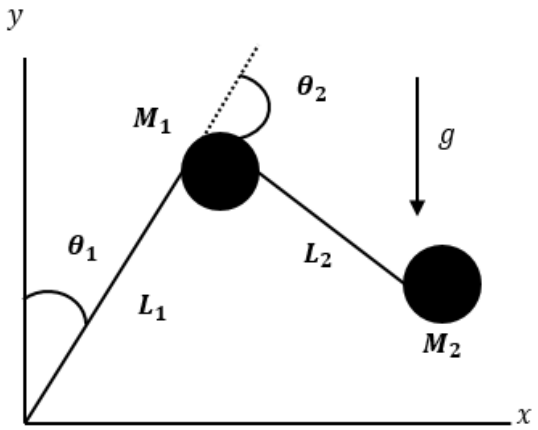

Figure 2.5: 2-DOF robot dynamics example.

Lastly, we show the nonlinear system employing Computed Torque Method. Figure 2.5 shows 2-DOF robot manipulator. Variable M_1 and M_2 are the point mass of each link. Variable L_1 and L_2 are the length of each link. Variable θ_1 and θ_2 are the angle of each link. Variable g is the gravitational acceleration. To describe 2-DOF robot dynamics, we consider Euler-Lagrange method. Positions of each links are given by

$$\begin{aligned} x_1 &= L_1 \sin\theta_1 \\ y_1 &= L_1 \cos\theta_1 \\ x_2 &= L_1 \sin\theta_1 + L_2 \sin(\theta_1 + \theta_2) \\ y_2 &= L_1 \cos\theta_1 + L_2 \cos(\theta_1 + \theta_2). \end{aligned} \tag{2.8}$$

So, kinetic energy of robot manipulator can be described as

$$\mathrm{K} = \tfrac{1}{2} M_1 \dot{x}_1^2 + \tfrac{1}{2} M_1 \dot{y}_1^2 + \tfrac{1}{2} M_2 \dot{x}_2^2 + \tfrac{1}{2} M_2 \dot{y}_2^2. \tag{2.9}$$

We simplified equation (2.6). It is given by

$$\begin{aligned} \mathrm{K} = {} & \tfrac{1}{2}(M_1 + M_2) L_1^2 \dot{\theta}_1^2 + \tfrac{1}{2} M_2 L_2^2 \left(\dot{\theta}_1^2 + \dot{\theta}_2^2\right) \\ & + M_2 L_2^2 \dot{\theta}_1 \dot{\theta}_2 + M_2 L_1 L_2 \left(\dot{\theta}_1 \dot{\theta}_2 + \dot{\theta}_1^2\right) \cos\theta_2. \end{aligned} \tag{2.10}$$

And potential energy of robot manipulator can be described as

$$\mathrm{P} = M_1 g L_1 \cos\theta_1 + M_2 g(L_1 \cos\theta_1 + L_2 \cos(\theta_1 + \theta_2)). \tag{2.11}$$

Lagrange equation is $\mathcal{L} = \mathrm{K} - \mathrm{P}$ and by Euler-Lagrange method, robot dynamics is given by

$$\tau = \frac{d}{dt}\left(\frac{\partial \mathcal{L}}{\partial \dot{\Theta}}\right) - \frac{\partial \mathcal{L}}{\partial \Theta}. \tag{2.12}$$

Equation (2.9) can change equation (2.1) forms. Therefore, final 2-DOF robot manipulator dynamics is given by

$$\begin{aligned}\begin{bmatrix}\tau_1 \\ \tau_2\end{bmatrix} = &\begin{bmatrix}(M_1+M_2)L_1^2 + M_2L_2^2 + 2M_2L_1L_2\cos\theta_2 & M_2L_2^2 + M_2L_1L_2\cos\theta_2 \\ M_2L_2^2 + M_2L_1L_2\cos\theta_2 & M_2L_2^2\end{bmatrix}\begin{bmatrix}\ddot{\theta}_1 \\ \ddot{\theta}_2\end{bmatrix} \\ &+ \begin{bmatrix}-M_2L_1L_2(2\dot{\theta}_1\dot{\theta}_2 + \dot{\theta}_2^2)\sin\theta_2 \\ M_2L_1L_2\dot{\theta}_1^2\sin\theta_2\end{bmatrix} \\ &+ \begin{bmatrix}-(M_1+M_2)gL_1\sin\theta_1 - M_2gL_2\sin(\theta_1+\theta_2) \\ -M_2gL_2\sin(\theta_1+\theta_2)\end{bmatrix}.\end{aligned} \tag{2.13}$$

To control robot manipulator, we choose the control input τ as follows:

$$\tau = \alpha\tau' + \beta \tag{2.14}$$

where $\alpha = \begin{bmatrix}(M_1+M_2)L_1^2 + M_2L_2^2 + 2M_2L_1L_2\cos\theta_2 & M_2L_2^2 + M_2L_1L_2\cos\theta_2 \\ M_2L_2^2 + M_2L_1L_2\cos\theta_2 & M_2L_2^2\end{bmatrix}$,

$\beta = \begin{bmatrix}-M_2L_1L_2(2\dot{\theta}_1\dot{\theta}_2 + \dot{\theta}_2^2)\sin\theta_2 \\ M_2L_1L_2\dot{\theta}_1^2\sin\theta_2\end{bmatrix} + \begin{bmatrix}-(M_1+M_2)gL_1\sin\theta_1 - M_2gL_2\sin(\theta_1+\theta_2) \\ -M_2gL_2\sin(\theta_1+\theta_2)\end{bmatrix}$.

And we also choose new control input τ' as follows:

$$\tau' = \ddot{\theta}_r + k_D \dot{e}_\theta + k_P e_\theta \tag{2.15}$$

where e_θ is angle error and $\dot{e}_\theta$ is angle velocity error.

Using equation (2.14) and (2.15), the closed loop dynamics is characterized by error equation.

$$\ddot{e}_\theta + k_D \dot{e}_\theta + k_P e_\theta = 0 \tag{2.16}$$

And then we validated robot manipulator control example using MATLAB simulation. Each angle references are $\theta_{r_1} = 0.8$rad and $\theta_{r_2} = 0.4$rad. The results show a stable dynamics. Results are illustrated in Figure 2.6.

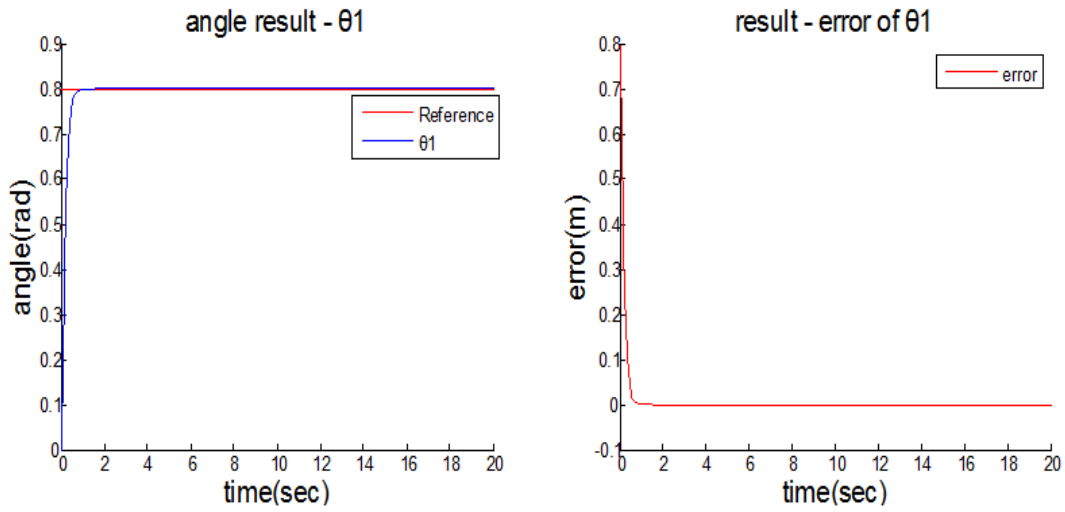

Figure 2.6: Results of 2-DOF robot dynamics example(θ_1)

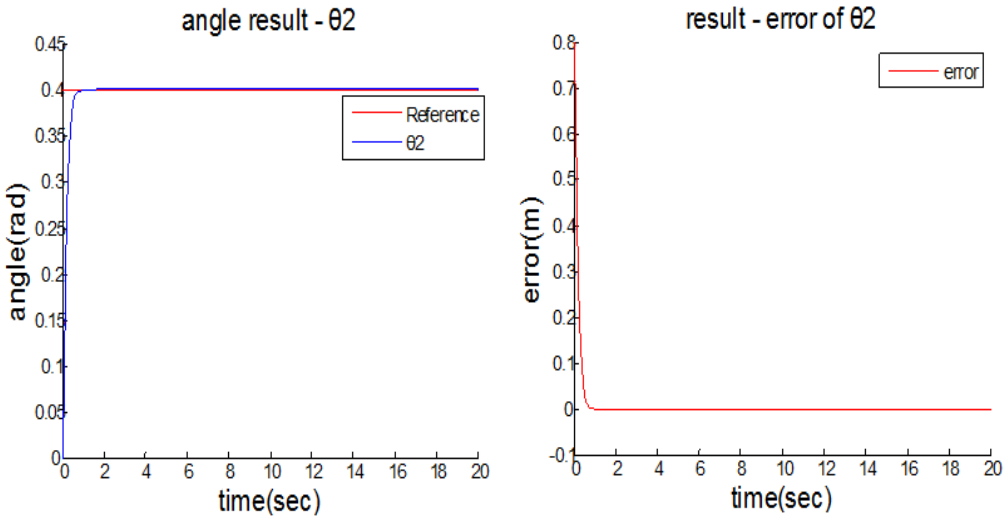

Figure 2.8: Results of 2-DOF robot dynamics example(θ_2)

Chapter 3

Quadrotor Model

The first step is to create an accurate and detailed mathematical model of the quadrotor in control design. In this chapter, we derive quadrotor dynamics using reference frame, rotation matrix, force and moments, kinematics and dynamics by Euler-Newton Equation.

3.1 Model Assumptions

In this subsection, assumptions made to obtain a simple but useful model are explained.

(1) The quadrotor is rigid body.
(2) The structure of quadrotor is symmetric.
(3) Aerodynamic drag forces are neglected.

Assumption 1 and 2 are simplified inertia matrix J which is diagonal form and constant matrix. Assumption 3 is useful because aerodynamics drag forces are very small so these forces are neglected in indoor environment.

3.2 Reference Frames

For the quadrotor, there are several coordinate systems [13]. This section is defined following coordinate frames – the inertial frame, the vehicle frame, the vehicle-1 frame, the vehicle-2 frame, and the body frame. Especially, the inertial frame and the body frame are important because those can represent the rotation of the quadrotor.

3.2.1 The Inertial Frame

The inertial frame is the orthonomal basis fixed in space $\{e_1, e_2, e_3\}$. It means that the inertial frame does not change when the robot moves and it is absolute frame in the space. Figure 3.1 is illustrated.

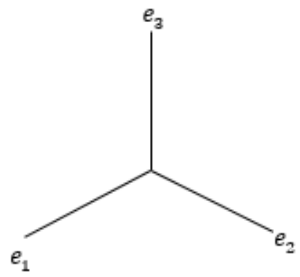

Figure 3.1: The inertial frame.

3.2.2 The Vehicle Frame

The origin of the vehicle frame F^v is at the center of mass of the quadrotor. However, the axes of F^v are aligned with the axis of the inertial frame F^i. The x-axis of the vehicle frame points e_1, the y-axis of the vehicle frame points e_2, the z-axis of the vehicle frame points e_3.

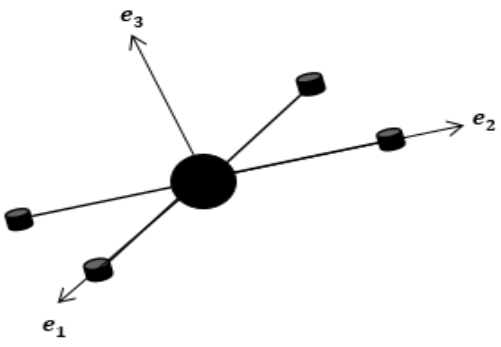

Figure 3.2: The vehicle frame of quadrotor.

3.2.3 The Vehicle-1 Frame

The vehicle-1 frame F^{v1} represents the rotation of the yaw angle (ψ) $\{e_1', e_2', e_3'\}$. The vehicle-1 frame is the yaw rotation coordinate frame which is positively rotated about e_3 by yaw angle ψ. The transformation from F^{v} to F^{v1} is defined as

$$R_z(\psi) = \begin{bmatrix} \cos\psi & \sin\psi & 0 \\ -\sin\psi & \cos\psi & 0 \\ 0 & 0 & 1 \end{bmatrix} \tag{3.1}$$

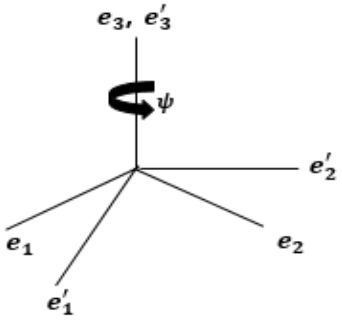

Figure 3.3: The vehicle-1 frame of quadrotor.

3.2.4 The Vehicle-2 Frame

The vehicle-2 frame F^{v2} represents the rotation of the pitch angle (θ) $\{E_1', E_2', E_3'\}$. The vehicle-2 frame is the pitch rotation coordinate frame which is positively rotated about e_1' by pitch angle θ. The transformation from F^{v1} to F^{v2} is defined as

$$R_y(\theta) = \begin{bmatrix} \cos\theta & 0 & -\sin\theta \\ 0 & 1 & 0 \\ \sin\theta & 0 & \cos\theta \end{bmatrix} \tag{3.2}$$

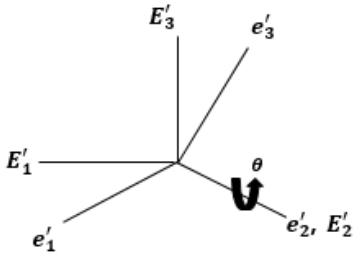

Figure 3.4: The vehicle-2 frame of quadrotor.

3.2.5 The Body Frame

The body frame F^b represents rotation of the roll angle (ϕ) $\{E_1,\ E_2,\ E_3\}$. The body frame is the roll rotation coordinate frame which is positively rotated about E_2' by roll angle ϕ. The transformation from F^{v2} to F^b is defined as

$$R_x(\phi) = \begin{bmatrix} 1 & 0 & 0 \\ 0 & \cos\phi & \sin\phi \\ 0 & -\sin\phi & \cos\phi \end{bmatrix} \tag{3.3}$$

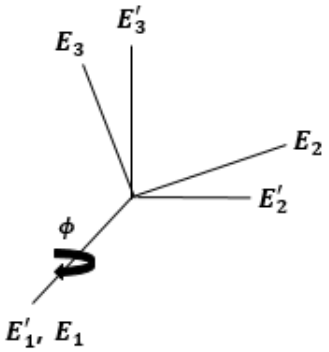

Figure 3.5: The body frame of quadrotor.

3.3 Rotation Matrix

The vector in the body frame does not apply the vector in the inertial frame. So the relationship between the body frame and the inertial frame is needed. It called rotation matrix. The rotation matrix $R \in SO(3)$ which is defined as $SO(3) \triangleq \{A \in R^{3\times3} | A^T A = I_3, det(A) = 1\}$. We define rotation matrix from the body frame to the inertial frame using ZYX Euler angles as

$$R = \begin{bmatrix} \cos\theta\cos\psi & \sin\phi\sin\theta\cos\psi - \cos\phi\sin\psi & \cos\phi\sin\theta\cos\psi + \sin\phi\sin\psi \\ \cos\theta\sin\psi & \sin\phi\sin\theta\sin\psi + \cos\phi\cos\psi & \cos\phi\sin\theta\sin\psi - \sin\phi\cos\psi \\ -\sin\theta & \sin\phi\cos\theta & \cos\phi\cos\theta \end{bmatrix} \tag{3.4}$$

3.4 Quadrotor Kinematics & Dynamics

Kinematics is a viewpoint which studies the motion of a body without consideration of the forces and torques acting on it. Kinematics usually don't use 1-DOF motion which is examples of dynamic models so it is important over 3-DOF motion description.

The dynamic systems can be gotten using two famous methods which are Newton-Euler method and Euler-Lagrange method. Both methods result in equivalent set of equations. For simple dynamics, Euler-Lagrange method is the useful choice because it is easy. However, the dynamics complexity increases, it is difficult to apply Euler-Lagrange method so it is reason that the Newton-Euler method has its advantages. We consider Newton-Euler method to get the quadrotor dynamics.

In this section, the quadrotor kinematics and dynamics which will be useful to get the equations of motion for the quadrotor are presented.

3.4.1 Kinematic Model

The translational motion of kinematics can be defined $x = [x_1 \quad x_2 \quad x_3]^T$ is position vector of the quadrotor in the inertial frame and $v = [v_1 \quad v_2 \quad v_3]^T$ is linear velocity vector of the quadrotor in the inertial frame. It is given by

$$\begin{aligned} \dot{x}_1 &= v_1 \\ \dot{x}_2 &= v_2 \\ \dot{x}_3 &= v_3 \end{aligned} \tag{3.5}$$

Next, the rotational motion equation of kinematics can be defined using the relationship between angular velocity in the body frame $\Omega = [p \quad q \quad r]^T$, Euler angle vector $\eta = [\phi \quad \theta \quad \psi]^T$ and Euler angle derivative component vector $\dot{\eta} = [\dot{\phi} \quad \dot{\theta} \quad \dot{\psi}]^T$ and it is given by

$$\begin{bmatrix} p \\ q \\ r \end{bmatrix} = \begin{bmatrix} \dot{\phi} \\ 0 \\ 0 \end{bmatrix} + R_x(\phi) \begin{bmatrix} 0 \\ \dot{\theta} \\ 0 \end{bmatrix} + R_x(\phi) R_y(\theta) \begin{bmatrix} 0 \\ 0 \\ \dot{\psi} \end{bmatrix} = \begin{bmatrix} 1 & 0 & -\sin\theta \\ 0 & \cos\phi & \sin\phi\cos\theta \\ 0 & -\sin\phi & \cos\phi\cos\theta \end{bmatrix} \begin{bmatrix} \dot{\phi} \\ \dot{\theta} \\ \dot{\psi} \end{bmatrix}. \quad (3.6)$$

The final equation form of the rotational motion is given by

$$\begin{aligned} \dot{\phi} &= p + q\sin\phi\tan\theta + r\cos\phi\tan\theta \\ \dot{\theta} &= q\cos\phi - r\sin\phi \\ \dot{\psi} &= q\frac{\sin\phi}{\cos\theta} + r\frac{\cos\phi}{\cos\theta} \end{aligned} \quad (3.7)$$

3.4.2 Dynamic Model

The dynamics of the quadrotor can be defined representing the rotational motion and the translational motion, and the translational motion equation of the quadrotor obtained from the second law of Newton. It is given by

$$\begin{aligned} \dot{v}_1 &= (\cos\phi\sin\theta\cos\psi + \sin\phi\sin\psi)\frac{f}{m} \\ \dot{v}_2 &= (\cos\phi\sin\theta\sin\psi - \sin\phi\cos\psi)\frac{f}{m} \\ \dot{v}_3 &= -g + (\cos\phi\cos\theta)\frac{f}{m} \end{aligned} \quad (3.8)$$

where $m \in R$ is quadrotor's mass, $g \in R$ is gravitational acceleration, $f \in R$ is total force in the quadrotor, $\dot{v}_1$, $\dot{v}_2$ and $\dot{v}_3$ are linear acceleration in the inertial frame.

The rotational motion equation of the quadrotor obtained from the second law of Newton. It is given by

$$\dot{p} = \frac{J_y - J_z}{J_x} qr + \frac{\tau_1}{J_x}$$

$$\dot{q} = \frac{J_z - J_x}{J_y} pr + \frac{\tau_2}{J_y} \tag{3.9}$$
$$\dot{r} = \frac{J_x - J_y}{J_z} pq + \frac{\tau_3}{J_z}$$

where J_x, J_y and J_z are diagonal components of the inertia matrix J, τ_1, τ_2 and τ_3 are torques on the quadrotor expressed in the body frame.

3.4.3 Force and Moments

In this subsection, we describe the relationship between total forces and torques with the angular speed squared of each motors ω_*^2.

The force and torque of each motors can be expressed as

$$F_* = k_F \omega_*^2 \tag{3.10}$$
$$\tau_* = k_M \omega_*^2$$

where k_F, k_M are motor force constant and motor moment constant, ω_*^2 are angular speed squared of each motors.

The forces and torques on the quadrotor can be written in matrix form as

$$\begin{bmatrix} f \\ \tau_1 \\ \tau_2 \\ \tau_3 \end{bmatrix} = \begin{bmatrix} k_F & k_F & k_F & k_F \\ 0 & lk_F & 0 & -lk_F \\ lk_F & 0 & -lk_F & 0 \\ -k_M & k_M & -k_M & k_M \end{bmatrix} \begin{bmatrix} \omega_f^2 \\ \omega_l^2 \\ \omega_b^2 \\ \omega_r^2 \end{bmatrix} \tag{3.11}$$

where l is length from rotors to the center of the quadrotor.

3.4.4 State Space Representation

The state space representation model of the quadrotor is essential to verify that quadrotor dynamics is flat system. So in this subsection, first, we define quadrotor's state vector $X(t)$ which defines the position and linear velocity in the inertial frame, Euler angle and angular velocity in the body frame. It is given by

$$X(t) = [x_1 \quad x_2 \quad x_3 \quad v_1 \quad v_2 \quad v_3 \quad \phi \quad \theta \quad \psi \quad p \quad q \quad r]^T \tag{3.11}$$

And control input vector $U(t)$ is defined as

$$U(t) = [u_1 \quad u_2 \quad u_3 \quad u_4]^T = [f \quad \tau_1 \quad \tau_2 \quad \tau_3]^T \tag{3.12}$$

The complete mathematical model of the quadrotor can be written in a state space representation using equation (3.5), (3.7), (3.8), (3.9), (3.11) and (3.12). It is given by

$$\begin{bmatrix} \dot{x}_1 \\ \dot{x}_2 \\ \dot{x}_3 \\ \dot{v}_1 \\ \dot{v}_2 \\ \dot{v}_3 \\ \dot{\phi} \\ \dot{\theta} \\ \dot{\psi} \\ \dot{p} \\ \dot{q} \\ \dot{r} \end{bmatrix} = \begin{bmatrix} v_1 \\ v_2 \\ v_3 \\ (\cos\phi\sin\theta\cos\psi + \sin\phi\sin\psi)\frac{u_1}{m} \\ (\cos\phi\sin\theta\sin\psi - \sin\phi\cos\psi)\frac{u_1}{m} \\ -g + (\cos\phi\cos\theta)\frac{u_1}{m} \\ p + q\sin\phi\tan\theta + r\cos\phi\tan\theta \\ q\cos\phi - r\sin\phi \\ q\frac{\sin\phi}{\cos\theta} + r\frac{\cos\phi}{\cos\theta} \\ \frac{J_y - J_z}{J_x}qr + \frac{u_2}{J_x} \\ \frac{J_z - J_x}{J_y}pr + \frac{u_2}{J_y} \\ \frac{J_x - J_y}{J_z}pq + \frac{u_4}{J_z} \end{bmatrix} \tag{3.13}$$

Chapter 4

Differential Flatness-Based Motion Planning

In Chapter 1, we explain the importance of quadrotor motion planning. And one of motion planning methods is differential flatness-based motion planning which makes smooth trajectories. In this chapter, first, we verify that quadrotor dynamics are flat system. And then, we focus on how to generate motion trajectories.

4.1 Differential Flatness

In this section, we show that the quadrotor dynamics with the four inputs is differentially flat. If system is a flat, we can consider the smooth motion trajectory generation. First, we define flatness [14].

Definition 1. *A dynamic system* $\dot{X} = f(X, U)$, $X \in R^n$, $U \in R^m$, *is flat if and only if there exist variables* $\exists\, y(t) \in R^m$

$$X(t) = \varphi_0\left(y(t), \dot{y}(t), \cdots, y^k(t)\right)$$
$$U(t) = \varphi_1\left(y(t), \dot{y}(t), \cdots, y^{(l)}(t)\right) \tag{4.1}$$

$$\frac{d}{dt}\varphi_0\left(y(t), \dot{y}(t), \cdots, y^{(k)}(t)\right) = f(\varphi_0\left(y(t), \dot{y}(t), \cdots, y^{(k)}(t)\right),\ \varphi_1\left(y(t), \dot{y}(t), \cdots, y^{(l)}(t)\right))$$

In this thesis, our choice of the flat outputs for the quadrotor are given by

$$y_1 = x_1$$
$$y_2 = x_2 \tag{4.2}$$
$$y_3 = x_3$$

$$y_4 = \psi$$

After then, we prove the quadrotor is flat system using our choice of the flat outputs. To prove a flat system, the parameterization of other state using flat outputs must be needed. By substituting the flat outputs to the state $X(t)$, they are given by

$$\begin{aligned} v_1 &= \dot{y}_1 \\ v_2 &= \dot{y}_2 \\ v_3 &= \dot{y}_3 \\ \phi &= \sin^{-1}\frac{\ddot{y}_1 \sin y_4 - \ddot{y}_2 \cos y_4}{\sqrt{\ddot{y}_1^2 + \ddot{y}_2^2 + (\ddot{y}_3 + 9.8)^2}} \\ \theta &= \tan^{-1}\frac{\ddot{y}_1 \cos y_4 + \ddot{y}_2 \sin y_4}{\ddot{y}_3 + 9.8} \\ p &= \dot{\phi} - \dot{\psi}\sin\theta \\ q &= \dot{\theta}\cos\phi + \dot{\psi}\sin\phi\cos\theta \\ r &= -\dot{\theta}\sin\phi + \dot{\psi}\cos\phi\cos\theta \end{aligned} \tag{4.3}$$

The parameterization of $\dot{\phi}, \dot{\theta}$ in function of the flat outputs are needed to verify that $p,\ q,\ r$ become parameterization of the flat outputs.

$$\dot{\phi} = \frac{\begin{array}{c}(\dddot{y}_1 \sin y_4 + \ddot{y}_1 \dot{y}_4 \cos y_4 - \dddot{y}_2 \cos y_4 + \ddot{y}_2 \dot{y}_4 \cos y_4)\left(\sqrt{\ddot{y}_1^2 + \ddot{y}_2^2 + (\ddot{y}_3 + 9.8)^2}\right) \\ -(\ddot{y}_1 \sin y_4 - \ddot{y}_2 \cos y_4)(0.5\frac{1}{\sqrt{\ddot{y}_1^2 + \ddot{y}_2^2 + (\ddot{y}_3 + 9.8)^2}})(2\ddot{y}_1\dot{y}_1 + 2\ddot{y}_2\dot{y}_2 + 2\ddot{y}_3\dot{y}_3 + 19.6\ddot{y}_1)\end{array}}{(\ddot{y}_1^2 + \ddot{y}_2^2 + (\ddot{y}_3 + 9.8)^2) * \cos\phi} \tag{4.4}$$

$$\dot{\theta} = \frac{(\dddot{y}_1 \cos y_4 - \ddot{y}_1 \dot{y}_4 \sin y_4 + \dddot{y}_2 \sin y_4 + \ddot{y}_2 \dot{y}_4 \cos y_4)(\ddot{y}_3 + 9.8) - (\ddot{y}_1 \cos y_4 + \ddot{y}_2 \sin y_4)\dddot{y}_3}{(\ddot{y}_3 + 9.8)^2 \cos^2\theta}$$

Through the above equation (4.4), (4.5) and (4.6), the state $X(t)$ can change our choice of the flat outputs $x_1,\ x_2,\ x_3$ and ψ.

Then, we verify that the quadrotor's control inputs $U(t)$ change the flat outputs. the parameterization of control inputs $U(t)$ are given by

$$
\begin{aligned}
u_1 &= \frac{m(g+\ddot{y}_3)}{\cos\phi\cos\theta} \\
u_2 &= J_x\dot{p} + qr(-J_y + J_z) \\
u_3 &= J_y\dot{q} + pr(J_x - J_z) \\
u_4 &= J_z\dot{r} + pq(-J_x + J_y)
\end{aligned} \tag{4.5}
$$

where $\dot{p}$, $\dot{q}$ and $\dot{r}$ are angular acceleration with respect to the body frame.

To convert $\dot{p}$, $\dot{q}$, $\dot{r}$ to the flat outputs of our choice, we need to verify that the parameterization of $\ddot{\phi}, \ddot{\theta}$ in function of the flat outputs. They are given by

$$
\ddot{\phi} = \frac{\begin{pmatrix} (\dddot{y}_1\sin y_4 + \ddot{y}_1\dot{y}_4\cos y_4 - \dddot{y}_2\cos y_4 + \ddot{y}_2\dot{y}_4\cos y_4)\left(\sqrt{\ddot{y}_1^2+\ddot{y}_2^2+(\ddot{y}_3+9.8)^2}\right) \\ -(\ddot{y}_1\sin y_4 - \ddot{y}_2\cos y_4)\left(0.5\frac{1}{\sqrt{\ddot{y}_1^2+\ddot{y}_2^2+(\ddot{y}_3+9.8)^2}}\right)(2\ddot{y}_1\dddot{y}_1+2\ddot{y}_2\dddot{y}_2+2\ddot{y}_3\dddot{y}_3+19.6\dddot{y}_1) \end{pmatrix}' (\ddot{y}_1^2+\ddot{y}_2^2+(\ddot{y}_3+9.8)^2)}{(\ddot{y}_1^2+\ddot{y}_2^2+(\ddot{y}_3+9.8)^2)^2 * \cos\phi} + \dot{\phi}^2\tan\phi
$$

$$
-((\dddot{y}_1\sin y_4 + \ddot{y}_1\dot{y}_4\cos y_4 - \dddot{y}_2\cos y_4 + \ddot{y}_2\dot{y}_4\cos y_4)\left(\sqrt{\ddot{y}_1^2+\ddot{y}_2^2+(\ddot{y}_3+9.8)^2}\right)
$$

$$
-(\ddot{y}_1\sin y_4 - \ddot{y}_2\cos y_4)(0.5\frac{1}{\sqrt{\ddot{y}_1^2+\ddot{y}_2^2+(\ddot{y}_3+9.8)^2}})(2\ddot{y}_1\dddot{y}_1+2\ddot{y}_2\dddot{y}_2+2\ddot{y}_3\dddot{y}_3+19.6\dddot{y}_1))(\ddot{y}_1^2+\ddot{y}_2^2+(\ddot{y}_3+9.8)^2)' \tag{4.6}
$$

$$
\ddot{\theta} = \frac{((\dddot{y}_3(\dddot{y}_1\cos y_4 - \ddot{y}_1\dot{y}_4\sin y_4 + \dddot{y}_2\sin y_4 + \ddot{y}_2\dot{y}_4\cos y_4) + (\ddot{y}_3+9.8)(\dddot{y}_1\cos y_4 - \ddot{y}_1\dot{y}_4\sin y_4 + \dddot{y}_2\sin y_4 + \ddot{y}_2\dot{y}_4\cos y_4)')}{-((\ddot{y}_1\cos y_4 + \ddot{y}_2\sin y_4)'\ddot{y}_3 + y_3^{(4)}(\ddot{y}_1\cos y_4 + \ddot{y}_2\sin y_4)))}{(\ddot{y}_3+9.8)^4\cos^2\theta}
$$

$$
+\frac{-2\ddot{y}_3(\ddot{y}_3+9.8)((\dddot{y}_1\cos y_4 - \ddot{y}_1\dot{y}_4\sin y_4 + \dddot{y}_2\sin y_4 + \ddot{y}_2\dot{y}_4\cos y_4)(\ddot{y}_3+9.8) - (\ddot{y}_1\cos y_4 + \ddot{y}_2\sin y_4)\dddot{y}_3)}{(\ddot{y}_3+9.8)^4\cos^2\theta}
$$

$$
+2\dot{\theta}^2\tan\theta
$$

As a result, the quadrotor dynamics can be written in the function of flat outputs x_1, x_2, x_3 and ψ.

4.2 Flat Output Trajectory Generation

Several methods can be used to design the smooth flat output trajectory generation in the flat system. In this paper, the Bezier polynomial function [14] is considered. This

method is advantaged because of the main reason which is the coefficients of the polynomial can be easily calculated in function of the initial and the final conditions. A general Bezier polynomial function is given by

$$y = a_n t^n + a_{n-1} t^{n-1} + \cdots + a_2 t^2 + a_1 t + a_0 \quad (4.7)$$

where t is time and $a_i (i = 0, \cdots, n)$ are constant coefficients to be calculated in function of the initial and final conditions.

The degree of Bezier polynomial function for flat output trajectory generation is 9th order polynomial function because 10 conditions are used. Those are 5 initial flat output conditions and 5 final flat output conditions to calculate the trajectory planning. Flat output trajectories are given by

$$y_i = a_9 t^9 + a_8 t^8 + a_7 t^7 + \cdots + a_3 t^3 + a_2 t^2 + a_1 t^1 + a_0 \text{ (i = 1,2,3,4)} \quad (4.8)$$

If you want to generate trajectory $y_1(0) = 0,\ \dot{y}_1(0) = 0,\ \ddot{y}_1(0) = 0,\ \dddot{y}_1(0) = 0,\ y_1^{(4)}(0) = 0$ and $y_1(4) = 1,\ \dot{y}_1(4) = 0,\ \ddot{y}_1(4) = 0,\ \dddot{y}_1(4) = 0,\ y_1^{(4)}(4) = 0$. It is given by

$$y_1(t) = 70(t/4)^9 - 315(t/4)^8 + 540(t/4)^7 - 420(t/4)^6 + 126(t/4)^5 \quad (4.9)$$

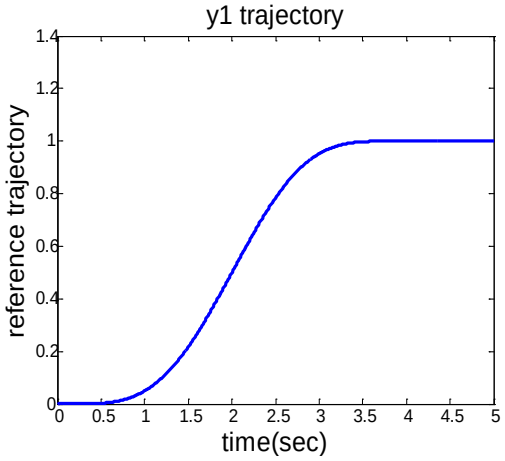

Figure 4.1: An example of flat output trajectory.

To prove feasible motion of quadrotor, we show the motion planning of quadrotor state $X(t)$ and control input $U(t)$ using the example - Equation (4.9). Flat outputs are same as Equation (4.9). It is illustrated at Figure 4.2.

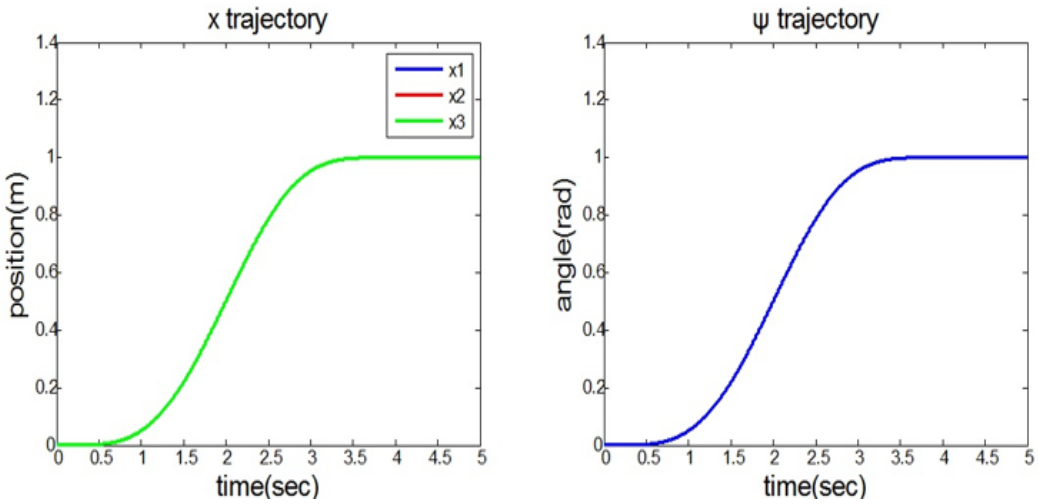

Figure 4.2: Flat output trajectories.

At this moment, other states also are flat. They are illustrated at Figure 4.3 and Figure 4.4.

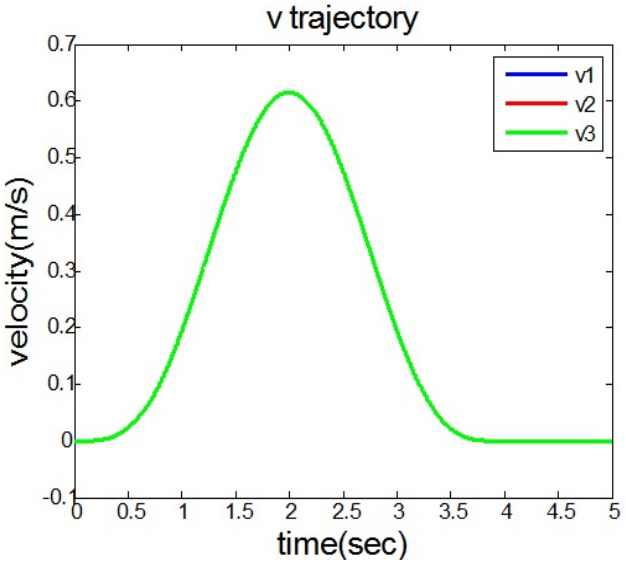

Figure 4.3: V trajectories (state).

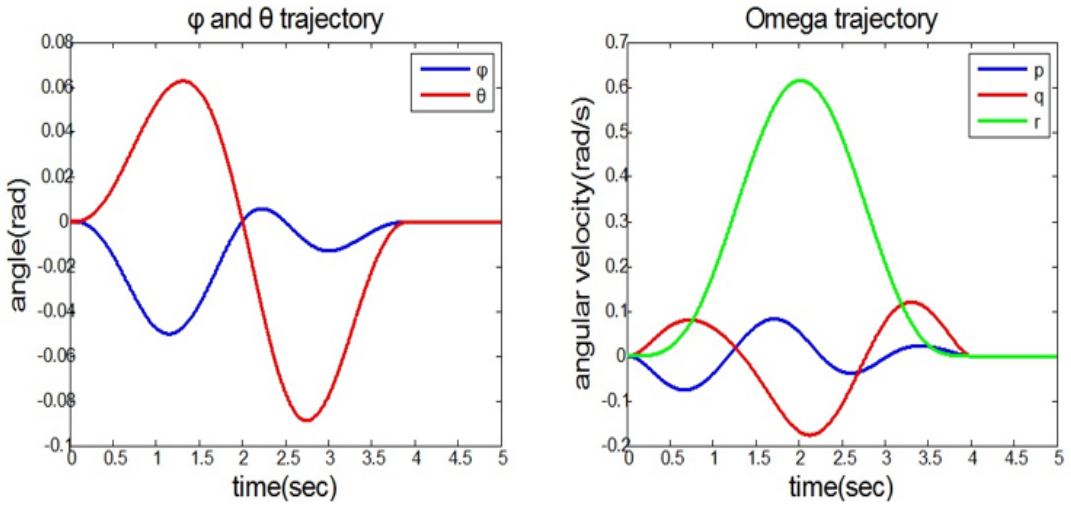

Figure 4.4: ϕ, θ and Ω trajectories (state).

Control inputs are illustrated at Figure 4.5.

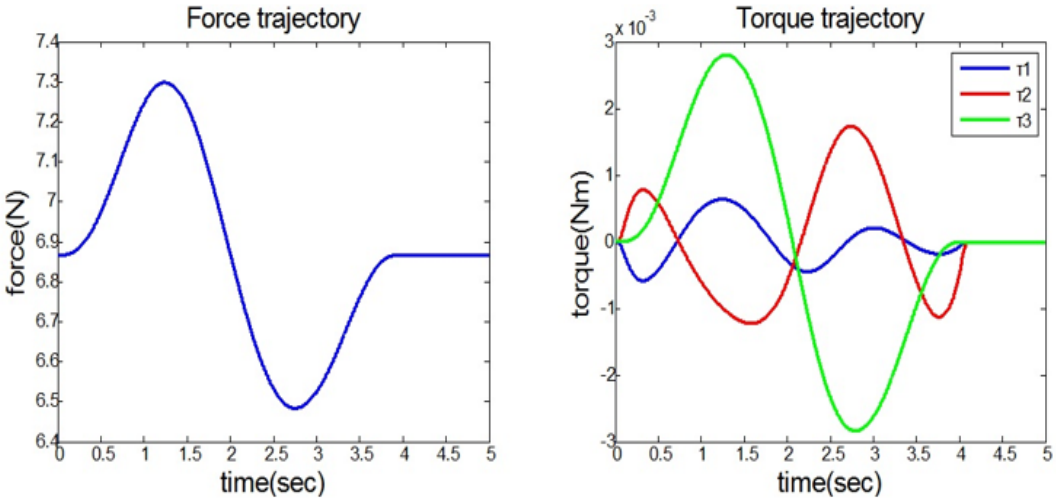

Figure 4.5: Force and torque trajectories (control input).

Chapter 5

Controller Design

In Chapter 3 and 4, we introduce dynamic model and flatness-based motion planning. In this chapter, we focus on control design formulation. In Section 5.1, we describe PID type position controller. In Section 5.2, we present flat output conversion for reference Euler angles. Finally, we describe force generator and attitude controller using Computed Torque Method in Section 5.3 and 5.4 respectively.

For the quadrotor control, the overall structure of the quadrotor control [8] is illustrated in Figure 5.1.

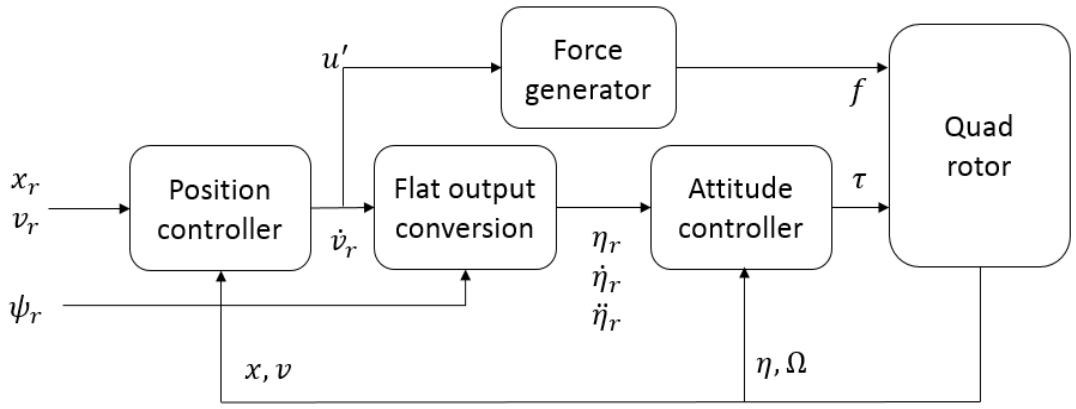

Figure 5.1: The overall structure of the quadrotor control.

5.1 Position Controller

The PID type controller applied to a variety of applications. The PID type controller has the advantages which are that parameter gains can adjust easily and it is very simple to design. We design the position controller using PID type controller.

We make the new control input u' which replaces the reference accelerations vector $\dot{v}_r$ because only the reference accelerations vector $\dot{v}_r$ doesn't overcome system

error. So we consider PID feedback of the position and linear velocity error. It is given by

$$u' = k_{PP}e_x + k_{DD}\dot{e}_x + k_{II}\int e_x \tag{5.4}$$

where e_x is position error, $\dot{e}_x$ is velocity error.

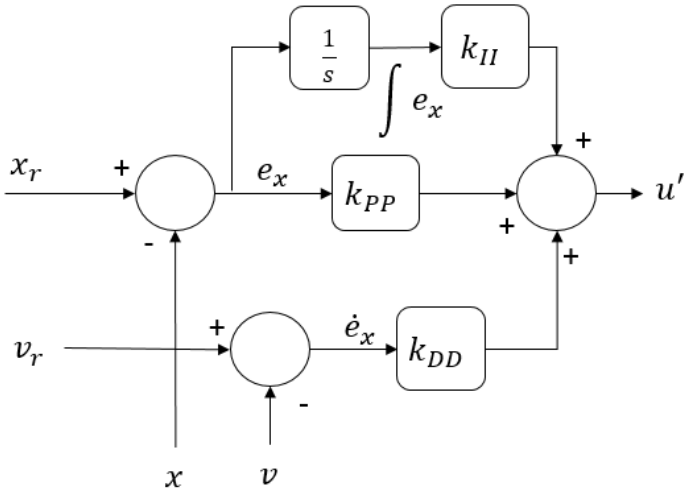

Figure 5.2: The structure of the position controller for the quadrotor.

5.2 Flat Output Conversion

In this section, we introduce the flat output conversion. The reference accelerations vector $\dot{v}_r$ and the reference yaw angle vector ψ_r are used to make reference Euler angles and its derivative components $\eta_r,\ \dot{\eta}_r,\ \ddot{\eta}_r$. This process can lead to the translational dynamics of the quadrotor. They are as follows

$$\begin{aligned}
\dot{v}_{r_1} &= (\cos\phi\sin\theta\cos\psi_r + \sin\phi\sin\psi_r)\frac{f}{m} \\
\dot{v}_{r_2} &= (\cos\phi\sin\theta\sin\psi_r - \sin\phi\cos\psi_r)\frac{f}{m} \\
\dot{v}_{r_3} &= -g + \cos\phi\cos\theta\frac{f}{m}
\end{aligned} \tag{5.5}$$

The reference roll angle vector ϕ_r and the reference pitch angle vector θ_r can be redefined using the translational dynamics of the quadrotor which substitute $\dot{v}_r$ reference accelerations vector and the yaw angle reference vector ψ_r. They are given by equation (5.6)

$$\phi_r = \sin^{-1}\left(\frac{\dot{v}_{r\_1}\sin\psi_r - \dot{v}_{r\_2}\cos\psi_r}{\sqrt{\dot{v}_{r\_1}{}^2 + \dot{v}_{r\_2}{}^2 + (\dot{v}_{r\_3}+9.8)^2}}\right) \tag{5.6}$$

$$\theta_r = \tan^{-1}\left(\frac{\dot{v}_{r\_1}\cos\psi_r + \dot{v}_{r\_2}\sin\psi_r}{\dot{v}_{r\_3}+9.8}\right)$$

And we make the Euler angle reference vector η_r using equation (5.6) and yaw angle reference vector ψ_r. It is given by

$$\eta_r = [\phi_r \quad \theta_r \quad \psi_r]^T. \tag{5.7}$$

5.3 Force Generator

Total force of the quadrotor expressed by the body frame can be redefined by translational z-axis dynamics of the quadrotor using the control input u_3' which replaces z-axis acceleration vector $\dot{v}_{r\_3}$. Force generator takes only the new control input u_3'. Because x and y-axis acceleration vectors $\dot{v}_{r\_1}$, $\dot{v}_{r\_2}$ is sufficiently considered to generate reference Euler angles which make desired torque so we don't consider the new control input u_1', u_2'. And $\cos\phi$ and $\cos\theta$ terms are to linearize quadrotor altitude dynamics. Equation (5.8) represents force generator.

$$f = m\left(\frac{u_3'+g}{\cos\phi\cos\theta}\right) \tag{5.8}$$

5.4 Attitude Controller by Computed Torque Method

In Chapter 2, we explain Computed Torque Method. This method is usually use robot manipulator controller. However, quadrotor have roll, pitch and yaw non-linear manner. And quadrotor is also rigid body. It is same as robot manipulator characteristics. So its method is suitable to the quadrotor attitude control. Its method considered the non-linear inner loop compensator and the outer feedback loop. The non-linear inner loop compensator is the key role to approximate linear model using the non-linear term feedback and calculate the torque. In this section, we explain quadrotor attitude controller using Computed Torque Method.

The rotational dynamics of the quadrotor can change the Euler angle representation. It is given by equation (5.9).

$$\tau = JC^{-1}\ddot{\eta} + J(\dot{C^{-1}})\dot{\eta} + C^{-1}\dot{\eta} \times (JC^{-1}\dot{\eta}) \tag{5.9}$$

where $C^{-1} = \begin{bmatrix} 1 & 0 & -\sin\theta \\ 0 & \cos\phi & \sin\phi\cos\theta \\ 0 & -\sin\phi & \cos\phi\cos\theta \end{bmatrix}$ and

$(\dot{C^{-1}}) = \begin{bmatrix} 0 & 0 & -\dot{\theta}\cos\theta \\ 0 & -\dot{\phi}\sin\phi & \dot{\phi}\cos\phi\cos\theta - \dot{\theta}\sin\phi\sin\theta \\ 0 & -\dot{\phi}\cos\phi & -\dot{\phi}\sin\phi\cos\theta - \dot{\theta}\cos\phi\sin\theta \end{bmatrix}$.

To control quadrotor attitude, choose the control input τ:

$$\tau = \alpha\tau' + \beta \tag{5.10}$$

where $\alpha = JC^{-1}$, $\beta = J(\dot{C^{-1}})\dot{\eta} + C^{-1}\dot{\eta} \times (JC^{-1}\dot{\eta})$.

And we introduce new control input τ':

$$\tau' = \ddot{\eta}_r + k_D\dot{e}_\eta + k_P e_\eta \tag{5.11}$$

where e_η is Euler angle error and $\dot{e}_\eta$ is Euler angle velocity error.

Then, closed loop rotational dynamics of the quadrotor is characterized by second order error dynamics.

$$\ddot{e}_\eta + k_D \dot{e}_\eta + k_P e_\eta = 0 \tag{5.12}$$

Convergence of the tracking error to zero is guaranteed [12] using equation (5.12). Figure 5.3 is described by the attitude controller via Computed Torque Method.

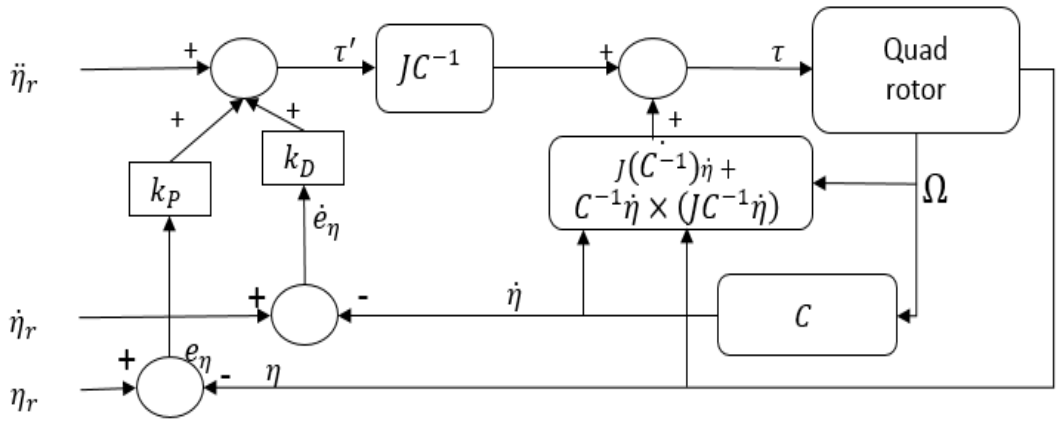

Figure 5.3: The structure of the attitude controller for the quadrotor.

Chapter 6

Simulations

In this chapter, we develop the quadrotor simulator the using previous chapters. In Section 6.1, we explain simulation parameter using simulator. Then we verify tracking performance and quadrotor system stability according to differential flatness-based motion planning. Finally, we introduce 3D visualization using Simulink 3D animation.

6.1 Simulation Parameters

Because quadrotor platforms are expensive and friable so simulation validations are important before experiment validations. And real model parameters are necessary. For the validation in the quadrotor simulation, The Ascending Technology Humming-bird [15] is considered. Its specification [8] is illustrated in Table 1.

Parameter mark	Value	Unit
m	0.6	Kg
g	9.8	m/s^2
J	diag(3.9×10^{-3},4.4×10^{-3},4.9×10^{-3})	m^2kg
k_F	6.11×10^{-8}	$N/_{rpm^2}$
k_m	1.5×10^{-9}	$Nm/_{rpm^2}$
l	0.17	m

Table 1. Parameter specification of the simulation (Hummingbird).

6.2 Simulation Results

Quadrotor can be used in disaster areas, surveillance and so on. For this reason, quadrotor is considered to have hovering capability and trajectory tracking.

To show hovering capability, we firstly verify quadrotor stability of one point trajectory tracking control according to differential flatness-based motion planning. And then, we verify trajectory tracking control performance using circle trajectory control. We consider MATLAB simulation.

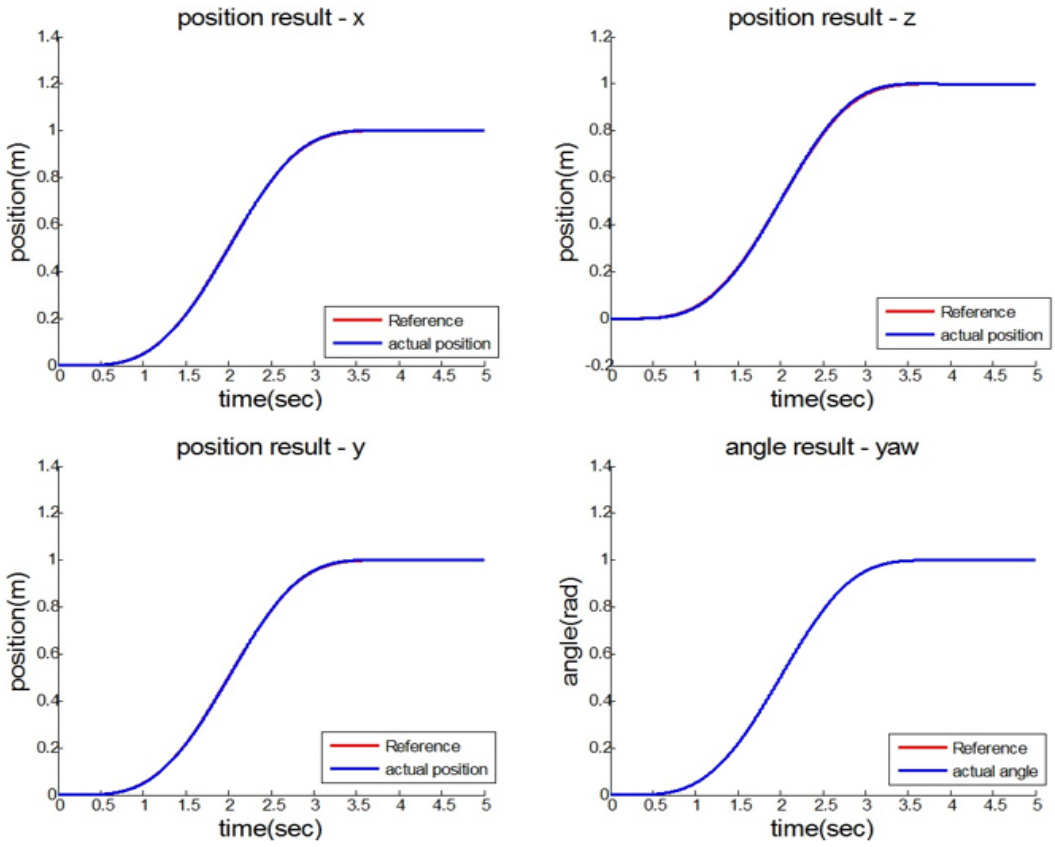

Figure 6.1: Results of one point tracking using motion planning(flat outputs).

Figure 6.1 is result of flat outputs of one point tracking using motion planning. Each flat output references are $y_{r\_1} = y_{r\_2} = y_{r\_3}$ are 1m and y_{r_4} is 1rad. Compare reference and real value, one point trajectory references considering motion planning show stable quadrotor dynamics.

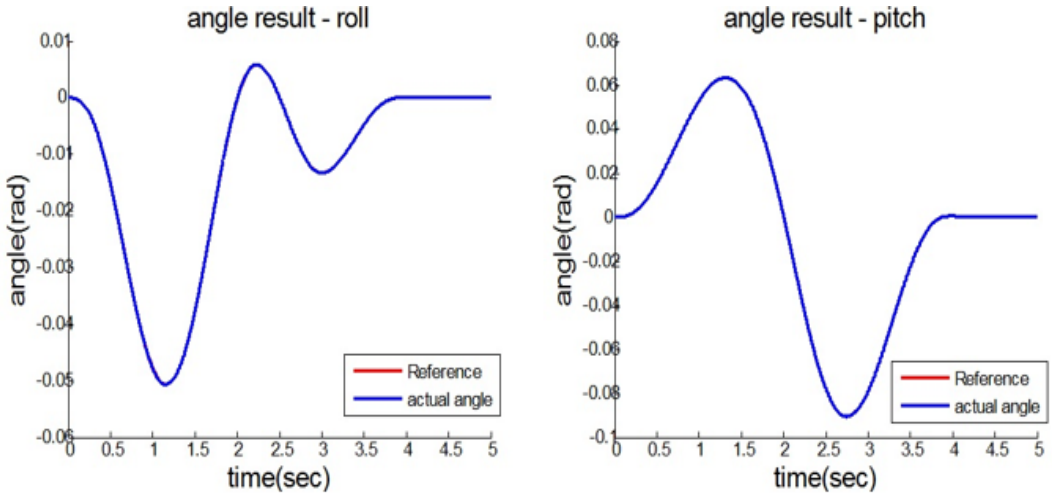

Figure 6.2: Results of one point tracking using motion planning(attitude).

Figure 6.2 is result of angles of one point tracking using motion planning. Compare reference and real value, attitude trajectory references considering motion planning also show stable quadrotor dynamics.

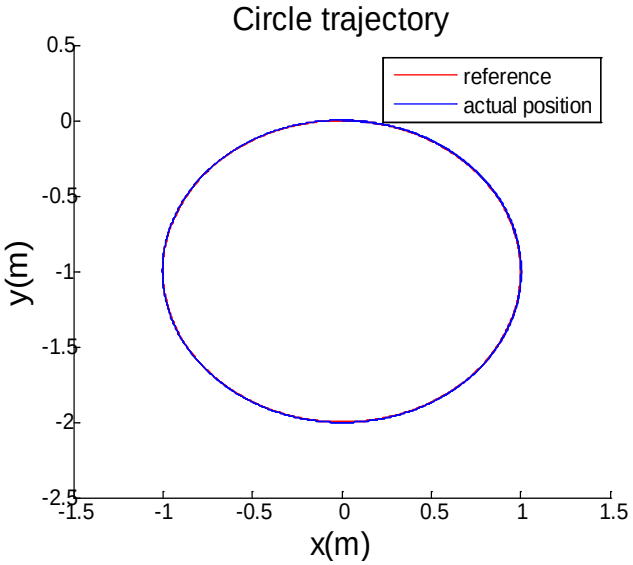

Figure 6.3: Result of circle trajectory tracking.

Figure 6.3 is result of flat outputs of circle trajectory tracking control. Each flat output reference radius is 1m. Circle trajectory references considering motion planning also show stable quadrotor dynamics.

6.3 3D Visualization

To validate the stability of the quadrotor, it is very important tool to visualize dynamic system behavior. This validation is possible thanks to blocks which called Simulink 3D Animation [16]. Furthermore, the 3D visualization allows to analyze the position and attitude of the quadrotor. The 3D quadrotor models are represented for the real trajectory results. Figure 6.3 shows 3D quadrotor model using Virtual Reality Modeling Language (VRML) which is proposed script language for 3D virtual environment in internet interface. Actually, there are 3D model development tools such as Solidworks, CAD, Catia and so on. We consider VRML for this thesis simulation.

An explicative video is linked here: https://youtu.be/By57LXeyO5M and https://www.youtube.com/watch?v=dA5BxAGpqbc.

Figure 6.3: 3D quadrotor model using VRML

Chapter 7

Conclusion and Future Work

In this thesis, the basics of quadrotor control are presented including reference frames, rotation matrix, kinematics and dynamics by Euler-Newton Equation. Then, differential flatness-based motion planning is considered for reference trajectory generation. Finally, position and attitude control of quadrotor dynamics are considered. PID type controller and Computed Torque Method are designed for position and attitude controls. The performance of the controller is validated using MATLAB simulations. The results show a stable dynamics despite the changes in roll, pitch and yaw motion in nonlinear manner.

The actual experimental validations are required because modeling error is presented in a variety of reasons. But this thesis considers only simulation validations so experimental validation of the actual quadrotor should be essential.

And disturbance is considered because of error of measurement and system environment variables. Disturbance acts on the control system in the form of additional input which is applied to the control input. The influence of disturbance must be analyzed because the influence of disturbance generates a system error so affect the stability of system.

Appendix

A. Simulator Code

For thesis validation, Results are validated using MATLAB simulations. MATLAB support MATLAB function block for simulations. In this appendix, we introduce 4 MATLAB function block: Flat output conversion, Attitude controller, Force generator and quadrotor dynamics.

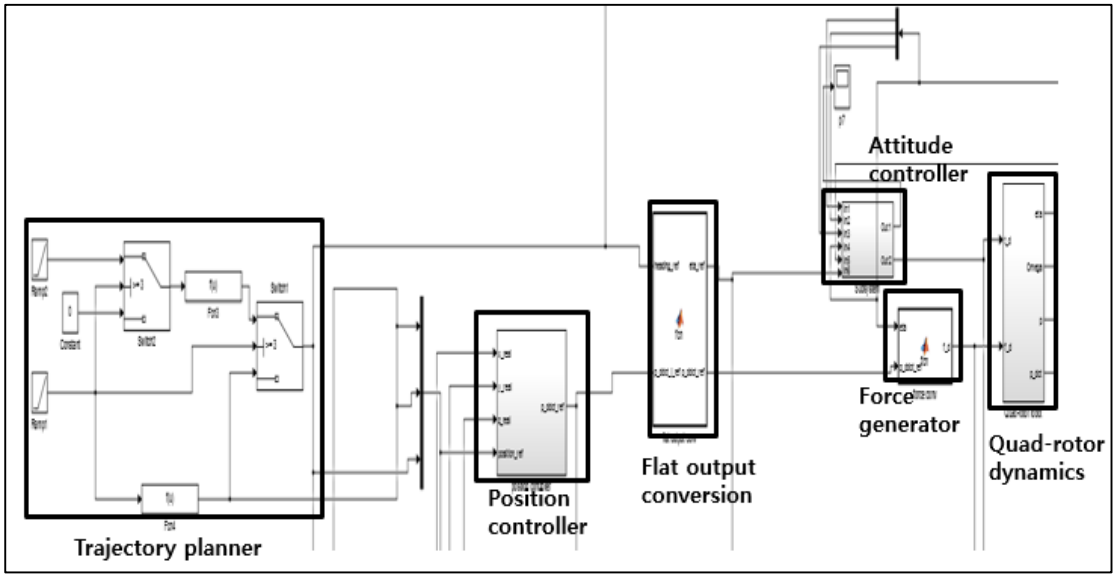

Figure A.1: Simulink block diagram

A.1 Flat Output Conversion

```
function [eta_ref, p_ddot_ref] = fcn(heading_ref, p_ddot_i_ref)

psi_ref = heading_ref(1,1);
x_ddot_ref = p_ddot_i_ref(1,1);
y_ddot_ref = p_ddot_i_ref(2,1);
```

```
z_ddot_ref = p_ddot_i_ref(3,1);
d =
sqrt(x_ddot_ref*x_ddot_ref+y_ddot_ref*y_ddot_ref+(z_ddot_ref+9.81)*(
z_ddot_ref+9.81));

phi_ref = asin((x_ddot_ref*sin(psi_ref) -
y_ddot_ref*cos(psi_ref))/(d));

theta_ref = atan((x_ddot_ref*cos(psi_ref) +
y_ddot_ref*sin(psi_ref))/(z_ddot_ref+9.81));

min_ang = -pi/2;
max_ang =  pi/2;

if (phi_ref < min_ang), phi_ref = min_ang; end
if (phi_ref > max_ang), phi_ref = max_ang; end

if (theta_ref < min_ang), theta_ref = min_ang; end
if (theta_ref > max_ang), theta_ref = max_ang; end

eta_ref   = [phi_ref; theta_ref; psi_ref];

p_ddot_ref = [x_ddot_ref;y_ddot_ref;z_ddot_ref];
```

A.2 Attitude Controller

```
function t_d = fcn(eta, O, eta_ddot_ref)

global J;

sphi = sin(eta(1,1));
cphi = cos(eta(1,1));
```

```
stht = sin(eta(2,1));
ctht = cos(eta(2,1));
spsi = sin(eta(3,1));
cpsi = cos(eta(3,1));

C = [1 0    -stht;
    0 cphi  sphi*ctht;
    0 -sphi cphi*ctht];

eta_dot = inv(C)*O;

phi_dot = eta_dot(1,1);
theta_dot = eta_dot(2,1);
psi_dot = eta_dot(3,1);

sphi = sin(eta(1,1));
cphi = cos(eta(1,1));
stht = sin(eta(2,1));
ctht = cos(eta(2,1));
spsi = sin(eta(3,1));
cpsi = cos(eta(3,1));

C_dot = [0 0            -theta_dot*ctht;
        0 -phi_dot*sphi  phi_dot*cphi*ctht-theta_dot*sphi*stht;
        0 -phi_dot*cphi  -phi_dot*sphi*ctht-theta_dot*cphi*stht];


O_hat = [0 -O(3,1) O(2,1);
        O(3,1) 0 -O(1,1);
        -O(2,1) O(1,1) 0];

t_d = J*C*eta_ddot_ref + O_hat*J*O + J*C_dot*eta_dot;
```

A.3 Force Generator

```
function f_d = fcn(eta, p_ddot_ref)

global m;

sphi = sin(eta(1,1));
cphi = cos(eta(1,1));
stht = sin(eta(2,1));
ctht = cos(eta(2,1));
spsi = sin(eta(3,1));
cpsi = cos(eta(3,1));

u = m*[0; 0; (p_ddot_ref(3,1)+ 9.81/(cphi*ctht))];

f_d = u(3,1);
```

A.4 Quadrotor Dynamics

```
function [O_dot, p_ddot, eta_dot] = fcn(O, eta, f_d, t_d)

global Kf Km L m g J; % variables

sphi = sin(eta(1,1));
cphi = cos(eta(1,1));
stht = sin(eta(2,1));
ctht = cos(eta(2,1));
spsi = sin(eta(3,1));
cpsi = cos(eta(3,1));
```

```
R = [cpsi*ctht cpsi*stht*sphi-spsi*cphi cpsi*stht*cphi+spsi*sphi;
    spsi*ctht spsi*stht*sphi+cpsi*cphi spsi*stht*cphi-cpsi*sphi;
    -stht    ctht*sphi             ctht*cphi]; % rotation matrix

C = [1 0    -stht;
    0 cphi  sphi*ctht;
     0 -sphi cphi*ctht];

O_hat = [0 -O(3,1) O(2,1);
        O(3,1) 0 -O(1,1);
        -O(2,1) O(1,1) 0]; % hatmap for omega

T = [Kf  Kf  Kf  Kf;
    0   L*Kf   0   -L*Kf;
    L*Kf   0   -L*Kf   0;
    -Km   Km   -Km   Km];

u = [f_d(1,1); t_d(1,1); t_d(2,1); t_d(3,1)]; % total force and
torque

w = inv(T)*u; % each motor angular velocity^2

F1 = Kf*w(1,1); % front motor force
F2 = Kf*w(2,1); % left motor force
F3 = Kf*w(3,1); % back motor force
F4 = Kf*w(4,1); % right motor force

t1 = Km*w(1,1); % front motor torque
t2 = Km*w(2,1); % left motor torque
t3 = Km*w(3,1); % back motor torque
t4 = Km*w(4,1); % right motor torque
```

```
F = [0; 0; F1+F2+F3+F4]; % total force with respect to body fixed
frame
T = [L*(F2-F4); L*(F1-F3); -t1+t2-t3+t4]; % total torque with re-
spect to body fixed frame

p_ddot = g + R*F/m; % translational dynamics
O_dot = inv(J)*(T -O_hat*J*O); % rotational dynamics
eta_dot = inv(C)*O; % rotational kinematics
```

Reference

[1] Bouabdallah, S., Noth, A., Siegwart, R. (2004). “PID vs LQ control Techniques Applied to an Indoor Micro Quadrotor” IEEE/RSJ International Conference on Intelligent Robots and Systems 2004, Sendal, Japan, pp. 2451-2456.

[2] Bouabdallah, S., Siegwart, R. (2005). “Backstepping and Sliding-mode Techniques Applied to an Indoor Micro Quadrotor” IEEE International Conference on Robotics and Automation 2005, Barcelona, Spain, pp. 2247-2252.

[3] Changsu, H., Zhiyuan, Z., Francis, C., Dongjun, L. (2014). “Passivity-based adaptive backstepping control of quadrotor-type UAVs” Robotics and Autonomous Systems, pp.1305-1315.

[4] Taeyoung, L. (2015). “Global Exponetial Attitude Tracking Controls on SO(3)” IEEE Transactions on Automatic Control, pp.1824-1829.

[5] Tse-Huai, W., Taeyoung, L. (2015). “Angular Velocity Observer for Velocity-Free Attitude Tracking Control on SO(3)” European Control Conference ,Linz, Austria, pp.2837-2842.

[6] Murray, R. (1996). “Trajectory Generation for A Towed Cable System using Differential Flatness” IFAC World Congress, San Francisco, United States of America.

[7] Justin, T., Loianno, G., Polin, J., Sreenath, K., Kumar, V. (2014). “Toward Autonomous Avian-Inspired Grasping for Micro Aerial Vehicles” Bioinspiration and Biomimetics, pp. 25010.

[8] Mellinger, D., Michael, N., Kumar, V. (2012). “Trajectory generation and control for precise aggressive maneuvers with quadrotors” The International Journal of Robotics Research, pp.664-674.

[9] Mellinger, D., Kushleyev, A., Kumar, V. (2012). “Mixed-Integer Quadratic Program Trajectory Generation for Heterogeneous Quadrotor Teams” IEEE International Conference on Robotics and Automation, Minnesota, United States of America, pp.477-483

[10] Turpin, M., Michael, N., Kumar, V. (2014). “Capt: Concurrent assignment and planning of trajectories for multiple robots” The International Journal of Robotics Research, pp.98-112

[11] Rechter, C., Bry, A., Roy, N. (2013). “Polynomial Trajectory Planning for Aggressive Quadrotor Flight in Dense Indoor Environments” In Proceedings of the International Symposium of Robotics Research (ISRR 2013).

[12] Craig, J. (2005). Introduction to Robotics Mechanics and Control, Pearson Education International, 402 pages.

[13] Beard, R. (2008). “Quadrotor Dynamics and Control” Brigham Young University, Utah, United

States of America, 47 pages

[14] Levine, J. (2009). Analysis and Control of Nonlinear Systems – A Flatness Based Approach, Springer, 331 pages.

[15] “Ascending Technologies, GmbH,” http://www.asctec.de.

[16] “Simulink 3D Animation User’s Guide” http://www.mathworks.co.kr.

요 약 문

미분 평탄성 및 토크 계산 제어를 이용한 쿼드로터 궤적 추종

무인비행기(UAV)는 정찰, 목표 추적, 환경 및 재난 모니터링 그리고 온라인 쇼핑 택배 배달까지 광범위한 곳에서 활용되고 있습니다. 이러한 활용을 위해서는, 무인비행기 위치 및 자세를 제어하는 것과 무인비행기의 동작 계획이 중요합니다. 하지만 UAV 위치 및 자세 제어기 설계에서 가장 어려운 부분은 롤, 피치 그리고 요에 관한 비선형 회전운동이 쿼드로터 위치 역학과 커플링 되어있다는 점입니다. 쿼드로터의 동작 계획 또한 중요합니다. 쿼드로터의 실현 가능한 동작과 모순된 궤적 계획은 제어기 설계를 어렵게 만들며, 좋지 않는 추종 성능을 야기합니다. 이 논문에서는 쿼드로터의 자세와 위치 동작 궤적 추종 제어를 고려했습니다. 첫 번째로, 쿼드로터 동역학과 관련된 기준 좌표계, 회전 행렬, 힘과 모멘트, 뉴턴 오일러 방식으로 기술하는 기구학 및 동역학에 대해 기술하였습니다. 다음으로, 쿼드로터의 기준 궤적 생성을 위해서 미분 평탄성을 기반으로 한 동작 계획에 대해서 설명하였습니다. 마지막으로, 쿼드로터 위치 및 자세 제어를 위한 PID 위치 제어기 및 토크 계산법을 이용한 자세 제어기를 설계하였습니다. 논문 결과는 MATLAB 시뮬레이션으로 검증하였습니다.

핵심어: 미분 평탄성, 동작 계획, 궤적 추종 제어, 쿼드로터, 무인비행기