DESIGN & CONTROL OF

AN OMNI-DIRECTIONAL QUAD-ROTOR

A Thesis

by

JAEWON KIM

Submitted to the Office of Graduate and Professional Studies of
Texas A&M University
in partial fulfillment of the requirements for the degree of

MASTER OF SCIENCE

Chair of Committee,	Raktim Bhattacharya
Committee Members,	Aniruddha Datta
	Moble Benedict
	Srinivas Shakkottai
Head of Department,	Miroslav M. Begovic

August 2017

Major Subject: Electrical Engineering

ABSTRACT

Unmanned Aerial Vehicle (UAV) is an unmanned air vehicle which can be operated by human or fly autonomously on the basis of flight plans. UAVs are usually utilized for military purposes that are too tedious, dirty, risky, or hazardous for normal manned air vehicles; however, they are also utilized for civil purposes like aerial photography or air surveillance. There are two types of UAVs. One is the fixed-wing UAV, i.e. an airliner, the other is the rotor-wing UAV, i.e. a helicopter. Rotor-wing UAVs have the weather gauge of fixed-wing UAVs. Because they can perform Vertical Take-Off and Landing (VTOL); it is able to hover at particular point. The advantages of the rotor-wing is as follows. First, it is mechanically simple; it's main components are n motors and n propellers. Second, they do not require complex mechanical parts to control their flight; it can fly and maneuver only by changing the speed of the motors. One of the successful design example is a four rotor UAV, also known as quadrotor.

In this work, design and control of an omni-directional quadrotor model is developed and simulated by using tilt-rotor mechanism. And also, a mathematical model of the quadrotor's dynamics is derived using Newton's law and Euler's law. In addition, linearized models are obtained, and therefore a linear controller, the Linear Quadratic Regulator (LQR), is derived. After that, non-linear controller for the quadrotor is provided. Finally, the behavior of the quadrotor under the proposed control strategies is observed in simulation by using the MATLAB, Simulink and Simmechanics.

DEDICATION

I dedicate this work to my parents. They have supported my master's program with great care. Thanks to them, I can finish my master's study and research successfully.

ACKNOWLEDGMENTS

I would like to thank my committee chair, Dr. Raktim Bhattacharya, and my committee members, Dr. Aniruddha Datta, Dr. Moble Benedict, and Dr. Srinivas Shakkottai for their guidance and support throughout the course of this research and carefully reviewing my research materials.

Thanks also go to my friends and colleagues and the department faculty and staff for making my time at Texas A&M University a great experience. Next, thanks to my mother and father for their encouragement, patience and love.

Finally, I would like to thank the Texas A&M University Office of Graduate and Professional Studies to allow me to construct this LaTeX thesis template.

CONTRIBUTORS AND FUNDING SOURCES

Contributors

This work was supervised by a thesis committee consisting of Professors Raktim Bhattacharya [advisor], Moble Benedict of the Department of Aerospace Engineering and Professors Aniruddha Datta, Srinivas Shakkottai of the Department of Electrical & Computer Engineering.

All work for the thesis was completed by the student, under the advisement of Professor Raktim Bhattacharya of the Department of Aerospace Engineering, Electrical & Computer Engineering.

Funding Sources

There are no outside funding contributions to acknowledge related to the research and compilation of this document.

NOMENCLATURE

UAV	Unmanned Aerial Vehicle
UAVs	Unmanned Aerial Vehicles
VTOL	Vertical Take Off and Landing
MPC	Model Predictive Control
LQR	Linear Quadratic Regulator
CAD	Computer Aided Design
MIMO	Multiple Input Multiple Output
DOF	Degree of Freedom

TABLE OF CONTENTS

LIST OF FIGURES

1. INTRODUCTION

1.1 Motivation

UAVs are getting more ubiquitous or omnipresent in modern research areas or industry fields [1]. Also, UAVs are utilized a wide range of applications including intelligence gathering, surveillance [2], rescue [3], instantaneous response, urban combat, and wireless sensor networks [4].

The helicopters and airplanes have become available in facilitating human life, and providing a bunch of application areas with large scale productions. The low speed limit points of airplanes and high speed limit points of helicopters have made their application areas different. In spite of the fact that a lot of exertion has been spent to consolidate the advantages of these aircraft into one with disposing of disadvantages such as tilt-wings and tail-sitters and none has been sufficiently effective until tilt-rotors [5].

Omni-directional quadrotor can help people in many areas such as agriculture applications or air surveillance purposes.

1.2 Related works

The idea of a tilt-rotor UAV has been studied before in simulation and actualized physical systems. A. Sanchez et al.[6] could design and implement a tilt-rotor system with dynamic equations and control laws. Also, Christos Papachristos et al.[7] researched the development of tilt-rotor UAV as a feasible platform for autonomous rescue actions. Another work into the development of a MPC scheme [8] and an open source platform [9] provide a continued interest in tilt-rotor UAVs to consolidate the mobility of helicopters with fixed wing UAV's long distance flight. [10]. More recent work presented the design, analysis, and implementation of a tilt-rotor UAV; and the work proposed two nacelle systems for changing thrust vectors. The UAV has the ability to tilt the thrust output using dual-nacelle [11].

1.3 Research objective

My work presents the design and control of an omni-directional quadrotor with tilt-rotor system, tiltable rotor mount. The proposed system utilizes four tiltable rotor mount for tilt mechanism which is for changing the direction of thrust or thrust vectors.

1.4 Thesis contributions

The primary purpose of this thesis is the design & control of an omni-directional quadrotor. These system concepts are provided for future research on the advanced controller design for aggressive maneuvers of UAVs. Suggested are five hovering models of the tiltrotor UAV. From the horizontal hovering model to vertical hovering model. To achieve the desired thesis goal, the contributions of the thesis are as follows:

- A study on the possibility of the design and control of an omni-directional quadrotor with tiltable rotor mount for stable maneuvering and hovering.
- A multi-body dynamic model of an omni-directional quadrotor was suggested based on the Newton-Euler method for control and simulation.
- A simplified dynamic model of the UAV was also provided for the vehicle system using a single body approach to simplify the control implementation.
- A MATLAB Simulink and Simmechanics block was implemented based on the dynamic model of the tilt-rotor UAV.
- The Simulink and Simmechanics block was used to simulate the UAV, verify hovering conditions, and control strategies.

2. MATHEMATICAL MODEL OF THE UAV

2.1 Primary notations

The quadrotor, an aircraft made up of four rotors, holds the electronic board in the middle and the engines at four ends. Before describing the mathematical model of a quadrotor, it is necessary to introduce the reference coordinates in which we describe the structure and the position. For the quadrotor, it is possible to use two reference systems. The first is fixed and the second is mobile. The fixed coordinate system, called also inertial, is a system where the first Newton's law is considered valid.

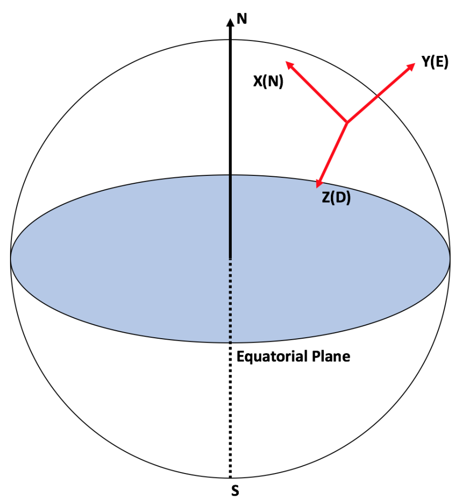


Figure 2.1: The NED fixed reference frame

As fixed coordinate system, we use the O_{NED} systems, where NED is for North-East-Down. As we can observe from the above Figure 2.1, its vectors are directed to North, East and to the center of the Earth.

The mobile reference system, called also body-fixed that we have previously mentioned is united with the barycenter of the quadrotor. In the scientific literature it is called O_{ABC} system, where ABC is for *Aircraft Body Center*. In the below Figure 2.2, it presents underlines the two coordinate systems.

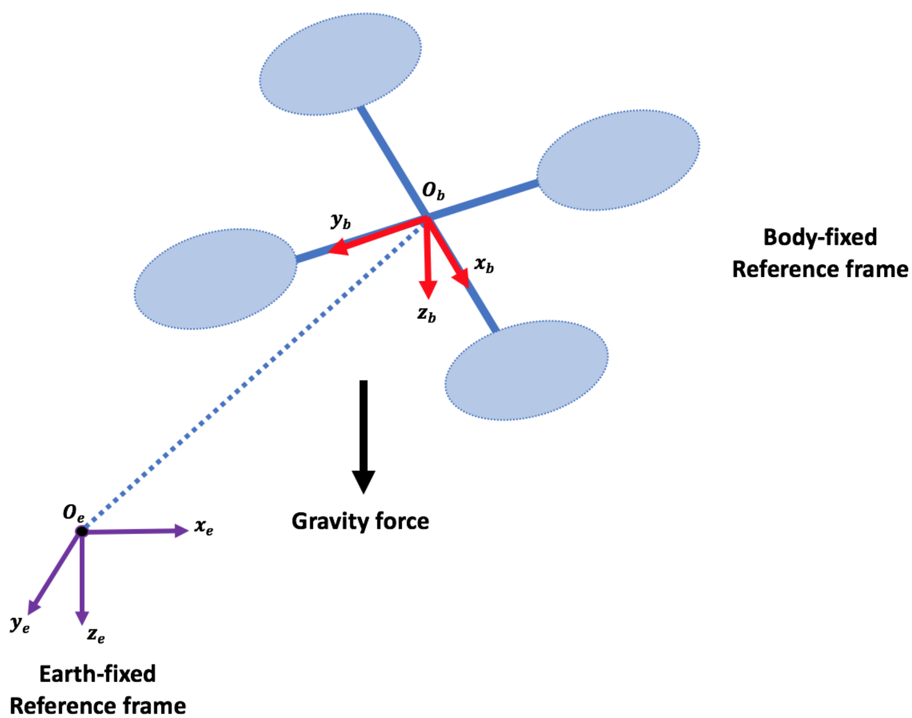


Figure 2.2: The earth-fixed reference frame and the body fixed reference frame

The earth-fixed inertial reference frame (0_e X_e Y_e Z_e) is a right-handed orthogonal axis-system with the origin at the quadrotor's centre of gravity at the beginning of the considered motion. This reference frame is fixed to the earth and is considered as the inertial frame of reference under simplifying conditions.

The body-fixed reference frame (0_b X_b Y_b Z_b) is a right-handed orthogonal axis-system with the origin at the quadrotor's centre of gravity. The reference frame remains fixed to the quadrotor even in perturbed motion.

The absolute position of the quadrotor is described by the three coordinates (x,y,z) of the centre of mass with respect to the earth reference frame.

The absolute attitude of quadrotor is described by the three Euler's angles (ψ,θ,ϕ). These three angles are respectively called yaw angle ($-\pi \leq \psi < \pi$), pitch angle ($-\pi/2 < \theta < \pi/2$) and roll angle ($-\pi/2 < \phi < \pi/2$).

2.2 Quadrotor dynamics

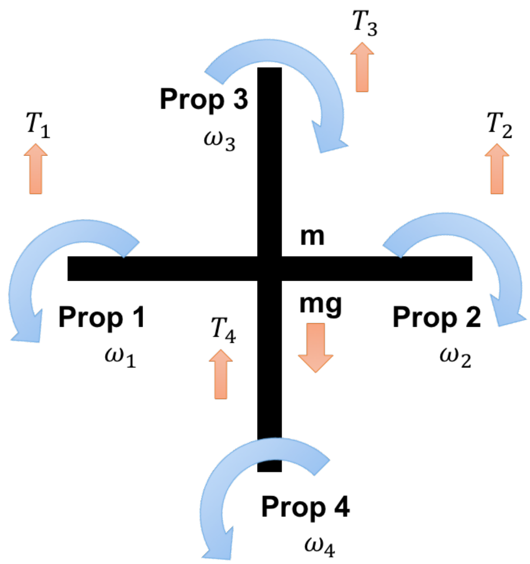


Figure 2.3: Main forces

ω_1 ω_2 ω_3 ω_4 : angular velocity of the propellers

T_1 T_2 T_3 T_4 : forces generated by the propellers

M_1 M_2 M_3 M_4 : moments generated by the propellers

m : mass of the quadrotor

mg : gravity force (weight of the quadrotor)

Equilibrium of forces : $\sum_{i=1}^{4} T_i = -mg$

Equilibrium directions : $T_1, T_2, T_3, T_4 \parallel g$

Equilibrium moments : $\sum_{i=1}^{4} M_i = 0$

Equilibrium rotation speeds : $(\omega 2 + \omega 3) - (\omega 1 + \omega 4) = 0$

2.3 Quadrotor dynamics modeling

The represented position(p = $[x\ y\ z]^T$) of the UAV in the inertia frame and the euler's angle(η = $[\phi\ \theta\ \psi]^T$) are related with the linear velocity(v = $[v_x\ v_y\ v_z]^T$) and angular velocity(ω = $[\omega_x\ \omega_y\ \omega_z]^T$) in the body-fixed frame.

$$\begin{aligned} \dot{P} &= Rv \\ \omega &= C\dot{\eta} \end{aligned} \tag{2.1}$$

R is the body-fixed frame rotation matrix about inertial frame.

$$R = R_z(\psi)R_y(\theta)R_x(\phi) \tag{2.2}$$

$$R = \begin{bmatrix} R_{11} & R_{12} & R_{31} \\ R_{21} & R_{22} & R_{32} \\ R_{31} & R_{32} & R_{33} \end{bmatrix}$$

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

$$R_x(\phi)=\begin{bmatrix}1 & 0 & 0\\ 0 & \cos(\phi) & -\sin(\phi)\\ 0 & \sin(\phi) & \cos(\phi)\end{bmatrix}\quad R_y(\theta)=\begin{bmatrix}\cos(\theta) & 0 & \sin(\theta)\\ 0 & 1 & 0\\ -\sin(\theta) & 0 & \cos(\theta)\end{bmatrix}\quad R_z(\psi)=\begin{bmatrix}\cos(\psi) & -\sin(\psi) & 0\\ \sin(\psi) & \cos(\psi) & 0\\ 0 & 0 & 1\end{bmatrix}$$

C is the matrix represent the relationship between the velocity component of euler's angle in the inertial frame and angular velocity vector of body-fixed frame.

$$\mathbf{R}_z(\psi)\mathbf{R}_y(\theta)\mathbf{R}_x(\phi)\begin{bmatrix}\omega_x\\ \omega_y\\ \omega_z\end{bmatrix}=\begin{bmatrix}0\\ 0\\ \dot{\psi}\end{bmatrix}+\mathbf{R}_z(\psi)\begin{bmatrix}0\\ \dot{\theta}\\ 0\end{bmatrix}+\mathbf{R}_z(\psi)\mathbf{R}_y(\theta)\begin{bmatrix}\dot{\phi}\\ 0\\ 0\end{bmatrix}$$

$$\begin{bmatrix}\omega_x\\ \omega_y\\ \omega_z\end{bmatrix}=\begin{bmatrix}\dot{\phi}\\ 0\\ 0\end{bmatrix}+\mathbf{R}_x^T(\phi)\begin{bmatrix}0\\ \dot{\theta}\\ 0\end{bmatrix}+\mathbf{R}_x^T(\phi)\mathbf{R}_y^T(\theta)\begin{bmatrix}0\\ 0\\ \dot{\psi}\end{bmatrix}$$

$$\begin{bmatrix}\omega_x\\ \omega_y\\ \omega_z\end{bmatrix}=\mathbf{C}\begin{bmatrix}\dot{\phi}\\ \dot{\theta}\\ \dot{\psi}\end{bmatrix},\quad \mathbf{C}=\begin{bmatrix}1 & 0 & -\sin\theta\\ 0 & \cos\phi & \sin\phi\cos\theta\\ 0 & -\sin\phi & \cos\phi\cos\theta\end{bmatrix}$$

After Differentiating equation (1),

$$\ddot{P} = R\dot{v} + \dot{R}v$$
$$\dot{\omega} = C\ddot{\eta} + \dot{C}\dot{\eta} \tag{2.3}$$

After rearrange the above equations,

$$\ddot{P} = R(\dot{v} + \omega \times v)$$
$$\dot{\omega} = C\ddot{\eta} + \dot{C}\dot{\eta} \tag{2.4}$$

Because $\dot{R}v = \omega \times (Rv)$

$$\dot{\mathbf{C}} = \left(\frac{\partial \mathbf{C}}{\partial \phi}\dot{\phi} + \frac{\partial \mathbf{C}}{\partial \theta}\dot{\theta} + \frac{\partial \mathbf{C}}{\partial \psi}\dot{\psi} \right) = \begin{bmatrix} 0 & 0 & -\dot{\theta}\cos\theta \\ 0 & -\dot{\phi}\sin\phi & \dot{\phi}\cos\phi\cos\theta - \dot{\theta}\sin\phi\sin\theta \\ 0 & -\dot{\phi}\cos\phi & -\dot{\phi}\sin\phi\cos\theta - \dot{\theta}\cos\phi\sin\theta \end{bmatrix}$$

Using Newton's 2nd law,

$$m\dot{v} + \omega \times (mv) = F + F_g$$
$$I\dot{\omega} + \omega \times (I\omega) = Q - Q_G \tag{2.5}$$

where, m is the mass of the UAV, I is mass moment of inertia. $m\dot{v}$ is the force generated by acceleration, $\omega \times (mv)$ is the centripetal force.

Since the UAV is designed line symmetry, the moments of inertia is like below. Especially, $I_{xx} = I_{yy}$

$$I = \begin{bmatrix} I_{xx} & 0 & 0 \\ 0 & I_{yy} & 0 \\ 0 & 0 & I_{zz} \end{bmatrix}$$

F_g is the gravity on the UAV. Since this gravity force should represent in the body-fixed frame, we should rotate the gravity vector(g° = [0 0 -g]^T) in the inertial frame to body-fixed frame.

$$F_g = mR^T g^\circ \tag{2.6}$$

Because $R^T = R^{-1}$

Q_G is the gyro effect, this is defined from the angular velocities of four rotors.

$$Q_G = \omega \times I_R \Omega_G \tag{2.7}$$

where, I_R is the moment of inertia, Ω_G = [0 0 $\Omega_1 - \Omega_2 + \Omega_3 - \Omega_4]^T$.

F is the force and G is the moment to control the UAV, this is related with angular velocities of four rotors like below.

F = [0 0 $F_1 + F_2 + F_3 + F_4]^T$

G = [$\ell(F_4 - F_2)$ $\ell(F_3 - F_1)$ $\tau_1 - \tau_2 + \tau_3 - \tau_4]^T$

where, $F_i = K_t\Omega_i^2$, $\tau_i = K_d\Omega_i^2$, ℓ is the distance between rotors, K_t and K_d is coefficient of thrust and torque related with ith rotor's angular velocity Ω_i.

From equation (4) and (5),

$$
\begin{aligned}
mR^T\ddot{p} &= F + mR^T g^\circ \\
\ddot{p} &= g^\circ + \frac{1}{m}RF
\end{aligned}
\tag{2.8}
$$

Also, the angular velocities of the UAV in the inertial frame is defined like below.

$$
\begin{aligned}
I(C\ddot{\eta} + \dot{C}\dot{\eta}) + C\dot{\eta} \times (IC\dot{\eta}) &= Q - C\dot{\eta} \times I_R\Omega_G \\
\ddot{\eta} &= (IC)^{-1}(Q - I\dot{C}\dot{\eta} - C\dot{\eta} \times (IC\dot{\eta} + I_R\Omega_G))
\end{aligned}
\tag{2.9}
$$

3. VEHICLE MODELING

3.1 UAV model design

CAD software, Solidworks, was utilized during the vehicle model design process to build rigid-body model of the UAV as is shown below in Fig. 3.1.

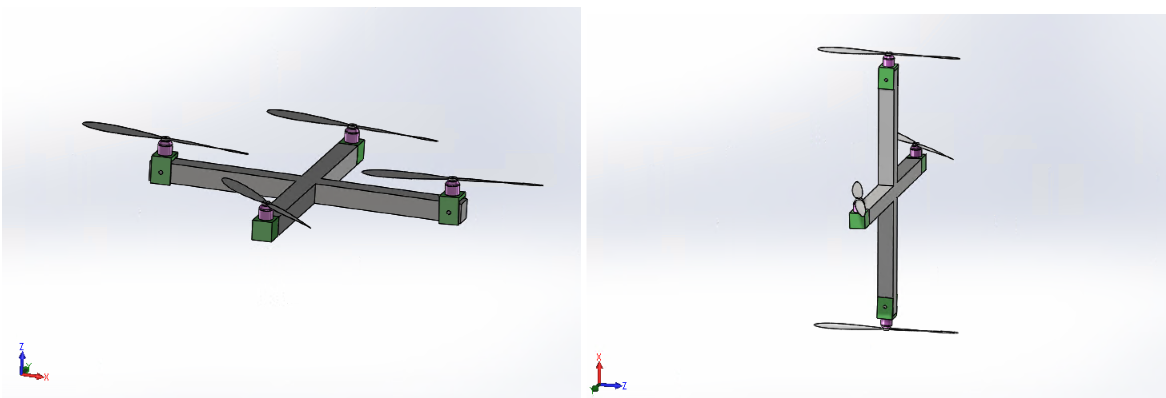


Figure 3.1: CAD model for omni-directional UAV

This aircraft model has the ability to hover under any euler's angle; therefore, by changing thrust vector of the tilt-rotor UAV, the vehicle can change their body angle according to the environments. The tiltable rotor mount can rotate their angle between -90° to 90°. In the above figure, the left one is normal (horizontal) hovering model, which rotor mount tilted angle is 0°, and right one is vertical hovering model, which rotor mount tilted angle is 90°. In this UAV system, there are 8 inputs, which is torques and angles of the each rotors and there are 13 outputs, which is I measured, position[x,y,z], linear velocity[u,v,w], angular position is represented by quaternion [q_1, q_2, q_3, q_4], and angular velocity[p,q,r]. Euler angle can calculated from quaternion values. Therefore, this tilt-rotor system is MIMO system.

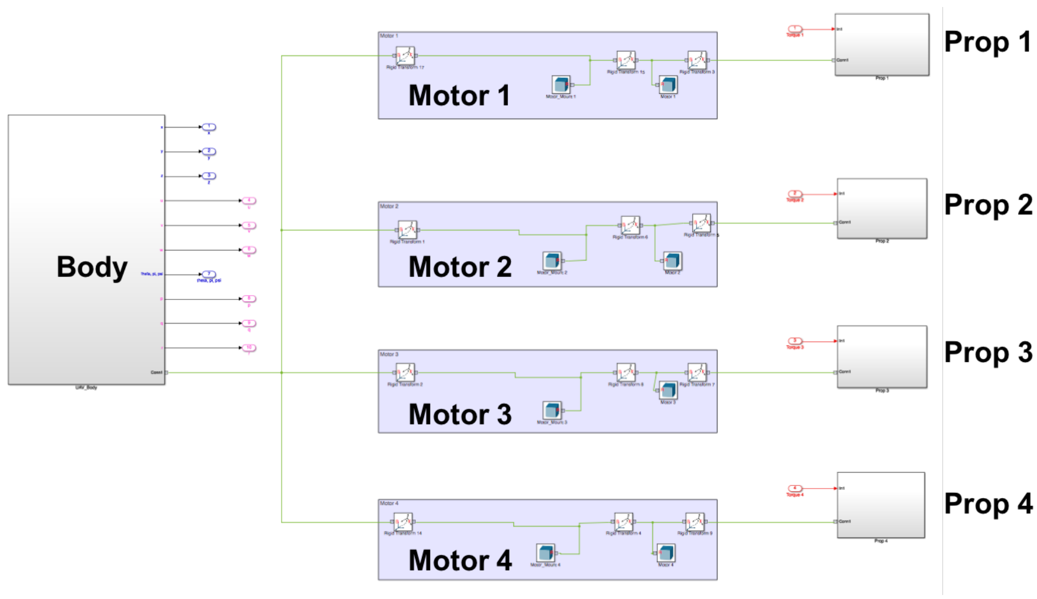


Figure 3.2: Overview of simulink and simmechanics model

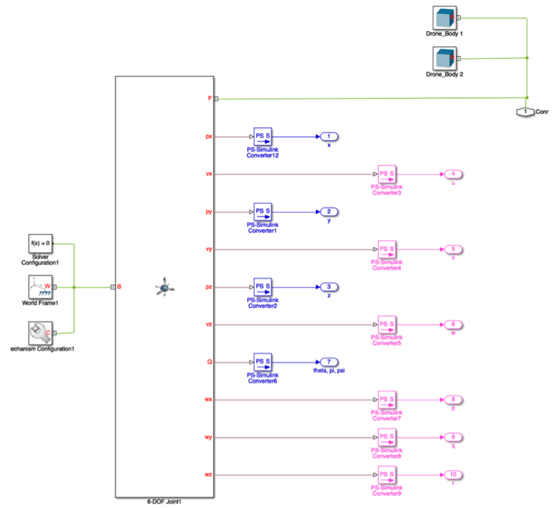


Figure 3.3: The simulink model for quadrotor rigid body

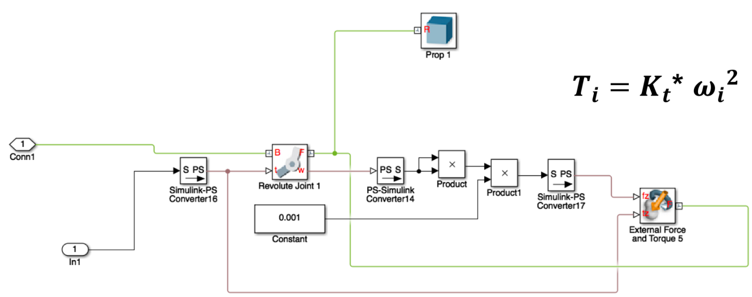


Figure 3.4: The simulink model for propeller dynamics

4. METHODOLOGY

4.1 Control design

This tilt-rotor UAV has the 4-rotors and in free flight 6-DOF; it is an under-actuated and unstable dynamically. Therefore, the controller or regulator design is a difficult work. Open-loop output of quadrotor is unstable; thus, feedback control is required to be able to fly the UAV.

4.2 Linearization

The control vector $u = [T_1, T_2, T_3, T_4]^T$. The linearization process is developed at certain equilibrium point $\bar{x}$, which is for certain input $\bar{u}$.

$$\hat{f}(\bar{x}, \bar{u}) = 0. \tag{4.1}$$

4.3 Linear Quadratic Regulator (LQR)

The objective of the optimal control is to determine control signal so that the system to be controlled. The LQR is one of the optimal control method that minimize a certain cost function. In other words, the optimization problem's solution is supposed to bring the state of system $x(t)$ to the desired value x_d minimizing some cost.

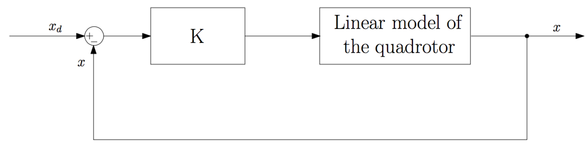


Figure 4.1: Linear Quadratic Regulator (LQR) control

Let's consider a dynamic system and set x as state of the system ans set u as input of the system.

$$\dot{x(t)} = f[x(t), u(t), t] \tag{4.2}$$

$$J = e[x(t_f)] + \int_{t_0}^{t_f} w[x(t), u(t), t]dt \tag{4.3}$$

where w is *the weight function* and e is the *final cost*; both of them are non-negative fuction such as $w(0,0,t) = 0$, $e(0) = 0$.

The objective is minimizing J.

$$\begin{cases} \dot{x} = A \cdot x + B \cdot u \\ y = C \cdot x \end{cases} \tag{4.4}$$

$$J = \int_{t_0}^{\infty} \{u(t)^T \cdot R \cdot u(t) + [x(t) - x_d(t)]^T \cdot Q \cdot [x(t) - x_d(t)]\}dt \tag{4.5}$$

- R is the cost of the actuators ($R = R^T$, positive definite matrix)
- Q is the cost of the states ($Q = Q^T$, positive semi-definite matrix)

$$u(t) = -K \cdot [x(t) - x_d(t)] \tag{4.6}$$

where

$$K = R^{-1} \cdot B^T \cdot S \tag{4.7}$$

The S is the Riccati's algebraic equation's solution matrix.

$$S \cdot A + A^T \cdot S - S \cdot B \cdot R^{-1} \cdot B^T \cdot S + C^T \cdot Q \cdot C = 0 \tag{4.8}$$

where S is a positive definite matrix.

The Riccati's algebraic equation can be solved by using MATLAB LQR fuction.

$$K = LQR(A, B, Q, R) \tag{4.9}$$

we can get

- There exists one soution for positive sefinite S of the Riccati's algebraic equation.

- The closed-loop system $\dot{x} = (A - B * K) * x$ is asymptotically stable with K

4.4 Controller application

In my model, there are 4 inputs and there are 13 outputs. Using this non-linear tilt-rotor UAV model, doing linearization first, for obtaining linear model of both horizontal hovering and vertical hovering. After get linear models, design and apply controller or regulator for hovering using LQR. The result of closed-loop system should be asymptotically stable with K.

4.5 Research issues

During last few decades, there are various way to design and control tilt-rotor UAVs have been provided, we need to find a way to find optimal and robust method to design and control the UAVs for various natural environments. In this thesis, using tiltable rotor mount, UAV can change their thrust vectors easily; from this, the UAV can change their body angle. Therefore, with this tilt mechanism, the UAV can fly under any arbitrary angles, omni-directional. Therefore, in this thesis, two important following issues will be focused:

- Tilt mechanism design for omni-directional UAV
- Control method for hovering under certain angles using the tilt mechanism

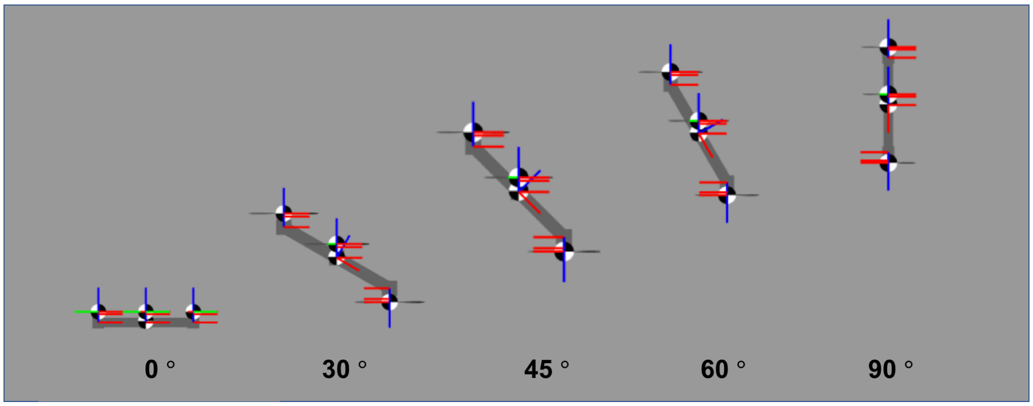


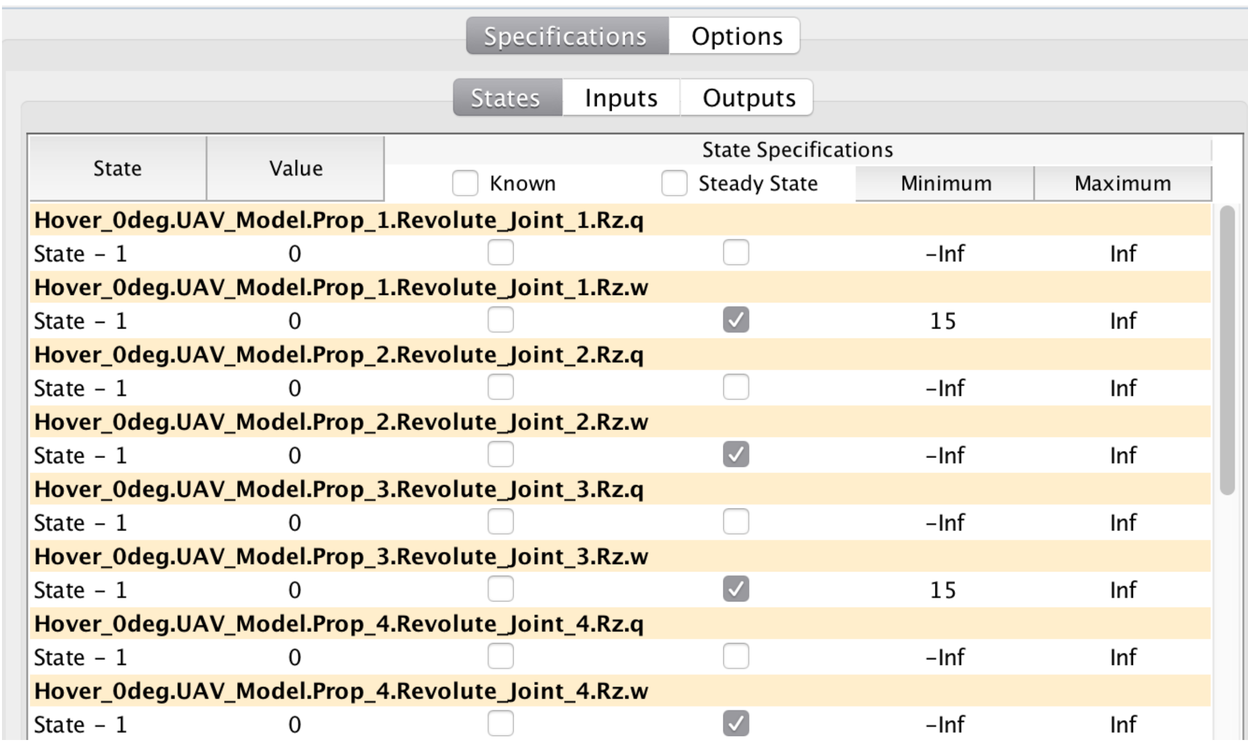
Figure 5.1: Five hovering modes

Euler angle	q1	q2	q3	q4
0°	1	0	0	0
30°	0.96593	0	0.25882	0
45°	0.92388	0	0.38268	0
60°	0.86603	0	0.5	0
90°	0.70711	0	0.70711	0

Figure 5.2: The relationship between euler angle and quaternion

5.1 Horizontal hovering case (0deg rotation of rotor mount)

First, I did the linearizaion at the operating point 0deg. The linearization conditions as are shown below in Fig. 5.3.



State	Value	Known	Steady State	Minimum	Maximum
Hover_0deg.UAV_Model.Prop_1.Revolute_Joint_1.Rz.q					
State - 1	0	☐	☐	-Inf	Inf
Hover_0deg.UAV_Model.Prop_1.Revolute_Joint_1.Rz.w					
State - 1	0	☐	☑	15	Inf
Hover_0deg.UAV_Model.Prop_2.Revolute_Joint_2.Rz.q					
State - 1	0	☐	☐	-Inf	Inf
Hover_0deg.UAV_Model.Prop_2.Revolute_Joint_2.Rz.w					
State - 1	0	☐	☑	-Inf	Inf
Hover_0deg.UAV_Model.Prop_3.Revolute_Joint_3.Rz.q					
State - 1	0	☐	☐	-Inf	Inf
Hover_0deg.UAV_Model.Prop_3.Revolute_Joint_3.Rz.w					
State - 1	0	☐	☑	15	Inf
Hover_0deg.UAV_Model.Prop_4.Revolute_Joint_4.Rz.q					
State - 1	0	☐	☐	-Inf	Inf
Hover_0deg.UAV_Model.Prop_4.Revolute_Joint_4.Rz.w					
State - 1	0	☐	☑	-Inf	Inf

Figure 5.3: Linearization condition for 0deg 1

Each propeller rotors have rotation angle and angular velocity. “Rz.q" means rotation angle, “Rz.w" means angular velocity. I set initial minimum value for “rotor1" and “rotor3" as “15" for initial linearizing condition.

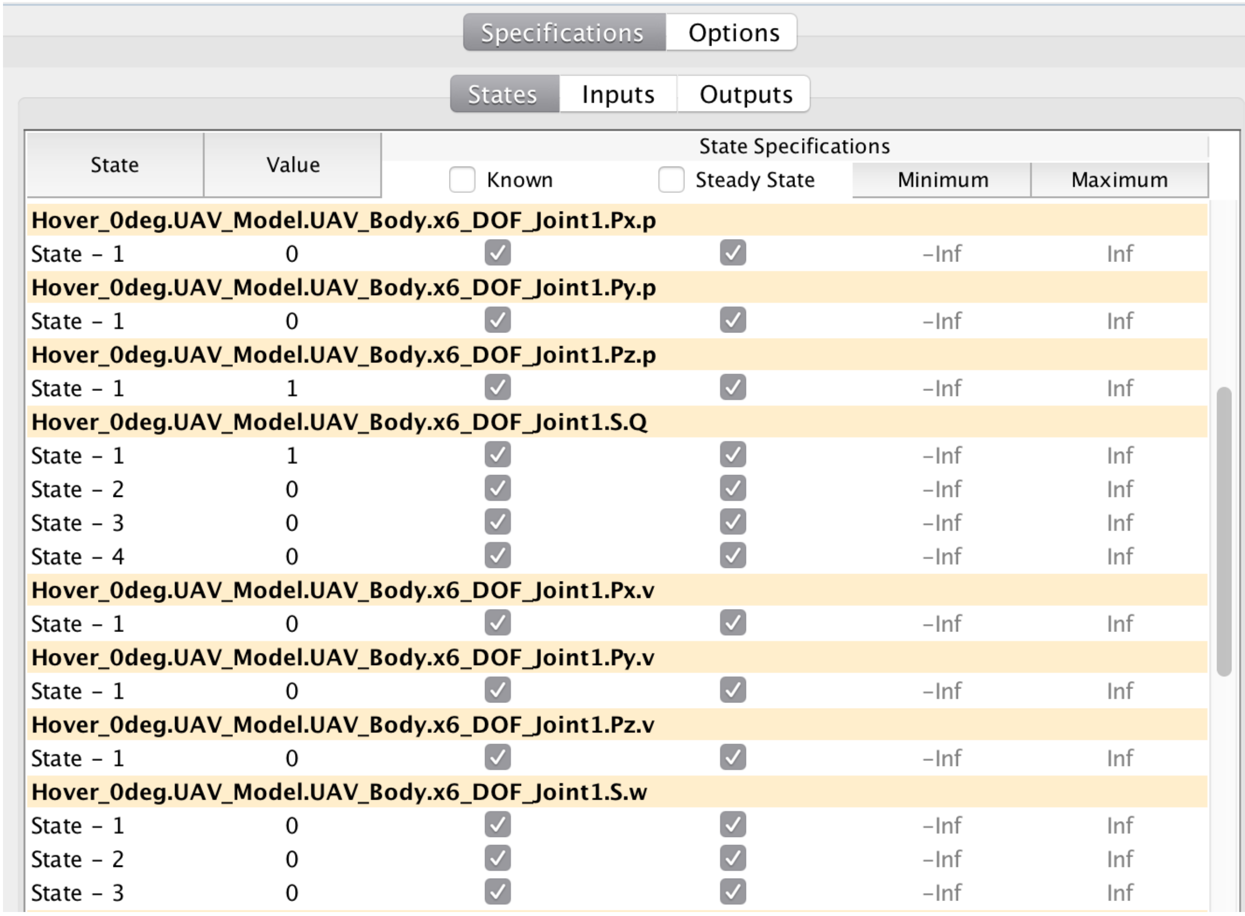
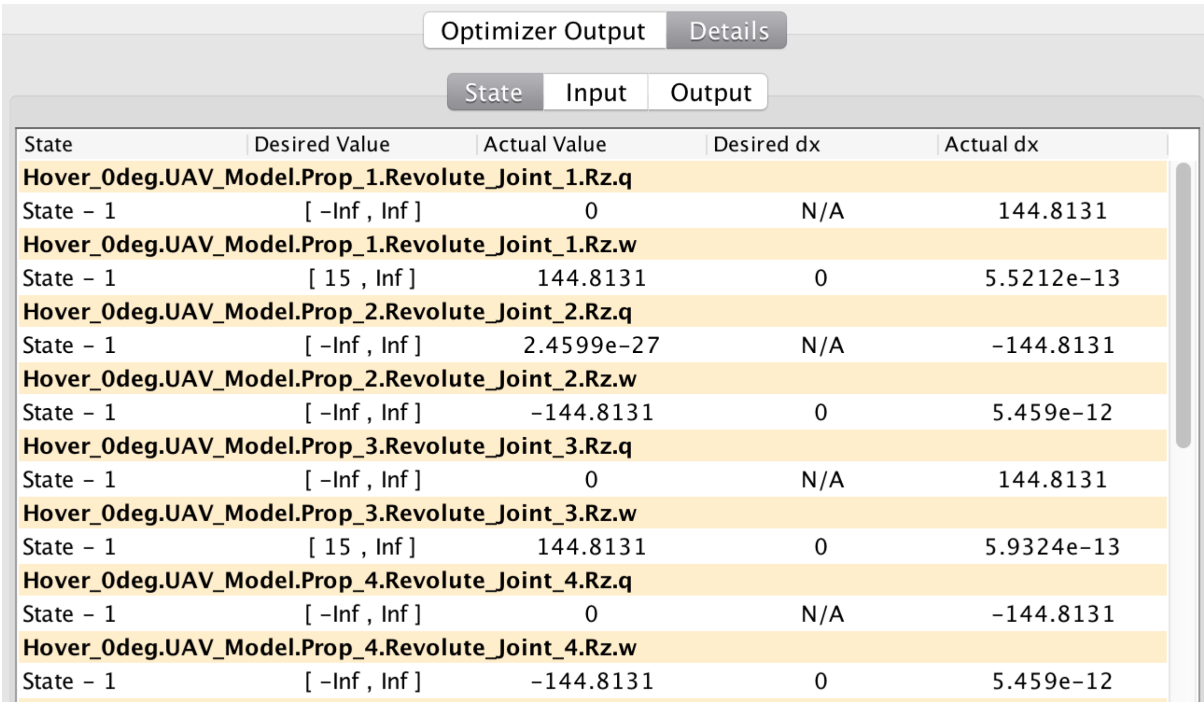


Figure 5.4: Linearization condition for 0deg 2

And then, the UAV has 6-DOF motions. For hovering, I set some values. “Px.p, Py.p, Pz.p" means “x, y, z". I set this value as “0, 0, 1". “S.Q" means quaternion of UAV. I set this value “1, 0, 0, 0" because “0deg" means “no rotatioin" at that time, the quaternion value is “1, 0, 0, 0". “Px.v, Py.v, Pz.v" means “linear velocity of UAV (u, v, w)". I set this value “0, 0, 0" because of hovering. “S.w" means angular velocity of UAV (p, q, r). I set this value “0, 0, 0" for hovering motion.

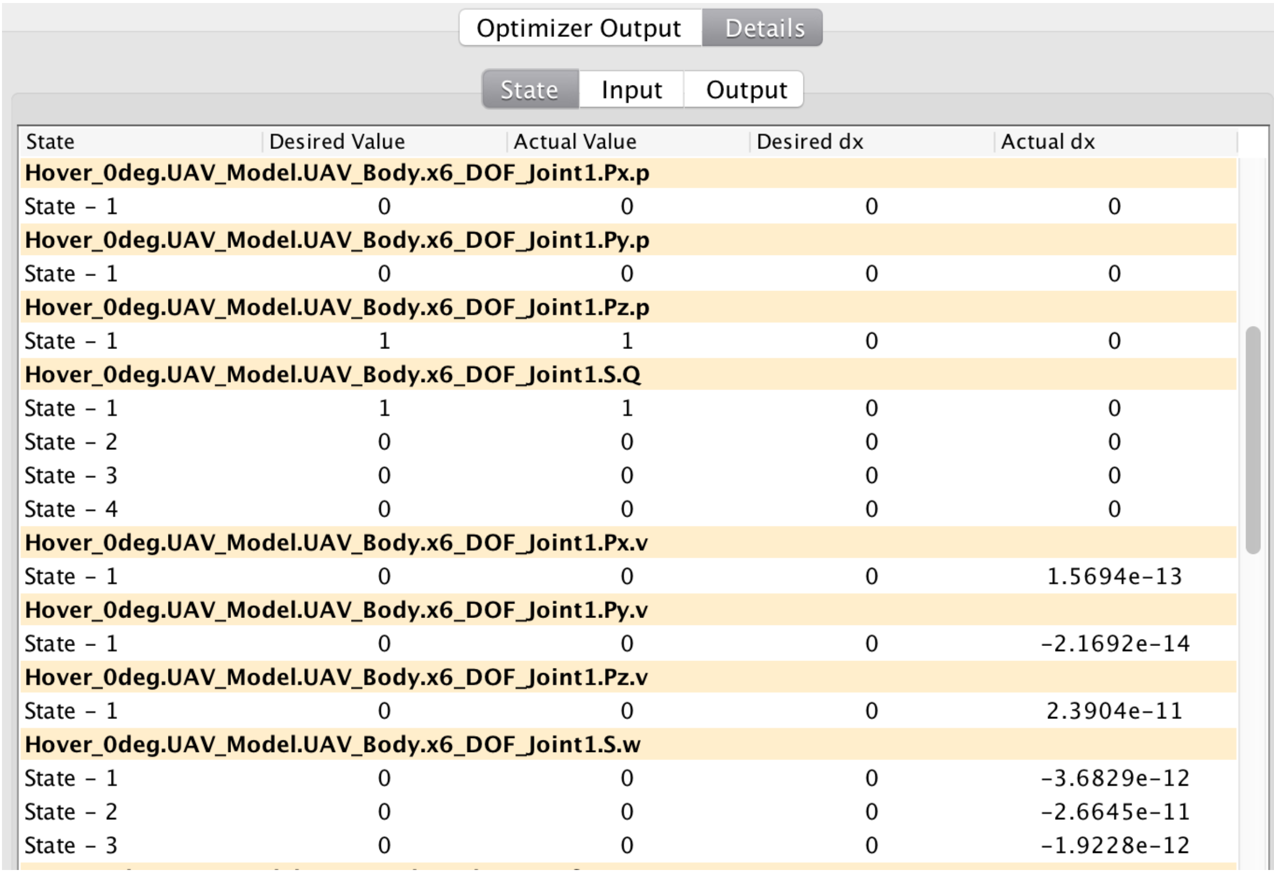
I got the results of linearizaion at the operating point 0deg. The results as are shown below in Fig. 5.5.



State	Desired Value	Actual Value	Desired dx	Actual dx
Hover_0deg.UAV_Model.Prop_1.Revolute_Joint_1.Rz.q				
State - 1	[-Inf , Inf]	0	N/A	144.8131
Hover_0deg.UAV_Model.Prop_1.Revolute_Joint_1.Rz.w				
State - 1	[15 , Inf]	144.8131	0	5.5212e-13
Hover_0deg.UAV_Model.Prop_2.Revolute_Joint_2.Rz.q				
State - 1	[-Inf , Inf]	2.4599e-27	N/A	-144.8131
Hover_0deg.UAV_Model.Prop_2.Revolute_Joint_2.Rz.w				
State - 1	[-Inf , Inf]	-144.8131	0	5.459e-12
Hover_0deg.UAV_Model.Prop_3.Revolute_Joint_3.Rz.q				
State - 1	[-Inf , Inf]	0	N/A	144.8131
Hover_0deg.UAV_Model.Prop_3.Revolute_Joint_3.Rz.w				
State - 1	[15 , Inf]	144.8131	0	5.9324e-13
Hover_0deg.UAV_Model.Prop_4.Revolute_Joint_4.Rz.q				
State - 1	[-Inf , Inf]	0	N/A	-144.8131
Hover_0deg.UAV_Model.Prop_4.Revolute_Joint_4.Rz.w				
State - 1	[-Inf , Inf]	-144.8131	0	5.459e-12

Figure 5.5: Linearization result for 0deg 1

As we can see in the above figure, the magnitude of angular velocities for 4 rotors are same, this means the UAV is hovering properly. Let me explain and verify this hovering motion using simulink and simmechanics later part.



Optimizer Output | Details

State | Input | Output

State	Desired Value	Actual Value	Desired dx	Actual dx
Hover_0deg.UAV_Model.UAV_Body.x6_DOF_Joint1.Px.p				
State - 1	0	0	0	0
Hover_0deg.UAV_Model.UAV_Body.x6_DOF_Joint1.Py.p				
State - 1	0	0	0	0
Hover_0deg.UAV_Model.UAV_Body.x6_DOF_Joint1.Pz.p				
State - 1	1	1	0	0
Hover_0deg.UAV_Model.UAV_Body.x6_DOF_Joint1.S.Q				
State - 1	1	1	0	0
State - 2	0	0	0	0
State - 3	0	0	0	0
State - 4	0	0	0	0
Hover_0deg.UAV_Model.UAV_Body.x6_DOF_Joint1.Px.v				
State - 1	0	0	0	1.5694e-13
Hover_0deg.UAV_Model.UAV_Body.x6_DOF_Joint1.Py.v				
State - 1	0	0	0	-2.1692e-14
Hover_0deg.UAV_Model.UAV_Body.x6_DOF_Joint1.Pz.v				
State - 1	0	0	0	2.3904e-11
Hover_0deg.UAV_Model.UAV_Body.x6_DOF_Joint1.S.w				
State - 1	0	0	0	-3.6829e-12
State - 2	0	0	0	-2.6645e-11
State - 3	0	0	0	-1.9228e-12

Figure 5.6: Linearization result for 0deg 2

As we can see in the above figure, the 6-DOF motion components match perfectly with what I set in the linearization conditions. After I got this linearization results, I made regulator for hovering at the operating point. Using full-state feedback, all states go to desired states. The plant can be written in state-space form $\dot{x} = Ax + Bu$, and that all of the n states x are available for the controller.

As we can see in the below figure, the error increases initially in order to get stabilize (The LQR regulator is meant to keep all the states near zero).

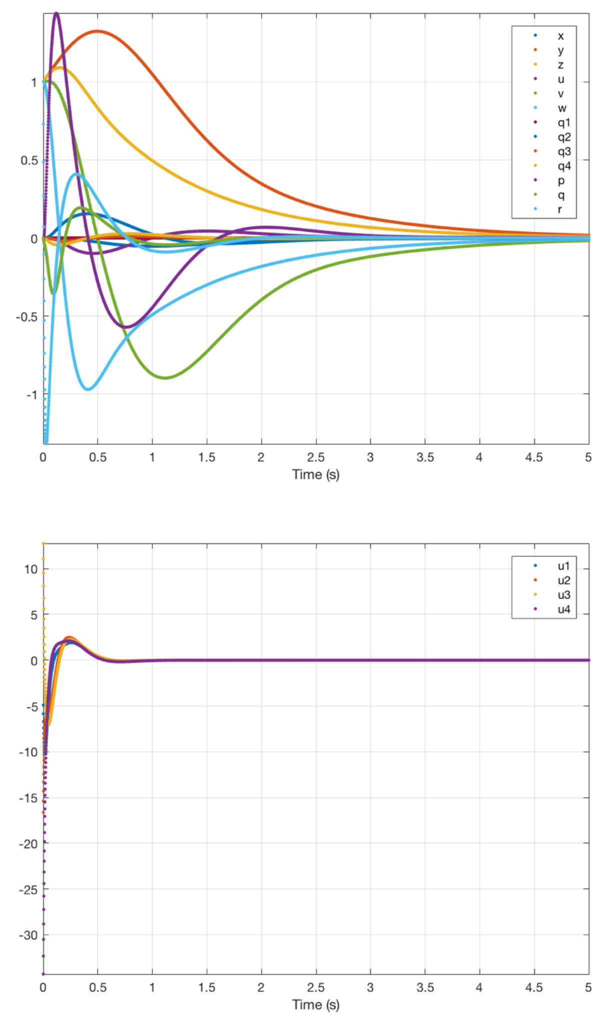
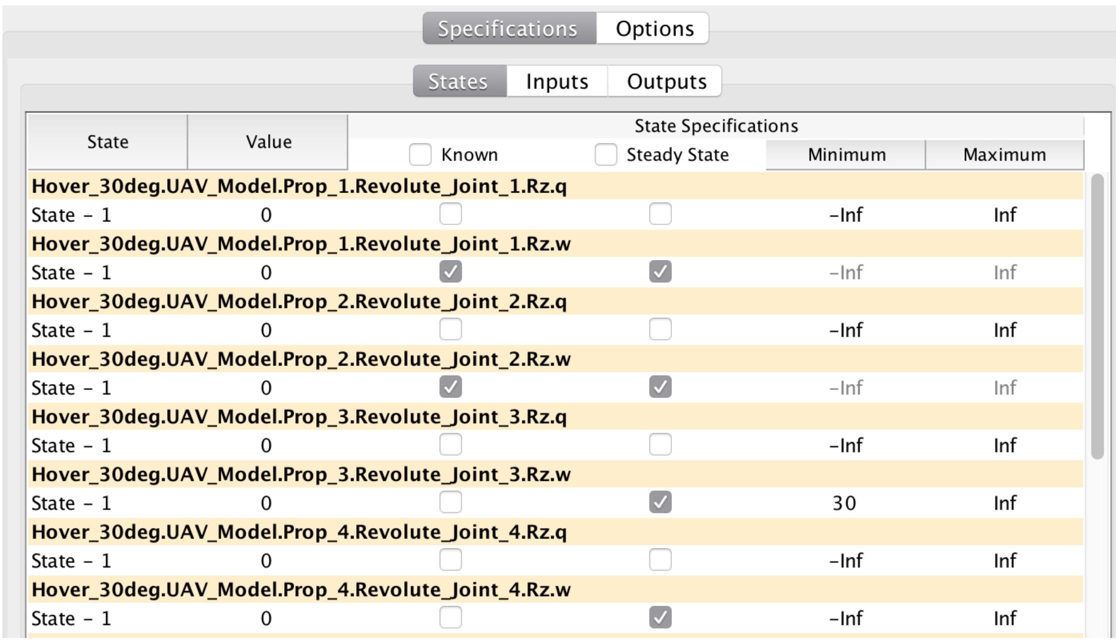


Figure 5.7: Regulator results for 0deg case

5.2 Incline hovering case (30deg rotation of rotor mount)

Second, I did the linearizaion at the operating point 30deg. The linearization conditions as are shown below in Fig. 5.8.

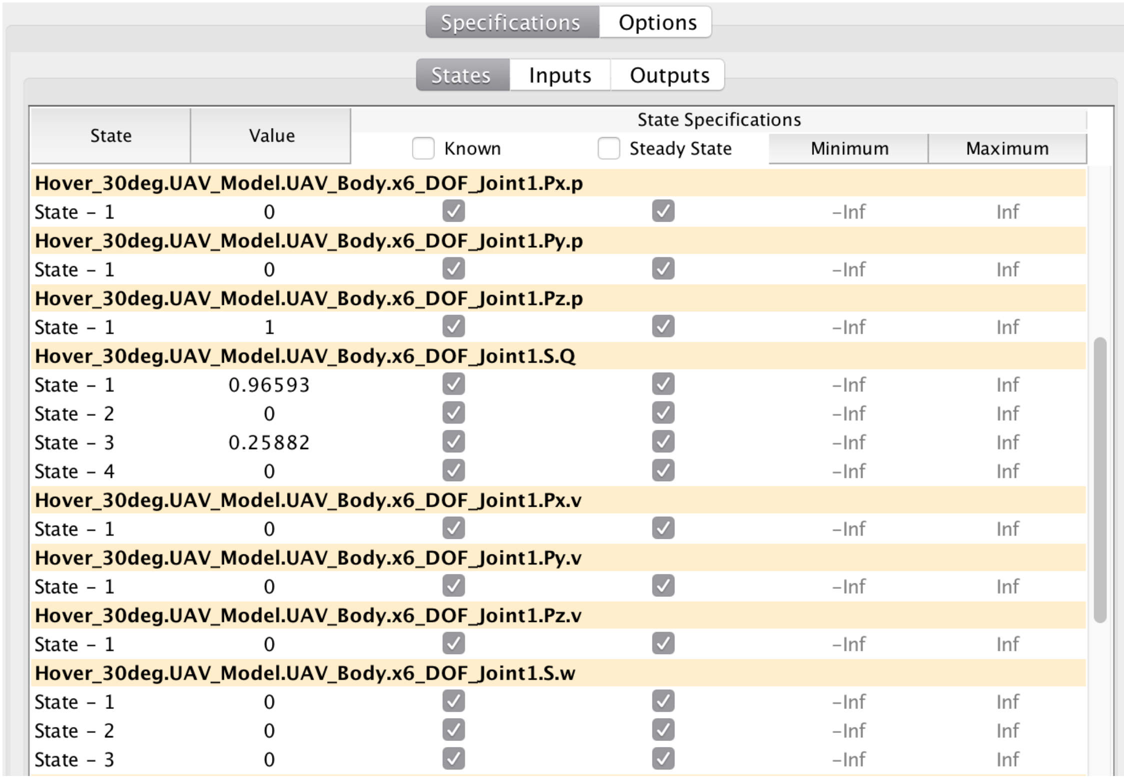

Specifications | Options

States | Inputs | Outputs

State	Value	Known	Steady State	Minimum	Maximum
Hover_30deg.UAV_Model.Prop_1.Revolute_Joint_1.Rz.q					
State - 1	0	☐	☐	-Inf	Inf
Hover_30deg.UAV_Model.Prop_1.Revolute_Joint_1.Rz.w					
State - 1	0	☑	☑	-Inf	Inf
Hover_30deg.UAV_Model.Prop_2.Revolute_Joint_2.Rz.q					
State - 1	0	☐	☐	-Inf	Inf
Hover_30deg.UAV_Model.Prop_2.Revolute_Joint_2.Rz.w					
State - 1	0	☑	☑	-Inf	Inf
Hover_30deg.UAV_Model.Prop_3.Revolute_Joint_3.Rz.q					
State - 1	0	☐	☐	-Inf	Inf
Hover_30deg.UAV_Model.Prop_3.Revolute_Joint_3.Rz.w					
State - 1	0	☐	☑	30	Inf
Hover_30deg.UAV_Model.Prop_4.Revolute_Joint_4.Rz.q					
State - 1	0	☐	☐	-Inf	Inf
Hover_30deg.UAV_Model.Prop_4.Revolute_Joint_4.Rz.w					
State - 1	0	☐	☑	-Inf	Inf

Figure 5.8: Linearization condition for 30deg 1

Each propeller rotors have rotation angle and angular velocity. “Rz.q" means rotation angle, “Rz.w" means angular velocity. I set initial minimum value for “rotor3" as “30" for initial linearizing condition.



Specifications | Options

States | Inputs | Outputs

State	Value	Known	Steady State	Minimum	Maximum
Hover_30deg.UAV_Model.UAV_Body.x6_DOF_Joint1.Px.p					
State - 1	0	☑	☑	-Inf	Inf
Hover_30deg.UAV_Model.UAV_Body.x6_DOF_Joint1.Py.p					
State - 1	0	☑	☑	-Inf	Inf
Hover_30deg.UAV_Model.UAV_Body.x6_DOF_Joint1.Pz.p					
State - 1	1	☑	☑	-Inf	Inf
Hover_30deg.UAV_Model.UAV_Body.x6_DOF_Joint1.S.Q					
State - 1	0.96593	☑	☑	-Inf	Inf
State - 2	0	☑	☑	-Inf	Inf
State - 3	0.25882	☑	☑	-Inf	Inf
State - 4	0	☑	☑	-Inf	Inf
Hover_30deg.UAV_Model.UAV_Body.x6_DOF_Joint1.Px.v					
State - 1	0	☑	☑	-Inf	Inf
Hover_30deg.UAV_Model.UAV_Body.x6_DOF_Joint1.Py.v					
State - 1	0	☑	☑	-Inf	Inf
Hover_30deg.UAV_Model.UAV_Body.x6_DOF_Joint1.Pz.v					
State - 1	0	☑	☑	-Inf	Inf
Hover_30deg.UAV_Model.UAV_Body.x6_DOF_Joint1.S.w					
State - 1	0	☑	☑	-Inf	Inf
State - 2	0	☑	☑	-Inf	Inf
State - 3	0	☑	☑	-Inf	Inf

Figure 5.9: Linearization condition for 30deg 2

And then, the UAV has 6-DOF motions. For hovering, I set some values. “Px.p, Py.p, Pz.p" means “x, y, z". I set this value as “0, 0, 1". “S.Q" means quaternion of UAV. I set this value “0.96593, 0, 0.25882, 0" because “30deg" means “ 30deg rotation" at that time, the quaternion value is “0.96593, 0, 0.25882, 0". “Px.v, Py.v, Pz.v" means “linear velocity of UAV (u, v, w)". I set this value “0, 0, 0" because of hovering. “S.w" means angular velocity of UAV (p, q, r). I set this value “0, 0, 0" for hovering motion.

I got the results of linearizaion at the operating point 30deg. The results as are shown below in Fig. 5.10.

Optimizer Output | Details

State | Input | Output

State	Desired Value	Actual Value	Desired dx	Actual dx
Hover_30deg.UAV_Model.Prop_1.Revolute_Joint_1.Rz.q				
State - 1	[-Inf , Inf]	-2.9936e-06	N/A	0
Hover_30deg.UAV_Model.Prop_1.Revolute_Joint_1.Rz.w				
State - 1	0	0	0	-2.4649e-09
Hover_30deg.UAV_Model.Prop_2.Revolute_Joint_2.Rz.q				
State - 1	[-Inf , Inf]	2.7937e-06	N/A	0
Hover_30deg.UAV_Model.Prop_2.Revolute_Joint_2.Rz.w				
State - 1	0	0	0	2.4649e-09
Hover_30deg.UAV_Model.Prop_3.Revolute_Joint_3.Rz.q				
State - 1	[-Inf , Inf]	0.00016314	N/A	204.7968
Hover_30deg.UAV_Model.Prop_3.Revolute_Joint_3.Rz.w				
State - 1	[30 , Inf]	204.7968	0	-4.1272e-09
Hover_30deg.UAV_Model.Prop_4.Revolute_Joint_4.Rz.q				
State - 1	[-Inf , Inf]	1.657e-06	N/A	-204.7968
Hover_30deg.UAV_Model.Prop_4.Revolute_Joint_4.Rz.w				
State - 1	[-Inf , Inf]	-204.7968	0	-2.1591e-09

Figure 5.10: Linearization result for 30deg 1

As we can see in the above figure, the magnitude of angular velocities for 2 rotors are same, this means the UAV is hovering properly. Let me explain and verify this hovering motion using simulink and simmechanics later part.

Optimizer Output | Details

State | Input | Output

State	Desired Value	Actual Value	Desired dx	Actual dx
Hover_30deg.UAV_Model.UAV_Body.x6_DOF_Joint1.Px.p				
State - 1	0	0	0	0
Hover_30deg.UAV_Model.UAV_Body.x6_DOF_Joint1.Py.p				
State - 1	0	0	0	0
Hover_30deg.UAV_Model.UAV_Body.x6_DOF_Joint1.Pz.p				
State - 1	1	1	0	0
Hover_30deg.UAV_Model.UAV_Body.x6_DOF_Joint1.S.Q				
State - 1	0.96593	0.96593	0	0
State - 2	0	0	0	0
State - 3	0.25882	0.25882	0	0
State - 4	0	0	0	0
Hover_30deg.UAV_Model.UAV_Body.x6_DOF_Joint1.Px.v				
State - 1	0	0	0	4.222e-05
Hover_30deg.UAV_Model.UAV_Body.x6_DOF_Joint1.Py.v				
State - 1	0	0	0	-3.076e-11
Hover_30deg.UAV_Model.UAV_Body.x6_DOF_Joint1.Pz.v				
State - 1	0	0	0	1.1254e-08
Hover_30deg.UAV_Model.UAV_Body.x6_DOF_Joint1.S.w				
State - 1	0	0	0	-1.021e-08
State - 2	0	0	0	-2.1014e-15
State - 3	0	0	0	-3.205e-09

Figure 5.11: Linearization result for 30deg 2

As we can see in the above figure, the 6-DOF motion components match perfectly with what I set in the linearization conditions. After I got this linearization results, I made regulator for hovering at the operating point. Using full-state feedback, all states go to desired states. The plant can be written in state-space form $\dot{x} = Ax + Bu$, and that all of the n states x are available for the controller.

As we can see in the below figure, the error increases initially in order to get stabilize (The LQR regulator is meant to keep all the states near zero).

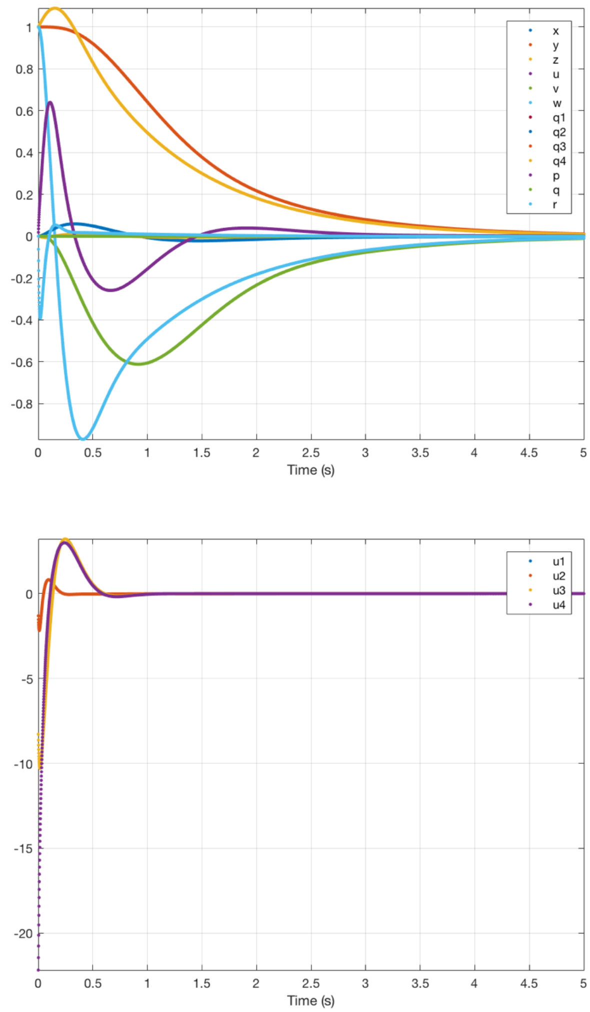


Figure 5.12: Regulator results for 30deg case

5.3 Incline hovering case (45deg rotation of rotor mount)

Third, I did the linearizaion at the operating point 45deg. The linearization conditions as are shown below in Fig. 5.13.

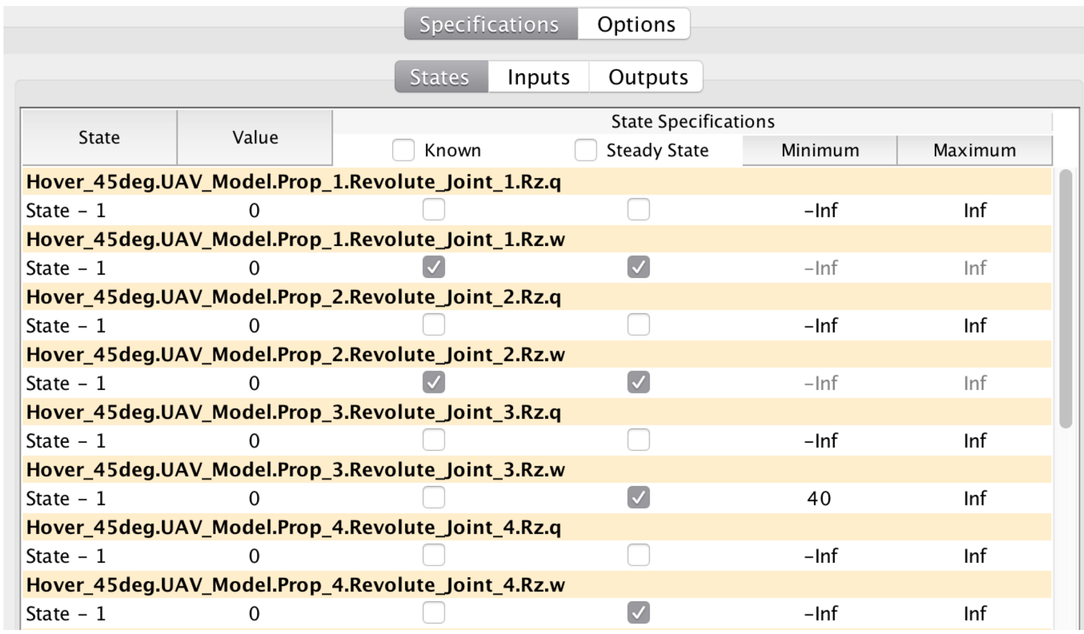


Figure 5.13: Linearization condition for 45deg 1

Each propeller rotors have rotation angle and angular velocity. “Rz.q" means rotation angle, “Rz.w" means angular velocity. I set initial minimum value for “rotor3" as “40" for initial linearizing condition.

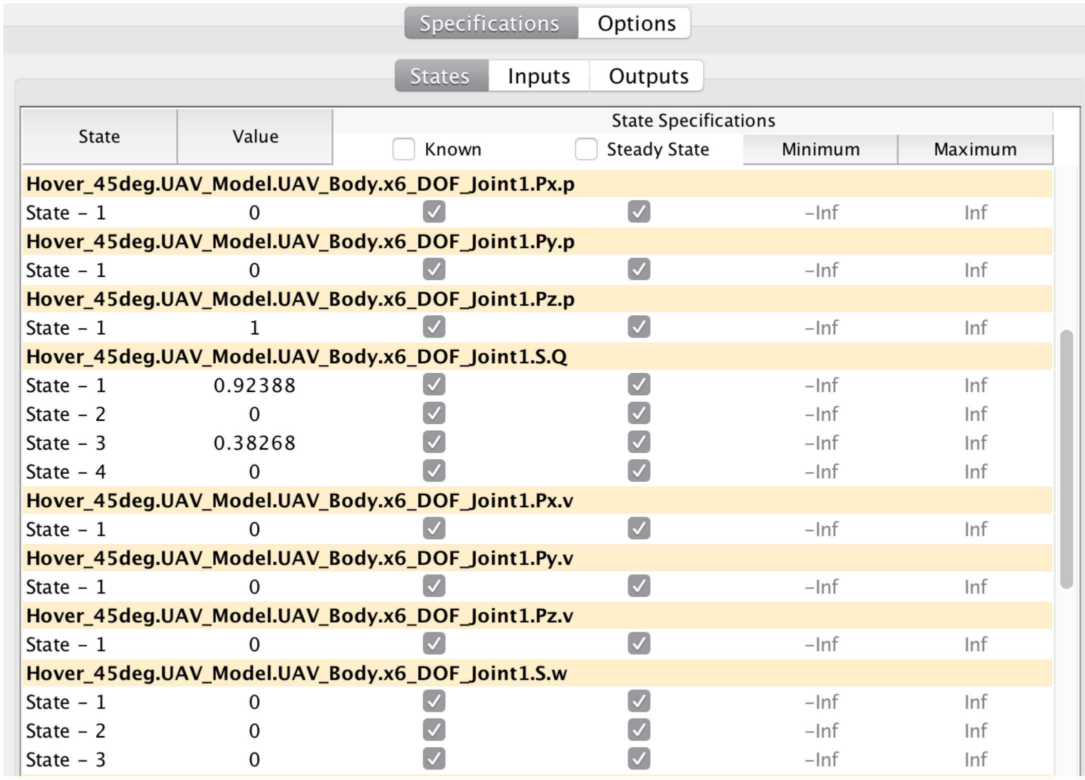
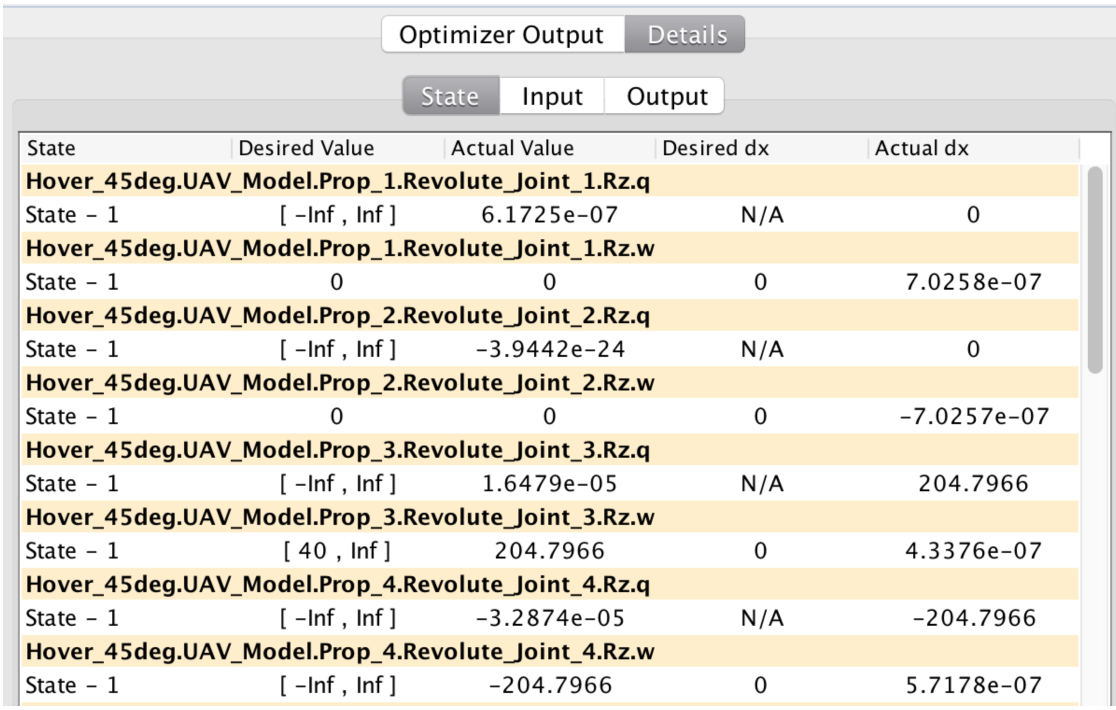


Figure 5.14: Linearization condition for 45deg 2

And then, the UAV has 6-DOF motions. For hovering, I set some values. “Px.p, Py.p, Pz.p" means “x, y, z". I set this value as “0, 0, 1". “S.Q" means quaternion of UAV. I set this value “0.92388, 0, 0.38268, 0" because “45deg" means “ 45deg rotation" at that time, the quaternion value is “0.92388, 0, 0.38268, 0". “Px.v, Py.v, Pz.v" means “linear velocity of UAV (u, v, w)". I set this value “0, 0, 0" because of hovering. “S.w" means angular velocity of UAV (p, q, r). I set this value “0, 0, 0" for hovering motion.

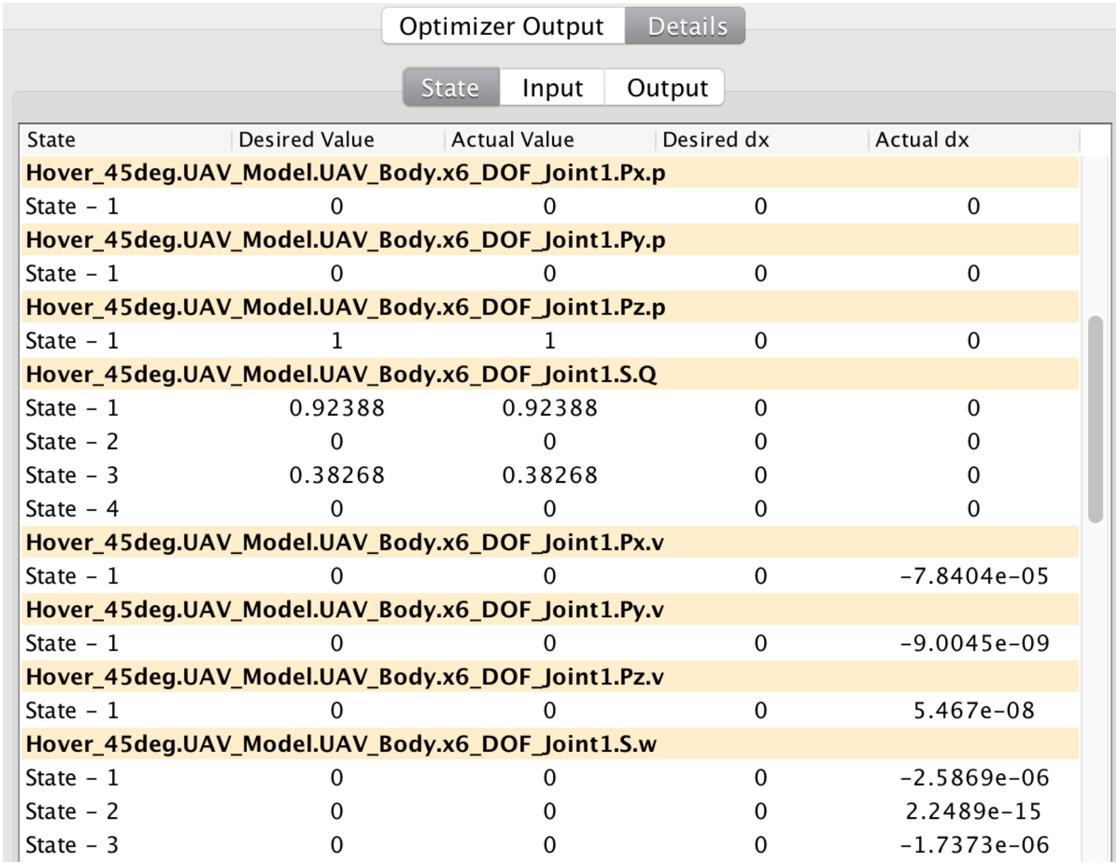
I got the results of linearizaion at the operating point 45deg. The results as are shown below in Fig. 5.15.



State	Desired Value	Actual Value	Desired dx	Actual dx
Hover_45deg.UAV_Model.Prop_1.Revolute_Joint_1.Rz.q				
State - 1	[-Inf , Inf]	6.1725e-07	N/A	0
Hover_45deg.UAV_Model.Prop_1.Revolute_Joint_1.Rz.w				
State - 1	0	0	0	7.0258e-07
Hover_45deg.UAV_Model.Prop_2.Revolute_Joint_2.Rz.q				
State - 1	[-Inf , Inf]	-3.9442e-24	N/A	0
Hover_45deg.UAV_Model.Prop_2.Revolute_Joint_2.Rz.w				
State - 1	0	0	0	-7.0257e-07
Hover_45deg.UAV_Model.Prop_3.Revolute_Joint_3.Rz.q				
State - 1	[-Inf , Inf]	1.6479e-05	N/A	204.7966
Hover_45deg.UAV_Model.Prop_3.Revolute_Joint_3.Rz.w				
State - 1	[40 , Inf]	204.7966	0	4.3376e-07
Hover_45deg.UAV_Model.Prop_4.Revolute_Joint_4.Rz.q				
State - 1	[-Inf , Inf]	-3.2874e-05	N/A	-204.7966
Hover_45deg.UAV_Model.Prop_4.Revolute_Joint_4.Rz.w				
State - 1	[-Inf , Inf]	-204.7966	0	5.7178e-07

Figure 5.15: Linearization result for 45deg 1

As we can see in the above figure, the magnitude of angular velocities for 2 rotors are same, this means the UAV is hovering properly. Let me explain and verify this hovering motion using simulink and simmechanics later part.



State	Desired Value	Actual Value	Desired dx	Actual dx
Hover_45deg.UAV_Model.UAV_Body.x6_DOF_Joint1.Px.p				
State - 1	0	0	0	0
Hover_45deg.UAV_Model.UAV_Body.x6_DOF_Joint1.Py.p				
State - 1	0	0	0	0
Hover_45deg.UAV_Model.UAV_Body.x6_DOF_Joint1.Pz.p				
State - 1	1	1	0	0
Hover_45deg.UAV_Model.UAV_Body.x6_DOF_Joint1.S.Q				
State - 1	0.92388	0.92388	0	0
State - 2	0	0	0	0
State - 3	0.38268	0.38268	0	0
State - 4	0	0	0	0
Hover_45deg.UAV_Model.UAV_Body.x6_DOF_Joint1.Px.v				
State - 1	0	0	0	-7.8404e-05
Hover_45deg.UAV_Model.UAV_Body.x6_DOF_Joint1.Py.v				
State - 1	0	0	0	-9.0045e-09
Hover_45deg.UAV_Model.UAV_Body.x6_DOF_Joint1.Pz.v				
State - 1	0	0	0	5.467e-08
Hover_45deg.UAV_Model.UAV_Body.x6_DOF_Joint1.S.w				
State - 1	0	0	0	-2.5869e-06
State - 2	0	0	0	2.2489e-15
State - 3	0	0	0	-1.7373e-06

Figure 5.16: Linearization result for 45deg 2

As we can see in the above figure, the 6-DOF motion components match perfectly with what I set in the linearization conditions. After I got this linearization results, I made regulator for hovering at the operating point. Using full-state feedback, all states go to desired states. The plant can be written in state-space form $\dot{x} = Ax + Bu$, and that all of the n states x are available for the controller.

As we can see in the below figure, the error increases initially in order to get stabilize (The LQR regulator is meant to keep all the states near zero).

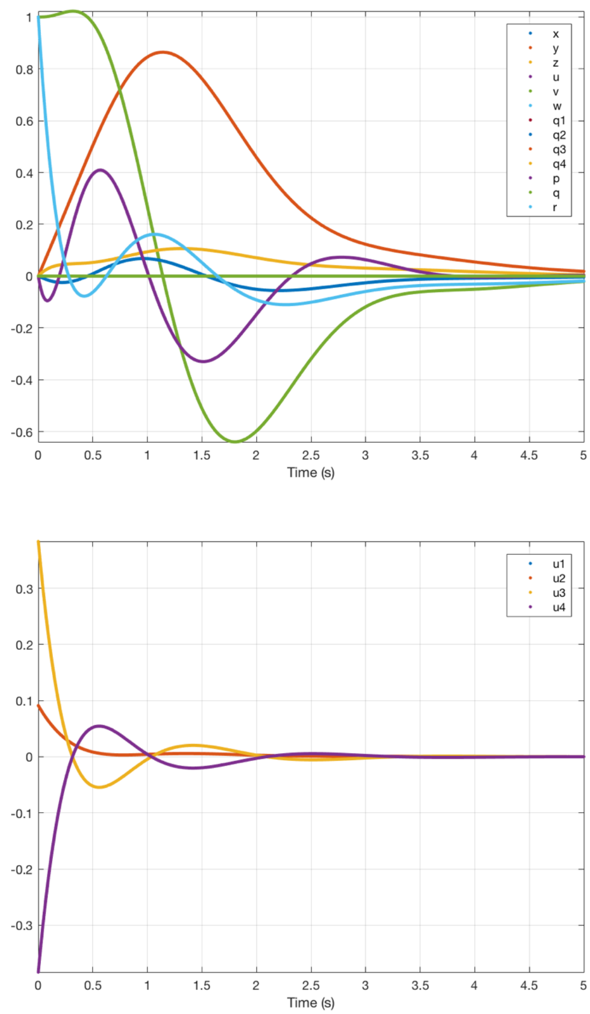


Figure 5.17: Regulator results for 45deg case

5.4 Incline hovering case (60deg rotation of rotor mount)

Third, I did the linearizaion at the operating point 60deg. The linearization conditions as are shown below in Fig. 5.18.

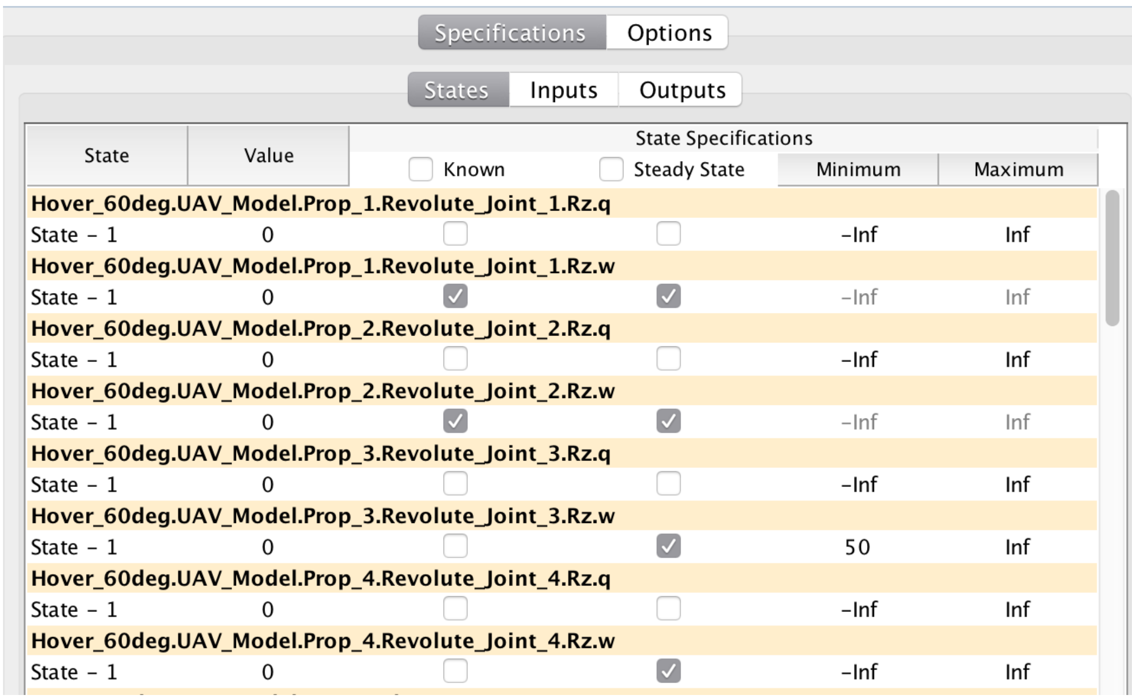


Figure 5.18: Linearization condition for 60deg 1

Each propeller rotors have rotation angle and angular velocity. “Rz.q" means rotation angle, “Rz.w" means angular velocity. I set initial minimum value for “rotor3" as “50" for initial linearizing condition.

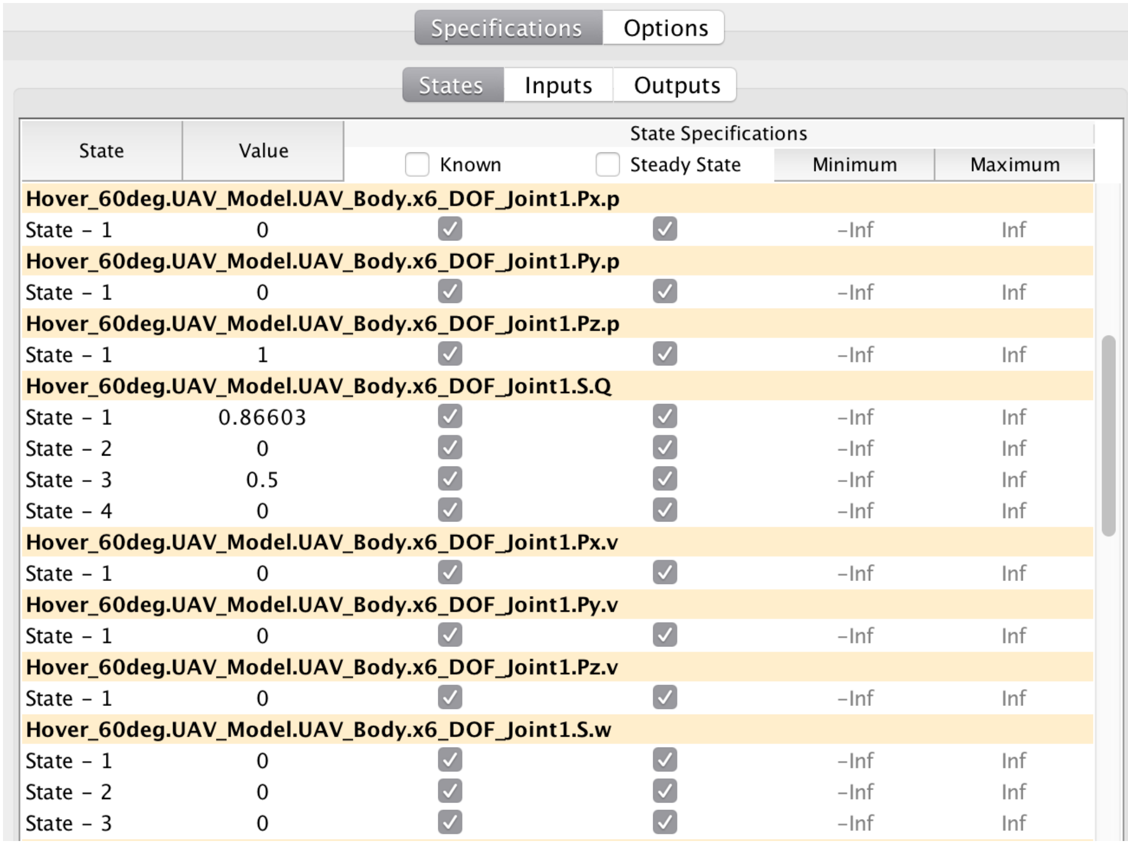
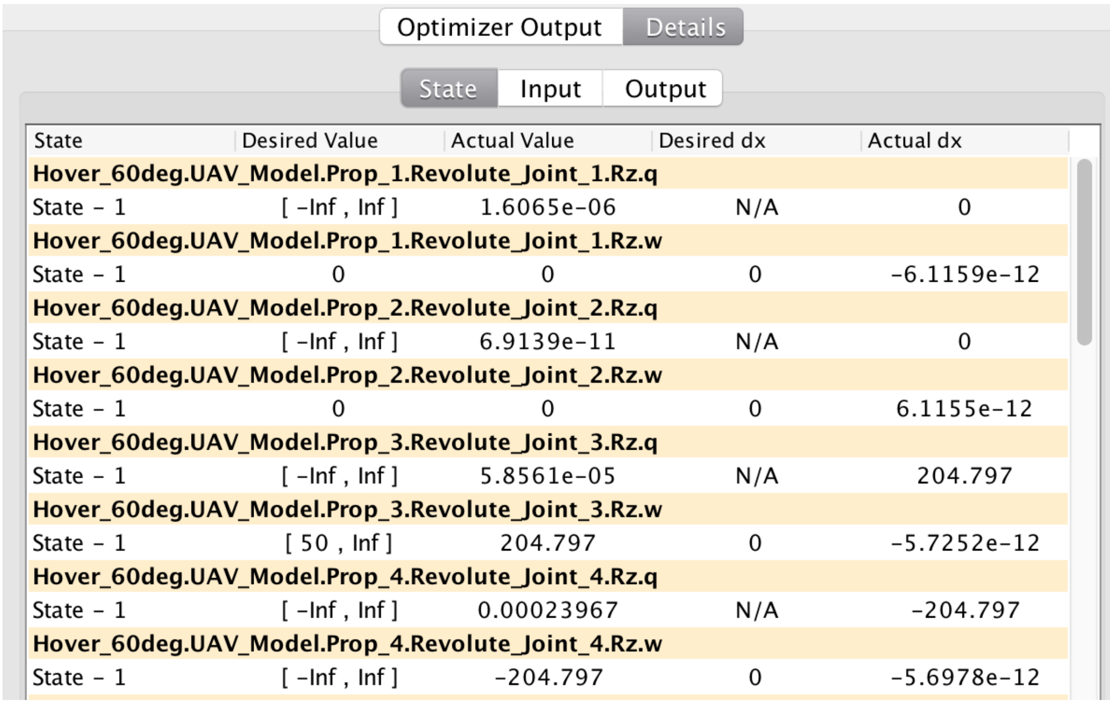


Figure 5.19: Linearization condition for 60deg 2

And then, the UAV has 6-DOF motions. For hovering, I set some values. “Px.p, Py.p, Pz.p" means “x, y, z". I set this value as “0, 0, 1". “S.Q" means quaternion of UAV. I set this value “0.86603, 0, 0.5, 0" because “60deg" means “ 60deg rotation" at that time, the quaternion value is “0.86603, 0, 0.5, 0". “Px.v, Py.v, Pz.v" means “linear velocity of UAV (u, v, w)". I set this value “0, 0, 0" because of hovering. “S.w" means angular velocity of UAV (p, q, r). I set this value “0, 0, 0" for hovering motion.

I got the results of linearizaion at the operating point 60deg. The results as are shown below in Fig. 5.20.



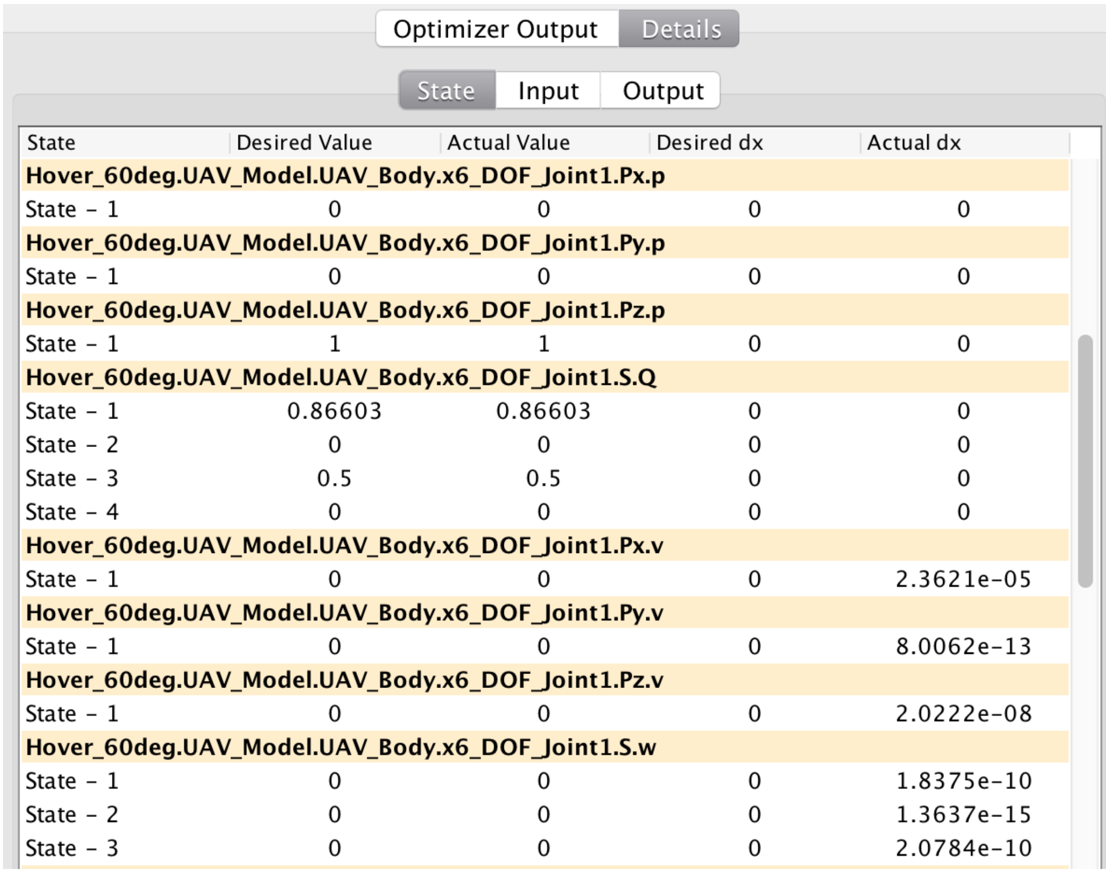
Optimizer Output | Details

State | Input | Output

State	Desired Value	Actual Value	Desired dx	Actual dx
Hover_60deg.UAV_Model.Prop_1.Revolute_Joint_1.Rz.q				
State - 1	[-Inf , Inf]	1.6065e-06	N/A	0
Hover_60deg.UAV_Model.Prop_1.Revolute_Joint_1.Rz.w				
State - 1	0	0	0	-6.1159e-12
Hover_60deg.UAV_Model.Prop_2.Revolute_Joint_2.Rz.q				
State - 1	[-Inf , Inf]	6.9139e-11	N/A	0
Hover_60deg.UAV_Model.Prop_2.Revolute_Joint_2.Rz.w				
State - 1	0	0	0	6.1155e-12
Hover_60deg.UAV_Model.Prop_3.Revolute_Joint_3.Rz.q				
State - 1	[-Inf , Inf]	5.8561e-05	N/A	204.797
Hover_60deg.UAV_Model.Prop_3.Revolute_Joint_3.Rz.w				
State - 1	[50 , Inf]	204.797	0	-5.7252e-12
Hover_60deg.UAV_Model.Prop_4.Revolute_Joint_4.Rz.q				
State - 1	[-Inf , Inf]	0.00023967	N/A	-204.797
Hover_60deg.UAV_Model.Prop_4.Revolute_Joint_4.Rz.w				
State - 1	[-Inf , Inf]	-204.797	0	-5.6978e-12

Figure 5.20: Linearization result for 60deg 1

As we can see in the above figure, the magnitude of angular velocities for 2 rotors are same, this means the UAV is hovering properly. Let me explain and verify this hovering motion using simulink and simmechanics later part.



State	Desired Value	Actual Value	Desired dx	Actual dx
Hover_60deg.UAV_Model.UAV_Body.x6_DOF_Joint1.Px.p				
State - 1	0	0	0	0
Hover_60deg.UAV_Model.UAV_Body.x6_DOF_Joint1.Py.p				
State - 1	0	0	0	0
Hover_60deg.UAV_Model.UAV_Body.x6_DOF_Joint1.Pz.p				
State - 1	1	1	0	0
Hover_60deg.UAV_Model.UAV_Body.x6_DOF_Joint1.S.Q				
State - 1	0.86603	0.86603	0	0
State - 2	0	0	0	0
State - 3	0.5	0.5	0	0
State - 4	0	0	0	0
Hover_60deg.UAV_Model.UAV_Body.x6_DOF_Joint1.Px.v				
State - 1	0	0	0	2.3621e-05
Hover_60deg.UAV_Model.UAV_Body.x6_DOF_Joint1.Py.v				
State - 1	0	0	0	8.0062e-13
Hover_60deg.UAV_Model.UAV_Body.x6_DOF_Joint1.Pz.v				
State - 1	0	0	0	2.0222e-08
Hover_60deg.UAV_Model.UAV_Body.x6_DOF_Joint1.S.w				
State - 1	0	0	0	1.8375e-10
State - 2	0	0	0	1.3637e-15
State - 3	0	0	0	2.0784e-10

Figure 5.21: Linearization result for 60deg 2

As we can see in the above figure, the 6-DOF motion components match perfectly with what I set in the linearization conditions. After I got this linearization results, I made regulator for hovering at the operating point. Using full-state feedback, all states go to desired states. The plant can be written in state-space form $\dot{x} = Ax + Bu$, and that all of the n states x are available for the controller.

As we can see in the below figure, the error increases initially in order to get stabilize (The LQR regulator is meant to keep all the states near zero).

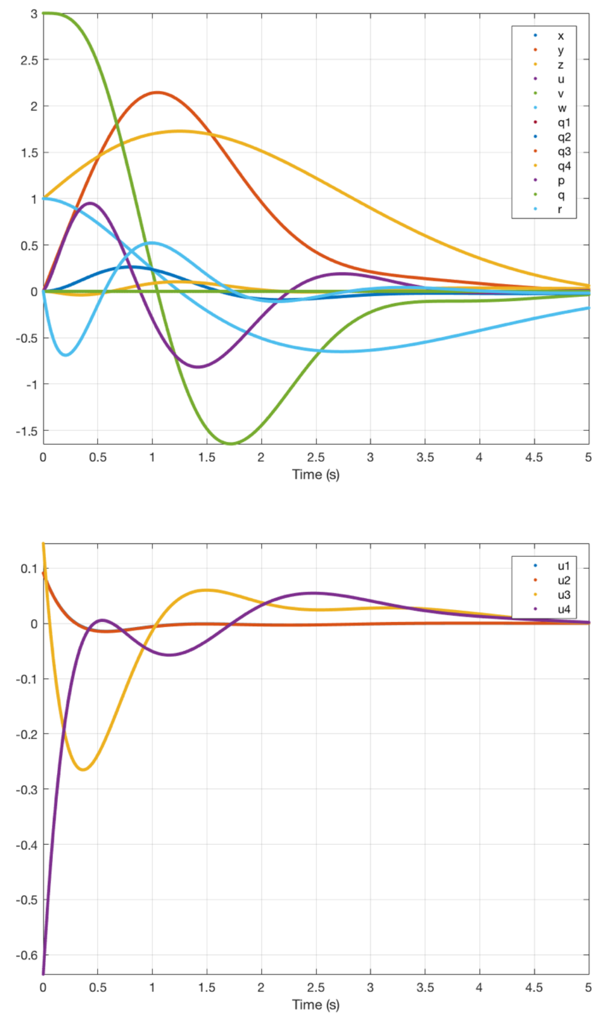
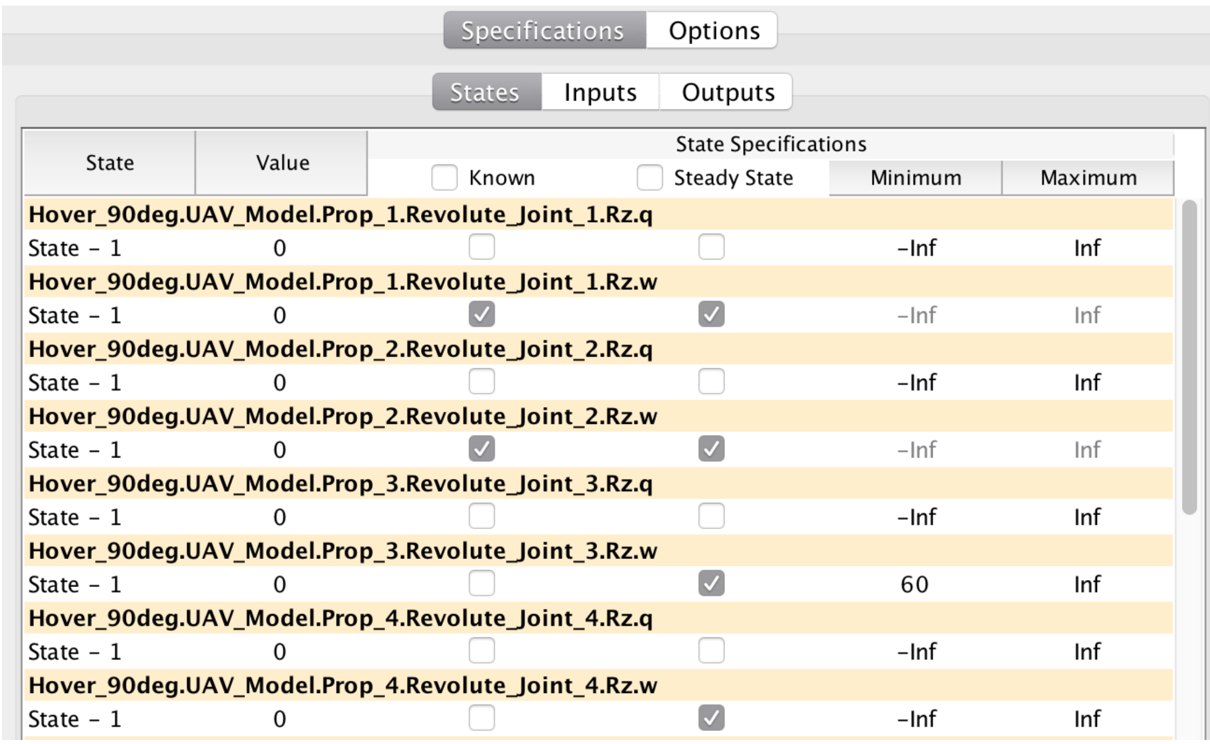


Figure 5.22: Regulator results for 60deg case

5.5 Vertical hovering case (90deg rotation of rotor mount)

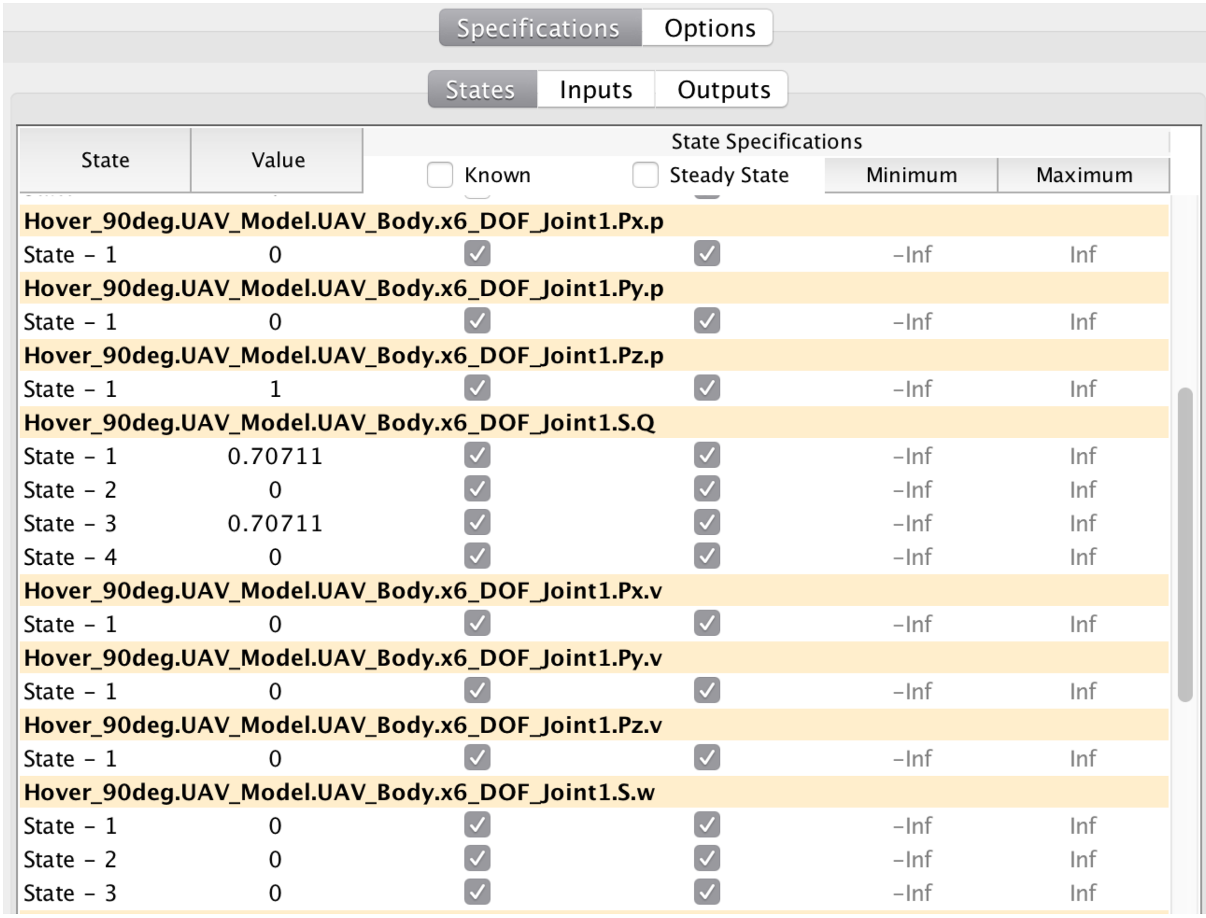
Third, I did the linearizaion at the operating point 90deg. The linearization conditions as are shown below in Fig. 5.23.



State	Value	Known	Steady State	Minimum	Maximum
Hover_90deg.UAV_Model.Prop_1.Revolute_Joint_1.Rz.q					
State - 1	0	☐	☐	-Inf	Inf
Hover_90deg.UAV_Model.Prop_1.Revolute_Joint_1.Rz.w					
State - 1	0	☑	☑	-Inf	Inf
Hover_90deg.UAV_Model.Prop_2.Revolute_Joint_2.Rz.q					
State - 1	0	☐	☐	-Inf	Inf
Hover_90deg.UAV_Model.Prop_2.Revolute_Joint_2.Rz.w					
State - 1	0	☑	☑	-Inf	Inf
Hover_90deg.UAV_Model.Prop_3.Revolute_Joint_3.Rz.q					
State - 1	0	☐	☐	-Inf	Inf
Hover_90deg.UAV_Model.Prop_3.Revolute_Joint_3.Rz.w					
State - 1	0	☐	☑	60	Inf
Hover_90deg.UAV_Model.Prop_4.Revolute_Joint_4.Rz.q					
State - 1	0	☐	☐	-Inf	Inf
Hover_90deg.UAV_Model.Prop_4.Revolute_Joint_4.Rz.w					
State - 1	0	☐	☑	-Inf	Inf

Figure 5.23: Linearization condition for 90deg 1

Each propeller rotors have rotation angle and angular velocity. “Rz.q" means rotation angle, “Rz.w" means angular velocity. I set initial minimum value for “rotor3" as “60" for initial linearizing condition.



Specifications | Options

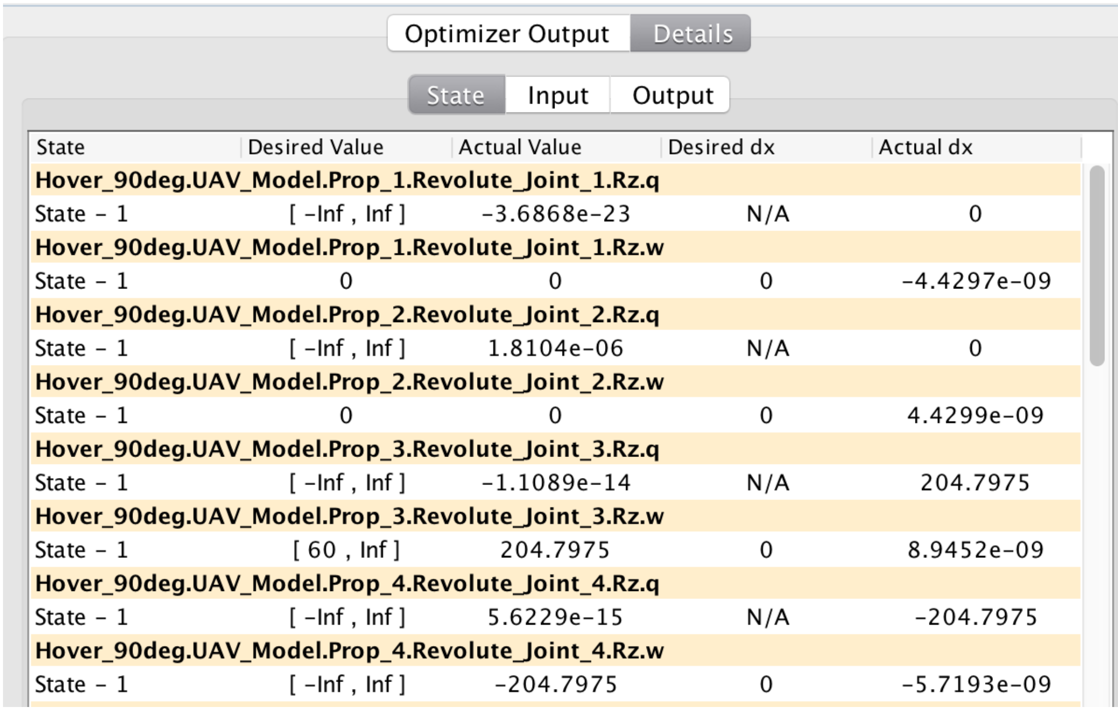
States | Inputs | Outputs

State	Value	Known	Steady State	Minimum	Maximum
Hover_90deg.UAV_Model.UAV_Body.x6_DOF_Joint1.Px.p					
State - 1	0	✓	✓	-Inf	Inf
Hover_90deg.UAV_Model.UAV_Body.x6_DOF_Joint1.Py.p					
State - 1	0	✓	✓	-Inf	Inf
Hover_90deg.UAV_Model.UAV_Body.x6_DOF_Joint1.Pz.p					
State - 1	1	✓	✓	-Inf	Inf
Hover_90deg.UAV_Model.UAV_Body.x6_DOF_Joint1.S.Q					
State - 1	0.70711	✓	✓	-Inf	Inf
State - 2	0	✓	✓	-Inf	Inf
State - 3	0.70711	✓	✓	-Inf	Inf
State - 4	0	✓	✓	-Inf	Inf
Hover_90deg.UAV_Model.UAV_Body.x6_DOF_Joint1.Px.v					
State - 1	0	✓	✓	-Inf	Inf
Hover_90deg.UAV_Model.UAV_Body.x6_DOF_Joint1.Py.v					
State - 1	0	✓	✓	-Inf	Inf
Hover_90deg.UAV_Model.UAV_Body.x6_DOF_Joint1.Pz.v					
State - 1	0	✓	✓	-Inf	Inf
Hover_90deg.UAV_Model.UAV_Body.x6_DOF_Joint1.S.w					
State - 1	0	✓	✓	-Inf	Inf
State - 2	0	✓	✓	-Inf	Inf
State - 3	0	✓	✓	-Inf	Inf

Figure 5.24: Linearization condition for 90deg 2

And then, the UAV has 6-DOF motions. For hovering, I set some values. “Px.p, Py.p, Pz.p" means “x, y, z". I set this value as “0, 0, 1". “S.Q" means quaternion of UAV. I set this value “0.70711, 0, 0.70711, 0" because “90deg" means “ 90deg rotation" at that time, the quaternion value is “0.70711, 0, 0.70711, 0". “Px.v, Py.v, Pz.v" means “linear velocity of UAV (u, v, w)". I set this value “0, 0, 0" because of hovering. “S.w" means angular velocity of UAV (p, q, r). I set this value “0, 0, 0" for hovering motion.

I got the results of linearizaion at the operating point 90deg. The results as are shown below in Fig. 5.25.



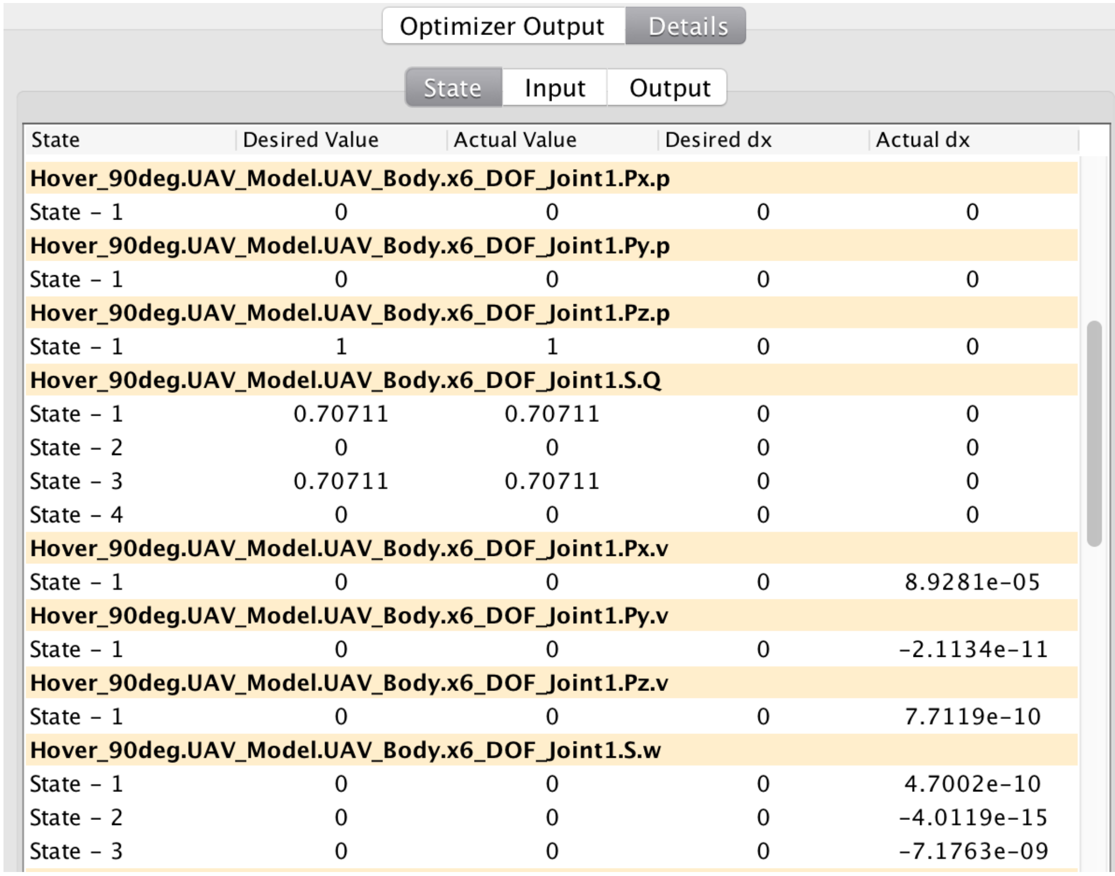
Optimizer Output | Details

State | Input | Output

State	Desired Value	Actual Value	Desired dx	Actual dx
Hover_90deg.UAV_Model.Prop_1.Revolute_Joint_1.Rz.q				
State - 1	[-Inf , Inf]	-3.6868e-23	N/A	0
Hover_90deg.UAV_Model.Prop_1.Revolute_Joint_1.Rz.w				
State - 1	0	0	0	-4.4297e-09
Hover_90deg.UAV_Model.Prop_2.Revolute_Joint_2.Rz.q				
State - 1	[-Inf , Inf]	1.8104e-06	N/A	0
Hover_90deg.UAV_Model.Prop_2.Revolute_Joint_2.Rz.w				
State - 1	0	0	0	4.4299e-09
Hover_90deg.UAV_Model.Prop_3.Revolute_Joint_3.Rz.q				
State - 1	[-Inf , Inf]	-1.1089e-14	N/A	204.7975
Hover_90deg.UAV_Model.Prop_3.Revolute_Joint_3.Rz.w				
State - 1	[60 , Inf]	204.7975	0	8.9452e-09
Hover_90deg.UAV_Model.Prop_4.Revolute_Joint_4.Rz.q				
State - 1	[-Inf , Inf]	5.6229e-15	N/A	-204.7975
Hover_90deg.UAV_Model.Prop_4.Revolute_Joint_4.Rz.w				
State - 1	[-Inf , Inf]	-204.7975	0	-5.7193e-09

Figure 5.25: Linearization result for 90deg 1

As we can see in the above figure, the magnitude of angular velocities for 2 rotors are same, this means the UAV is hovering properly. Let me explain and verify this hovering motion using simulink and simmechanics video later part.



Optimizer Output | Details

State | Input | Output

State	Desired Value	Actual Value	Desired dx	Actual dx
Hover_90deg.UAV_Model.UAV_Body.x6_DOF_Joint1.Px.p				
State - 1	0	0	0	0
Hover_90deg.UAV_Model.UAV_Body.x6_DOF_Joint1.Py.p				
State - 1	0	0	0	0
Hover_90deg.UAV_Model.UAV_Body.x6_DOF_Joint1.Pz.p				
State - 1	1	1	0	0
Hover_90deg.UAV_Model.UAV_Body.x6_DOF_Joint1.S.Q				
State - 1	0.70711	0.70711	0	0
State - 2	0	0	0	0
State - 3	0.70711	0.70711	0	0
State - 4	0	0	0	0
Hover_90deg.UAV_Model.UAV_Body.x6_DOF_Joint1.Px.v				
State - 1	0	0	0	8.9281e-05
Hover_90deg.UAV_Model.UAV_Body.x6_DOF_Joint1.Py.v				
State - 1	0	0	0	-2.1134e-11
Hover_90deg.UAV_Model.UAV_Body.x6_DOF_Joint1.Pz.v				
State - 1	0	0	0	7.7119e-10
Hover_90deg.UAV_Model.UAV_Body.x6_DOF_Joint1.S.w				
State - 1	0	0	0	4.7002e-10
State - 2	0	0	0	-4.0119e-15
State - 3	0	0	0	-7.1763e-09

Figure 5.26: Linearization result for 90deg 2

As we can see in the above figure, the 6-DOF motion components match perfectly with what I set in the linearization conditions. After I got this linearization results, I made regulator for hovering at the operating point. Using full-state feedback, all states go to desired states. The plant can be written in state-space form $\dot{x} = Ax + Bu$, and that all of the n states x are available for the controller.

As we can see in the below figure, the error increases initially in order to get stabilize (The LQR regulator is meant to keep all the states near zero).

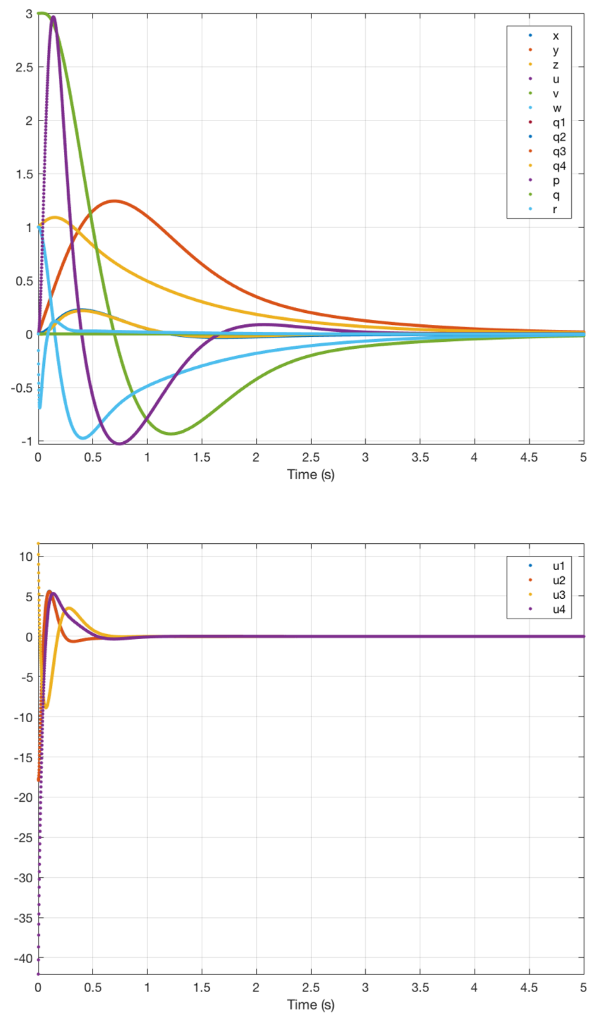


Figure 5.27: Regulator results for 90deg case

6.1 Control methods

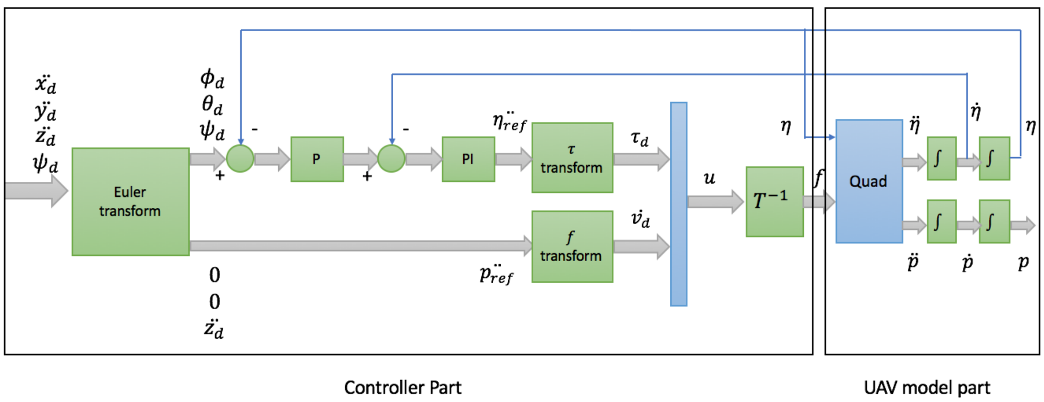


Figure 6.1: Control concept

6.1.1 Euler transform

The relationship between the acceleration of the UAV $\ddot{x}_d$, $\ddot{y}_d$ and ϕ_d, θ_d is like below.

$$\dot{P} = Rv \tag{6.1}$$

After differentiation,

$$\ddot{P} = R\dot{v} \tag{6.2}$$

where, ignore $\dot{R}v$.

$$\begin{bmatrix} \ddot{x} \\ \ddot{y} \\ \ddot{z} - g_z \end{bmatrix} = \begin{bmatrix} cos\theta\ cos\psi & sin\phi\ sin\theta\ cos\psi - cos\phi\ sin\psi & cos\phi\ sin\theta\ cos\psi + sin\phi\ sin\psi \\ cos\theta\ sin\psi & sin\phi\ sin\theta\ sin\psi + cos\phi\ cos\psi & cos\phi\ sin\theta\ sin\psi - sin\phi\ cos\psi \\ -sin\theta & sin\phi\ cos\theta & cos\phi\ cos\theta \end{bmatrix} \begin{bmatrix} 0 \\ 0 \\ \dot{v}_z \end{bmatrix}$$

$$\ddot{x} = \dot{v}_z(cos\phi\ sin\theta\ cos\psi + sin\phi\ sin\psi)$$

$$\ddot{y} = \dot{v}_z(cos\phi\ sin\theta\ sin\psi - sin\phi\ cos\psi)$$

$$\ddot{z} - g_z = \dot{v}_z\ cos\phi\ cos\theta$$

$$\sin\phi = \frac{\ddot{x}\ sin\psi\ -\ \ddot{y}\ cos\psi}{\sqrt{\ddot{x}^2 + \ddot{y}^2 + (\ddot{z} - g_z)^2}}$$

$$tan\theta = \frac{\ddot{x}\ cos\psi +\ \ddot{y}\ sin\psi}{\ddot{z} - g_z}$$

$$\phi_d \approx \frac{\ddot{x_d}\ sin\psi_d\ -\ \ddot{y_d}\ cos\psi_d}{\sqrt{\ddot{x_d}^2 + \ddot{y_d}^2 + (\ddot{z_d} - g_z)^2}}$$

$$\theta_d \approx \frac{\ddot{x_d}\ cos\psi_d +\ \ddot{y_d}\ sin\psi_d}{\ddot{z_d} - g_z}$$

Figure 6.2: The euler transform

6.1.2 The τ transform

The relationship equation (Euler's equation) between torque, angular velocity, and angular acceleration is like below.

$$\tau = I\dot{\omega} + \omega \times I\omega \tag{6.3}$$

$$\omega = C\dot{\eta} \tag{6.4}$$

After differentiation,

$$\dot{\omega} = \dot{C}\dot{\eta} + C\ddot{\eta} \tag{6.5}$$

From equation (6.3), (6.4), (6.5), we can rearrange like below.

$$\tau = I\dot{C}\dot{\eta} + IC\ddot{\eta} + C\dot{\eta} \times (IC\dot{\eta}) \tag{6.6}$$

where, the ignore disturbance $I\dot{C}\dot{\eta}$, $C\dot{\eta} \times (IC\dot{\eta})$. After then,

$$\tau_d \approx IC\ddot{\eta_{ref}} \tag{6.7}$$

6.1.3 The f transform

$$\dot{P} = Rv \tag{6.8}$$

After differentiation,

$$\begin{aligned} \ddot{P} &= \dot{R}v + R\dot{v} \\ &= \omega \times Rv + R\dot{v} \end{aligned} \tag{6.9}$$

After removing ω,

$$\ddot{P} = C\dot{\eta} \times Rv + R\dot{v} \tag{6.10}$$

Rearranging about $\dot{v}$ is like below,

$$\dot{v} = R^{-1}(\ddot{p} - C\dot{\eta} \times Rv) \tag{6.11}$$

where, the ignore disturbance $C\dot{\eta} \times Rv$. After then,

$$\dot{v_d} \approx R^{-1}\ddot{p_{ref}} \tag{6.12}$$

6.1.4 Calculation of the T^{-1}

The forces(f_1, f_2, f_3, f_4) generated from rotating motion of four rotors are related with the rigid body like below.

$$\begin{bmatrix} \ddot{z}_v \\ \tau \end{bmatrix} = \underbrace{\begin{bmatrix} \ddot{z}_v \\ \tau_x \\ \tau_y \\ \tau_z \end{bmatrix}}_{\mathbf{u}} = \underbrace{\begin{bmatrix} 1/m & 1/m & 1/m & 1/m \\ 0 & l & 0 & -l \\ -l & 0 & l & 0 \\ r & -r & r & -r \end{bmatrix}}_{\mathbf{T}} \underbrace{\begin{bmatrix} f_1 \\ f_2 \\ f_3 \\ f_4 \end{bmatrix}}_{\mathbf{f}}$$

Figure 6.3: The calculation for T^{-1}

where, m is the mass of the UAV, l is the distance between the face two rotors, r is the coefficient between the force and moment.

$$\begin{aligned} u &= Tf \\ f &= T^{-1}u \end{aligned} \tag{6.13}$$

6.1.5 The position (x, y, z) control

The controller to control x, y, z is like below.

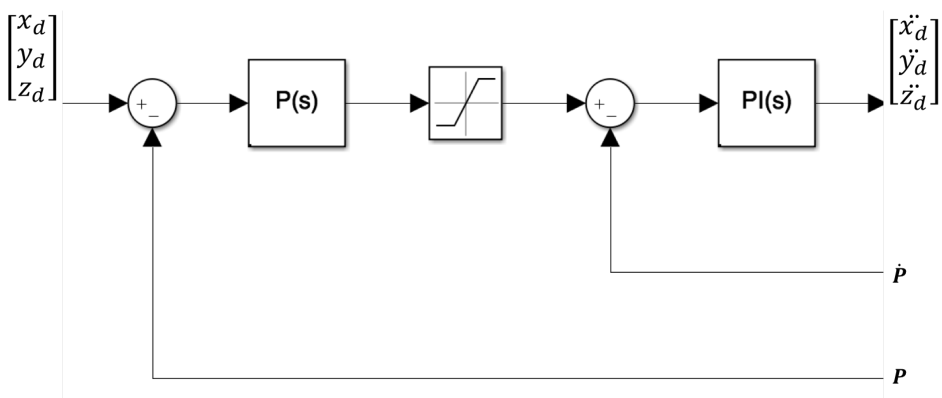


Figure 6.4: Controller concept to control x, y, z

6.1.6 Simulink model to control x, y, z

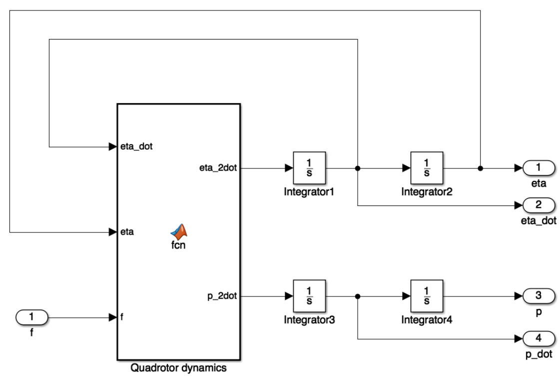


Figure 6.5: Simulink model for quadrotor dynamics

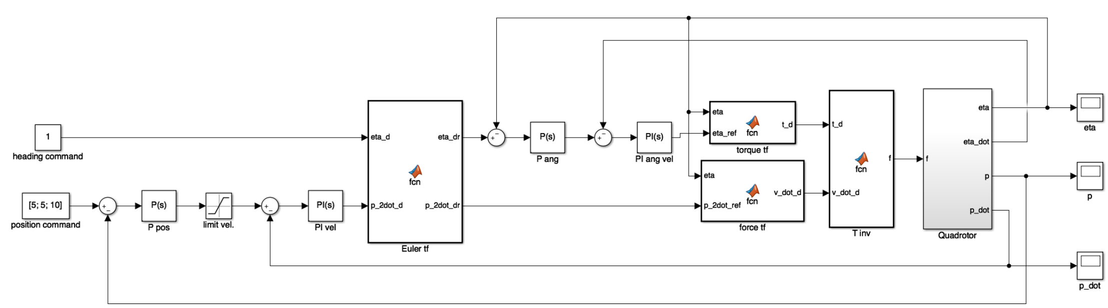


Figure 6.6: Simulink model for position & heading control

6.1.7 Simulink results

I set the position (5, 5, 10) and heading 1 rad, the results are like below.

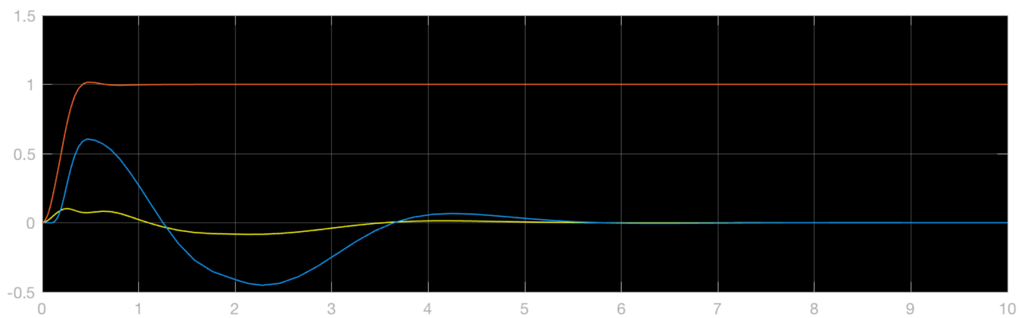


Figure 6.7: The angle of the UAV

where, red line is ψ, the blue line is θ, the yellow line is ϕ.

As we can see in the above figure, the heading of the UAV(ψ) go to the desired value 1 rad less than 0.5 second.

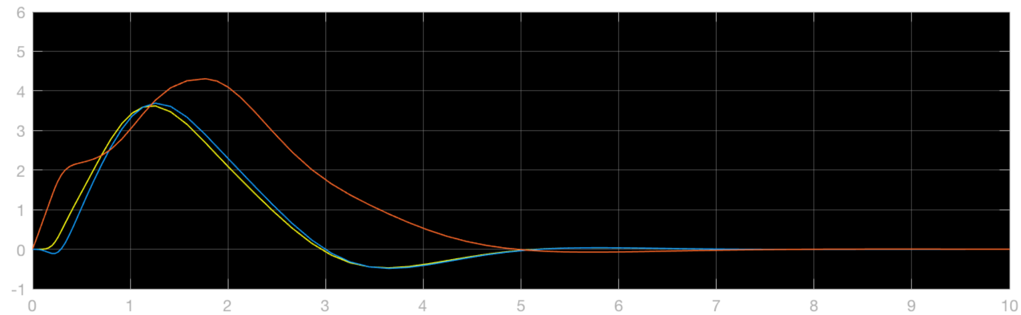


Figure 6.8: The velocity of the UAV

where, red line is $\dot{z}$, the blue line is $\dot{y}$, the yellow line is $\dot{x}$. As we can see in the above figure, the velocity of the UAV($\dot{x}$, $\dot{y}$, $\dot{z}$) go to the desired value 0.

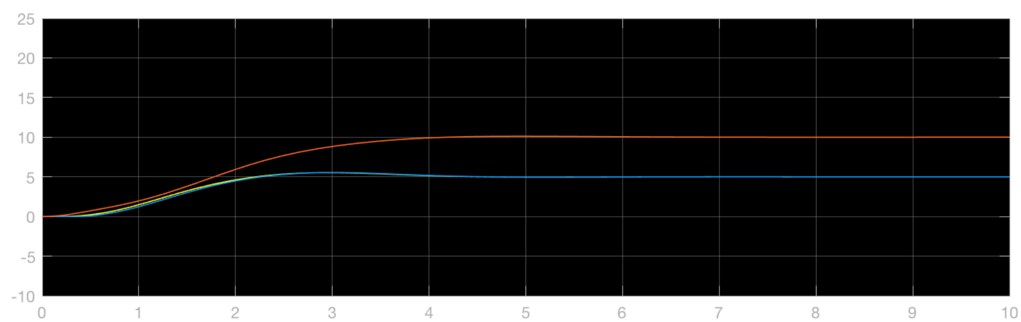


Figure 6.9: The position of the UAV

where, red line is z, the blue line is y, the yellow line is x. As we can see in the above figure, the position of the UAV(x, y, z) go to the desired value (5, 5, 10).

7. RESEARCH SUMMARY & FUTURE WORKS

7.1 Research summary

- The method of design and control of an omni-directional quadrotor with tilt-rotor system is provided
- The behavior of quadrotor under suggested control strategies is observed in MATLAB, Simulink, Simmechanics

7.2 Future works

- Output feedback controller
- Thrust coefficient experiment

I used full-state feedback control method; however, if I apply output-feedback controller in the near future, I can reduce the number of sensors to measure outputs.

REFERENCES

[1] D. J. Pines and F. Bohorquez, "Challenges facing future micro-air-vehicle development," *Journal of aircraft*, vol. 43, no. 2, pp. 290–305, 2006.

[2] T. Samad, J. S. Bay, and D. Godbole, "Network-centric systems for military operations in urban terrain: the role of uavs," *Proceedings of the IEEE*, vol. 95, no. 1, pp. 92–107, 2007.

[3] T. Tomic, K. Schmid, P. Lutz, A. Domel, M. Kassecker, E. Mair, I. L. Grixa, F. Ruess, M. Suppa, and D. Burschka, "Toward a fully autonomous uav: Research platform for indoor and outdoor urban search and rescue," *IEEE robotics & automation magazine*, vol. 19, no. 3, pp. 46–56, 2012.

[4] S. Waharte and N. Trigoni, "Supporting search and rescue operations with uavs," in *Emerging Security Technologies (EST), 2010 International Conference on*, pp. 142–147, IEEE, 2010.

[5] C. Özgen, *MODELING, STABILITY ANALYSIS AND CONTROL SYSTEM DESIGN OF A SMALL-SIZED TILTROTOR UAV*. PhD thesis, MIDDLE EAST TECHNICAL UNIVERSITY, 2009.

[6] A. Sanchez, J. Escareno, O. Garcia, and R. Lozano, "Autonomous hovering of a noncyclic tiltrotor uav: Modeling, control and implementation," *IFAC Proceedings Volumes*, vol. 41, no. 2, pp. 803–808, 2008.

[7] S. Dhaliwal and A. Ramirez-Serrano, "Control of an unconventional vtol uav for search and rescue operations within confined spaces based on the marc control architecture," in *Safety, Security & Rescue Robotics (SSRR), 2009 IEEE International Workshop on*, pp. 1–6, IEEE, 2009.

[8] C. Papachristos, K. Alexis, G. Nikolakopoulos, and A. Tzes, "Model predictive attitude control of an unmanned tilt-rotor aircraft," in *Industrial Electronics (ISIE), 2011 IEEE International Symposium on*, pp. 922–927, IEEE, 2011.

[9] F. Goncalves, J. Bodanese, R. Donadel, G. Raffo, J. Normey-Rico, and L. Becker, "Small scale uav with birotor configuration," in *Unmanned Aircraft Systems (ICUAS), 2013 International Conference on*, pp. 761–768, IEEE, 2013.

[10] O. Kreisher, "Finally, the osprey," *Air Force Magazine*, 2009.

[11] P. Heslinga, "Analysis and realization of a dual-nacelle tiltrotor aerial vehicle," 2014.

[12] M. Wierema, "Design, implementation and flight test of indoor navigation and control system for a quadrotor uav," *Master of Science in Aerospace Engineering at Delft University of Technology*, 2008.

[13] F. Outamazirt, F. Li, L. Yan, and A. Nemra, "Autonomous navigation system using a fuzzy adaptive nonlinear H_∞ filter," *Sensors*, vol. 14, no. 9, pp. 17600–17620, 2014.