AUTONOMOUS LANDING OF QUADROTOR ON A MOVING UGV WITH THE OPTIMAL CONTROL POLICIES

by

SUHAS PRIYATHAM MANDA

Presented to the Faculty of the Graduate School of
The University of Texas at Arlington in Partial Fulfillment
of the Requirements
for the Degree of

MASTER OF SCIENCE

THE UNIVERSITY OF TEXAS AT ARLINGTON
December 2020

To my parents, younger brother and supportive friend.

ACKNOWLEDGEMENTS

In this occasion, I am very much pleased to express my sincere gratitude to all the esteemed personalities for their generous guidance and motivation that helped me in accomplishing this great endeavor on *Autonomous landing of quadrotor on a moving UGV with the optimal control policies.*

Primarily I would like to thank Dr. Frank Lewis for his immense inspiration and for filling great enthusiasm in me during the coursework. I feel lucky and blessed to have him as my professor.

Further, I would like to express my heartfelt gratitude to Yusuf Kartal for patiently listening to my progressing hypothesis and giving invaluable guidance until the completion of the work. I could not have accomplished this work without his continuous motivation and unconditional support. And I thank him for serving as a chairperson.

I also take this opportunity to express my sincere thanks to Dr. Manry for his inspiring lectures on neural networks.

Further, I express my gratitude to Dr. Yan Wan and Dr. Jonathan Bredow for their willingness to serve on my committee. I especially thank the University of Texas at Arlington Research Institute family for providing me the experimentation platform.

Penultimately, my father, being an electrical engineer, motivated every moment of my life, and needless to say, he is my inspiration.

Finally, I take this opportunity to express my gratitude to my family for being supportive. Further, I remember that I am greatly influenced by my mathematics professor in high school, Mr. Vasu, and my friend Bala Gopal.

November 11, 2020

ABSTRACT

AUTONOMOUS LANDING OF QUADROTOR ON A MOVING UGV WITH THE OPTIMAL CONTROL POLICIES

SUHAS PRIYATHAM MANDA, M.S

The University of Texas at Arlington, 2020

Academic Advisors: Dr. Frank Lewis, and Yusuf Kartal

This thesis proposes an offline method that uses an integral reinforcement learning (IRL) technique along with the system identification to determine the optimal control of a system with completely unknown dynamics. Unmanned aerial vehicles (UAV) that are particularly deployed to track and land on an arbitrarily moving unmanned ground vehicles (UGV), demand a high performance controller to perform precise tracking. One way of designing an optimal tracking controller is developing linear quadratic integrators (LQI) with a quadratic type of cost function that solves Riccati equation. However, this approach requires prior knowledge of the linearized UAV system dynamics. We overcome this problem by employing an IRL technique that solves LQI through system identification. Usually, IRL techniques adopt a conventional way of solving the Hamilton–Jacobi–Bellman (HJB) equation with value function approximation. The proposed approach evaluates the optimal control using IRL that solves the HJB equation using system identification instead of value function approximation. Assuming that the UAV system dynamics are linear time-invariant over a particular flight condition, we identify the linear model by analyzing the input

and output data samples from a linear regression perspective, where we use the conjugate gradient descent optimization algorithm. This approach addresses the challenge to compute optimal control without the need to know UAV dynamics. We have rigorously tested and simulated the proposed method on various flight trajectories. The test results have shown significant improvement in the control policy over each iteration of IRL. After validating the proposed method in simulation, we have implemented this approach on a real UAV to track and land on a UGV.

TABLE OF CONTENTS

LIST OF ILLUSTRATIONS

LIST OF TABLES

CHAPTER 1

INTRODUCTION

1.1 Background and motivation

Unmanned vehicles, especially unmanned ground vehicle (UGV) and unmanned aerial vehicle (UAV), which gained great strategic significance due to high mobility and the potential to perform a specific task, have become a research hotspot in the fields of military surveillance and reconnaissance [1, 2, 3], and transportation [4]. Another factor to have such diverse applications is, because of the considerably simple dynamics. However, despite having simple dynamics, it is challenging to design a robust controller because of the under-actuation, aerodynamic uncertainty [5], coupling between lateral and longitudinal motions [6], and unmodelled disturbances. In addition to all these, achieving coordination among multiple unmanned vehicles adds an extra burden on the controller.

The initiative to solve the control problem is made by PID, which is one of the classical nonlinear controllers, yet still used for position control in unmanned vehicles. Though it is robust to slight uncertainties, the classical PID [7] may not assure the desired performance [8] in variable operating regions for UAVs. Therefore, this created a need to improvise PID [9, 10], resulting in the introduction of backstepping [11], sliding mode, and H-infinity [12, 13] controllers. Moreover, research has been carried out to control the linearized model of the quadrotor. The Linear Quadratic Regulator (LQR) [14] is a well-known feedback controller that computes the optimal feedback gains for a linear time invariant system subjected to a quadratic cost function by solving Riccati equation [15]. With the recent developments in the field of RAM/ROM

products, fuzzy logic, intelligent, reinforcement learning (RL) [16, 17] based control strategies have been employed to improve tracking and robustness, as they require high memory.

Inspired by learning mechanisms observed in the biological animals, RL is structured to learn the best policy, to maximize the performance by interacting with an unknown environment [16, 18]. The RL algorithms requires to remember successful control decisions, so that they will become more likely to be used for a second time. Integral reinforcement learning (IRL) computes the optimal control solution with a data-based approach to solve Hamilton–Jacobi–Bellman equation using partial knowledge on system dynamics [19]. [17] presents a solution to the optimal tracking control problem using IRL.

On the other side, parallel to the development in controllers, research has been conducted to overcome aerodynamic uncertainties by parameter estimation using system identification. [20] presents system identification using autoregressive with the exogenous model method [21, 22] for linear estimation and the Hammerstein-Wiener method [23, 24] for nonlinear estimation. However, the linear estimation can be considered as a linear regression problem, which can also be solved using the conjugate gradient [25, 26] approach. Intuitive techniques like usage of neural networks (NN) for system identification [27], did not come into practice [28, 29] until the recent outbreak in computational power. [30] proposed a closed-loop identification method based on a reinforcement learning algorithm for multiple-input multiple-output (MIMO) systems.

In this thesis, we propose a technique that guarantees a UAV to track and land on an arbitrarily moving UGV. The expectation of the trajectory followed by UGV is undetermined by UAV. To track such an arbitrary system, the design of the controller for UAV demands an exceptional tracking performance. We use the integral

reinforcement learning technique (IRL), which can compute the optimal control to deliver the desired tracking performance. However, the IRL calculates the optimal control based on the knowledge of system dynamics. We propose a novel approach that integrates the conjugate gradient algorithm with integral reinforcement learning to determine optimal control. This approach addresses the challenge of computing optimal control for a system with unknown dynamics. This approach is tested rigorously on various flight trajectories, where the tracking is improved incredibly over each iteration of the proposed IRL algorithm.

1.2 Thesis outline

The rest of the thesis is structured as follows:

- Chapter 2: The purpose of this chapter is to serve the basic understanding with the mathematical model of the unmanned vehicles, UAV and UGV. Starting with a brief explanation of reference frames and Euler angles, we derive the nonlinear model of a UAV based on Newton-Euler formalism, then followed by linearizing the derived nonlinear model subjected to small angle approximations. Lastly, for a skid steered UGV, we derive the kinematic model.
- Chapter 3: This chapter is dedicated to providing the strategies adopted to control the UAV and UGV. We discuss the linear quadratic integrator, enabling the UAV to track a predefined trajectory. Further, we prove the stability of the controller. Later we discuss the classical PID for UGV to achieve velocity tracking.
- Chapter 4: In this chapter, we proposed a novel approach to compute the optimal gain for a system with unknown dynamics by incorporating the conjugate gradient in integral reinforcement learning. Moreover, we provided a pseudo code for the proposed algorithm.

- Chapter 5: In this chapter, we initially shown the design of UAV and UGV models from chapter 2 and 3. Then, we illustrated the simulation results obtained by using the proposed algorithm and discussed the differences between theoretical and estimated models.
- Chapter 6: In this chapter, we briefly explained the experimental setup and the lab environment. Then we presented the implementation of tracking and landing on real UAV and UGV.
- Chapter 7: Finally, in this chapter, we drawn conclusions based on the proposed method and provided recommendations for further research.

CHAPTER 2

MATHEMATICAL MODEL

This chapter introduces a nonlinear model of the quadrotor, an unmanned aerial vehicle (UAV), along with the kinematics of an Unmanned Ground Vehicle (UGV). Firstly, we give a brief explanation of the reference frames. Secondly, we introduce Euler angles and their rotations in 3-dimensional (3D) space. Then, using the kinematics equations developed for the translational and rotational motion of the quadrotor, we introduce the nonlinear system dynamics of the UAV. Lastly, we rigorously reveal the linearized model, obtained by making use of the small perturbations around the equilibrium HOVER. Thereafter, finish the chapter by presenting the UGV kinematics.

2.1 Reference frames

A reference frame is a framework comprising an observer with a coordinate system of guided lines symbolically attached to a body, used to analyze the position and vector quantities. In aerospace, we often come across vehicle-carried, earth-fixed, and mobile reference frames. Based on the flat earth assumption, throughout this work, we will consider vehicle-carried and earth-fixed frames as the fixed reference frame.

The fixed frame is often referred as inertial frame spans 3D space in which Newton's laws are valid. In the fixed frame, based on our assumption, we use the *North-East-Down* convention so that X-axis position in 3D space aligns with

geographic North, Y-axis aligns with geographic East, and Z-axis points towards the center of the earth, with unit vectors being $\hat{i_e}$, $\hat{j_e}$, $\hat{k_e}$ respectively.

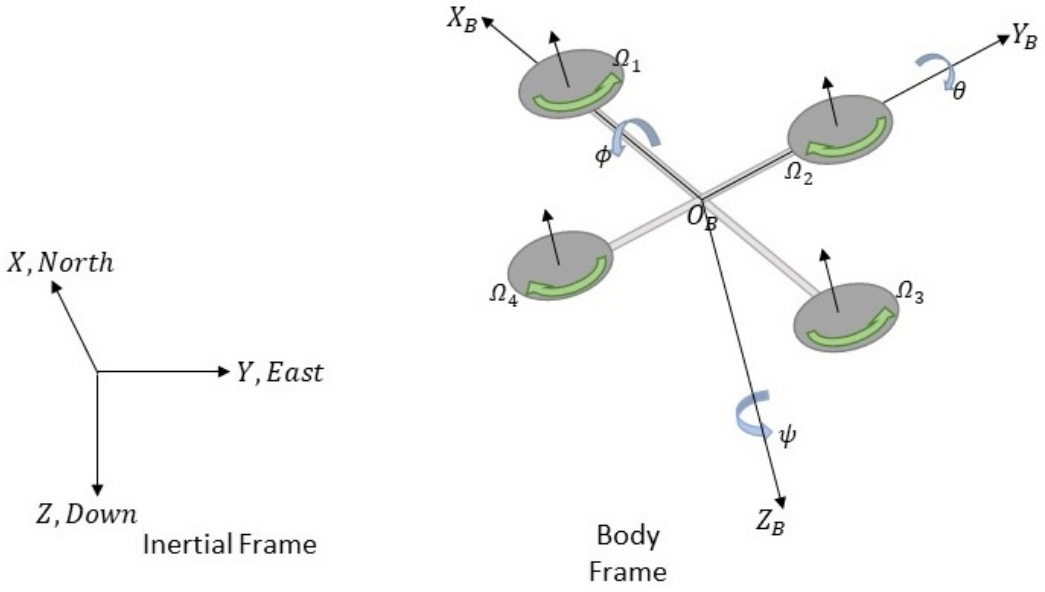


Figure 2.1. Frames and coordinate systems of a quadrotor.

The mobile frame often referred to as body frame, is described as an alignment in association with the body of the UAV, in our case it is quadrotor. The origin of the body frame, O_B, is assumed to be the center of mass of the quadrotor, while axes aligning with the arms of the quadrotor with unit vectors $\hat{i_B}$, $\hat{j_B}$, $\hat{k_B}$ (variables with suffix B refers to body frame).

2.2 Euler angles and rotations

Euler angles reflect a rigid body's orientation with respect to a fixed frame. Three parameters are required to define the orientation in a Euclidean 3-dimensional space. These parameters, ϕ(roll), θ(pitch), ψ(yaw), represent the amount of rotation

about the axes, X, Y, Z, respectively. The basic rotation matrices, $\mathbf{R_x}(\phi)$, $\mathbf{R_y}(\theta)$, $\mathbf{R_z}(\psi)$, using the right-hand rule are given by [31]

$$\mathbf{R_x}(\phi) = \begin{bmatrix} 1 & 0 & 0 \\ 0 & C_\phi & -S_\phi \\ 0 & S_\phi & C_\phi \end{bmatrix}, \tag{2.1}$$

$$\mathbf{R_y}(\theta) = \begin{bmatrix} C_\theta & 0 & S_\theta \\ 0 & 1 & 0 \\ -S_\theta & 0 & C_\theta \end{bmatrix}, \tag{2.2}$$

$$\mathbf{R_z}(\psi) = \begin{bmatrix} C_\psi & -S_\psi & 0 \\ S_\psi & C_\psi & 0 \\ 0 & 0 & 1 \end{bmatrix}, \tag{2.3}$$

where the notations $C.$ and $S.$ represents trigonometric functions $cos(\cdot)$ and $sin(\cdot)$.

We use the ZYX rotation sequence [31], to rotate a point about the origin of the fixed frame. The effective rotation matrix [32] will be the product of the base rotation matrices in Z-Y-X order. Then, the rotation matrix from X-frame to Y-frame to Z-frame is

$$\begin{aligned} \mathbf{R} = \mathbf{R_{zyx}}(\phi, \theta, \psi) &= \mathbf{R_z}(\phi) \cdot \mathbf{R_y}(\theta) \cdot \mathbf{R_x}(\psi) \\ &= \begin{bmatrix} C_\theta C_\psi & S_\phi S_\theta C_\psi - C_\theta S_\psi & C_\phi S_\theta C_\psi + S_\phi S_\psi \\ C_\theta S_\psi & S_\phi S_\theta S_\psi + C_\theta S_\psi & C_\phi S_\theta S_\psi - S_\phi C_\psi \\ -S_\theta & S_\phi C_\theta & C_\phi C_\theta \end{bmatrix}, \end{aligned} \tag{2.4}$$

and, $\mathbf{R}$ is a Special Orthogonal $SO(3)$ [33] matrix with the determinant equal to 1.

To transform angular velocities from the body frame to earth frame we use the matrix $\mathbf{T}$ given by

$$\mathbf{T} = \begin{bmatrix} 1 & S_\phi T_\theta & C_\phi T_\theta \\ 0 & C_\phi & -S_\phi \\ 0 & S_\phi / C_\theta & C_\phi / C_\theta \end{bmatrix}, \tag{2.5}$$

where the notations $T_{(.)}$ represents trigonometric function $tan(\cdot)$

2.3 Nonlinear model of quadrotor

The nonlinear model is constructed on examining kinematics and dynamics of the quadrotor. While kinematics studies the motion of objects without considering the forces that are responsible for the motion, dynamics studies the motion that results from the forces. In this section, we first begin with the quadrotor's rotational and translational kinematics. Then, using the kinematics, we formulate the dynamic model. Finally, we present the nonlinear mathematical model of the quadrotor.

Let us define the position vector essentially center of mass of the quadrotor, with respect to the inertial frame as $[\, x \;\; y \;\; z \,]^T$, and the orientation vector of the quadrotor i.e., Euler angles as $[\, \phi \;\; \theta \;\; \psi \,]^T$. The vectors with linear and angular velocities with respect to body frame is expressed as $\mathbf{v_B} = [\, u \;\; v \;\; w \,]^T$ and $\omega_\mathbf{B} = [\, p \;\; q \;\; r \,]^T$. Now, transformation of the velocities between the reference frames is given by

$$\mathbf{v} = \mathbf{R} \cdot \mathbf{v_B}, \tag{2.6}$$

$$\omega = \mathbf{T} \cdot \omega_\mathbf{B}, \tag{2.7}$$

where, $\mathbf{v} = [\, \dot{x} \;\; \dot{y} \;\; \dot{z} \,]^T$ is velocity in the inertial frame, and $\omega = [\, \dot{\phi} \;\; \dot{\theta} \;\; \dot{\psi} \,]^T$ is Euler rate in inertial frame.

The kinematics of the quadrotor are defined by these equations

$$\begin{aligned}
\dot{x} &= u(C_\theta C_\psi) + v(S_\phi S_\theta C_\psi - C_\theta S_\psi) + w(C_\phi S_\theta C_\psi + S_\phi S_\psi), \\
\dot{y} &= u(C_\theta S_\psi) + v(S_\phi S_\theta S_\psi + C_\theta S_\psi) + w(C_\phi S_\theta S_\psi - S_\phi C_\psi), \\
\dot{z} &= -u(S_\theta) + v(S_\phi C_\theta) + w(C_\phi C_\theta),
\end{aligned} \tag{2.8}$$

$$\begin{aligned}
\dot{\phi} &= p + q(S_\phi T_\theta) + r(C_\phi T_\theta), \\
\dot{\theta} &= q(C_\phi) - r(S_\phi), \\
\dot{\psi} &= q\frac{S_\phi}{C_\theta} + r\frac{C_\phi}{C_\theta}.
\end{aligned} \tag{2.9}$$

To determine the resultant force $\mathbf{F_B}$ and total moment $\mathbf{m_B}$ on the quadrotor we apply Newton's second law and Euler's equation to (2.6), and (2.7). Then we get

$$\mathbf{F_B} = m(\dot{\mathbf{v}}_\mathbf{B} + \omega_\mathbf{B} \times \mathbf{v_B}), \tag{2.10}$$

$$\mathbf{m_B} = \mathbf{I}\cdot \dot{\omega_\mathbf{B}} + \omega_\mathbf{B} \times (\mathbf{I}\cdot \omega_\mathbf{B}), \tag{2.11}$$

where $\times$ represents cross product, m is mass of the quadrotor, $\mathbf{F_B} = [\, F_{x_B} \;\; F_{y_B} \;\; F_{z_B} \,]^T$ is the force vector $(F_{x_B}\hat{i_B} + F_{y_B}\hat{j_B} + F_{z_B}\hat{k_B})$ in reference to the body frame, $\mathbf{m_B} = [\, m_{x_B} \;\; m_{y_B} \;\; m_{z_B} \,]^T$ is moment in the body frame, and $\mathbf{I}$ is inertia matrix containing scalar moments and cross-products of inertia [31],

$$\mathbf{I} = \begin{bmatrix} I_{xx} & -I_{xy} & -I_{xz} \\ -I_{yx} & I_{yy} & -I_{yz} \\ -I_{zx} & -I_{zy} & I_{zz} \end{bmatrix}, \tag{2.12}$$

where I_{xx}, I_{yy}, I_{zz}, are moment of inertia along X_B, Y_B, Z_B, axes respectively,

$$\begin{aligned}
I_{xx} &= \int (y^2 + z^2) dm, \\
I_{yy} &= \int (z^2 + x^2) dm, \\
I_{zz} &= \int (x^2 + y^2) dm.
\end{aligned} \tag{2.13}$$

Cross-product of inertia elements,

$$\begin{aligned} I_{xy} = I_{yx} &= \int xydm, \\ I_{xz} = I_{zx} &= \int xzdm, \\ I_{yz} = I_{zy} &= \int yzdm, \end{aligned} \tag{2.14}$$

are measures of symmetry. They are all equal to zero [33] as the structure of the quadrotor is symmetric along the planes X_BY_B, Y_BZ_B, and Z_BX_B. This reduces I to be a diagonal matrix, given by [34] as

$$\mathbf{I} = \begin{bmatrix} I_{xx} & 0 & 0 \\ 0 & I_{yy} & 0 \\ 0 & 0 & I_{zz} \end{bmatrix}, \tag{2.15}$$

Usually, the moment of inertia is measured by building the mechanical structure of the quadrotor using CAD tools. In [35], inertia is identified experimentally by creating a rotational pendulum.

Now, the dynamics in reference to body frame are

$$\begin{aligned} F_{xB} &= m(\dot{u} + qw - rv), \\ F_{y_B} &= m(\dot{v} - pw + ru), \\ F_{zB} &= m(\dot{u} + pv - qu), \end{aligned} \tag{2.16}$$

$$\begin{aligned} m_{xB} &= \dot{p}I_{xx} - qr(I_{yy} - I_{zz}), \\ m_{y_B} &= \dot{q}I_{yy} - pr(I_{zz} - I_{xx}), \\ m_{zB} &= \dot{r}I_{zz} - pq(I_{xx} - I_{yy}), \end{aligned} \tag{2.17}$$

These forces (2.15) and moments (2.16) can be varied by changing the four-rotor speeds through which attitude and position can be controlled. For instance, four motors producing the same thrust results in a vertical motion; unequal rotor thrust of

left-right rotors and front-rear rotors result in rolling and pitching moments respectively; unbalanced torque with pairs on diagonally opposite rotors results in yawing.

If the quadrotor is exposed to wind disturbances, in addition to gravitational pull, the wind also exerts a force. This gives us,

$$\mathbf{F_B} = \mathbf{R}^T[\,0\ 0\ mg\,]^T - F_t\hat{k_B} + \mathbf{F_w}^T, \tag{2.18}$$

where g is acceleration due to gravity, F_t is the thrust generated by the rotors along Z_B-axis, and $\mathbf{F_w} = [\,F_{wx}\ F_{wy}\ F_{wz}\,]^T$ holding the disturbance in body frame. Similarly, quadrotor's external moments are given by

$$\mathbf{m_B} = \tau_\mathbf{B} - \mathbf{g_a} + \tau_\mathbf{w}, \tag{2.19}$$

where $\mathbf{g_a}$ is gyroscopic moments which is negligible, $\tau_\mathbf{w}$ is the wind disturbance vector $[\,\tau_{wx}\ \tau_{wy}\ \tau_{wz}\,]^T$ in body frame, and $\tau_B = [\,\tau_x\ \tau_y\ \tau_z\,]^T$ are the control torques produced by the rotors in body frame. Significance of F_t and $\tau_\mathbf{B}$ can be further understood under the dynamics of actuator.

Equations corresponding to the dynamics of the quadrotor drawn from the pairs (2.16)-(2.18) and (2.17)-(2.19) and given by

$$\begin{aligned}
-mgS_\theta + F_{wx} &= m(\dot{u} + qw - rv),\\
mgS_\phi C_\theta + F_{wy} &= m(\dot{v} - pw + ru),\\
mgC_\phi C_\theta + F_{wz} - F_t &= m(\dot{u} + pv - qu),\\
\tau_x + \tau_{wx} &= \dot{p}I_{xx} - qr(I_{yy} - I_{zz}),\\
\tau_y + \tau_{wy} &= \dot{q}I_{yy} - pr(I_{zz} - I_{xx}),\\
\tau_z + \tau_{wz} &= \dot{r}I_{zz} - pq(I_{xx} - I_{yy}).
\end{aligned} \tag{2.20}$$

The interrelation between the system inputs (thrust and torques) and actuators is described by actuator dynamics. Using actuator dynamics, we determine the thrust

and torques that the rotors can generate at a given rotational speed. The system inputs are directly proportional to the square of rotor speeds and given by

$$\begin{bmatrix} F_t \\ \tau_x \\ \tau_x \\ \tau_x \end{bmatrix} = \begin{bmatrix} b & b & b & b \\ -bl & 0 & bl & 0 \\ 0 & -bl & 0 & bl \\ -d & d & -d & d \end{bmatrix} \begin{bmatrix} {\Omega_1}^2 \\ {\Omega_2}^2 \\ {\Omega_3}^2 \\ {\Omega_4}^2 \end{bmatrix}, \tag{2.21}$$

where b is the thrust factor, d is the drag factor and l is the distance from the center to the rotor assuming that the quadrotor is symmetrical. Ω_1, Ω_2, Ω_3, Ω_4 are rotational speeds of the four rotors. The dynamics of the quadrotor in terms of actuator dynamics are

$$\begin{aligned} &-mgS_\theta + F_{wx} = m(\dot{u} + qw - rv), \\ &mgS_\phi C_\theta + F_{wy} = m(\dot{v} - pw + ru), \\ &mgC_\phi C_\theta + F_{wz} - b({\Omega_1}^2 + {\omega_2}^2 + {\omega_3}^2 + {\omega_4}^2) = m(\dot{u} + pv - qu), \\ &bl(-{\Omega_1}^2 + {\Omega_3}^2) + \tau_{wx} = \dot{p}I_{xx} - qr(I_{yy} - I_{zz}), \\ &bl(-{\Omega_2}^2 + {\Omega_4}^2) + \tau_{wy} = \dot{q}I_{yy} - pr(I_{zz} - I_{xx}), \\ &d(-{\Omega_1}^2 + {\Omega_2}^2 - {\Omega_3}^2 + {\Omega_4}^2) + \tau_{wz} = \dot{r}I_{zz} - pq(I_{xx} - I_{yy}). \end{aligned} \tag{2.22}$$

The nonlinear model of quadrotor is described by preferring the states with Euler angles, fixed frame position and velocities, and body frame angular velocities. So the state vector is defined as $\mathbf{x} = [\, x \;\; y \;\; z \;\; \phi \;\; \theta \;\; \psi \;\; \dot{x} \;\; \dot{y} \;\; \dot{z} \;\; p \;\; q \;\; r \,]^T$. Then the first derivative of the states will be

$$\dot{\mathbf{x}} = [\, \dot{x} \;\; \dot{y} \;\; \dot{z} \;\; \dot{\phi} \;\; \dot{\theta} \;\; \dot{\psi} \;\; \ddot{x} \;\; \ddot{y} \;\; \ddot{z} \;\; \dot{p} \;\; \dot{q} \;\; \dot{r} \,]^T. \tag{2.23}$$

For small angles of movements [36] the fixed frame and body frame angular velocities are assumed to be equal. Therefore, $[\, \dot{\phi} \;\; \dot{\theta} \;\; \dot{\psi} \,]^T = [\, p \;\; q \;\; r \,]^T$. So, the first derivatives of

both linear and angular positions can be readily accessible from kinematic equations. However, by applying the Newton's law on (2.18) we can write,

$$m\dot{\mathbf{v}} = \mathbf{R}\cdot\mathbf{F_B} = [\,0\ 0\ mg\,]^T - F_t\mathbf{R}\cdot\hat{\mathbf{k}_\mathbf{B}} + \mathbf{R}\cdot{\mathbf{F_w}}^T, \tag{2.24}$$

where $\dot{\mathbf{v}} = [\,\ddot{x}\ \ddot{y}\ \ddot{z}\,]^T$, to acquire the second derivatives. Disregarding the wind disturbance, the linear accelerations are

$$\begin{aligned}
\ddot{x} &= -F_t(C_\phi S_\theta C_\psi + S_\phi S_\psi)/m, \\
\ddot{y} &= -F_t(C_\phi S_\theta S_\psi - S_\phi C_\psi)/m, \\
\ddot{z} &= -F_t(C_\phi C_\theta)/m + g.
\end{aligned} \tag{2.25}$$

Further, $\dot{p}, \dot{q}, \dot{r}$, can be obtained by subjecting the equations (2.20). Thus the final mathematical model of the quadrotor considering all the wind disturbances in the fixed frame is given by the following equations

$$
\begin{aligned}
\dot{x} &= u(C_\theta C_\psi) + v(S_\phi S_\theta C_\psi - C_\theta S_\psi) + w(C_\phi S_\theta C_\psi + S_\phi S_\psi),\\
\dot{y} &= u(C_\theta S_\psi) + v(S_\phi S_\theta S_\psi + C_\theta S_\psi) + w(C_\phi S_\theta S_\psi - S_\phi C_\psi),\\
\dot{z} &= -u(S_\theta) + v(S_\phi C_\theta) + w(C_\phi C_\theta),\\
\dot{\phi} &= p + q(S_\phi T_\theta) + r(C_\phi T_\theta),\\
\dot{\theta} &= q(C_\phi) - r(S_\phi),\\
\dot{\psi} &= q\frac{S_\phi}{C_\theta} + r\frac{C_\phi}{C_\theta},\\
\ddot{x} &= -\frac{F_t}{m}(C_\phi S_\theta C_\psi + S_\phi S_\psi) + \frac{f_{wx}}{m},\\
\ddot{y} &= -\frac{F_t}{m}(C_\phi S_\theta S_\psi - S_\phi C_\psi) + \frac{f_{wy}}{m},\\
\ddot{z} &= -\frac{F_t}{m}(C_\phi C_\theta) + g + \frac{f_{wz}}{m},\\
\dot{p} &= \frac{I_{yy} - I_{zz}}{I_{xx}}qr + \frac{\tau_x + \tau_{wx}}{I_{xx}},\\
\dot{q} &= \frac{I_{zz} - I_{xx}}{I_{yy}}pr + \frac{\tau_y + \tau_{wy}}{I_{yy}},\\
\dot{r} &= \frac{I_{xx} - I_{yy}}{I_{zz}}pq + \frac{\tau_z + \tau_{wz}}{I_{zz}},
\end{aligned}
\tag{2.26}
$$

where f_{wx}, f_{wy}, f_{wz} are the disturbances caused due to the same wind as in $\mathbf{F_{wB}}$ but in reference to the fixed frame. Potentially, $\mathbf{R}\cdot\mathbf{F_{wB}} = \mathbf{f_w} = [\, f_{wx} \;\; f_{wy} \;\; f_{wz} \,]^T$.

2.4 Linear model of quadrotor

In this section, we show how to linearize the nonlinear model with states $\mathbf{x} = [\, x \;\; y \;\; z \;\; \phi \;\; \theta \;\; \psi \;\; u \;\; v \;\; w \;\; p \;\; q \;\; r \,]^T$ around the equilibrium point to formulate the

linear model. Notice that this point is called as hover flight regime. Thus considering the states $\mathbf{x}$, $\dot{\mathbf{x}}$ in linear model will be

$$\dot{\mathbf{x}} = [\, \dot{x} \quad \dot{y} \quad \dot{z} \quad \dot{\phi} \quad \dot{\theta} \quad \dot{\psi} \quad \dot{u} \quad \dot{v} \quad \dot{w} \quad \dot{p} \quad \dot{q} \quad \dot{r} \,]^T. \tag{2.27}$$

So the nonlinear model for these states can be written from the kinematics (1.8)-(1.9) and the dynamics (1.20) equations, as

$$\begin{aligned}
\dot{x} &= u(C_\theta C_\psi) + v(S_\phi S_\theta C_\psi - C_\theta S_\psi) + w(C_\phi S_\theta C_\psi + S_\phi S_\psi),\\
\dot{y} &= u(C_\theta S_\psi) + v(S_\phi S_\theta S_\psi + C_\theta S_\psi) + w(C_\phi S_\theta S_\psi - S_\phi C_\psi),\\
\dot{z} &= -u(S_\theta) + v(S_\phi C_\theta) + w(C_\phi C_\theta),\\
\dot{\phi} &= p + q(S_\phi T_\theta) + r(C_\phi T_\theta),\\
\dot{\theta} &= q(C_\phi) - r(S_\phi),\\
\dot{\psi} &= q\frac{S_\phi}{C_\theta} + r\frac{C_\phi}{C_\theta},\\
\dot{u} &= -qw + rv - gS_\theta + \frac{F_{wx}}{m},\\
\dot{v} &= pw - ru + gS_\phi C_\theta + \frac{F_{wy}}{m},\\
\dot{w} &= -pv + qu - gC_\phi C_\theta + \frac{F_{wz} - F_t}{m},\\
\dot{p} &= \frac{I_{yy} - I_{zz}}{I_{xx}}qr + \frac{\tau_x + \tau_{wx}}{I_{xx}},\\
\dot{q} &= \frac{I_{zz} - I_{xx}}{I_{yy}}pr + \frac{\tau_y + \tau_{wy}}{I_{yy}},\\
\dot{r} &= \frac{I_{xx} - I_{yy}}{I_{zz}}pq + \frac{\tau_z + \tau_{wz}}{I_{zz}}.
\end{aligned} \tag{2.28}$$

When a constant input $\mathbf{u_{eq}} = [\, mg \;\; 0 \;\; 0 \;\; 0\,]^T$ is applied, quadrotor remains hovering in an equilibrium point say, $\mathbf{x_{eq}} = [\, x_{eq} \;\; y_{eq} \;\; z_{eq} \;\; 0 \;\; 0 \;\; 0 \;\; 0 \;\; 0 \;\; 0 \;\; 0 \;\; 0 \;\; 0\,]^T$. The nonlinear equations (2.28) are approximated to obtain $\mathbf{f}(\mathbf{x}, \mathbf{u}, \mathbf{d_w})$,

$$\mathbf{f}(\mathbf{x}, \mathbf{u}, \mathbf{d_w}) = \begin{cases} \dot{x} = u + v(\phi\theta - S_\psi) + w(\theta + \phi\psi), \\ \dot{y} = u(\psi) + v(\phi\theta\psi + \psi) + w(\theta\psi - \phi), \\ \dot{z} = -u(\theta) + v(\phi) + w(\phi), \\ \dot{\phi} = p + q(\phi\theta) + r(\theta), \\ \dot{\theta} = q - r(\phi), \\ \dot{\psi} = q(\phi) + r, \\ \dot{u} = -qw + rv - g\theta + \frac{F_{wx}}{m}, \\ \dot{v} = pw - ru + g\phi + \frac{F_{wy}}{m}, \\ \dot{w} = -pv + qu - g + \frac{F_{wz} - F_t}{m}, \\ \dot{p} = \frac{I_{yy} - I_{zz}}{I_{xx}} qr + \frac{\tau_x + \tau_{wx}}{I_{xx}}, \\ \dot{q} = \frac{I_{zz} - I_{xx}}{I_{yy}} pr + \frac{\tau_y + \tau_{wy}}{I_{yy}}, \\ \dot{r} = \frac{I_{xx} - I_{yy}}{I_{zz}} pq + \frac{\tau_z + \tau_{wz}}{I_{zz}}, \end{cases} \tag{2.29}$$

where $\mathbf{d_w} = [\, F_{wx} \;\; F_{wy} \;\; F_{wz} \;\; \tau_{wx} \;\; \tau_{wy} \;\; \tau_{wz}\,]^T$ is the wind disturbance. We use (2.29) to acquire the continuous linear time-invariant state-space model. The linearization [34] corresponding to the hover state $\mathbf{x_{eq}}$ for the input $\mathbf{u_{eq}}$ is given as

$$\dot{\mathbf{x}} = \left.\frac{\partial \mathbf{f}(\mathbf{x}, \mathbf{u}, \mathbf{d_w})}{\partial \mathbf{x}}\right|_{\substack{\mathbf{x_{eq}} \\ \mathbf{u_{eq}}}} (\mathbf{x} - \mathbf{x_{eq}}) + \left.\frac{\partial \mathbf{f}(\mathbf{x}, \mathbf{u}, \mathbf{d_w})}{\partial \mathbf{x}}\right|_{\substack{\mathbf{x_{eq}} \\ \mathbf{u_{eq}}}} (\mathbf{u} - \mathbf{u_{eq}}) + \left.\frac{\partial \mathbf{f}(\mathbf{x}, \mathbf{u}, \mathbf{d_w})}{\partial \mathbf{d_w}}\right|_{\substack{\mathbf{x}=\mathbf{x_{eq}} \\ \mathbf{u}=\mathbf{u_{eq}}}} (\mathbf{d_w}),$$

$$\dot{\mathbf{x}} = \mathbf{A} \cdot \mathbf{x} + \mathbf{B} \cdot \mathbf{u} + \mathbf{D_w} \cdot \mathbf{d_w}. \tag{2.30}$$

By, solving above equations the system matrices can be obtained as

$$\mathbf{A} = \begin{bmatrix} 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 0 & -g & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & g & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \end{bmatrix}, \tag{2.31}$$

$$\mathbf{B} = \begin{bmatrix} 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ -\frac{1}{m} & 0 & 0 & 0 \\ 0 & \frac{1}{I_{xx}} & 0 & 0 \\ 0 & 0 & \frac{1}{I_{yy}} & 0 \\ 0 & 0 & 0 & \frac{1}{I_{zz}} \end{bmatrix}, \mathbf{D_w} = \begin{bmatrix} 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 \\ \frac{1}{m} & 0 & 0 & 0 & 0 & 0 \\ 0 & \frac{1}{m} & 0 & 0 & 0 & 0 \\ 0 & 0 & \frac{1}{m} & 0 & 0 & 0 \\ 0 & 0 & 0 & \frac{1}{I_{xx}} & 0 & 0 \\ 0 & 0 & 0 & 0 & \frac{1}{I_{yy}} & 0 \\ 0 & 0 & 0 & 0 & 0 & \frac{1}{I_{zz}} \end{bmatrix}.$$

2.5 Kinematic model of skid-steer UGV

Skid-steering is a drive mechanism that independently drives the wheels or tracks on either side of the vehicle, and rotating the left and right wheels with different angular velocities govern the direction of motion. Such drive mechanisms have simple and robust construction as they do not need an explicit turning mechanism. Besides,

they can corner in a very small radius of curvature, which makes them extremely maneuverable and ideal for robotic applications.

It is a little challenging to obtain a mathematical model due to the complicated wheel-ground interactions and kinematic constraints [37]. To attain an acceptable mathematical model, we make the following assumptions as in [38],

- The center of mass of the vehicle coincides with its geometric center,
- Vehicle movies with the wheels always in contact with the horizontal surface,
- Two wheels on either side rotate at the same speed,
- Slip and skid forces between the tires and the ground are the only external forces acting on and do not change with the position on the surface,

These bring down to formulate the motion in 2D space and enable us to neglect drag force and forces due to the rolling of tires.

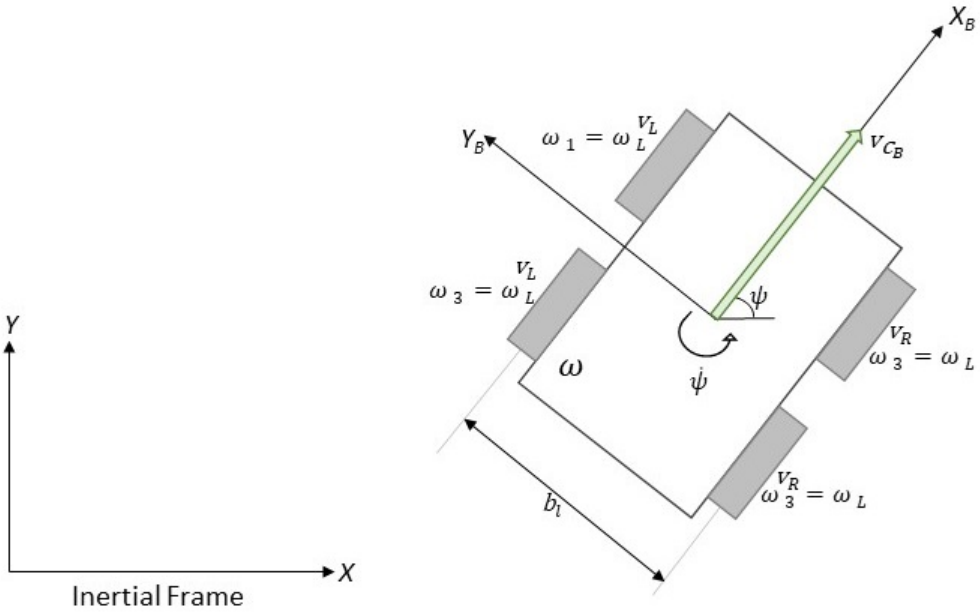


Figure 2.2. Skid-steered UGV.

Defining, v_1, v_2, v_3, v_4 and ω_1, ω_2, ω_3, ω_4 as linear velocity of the wheel's center and angular velocities of front-left, front right, rear-left, and rear-right wheels respectively. From the earlier assumptions, we have,

$$\begin{aligned} v_L &\triangleq v_1 = v_3, \quad v_R \triangleq v_2 = v_4, \\ \omega_L &\triangleq \omega_1 = \omega_3, \quad \omega_R \triangleq \omega_2 = \omega_4, \end{aligned} \tag{2.32}$$

where ω_L, ω_R is the angular velocities on left and right wheels, v_L, v_R are linear velocities of left and right wheel's centers in the vehicle's body frame. v_L and v_R in terms of velocity of the center of mass (or simply, vehicle velocity) in body frame, v_{C_B}, are given by [38],

$$\begin{aligned} v_L &= v_{C_B} - \frac{b_l}{2}\dot{\psi}, \\ v_R &= v_{C_B} + \frac{b_l}{2}\dot{\psi}, \end{aligned} \tag{2.33}$$

where b_l is the lateral wheel base and $\dot{\psi}$ is the yaw rate of the vehicle. The same equations can be further expressed as,

$$\begin{aligned} v_{C_B} &= \frac{1}{2}(v_L + v_R), \\ \dot{\psi} &= \frac{1}{b_l}(v_L - v_R). \end{aligned} \tag{2.34}$$

Longitudinal wheel slip (S): It is defined as the ratio of difference between the wheel tangential velocity and the velocity of wheel center relative to surface of the ground, which is represented by,

$$S_L = \frac{r_L\omega_L - v_L}{r_L\omega_L}, \qquad S_R = \frac{r_R\omega_R - v_R}{r_R\omega_R}, \tag{2.35}$$

where r_L, r_R are the radii of the left and right wheels respectively. Note that the tangential velocity of the wheel is radius times its angular velocity. Also note that the condition $0 \leq S \leq 1$ holds when the wheel is under traction, and $-\infty < S \leq 0$ holds when under braking.

Now, rewriting the body frame vehicle velocity,

$$v_{C_B} = \frac{1}{2}\big[(1 - S_L)r_L\omega_L + (1 - S_R)r_R\omega_R\big]. \tag{2.36}$$

The vehicle can move either forward or backward with respect to its body frame. Thus the component v_{C_B} is only along X_B-axis. For the kinematic model, it is essential to have the vehicle velocity in a fixed frame. Transformation of velocity into the fixed frame is achieved with the help of rotation matrix $\mathbf{R}_{zyx}(0, 0, \psi)$. The assumption that the vehicle moves on a horizontal surfaces, forces the components ϕ and θ in $\mathbf{R}$ to be equal to zero.

$$\begin{bmatrix} C_\psi & -S_\psi & 0 \\ S_\psi & C_\psi & 0 \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} v_{C_B} \\ 0 \\ 0 \end{bmatrix} = \begin{bmatrix} \dot{x} \\ \dot{y} \\ 0 \end{bmatrix}, \tag{2.37}$$

where $\dot{x}$ and $\dot{y}$ are the velocities in fixed frame along $\hat{i_e}$ and $\hat{j_e}$ respectively. Therefore, the kinematics of the skid-steer UGV are defined by,

$$\dot{x} = \frac{1}{2}\big[(1 - S_L)r_L\omega_L + (1 - S_R)r_R\omega_R\big]C_\psi, \tag{2.38}$$

$$\dot{y} = \frac{1}{2}\big[(1 - S_L)r_L\omega_L + (1 - S_R)r_R\omega_R\big]S_\psi, \tag{2.39}$$

$$\dot{\psi} = \frac{1}{b_l}\big[(1 - S_L)r_L\omega_L - (1 - S_R)r_R\omega_R\big]C_\psi. \tag{2.40}$$

Upon further examination, it is evident to say

$$v_{C_B} = \sqrt{\dot{x} + \dot{y}}. \tag{2.41}$$

CHAPTER 3

CONTROL STRATEGIES

This chapter explains the control techniques to operate unmanned vehicles. Tracking smooth trajectories is always a challenging problem for a quadrotor. It is mainly because the quadrotor is an underactuated system with only four independent inputs to control six degrees of freedom. Moreover, while deriving the mathematical model in the earlier chapter, we made some approximations. Nevertheless, advanced strategies are being developed either by improving the controller or by creating the trajectory. [11] introduces nonlinear backstepping for a Proportional Integral Differential (PID) controller to resolve the tracking problem. In [39], a real-time generation of optimal trajectories with nonlinear control is used to address the issue in indoor environments.

In section 3.1, we will limit ourselves unfolding the linear quadratic control method for trajectory tracking of the quadrotor. In 3.2, the PID controller is discussed for the velocity tracking in UGV.

3.1 Linear quadratic tracking

Linear Quadratic Regulator (LQR) is a conventional controller that provides optimal feedback gain and enables the closed-loop system to be stable over a fixed equilibrium point. The necessity to perform command tracking, termed as linear quadratic tracking in control theory, is solved by extending the scope of LQR design by introducing integral feedback, named as Linear Quadratic Integrator (LQI).

It is necessary to know the LQR control even before stating our control problem. Accordingly, let us briefly go through LQR first by defining the linear time-invariant system (2.30) without considering the disturbances,

$$\dot{\mathbf{x}} = \mathbf{A}\cdot\mathbf{x} + \mathbf{B}\cdot\mathbf{u}, \tag{3.1}$$

The quadratic performance index [15] related to the system (3.1) is

$$J(t_0) = \frac{1}{2}\int_{t_0}^{T} (\mathbf{x}^{\mathrm{T}}\mathbf{Q}\mathbf{x} + \mathbf{u}^{\mathrm{T}}\mathbf{R}\mathbf{u})dt, \tag{3.2}$$

where the time interval $[t_0,\ T]$ is the period we consider to determine the optimal control $u^*(t)$ that minimizes J. The weighting matrix Q is symmetric and positive semi-definite and R is symmetric and positive definite. The Hamiltonian is given by [15]

$$\mathbf{H} = \frac{1}{2}(\mathbf{x}^{\mathrm{T}}\mathbf{Q}\mathbf{x} + \mathbf{u}^{\mathrm{T}}\mathbf{R}\mathbf{u}) + \lambda^{\mathrm{T}}(\mathbf{A}\mathbf{x} + \mathbf{B}\mathbf{u}), \tag{3.3}$$

where $\lambda(t)$ is an undetermined multiplier. The state and costate equations [15] are

$$\dot{\mathbf{x}} = \frac{\partial \mathbf{H}}{\partial \lambda} = \mathbf{A}\mathbf{x} + \mathbf{B}\mathbf{u}, \tag{3.4}$$

$$-\dot{\lambda} = \frac{\partial \mathbf{H}}{\partial \mathbf{x}} = \mathbf{Q}\mathbf{x} + \mathbf{A}^{\mathrm{T}}\lambda, \tag{3.5}$$

and the stationary condition [15] is

$$0 = \frac{\partial \mathbf{H}}{\partial \mathbf{u}} = \mathbf{R}\mathbf{u} + \mathbf{B}^{\mathrm{T}}\lambda. \tag{3.6}$$

From the stationary condition, it is possible to write the control as

$$\mathbf{u}(t) = -\mathbf{R}^{-1}\mathbf{B}^{\mathrm{T}}\lambda(t). \tag{3.7}$$

Minimizing J for a given initial state $\mathbf{x}(t_0)$ and free final state $\mathbf{x}(T)$ with closed-loop control, the two-point boundary problem can be solved using the sweep method

(Bryson and Ho 1975). So, we can assume that $\mathbf{x}(t)$ and $\lambda(t)$ are linearly dependent on some unknown positive semi-definite symmetric matrix function $\mathbf{S}(t)$ [15]

$$\lambda(t) = \mathbf{S}(t)\mathbf{x}(t), \qquad \forall \quad t \in [t_0, T]. \tag{3.8}$$

Differentiating the assumption (3.8) and further substituting (3.1), (3.7),

$$\begin{aligned}
\dot{\lambda}(t) &= \dot{\mathbf{S}}(t)\mathbf{x}(t) + \mathbf{S}(t)\dot{\mathbf{x}}(t), \\
\dot{\lambda}(t) &= \dot{\mathbf{S}}\mathbf{x}(t) + \mathbf{S}(\mathbf{A}\mathbf{x} + \mathbf{B}\mathbf{u}), \\
\dot{\lambda}(t) &= \dot{\mathbf{S}}\mathbf{x}(t) + \mathbf{S}(\mathbf{A}\mathbf{x} - \mathbf{B}\mathbf{R}^{-1}\mathbf{B}^{\mathrm{T}}\lambda(t)), \\
\dot{\lambda}(t) &= \dot{\mathbf{S}}\mathbf{x}(t) + \mathbf{S}\big(\mathbf{A}\mathbf{x} - \mathbf{B}\mathbf{R}^{-1}\mathbf{B}^{\mathrm{T}}\mathbf{S}\mathbf{x}(t)\big).
\end{aligned}$$

Considering the costate equation, we can further write it as

$$\begin{aligned}
-\big(\mathbf{Q}\mathbf{x}(t) + \mathbf{A}^{\mathrm{T}}\mathbf{S}\mathbf{x}(t)\big) &= \dot{\mathbf{S}}\mathbf{x}(t) + \mathbf{S}\mathbf{A}\mathbf{x} - \mathbf{S}\mathbf{B}\mathbf{R}^{-1}\mathbf{B}^{\mathrm{T}}\mathbf{S}\mathbf{x}(t), \\
-\dot{\mathbf{S}} = \mathbf{A}^{\mathrm{T}}\mathbf{S} + \mathbf{S}\mathbf{A} - \mathbf{S}\mathbf{B}\mathbf{R}^{-1}\mathbf{B}^{\mathrm{T}}\mathbf{S} + \mathbf{Q}, &\qquad \forall \quad t \leq T.
\end{aligned} \tag{3.9}$$

(3.9) is a matrix Riccati equation, and solving it backward in time for $\mathbf{S}$ gives us the optimal control

$$\mathbf{u}^*(t) = -\mathbf{R}^{-1}\mathbf{B}^{\mathrm{T}}\mathbf{S}\mathbf{x}(t). \tag{3.10}$$

Representing Kalman gain K as [15]

$$\mathbf{K} = \mathbf{R}^{-1}\mathbf{B}^{\mathrm{T}}\mathbf{S}, \tag{3.11}$$

we have optimal control

$$\mathbf{u}^*(t) = -\mathbf{K}\mathbf{x}(t). \tag{3.12}$$

Note that the Riccati equation is nonlinear and is not straightforward to solve. However, efficient numerical methods, [40], [41], are available to find the solution. We will use MATLAB functions to solve the equation.

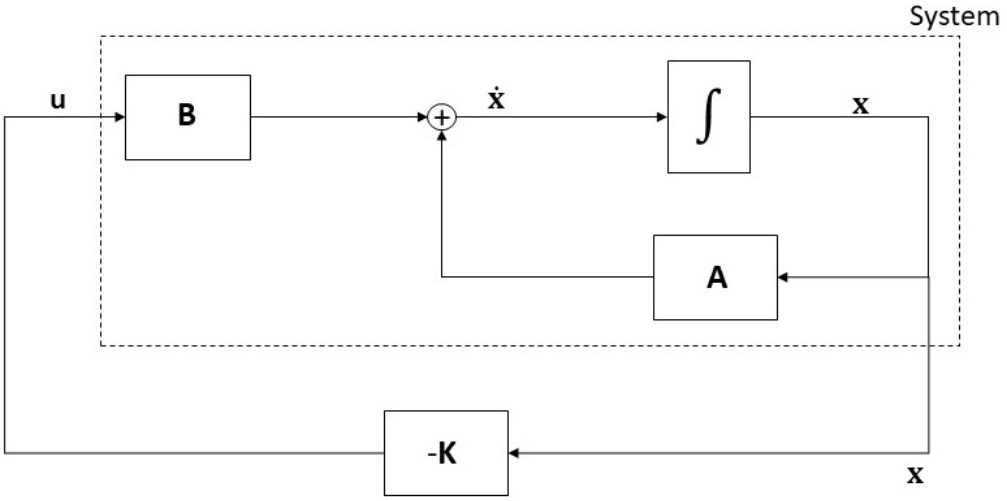


Figure 3.1. Schematic of LQR.

Now, let us get back to LQI to the solve the tracking problem using LQR for the linear time-invariant system

$$\begin{aligned} \dot{\mathbf{x}} &= \mathbf{A}\cdot\mathbf{x} + \mathbf{B}\cdot\mathbf{u}, \\ \mathbf{y} &= \mathbf{C}\cdot\mathbf{x}. \end{aligned} \tag{3.13}$$

Redefining the state vector as

$$\mathbf{x} = \begin{bmatrix} \mathbf{x_T} \\ \cdots\cdots \\ \mathbf{x_N} \end{bmatrix}, \tag{3.14}$$

where $\mathbf{x_T}$ holds the states we are interested to track, in our case the position, and $\mathbf{x_N}$ are non tracking states, we choose the matrix $\mathbf{C}$ in (3.1) such that

$$\mathbf{y} = \mathbf{C}\mathbf{x} = \mathbf{x_T}. \tag{3.15}$$

Control problem: To design the optimal control $\mathbf{u}^*(t)$, such that a part of the state vector, $\mathbf{x}(t)$, of the linear system (3.13) tracks a command reference signal $\mathbf{r_c}(t)$.

Mathematically, the problem is expressed as

$$\mathbf{x_T}(t) \rightarrow \mathbf{r_c}(t) \qquad \forall t.$$

Solution: Augment the system dynamics with integral of the state error [42]. Therefore with the augmented dynamics, control is given by

$$\mathbf{u}(t) = -\mathbf{K}\bar{\mathbf{x}}, \tag{3.16}$$

where,

$$\bar{\mathbf{x}} = \begin{bmatrix} \mathbf{x}(t) \\ \hdashline \mathbf{x_i}(t) \end{bmatrix},$$

and $\mathbf{x_i}(t)$ is integral of the state error. The difference between $\mathbf{r_c}(t)$ and $\mathbf{x_T}(t)$ gives the state error,

$$\mathbf{x_e}(t) = \mathbf{r_c}(t) - \mathbf{x_T}(t). \tag{3.17}$$

Thus,

$$\mathbf{x_i}(t) = \int \mathbf{x_e}(t)dt = \int \big(\mathbf{r_c}(t) - \mathbf{x_T}(t)\big)dt. \tag{3.18}$$

(3.16) will yield an optimal control solution when the gain $\mathbf{K}$ is solved from the matrix Riccati equation (3.9). However, it is to be noted that changing the system dynamics will affect the system matrices we defined in (3.13) and can no longer be used to solve for $\mathbf{K}$. Therefore, we redefine the system matrices by augmenting them [42], [43] as

$$\mathbf{A_{new}} = \left[\begin{array}{c:c} \mathbf{A} & 0 \\ \hdashline -\mathbf{C} & 0 \end{array}\right], \quad \mathbf{B_{new}} = \left[\begin{array}{c} \mathbf{B} \\ \hdashline 0 \end{array}\right]. \tag{3.19}$$

Using (3.19) to solve the Riccati equation will make the control defined in (3.16) to be optimal.

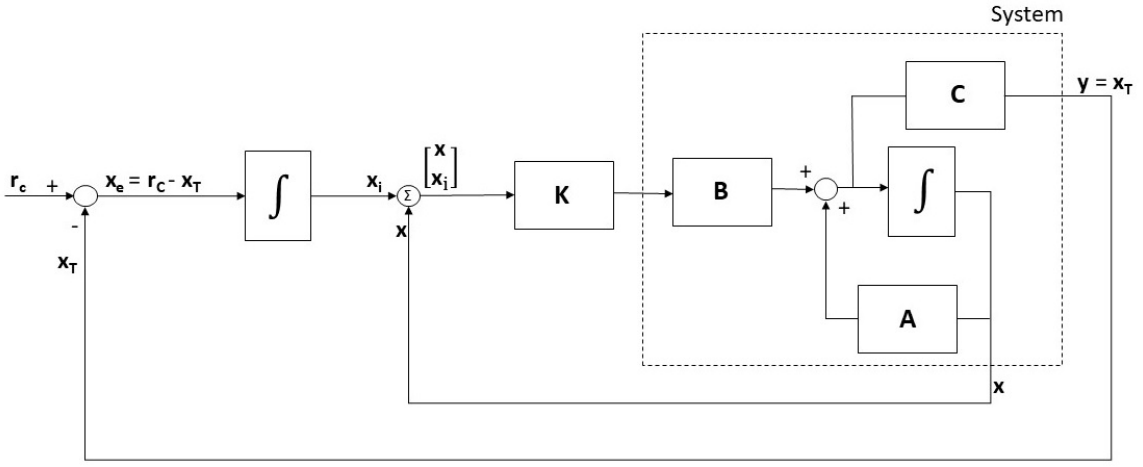


Figure 3.2. Schematic of LQI.

3.1.1 Stability of LQI

Here, we will show that the LQI control mentioned in section 3.1 is stabilizing the system, by using Lyapunov function $L(\mathbf{x})$.

The linear time-invariant system with the augmented states is given by,

$$\dot{\bar{\mathbf{x}}} = \mathbf{A_{new}} \cdot \bar{\mathbf{x}} + \mathbf{B_{new}} \cdot \mathbf{u}, \tag{3.20}$$

with performance index,

$$J_{new} = \frac{1}{2} \int_{t_0}^{T} (\bar{\mathbf{x}}^{\mathrm{T}} \mathbf{Q_{new}} \bar{\mathbf{x}} + \mathbf{u}^{\mathrm{T}} \mathbf{R} \mathbf{u}) dt. \tag{3.21}$$

Note that, the augmented system with $\mathbf{A_{new}}$, $\mathbf{B_{new}}$ is still controllable. Then the algebraic Riccati equation with the updated variables, and the optimal gain $\mathbf{K}$ is given by,

$$\mathbf{P}\mathbf{A_{new}} + \mathbf{A_{new}}^{\mathrm{T}}\mathbf{P} - \mathbf{P}\mathbf{B_{new}}\mathbf{R}^{-1}\mathbf{B_{new}}^{\mathrm{T}}\mathbf{P} + \mathbf{Q_{new}} = 0, \tag{3.22}$$

$$\mathbf{K} = \mathbf{R}^{-1}\mathbf{B_{new}}^{\mathrm{T}}\mathbf{P}. \tag{3.23}$$

where, $\mathbf{P}$ is a symmetric positive semi-definite matrix. Let us assume the Lyapunov function to be

$$L(\bar{\mathbf{x}}) = \bar{\mathbf{x}}^T \mathbf{P} \bar{\mathbf{x}}. \tag{3.24}$$

Differentiating the Lyapunov function gives

$$\dot{L} = \dot{\bar{\mathbf{x}}}^T \mathbf{P} \bar{\mathbf{x}} + \bar{\mathbf{x}}^T \mathbf{P} \dot{\bar{\mathbf{x}}}. \tag{3.25}$$

Using the equations (3.20) and (3.23) with the relation $\mathbf{u} = -\mathbf{K}\bar{\mathbf{x}}$, $\dot{L}$ can be rewritten as,

$$\begin{aligned}
\dot{L} &= \big[\mathbf{A_{new}} \cdot \bar{\mathbf{x}} + \mathbf{B_{new}} \cdot \mathbf{u}\big]^T \mathbf{P} \bar{\mathbf{x}} + \bar{\mathbf{x}}^T \mathbf{P} \big[\mathbf{A_{new}} \cdot \bar{\mathbf{x}} + \mathbf{B_{new}} \cdot \mathbf{u}\big], \\
\dot{L} &= \big[\mathbf{A_{new}} \bar{\mathbf{x}} - \mathbf{B_{new}} \mathbf{R}^{-1} \mathbf{B_{new}}^{\mathrm{T}} \mathbf{P} \bar{\mathbf{x}}\big]^T \mathbf{P} \bar{\mathbf{x}} + \bar{\mathbf{x}}^T \mathbf{P} \big[\mathbf{A_{new}} \bar{\mathbf{x}} - \mathbf{B_{new}} \mathbf{R}^{-1} \mathbf{B_{new}}^{\mathrm{T}} \mathbf{P} \bar{\mathbf{x}}\big], \\
\dot{L} &= \bar{\mathbf{x}}^T \Big[\big(\mathbf{A_{new}}^T - \mathbf{P} \mathbf{B_{new}} \mathbf{R}^{-1} \mathbf{B_{new}}^T\big) \mathbf{P} + \mathbf{P} \big(\mathbf{A_{new}} - \mathbf{B_{new}} \mathbf{R}^{-1} \mathbf{B_{new}}^T \mathbf{P}\big) \Big] \bar{\mathbf{x}}, \\
\dot{L} &= \bar{\mathbf{x}}^T \big[\mathbf{A_{new}}^{\mathrm{T}} \mathbf{P} - \mathbf{P} \mathbf{B_{new}} \mathbf{R}^{-1} \mathbf{B_{new}}^{\mathrm{T}} \mathbf{P} + \mathbf{P} \mathbf{A_{new}} - \mathbf{P} \mathbf{B_{new}} \mathbf{R}^{-1} \mathbf{B_{new}}^T \mathbf{P}\big] \bar{\mathbf{x}}.
\end{aligned} \tag{3.26}$$

Using (3.22), (3.26) can be simplified as,

$$\begin{aligned}
\dot{L} &= \bar{\mathbf{x}}^T \big[- \mathbf{Q_{new}} - \mathbf{P} \mathbf{B_{new}} \mathbf{R}^{-1} \mathbf{B_{new}}^T \mathbf{P}\big] \bar{\mathbf{x}}, \\
\dot{L} &= -\bar{\mathbf{x}}^T \big[\mathbf{Q_{new}} + \mathbf{P} \mathbf{B_{new}} \mathbf{R}^{-1} \mathbf{B_{new}}^T \mathbf{P}\big] \bar{\mathbf{x}}.
\end{aligned} \tag{3.27}$$

Notice that, $\mathbf{Q_{new}}$ is a positive semi-definite matrix and $\mathbf{R}$ is positive a definite matrix. If $\mathbf{R}$ is positive definite, then $\mathbf{R}^{-1}$ will be positive definite as well.

For some matrices M and T, MTM^T will be in quadratic form and will be positive definite as long as T is positive definite. Notice that $\mathbf{P} \mathbf{B_{new}} \mathbf{R}^{-1} \mathbf{B_{new}}^T \mathbf{P}$ can be expressed as $(\mathbf{P} \mathbf{B_{new}}) \mathbf{R}^{-1} (\mathbf{P} \mathbf{B_{new}})^T$, which is in quadratic form and is positive definite.

Hence, $(\mathbf{Q_{new}} + \mathbf{P} \mathbf{B_{new}} \mathbf{R}^{-1} \mathbf{B_{new}}^T \mathbf{P})$ is always a positive definite. Therefore, $\dot{L}$ is negative definite,

$$\dot{L}(\mathbf{x}) < 0. \tag{3.28}$$

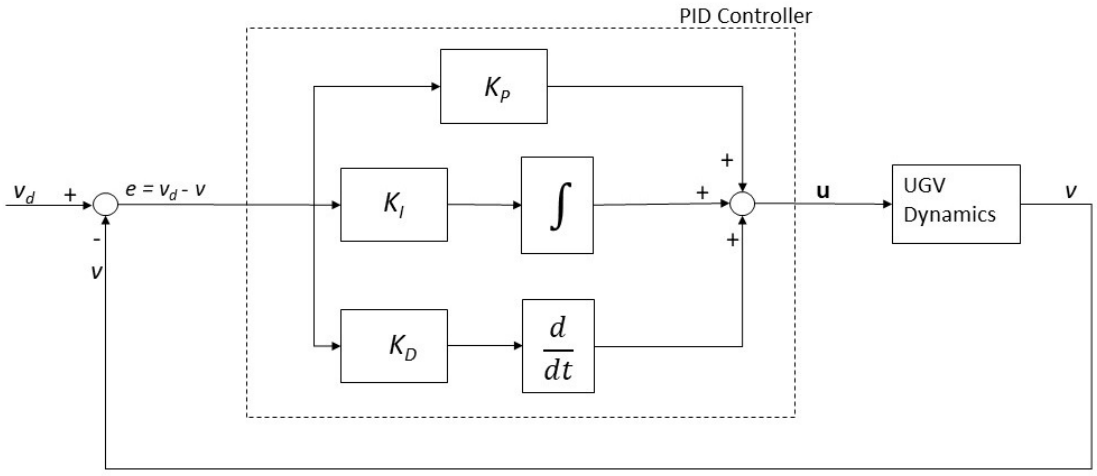


Figure 3.3. Schematic of PID.

So, we can conclude that the system is always asymptotically stable.

3.2 PID control for UGV

Despite being one of the classical control strategies, PID is the widely used controller to date. This is because of the robust performance it can deliver to systems with diverse dynamics while having simple tuning parameters. Most existing control techniques deployed in unmanned vehicles rely on some kind of PID Control. In [11] and [44] PID is used with backstepping for attitude control of UAV, making the system robust. In [45], an algorithm is presented to optimize the PID parameters to improve the controller efficiency. This section focuses on trajectory tracking with PID using velocity control in a UGV.

Let K_P, K_I, and K_D denote the proportional, integral, and differential gains of the PID controller, as shown in fig (3.3). The error signal $e(t)$, in our case, the

difference between the desired and estimated velocity of the UGV, fed to the to the controller is given by,

$$e(t) = v_d(t) - v(t), \tag{3.29}$$

where $v_d(t)$ is the desired velocity, and $V(t)$ is the estimated velocity. The desired input generated by the PID corresponding to $e(t)$ is

$$u(t) = K_P e(t) + K_I \int e(t)dt + K_D \frac{de(t)}{dt}. \tag{3.30}$$

Applying Laplace transform to (3.21),

$$U(s) = K_P E(t) + \frac{1}{s} K_I E(s) + K_D s E(s), \tag{3.31}$$

we can further realise the characteristics in s domain, with the transfer function

$$\frac{U(s)}{E(s)} = K_P E(t) + \frac{1}{s} K_I + K_D s. \tag{3.32}$$

The performance of the system in terms of rise-time, overshoot, settling time, and steady-state error can be set to the desired values by tuning these gains appropriately.

CHAPTER 4

SYSTEM IDENTIFICATION WITH SIMULTANEOUS OPTIMAL CONTROL ESTIMATION

System identification is one of the essential aspects to be considered before designing a controller. It helps us in understanding the system model with the intent that we can examine the performance of the control strategy while deploying a suitable controller. By analyzing the input and output data of the system, the system identification algorithm applies an estimation method to derive values of the parameters in the estimated model. The identification algorithms usually include Artificial Neural Networks (NN) [27] and Deep learning [28] techniques. As the modern industry keep on evolving exponentially, conventional identification techniques as in [27] have became extremely difficult to analyze the system dynamics efficiently because of the uncertainty, time delays, and input-output constraints [30]. However, due to the advancement in computational power, efficient algorithms, [29], [28], are being employed. [29] derives a mathematical relationship between the network weights and the transfer function parameters. A closed-loop identification method based on a reinforcement learning algorithm is proposed for multiple-input multiple-output systems in [30].

In this chapter, we carry out system identification by analyzing the measured data from the perspective of the linear regression problem. Because we already know that our system, the quadrotor, can be linearized. So we begin with the conjugate gradient algorithm to solve linear regression in section 4.1. Thereby in section 4.2, we explain the IRL technique to determine optimal gain. Finally, in section 4.3,

we propose an approach to evaluate the optimal control policy by simultaneously estimating the system.

4.1 Conjugate gradient

Conjugate gradient (CG) algorithm [46], [47] is an iterative method primarily invented for minimizing a quadratic function

$$F(\boldsymbol{a}) = \frac{1}{2}\boldsymbol{a^T} G\boldsymbol{a} - \boldsymbol{h}\cdot\boldsymbol{a} + \boldsymbol{c} = 0, \tag{4.1}$$

where G is $n \times n$ symmetric and positive definite matrix and $\boldsymbol{a} \in \mathbb{R}^n$. It is identical to solving for $\boldsymbol{a}$ in $\nabla F(\boldsymbol{a}) = 0$, which is a linear equation

$$G\boldsymbol{a} = \boldsymbol{h}. \tag{4.2}$$

Let us represent (4.2) as a linear system $r(\boldsymbol{a})$,

$$r(\boldsymbol{a}) = G\boldsymbol{a} - \boldsymbol{h}, \tag{4.3}$$

so that at an instant $\boldsymbol{a} = \boldsymbol{a_k}$, we have r_k as,

$$r_k = G\boldsymbol{a_k} - \boldsymbol{h}. \tag{4.4}$$

A set of nonzero vectors $\{\boldsymbol{p_0}, \boldsymbol{p_1}, \ldots, \boldsymbol{p_{n-1}}\}$ are said to be conjugate and linearly independent with respect to the matrix A [46] if

$$\boldsymbol{p_i}^T G\boldsymbol{p_j} = 0 \qquad \forall \quad i \neq j. \tag{4.5}$$

We can generate the sequence $\{\boldsymbol{a_k}\}$ as

$$\boldsymbol{a_{k+1}} = \boldsymbol{a_k} + \alpha_k \boldsymbol{p_k}, \tag{4.6}$$

with initial point $\boldsymbol{a_0} \in \mathbb{R}^n$ and the set of conjugate vector directions, where α_k is the one dimensional minimizer of the quadratic function $F(\boldsymbol{a})$ along $\boldsymbol{a_k} + \alpha\boldsymbol{p_k}$ given as [46]

$$\alpha_k = -\frac{{r_k}^T \boldsymbol{p_k}}{\boldsymbol{p_k}^T G\boldsymbol{p_k}} \tag{4.7}$$

Theorem 1: *For any* $\boldsymbol{a_o} \in \mathbb{R}^n$ *the sequence* $\{\boldsymbol{a_k}\}$ *generated by the conjugate direction algorithm (4.6), (4.7) converges to the solution* $\boldsymbol{a}^*$ *of the linear system (4.2) in at most n steps.* [46]

Proof: The direction vectors $\{\boldsymbol{p_i}\}$ covers the entire space $\mathbb{R}^n$ as they are linearly independent. Hence, we can write the difference between $\boldsymbol{a_o}$ and the solution $\boldsymbol{a}^*$ for some scalar values of σ_k as

$$\boldsymbol{a}^* - \boldsymbol{a_o} = \sigma_0 \boldsymbol{p_o} + \sigma_1 \boldsymbol{p_1} + \cdots + \sigma_{n-1} \boldsymbol{p_{n-1}}. \tag{4.8}$$

By multiplying (4.8) with $\boldsymbol{p_k}^{\boldsymbol{T}} A$ and using the property (4.5), we get

$$\sigma_k = \frac{\boldsymbol{p_k}^T G(\boldsymbol{a}^* - \boldsymbol{a_o})}{\boldsymbol{p_k}^T G \boldsymbol{p_k}}. \tag{4.9}$$

We now establish the result by showing that these coefficients σ_k coincide with the step lengths α_k generated by (4.7). If $\boldsymbol{a_k}$ is generated by algorithm (4.6), (4.7), then $\boldsymbol{a_k}$ is given by [46]

$$\boldsymbol{a_k} = \boldsymbol{a_o} + \sigma_0 \boldsymbol{p_o} + \sigma_1 \boldsymbol{p_1} + \cdots + \sigma_{k-1} \boldsymbol{p_{k-1}}. \tag{4.10}$$

Again by multiplying (4.10) with $\boldsymbol{p_k}^{\boldsymbol{T}} G$ and using the property (4.5), we get [46]

$$\boldsymbol{p_k}^T G(\boldsymbol{a_k} - \boldsymbol{a_o}) = 0, \tag{4.11}$$

and therefore,

$$\boldsymbol{p_k}^T G(\boldsymbol{a}^* - \boldsymbol{a_o}) = \boldsymbol{p_k}^T G(\boldsymbol{a}^* - \boldsymbol{a_k}) = \boldsymbol{p_k}^T G(\boldsymbol{a_k} - \boldsymbol{a_o}) = -\boldsymbol{p_k}^T r_k. \tag{4.12}$$

Comparing the equations (4.12), (4.7) and (4.9), we can say that $\sigma_k = \alpha_k$.

Theorem 2 [46]: *Let* $\boldsymbol{a_o} \in \mathbb{R}^n$ *be any starting point and suppose that the sequence* $\{\boldsymbol{a_k}\}$ *is generated by the conjugate direction algorithm (4.6), (4.7). Then*

$$r_k{}^T p_i = 0, for i = 0, 1, \ldots, k-1, \tag{4.13}$$

and $\boldsymbol{a_k}$ *is the minimizer of* $F(\boldsymbol{a}) = \frac{1}{2}\boldsymbol{a^T}G\boldsymbol{a} - \boldsymbol{h}\cdot\boldsymbol{a}$ *over the set*

$$\{\boldsymbol{a} \mid \boldsymbol{a} = \boldsymbol{a_0} + span\{\boldsymbol{a_0}, \boldsymbol{a_1}, \ldots, \boldsymbol{a_{k-1}}\}\} \tag{4.14}$$

Note that the notation $span\{a_0, a_1, \ldots, a_{k-1}\}$ is used to denote the set of all the linear combinations of the vectors $a_0, a_1, \ldots, a_{k-1}$.

In CG method, each direction $\boldsymbol{p_k}$ is chosen to be a linear combination of the negative residual $-\boldsymbol{r_k}$ and the previous direction p_{k-1}. We write

$$\boldsymbol{p_k} = -r_k + \beta_k \boldsymbol{p_{k-1}}, \tag{4.15}$$

where β_k is a scalar to be determined by the requirement that $\boldsymbol{p_{k-1}}$ and $\boldsymbol{p_k}$ must be conjugate with respect to G. By multiplying (4.15) by $\boldsymbol{p_{k-1}}^T G$ and imposing the condition $\boldsymbol{p_{k-1}}^T G \boldsymbol{p_k} = 0$, we find that

$$\beta_k = \frac{{r_k}^T G \boldsymbol{p_{k-1}}}{\boldsymbol{p_{k-1}}^T G \boldsymbol{p_k}} \tag{4.16}$$

Conjugate gradient preliminary version:

Given $\boldsymbol{a_0}$*;*

Set $r_0 \leftarrow G\boldsymbol{a_0} - \boldsymbol{h}$, $p_0 \leftarrow -r_0$, $k \leftarrow 0$*;*

while $r_k \neq 0$*;*

$\alpha_k \leftarrow -\frac{{r_k}^T p_k}{\boldsymbol{p_k}^T G \boldsymbol{p_k}}$*;* (4a)

$\boldsymbol{a_{k+1}} \leftarrow \boldsymbol{a_k} + \alpha_k \boldsymbol{p_k}$*;*

$r_{k+1} \leftarrow G\boldsymbol{a_{k+1}} - h$*;*

$\beta_{k+1} \leftarrow \frac{{r_{k+1}}^T G \boldsymbol{p_k}}{\boldsymbol{p_k}^T G \boldsymbol{p_k}}$*;*

$\boldsymbol{p_{k+1}} \leftarrow -\boldsymbol{r_{k+1}} + \beta_{k+1}\boldsymbol{p_k}$*;* (4b)

$k \leftarrow k + 1$*;*

end(while)

Theorem 3: *Suppose that the k^{th} iterate generated by the conjugate gradient method is not the solution point $\boldsymbol{a}^*$. The following four properties hold:*

$$r_k{}^T r_i = 0, \quad for\ i = 0, 1, \ldots, k-1, \tag{4.17}$$

$$span\{r_0, r_1, \ldots, r_k\} = span\{r_0, Gr_0, \ldots, G^k r_0\}, \tag{4.18}$$

$$span\{\boldsymbol{p_0}, \boldsymbol{p_1}, \ldots, \boldsymbol{p_k}\} = span\{r_0, Gr_0, \ldots, G^k r_0\}, \tag{4.19}$$

$$\boldsymbol{p_k}{}^T G \boldsymbol{p_i} = 0, \quad for\ i = 0, 1, \ldots, k-1. \tag{4.20}$$

Therefore, the sequence $\{\boldsymbol{a_k}\}$ converges to $\boldsymbol{a}^*$ in at most n iterations [46].

By using the theorem 2 and 3, we can derive a slightly more economical form of the CG method. First, we can use (4b) and (4.13) to replace the formula (4a) for α_k by

$$\alpha_k = \frac{r_k{}^T r_k}{\boldsymbol{p_k}{}^T G \boldsymbol{p_k}}. \tag{4.21}$$

Second, from the relations (4.4) and (4.6), it is possible to write $\alpha_k G \boldsymbol{p_k} = r_{k+1} - r_k$. So, by applying (4b) and (4.13) once again we can simplify the formula for $\beta_{\boldsymbol{k+1}}$ to

$$\beta_{k+1} = \frac{r_{k+1}{}^T r_{k+1}}{r_k{}^T r_k} \tag{4.22}$$

By using these formulae, we obtain the standard form of the conjugate gradient method given in the next subsection.

4.1.1 Conjugate gradient implementation

In this subsection, we provide the pseudo code [46] to implement for estimating the weights for linearly dependent data.

Given $\boldsymbol{a_o}$*;*

Set $r_0 \leftarrow G\boldsymbol{a_o} - \boldsymbol{h}$, $p_0 \leftarrow -r_0$, $k \leftarrow 0$*;*

while $r_k \neq 0$*;*

$\alpha_k \leftarrow \frac{{r_k}^T r_k}{\boldsymbol{p_k}^T G \boldsymbol{p_k}}$*;*

$\boldsymbol{a_{k+1}} \leftarrow \boldsymbol{a_k} + \alpha_k \boldsymbol{p_k}$*;*

$r_{k+1} \leftarrow r_k + \alpha_k G \boldsymbol{p_k}$*;*

$\beta_{k+1} \leftarrow \frac{{r_{k+1}}^T r_{k+1}}{{r_k}^T r_k}$*;*

$\boldsymbol{p_{k+1}} \leftarrow -\boldsymbol{r_{k+1}} + \beta_{\boldsymbol{k+1}} \boldsymbol{p_k}$*;*

$k \leftarrow k + 1$*;*

end(while)

4.2 Integral reinforcement learning

The Hamilton–Jacobi–Bellman (HJB) Equation is well known to solve for the optimal control solutions [16], [19]. Integral Reinforcement Learning (IRL) [16], [17], [48] uses an iterative technique to approximately solve the HJB equation and guarantees that the attained control will converge to the optimal solution. Therefore, let us begin this section with HJB associated with quadratic performance function for a continuous time LQR case. Later, we will present the policy iteration algorithm to acquire optimal control.

For the linear time invariant system (3.13), the quadratic cost function is given as,

$$J(t) = \frac{1}{2} \int_t^{\infty} (\mathbf{x}^{\mathrm{T}} \mathbf{Q} \mathbf{x} + \mathbf{u}^{\mathrm{T}} \mathbf{R} \mathbf{u}) dt. \tag{4.23}$$

The HJB equation is

$$-J_t = \min_{\mathbf{u}} \Big[{J_x}^T \dot{\mathbf{x}} + \frac{1}{2}\mathbf{x}^{\mathrm{T}}\mathbf{Q}\mathbf{x} + \frac{1}{2}\mathbf{u}^{\mathrm{T}}\mathbf{R}\mathbf{u} \Big], \tag{4.24}$$

$$-J_t = \min_{\mathbf{u}} \Big[{J_x}^T (\mathbf{A}\mathbf{x} + \mathbf{B}\mathbf{u}) + \frac{1}{2}\mathbf{x}^{\mathrm{T}}\mathbf{Q}\mathbf{x} + \frac{1}{2}\mathbf{u}^{\mathrm{T}}\mathbf{R}\mathbf{u} \Big], \tag{4.25}$$

where, J_t, and J_x are partial derivatives of $J(t)$ with respect to t, and $\mathbf{x}$.

Assuming a solution for $J(t)$ in quadratic form,

$$J(t) = \frac{1}{2}\mathbf{x}^{\mathrm{T}}\mathbf{S}(t)\mathbf{x}, \tag{4.26}$$

we can express the partial derivatives as,

$$J_x = \frac{\partial J(t)}{\partial \mathbf{x}} = \mathbf{S}\mathbf{x}, \tag{4.27}$$

$$J_t = \frac{\partial J(t)}{\partial t} = \frac{1}{2}\mathbf{x}^{\mathrm{T}}\dot{\mathbf{S}}(t)\mathbf{x}. \tag{4.28}$$

Differentiating the HJB equation (4.25) with respected to $\mathbf{u}$ to find the minimum,

$$\mathbf{B}^T J_x + \mathbf{R}\mathbf{u} = 0, \tag{4.29}$$

$$\mathbf{u} = -\mathbf{R}^{-1}\mathbf{B}^T J_x,$$

$$\mathbf{u} = -\mathbf{R}^{-1}\mathbf{B}^T\mathbf{S}\mathbf{x}. \tag{4.30}$$

Now, to solve the HJB at minimum $\mathbf{u}$, substitute (4.31) in (4.25).

$$-J_t = {J_x}^T(\mathbf{A}\mathbf{x} - \mathbf{B}\mathbf{R}^{-1}\mathbf{B}^T\mathbf{S}\mathbf{x}) + \frac{1}{2}\mathbf{x}^{\mathrm{T}}\mathbf{Q}\mathbf{x} + \frac{1}{2}\mathbf{x}^T\mathbf{S}\mathbf{B}\mathbf{R}^{-1}\mathbf{R}\mathbf{R}^{-1}\mathbf{B}^T\mathbf{S}\mathbf{x}, \tag{4.31}$$

Further, substituting (4.27) and (4.28), and solving,

$$-\frac{1}{2}\mathbf{x}^{\mathrm{T}}\dot{\mathbf{S}}\mathbf{x} = \mathbf{x}^T\mathbf{S}\big(\mathbf{A}\mathbf{x} - \mathbf{x}^T\mathbf{S}\mathbf{B}\mathbf{R}^{-1}\mathbf{B}^T\mathbf{S}\mathbf{x}\big) + \frac{1}{2}\mathbf{x}^{\mathrm{T}}\mathbf{Q}\mathbf{x} + \frac{1}{2}\mathbf{x}^T\mathbf{S}\mathbf{B}\mathbf{R}^{-1}\mathbf{R}\mathbf{R}^{-1}\mathbf{B}^T\mathbf{S}\mathbf{x},$$

$$-\frac{1}{2}\mathbf{x}^{\mathrm{T}}\dot{\mathbf{S}}\mathbf{x} = \mathbf{x}^T\mathbf{S}\mathbf{A}\mathbf{x} - \mathbf{B}\mathbf{R}^{-1}\mathbf{B}^T\mathbf{S}\mathbf{x} + \frac{1}{2}\mathbf{x}^{\mathrm{T}}\mathbf{Q}\mathbf{x} + \frac{1}{2}\mathbf{x}^T\mathbf{S}\mathbf{B}\mathbf{R}^{-1}\mathbf{B}^T\mathbf{S}\mathbf{x},$$

$$-\frac{1}{2}\dot{\mathbf{S}} = \mathbf{S}\mathbf{A} - \frac{1}{2}\mathbf{S}\mathbf{B}\mathbf{R}^{-1}\mathbf{B}^T\mathbf{S} + \frac{1}{2}\mathbf{Q},$$

$$-\dot{\mathbf{S}} = \mathbf{S}\mathbf{A} + \mathbf{A}^T\mathbf{S} - \mathbf{S}\mathbf{B}\mathbf{R}^{-1}\mathbf{B}^T\mathbf{S} + \mathbf{Q}. \tag{4.32}$$

It is to be noted that (4.32) is matrix Riccati equation same as (3.9) we derived in chapter 3. Therefore, the minimum $\mathbf{u}$ in (4.30) is the optimal control $\mathbf{u}^*$.

Now, let us represent r with the quadratic function

$$r(\mathbf{x}, \mathbf{u}) = \mathbf{x}^{\mathrm{T}}\mathbf{Q}\mathbf{x} + \mathbf{u}^{\mathrm{T}}\mathbf{R}\mathbf{u}. \tag{4.33}$$

Value function V can be expressed in terms of r as,

$$\begin{aligned} V(\mathbf{x}(t)) &= \int_t^{\infty} r(\mathbf{x}, \mathbf{u}) d\tau, \\ V(\mathbf{x}(t)) &= \int_t^{t+T} r(\mathbf{x}, \mathbf{u}) d\tau + \int_{t+T}^{\infty} r(\mathbf{x}, \mathbf{u}) d\tau, \\ V(\mathbf{x}(t)) &= \int_t^{t+T} r(\mathbf{x}, \mathbf{u}) d\tau + V(\mathbf{x}(t+T)). \end{aligned} \tag{4.34}$$

(4.34) is a continuous time Bellman equation [19]. The optimal value V^* is obtained on minimizing (4.34) subjected to control $\mathbf{u}$, given as,

$$\begin{aligned} V^*(\mathbf{x}(t)) &= \min_{\mathbf{u}} \Big[\int_t^{t+T} r(\mathbf{x}, \mathbf{u}) d\tau + V(\mathbf{x}(t+T)) \Big], \\ V^*(\mathbf{x}(t)) &= \min_{\mathbf{u}} \Big[\int_t^{t+T} r(\mathbf{x}, \mathbf{u}) d\tau \Big] + V^*(\mathbf{x}(t+T)). \end{aligned} \tag{4.35}$$

(4.35) is known as continuous time Bellman optimality equation [19]. Notice that in order to determine the optimal value at time t, one must know the optimal value at time $t + T$. Therefore, to solve for the optimal value, Bellman equation yields a backwards-in-time procedure [16], which is the base for Dynamic programming algorithms. The drawback is that these are off-line methods and the system dynamics are to be known.

In comparison to off-line methods, Policy Iteration (PI) method [Sutton and Barto 1998], [16],[19],[49] is adopted to solve for the optimal control by exploiting the Bellman equation (4.34) on-line real-time without the necessity to know the system dynamics. The PI algorithm consists of policy evaluation followed by policy improvement.

Policy Iteration:

step 0: Initialization

Select an admissible control policy π_0 which stabilizes the system.

step 1: Policy evaluation

For the iteration k, evaluate the value of the current policy using IRL Bellman equation

$$V_k = \int_t^{t+T} r(\mathbf{x}, \mathbf{u}) d\tau + V_k\big(\mathbf{x}(t+T)\big). \tag{4.36}$$

It is to be noted that there are no system dynamics involved in (4.36). This is same as solving the differential equivalent,

$$H(\mathbf{x}, \frac{\partial V}{\partial \mathbf{x}}, \mathbf{u}) = \dot{V} + r(\mathbf{x}, \mathbf{u}) = \Big(\frac{\partial V}{\partial \mathbf{x}}\Big)^T \dot{\mathbf{x}} + \mathbf{r}(\mathbf{x}, \mathbf{u}) = 0,$$

$$H(\mathbf{x}, \frac{\partial V}{\partial \mathbf{x}}, \mathbf{u}) = \Big(\frac{\partial V}{\partial \mathbf{x}}\Big)^T \big(\mathbf{A}\mathbf{x} + \mathbf{B}\mathbf{u}\big) + \mathbf{r}(\mathbf{x}, \mathbf{u}) = 0 \tag{4.37}$$

(4.37) is Bellman equation in terms of Hamilton function, which is dependent on the system dynamics $\mathbf{A}, \mathbf{B}$.

step 2: Policy Improvement

Update to an improved policy using

$$\pi_{k+1}(x) = -\mathbf{R}^{-1}\mathbf{B}^T \frac{\partial V_k}{\partial \mathbf{x}}. \tag{4.38}$$

From (4.27), we can rewrite the control policy as

$$\pi_{k+1}(x) = -\mathbf{R}^{-1}\mathbf{B}^T \mathbf{S_k} \mathbf{x}. \tag{4.39}$$

step 3: Check for convergence

Go to step 1 if not converged.

The IRL Bellman equation (4.36) is preferred to solve by using value function approximation [48]. In the value function approximation approach, according to the

Weierstrass high-order approximation theorem [48], value function is approximated to a single-layer NN by the Weierstrass approximator network. However, we will stick to solve the differential Riccati equation to evaluate the value function with the help of system identification using CG.

4.3 Optimal gain calculation through IRL and CG identified system

In this section, we develop a procedure to find an optimal control solution using the CG while performing policy iteration. In the policy evaluation step, the value function (4.36) is approximated to a neural network to determine the value of the current policy, rather than using the Bellman equation [16], [19]. However, instead of using value function approximation, we adopt CG mentioned from section 4.1 to estimate the system dynamics.

In order to perform linear regression using CG, it is essential to represent the time-invariant system (3.1) in a linear equation format as in (4.2). Then, (3.1) can be expressed as

$$\dot{\mathbf{x}} = \left[\begin{array}{c:c} \mathbf{A} & \mathbf{B} \end{array}\right] \left[\begin{array}{c} \mathbf{x} \\ \hdashline \mathbf{u} \end{array}\right], \tag{4.40}$$

$$\dot{\mathbf{x}} = W\boldsymbol{\delta}. \tag{4.41}$$

The CG algorithm takes in samples of $\dot{\mathbf{x}}$, and $\boldsymbol{\delta}$ to estimate W. Mathematically, for some function say f_{cg},

$$\mathbf{A_{est}}, \mathbf{B_{est}} = f_{cg}(\mathbf{x}, \mathbf{u}, \dot{\mathbf{x}}). \tag{4.42}$$

However, the control input $\mathbf{u}$ is somehow always related to the current states and/or the error. For LQI, this depends on $\bar{\mathbf{x}}$, and the relation is given by (3.16).

Therefore,

$$\mathbf{A_{est}}, \mathbf{B_{est}} = f_{cg}(\mathbf{x}, -\mathbf{K}\bar{\mathbf{x}}, \dot{\mathbf{x}}),$$
$$\mathbf{A_{est}}, \mathbf{B_{est}} = g_{cg}(\bar{\mathbf{x}}, \dot{\mathbf{x}})\Big|_{\mathbf{K}}, \tag{4.43}$$

which essentially says that the pair $\mathbf{A_{est}}$ and $\mathbf{B_{est}}$ is a function, say g_{cg}, of $\bar{\mathbf{x}}$, $\dot{\mathbf{x}}$ at a given gain $\mathbf{K}$.

To support the (4.43), we can redefine the system (3.1) in linear form as

$$\dot{\mathbf{x}} = \left[\begin{array}{c:c} \mathbf{A} - \mathbf{BK_1} & -\mathbf{BK_2} \end{array}\right] \bar{\mathbf{x}}, \tag{4.44}$$

where

$$\left[\begin{array}{c:c} \mathbf{K_1} & \mathbf{K_2} \end{array}\right] = \mathbf{K}. \tag{4.45}$$

To conclude, either depending on (4.40) or (4.43) for system estimation yield the same results. Note that (4.43) is independent of the control input $\mathbf{u}$.

The $\mathbf{B_{est}}$ obtained from CG is then used to determine the control policy (4.39) for policy improvement. Therefore, the improved control is

$$\pi(\mathbf{x}) = -\mathbf{R}^{-1}\mathbf{B_{est}}^T\mathbf{S}\mathbf{x}. \tag{4.46}$$

Policy iteration with system estimation:

step 0: Initialization

Select an admissible gain $\mathbf{K_0} = \mathbf{K_{ad}}$ which stabilizes the system. The control corresponding to $\mathbf{K_0}$ is π_0.

Choose $\mathbf{Q}$ and $\mathbf{R}$.

step 1: Policy Evaluation

For the iteration k, evaluate the value of the current policy from

$$V_k = \mathbf{x}^T \mathbf{S}_k \mathbf{x}, \tag{4.47}$$

by using the differential Riccati equation,

$$-\dot{\mathbf{S}}_k = \mathbf{S}_k \mathbf{A}_k + \mathbf{A}_k{}^T \mathbf{S}_k - \mathbf{S}_k \mathbf{B}_k \mathbf{R}^{-1} \mathbf{B}_k{}^T \mathbf{S}_k + \mathbf{Q}. \tag{4.48}$$

The matrices $\mathbf{A}_k$ and $\mathbf{B}_k$ are determined from the system estimation using CG by analysing the data samples of $\mathbf{x}$, $\mathbf{u}$, and $\dot{\mathbf{x}}$, collected after running the system with control $\pi_{k-1}(\mathbf{x})$.

$$\mathbf{A_k}, \mathbf{B_k} \equiv \mathbf{A_{est}}, \mathbf{B_{est}} = f_{cg}(\mathbf{x}, \pi_{\mathbf{k-1}}, \dot{\mathbf{x}}). \tag{4.49}$$

step 2: Policy Improvement

Update to an improved policy by using

$$\pi_{k+1}(\mathbf{x}) = -\mathbf{R}^{-1} \mathbf{B_k}{}^T \nabla V_k = -\mathbf{R}^{-1} \mathbf{B_k}{}^T \mathbf{S}_k \mathbf{x}. \tag{4.50}$$

step 3: Check for convergence

If $\big| ||\mathbf{K}_k|| - ||\mathbf{K}_{k-1}|| \big| < \epsilon$ stop, else go to *step 1.*

ϵ is the threshold limit of check the convergence. Note that $\mathbf{K}_k = \mathbf{R}^{-1} \mathbf{B_k}{}^T \mathbf{S}_k$

CHAPTER 5

DESIGN AND SIMULATION RESULTS

In this chapter, the algorithm proposed in chapter 4 will be tested in MATLAB and Simulink. Starting with an admissible gain, we run the algorithm for several iterations until we reach an optimal control value. At the optimal control value, the performance of the estimated system matches the theoretical model. We begin showing the theoretical model of the quadrotor, and the UGV in section 5.1, and in section 5.2 respectively. Simulation results at different iterations are shared in section 5.3. In section 5.4, we give the comparison between the estimated model and theoretical model.

5.1 Theoretical model of quadrotor

The theoretical model in the simulation environment presented in this section is based on commercially available Parrot AR.Drone 2.0. Hence, all the quadrotor parameters mentioned in chapter 2 are assigned with the values corresponding to AR.Drone 2.0 as given in table 5.1.

Table 5.1. AR.Drone 2.0 parameters

Parameter	symbol	value
Mass	m	$0.46kg$
Moment of inertia long X-axis	I_{xx}	$0.0024kg/m^2$
Moment of inertia long Y-axis	I_{yy}	$0.0024kg/m^2$
Moment of inertia long Z-axis	I_{zz}	$0.004803741kg/m^2$
Acceleration due to gravity	g	$9.81m/s^2$

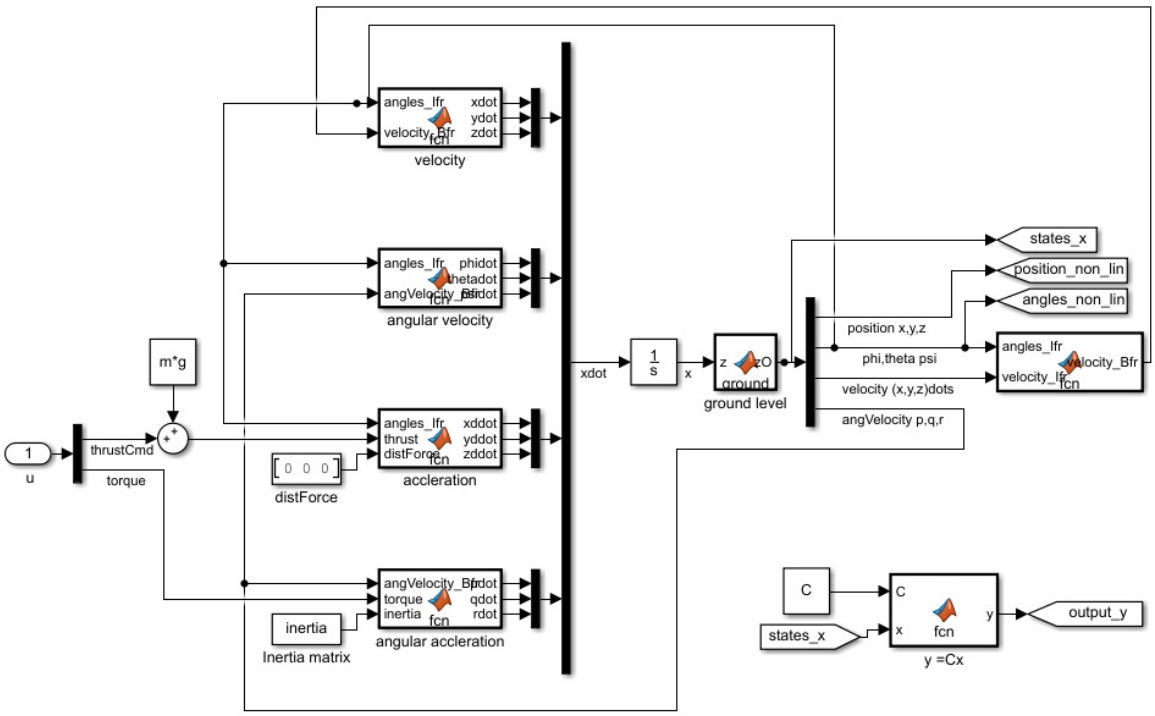


Figure 5.1. Theoretical model of quadrotor in Simulink.

The implementation of the mathematical model we discussed earlier in chapter 2 in Simulink is shown in Fig 5.1. The feedback portion of the LQI from chapter 3 with error being integrated is shown in Fig 5.2.

5.2 Theoretical model of UGV

The theoretical model in the simulation environment presented in this section is based on Husky from Clear Path Robotics. Therefore, we use the parameters mentioned in chapter 2 associated with Husky, given in table 5.2.

Fig 5.3 shows the implementation of the mathematical model we discussed earlier in chapter 2 in Simulink environment. The system inputs for the Husky are linear velocity in body frame, v_{C_B}, and yaw rate, $dot\psi$. Therefore, we modify the

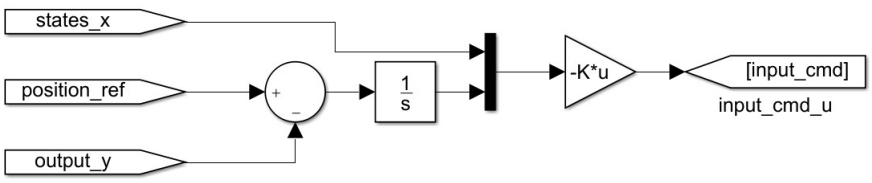


Figure 5.2. LQI feedback in Simulink.

Table 5.2. AR.Drone 2.0 parameters

Parameter	symbol	value
Lateral wheel base	b_l	$0.555m$
Radius of left wheel	r_L	$0.1524m$
Radius of right wheel	r_R	$0.1524m$

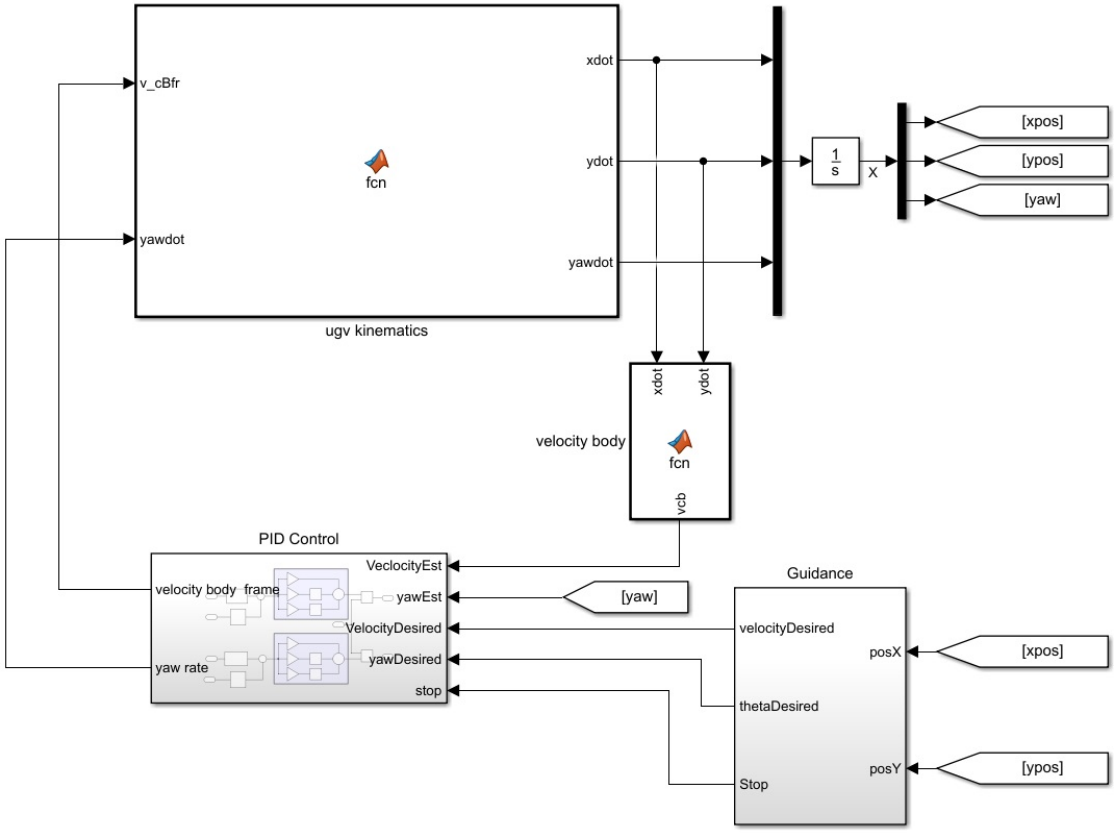


Figure 5.3. Theoretical model of UGV in Simulink.

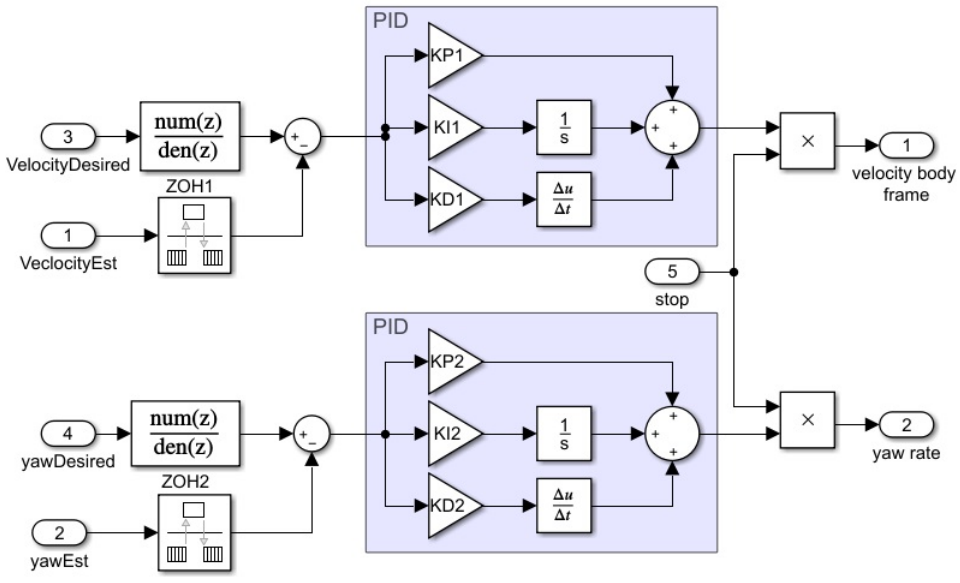


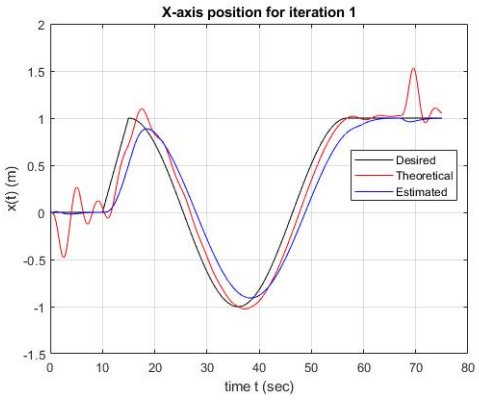
Figure 5.4. PID controller in Simulink.

kinematics accordingly from the equations (2.36) and (2.40) to make the theoretical model resemble the actual system. Consequently, we use a pair of PID controllers, each for every input, as shown in Fig 5.4.

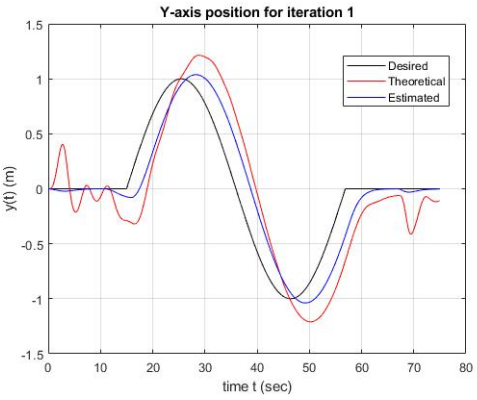
5.3 Simulation results

In this section, we show the simulation results of the algorithm proposed in section 4.2 tested vigorously on different trajectories. We begin this section by showing the time responses of the output variables, x, y, z, ψ during circular and 8-figure trajectories. Later, we show the ability of the algorithm in tracking circular and 8-figure trajectories. Please note that the results shared here are obtained by running the algorithm in section 4.2 for five iterations.

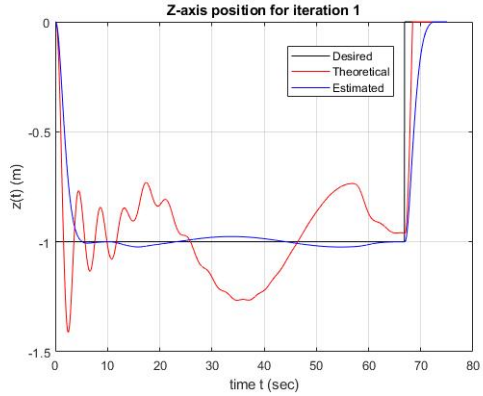
Figure 5.5 and 5.6 shows the responses during the first iteration of circular and 8-figure trajectories, respectively. Note that the theoretical responses corresponding to figure 5.5 and 5.6 are acquired through the admissible gain. It is evident to say



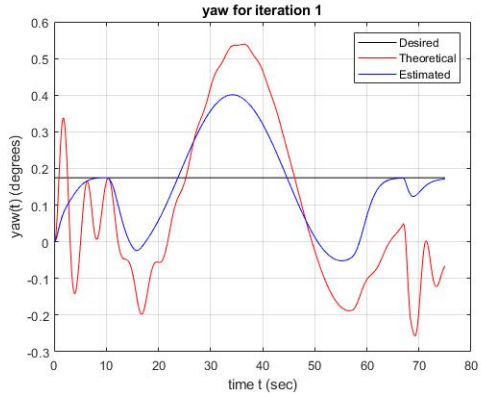
(a) $x(t)$ response during 1^{st} iteration



(b) $y(t)$ response during 1^{st} iteration



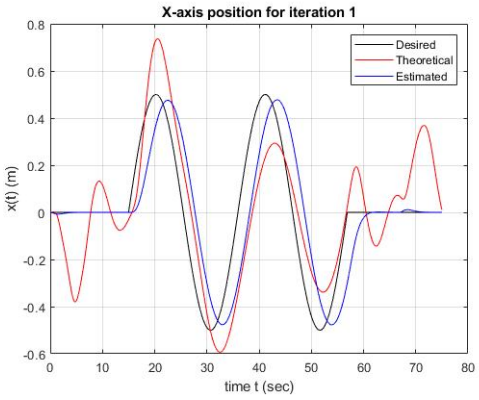
(c) $z(t)$ response during 1^{st} iteration



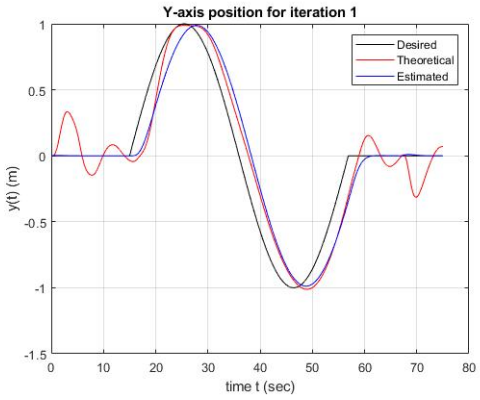
(d) $\psi(t)$ response during 1^{st} iteration

Figure 5.5. Response of x, y, z, ψ during first iteration of circular trajectory.

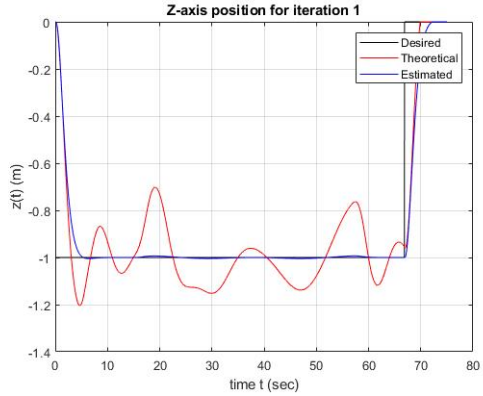
that, these theoretical responses have unsatisfactory tracking ability. However, we see a significant improvement in tracking from the estimated model, despite being the first iteration.



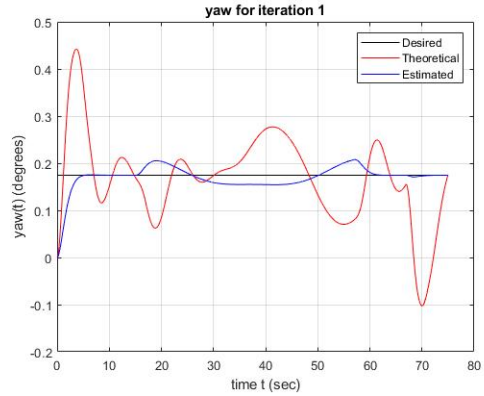
(a) $x(t)$ response during 1^{sh} iteration



(b) $y(t)$ response during 1^{st} iteration

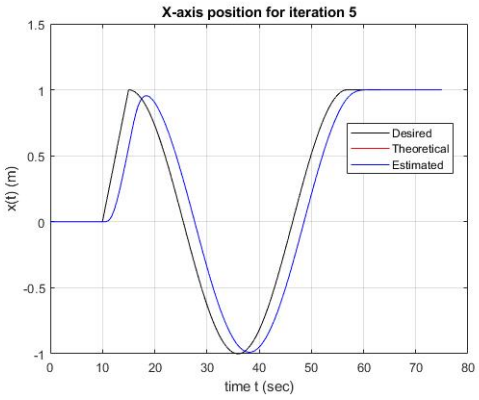


(c) $z(t)$ response during 1^{st} iteration

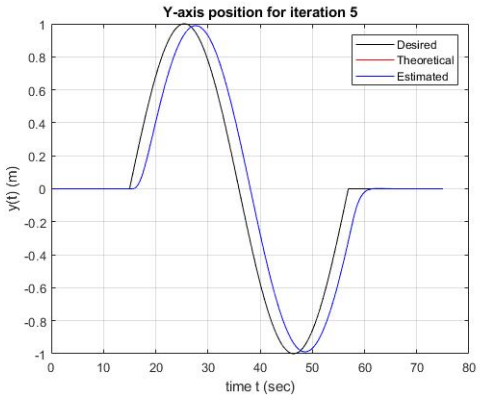


(d) $\psi(t)$ response during 1^{st} iteration

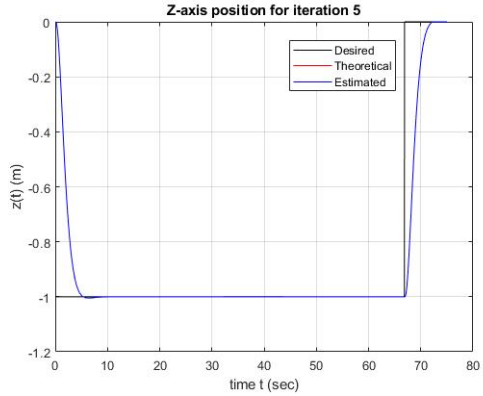
Figure 5.6. Response of x, y, z, ψ during first iteration of 8-figure trajectory.



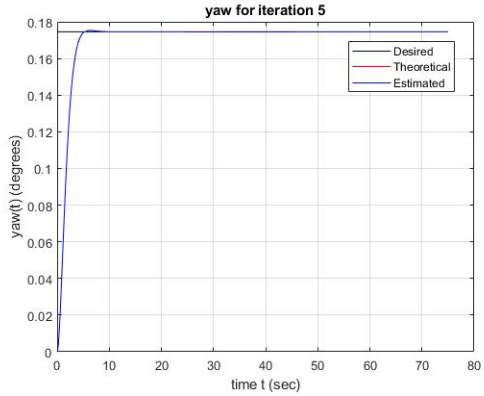
(a) $x(t)$ response during 5^{th} iteration



(b) $y(t)$ response during 5^{th} iteration



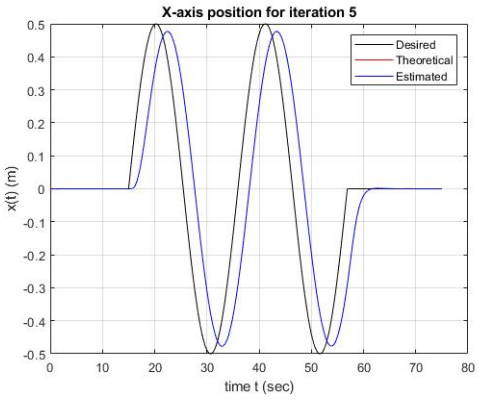
(c) $z(t)$ response during 5^{th} iteration



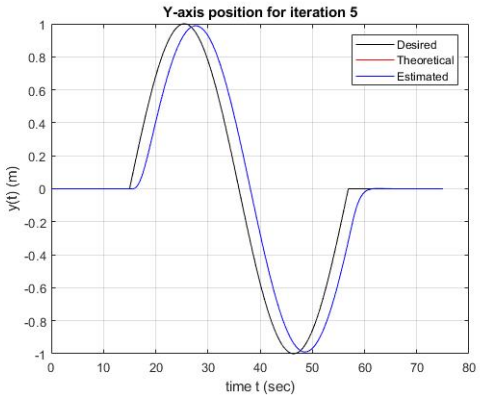
(d) $\psi(t)$ response during 5^{th} iteration

Figure 5.7. Response of x, y, z, ψ during fifth iteration of circular trajectory.

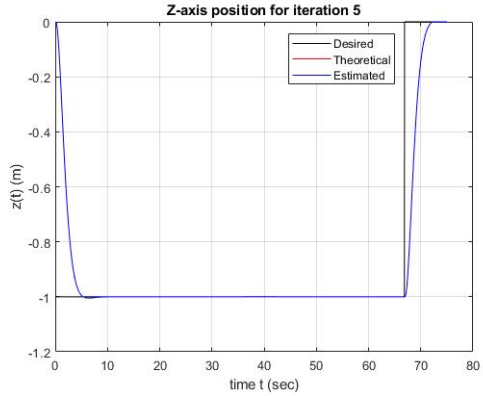
Figures 5.7 and 5.8 shows the responses of circular and 8-figure trajectories respectively, during the fifth iteration. It is apparent to notice that the theoretical and the estimated model deliver the same performance.



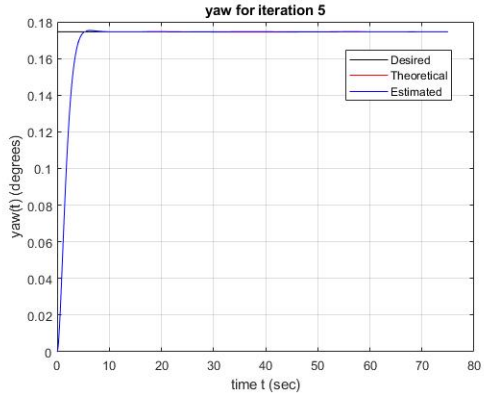
(a) $x(t)$ response during 5^{th} iteration



(b) $y(t)$ response during 5^{th} iteration



(c) $z(t)$ response during 5^{th} iteration



(d) $\psi(t)$ response during 5^{th} iteration

Figure 5.8. Response of x, y, z, ψ during fifth iteration of 8-figure trajectory.

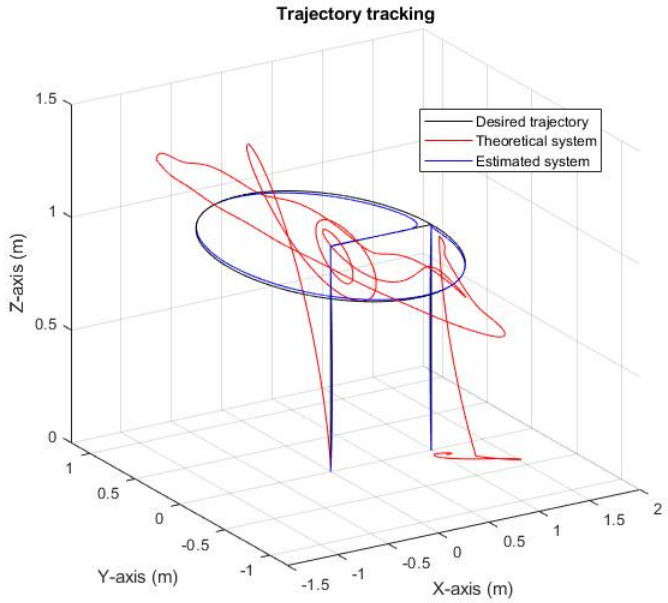


Figure 5.9. Circular trajectory during first iteration.

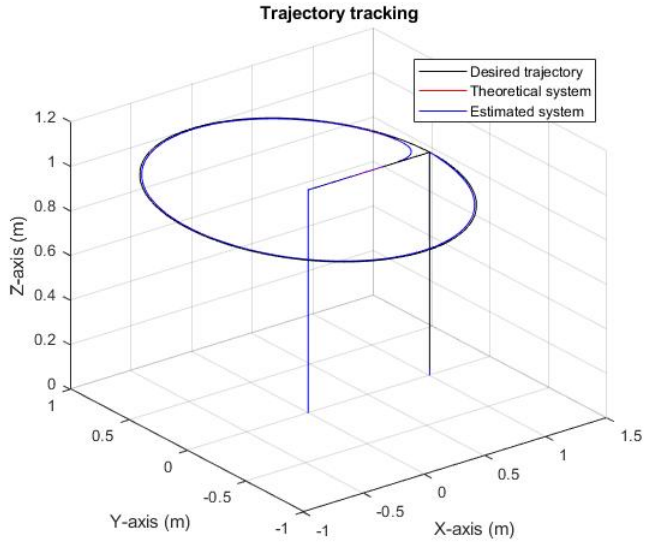


Figure 5.10. Circular trajectory during last iteration.

To check the trajectory tracking ability, we can look into figures 5.9 - 5.12. Figures 5.9 and 5.10 show the circular trajectory in 3D space during the first and last

iterations, respectively. Figures 5.11 and 5.12 show the 8-figure trajectory during the first and last iterations, respectively.

Note that even though the trajectory obtained from the theoretical model during the first iterations is abysmal, the chosen admissible gain keeps the system stable and helps in linearizing the system.

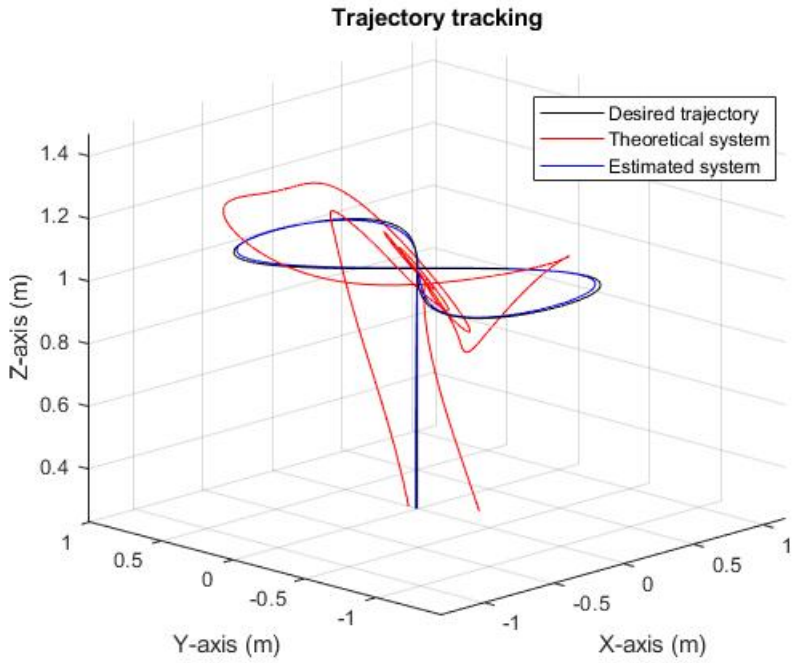


Figure 5.11. 8-fig trajectory during first iteration.

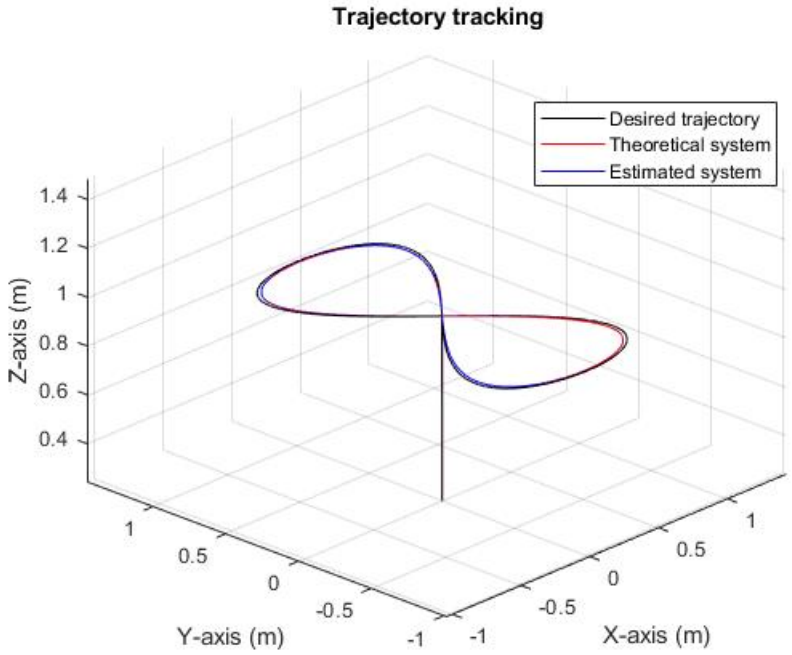


Figure 5.12. 8-fig trajectory during last iteration.

5.4 Theoretical vs estimated system

In this section, we reveal the estimated linear model obtained finally from the algorithm 4.2 and compare it with the linear model we derived in section 2.7. Then we show the gain K calculated by the estimated model.

We draw the **A** and **B** matrices of the state-space model (3.1) by substituting the parameters from table 5.1 in (2.31). Thus we obtain the system matrices

$$\mathbf{A} = \begin{bmatrix} 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 0 & -9.81 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 9.81 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \end{bmatrix}, \tag{5.1}$$

and

$$\mathbf{B} = \begin{bmatrix} 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ -2.17 & 0 & 0 & 0 \\ 0 & 416.6 & 0 & 0 \\ 0 & 0 & 416.6 & 0 \\ 0 & 0 & 0 & 208.1 \end{bmatrix}. \tag{5.2}$$

The estimated linear model we obtain from the simulation are given by $\mathbf{A_{est}}$ and $\mathbf{B_{est}}$ as

$$\mathbf{A_{est}} = \begin{bmatrix} 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & -1.7 & -9.81 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 9.81 & -1.7 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & -0.01 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \end{bmatrix}, \tag{5.3}$$

$$\mathbf{B_{est}} = \begin{bmatrix} 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & -0.22 & -0.01 & 0 \\ 0 & -0.02 & 0.02 & 0.06 \\ 0 & 0 & 0 & 0.03 \\ -2.17 & -2.77 & 1.56 & 0.01 \\ 0 & 416.6 & 0 & 0 \\ 0 & 0 & 416.6 & 0 \\ 0 & 0 & 0 & 208.1 \end{bmatrix}. \tag{5.4}$$

Note that the the simulation results of the estimated model we have been sharing in the section 5.3 are from the matrices $\mathbf{A_{est}}$ and $\mathbf{B_{est}}$. This model in Simulink environment is shown in the figure 5.13

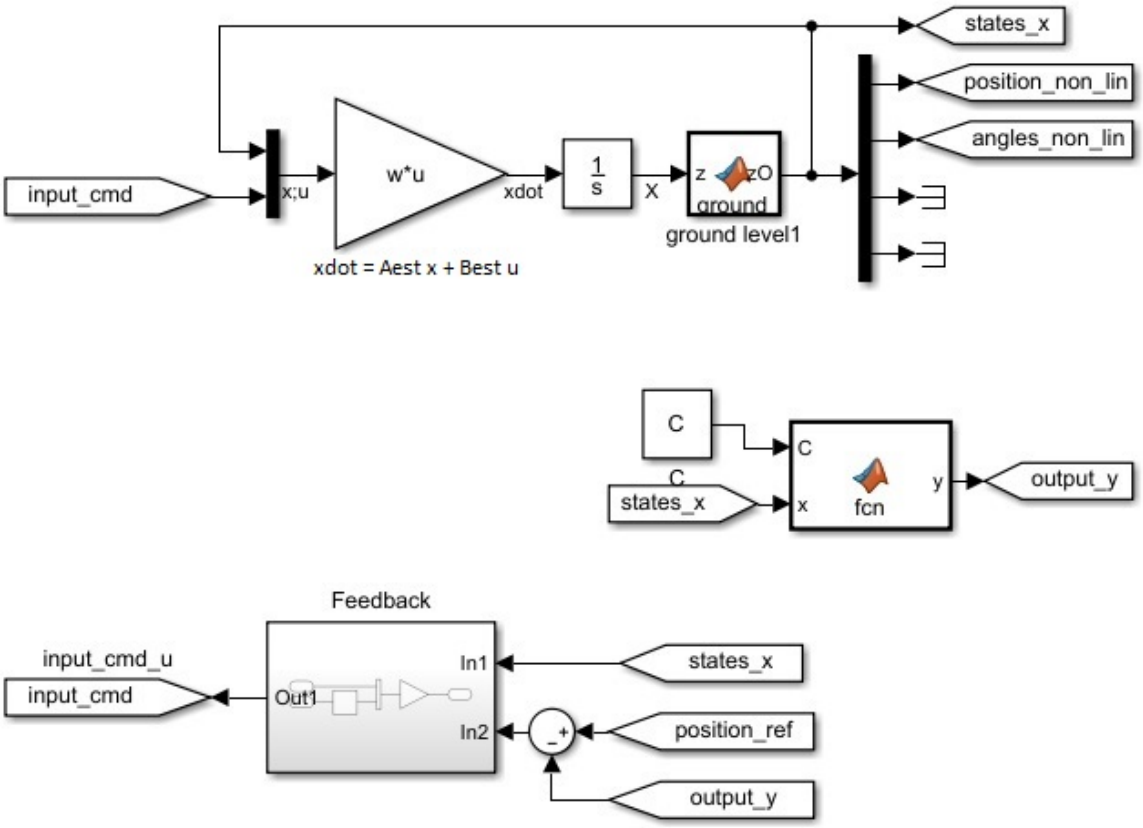


Figure 5.13. Estimated model of quadrotor in Simulink.

The gain $\mathbf{K}$ obtained by solving the Riccati equation (3.9) by using $\mathbf{A}$, $\mathbf{B}$ for the augmented matrices (3.19), as discussed in section 3.1, yields to the optimal control. We denote this gain by $\mathbf{K_{opt}}$. Upon solving for this gain in MATLAB, we have,

$$\mathbf{K_{opt}} = \begin{bmatrix} 0 & 0 & -4.28 & 0 & 0 & 0 & 0 & 0 & -2.99 & 0 & 0 & 0 & 0 & 0 & -2.23 & 0 \\ 0 & -4.88 & 0 & 13.8 & 0 & 0 & 0 & 4.2 & 0 & 2.25 & 0 & 0 & 0 & -2.2 & 0 & 0 \\ -4.88 & 0 & 0 & 0 & 13.8 & 0 & -4.2 & 0 & 0 & 0 & 2.25 & 0 & -2.23 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 3.87 & 0 & 0 & 0 & 0 & 0 & 2.24 & 0 & 0 & 0 & -2.23 \end{bmatrix}.$$

The gain obtained by solving the Riccati equation (3.9) by using $\mathbf{A_{est}}$, $\mathbf{B_{est}}$ for the augmented matrices (3.19) is given by

$$\mathbf{K_{est}} = \begin{bmatrix} 0 & 0.01 & -4.28 & 0.08 & 0 & 0 & 0 & 0 & -2.99 & -0.01 & 0.01 & 0 & 0 & 0 & -2.23 & 0 \\ -0.8 & 4.88 & -0.01 & 13.8 & 0 & 0 & -0.7 & 4.14 & 0 & 2.25 & 0 & 0 & 0.38 & -2.2 & 0 & 0 \\ -4.8 & -0.8 & 0 & 0 & 13.8 & 0 & -4.1 & -0.7 & 0 & 0 & 2.25 & 0 & -2.2 & 0.38 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 3.87 & 0 & 0 & 0 & 0 & 0 & 2.24 & 0 & 0 & 0 & -2.23 \end{bmatrix}.$$

To compare $\mathbf{K_{est}}$ with the initial admissible gain $\mathbf{K_{ad}}$, we have considered $\mathbf{K_{ad}}$ to be,

$$\mathbf{K_{ad}} = \begin{bmatrix} 0.78 & 0.45 & -3.69 & 1.25 & 1.15 & 0.27 & 1.13 & 0.13 & -2.79 & 1.32 & 1.28 & 0.87 & 0.31 & 1.22 & 3.20 & 0.01 \\ 0.09 & 6.02 & 1.25 & 14.3 & 1.20 & 0.26 & 0.01 & 4.31 & 1.23 & 2.30 & 0.00 & 0.16 & 1.34 & -1.26 & 0.74 & 1.07 \\ -4.53 & 1.19 & 0.64 & 0.27 & 14.3 & 0.03 & -3.67 & 0.78 & 0.12 & 1.39 & 2.46 & 1.46 & 3.58 & 0.60 & 0.01 & 1.39 \\ 0.72 & 0.12 & 0.41 & 0.64 & 0.85 & 4.57 & 0.20 & 0.91 & 0.04 & 1.16 & 1.29 & 3.44 & 0.70 & 1.30 & 0.17 & 0.17 \end{bmatrix}.$$

CHAPTER 6

EXPERIMENTAL DESIGN AND IMPLEMENTATION

In this chapter, we provide the details of the experimental setup, followed by the implementation results. All the implementations are performed with the Robot Operating System (ROS) as in [50]. ROS is a flexible framework with a set of tools, libraries, and conventions for designing robot software. We have generated standalone ROS nodes from the Simulink models to run on a Ubuntu Linux system. Please note that we have not included the installation and setup of ROS. However, we have briefly provided the details of the components we use and their integration with ROS. Section 6.1 explains the Leap motion controller. In section 6.2, we explain Vicon motion capturing. In section 6.3, we show the implementation of autonomous tracking and landing of a quadrotor on moving UGV.

6.1 Leap motion controller

The Leap Motion Controller is a 3D optical hand tracking device that tracks the hand's motions up to 200 fps [51], with incomparable accuracy and precision. It consists of a pair of stereo cameras with a typical field of view of $140\times120^{\circ}$, accompanying three infrared LEDs to draw a robust and reliable skeletal hand motion model. Many hand gesture recognition techniques and applications rely on this skeletal model. [52] and [53] analyze the hand motion data using the back-propagation and Deep Neural Networks, respectively, for gesture recognition. [51] proposed the design of an anthropomorphic robot hand using twenty servo motor for twenty degrees of freedom to reproduce the exact and every gesture of our hand. [54] studies

controlling the SCARA robotic arm using leap motion. In this section, we explain the usage of leap motion, relying on the skeletal model, to control and drive the UGV, Husky.

The location and orientation information of the palm is collected from the skeletal hand motion data by deploying a ROS node on Linux system after connecting the leap motion controller physically to the system. Depending on the location and orientation we generate the commands (2.40) and (2.41) accordingly to drive the Husky. (ROS driver for Leap motion sensor, "leap_motion", [55] needs to be installed in order to access the data, and can be available at the Git repository [56]).

The data collected from the leap motion controller will be with respect to the body frame coordinates as shown in figure 6.1.

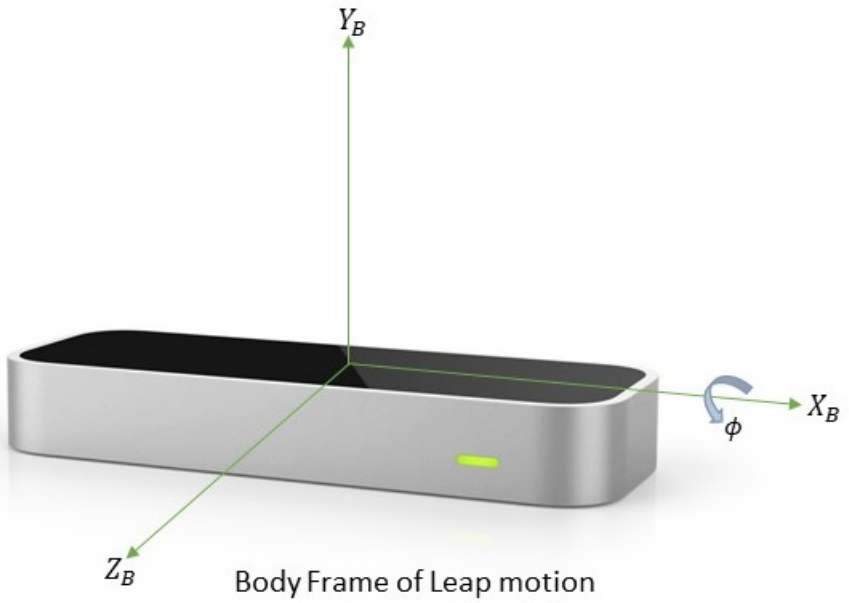


Figure 6.1. Leap motion controller.

The direction of motion of the UGV, Husky, is determined based on the location of the palm in X_BZ_B plane. Figure 6.2 shows the sectors corresponding to the direction of motion. The velocity and acceleration are determined based on the orientation, which is the pitch of the palm, as shown in table 6.1.

Table 6.1. Acceleration control using leap motion

Pitch angle of palm (ϕ)	Acceleration
$\phi < -25°$	acceleration > 0
$-25° < \phi < 20°$	acceleration $= 0$
$\phi > 20°$	acceleration < 0

6.2 Vicon motion capture system

Vicon motion capturing system provides the most precise and reliable data in tracking the movement of objects by employing passive optical motion capture technology [57], [58]. The optical-passive technique utilizes retroreflective markers [59] that are tracked by infrared cameras. The accurate spatial location information is obtained from the Vicon camera system by capturing the position of the markers affixed to the objects. Majority of the indoor fight demonstrations [11], [44], [60], [61], [62], [58] rely on Vicon system for positioning. In this section, we briefly discuss our approach in integrating Vicon for the implementations.

Figures 6.4 and 6.5 shows Vicon markers glued to the A.R Drone and Husky, respectively. The real-time data of A.R. Drone and Husky captured by the Vicon system through these markers is transmitted to a terminal, then to the ground station computer [58] (Vicon Master Computer) through an ethernet cable. Finally, this data is transmitted to the Linux machine installed with "Vicon_bridge" ROS driver package

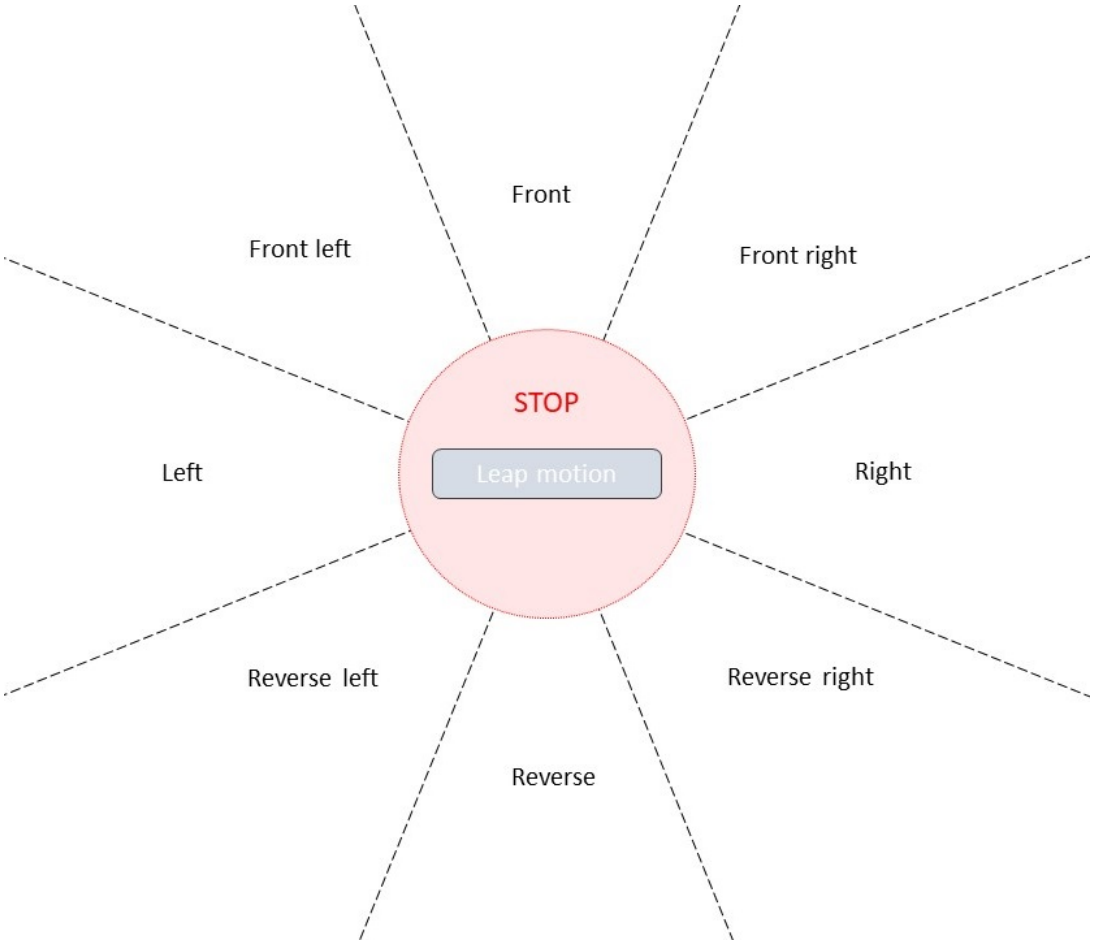


Figure 6.2. Direction of motion based on palm location with respect to Leap motion.

which runs the ROS node. (ROS driver for Vicon system, "Vicon_bridge", [63] needs to be installed in order to access the data, and can be available at the Git repository [64]). The position and orientation information is published by this ROS node.

6.3 Autonomous tracking and landing of quadrotor on a moving UGV

In this section, we explain the experimental framework and the demonstrations conducted based on the simulations performed in the indoor lab environment at UTA Research Institute. We are relying on the Vicon motion capture system, with sixteen Vicon cameras equipped in the lab, to locate the unmanned vehicles in inertial frame.

Figure 6.3. A.R Drone 2.0 with Vicon markers.

Figure 6.4. Husky with Vicon markers.

Note that the ROS packages and the installation instructions for Husky, and AR.Drone are available at [65], [66] and [67], [68] respectively. These packages help

us to communicate with Husky and AR.Drone through ROS topics to subscribe to the command inputs.

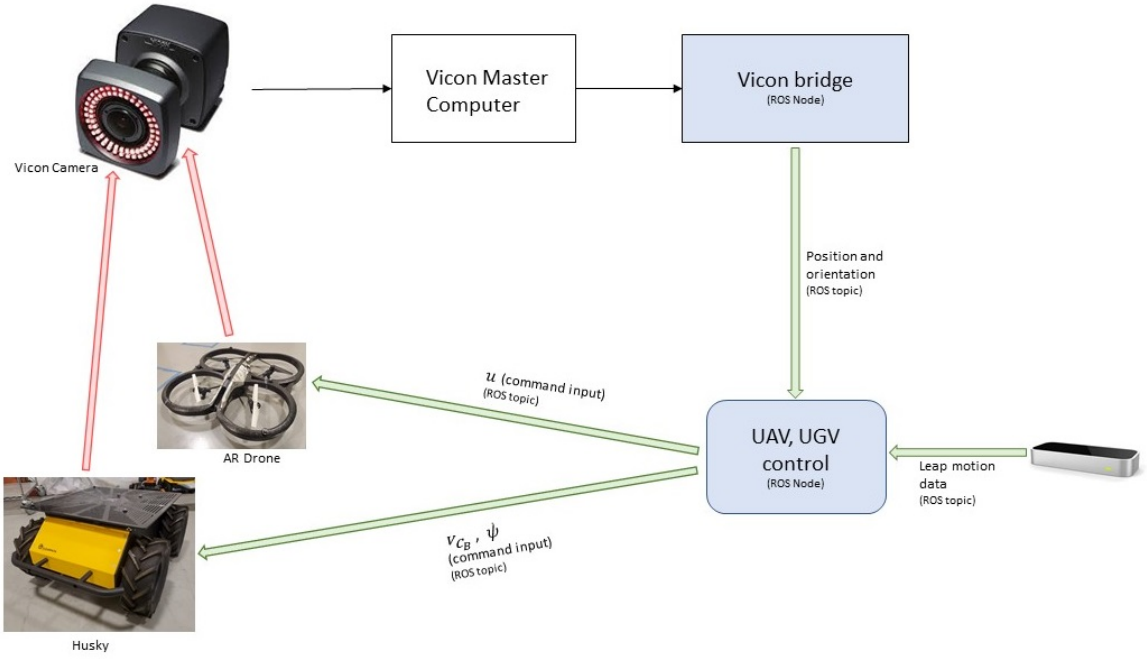


Figure 6.5. Communication flow in the experiments.

The communication flow between the elements is shown in figure 6.5. The ROS topics containing the position and location information of the Husky, and AR.Drone is subscribed by the ROS nodes UAV, UGV control for the feedback purpose. The input commands to the UGV are published by the UGV control node based on the palm information topic, subscribed from leap motion. While UGV moves in an arbitrary trajectory, UAV is made to track the UGV. Therefore, the UAV control node subscribes to the UGV location topic from Vicon to publish the input commands to the UAV.

The Simulink models shown in chapter 5 are converted into standalone ROS nodes using ROS toolbox in MATLAB/Simulink software. We have written Linux shell script to launch and run these ROS nodes. Note that the cooperation control of UAV and UGV is based on [44].

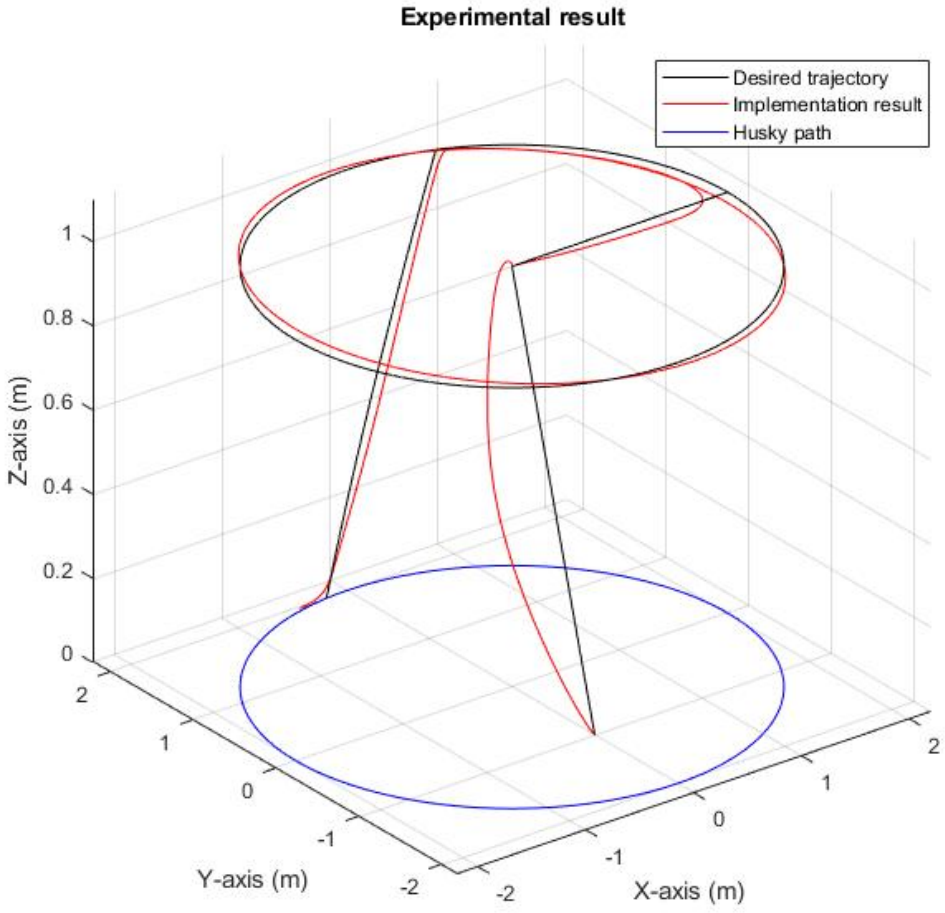


Figure 6.6. Implementation results.

The tracking performance with the optimal control policy for the practical implementation can be inferred from the figure 6.6, while Husky is moving in a

circular trajectory. Figure 6.7 shows the cooperation control of unmanned vehicles in the lab. The experiments described in this chapter are recorded to illustrate visually. The videos are available from the following links: https://youtu.be/Hx7D5e7yBB4, https://youtu.be/e8AslCdRxfg, and
https://youtu.be/IpYZVY6k34k.

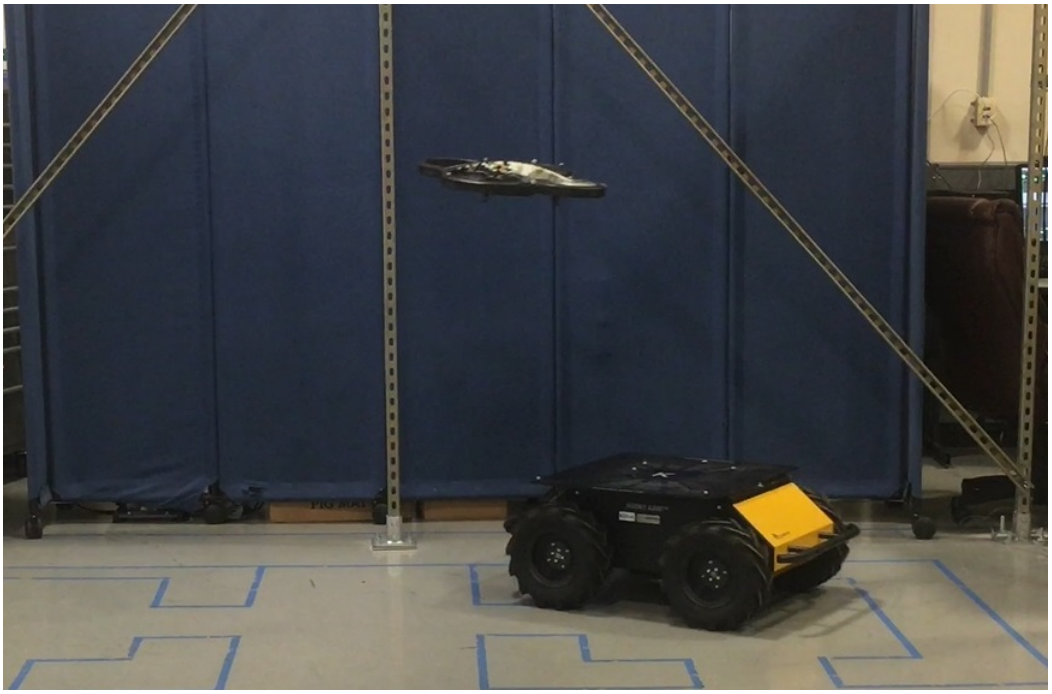


Figure 6.7. AR.Drone following Husky.

CHAPTER 7

CONCLUSION

In this thesis, we have presented the approach for calculating the optimal control of a UAV to track and land on a moving UGV. The significance of this algorithm relies on the ability to determine optimal gain in the absence of system dynamics. This method deploys solving the IRL Bellman equation through system identification instead of value function approximation. Through rigorous experimentation, we showed that when started with an admissible control policy, this algorithm delivers an efficient control over each iteration. Further, to strengthen this algorithm, the efforts in implementing it on AR Drone helped in tracking and landing on the arbitrarily moving Husky. The UAV system can be approximated to a linear time-invariant system. The scope of future research will be in the field of exploring control strategies for nonlinear systems. Furthermore, research can be conducted to examine the performance of this approach to nonlinear systems.

APPENDIX A

NOMENCLATURE

$\hat{i_e}$, $\hat{j_e}$, $\hat{k_e}$: unit vectors in Earth frame

$\hat{i_B}$, $\hat{j_B}$, $\hat{k_B}$: unit vectors in Body frame

ϕ, θ, ψ : Euler angles roll, pitch and yaw

$\mathbf{R_x}(\phi)$, $\mathbf{R_y}(\theta)$, $\mathbf{R_z}(\psi)$: rotation matrices along X, Y, Z axes

$\mathbf{R}$: rotation matrix from Body to Earth frame

$\mathbf{T}$: transform angular velocities from the Body to Earth frame

x : current position in X-axis in Earth frame

y : current position in Y-axis in Earth frame

z : current position in Z-axis in Earth frame

u : forward velocity in Body frame

v : sideward velocity in Body frame

w : vertical velocity in Body frame

p : roll rate

q : pitch rate

r : yaw rate

$\mathbf{v_B}$: linear velocity vector in Body frame

$\mathbf{v}$: linear velocity vector in Earth frame

$\omega_\mathbf{B}$: angular velocity vector in Body frame

ω : angular velocity vector in Earth frame $\mathbf{F_B}$: force vector in Body frame

$\mathbf{m_B}$: moment vector in Body frame

$\mathbf{I}$: inertia matrix

F_t : thrust generated by the rotors along Z_B-axis

$\mathbf{F_w}$: force due to wind disturbances in Body frame

$\tau_\mathbf{B}$: control torque produced by the rotors in body frame

$\tau_\mathbf{w}$: torque due to wind disturbance in Body frame

m : mass of the quadrotor

g : acceleration due to gravity

Ω_1, Ω_2, Ω_3, Ω_4 : rotational speeds of the four rotors

b : thrust factor

d : drag factor

l : distance from the center to the rotor

$\mathbf{x}$: state vector

$\mathbf{u}$: control input

$\mathbf{u_{eq}}$: control input to make the quadrotor hover

$\mathbf{x_{eq}}$: states corresponding to the input $\mathbf{u_{eq}}$

$\mathbf{A}$, $\mathbf{B}$: linearized system matrices of quadrotor

v_1, v_2, v_3, v_4 : linear velocity of UGV wheel centers

ω_1, ω_2, ω_3, ω_4 : angular velocity of UGV wheels

v_{C_B} : UGV velocity in Body frame

b_l : lateral wheel base

r_L, r_R : radius of left and right wheels

S_L, S_R : longitudinal left and right wheel slips

$\mathbf{A}$, $\mathbf{B}$, $\mathbf{C}$: linear time invariant system matrices

$J(t_0)$: quadratic performance index

$\mathbf{Q}$, $\mathbf{R}$, $\mathbf{S}$, $\mathbf{P}$: weighing matrices in LQR

$\mathbf{H}$: Hamiltonian

$\mathbf{K}$: feedback gain

$\mathbf{x_T}$: tracking states

$\mathbf{x_N}$: non tracking states

$\mathbf{r_c}(t)$: command reference signal

$\mathbf{x_i}(t)$: integral of the state error

$\bar{\mathbf{x}}$: $\mathbf{x}(t)$ appended with $\mathbf{x_i}(t)$

$\mathbf{x_e}(t)$: error in tracking states

$\mathbf{A_{new}}$, $\mathbf{B_{new}}$: augmented system matrices for LQI

J_{new} : performance index of LQI

$\mathbf{Q_{new}}$: Weighing matrix for LQI

$L(\bar{\mathbf{x}})$: Lyapunov function

K_P : proportional gain

K_I : integral gain

K_D : differential gain

$v(t)$: instantaneous velocity

$v_d(t)$: desired velocity

$e(t)$: velocity error signal

$E(s)$: Laplace transform of error signal

$U(s)$: Laplace transform of input signal

$F(\boldsymbol{x})$: quadratic function

J_t, J_x : partial derivatives of $J(t)$ with respect to t, and $\mathbf{x}$

$V(\mathbf{x}(t))$: value function

π_0 : initial admissible policy

π_k : policy during iteration k

f_{cg}, g_{cg} : functions to represent CG

W : augmented $\mathbf{A}$ and $\mathbf{B}$

$\boldsymbol{\delta}$: $\mathbf{x}$ appended with $\mathbf{u}$

$\mathbf{A_{est}}$, $\mathbf{B_{est}}$: system matrices identified by CG

ϵ : threshold limit of check for convergence

REFERENCES

[1] Z. He and L. Zhao, "A simple attitude control of quadrotor helicopter based on ziegler-nichols rules for tuning pd parameters," *The Scientific World Journal*, 2014.

[2] R. Lalantha and R. Munasinghe, "Feasibility study of a novel cross assembled multi-quadrotor unmanned aerial vehicle," in *2018 IEEE International Conference on Information and Automation for Sustainability (ICIAfS)*, 2018, pp. 1–6.

[3] F. Fraundorfer, L. Heng, D. Honegger, G. H. Lee, L. Meier, P. Tanskanen, and M. Pollefeys, "Vision-based autonomous mapping and exploration using a quadrotor mav," in *2012 IEEE/RSJ International Conference on Intelligent Robots and Systems*, 2012, pp. 4557–4564.

[4] B. S. E. G. D. K. Hong Chul Yang, Rami AbouSleiman and O. Rawashdeh, "Implementation of an autonomous surveillance quadrotor system," *AIAA Infotech@ Aerospace Conference*, 2009.

[5] M. J. R. Michail G. Michailidis* and K. P. Valavanis, "A survey of controller designs for new generation uavs: The challenge of uncertain aerodynamic parameters," *International Journal of Control, Automation and Systems*, vol. 18, pp. 801–816, 2019.

[6] K. J. Orlik-Ruckemann, "Aerodynamic coupling between lateral and longitudinal degrees of freedom," *AIAA JOURNAL*, vol. 15, pp. 1792–1799, 1977.

[7] B. Erginer and E. Altug, "Modeling and pd control of a quadrotor vtol vehicle," in *2007 IEEE Intelligent Vehicles Symposium*, 2007, pp. 894–899.

[8] S. W. Sung and I.-B. Lee, "Limitations and countermeasures of pid controllers," *Industrial & Engineering Chemistry Research*, vol. 35, no. 8, pp. 2596–2610, 1996.

[9] C. T. Ton and W. MacKunis, "Robust attitude tracking control of a quadrotor helicopter in the presence of uncertainty," in *2012 IEEE 51st IEEE Conference on Decision and Control (CDC)*, 2012, pp. 937–942.

[10] F. R. R. R. A. García and M. G. Ortega, "Robust pid control of the quadrotor helicopter," *IFAC Proceedings Volumes*, vol. 45, pp. 229–234, 2012.

[11] Y. Kartal, V. L. Patrik K, A. Dogan, and F. Lewis, "Backstepping approach for design of pid controller with guaranteed performance for micro-air uav," *Control Theory and Technology*, vol. 18, pp. 19–33, 2019.

[12] Y. ohnson and S. Dasgupta, "Robust hurwitz stability and performance analysis of h-infinity controlled forward-velocity dynamics of uavs in close formation flight using bounded phase conditions in a kharitonov framework," *Journal of The Institution of Engineers (India): Series C*, vol. 95, p. 223–231, 2014.

[13] M. G. O. Guilherme V. Raffo and F. R. Rubio, "Path tracking of a uav via an underactuated h-infinity control strategy," *European Journal of Contro*, vol. 2, p. 194–213, 2011.

[14] M. Farrell, J. Jackson, J. Nielsen, C. Bidstrup, and T. McLain, "Error-state lqr control of a multirotor uav," in *2019 International Conference on Unmanned Aircraft Systems (ICUAS)*, 2019, pp. 704–711.

[15] F. L. Lewis, D. Vrabie, and V. L. Syrmos, *Optimal Control*, John Wiley Sons, 2012.

[16] F. L. Lewis and D. Vrabie, "Reinforcement learning and adaptive dynamic programming for feedback control," *IEEE Circuits and Systems Magazine*, vol. 9, no. 3, pp. 32–50, 2009.

[17] H. Modares and F. L. Lewis, "Optimal tracking control of nonlinear partially-unknown constrained-input systems using integral reinforcement learning," *Automatica*, vol. 50, p. 1780–1729, 2014.

[18] R. S. Sutton and A. G. Barto, "Reinforcement learning: An introduction," *Cambridge, U.K.: Cambridge Univ. Press*, 1998.

[19] D. V. Kyriakos G. Vamvoudakis and F. L. Lewis, "Online adaptive algorithm for optimal control with integral reinforcement learning," *International Journal of Robust and Nonlinear Control*, vol. 24, p. 2686–2710, 2014.

[20] B. Khalil and A. Yesildirek, "System identification of uav under an autopilot trajectory using arx and hammerstein-wiener methods," in *7th International Symposium on Mechatronics and its Applications*, 2010, pp. 1–5.

[21] C. Alippi and V. Piuri, "Experimental neural networks for prediction and identification," *IEEE Transactions on Instrumentation and Measurement*, vol. 45, no. 2, pp. 670–676, 1996.

[22] C. J. Li and Y. C. Jeon, "Genetic algorithm in identifying non linear auto regressive with exogenous input models for non linear systems," in *1993 American Control Conference*, 1993, pp. 2305–2309.

[23] D. Wang and F. Ding, "Hierarchical least squares estimation algorithm for hammerstein–wiener systems," *IEEE Signal Processing Letters*, vol. 19, no. 12, pp. 825–828, 2012.

[24] M. Sano and Lianming Sun, "Identification of hammerstein-wiener system with application to compensation for nonlinear distortion," in *Proceedings of the 41st SICE Annual Conference. SICE 2002.*, vol. 3, 2002, pp. 1521–1526 vol.3.

[25] L. Lasdon, S. Mitter, and A. Waren, "The conjugate gradient method for optimal control problems," *IEEE Transactions on Automatic Control*, vol. 12, no. 2, pp. 132–138, 1967.

[26] T. Blumensath and M. E. Davies, "Gradient pursuits," *IEEE Transactions on Signal Processing*, vol. 56, no. 6, pp. 2370–2382, 2008.

[27] J. Sjöberg, H. Hjalmarsson, and L. Ljung, "Neural networks in system identification," *IFAC Symposium on System Identification (SYSID'94)*, pp. 359–382, 1994.

[28] A. Ayyad, M. Chehadeh, M. I. Awad, and Y. Zweiri, "Real-time system identification using deep learning for linear processes with application to unmanned aerial vehicles," *IEEE Access*, vol. 8, pp. 122 539–122 553, 2020.

[29] T. A. Tutunji, "Parametric system identification using neural networks," *Applied Soft Computing*, vol. 47, pp. 251–261, 2016.

[30] M. Jiang and Q. Jin, "Multivariable system identification method based on continuous action reinforcement learning automata."

[31] B. L. Stevens, F. L. Lewis, and E. N. Johnson, *Aircraft Control and Simulation: Dynamics, Controls Design, and Autonomous Systems*, John Wiley Sons, 2016.

[32] D. Lee, T. C. Burg, D. M. Dawson, D. Shu, B. Xian, and E. Tatlicioglu, "Robust tracking control of an underactuated quadrotor aerial-robot based on a parametric uncertain model," pp. 3187–3192, 2009.

[33] H. D. Curtis, *Orbital Mechanics for Engineering Students*, Elsevier Butterworth-Heinemann, 2005.

[34] C. Powers, D. Mellinger, and V. Kumar, *Handbook of Unmanned Aerial Vehicles: Quadrotor Kinematics and Dynamics*, Springer Netherlands, 2015.

[35] L. Derafa, T. Madani, and A. Benallegue, "Dynamic modelling and experimental identification of four rotors helicopter parameters," in *2006 IEEE International Conference on Industrial Technology*, 2006, pp. 1834–1839.

[36] A. Das, K. Subbarao, and F. Lewis, "Dynamic inversion with zero-dynamics stabilisation for quadrotor control," *IET, Control Theory Applications*, vol. 3, p. 303–314, 2009.

[37] D. Wang and C. B. Low, "Modeling and analysis of skidding and slipping in wheeled mobile robots: Control design perspective," *IEEE Transactions on Robotics*, vol. 24, no. 3, pp. 676–687, 2008.

[38] J. Yi, H. Wang, J. Zhang, D. Song, S. Jayasuriya, and J. Liu, "Kinematic modeling and analysis of skid-steered mobile robots with applications to low-cost inertial-measurement-unit-based motion estimation," *IEEE Transactions on Robotics*, vol. 25, no. 5, pp. 1087–1097, 2009.

[39] D. Mellinger and V. Kumar, "Minimum snap trajectory generation and control for quadrotors," in *2011 IEEE International Conference on Robotics and Automation*, 2011, pp. 2520–2525.

[40] J. Willems, "Least squares stationary optimal control and the algebraic riccati equation," *IEEE Transactions on Automatic Control*, vol. 16, no. 6, pp. 621–634, 1971.

[41] A. Laub, "A schur method for solving algebraic riccati equations," *IEEE Transactions on Automatic Control*, vol. 24, no. 6, pp. 913–921, 1979.

[42] P. C. Young and J. C. Willems, "An approach to the linear multivariable servomechanism problem," *International Journal of Control*, vol. 15, no. 5, pp. 961–972, 1972.

[43] I. Kisszölgyémi, K. Beneda, and Z. Faltin, "Linear quadratic integral (lqi) control for a small scale turbojet engine with variable exhaust nozzle," in *2017 International Conference on Military Technologies (ICMT)*, 2017, pp. 507–513.

[44] Y. Kartal, K. Subbarao, N. R. Gans, and F. Lewis, "Distributed backstepping based control of multiple uav formation flight subject to time delays," *IET Control Theory and Applications*, pp. 1–11, 2020.

[45] A. A. Aly, "Distributed backstepping based control of multiple uav formation flight subject to time delays," *Intelligent Control and Automation*, vol. 2, pp. 69–76, 2011.

[46] J. Nocedal and S. J. Wright, *Numerical Optimization*, Springer Series in Operation Research and Financial Engineering, 2006.

[47] C. T. Kelley, *Iterative Methods for Linear and Nonlinear Equations*, Society for Industrial and Applied Mathematics, 1995.

[48] F. L. L. Hamidreza Modares and M.-B. Naghibi-Sistani, "Integral reinforcement learning and experience replay for adaptive optimal control of partially-unknown constrained-input continuous-time systems," *Automatica*, vol. 50, pp. 193–202, 2014.

[49] J. Y. Lee, J. B. Park, and Y. H. Choi, "Integral reinforcement learning for continuous-time input-affine nonlinear systems with simultaneous invariant explorations," *IEEE Transactions on Neural Networks and Learning Systems*, vol. 26, no. 5, pp. 916–932, 2015.

[50] S. Gatesichapakorn, J. Takamatsu, and M. Ruchanurucks, "Ros based autonomous mobile robot navigation using 2d lidar and rgb-d camera," in *2019 First International Symposium on Instrumentation, Control, Artificial Intelligence, and Robotics (ICA-SYMP)*, 2019, pp. 151–154.

[51] L. Heisnam and B. Suthar, "20 dof robotic hand for tele-operation: — design, simulation, control and accuracy test with leap motion," in *2016 International Conference on Robotics and Automation for Humanitarian Applications (RAHA)*, 2016, pp. 1–5.

[52] A. Dzikri and D. E. Kurniawan, "Hand gesture recognition for game 3d object using the leap motion controller with backpropagation method," in *2018 International Conference on Applied Engineering (ICAE)*, 2018, pp. 1–5.

[53] Q. Yang, W. Ding, X. Zhou, D. Zhao, and S. Yan, "Leap motion hand gesture recognition based on deep neural network," in *2020 Chinese Control And Decision Conference (CCDC)*, 2020, pp. 2089–2093.

[54] C. Chen, L. Chen, X. Zhou, and W. Yan, "Controlling a robot using leap motion," in *2017 2nd International Conference on Robotics and Automation Engineering (ICRAE)*, 2017, pp. 48–51.

[55] http://wiki.ros.org/leap_motion.

[56] https://github.com/ros drivers/leap_motion.

[57] Y. Bai, H. Hu, Y. Li, C. Zhao, L. Luo, and R. Wang, "Research methods for human activity space based on vicon motion capture system," in *2017 5th International Conference on Enterprise Systems (ES)*, 2017, pp. 202–206.

[58] L. T. Goodarzi F A, Lee D, "Geometric adaptive tracking control of a quadrotor unmanned aerial vehicle on se(3) for agile maneuvers," *Journal of Dynamic Systems Measurement Control*, vol. 137, no. 9, p. 393–398, 2015.

[59] H. G. Aalerud A, Dybedal J, "Automatic calibration of an industrial rgb-d camera network using retroreflective fiducial markers," *Sensors (Basel, Switzerland)*, vol. 19, no. 7, 2019.

[60] S. Al Habsi, M. Shehada, M. Abdoon, A. Mashood, and H. Noura, "Integration of a vicon camera system for indoor flight of a parrot ar drone," in *2015 10th International Symposium on Mechatronics and its Applications (ISMA)*, 2015, pp. 1–6.

[61] F. Ruffier and F. Expert, "Visual motion sensing onboard a 50-g helicopter flying freely under complex vicon-lighting conditions," in *2012 ICME International Conference on Complex Medical Engineering (CME)*, 2012, pp. 634–639.

[62] Y. Xu, Y. Zhang, Y. Wang, and X. Wang, "Physical experimental realization of modified artificial physics method based on uavs formation control," in *2017 9th International Conference on Intelligent Human-Machine Systems and Cybernetics (IHMSC)*, vol. 2, 2017, pp. 3–6.

[63] http://wiki.ros.org/vicon_bridge.

[64] https://github.com/ethz asl/vicon_bridge.

[65] http://wiki.ros.org/Robots/Husky.

[66] https://github.com/husky/husky/tree/kinetic devel.

[67] http://wiki.ros.org/ardrone_autonomy.

[68] https://github.com/AutonomyLab/ardrone_autonomy.

BIOGRAPHICAL STATEMENT

Suhas Priyatham MANDA received his Bachelor of Engineering degree in Electronics and Communications Engineering at Osmania University, India in 2016. He is a Master of Science student at Electrical Engineering Department in University of Texas at Arlington. His current interests include nonlinear control, machine learning, distributed controls, wireless sensor networks, autonomous and intelligent robots.

Email: suhaspriyatham.manda@mavs.uta.edu