2017

Position Control of an Unmanned Aerial Vehicle From a Mobile Ground Vehicle

Astha Tiwari
Michigan Technological University, asthat@mtu.edu

POSITION CONTROL OF AN UNMANNED AERIAL VEHICLE FROM A MOBILE GROUND VEHICLE

By

Astha Tiwari

A THESIS

Submitted in partial fulfillment of the requirements for the degree of

MASTER OF SCIENCE

In Electrical Engineering

MICHIGAN TECHNOLOGICAL UNIVERSITY

2017

This thesis has been approved in partial fulfillment of the requirements for the Degree of MASTER OF SCIENCE in Electrical Engineering.

Department of Electrical and Computer Engineering

Thesis Advisor:	*Dr. Timothy C. Havens*
Committee Member:	*Dr. Jeremy Bos*
Committee Member:	*Dr. Zhaohui Wang*
Department Chair:	*Dr. Daniel R Fuhrmann*

Dedication

To my family, teachers and friends

for their unconditional love and unwavering faith in me without which I would neither be who I am nor would this work be what it is today.

Contents

List of Figures

List of Tables

Acknowledgments

I would like to gratefully thank various people who have helped me out throughout this excursion. First and foremost, I would like to thank my parents and family for their immense support and encouragement. I owe my most profound appreciation to my advisor Dr. Timothy Havens, who has been exceptionally patient and guided me at each progression of the process. Without his support, positivity and enthusiasm my work would not have been completed. I am deeply grateful to my committee members, Dr. Jeremy Bos and Dr. Zhaohui Wang for being so warm and co-operative.

I owe a great debt of gratitude to my lab-mates Joshua Manela and Joseph Rice for helping me out on whatever point I was stuck with any problem. I am indebted to Donald Williams for his unwavering support and motivation which helped me to be on focussed toward my work. Last but not the least, I am eternally thankful to all my friends for bearing with me through the last couple of years and helping me make this possible.

Abstract

The core objective of this research is to develop a closed loop feedback system between an unmanned aerial vehicle (UAV) and a mobile ground vehicle. With this closed loop system, the moving ground vehicle aims to navigate the aerial vehicle remotely. The ground vehicle uses a pure pursuit algorithm to traverse through a pre-defined path. A Proportional-Integral-Derivative (PID) controller is implemented for position control and attitude stabilization of the unmanned aerial vehicle.

The problem of tracking and pose estimation of the UAV based on vision sensing is investigated. An estimator to track the states of the UAV using the images obtained from a single camera mounted on the ground vehicle is developed. This estimator coupled with a Kalman filter determines the UAV's three dimensional position. The relative position of the UAV from the moving ground vehicle and control output from a joint centralized PID controller is used to further navigate the UAV and follow the motion of the ground vehicle in closed loop to avoid time delays. This closed loop system is simulated in MATLAB and Simulink to validate the proposed control and estimation approach. The results obtained validate the control architecture proposed to attain a closed loop feedback between the UAV and the mobile ground vehicle. UAV closely follows the ground vehicle and various simulation support the parameter settings.

Chapter 1

Introduction

Over the past decade interest in UAV has grown significantly primarily because of its increased availability. Vision based control for quadcopter is an extremely interesting problem with a wide range of practical applications. Quadcopters, in particular are much more promising where human access is limited. Quadcopters are now being deployed for tasks such as remote inspection and monitoring, surveillance, mapping and target tracking.

Quadrotors have been developed with controls aiding good maneuverability, simple mechanics, and the ability to hover, take-off and land vertically with precision. Due to their small size, they can get close to targets of interest and also remains undetected at low heights. The main drawbacks for a quadcopter are its high-power consumption

and payload restriction. Due to limited payload capability, the number of sensors on-board are restricted. To overcome this limitation vision-based techniques and remote control for the quadcopter are primary areas of research these days.

Vision-based sensors are low in cost and weight and can be easily deployed with quadcopters for diverse applications, especially in GPS-denied areas, like indoor environments. A common application for quadcopters currently is for target detection and tracking. Vision-based remote navigation for a quadcopter is well suited for such tasks. UAV's can be used for sensing from above; hence, we propose an application whera a UAV flies in front of a moving ground vehicle in order to sense hazards or objects. The portion of this problem solved in this thesis is the tracking and control of the UAV.

Work presented in this thesis deals primarily with the development of a tracking (or localization) and control method for a UAV in collaboration with a moving ground vehicle. To approach this problem a closed loop feedback control system for a quadcopter to follow a moving ground vehicle is developed. A novel algorithm is developed for vision-based estimation for detecting and tracking of the quadcopter by using image processing. Six degree-of-freedom(6DOF) tracking of the quadcopter is accomplished with a monocular camera mounted on the ground vehicle, which tracks coloured markers on the UAV. This tracking result is then passed to a Kalman filter, which is used to estimate the pose of the UAV.This approach gives very precise pose

estimation of position, altitude, and orientation of the quadcopter with respect to its counterpart on ground.Finally, a control input for PID controller is communicated to the UAV in order to maintain a predetermined path; e.g. maintaining a position of 25 meters in front of the vehicle, or weave back and forth in front of the vehicle.

1.1 Literature Review

1.1.1 Quadcopter Control

Quadcopter control and its implementation in various applications have made significant progress over time. Nowadays autonomous navigation for a quadcopter is extremely easy to attain with multiple algorithms and methods from which to choose. A quadrotor has four equally spaced rotors placed at the corners of a square structure. It generally uses two pairs of identical propellers. Two of these propellers spin clockwise and the another two in a counter-clockwise direction. By varying the speed of each rotor, any desired thrust can be generated to lift and maneuver the craft. To navigate the quadcopter, we need to control these four independent rotors.

In general, fundamental quadcopter control in itself is a difficult problem. With six degrees of freedom, three rotational and three translational, and only four independent inputs make this system severely underactuated. The rotational and translational

degrees of freedom are coupled making the system dynamics non-linear in nature. Except for air friction, which is negligible, quadcopters do not have friction to suppress or aid their motion; so, they must provide their own damping for stability and movement.

In [3], authors presented the modelling and simulation of an autonomous quadrotor micro-copter in a virtual outdoor scenario. They also developed flight simulator to include controller design and model validation. In [13] and [26], a study of the kinematics and dynamics of a quadrotor are discussed for better understanding of the underlying physics and mathematics involved in the modelling of a quadrotor. These work deal with the implementation of flight dynamics but bot the control for the navigation.

Several researchers have presented their approach to achieve control for the quadcopter. Research presented in [31] and [6] applies a PID controller for attitude stabilization of a quadcopter. In both these work position control to fly the quadcopter is not talked about. Their control model is limited for hovering applications only. A simple PID control is also sufficient for hovering and attitude control. Using more sophisticated controllers improves the performance, but increases the computation power needed. These controllers are tougher to design and deliver very little improvement in minimal close to hover flight conditions. There will always be a tradeoff between computational capacity and performance requirements. Emphasis was also

on adaptive control schemes for quadcopter control. It is believed that such control can improve the quadcopter performance in terms of model uncertainties, varying pay-loads, and disturbance rejections. Research shown in [25] shows that both PID and adaptive control give similar results. Very little improvement was observed when using the adaptive control. Also, the controller developed is not robust and depends on varying the parameters for different navigation tasks. Experiments in [36] and [41] confirm that quadcopter stability can be achieved with PID controller for target tracking tasks but the results of these work lack the optimal tuning of the PID controller parameters.

Work in [42] defines the feedback linearization base control method for landing a UAV on a base vehicle moving with a constant speed. This method reduces the problem to a fixed pre-defined trajectory for the quadrotor. This method is difficult to implement correctly because we can not identify the actual parameters in the decoupling matrix for the UAV.The authors in [43] successfully demonstrates autonomous landing of UAVs on moving ground vehicles. Takeoff to a fixed attitude and then tracking and landing are achieved solely using PID controller for the attitude and position control. A PID controller is shown to be sufficient to achieve linear flight regime in hover condition as demonstrated in [22].

To overcome these issues of robustness and computational load for the control, in this thesis we implement PID control for both position control and attitude stabilization.

The daunting task with the PID controller is to tune the parameters in such a way; so, as to achieve a robust position control for a quadcopter. To optimally tune the parameters, Ziegler-Nicols rule was used for both attitude and position control of the quadcopter. This tuning gave the PID loop best disturbance rejection and made the system extremely robust. In all the previous cited studies, quadcopter control is the focus onto but vision based integration using image processing techniques are not discussed.

1.1.2 Tracking and Vision-Based Estimation

Our objective is to develop a vision-based algorithm for localization of the quadcopter with respect to the ground vehicle. Using multiple frames makes the entire problem difficult to implement and computationally very expensive. So, we implement image processing for one frame at a time. Heavy computational-load algorithms can not be used on low power computers onboard a typically small quadcopter. The main challenge addressed here is tracking and estimating the relative altitude between the quadcopter and ground vehicle based on monocular camera measurements. For this purpose, sensor fusion of a monocular camera with GPS or other laser-based sensors can be used for estimation to increase accuracy in measurements obtained.

Firstly we had to be certain that vision-based sensors would be best for our application. For this purpose we scrutinized other sensors such as radar. A modified aircraft tracking algorithm to include radar measurements taken from a stationary target position are used in both [17] and [35]. For this algorithm, it is important that the target is in close vicinity of the measuring device, which is not always possible. To overcome this, the authors of [39] develop an alternative to fixed sensor location by assuming moving target indicators or position measurements to be directly available from aircraft sensor. Drawback of using radar as inferred from the above cited work was its limited area of operation. Also, using the radar increased the payload for the craft.

It is to be noted that in general for target tracking problems, cameras have an advantage over active range sensors, as they are passive and thus targets are unaware of observation. Currently, Simultaneous Localization and Mapping(SLAM) techniques are widely used and implemented for monocular vision systems, as discussed in [29] and [5]. In [5], authors assume that an air vehicle maintains hover position and all features are in one plane. This reduces the problem to two dimensions but makes the controller application very limited. Similarly, in [29], the authors use artificially placed features of known dimension to reconstruct three-dimensional pose for the aircraft from a two-dimensional image obtained from a monocular camera. Both these approaches propose an algorithm to simplify the pose estimation problem for a quadcopter. In [33], an algorithm is presented which relies on visual markers and wireless

transmission of camera images to a ground computer, which then remotely controls the UAV. In [21], [44] and [11], monocular visual stabilization techniques that measure relative motion are discussed and developed. These systems utilize an on-board camera setup for simultaneous localization and mapping (SLAM) with the UAV. Similar research has been conducted with stereo camera systems, which are computationally more expensive. Work presented in [24] and [32] use ground mounted fixed stereo cameras to remotely control the quadcopter from a ground computer, whereas in [10] a forward-facing stereo camera is mounted on the quadcopter to achieve closed-loop control for quadcopter motion with final processing on the ground station. In all these above mentioned work the camera is mounted on-board the UAV facing downwards. This arrangement also increases the payload and thereby effecting the size and power consumption for the UAV. The algorithms developed are computational expensive due to SLAM implementation.

In [23] and [20], the researchers propose a vision based autopilot relying only on optic flow. The main limitation of this method is that for accurate altitude stabilization, the quadrotor should have significant motion in the field of view, which fails for a hover condition. The work presented in [8] and [7] is very interesting as it aims at tracking a moving ground vehicle with markers from a UAV using solely vision sensing. The control takes action by gathering images from a camera mounted on the UAV and then, by appropriate image processing techniques estimates the position and velocity of the ground vehicle. The algorithm developed here is very efficient

but overlooks to include backup in cases of insufficient data when images could not be obtained.Quadcopters are finding applications in various fields while implementing SLAM. To overcome the payload restrictions, mounting the camera on the quadcopter will be troublesome to implement in an experimental setup. Hence, in this case, a camera on a ground vehicle is beneficial.

1.1.3 Closed Loop Feedback Control for UAV and ground vehicle

With the vision-based estimation and tracking the next section of this thesis was to develop a closed loop feedback control for UAV and a moving ground vehicle for navigating a UAV remotely.Numerous controller designs for autonomous landing of a UAV on a moving target have been developed.In [9] and [27], a UAV and Unmanned Ground Vehicle(UGV) work in collaboration and should know about mutual positions and estimate relative pose of the UGV to the UAV, and in [34] mutual localization is achieved to determine the relative position of the UAV with respect to the UGV. When estimating the position of the UGV from the camera on-board the UAV for closed loop control system, many researchers have worked with color detection for image processing. It consists in placing colored markers on a the UGV surface. These techniques are discussed in [30] and [12], where they go through the acquired videos from the UAV until a match is found with tracked color. Once the location of the

tracked object is found, the relative pose of the UGV is extracted. Color tracking methods are mostly reliable due to easy implementation but have some drawbacks. This method is sensitive to lighting conditions and depends on the environment. In [40], the author discusses a feature-based (black and white pattern) detection for multiple robot localization. The presented method allows the tracking of several features at a time with good precision and accuracy. The above mentioned works deal with tracking of the ground vehicle and hence deals only for two dimensions.

In [37], the authors propose a nonlinear control technique for landing a UAV on a moving boat. Here the attitude dynamics are controlled using fast time scale with guaranteed Lyapunov stability. The authors present the simulation results for the same with guaranteed robustness for bounded tracking errors in the attitude control. The method relies on time scale separation of attitude and position control which makes the controller algorithm computationally expensive and dependent on the communication protocol used. A tether is used for communication between the boat and UAV. Similarly, authors in [28] work to derive a control technique for coordinated autonomous landing on a skid-steered UGV. First a linearized model of the UAV and the UGV are developed and then control strategies are incorporated in the closed loop to study the coordinated controller action. In this work, the non-linearities of the quadcopter are overlooked.

The effects of airflow disturbances are investigated and shown in [19] with non-linear

controller design and simulation for landing the quadcopter on a moving target. [15] focuses on developing an estimation technique to track the position and velocity of a moving ground target as viewed from a single monocular camera on-board the UAV. Here an inertial measurement unit (IMU) with three axes accelerometer and rate gyro sensors along with a GPS unit give the states for the UAV which, when combined with the camera estimates, give the target states. Similarly, in both [4] and [1], the author develops an object tracking controller using an on-board vision system for the quadcopter. It also demonstrates stable closed loop tracking for all three control axes, individually and collaboratively. These work don't aim at reducing the size of the quadcopter and minimizing the power needed for its operation as in all these work the camera is mounted on the UAV and also the pose estimation is done by the camera on-board the quadcopter.

Work in [18] takes the coordinated control for a UAV and UGV a step further by presenting an algorithm for tracking a ground vehicle from a camera mounted on the UAV with a constant stream of video of the motion. The conventional vision based tracking algorithm also uses a fuzzy logic controller for a more precise result. This makes the control action computationally expensive and slows down the process. Work presented in thesis intends to attain target tracking based only on vision sensing from a single camera mounted on the ground vehicle. Accurate altitude and yaw estimation of the UAV is achieved using image processing algorithms discussed in chapter 5.

1.2 Motivation and Research Objective

Navigation and position control of an unmanned aerial vehicle has been a subject of intense study in the past decades. Typically sensors on board of the aerial vehicle include an inertial measurement unit(IMU) with three axes accelerometers and rate gyro sensors, a global positioning system (GPS), and a camera, which are used to give the position for the vehicle. Various sensor fusion algorithms are deployed for the control of the UAV position. Due to payload restriction and with intent to reduce the size, researchers are focussing on techniques to remotely control and localize the UAV.

In the literature review, recent developments in the field of quadcopter control and vision based tracking were discussed. In this research, we intend to accomplish reliable target tracking of a quadcopter using a single monocular camera onboard a moving ground vehicle unlike for most cases as discussed in section 1. Position, altitude, and rotation for the target quadcopter relative to the ground vehicle are estimated based on images obtained and further processing for vision based tracking. Algorithms and control designs are implemented and simulated in MATLAB and Simulink.

A novel algorithm presented in this thesis proposes an efficient and accurate pose estimation of the quadcopter from a ground vehicle. In [45], an assumption is made

that the yaw of the quadcopter is inherently known and thus there is no need to estimate. From this research, an understanding is developed to implement a consistent quadcopter controller for attitude stabilization and position navigation. Coordinated landing of a quadcopter on a moving ground vehicle is discussed in [2], [38] and [28]. In these works the camera is mounted on the ground vehicle or at the ground station but the camera is stationary. Conventionally, quadcopters carry a camera on them for vision based navigation, but for the work in this a thesis camera is mounted on the moving ground vehicle. The camera faces up and forwards and provides images of the moving quadcopter. This simplifies the problem statement as it aids in pose estimations from ground vehicle and simplifies the quadcopter control problem. This also helps in ground air interactions for multi-agent systems. Also, to deal with situations of temporary target loss the vision-based tracking algorithm was coupled with Kalman filter to estimate position.

1.3 Thesis Outline

In this thesis, we implemented a vision-based tracking and remote navigation method for quadcopter control from a ground vehicle. The vision-based estimation algorithm is successfully achieved with anticipated control strategies and tolerable error in pose estimation. Then using predefined path from a ground vehicle is used to navigate the quadcopter. More complex control algorithms can be implemented for future

work. There are various approaches to deal with quadcopter attitude and position control and stabilization; work presented here focus on integration of image processing algorithms with a novel PID controller for quadcopter navigation from a remote moving ground vehicle. For this purpose, PID control laws are implemented on the quadcopter for both position control and attitude stabilization.

This thesis is divided into 7 chapters. In the first chapter an introduction to the thesis is given discussing the literature review of the previous work done in the field, motivation behind the project, and an outline for the project itself. Chapter 2 discusses the mathematical model for a quadcopter. Derivation for a quadcopters non-linear equation of motion are given.

Chapter 3 focus on the control laws for the non-linear dynamic model of the quadcopter. Here we have modeled a classic controller for attitude stabilization and position tracking. In Chapter 4, the mathematical model derived for the quadcopter and its position control are modeled in Simulink. Chapter 5 develops an algorithm for vision-based estimation and tracking of the quadcopter. We develop our image processing for detection and tracking of the quadcopter in MATLAB. Also, we validate the algorithm by simulating a video of a randomly moving quadcopter and results of this simulation are included.

Chapter 6 aims at developing the overall control architecture for position control of a quadcopter in closed loop feedback with a moving ground vehicle. Initially we derive

the kinematic equations for the ground vehicle and then outline the overall system. Chapter 7 shows simulation and results of the overall model developed in various cases to validate the work. Finally, Chapter 8 provides a summary of the project and explores the possible future research in this field.

Chapter 2

Quadcopter Dynamics

Quadcopters are under-actuated systems in which four independent inputs are used to control motion in six degrees of freedom, three translational and three rotational. The resulting dynamics are highly nonlinear primarily because of aerodynamic effects. Quadcopters are lifted and propelled using four vertically oriented propellers. Quadrotors use two pairs of identical fixed pitched rotors, two in clockwise direction placed diagonally with respect to each other and the other two in counter-clockwise direction placed diagonally. Figure 2.1 [16] shows the configuration of the quadcopter.

To control and maneuver the quadcopter, independent variation in the speed of rotor can be changed accordingly. Change in speed of each rotor affects the total

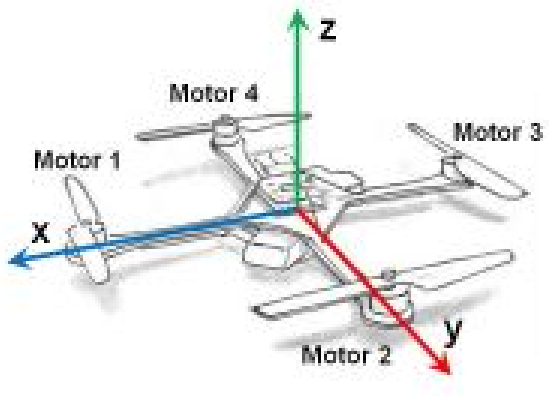

Figure 2.1: Configuration of a Quadcopter [16]

thrust which can be further controlled to create desired total torque to navigate the quadcopter in all three directions. Figure 2.2 [16] shows the motion principle for a quadcopter. By varying the roll, movement in x direction can be controlled. Similarly, by controlling the pitch of the craft ,movement in y direction is achieved. To increase the roll, the thrust on motor 1 is decreased and for the diagonally placed motor 3, thrust is increased. This results in a moment which tilts the quadcopter by creating a component of thrust in the x direction. Motor 2 and motor 4 work together in a similar manner to maintain the position in the y direction. Quadcopter Dynamics are discussed and a mathematical formulation for the Quadcopter model is derived and discussed in detail in [16]. Maintaining the altitude of the quadcopter, i.e., its position in the z-axis is achieved by equally distributing the thrust needed on all four motors. Decreasing the thrust on one pair of motors and increasing the thrust by an equal amount on the other two motors creates a moment about z-axis of the quadcopter triggering it to yaw. To determine all the equation governing the

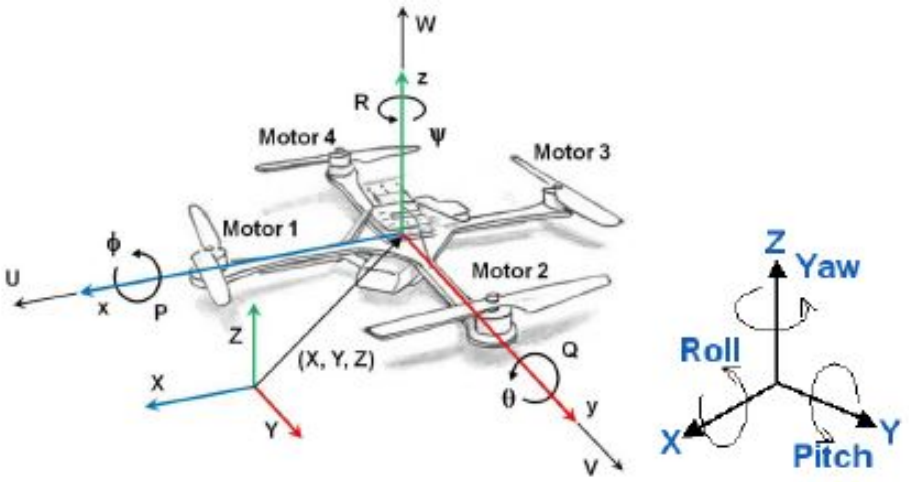

Figure 2.2: Motion Principle for the Quadcopter [16]

motion of the quadcopter we need to analyze it in the inertial reference frame and the fixed body frame. The inertial reference frame is defined by the ground with gravitational force in the negative z-axis and the fixed body frame is defined when the rotor axes point in the positive z-axis and the arms pointing in the x and y direction. The inertial reference frame is used to measure the position and velocity of the quadcopter whereas to determine the orientation and angular speeds we need to assess the body-fixed frame. Lets define the state vectors as follows:

$$\text{Position Vector: } \mathbf{x} = [x, y, z]^T \text{ \& Velocity Vector: } \mathbf{v} = [v_x, v_y, v_z]^T, \tag{2.1}$$

$$\text{Orientation Vector: } \boldsymbol{\alpha} = [\phi, \theta, \varphi]^T \text{ \& Angular Velocity Vector: } \boldsymbol{\omega} = [\omega_x, \omega_y, \omega_z]^T, \tag{2.2}$$

Where $\mathbf{X}$, the position vector, is defined relative to the inertial reference frame and

the other three vectors are defined relative to the fixed body frame. Also, the standard Euler angles, with notations ϕ, θ *and* φ are called pitch, roll and yaw of the quadcopter, respectively. The angular velocity vector $\omega \neq \dot{\alpha}$ and instead is the vector pointing along the axis of rotation, given by the following relation.

$$\boldsymbol{\omega} = \begin{bmatrix} 1 & 0 & -\sin(\theta) \\ 0 & \cos(\phi) & \cos(\theta)\sin(\phi) \\ 0 & -\sin(\phi) & \cos(\theta)\cos(\phi) \end{bmatrix} \dot{\boldsymbol{\alpha}}. \tag{2.3}$$

The matrix above can also be called the transformation matrix from the inertial frame to the fixed-body frame and the transformation matrix from the body frame to the inertial frame is given by

$$\dot{\boldsymbol{\alpha}} = \mathbf{W}_n \boldsymbol{\omega} = \begin{bmatrix} 1 & \sin\phi\tan\theta & \cos\phi\tan\theta \\ 0 & \cos\phi & -\sin\phi \\ 0 & \frac{\sin\phi}{\cos\theta} & \frac{\cos\phi}{\cos\theta} \end{bmatrix} \boldsymbol{\omega}. \tag{2.4}$$

Similarly, we can determine the relation between the fixed-body frame and the inertial frame by using the rotation matrix R, going from the body frame to the inertial frame. The rotation sequence of any aircraft is described as a rotation about the z-axis (yaw) followed by the rotation about y-axis (pitch) and eventually by the rotation about the x-axis (roll). Using the three rotations, a composite rotation matrix can be formed

which goes from the fixed body frame to the inertial frame and is given by

$$\mathbf{R} = \begin{bmatrix} \cos\varphi\cos\theta & (\cos\varphi\sin\theta\sin\phi - \sin\varphi\cos\phi) & (\cos\varphi\sin\theta\cos\phi + \sin\varphi\sin\phi) \\ \sin\varphi\cos\theta & (\sin\varphi\sin\theta\sin\phi + \cos\varphi\cos\phi) & (\sin\varphi\sin\theta\cos\phi - \cos\varphi\sin\phi) \\ -\sin\theta & -\cos\theta\sin\phi & \cos\theta\cos\phi \end{bmatrix}. \tag{2.5}$$

It is to be noted that this rotation matrix is orthogonal and hence $\mathbf{R}^{-1} = \mathbf{R}^T$, which is the rotation matrix from the inertial reference frame to the fixed-body frame. Together, the relationship between linear velocities and angular rates in the inertial and fixed-body frame are given by

$$\begin{bmatrix} \dot{\mathbf{X}} \\ \dot{\boldsymbol{\alpha}} \end{bmatrix} = \begin{bmatrix} \mathbf{R}(\boldsymbol{\alpha}) & \mathbf{0} \\ \mathbf{0} & \mathbf{W}_n(\boldsymbol{\alpha}) \end{bmatrix} \begin{bmatrix} \mathbf{v} \\ \boldsymbol{\omega} \end{bmatrix}. \tag{2.6}$$

The equations of motion for the quadcopter are expressed in terms of fixed-body velocities and angular rates. We implement the Lagrangian approach extended to quasi-coordinates for the equation of motion. Here we can not directly use the angular velocities to determine the value for orientation vector, so we use the quasi-coordinate formulation, where the components of angular velocities are defined in terms of the derivative values with respect to time. The equations can be written as follows

$$L = T - V, \tag{2.7}$$

$$L = \frac{1}{2}m\mathbf{v}^T\mathbf{v} + \frac{1}{2}\boldsymbol{\omega}^T\mathbf{I}\boldsymbol{\omega} + mgz. \tag{2.8}$$

In the above equation, m is the mass of the quadcopter and $\mathbf{I}$ is its moment of inertia. Now we need to determine its moment of inertia. For this, the quadcopter is assumed to have a perfectly symmetrical structure along all 3 axes and its center of mass is located at the geometric center of its arms. Thus, the inertia matrix becomes a diagonal matrix with $I_{xx} = I_{yy}$, and the inertia matrix can be reduced to

$$\mathbf{I} = \begin{bmatrix} I_{xx} & 0 & 0 \\ 0 & I_{yy} & 0 \\ 0 & 0 & I_{zz} \end{bmatrix}. \tag{2.9}$$

Now, if we implement the quasi-coordinate Lagrangian formulation, we obtain the following equations,

$$\frac{d}{dt}(\frac{\partial L}{\partial \mathbf{v}})^T + \boldsymbol{\omega}(\frac{\partial L}{\partial \mathbf{v}}) - (\mathbf{T}_1)^T(\frac{\partial L}{\partial \mathbf{X}}) = \boldsymbol{\tau}_1, \tag{2.10}$$

$$\frac{d}{dt}(\frac{\partial L}{\partial \boldsymbol{\omega}})^T + \mathbf{v}(\frac{\partial L}{\partial \mathbf{v}}) + \boldsymbol{\omega}(\frac{\partial L}{\partial \boldsymbol{\omega}}) - (\mathbf{T}_2)^T(\frac{\partial L}{\partial \boldsymbol{\alpha}}) = \boldsymbol{\tau}_2. \tag{2.11}$$

When we expand 2.10 and 2.11 for each variable of the state vector using the properties of partial derivatives, we obtain the following six equations governing the motion of a quadcopter:

$$m[\dot{v}_x - v_y\omega_z + v_z\omega_y - g\sin(\theta)] = 0, \tag{2.12}$$

$$m[\dot{v}_y - v_z\omega_x + v_x\omega_z + g\cos(\theta)\sin(\phi)] = 0, \tag{2.13}$$

$$m[\dot{v}_z - v_x\omega_y + v_y\omega_x + g\cos(\theta)\cos(\phi)] = u_1, \tag{2.14}$$

$$I_{xx}\dot{\omega}_x + (I_{zz} - I_{yy})\omega_y\omega_z = u_2, \tag{2.15}$$

$$I_{yy}\dot{\omega}_y + (I_{xx} - I_{zz})\omega_x\omega_z = u_3, \tag{2.16}$$

$$I_{zz}\dot{\omega}_z = u_4, \tag{2.17}$$

$$\boldsymbol{\tau} = \begin{bmatrix} 0 \\ 0 \\ u_1 \\ u_2 \\ u_3 \\ u_4 \end{bmatrix} = \begin{bmatrix} 0 \\ 0 \\ T_1 + T_2 + T_3 + T_4 \\ (T_4 - T_2)l \\ (T_3 - T_1)l \\ (T_2 + T_4 - T_1 - T_3)\mu \end{bmatrix}. \tag{2.18}$$

In 2.12 - 2.18 above T_{1-4} are the forces applied by the four motors on the quadcopter, μ is the torque coefficient and is a constant value for a particular setup, and l is the length of the arm from the center of gravity to the motors. Since the quadcopter is assumed to be symmetric the length is same for all four motors. All the equations written above are derived and formulated in fixed-body frame and can be easily

converted into the inertial frame as shown in the following steps:

$$\begin{bmatrix} \mathbf{R}^{-1} & \mathbf{0} \\ \mathbf{0} & (\mathbf{W}_n)^{-1} \end{bmatrix} \begin{bmatrix} \dot{\mathbf{X}} \\ \dot{\boldsymbol{\alpha}} \end{bmatrix} = \begin{bmatrix} \mathbf{v} \\ \boldsymbol{\omega} \end{bmatrix}, \tag{2.19}$$

$$\begin{bmatrix} \mathbf{R}^{-1} & \mathbf{0} \\ \mathbf{0} & (\mathbf{W}_n)^{-1} \end{bmatrix} \begin{bmatrix} \ddot{\mathbf{X}} \\ \ddot{\boldsymbol{\alpha}} \end{bmatrix} - \begin{bmatrix} \dot{\mathbf{R}} & \mathbf{0} \\ \mathbf{0} & \dot{\mathbf{W}}_n \end{bmatrix} \begin{bmatrix} \mathbf{v} \\ \boldsymbol{\omega} \end{bmatrix} = \begin{bmatrix} \dot{\mathbf{v}} \\ \dot{\boldsymbol{\omega}} \end{bmatrix}. \tag{2.20}$$

Using the transformations in 2.19 and 2.20 we can write the equations of motion in the inertial frame as

$$m\ddot{x} = (\sin(\varphi)\sin(\phi) + \cos(\varphi)\cos(\phi)\sin(\theta))u_1, \tag{2.21}$$

$$m\ddot{y} = (-\cos(\varphi)\sin(\phi) + \sin(\varphi)\cos(\phi)\sin(\theta))u_1, \tag{2.22}$$

$$m\ddot{z} = (\cos(\theta)\cos(\phi))u_1 - mg, \tag{2.23}$$

$$\mathbf{M}_{\boldsymbol{\alpha}}\ddot{\boldsymbol{\alpha}} + \frac{1}{2}\dot{\mathbf{M}}_{\boldsymbol{\alpha}}\dot{\boldsymbol{\alpha}} = \begin{bmatrix} u_2 \\ u_3\cos(\phi) - u_4\sin(\phi) \\ -u_2\sin(\theta) + u_3\cos(\theta)\sin(\phi) + u_4\cos(\theta)\cos(\phi) \end{bmatrix}, \tag{2.24}$$

where $\mathbf{M_{\alpha}} =:$

$$\begin{bmatrix} I_{xx} & 0 & -I_{xx}\sin(\theta) \\ 0 & (I_{yy}+I_{zz})\sin^2(\phi) & (I_{yy}-I_{zz})\cos(\phi)\cos(\theta)\sin(\varphi) \\ -I_{xx}\sin(\theta) & (I_{yy}-I_{zz})\cos(\phi)\cos(\theta)\sin(\varphi) & I_{xx}\sin^2(\theta)+I_{yy}\cos^2(\theta)\sin^2(\phi)+ \\ & & I_{zz}\cos^2(\phi)\cos^2(\theta) \end{bmatrix}. \tag{2.25}$$

From 2.25, it can be easily inferred that the equations of motion governing the translational motion of the quadcopter are much easier to understand and implement in the inertial frame. Similarly, for rotational motion of the quadcopter, equations from body-fixed frame are easier to understand and implement. To summarize, the dynamics of the quadcopter can be designed using the following set of equations:

$$m\ddot{x} = (\sin(\varphi)\sin(\phi) + \cos(\varphi)\cos(\phi)\sin(\theta))T, \tag{2.26}$$

$$m\ddot{y} = (-\cos(\varphi)\sin(\phi) + \sin(\varphi)\cos(\phi)\sin(\theta))T, \tag{2.27}$$

$$m(\ddot{z}+g) = (\cos(\phi)\cos(\theta))T, \tag{2.28}$$

$$I_{xx}\dot{\omega}_x = M_x - (I_{zz}-I_{yy})\omega_y\omega_z, \tag{2.29}$$

$$I_{yy}\dot{\omega}_y = M_y - (I_{xx}-I_{zz})\omega_x\omega_z, \tag{2.30}$$

$$I_{zz}\dot{\omega}_z = M_z. \tag{2.31}$$

where (x, y, z) gives the position of the quadcopter in the inertial frame; (ϕ, θ, φ) are the roll, pitch, and yaw angles respectively; T is the amount of total thrust from all the four motors; M_x, M_y and M_z are the moments about the respective axis generated due to the thrust differences in the opposing pairs of motors; m is the total mass of the Quadcopter, and I_{xx} , I_{yy} , I_{zz} are the moments of inertia for the quadcopter for each individual axis. To determine the total thrust T and the moments M_x, M_y, M_z, we can use

$$\begin{bmatrix} T \\ M_x \\ M_y \\ M_z \end{bmatrix} = \begin{bmatrix} 1 & 1 & 1 & 1 \\ 0 & -l & 0 & l \\ -l & 0 & l & 0 \\ -\mu & \mu & -\mu & \mu \end{bmatrix} \begin{bmatrix} T_1 \\ T_2 \\ T_3 \\ T_4 \end{bmatrix}, \tag{2.32}$$

where T_{1-4} is the thrust generated from the four motors and l is the length of the arm i.e. the distance between the rotor and the center of the quadcopter. Using the above discussed mathematical model derived from the underlying physics of the quadcopter, a MATLAB level-2 S-function was coded to be included in the final model of the quadcopter. This function models the dynamics of the quadcopter motion with the given inputs. In the next chapter we develop the altitude and position controller for the quadcopter's motion.

Chapter 3

Quadcopter Control

After developing a simulated model for the quadcopter using its kinematic equations, the next concern was to develop a control method for quadcopter navigation. While developing the control, all six degrees of freedom had to be considered. Two sets of controllers were designed: one for the altitude and attitude control and the another for position control in the x and y direction.

Controlling the six degrees of freedom of the quadcopter is a challenging task and various kinds of controllers can be implemented to achieve it. In [14], the authors discuss the same. They implemented classical controllers on the quadcopter dynamic model to study the best suitable for a given purpose.

Since we intend to navigate the quadcopter from a ground vehicle, our quadcopter

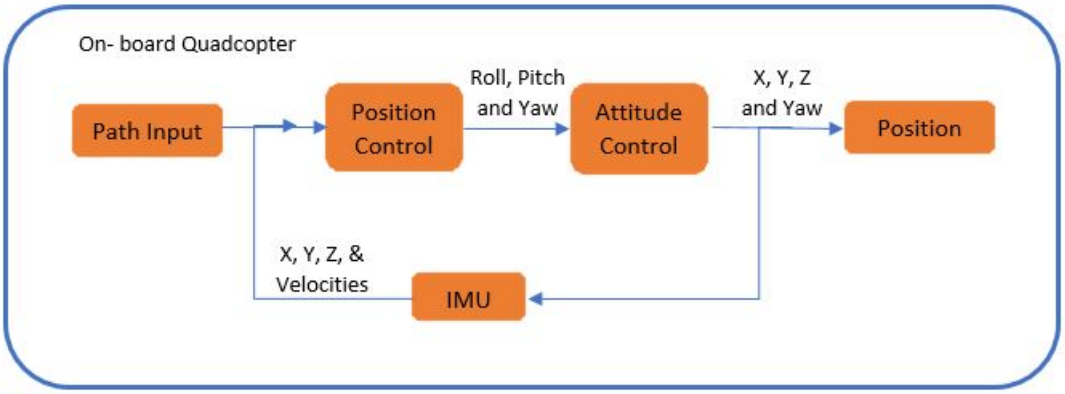

Figure 3.1: Control Structure for the Quadcopter

must follow the path instructions given to it; hence, its position controller should be very prompt in its execution. Although various control algorithm could be developed, a classic PID controller was selected. Any other control method would be computationally expensive for our task. Figure 3.1 shows the control architecture of our quadcopter.

3.1 Attitude Controller

First we discuss the attitude controller. This altitude stabilization controller is implemented to stabilize the quadcopters position in the z axis and its roll, yaw, and pitch angles. This is achieved by implementing a PID controller for all four degrees

of freedom. Control laws for altitude stabilization are implemented using:

$$\phi_{cor} = k_p(\phi_d - \phi) + k_d(\dot{\phi}_d - \dot{\phi}) + k_i \int (\phi_d - \phi)dt, \tag{3.1}$$

$$\theta_{cor} = k_p(\theta_d - \theta) + k_d(\dot{\theta}_d - \dot{\theta}) + k_i \int (\theta_d - \theta)dt. \tag{3.2}$$

where, ϕ_{cor} and θ_{cor} are the control output obtained from the controller after correction, ϕ_d and θ_d are the desired pitch and roll for the quadcopter respectively. Also, k_p, k_d and k_i are tunable PID controller parameters. Please note that the quadcopter is axis-symmetric, so, for roll and pitch, the gains are kept the same for both. Also, we need to ensure that the quadcopter is always in the field of view of the ground vehicle, hence the roll and pitch are constrained to $\pm$ 10 degrees. This restriction limits the acceleration of the quadcopter. For attitude stabilization we need to implement the controller action for the z-axis and also the yaw control. This was achieved by using:

$$\varphi_{cor} = k_p(\varphi_d - \varphi) + k_d(\dot{\varphi}_d - \dot{\varphi}) + k_i \int (\varphi_d - \varphi)dt \tag{3.3}$$

$$z_{cor} = k_p(z_d - z) + k_d(\dot{z}_d - \dot{z}) + k_i \int (z_d - z)dt - g_{offset} \tag{3.4}$$

where, φ_{cor} and z_{cor} are the control output obtained from the controller after correction, φ_d and z_d are the desired yaw and position in z for the quadcopter respectively. Also, k_p, k_d and k_i are tunable PID controller parameters. It is this altitude control which maintains the altitude of the quadcopter relative to the ground target.

3.2 Position Controller

The position controller is called the outer loop controller as it gives the referenced inputs to the attitude controller as shown in the figure 3.1. A simple PD controller is used to give commands for attitude of the vehicle and eventually navigate the quadcopter to the desired position. Inertial-frame velocity is used as the state to be controlled as this is what is controlled by attitude. Thus we find position error in the body frame and map this to the body frame desired velocity. This desired velocity is then mapped to a desired attitude. Control laws for the desired position are

$$\theta_d = k_{p1} v_{xerr} - k_{d1} \dot{v}_{xerr}, \tag{3.5}$$

$$\phi_d = k_{d2} \dot{v}_{yerr} - k_{p2} v_{yerr}. \tag{3.6}$$

For 3.5 and 3.6, the values of v_{xerr} and v_{yerr} are determined in the body frame using the following with the input values of desired path:

$$v_{xerr} = x_{err} cos(\varphi) + y_{err} sin(\varphi), \tag{3.7}$$

$$v_{yerr} = y_{err} cos(\varphi) - x_{err} sin(\varphi). \tag{3.8}$$

Please note that the values of $x_{err} = x_{cmd} - x_{fd}$ and $y_{err} = y_{cmd} - y_{fd}$ are determined to be used in 3.7 and 3.8. Here x_{cmd} & y_{cmd} are the commanded value for x & y respectively and x_{fd} & y_{fd} are the feedback values obtained from control loop.

The values of $\theta_d, \phi_d, \varphi_d$ and z are fed into the attitude controller and then the outputs from the attitude controller are used to control the thrust input to the four rotors on the quadcopter for position control and navigation.

These corrected controller output values are sent to the respective motor according to the mixing laws:

$$M_1 = z_{corr} - \varphi_{corr} - \theta_{corr}, \tag{3.9}$$

$$M_2 = z_{corr} - \varphi_{corr} + \theta_{corr}, \tag{3.10}$$

$$M_3 = z_{corr} + \varphi_{corr} + \phi_{corr}, \tag{3.11}$$

$$M_4 = z_{corr} + \varphi_{corr} - \phi_{corr}. \tag{3.12}$$

These equations determine the movement of the quadcopter. Prior to implementation of the feedback control loop with the ground vehicle, some simulations were run to tune the PID controller and achieve position tracking for the quadcopter. It is to be noted that PID parameters are different for each individual control and have to be tuned to achieve a robust controller action for attitude stabilization and position control for any randomly generated path. The PID control can easily track the

commanded inputs with small error using an optimal set of tuned parameters. Tuning the PID controller parameters was done using the Ziegler- Nichols method. Initially, the k_d and k_i values are set to zero. Then the P gain, $k_p$is then increased slowly from zero until it reaches the ultimate gain K_u, at which the output of the control loop has stable and consistent oscillations. Then for a classic PID controller the values of K_u and the oscillation period T_u are used to set the P, I, and D gains. Ideally, the P gain is set as $k_p = 0.6K_u$, the I gain as $k_i = \frac{2}{T_u}$ and the D gain as $k_d = \frac{T_u}{8}$. These values give the starting point and the using some simulations for attitude stabilization and position, an optimal set of final parameters were obtained for each individual PID controller implemented. Using these control actions and the quadcopter model developed in Chapter 2, we designed a Simulink model for the standalone position controller for the quadcopter. Using the equations from Chapter 2 and Chapter 3, this model is developed and discussed in detail next.

Chapter 4

Quadcopter Model Developement

Using the quadcopter dynamics and control law, a Simulink model was developed to simulate the attitude stabilization and position tracking of a quadcopter to simulate different paths. The model developed for this project consists of four subsystems:

- Position Controller
- Attitude Controller
- Quadcopter Control Mixing
- Quadcopter Dynamics

A block diagram of the whole quadcopter system is provided in Figure 4.1.

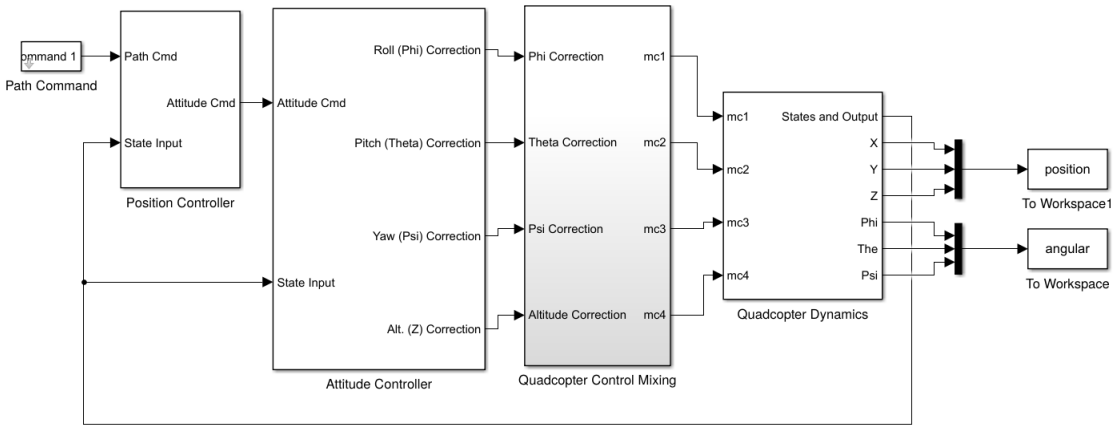

Figure 4.1: Block diagram of quadcopter

Each of the subsystems shown in 4.1 are explained in the following sections.

4.1 Position Controller Model

This subsystem implements the control laws for position control of the quadcopter. The subsystem is shown in Figure 4.2. The inputs to this system are the Path command and the State inputs. The outputs from this subsystem is the altitude command which is then sent to the attitude controller subsystem.

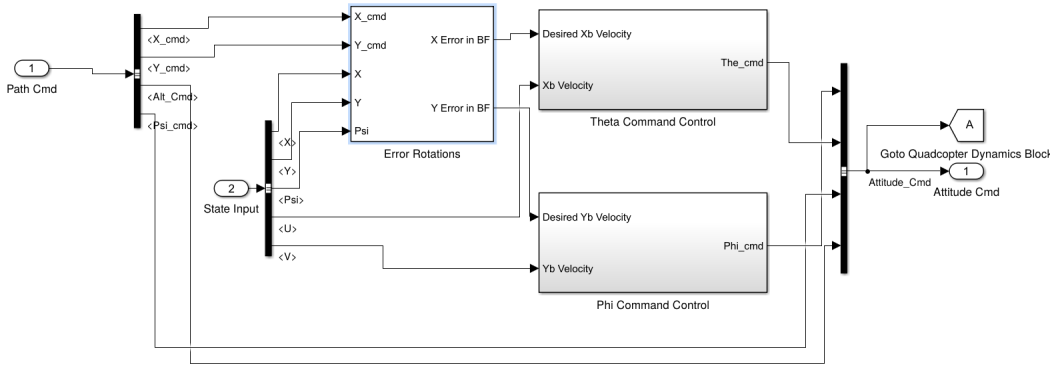

Figure 4.2: Postion Control Subsystem of the Quadcopter

The first block in the subsystem calculates the error feedback using equations 3.5 - 3.8. From these PD control laws, error is evaluated, which is then used to develop the position controller for the quadcopter. This subsystem is known as lower level controller, which is used for navigation of the quadcopter in x and y directions. Output from this subsystem is then fed into the higher-level controller for attitude stabilization of the quadcopter.

4.2 Attitude Controller Model

This subsystem implements the control laws for attitude stabilization and control of the quadcopter. The subsystem is shown in Figure 4.3. The inputs to this system are the altitude command from the position controller subsystem and the State inputs

for the closed loop quadcopter system.

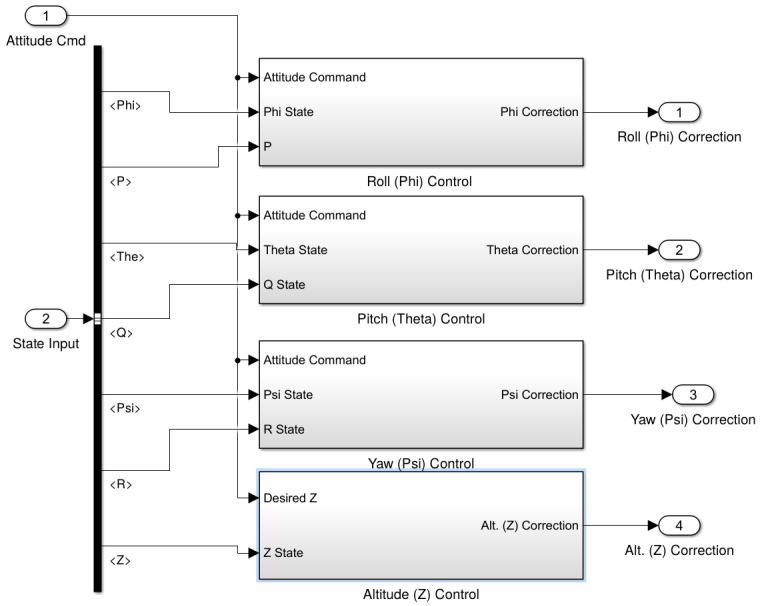

Figure 4.3: Attitude Control Subsystem of the Quadcopter

This subsystem consists of four sub blocks, each of which implements a PID controller using the control laws interpreted in equations 3.1 - 3.4. Each of these sub blocks gives a corrected value of roll, pitch, yaw, and altitude. These values are the outputs from this block which are then sent to the Quadcopter Control Mixing subsystem for further processing. This block completes the controller portion of the quadcopter. These are then used with quadcopter dynamics to navigate and follow the chosen path.

4.3 Quadcopter Control Mixing Model

Using the output from the attitude stabilization block and mixing equations 4.1 - 4.4, value of the throttle command input is computed. This subsystem is shown in Figure 4.4.

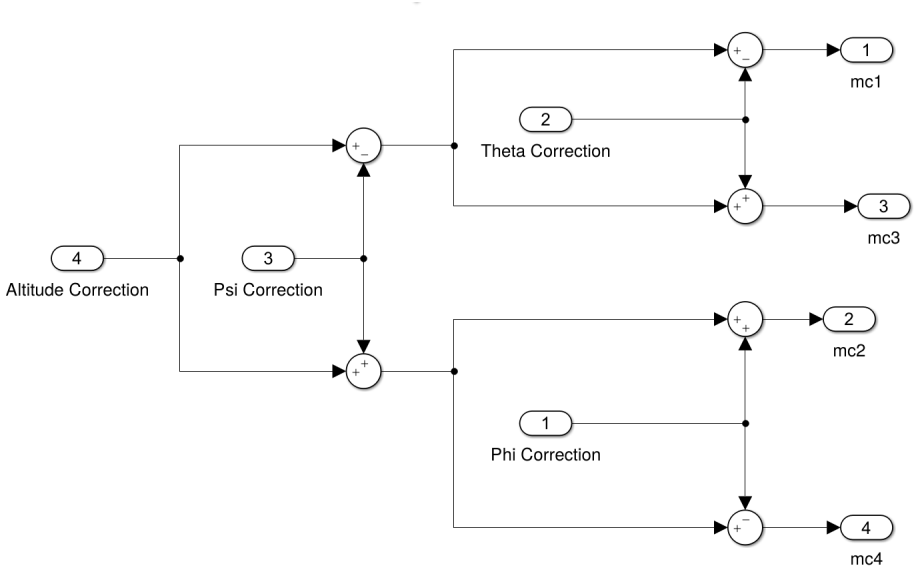

Figure 4.4: Quadcopter Control Mixing Subsystem

Quadcopter Control Mixing block takes the correction values for roll, pitch, yaw, and Z, and then mixes them by letting each correction to be sent to the correct motor accordingly. The equations for the output are

$$mc1 = \delta z - \delta\varphi - \delta\theta, \tag{4.1}$$

$$mc2 = \delta z - \delta\varphi + \delta\theta, \tag{4.2}$$

$$mc3 = \delta z + \delta\varphi + \delta\phi, \tag{4.3}$$

$$mc4 = \delta z + \delta\varphi - \delta\phi. \tag{4.4}$$

where $mc1 - mc4$ are percentage throttle command for motors 1-4. Please note that the output of the quadcopter control mixing block is in terms of percent throttle and it allows tuning of the controller, making it a more realistic quadcopter system with properly tuned controller parameters.

4.4 Quadcopter Dynamics Model

In the quadcopter dynamics subsystem, the motor dynamics block restricts the input commands between 0% (minimum) and 100% (maximum) of the throttle. It simulates the motor cutoff behavior at negligible throttle signal and uses the linear relation from equations 4.1 - 4.4 to compute percentage throttle signal considering first order delay function. The output of this block gives the RPM for each motor at any given time. The disturbance block in this subsystem just adds external disturbance effects such as wind forces on the quadcopter for more realistic behavior in simulation. The quadcopter dynamics block is shown in Figures 4.5 and 4.6

Now the heart of the entire simulation where the quadcopter dynamics are modeled.

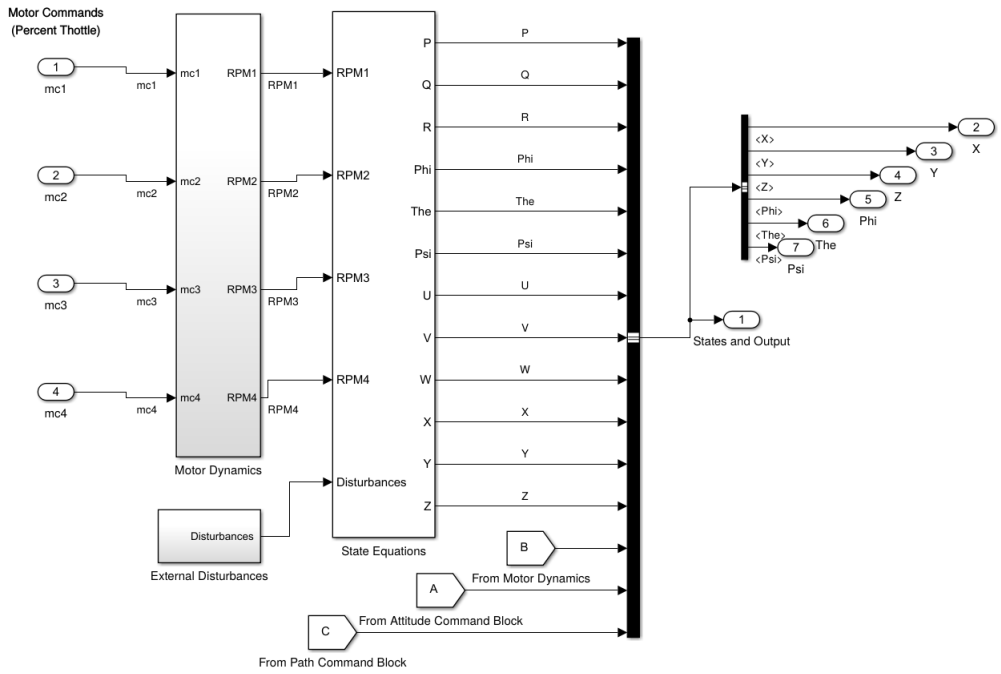

Figure 4.5: Quadcopter Dynamics Subsystem

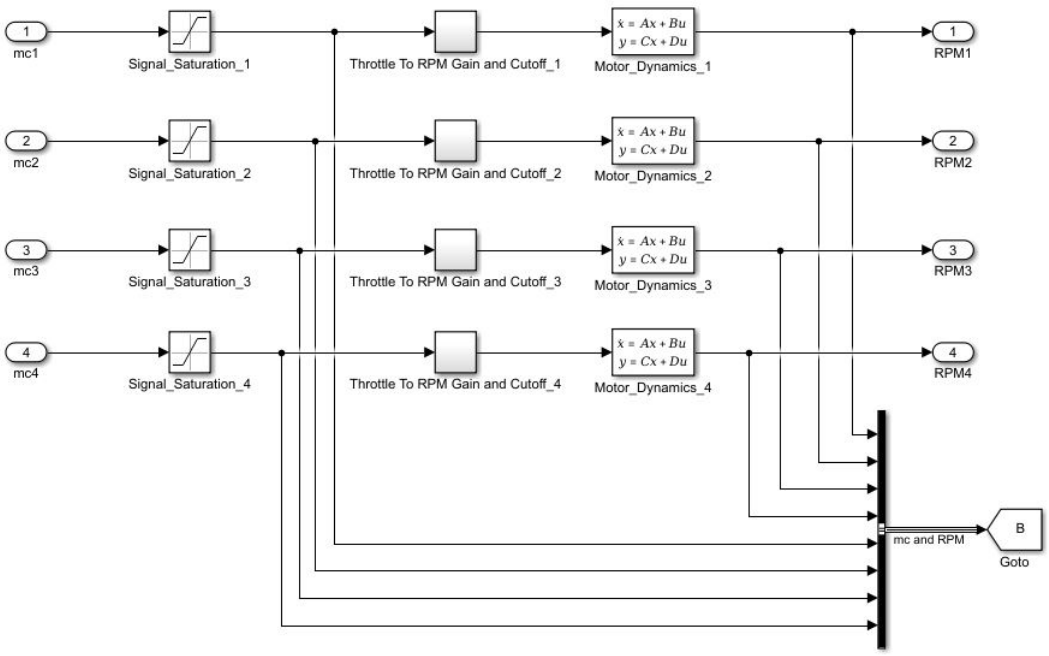

Figure 4.6: Quadcopter Motor Dynamics Subsystem

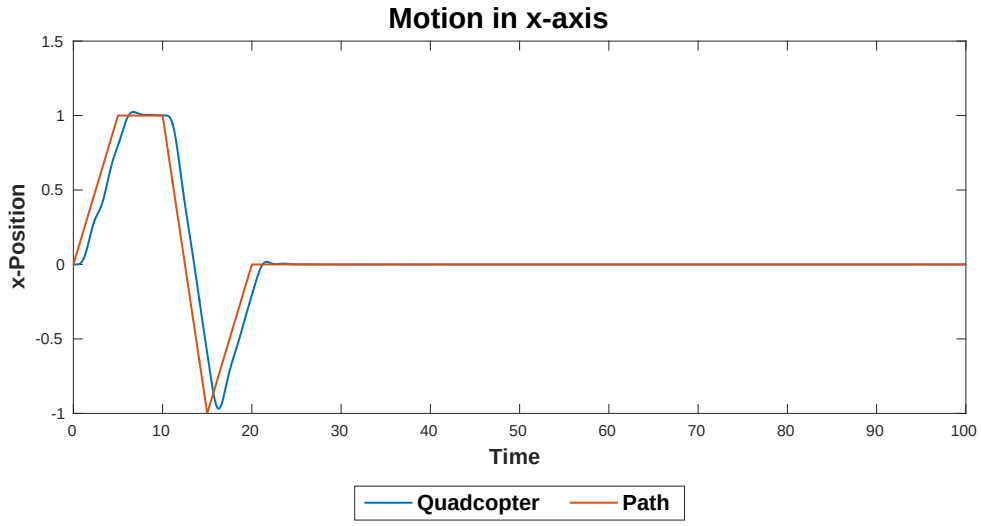

Figure 4.7: Plot of Position Controller Action in x direction

The state equations execute a level 2 S-Function in MATLAB code using all the model dynamic equations as derived in Chapter 2. The output of this subsystem are all the states and outputs, which are fed back making this as a closed loop control for the quadcopter. This quadcopter model with the control method is then validated by moving it through a pre-defined path. The simulation results of x, y and z positions are shown in Figures 4.7, 4.8, and 4.9 .

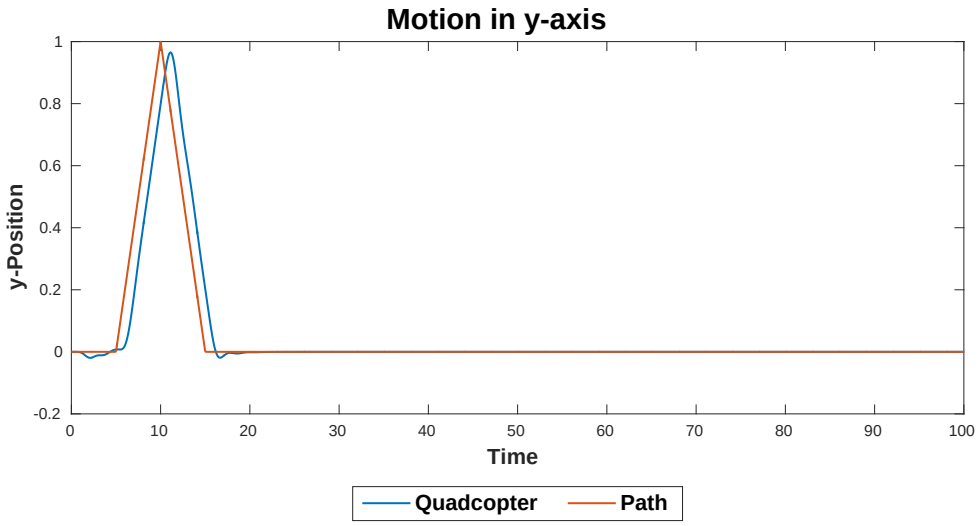

Figure 4.8: Plot of Position Controller Action in y direction

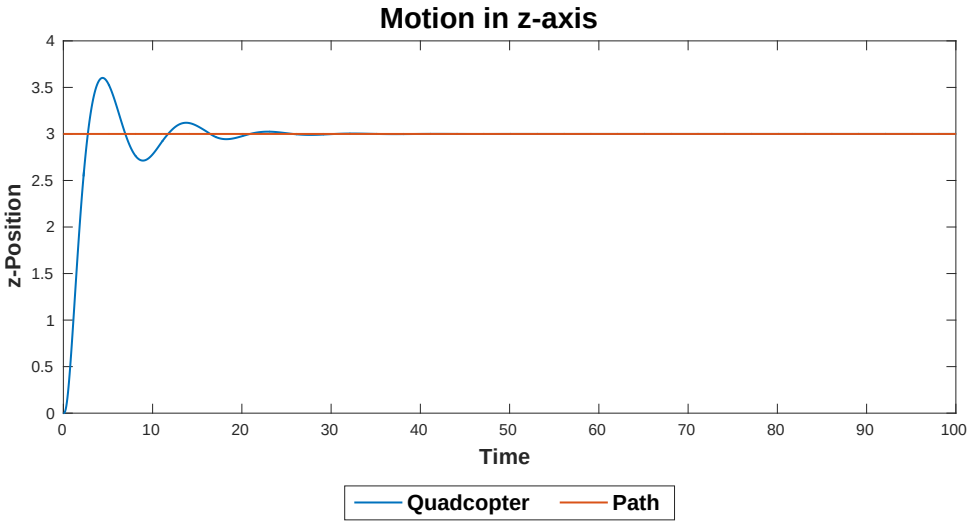

Figure 4.9: Plot of Position Controller Action in z direction

Chapter 5

Vision Based Tracking and Detection

In this section, we develop and validate the method for tracking the quadcopter using a 2-D image obtained from the camera mounted on the ground vehicle. The camera on the ground vehicle is facing up and forward capturing images of the moving quadcopter above it. The advantage of this setup, where the camera is on the ground vehicle is that the quadcopter can be easily kept in the field of view of the camera. In addition, limits on the data processing and the extra payload due to camera weight are offloaded to the ground vehicle. It is assumed that our quadcopter flies with a known constellation of LEDs or markers. Also, we know the dimensions of our quadcopter. This allows us to easily recover the 3-D pose from a single 2-D frame. The known

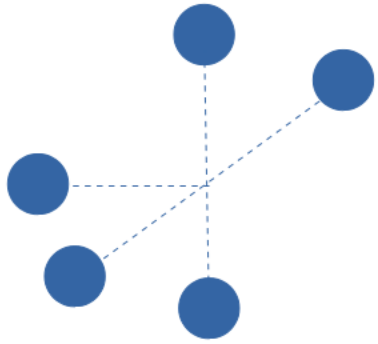

Figure 5.1: Marker constellation of 5 elements arranged on a 3D axis with length of 25 cm for each leg.

constellation of the quadcopter is assumed to be fixed with five elements arranged in a 3-D plane with axis length of 25 cm (arm length of the quadcopter), as shown in the Figure 5.1. Four elements of the constellation are arranged in a plane in the form of a square and the fifth element is set out of the plane. It is this fifth element that helps in resolving the ambiguity between range and yaw for the quadcopter. Also, the 3-D pose estimation problem can also be resolved by tracking with time and, by including the fifth element, it improves the accuracy of the estimation.

Here we have considered a single camera to track the quadcopter. The camera records a two-dimensional image from a three-dimensional scene. Let's give an outline of the forward model describing the transformations of the five elements from a three-dimensional coordinate system to the two-dimensional image coordinate system. In Figure 5.2, the target coordinate system is defined by the pose of the elements on the

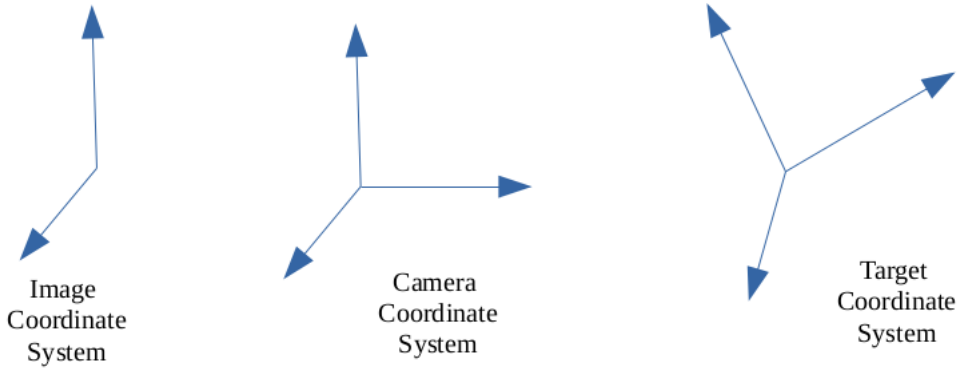

Figure 5.2: Different Coordinate Systems

quadcopter, while the three-dimensional camera coordinate and the two dimensional image coordinate system are defined by the camera orientation.

5.1 Forward Camera Model

Please note that the five marker locations are known and fixed in the target coordinate system. The transformation can be described by homogeneous coordinates. This simplifies the translational step which is easily reduced to a simple matrix multiplication step. The transformation from locations of the five markers in the target coordinate system to the recorded positions of the markers in the two-dimensional image can be explained by

$$x = x' \frac{f}{z'} and y = y' \frac{f}{z'}, \tag{5.1}$$

$$\begin{bmatrix} x' \\ y' \\ z' \end{bmatrix} = \begin{bmatrix} f & 0 & 0 & 0 \\ 0 & f & 0 & 0 \\ 0 & 0 & 1 & 0 \end{bmatrix} \begin{bmatrix} \mathbf{R}_{3x3} & \mathbf{T}_{3x1} \\ 0 \quad 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} U \\ V \\ W \\ 1 \end{bmatrix} = \mathbf{P} * \mathbf{R} * \mathbf{C}. \tag{5.2}$$

In the above two equations, x *and* y are the actual image coordinates and x', y' *and* z' are the homogeneous image coordinates, P is the projection matrix, R is the rotation matrix and C expresses the marker coordinates in the target coordinate system. The target coordinate system given by U, V, W are rotated and translated to give the coordinates in three dimensions, with respect to the camera frame of reference. Then it is scaled according to the focal length of the sensor and projected into a two-dimensional image. The coordinates are in two dimensional homogeneous coordinates with an offset of z' over z axis. The homogeneous coordinates are re-scaled such that $z' = 1$.

5.2 Inverse Camera Model

When performing the inverse to get three-dimensional pose from a two-dimensional image, we don't know the z' offset value, so we get a non-linear optimization problem.

The six degree of freedom optimization problem is

$$[\phi, \theta, \varphi, T_x, T_y, T_z] = argmin\left(||p_{img} - p_{est}(\phi, \theta, \varphi, T_x, T_y, T_z)||_2^2\right) \tag{5.3}$$

where $\phi, \theta \ and \ \varphi$ are the roll, pitch, and yaw of the quadcopter, respectively. $T_x, T_y \ and \ T_z$ are the target translation, p_{img} is (5×2) matrix of the measured marker coordinates and p_{est} is the estimated marker coordinates in a single frame.

The coordinates of each marker are known for both the target and image coordinate system. The values for rotation matrix R_{3x3} and translation matrix T_{3x1} are unknown. The rotation matrix R_{3x3} is characterized by the roll, pitch, and yaw of the quadcopter; therefore, these three angles are sufficient to evaluate the matrix.

5.3 Kalman Filter Tracking

Measurements of the relative position and orientation between the quadcopter and the moving ground vehicle obtained from image processing and estimation are noisy. These measurements along with the noisy signals are used to estimate the target's position and velocity in the inertial frame.

The single frame pose estimations of the UAV can be combined to form a track of the UAV's motion over time. By fusing the single frame results and applying a prior

model for the expected motion characteristics of the platform, we can mitigate errors in the single frame pose estimates. The conventional method for fusing a discrete set of motion measurement is the well-known Kalman filter. As such, the Kalman filer is applied here. A Kalman filter is designed to predict the position of the quadcopter at the next time step. This helps in cases of temporary loss of image or target occlusion. The Kalman filter can be designed considering constant velocity model or constant acceleration model. In this case since the acceleration is assumed to be negligible and velocity of the quadcopter is changing we choose the constant acceleration model. A Kalman filter works in two steps: the prediction step and the update step. In the prediction step, the Kalman filter gives the estimate of the current state variables including uncertainties. Once the next measurement is obtained, the estimates are calculated using a weighted average.

A Kalman filter takes the dynamic model of the system with control inputs and measurements to give an estimate of its state for the next step. It is a recursive estimator and hence for the current state update at time (t) only the estimated state at time (t-1) and measurement at time (t) are needed. For this project, a Kalman filter is implemented for tracking the five marker constellation of the quadcopter. Designing this filter is important and focuses on the following three features to get the best possible tracking

1. Prediction for location at next time step;

2. Noise reduction caused by inaccurate detections;

3. Proper tagging for multiple objects to their respective tracks.

Before implementing the filter, detecting the five markers from a two-dimensional image is important. For this we simply subtract the Foreground from the image and detect all the circular objects in the image with a specified range for radius. This process is not ideal and hence introduces noise in the detected measurement, but it is simple and efficient. These five marker positions are then used with the inverse model for the camera to get the position and orientation of the quadcopter.

To overcome error in measurement, a discrete Kalman Filter is used. The Kalman filter determines the position whether it is detected in the image or not. This helps if we do not receive a position update or the quadcopter can not be detected in the image. For such scenarios the Kalman filter uses the previous estimate to give a best estimate of the position. So, when the position is estimated the Kalman filter predicts its state and then uses fresh measurements to correct its state and produce a filtered position.

A discrete Kalman filter is formulated as follows. A state vector for target's position and its orientation to be estimated is defined as $x = [X_t^T, \alpha]^T$. The notation $\hat{x}_{n|m}$ is the estimate of x at time n for measurements up to time $m \leqslant n$. The state of the

filter is represented by $\hat{x}_{k|k}$, the posteriori state estimate, and $\hat{P}_{k|k}$, the posteriori error covariance matrix which gives a measure of the estimate's accuracy. The two steps for the filter, Predict and Update, are as follows.

Predict:

$$\hat{x}_{k|k-1} = F_k \hat{x}_{k-1|k-1} + B_k u_k, \tag{5.4}$$

$$\hat{P}_{k|k-1} = F_k \hat{P}_{k-1|k-1} F_k^T + Q_k. \tag{5.5}$$

Update:

$$y_k = G_k x_{k-1|k-1} + n_k, \tag{5.6}$$

$$S_k = G_k P_{k|k-1} G_k^T + R_k, \tag{5.7}$$

$$K_k = P_{k|k-1} H_k^T S_k^{-1}, \tag{5.8}$$

$$\hat{x}_{k|k} = \hat{x}_{k|k-1} + K_k yk, \tag{5.9}$$

$$P_{k|k} = (I - K_k H_k) P_{k|k-1}. \tag{5.10}$$

where, F_k is the state transition matrix, B_k is the control-input applied to the control vector u_k, H_k is the observation matrix, n_k is the observation noise assumed to be zero mean Gaussian noise such that $n_k \sim \aleph(0, R)$, S_k is the innovation covariance matrix, K_k is the Optimal Kalman gain, and $x_{k|k}$ gives the updated state estimate.

The Kalman filter is implemented on our quadcopter system for tracking. To validate

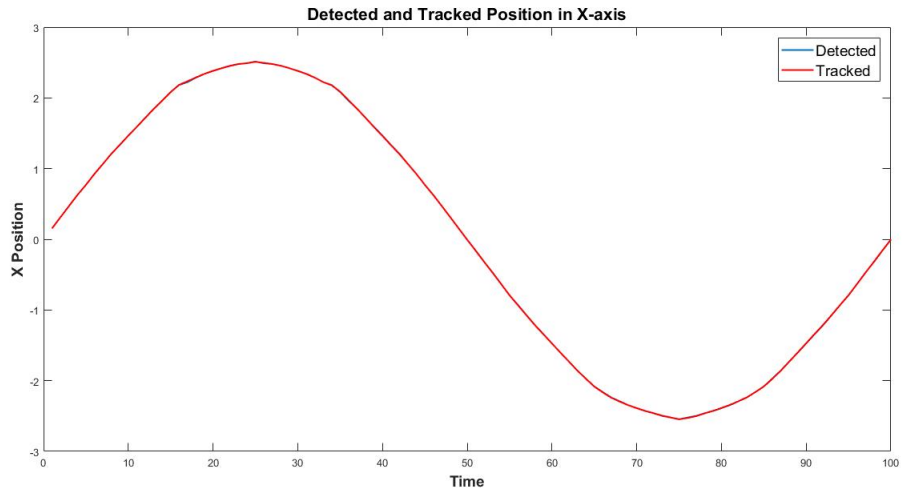

Figure 5.3: Detected and tracked position of a quadcopter in x-direction

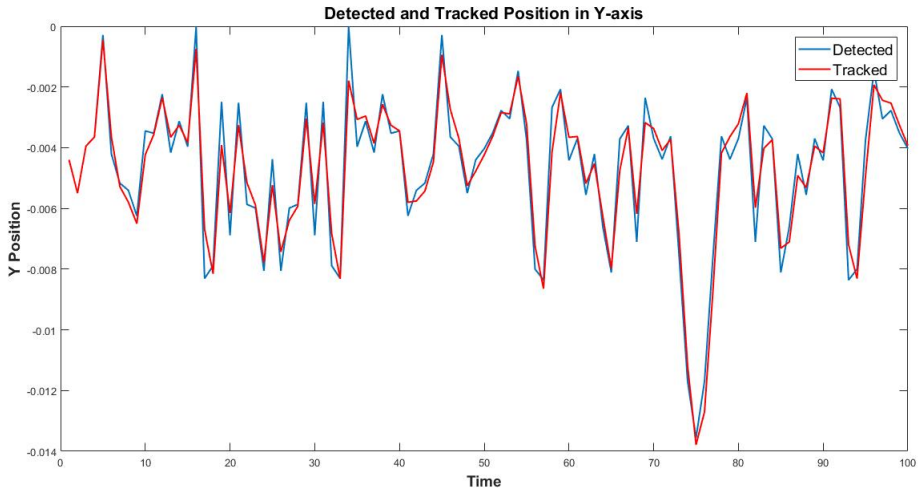

Figure 5.4: Detected and tracked position of a quadcopter in y-direction

the detection and tracking algorithm we developed a trial run. A simulated video of a randomly moving quadcopter was fed into our algorithm and detection and tracking was implemented for each single frame. The results obtained from this validation are shown in Figure 5.3, 5.4 and 5.5. From these figures, it can be easily inferred

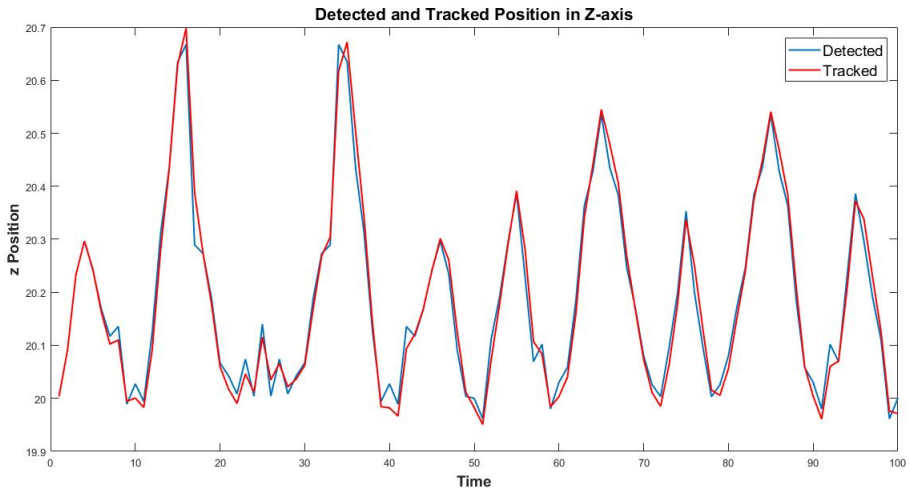

Figure 5.5: Detected and tracked position of a quadcopter in z-direction

that the detection and tracking algorithm using the Kalman filter works well even with rapidly changing y and z position. The entire simulation also was done with substantial measurement and observation noise. We ran multiple simulations to get an optimum set of parameters to configure our Kalman filter. The error between the tracked and detected position is negligible and our estimation algorithm works efficiently.

Chapter 6

Closed Loop Feedback with Ground Vehicle

Now that we have a working position control for our quadcopter, a camera model and an efficient pose estimation algorithm to detect and track the moving quadcopter by using camera inputs only, our next step is to put this algorithm in closed loop feedback architecture with a mobile ground vehicle. In this Chapter we develop a model a moving ground vehicle and then put it in the closed loop system with the quadcopter. The camera is on-board the ground vehicle and the ground vehicle is in motion. One assumption that we make for this closed loop feedback system is that we are in a perfect communication world and the path inputs sent from the ground vehicle reach the quadcopter immediately. First, we design a moving ground vehicle

and then develop our closed loop feedback system with the quadcopter.

6.1 Moving Ground Vehicle

Several types of ground vehicles have been developed by over the past decades. Ground vehicles can navigate on the ground autonomously or by remote control. These can be deployed to places where human operations are dangerous and inconvenient. As per its application these can be developed with suitable platform, sensors, controllers, communication links and guidance interface.

The platform for the vehicle is developed according to the terrain in its area of operation. As the primary objective for the vehicle is navigation, several sensors are placed on the platform. These include, but are not limited to, wheel encoders, compasses, gyroscopes, global positioning system (GPS), and cameras. As per requirement, a control system is used for the vehicle to perform decision making process while navigating. Communication links and guidance interface are needed to collaborate with other systems and possible remote operation for the ground vehicle. The vehicle takes the sensor inputs and uses the control algorithms to determine next action for a predefined task.

For this problem, we have considered the simplest form of the ground vehicle. Here we

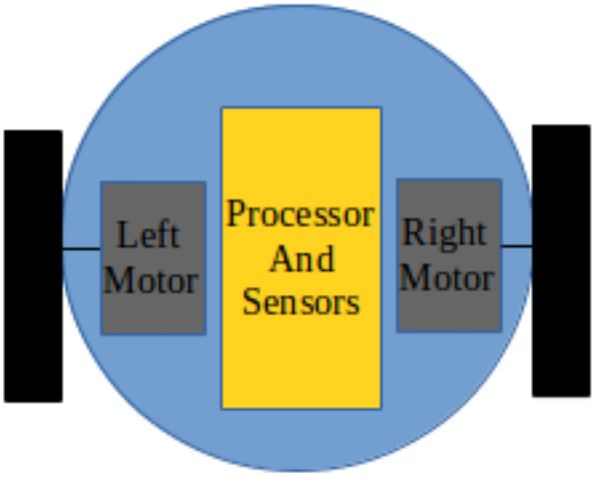

Figure 6.1: Two Wheel Differential Drive

use a two-wheel differential drive ground vehicle with a pure pursuit control algorithm. A differential drive wheeled vehicle is a drive system which has independent actuators for each wheel; in this case, we have two independent motors for the movement of the vehicle. Both the motors are mounted on a common axis and each can be independently driven clockwise or counter-clockwise for the motion of the vehicle. A two-wheel differential drive vehicle is shown in the Figure 6.1.

For the ground vehicle, it is common to use a two-dimensional planar model. In this part of the thesis, we make use of the forward kinematics of a robot to build a two-wheeled differential drive ground vehicle to be used for the closed loop feedback control of the quadcopter.

To design and control the motion of the ground vehicle, we need to use the angular

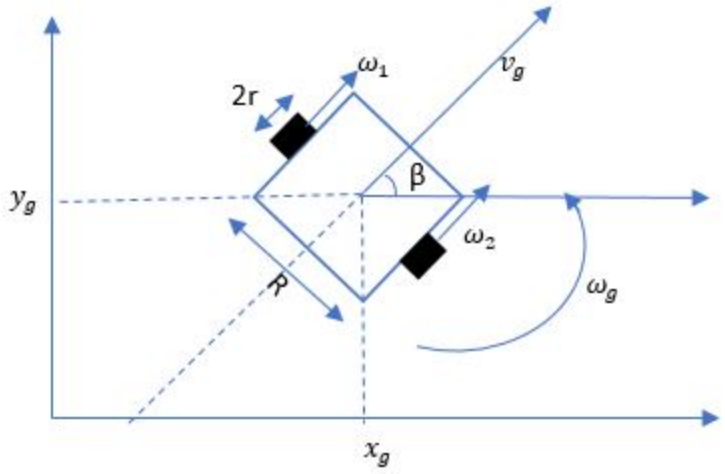

Figure 6.2: Vehicle in an Initial Frame of Reference

velocities of the wheels on the drive to calculate its linear velocity and angular velocity to give position updates for the vehicle at every instant. Figure 6.2 represents the vehicle within an initial frame of reference.

As per the kinematics of this ground vehicle, each of the two motors provide a torque to drive the wheel which is directly proportional to the velocity of the wheel. Since the velocities of the two wheels can be controlled individually, we can drive the ground vehicle to any desired position by making appropriate changes to get the resultant velocity.

In for the kinematic model of the vehicle, we define the state vector for position of the ground vehicle as $p_g = [x_g, y_g, \beta_g]^T$, where x_g and y_g represent the position of the center of mass of our two-wheel differential drive vehicle and β_g is the orientation in the body frame. Also, let us define the velocity vector for the ground vehicle as

$\mathbf{V}_g = [v_g, \boldsymbol{\omega}_g]^T$, where v_g is the linear velocity for the vehicle and $\boldsymbol{\omega}_g$ is the angular velocity for the vehicle at the center of mass. To evaluate the values for velocity vector we use the following equations,

$$v_g = r\frac{\omega_1 + \omega_2}{2}, \tag{6.1}$$

$$\omega_g = r\frac{\omega_2 - \omega_1}{2}. \tag{6.2}$$

Further, to design the kinematic model governing the motion of the ground vehicle we use the following,

$$\dot{x}_g = r\cos(\beta_g)\frac{\omega_1 + \omega_2}{2}, \tag{6.3}$$

$$\dot{y}_g = r\sin(\beta_g)\frac{\omega_1 + \omega_2}{2}, \tag{6.4}$$

$$\dot{\beta}_g = r\frac{\omega_2 - \omega_1}{R}. \tag{6.5}$$

where R is the distance between the two wheels, r is the radius of the wheel, and ω_1 and ω_2 are the angular velocities for each wheel. The value obtained from equation 6.2 for $\boldsymbol{\omega}$ is the derivative of the orientation β_g of the ground vehicle.

For position update of the moving vehicle in discrete time steps, we have the following,

$$x_{g|t+\delta t} = x_{g|t} + \dot{x}_{g|t} * \delta t, \tag{6.6}$$

$$y_{g|t+\delta t} = y_{g|t} + \dot{y}_{g|t} * \delta t. \tag{6.7}$$

The above two equations give the updated position, $[x_{g|t+\delta t}, y_{g|t+\delta t}]$ at time step of δt, for the ground vehicle if the position at time t is known to be $[x_{g|t}, y_{g|t}]$. For our problem statement, we need to make sure that the vehicle follows the desired path. In short, we need to develop a position controller for this ground vehicle. This can be easily achieved by using a simple PD controller to maintain the orientation of the vehicle which then directly effects the linear velocities, shown in equations 6.3 and 6.4. The following equations are used to control the orientation of the ground vehicle:

$$u = k_{pg}(\beta_d - \beta_g) + k_{dg}(\dot{\beta}_d - \dot{\beta}_g), \tag{6.8}$$

$$\beta_{g|t+\delta t} = u * \delta t + \beta_{g|t}, \tag{6.9}$$

where k_{pg} and k_{dg} are the PD controller parameters, β_d is the desired orientation, and u is the control input to be applied for update in the present orientation. This algorithm for the position control of the vehicle is known as the pure pursuit to goal algorithm. Using the model described here, the dynamic model of the ground vehicle was developed in MATLAB.

6.2 Joint Central Control for Closed Loop Operation

We have developed a moving ground vehicle and also in the Chapter 4, we developed the model for quadcopter navigation. Both systems have an intact position controller on them individually. Also, in Chapter 5 we developed the vision based tracking and detection algorithm. Our final step is to put all these pieces together to develop a closed loop feedback system for position control of a quadcopter from a moving ground vehicle.

The main objective is for the quadcopter to follow the ground vehicle using a single monocular camera input for vision based estimation of the position of the quadcopter. The ground vehicle is moving with a constant speed in the forward direction. This leads to an another issue that needs to be addressed before we put these systems in a closed loop: the relative position between the two vehicles. Since the quadcopter and the ground vehicle are both moving, we need to work our way through the relative position updates and not just the direct position updates for our vision based estimation to give proper results.

For this we need to develop a joint decentralized controller for the quadcopter and the ground vehicle to interact and share the position updates as required. The goal

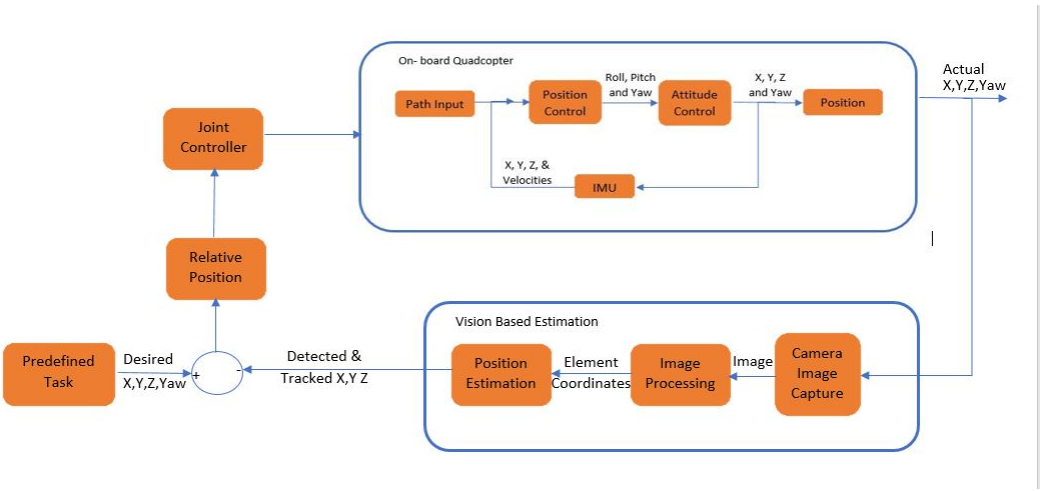

Figure 6.3: Closed Loop Feedback Control Architecture

of this controller is to drive the relative position error between the two vehicles to zero. By stabilizing the position error, we ensure that both the vehicles will drive together at some specified speed. For this we implement a PD controller to minimize the error for the three dimensional position update for the quadcopter. With all the elements in place the closed loop feedback controller for our problem can be developed as shown in Figure 6.3.

The control architecture is developed and we validate our model by running simulated runs in MATLAB and Simulink. The results of the simulation are shown in the next chapter.

Chapter 7

Results and Discussion

To verify our mathematical model developed along with the control laws and vision-based estimation algorithms for a quadcopter to be navigated from a ground vehicle, various simulations were performed. The main objective of each simulation was to understand and access the performance of our closed loop feedback architecture for all three axes. The simulations were also helpful in tuning the parameters of the joint centralized controller. This main controller works to minimize the error in the relative position of the quadcopter with respect to the moving ground vehicle.

For first two simulations we keep the quadcopter at a hover position i.e., at a constant z coordinate with respect to the ground vehicle and vary its position in y and x direction one by one to access the performance of the joint centralized controller.

In the third set of simulation we vary both x and y coordinate simultaneously to better understand the interdependence. In the fourth simulation scenario an improved controller is developed to overcome the error caused due to interdependence of x and y position controllers on the quadcopter. After these four simulations we obtain an optimal parameter set for the joint controller and in the fifth simulation, we test our system by varying the desired position in all three axes. This simulation set shows the overall performance of our system. These simulation validate the joint controller performance and parameters for varying x, y, and z.

Now to better understand the performance of our closed loop feedback system for varying camera parameters next set of simulations are carried out. For these simulations, we increase and decrease the focal length and resolution for the camera model and analyse the system performance. All the results of our simulations with varied input parameters are shown in the following subsections.

7.1 x and z Constant while y Varies

Figure 7.1 shows the detected path from the camera, tracking result from the Kalman filter, motion of the ground vehicle, path for the quadcopter, and the desired path for the quadcopter in the y direction. Figure 7.2 shows the mean squared error(MSE) in the quadcopter position and the desired position in the y direction. Figure 7.3 shows

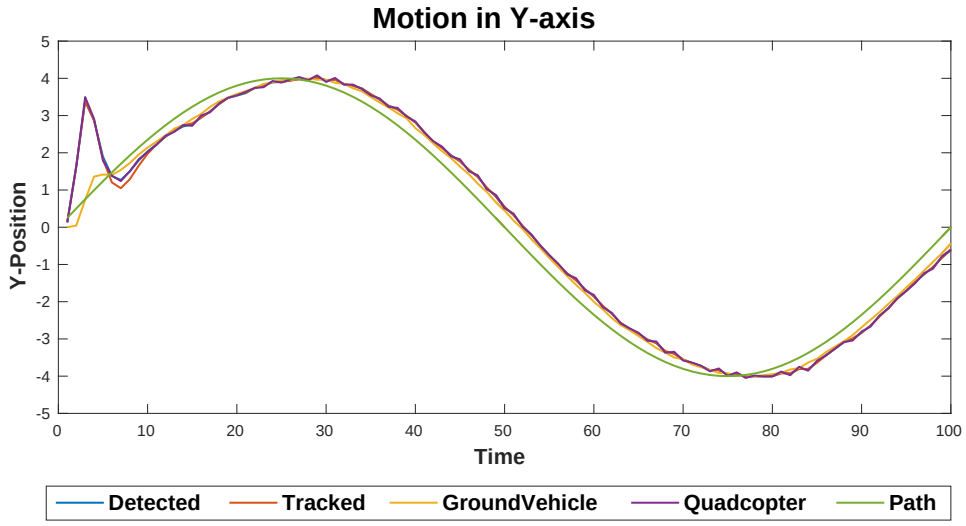

Figure 7.1: Case 1:Results for Closed Loop for Varying y

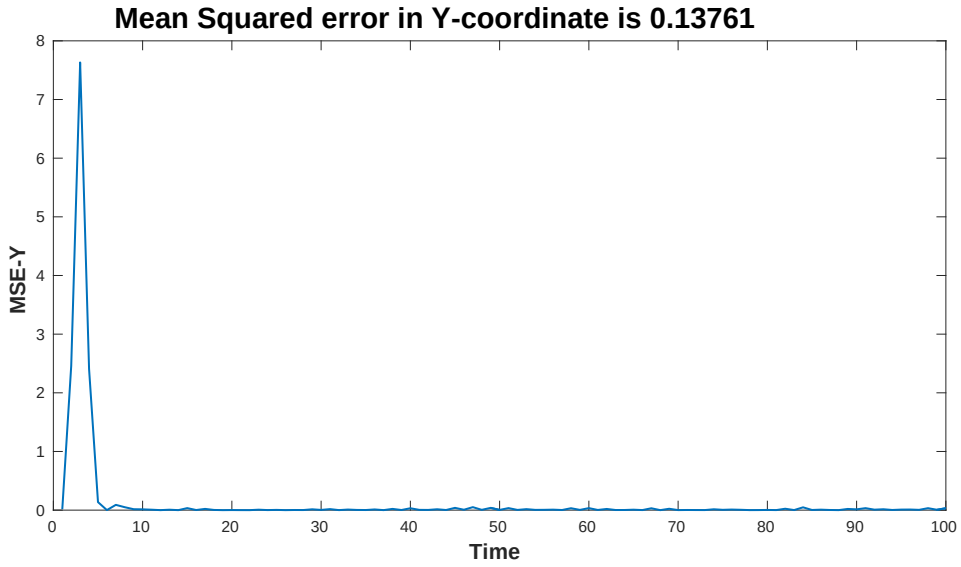

Figure 7.2: Case 1: Mean Squared Error for Varying y

the detected path from the camera, tracking result from the Kalman filter, motion of the ground vehicle, path for the quadcopter, and the desired path for the quadcopter in x direction. Figure 7.4 shows the MSE in the quadcopter position and the desired

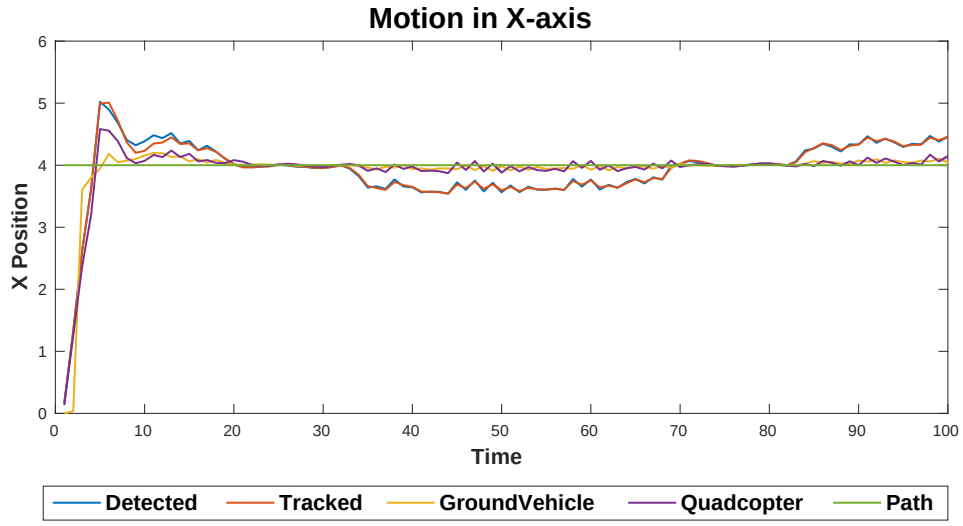

Figure 7.3: Case 1: Results for Closed Loop when $x = 4$

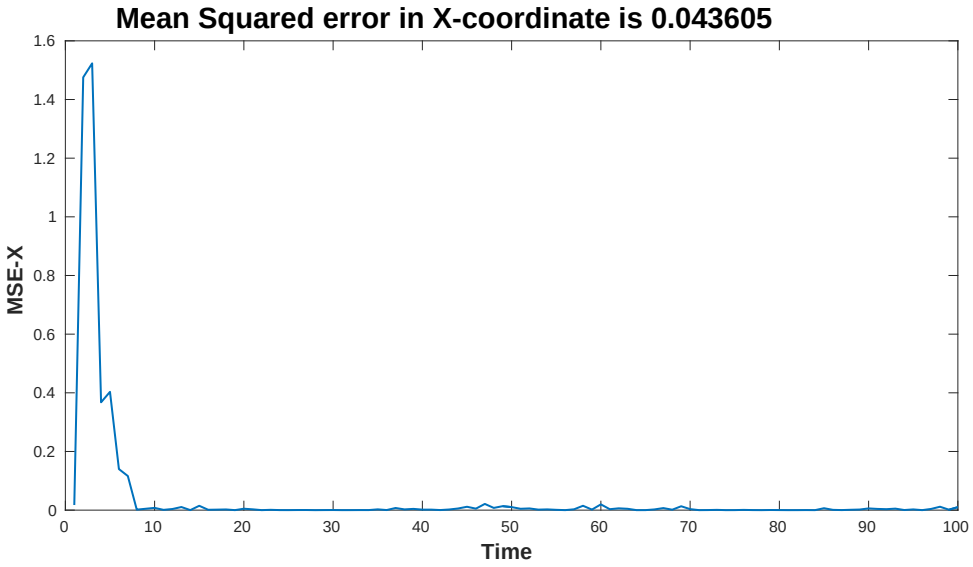

Figure 7.4: Case 1: Mean Squared Error for x-axis

position in the x direction. Figure 7.5 shows the detected path from the camera, tracking result from the Kalman filter, position of the ground vehicle in the z direction which is always $z = 0$, and the desired path for the quadcopter in the z direction.

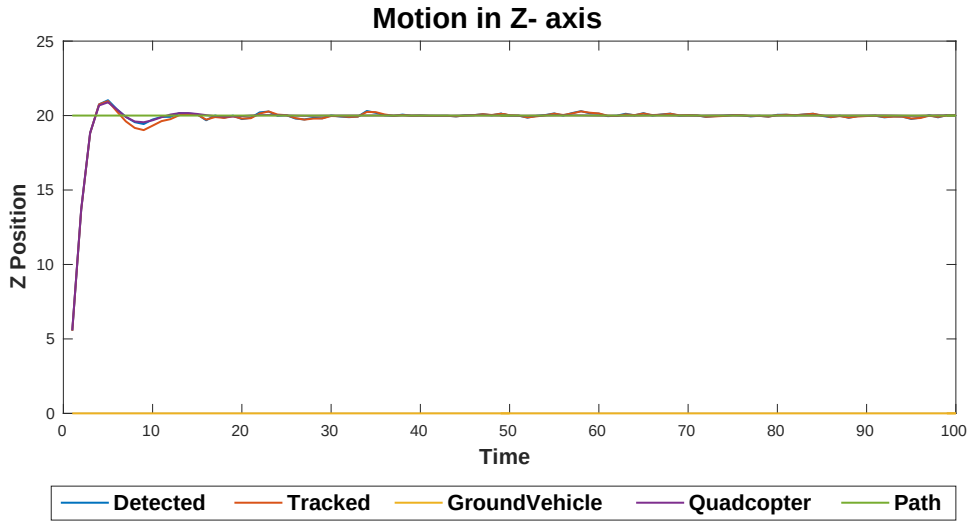

Figure 7.5: Case 1: Results for Closed Loop when $z = 20$

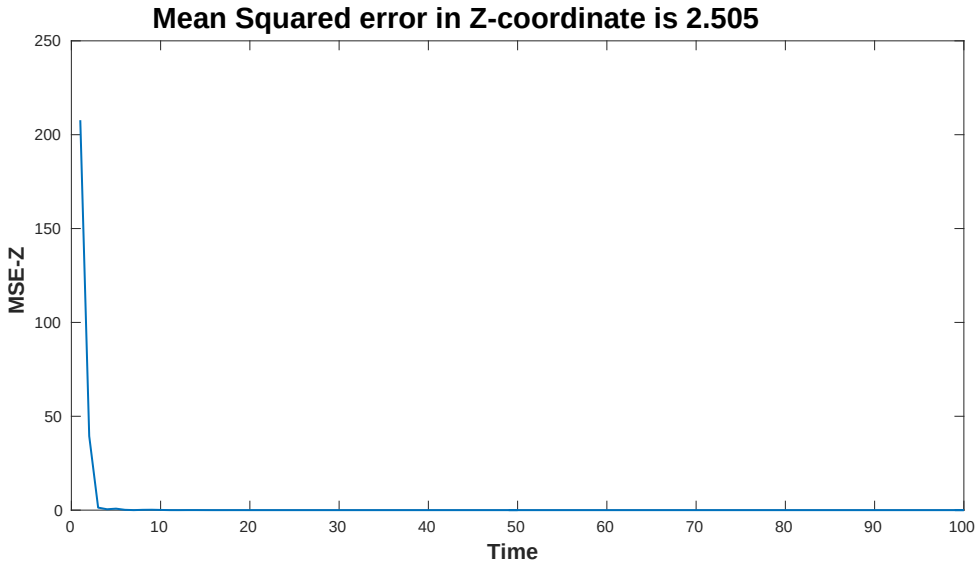

Figure 7.6: Case 1: Mean Squared Error for z-axis

Figure 7.6 shows the MSE in the quadcopter position and the desired position in the z direction. Please note that for each simulation case we obtain similar six plots which are used to analyse the performance of our system. Three plots indicating the

motion of the ground vehicle and the UAV for a given path and other three showing the squared error in the desired position of the UAV and its actual position for each x, y, and z axis.

In the Figures 7.1 -7.6 we can see that the quadcopter follows the ground vehicle closely with minimal error in y direction. For all three axes we see that the system takes 4 seconds to settle and reach to the position as instructed from the initial position of (0,0,0).From that time on-ward the quadcopter closely follows the y-coordinate as needed. If we ignore the first 4 seconds of this simulation then the MSE values for each axes is computed as $x_{MSE} = 0.0139$, $y_{MSE} = 0.0377$, and $z_{MSE} = 0.0205$. These values are extremely small and confirms the performance of our feedback system. In this case we have kept the x and z position to be constant and for the z-Axis the quadcopter maintains the hover position exactly with negligible error, but we notice a small dip in the x position at around 40 seconds into the simulation. This occurs because at this time the y-coordinate changes steeply and thereby effecting the position control for x-axes. This change in y effects the position controller of the x direction because they are dependent on yaw (φ) correction making them interdependent. Overall the closed loop controller seems to be working efficiently in the y direction. The ground vehicle follows the instructed path and the quadcopter follows the ground vehicle using the relative pose sent to it.

7.2 y and z Constant while z Varies

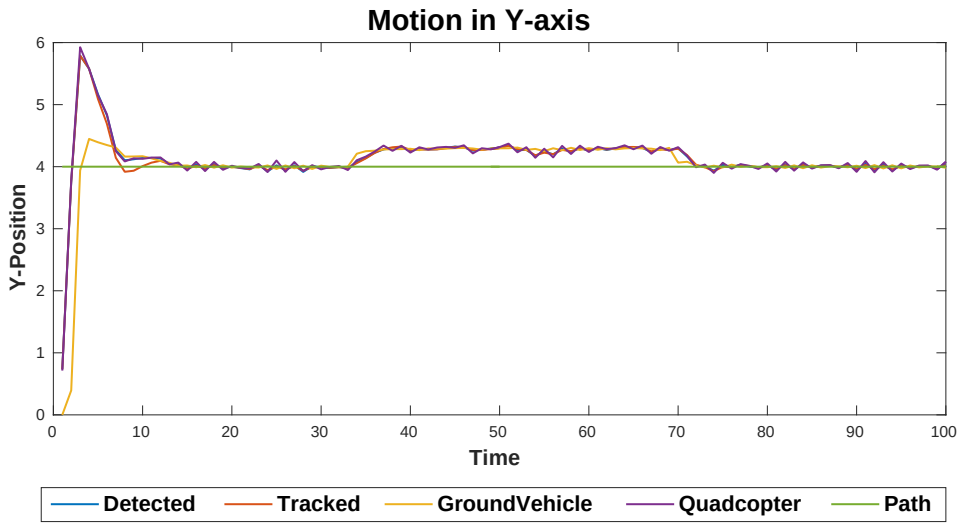

Figure 7.7: Case 2: Results for Closed Loop when $y = 4$

From the Figures 7.7 − 7.12, we can infer that the quadcopter follows the ground vehicle closely with minimal error in x direction. For all three axes we see that the system takes 4 seconds to settle and reach to the position as instructed from their initial position of (0,0,0). These 4 seconds are like the settling time for the quadcopter and are essential for the quadcopter to reach a hover position. From that time on-ward the quadcopter closely follows the x-coordinate as needed except whenever there is a substantial and steep change in the y position input. If we ignore the first 4 seconds of this simulation then the MSE values for each axes is computed as $x_{MSE} = 0.0171$, $y_{MSE} = 0.0267$, and $z_{MSE} = 0.0169$. These values are extremely

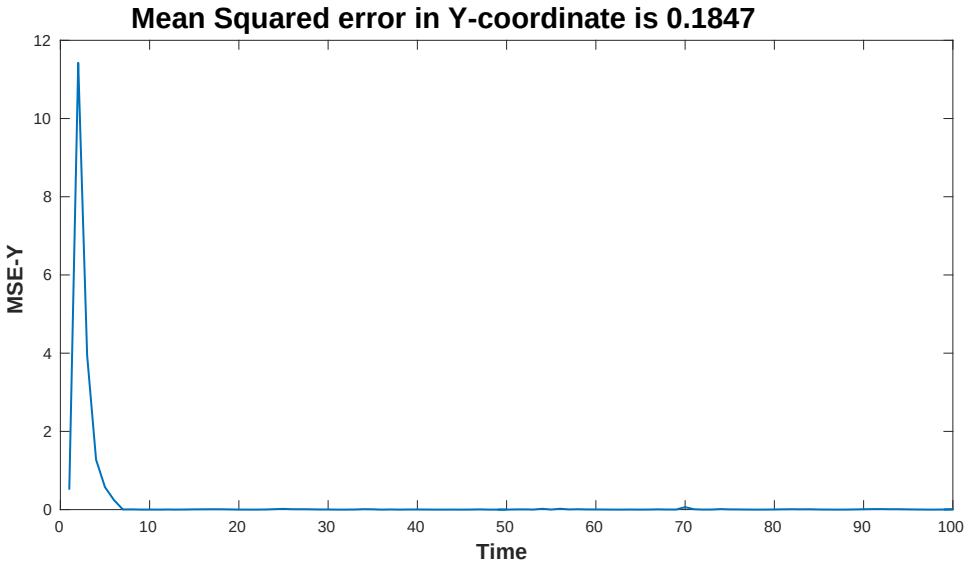

Figure 7.8: Case 2: Mean Squared Error for y-axis

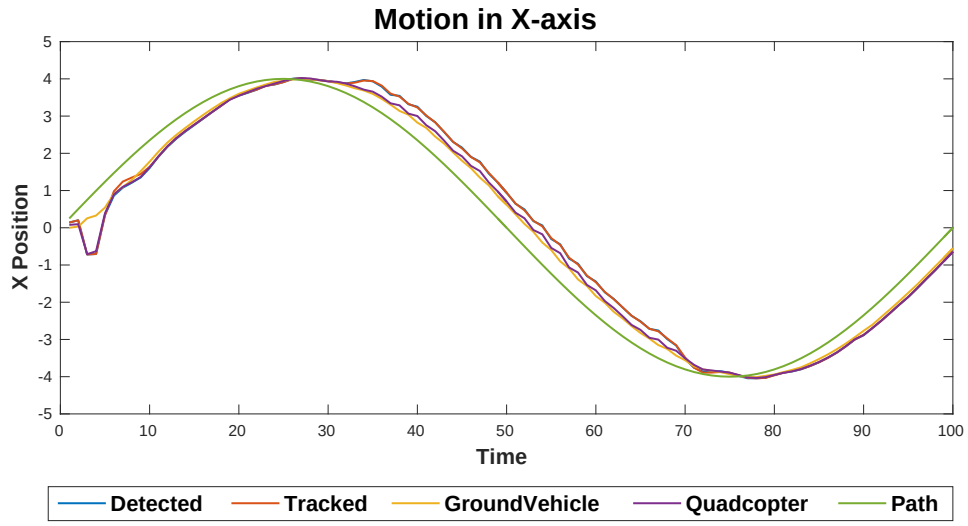

Figure 7.9: Case 2: Results for Closed Loop when x Varies

small and confirms the performance of our feedback system. In this case we have kept the y and z position to be constant and for z-axis the quadcopter maintains the hover position exactly with negligible error, but for y-coordinate we notice a small

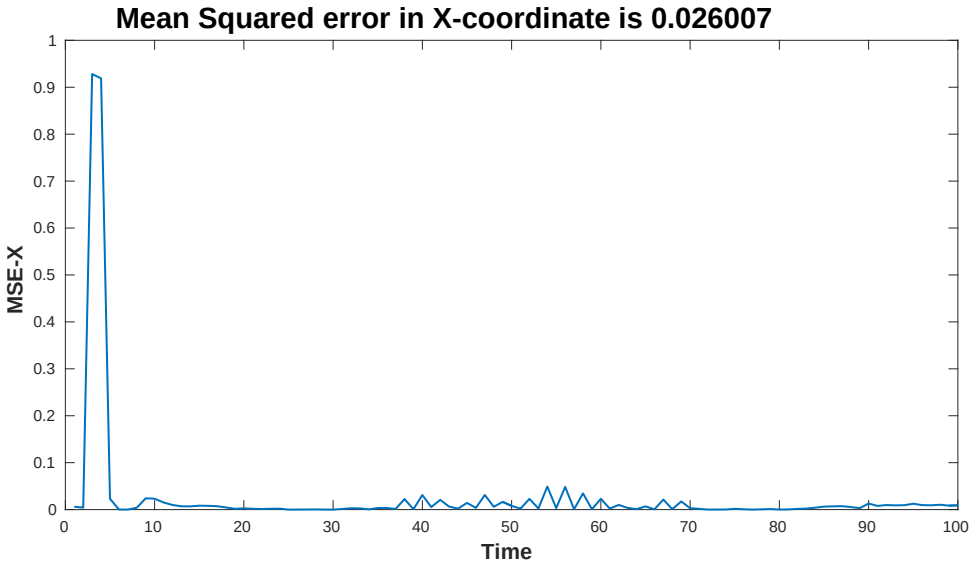

Figure 7.10: Case 2: Mean Squared Error for x-axis

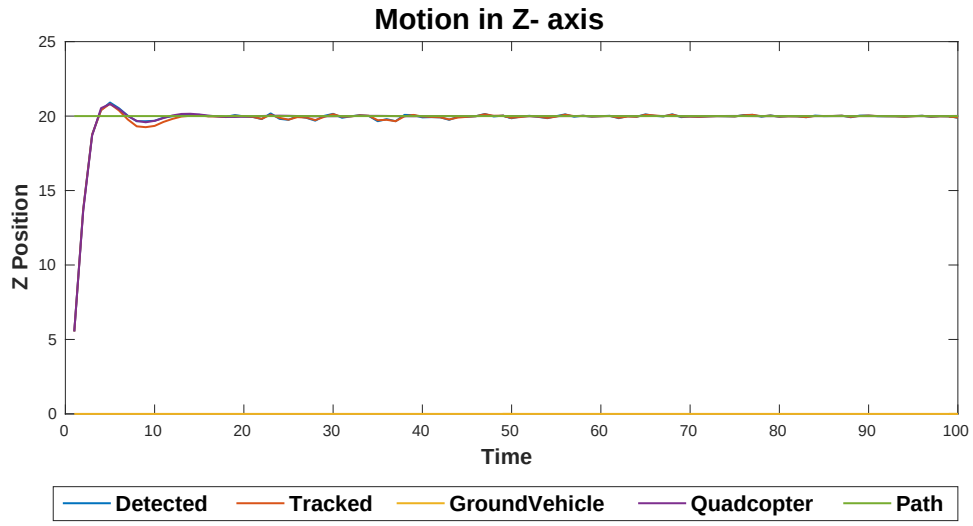

Figure 7.11: Case 2: Results for Closed Loop when $z = 20$

dip in the position at around 40 seconds into the simulation. This occurs because at this time the x-coordinate changes steeply and thereby effecting the position control for y-axis. The reason for this is as explained in the previous simulation. Overall the

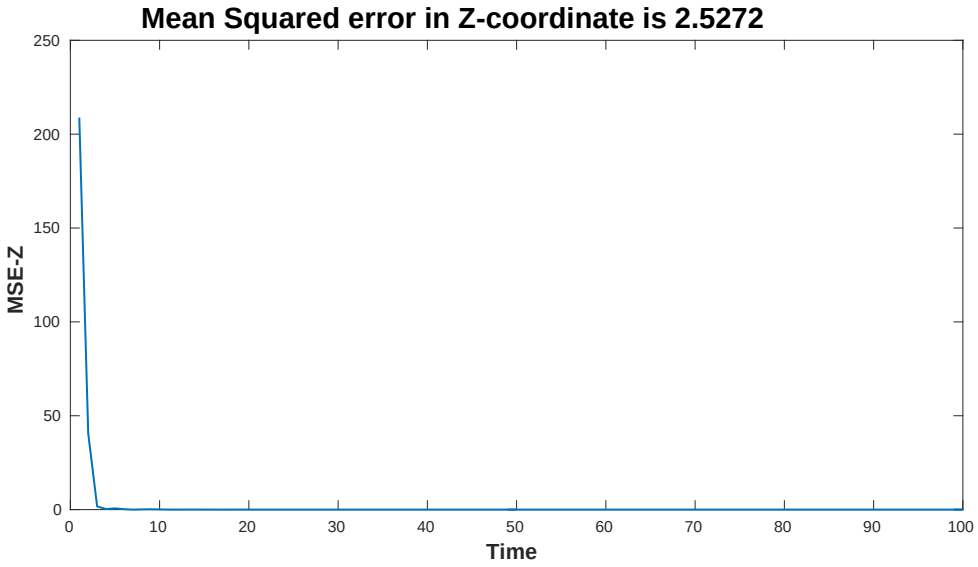

Figure 7.12: Case 2: Mean Squared Error for z-axis

closed loop controller seems to be working efficiently. The ground vehicle follows the instructed path and the quadcopter follows the ground vehicle using the relative pose sent to it. For this set of simulation parameter only the path for the x and y axes were interchanged. The parameters set for the controllers at the outer and inner level of the closed loop are very robust and no changes were needed.

The simulations in 7.1 and 7.2 are used to find an optimal set of controller parameters for the joint centralized controller on board the ground vehicle. These parameters were tuned to ensure that the system has the ability to follow any random motion in x or y direction with negligible error. This step makes the system stable and ensure good performance in presence of noise.

7.3 x and y Vary while z Constant

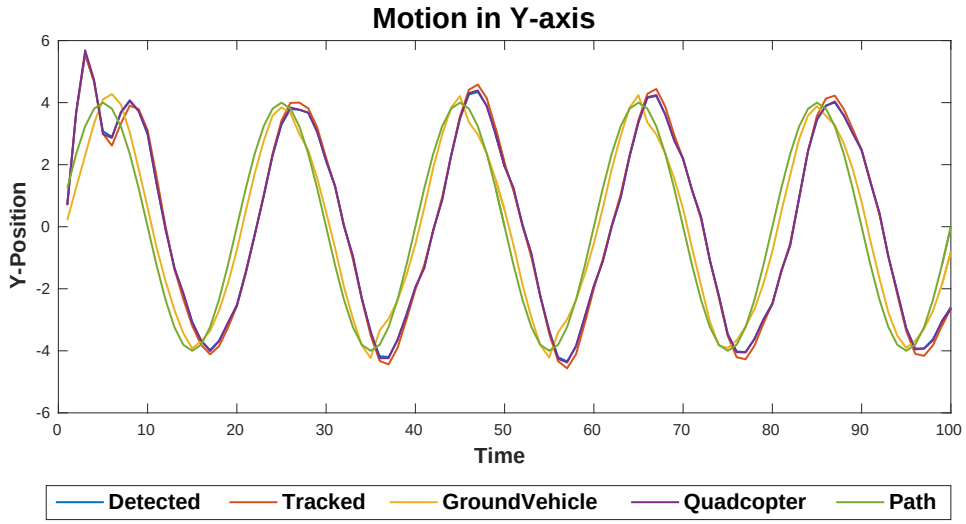

Figure 7.13: Case 3: Results for Closed Loop when y varies

From Figures 7.13 − 7.18, we can see that the quadcopter tries to follow the ground vehicle but the results are not as close as the simulations in 7.1 and 7.2. This could be attributed to frequent change in position of the y-axis. Again for all three axes we see that the system takes 4 seconds to settle and reach to the position as instructed from their initial position of (0,0,0). From that time on-ward the quadcopter tries follows the y-coordinated as needed but suffers through tracking error that occurs due to frequent and steep change in y coordinate for the quadcopter. Also in this case we are varying the x coordinate and the controller action enables the qudcopter to follow the ground vehicle. As we know the x and y axes position controls are intricately

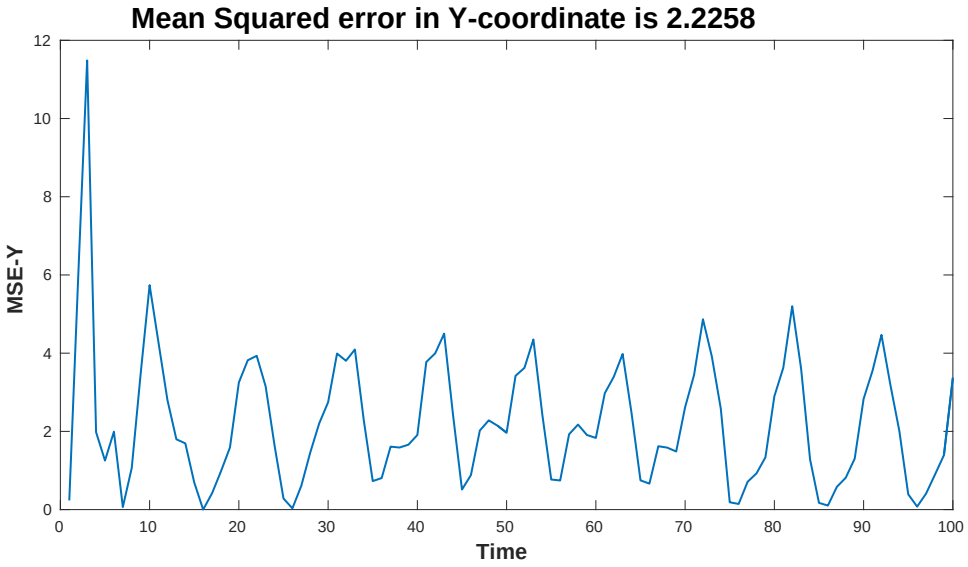

Figure 7.14: Case 3: Mean Squared Error for y-axis

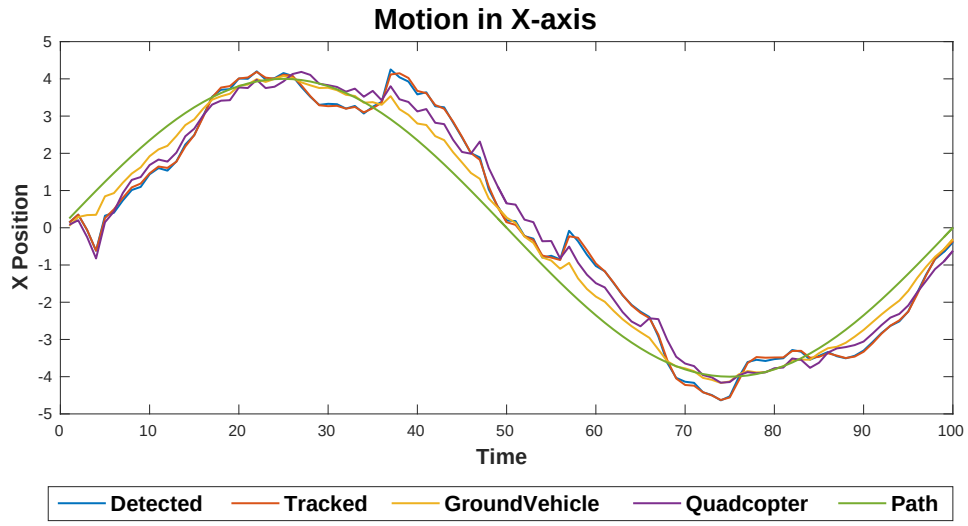

Figure 7.15: Case 3: Results for Closed Loop when x varies

interlinked, so frequent changes in the y coordinate affects the position controller in the x-axis. Even if we ignore the first 4 seconds of settling time the MSE values for each axes is computed as $x_{MSE} = 0.1243$, $y_{MSE} = 1.1124$, and $z_{MSE} = 0.0128$.

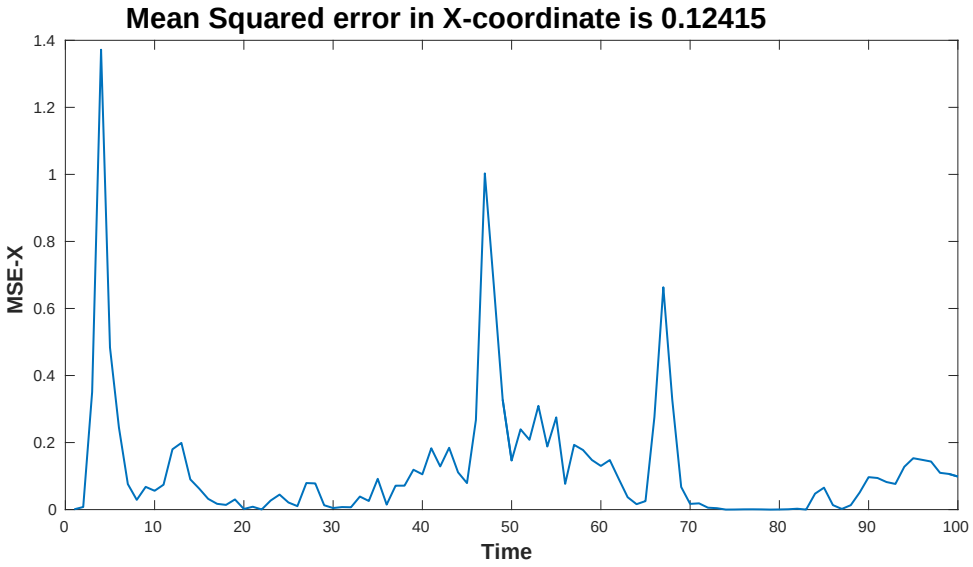

Figure 7.16: Case 3: Mean Squared Error for x-axis

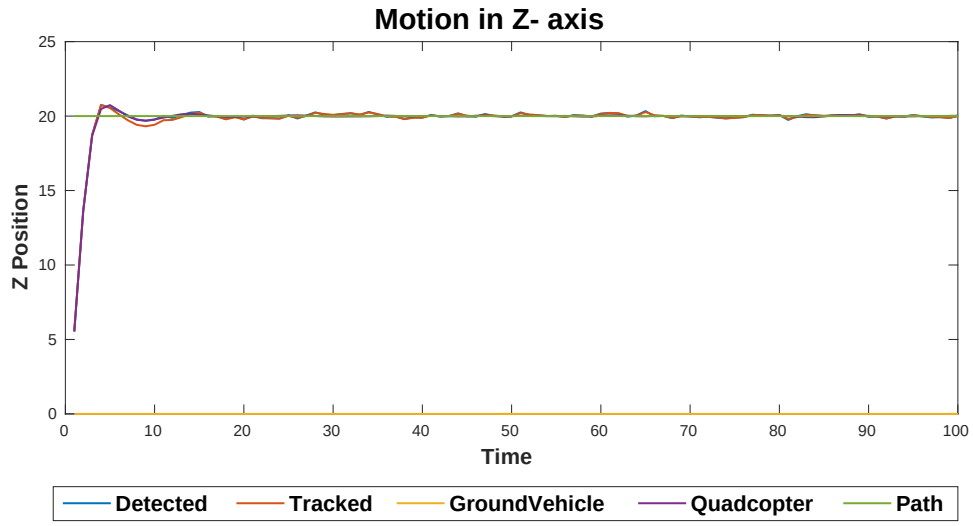

Figure 7.17: Case 3: Results for Closed Loop when $z = 20$

These values are extremely high compared to earlier cases. The errors in the x and y coordinates increase marginally; so, in order to maintain the performance of our system, we design an improved controller for the quadcopter position control. As

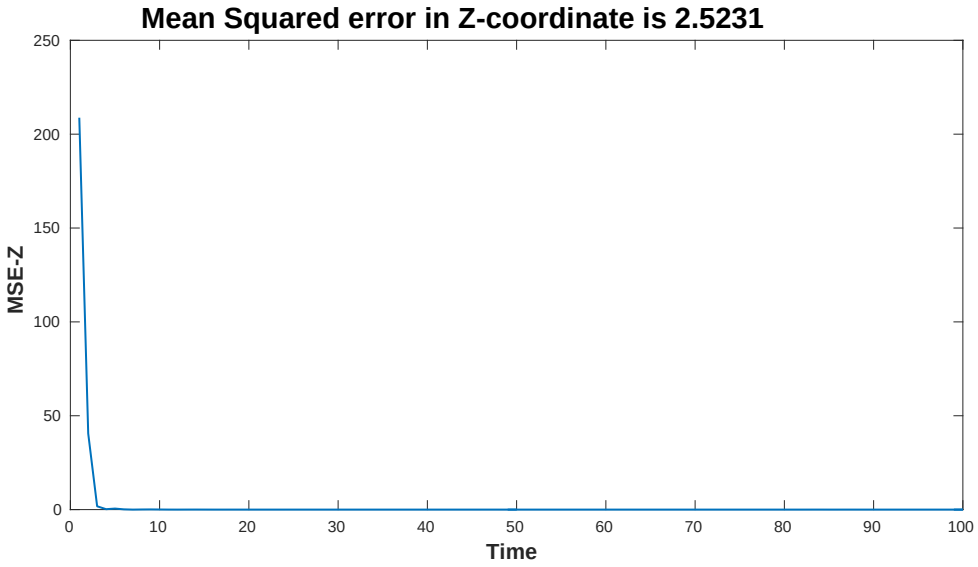

Figure 7.18: Case 3: Mean Squared Error for z-axis

per the controller for the z position which is still kept to be constant the quadcopter maintains the hover position exactly with negligible error. In the next section we incorporate an improved position controller for the quadcopter and run the simulation for the same desired path.

7.4 x, y Vary and z Constant with Feedforward Controller

Our objective for simulation in 7.3 was to vary the x and y coordinate but the tracking error increase marginally caused due to frequent changes in the y coordinate and also interdependence of the position controller of the x and y direction on board the

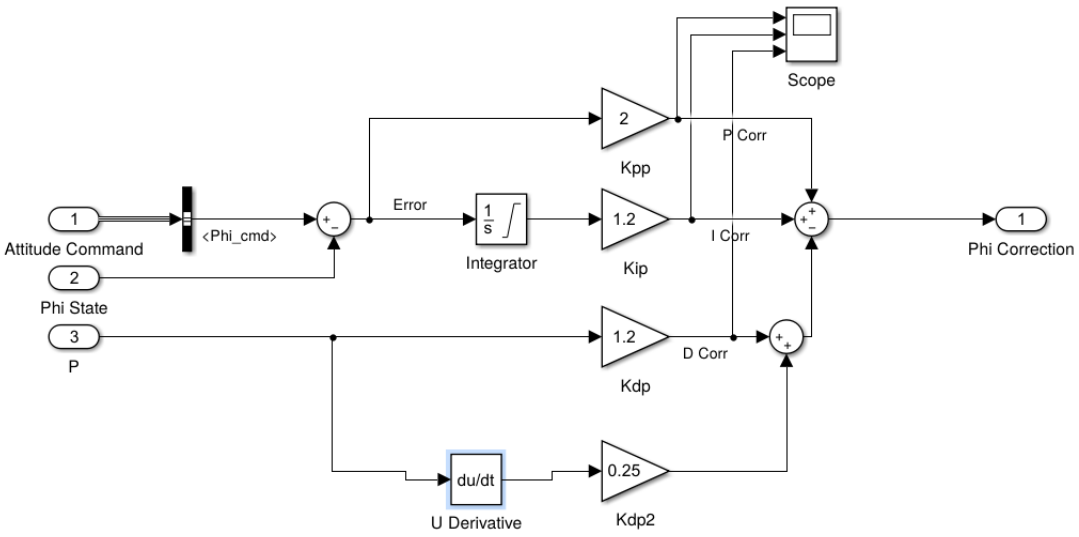

Figure 7.19: Conventional PID with Feed Forward Control

quadcopter. This adversely affects our overall closed loop feedback. The tracking error occurs when the position controller suffers through large changes in the set point. For each set point change a huge position error is observed. All the error terms accumulates significantly during the rise and continue increasing. To overcome this we modify the attitude controller of the quadcopter by including a feed-forward control element for the quadcopter to restrict the tracking error. With feed-forward control, the disturbances are measured and accounted for before they have time to affect the system. We modify our conventional PID controller as follows. Feed-forward control technique is known to improve the systems performance drastically by reducing the tracking error by compensating the measured disturbances. In general the feed-forward compensation is applied through desired velocity and acceleration. For our

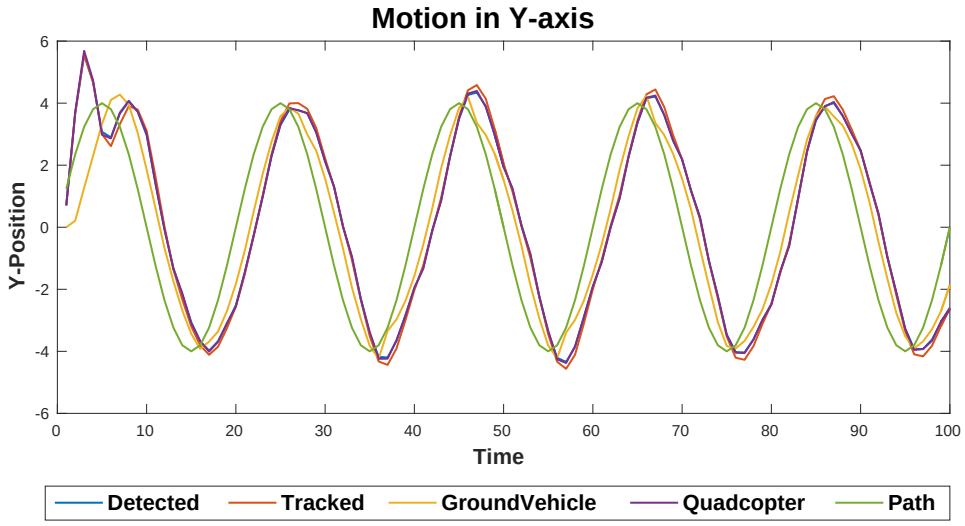

Figure 7.20: Case 4: Results for Closed Loop when y varies

case we use the angular velocities ω_x, ω_y, and ω_z. This was implemented for the high level controller for attitude stabilization on the quadcopter as shown in 7.19. After this modification and keeping other parameters similar to the previous simulation in 7.3, we run the simulation with varying values for x and y while maintaining a hover position for the quadcopter in the z direction at $z = 20m$. The results obtained are shown in Figures 7.20-7.25. Figure 7.21 when compared to Figure 7.14 shows drastic improvement in the tracking error in the y direction. Similar behavior is observed in the x direction also. So we can infer that this modification improves the overall performance of our closed loop feedback setup. The error has reduced considerably. After the time $t = 4s$ the quadcopter follows the desired path in both x and y direction closely. The MSE values for each axes from time $t = 4s$ onwards is computed as $x_{MSE} = 0.0551$, $y_{MSE} = 0.4498$, and $z_{MSE} = 0.0128$. Overall improvement in the

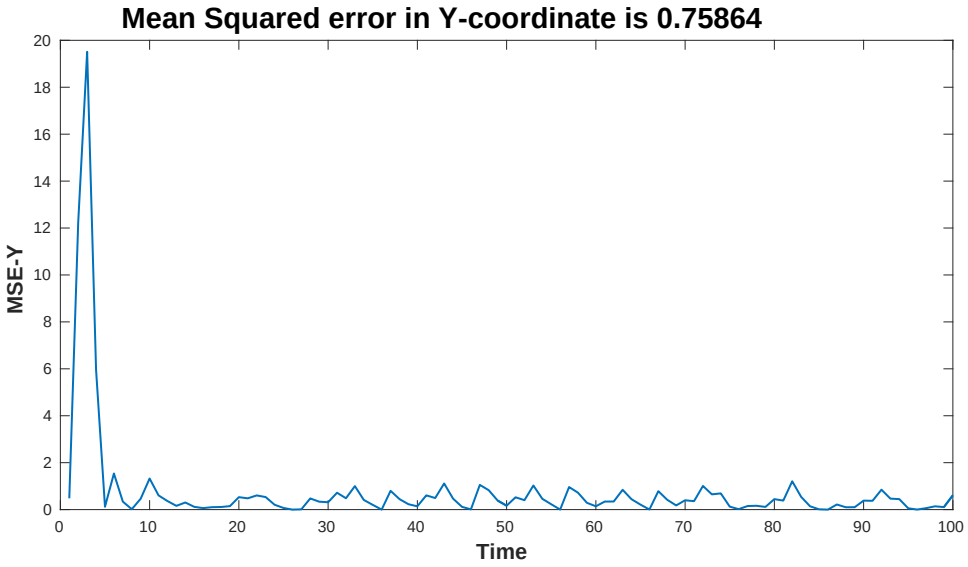

Figure 7.21: Case 4: Mean Squared Error for y-axis

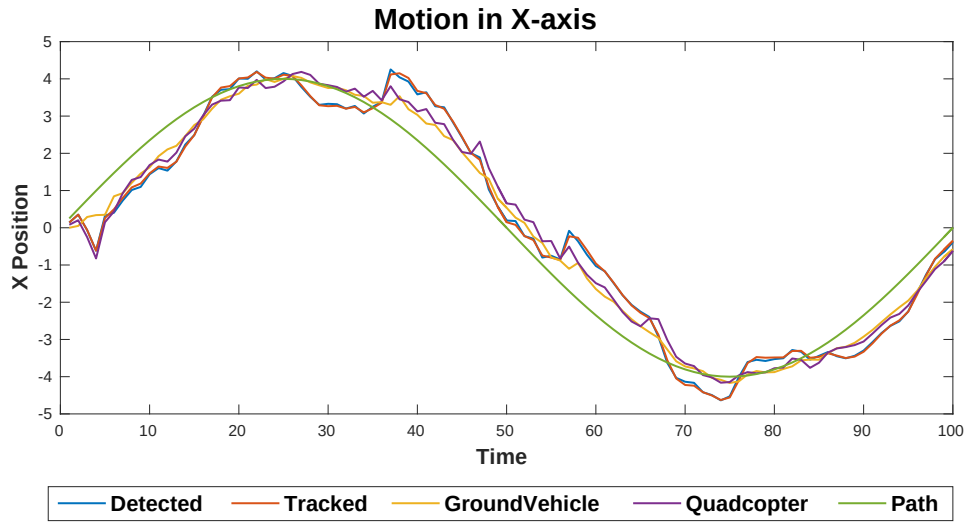

Figure 7.22: Case 4: Results for Closed Loop when x varies

system is observed. Now we run the next simulations keeping this modification for our system. From this simulation, we have obtained best parameters for our closed loop architecture and now in 7.6, we access the performance of our system for varying

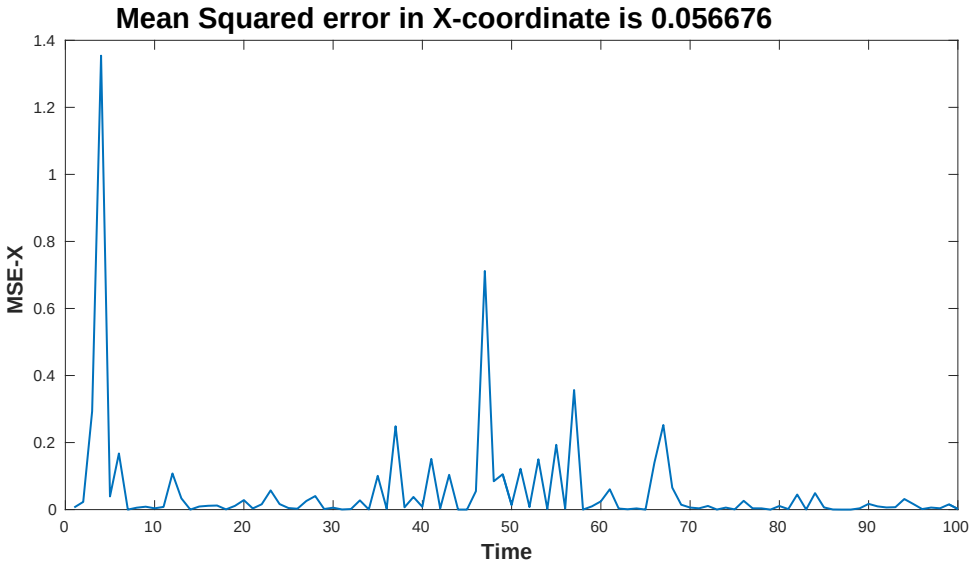

Figure 7.23: Case 4: Mean Squared Error for x-axis

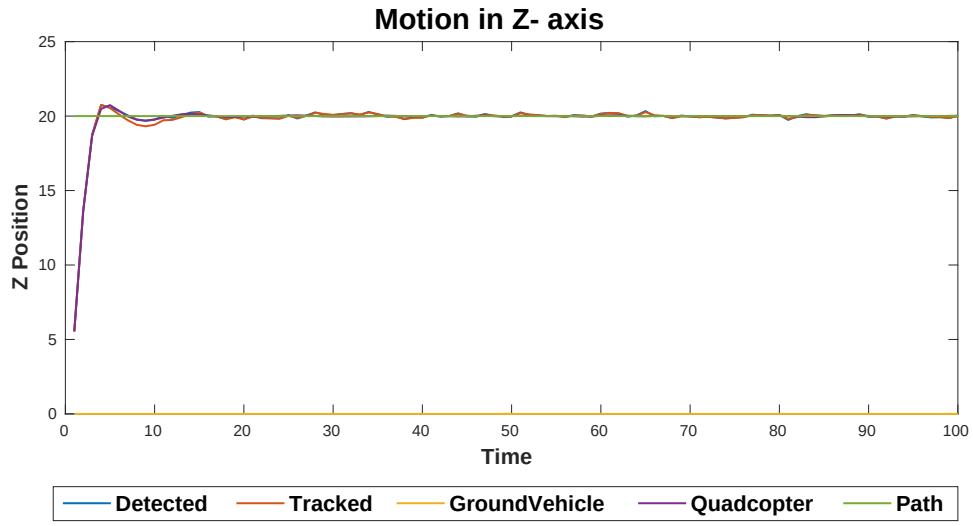

Figure 7.24: Case 4: Results for Closed Loop when $z = 20$

x, y, and z coordinates.

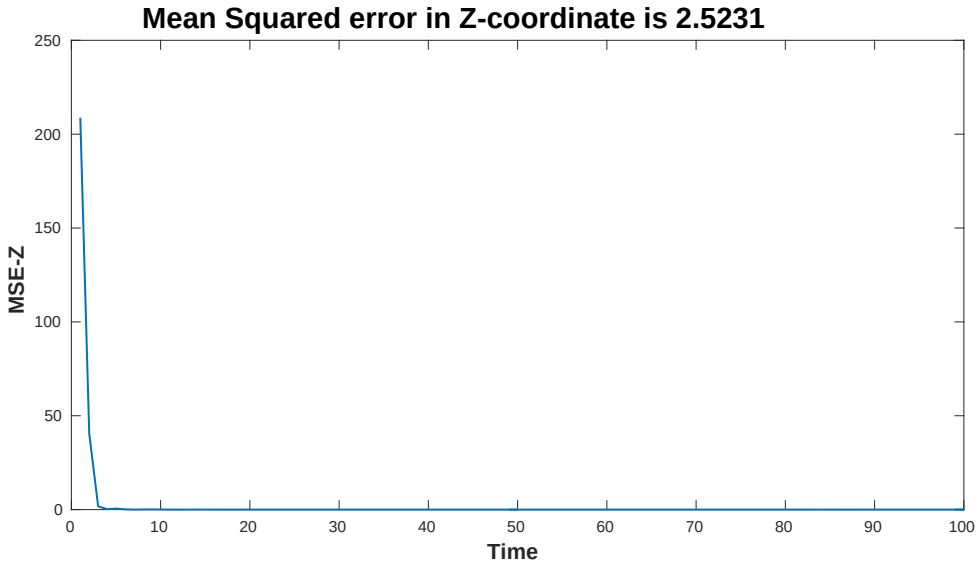

Figure 7.25: Case 4: Mean Squared Error for z-axis

7.5 x, y Vary with respect to ground vehicle and z is Constant

From the Figures 7.26 - 7.31, we can infer that the quadcopter follows the desired path closely and stays in the field of view. For all three axes we see that the system takes 4 seconds to settle and reach to the position as instructed from their initial position of (0,0,0). For this simulation we vary the x and y coordinates with respect to the grounf vehicle while maintaining a constant hover position. This simulation shows that the quadcopter not only follows the moving ground vehicle but can also move as per the path desired or as per the instructions obtained from the ground vehicle. This will be helpful in circumstances when we have to obtain a broader field

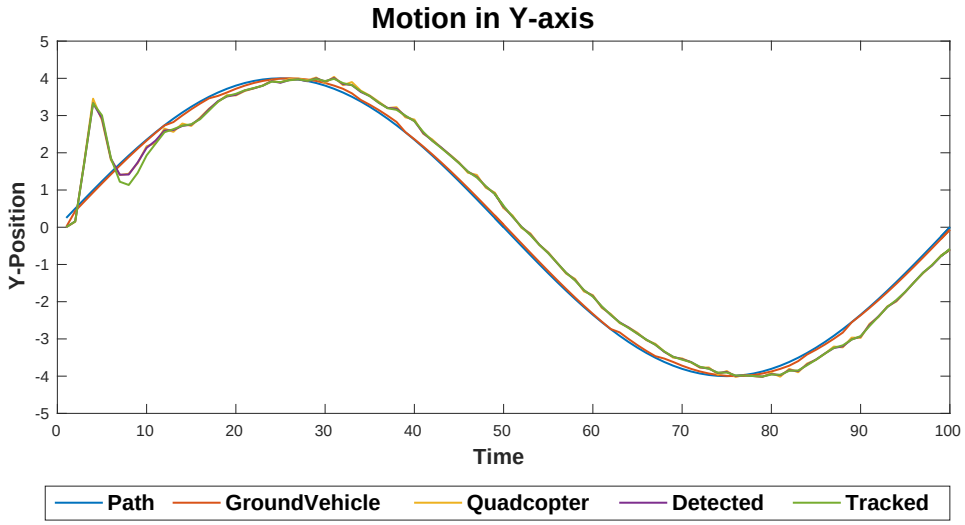

Figure 7.26: Case 5: Results for Closed Loop when y varies

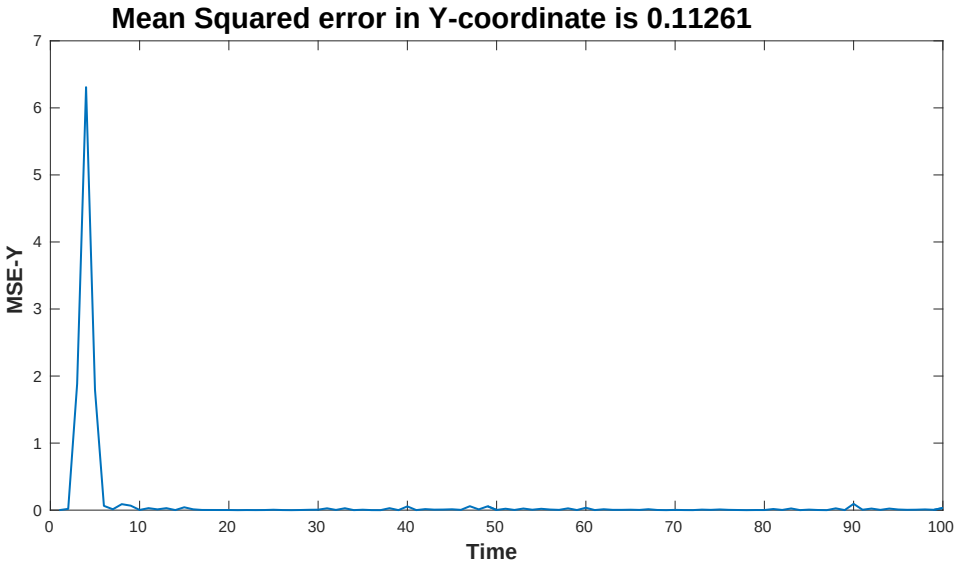

Figure 7.27: Case 5: Mean Squared Error for y-axis

of view from the quadcopter or have to dodge an obstruction in the aerial path. The error in position for all the three axes decreases. For the figure 7.28 we notice that ground vehicle stays at $x = 4$ while the quadcopter moves left and right with respect

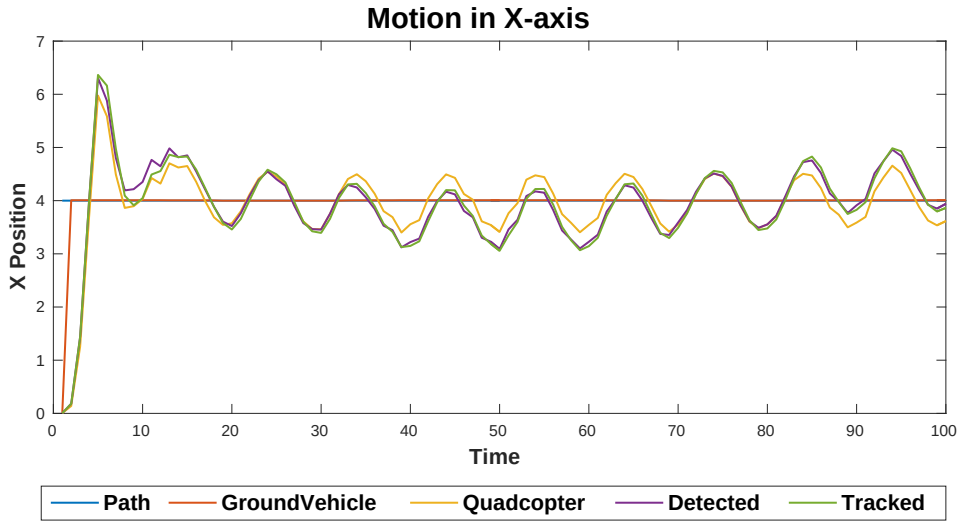

Figure 7.28: Case 5: Results for Closed Loop when x varies

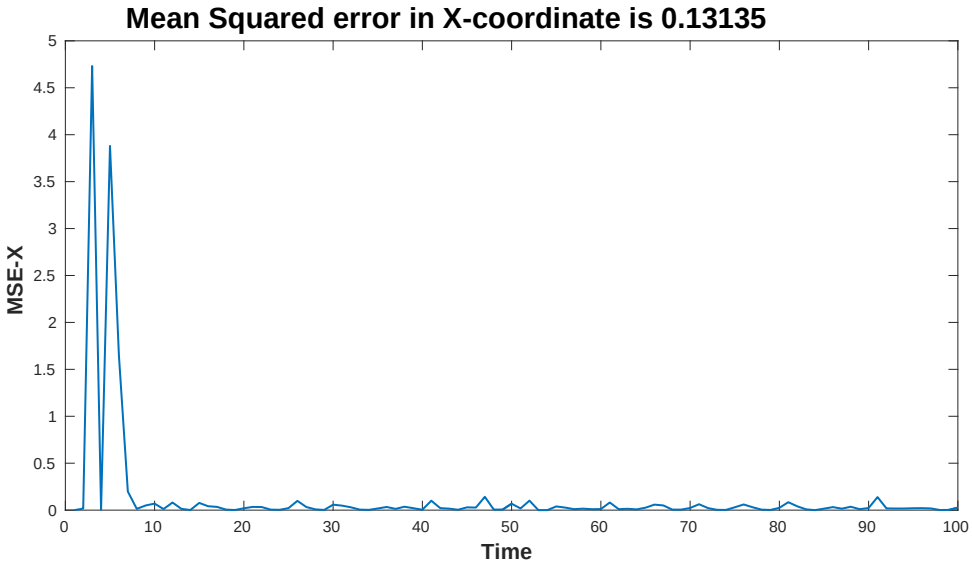

Figure 7.29: Case 5: Mean Squared Error for x-axis

to the ground vehicle. Eventually the error for all three axes is extremely small. If we measure the MSE value after the quadcopter reaches its hover position, i.e., after time $t = 4s$, we get $x_{MSE} = 0.0865$, $y_{MSE} = 0.0965$, and $z_{MSE} = 0.0465$. This case

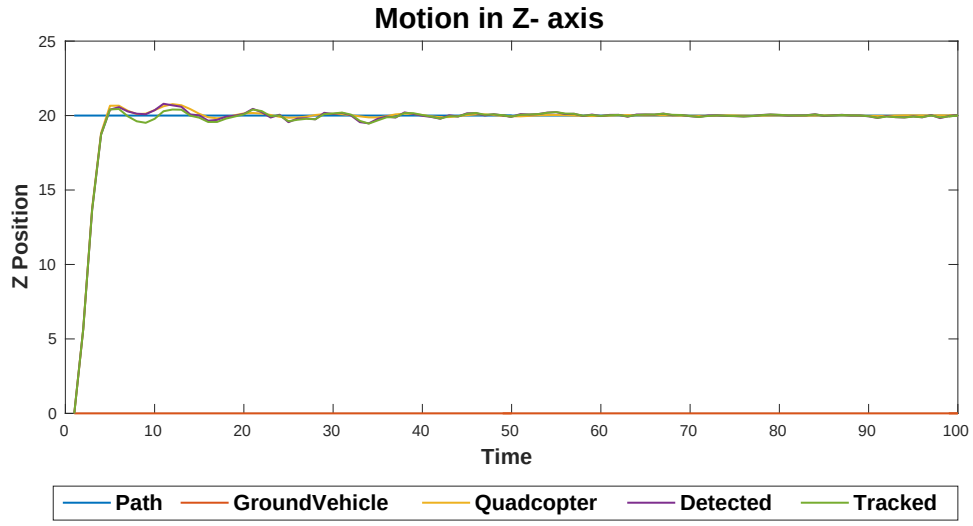

Figure 7.30: Case 5: Results for Closed Loop when z varies

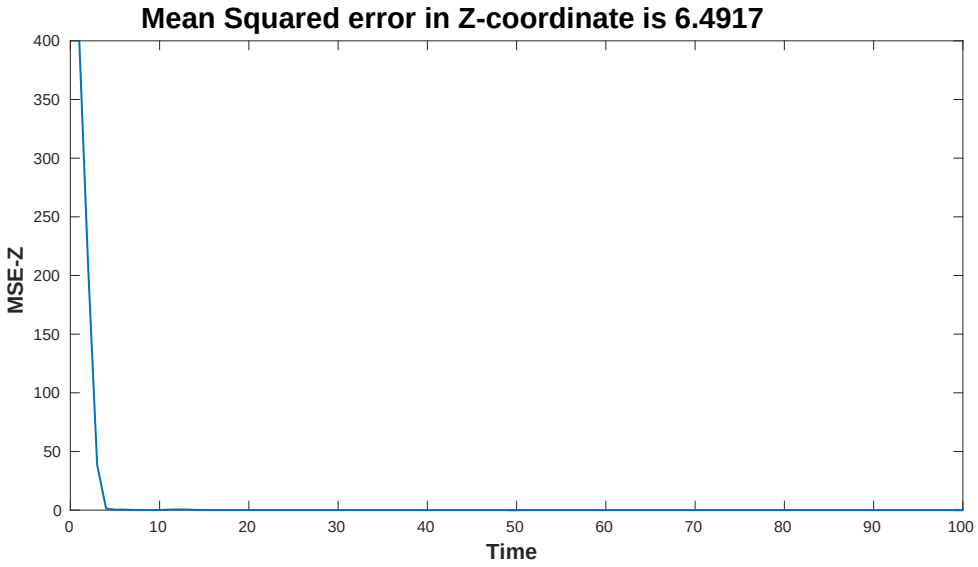

Figure 7.31: Case 5: Mean Squared Error for z-axis

shows that the detection and tracking algorithm works for broader field of view also which is highly desirable for real life implementation.

7.6 x, y and z all Vary

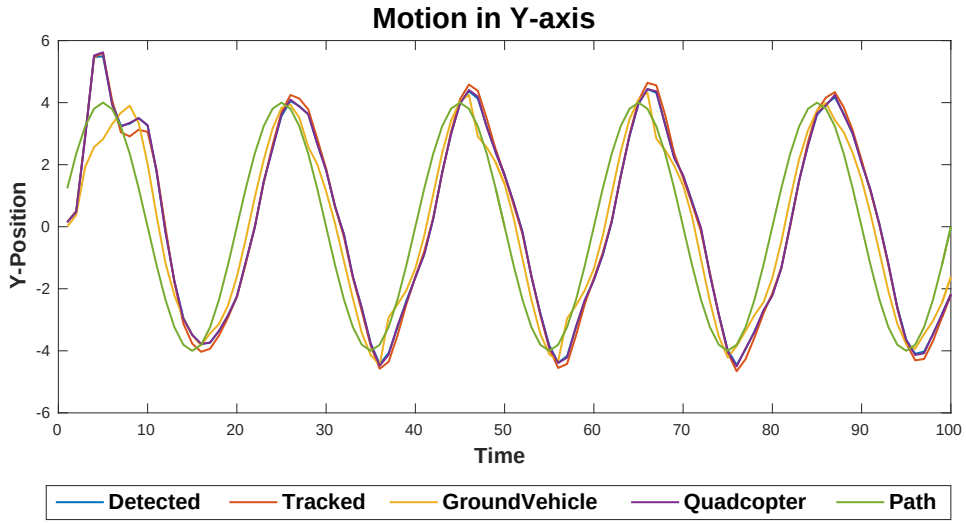

Figure 7.32: Case 5: Results for Closed Loop when y varies

From the Figures 7.32 - 7.37, we can infer that the quadcopter follows the ground vehicle closely with minimal error. For all three axes we see that the system takes 4 seconds to settle and reach to the position as instructed from their initial position of (0,0,0). For this simulation we vary the coordinates in all the three axes. The results obtained are very satisfactory as now with the feed-forward controller the entire system becomes extremely efficient and the error in the position drops with time and whenever we have steep changes in the coordinates the controller actively tries to overcome it. The error in position for all the three axes decreases. For the figure 7.36 we notice that the detected and the tracked location for the z-axes deviate

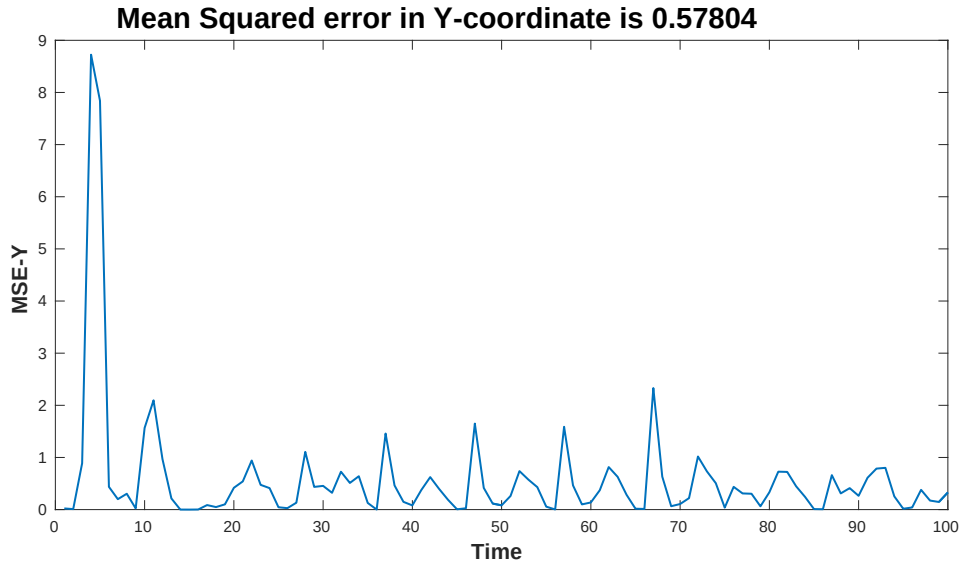

Figure 7.33: Case 5: Mean Squared Error for y-axis

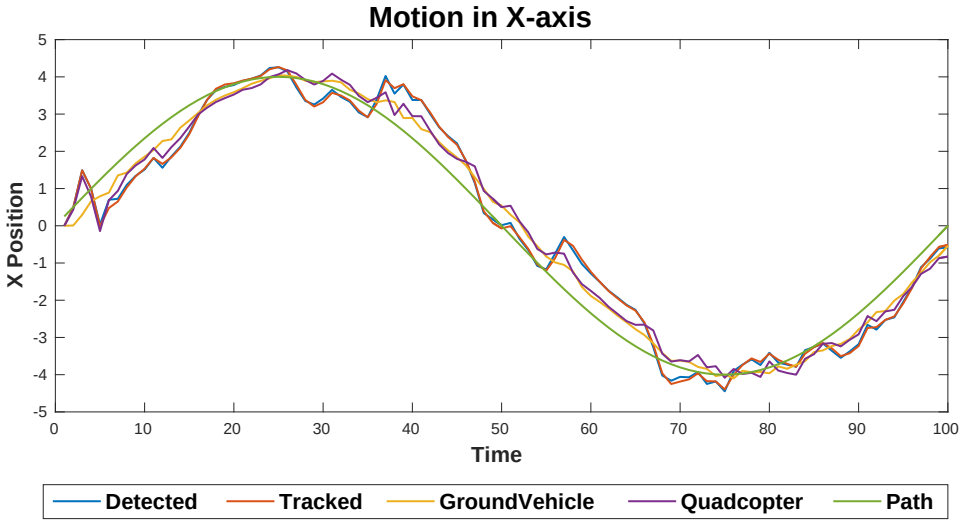

Figure 7.34: Case 5: Results for Closed Loop when x varies

from the expected path but then with joint centralized controller on board of the ground vehicle which deals with the relative position send the controller input to overcome this deviation. Eventually the error for all three axes is extremely small. If

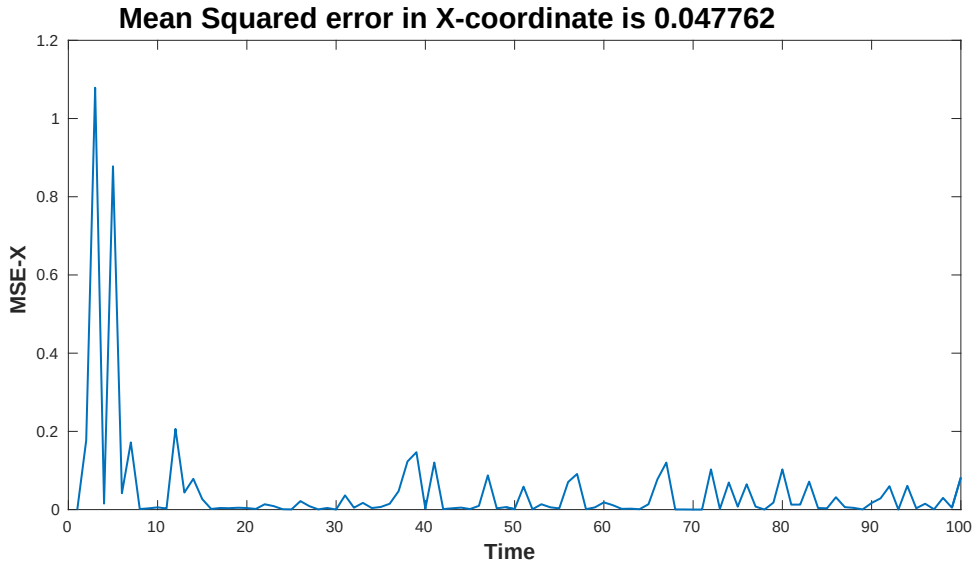

Figure 7.35: Case 5: Mean Squared Error for x-axis

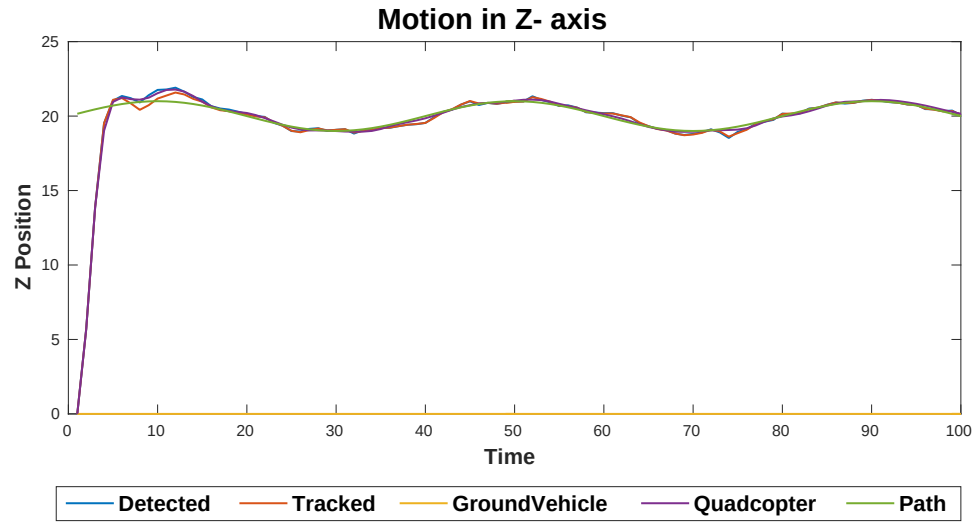

Figure 7.36: Case 5: Results for Closed Loop when z varies

we measure the MSE value after the quadcopter reaches its hover position, i.e., after time $t = 4s$, we get $x_{MSE} = 0.0363$, $y_{MSE} = 0.5864$, and $z_{MSE} = 0.0649$. This case shows that the detection and tracking algorithm works with changing z position also

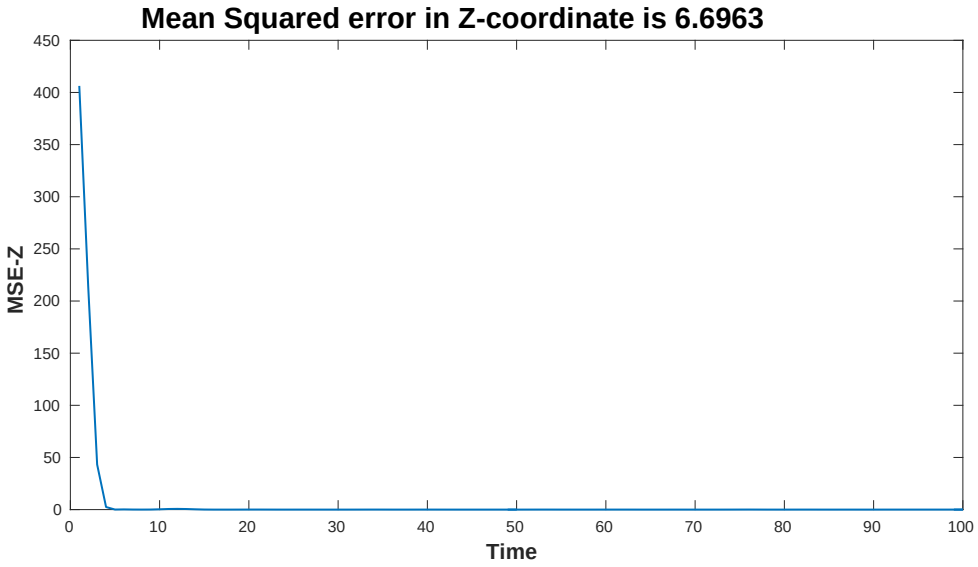

Figure 7.37: Case 5: Mean Squared Error for z-axis

which is highly desirable for real life implementation.

7.7 Changing Focal Length of Camera

Our vision based tracking algorithm relies heavily on the quality of the images. To determine the performance of the algorithm for vision based detection and tracking, we change the camera parameters. We run the next set of simulation to understand the effect of varying focal length.

The focal length of a camera determines the optical power of the camera. Focal length is a measure how the lens focusses the light into a point. For our system, we have modelled a camera with a focal length of $45mm$. The algorithm developed uses

Kalman filter tracking which has parameters tuned to incorporate measurement noise and observation noise. It is expected that even for changing focal length the tracking would not be affected too adversely. To demonstrate this, we first increase the focal length of the camera to $65mm$ in 7.7.1 and then decrease it to $25mm$ in 7.7.2. The results obtained are then discussed in 7.7.3.

7.7.1 Increased Focal Length

For this case the focal length of the camera is increased to $f = 65mm$. Longer focal length improves the images for far away objects by flattening the images. For an increased focal length of the camera as expected the error for z axis decreases while for the x and y axes error remains approximately same. The results obtained are shown in Figure $7.38 - 7.43$.

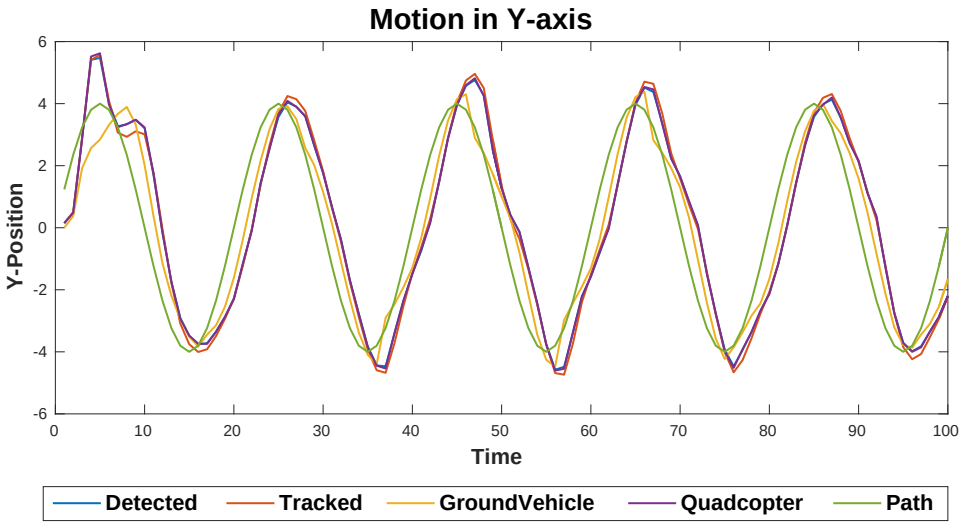

Figure 7.38: Case 6.1: Results for Closed Loop when y varies

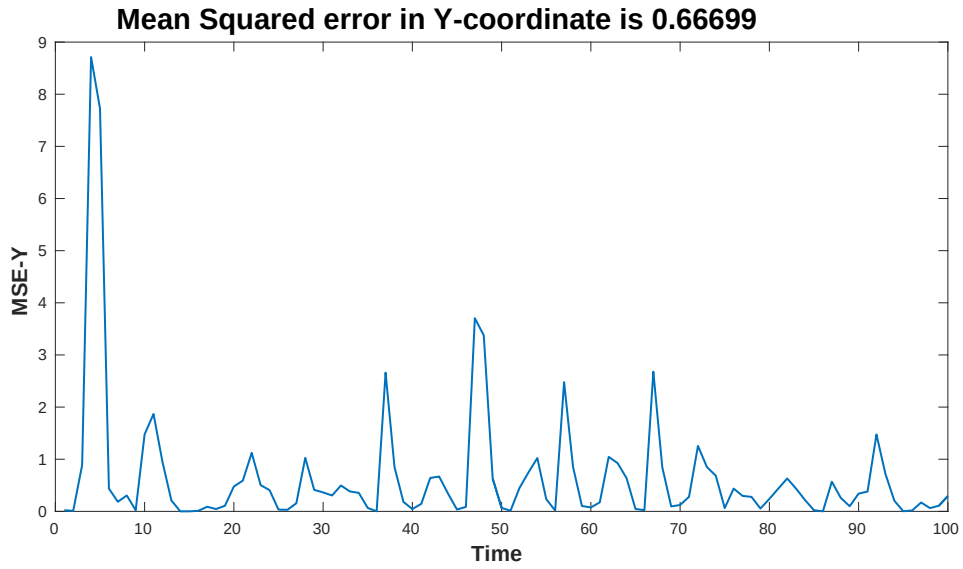

Figure 7.39: Case 6.1: Mean Squared Error for y-axis

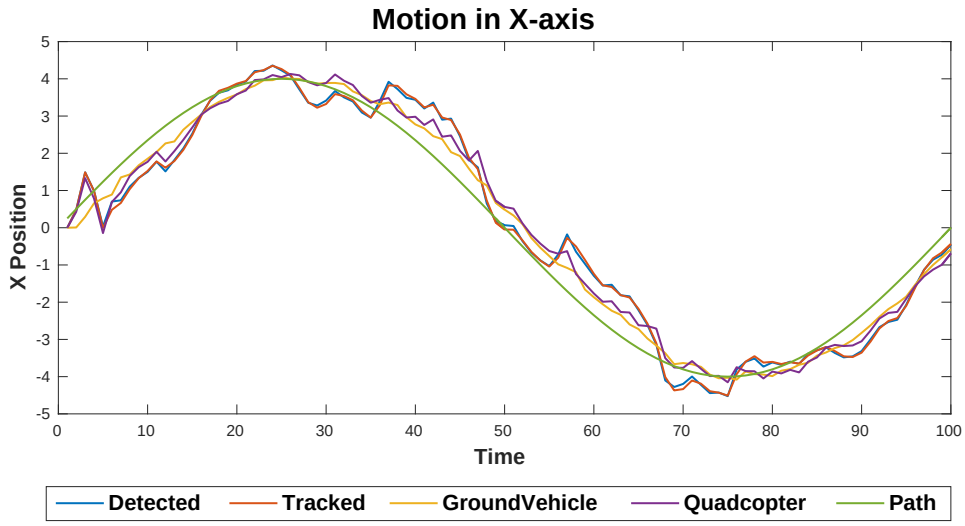

Figure 7.40: Case 6.1: Results for Closed Loop when x varies

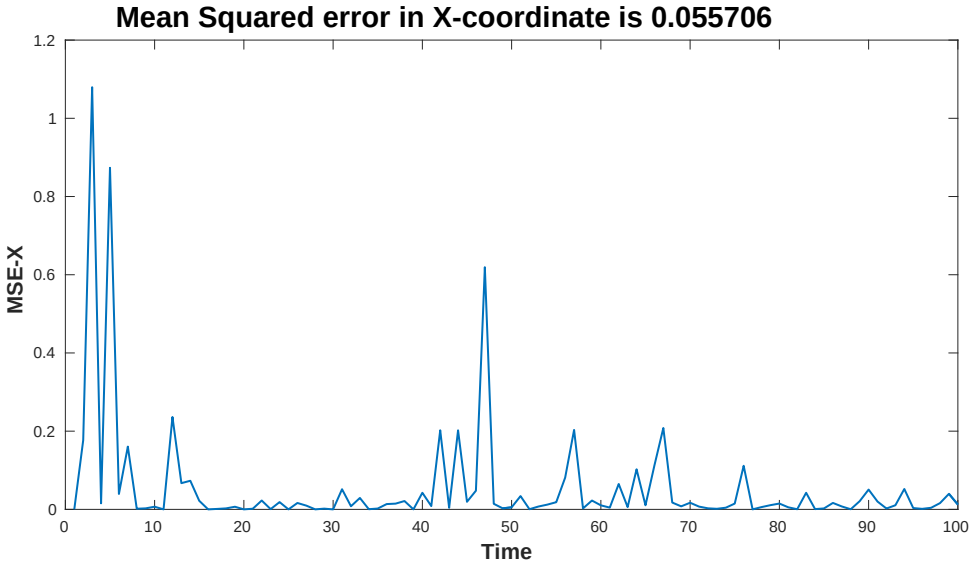

Figure 7.41: Case 6.1: Mean Squared Error for x-axis

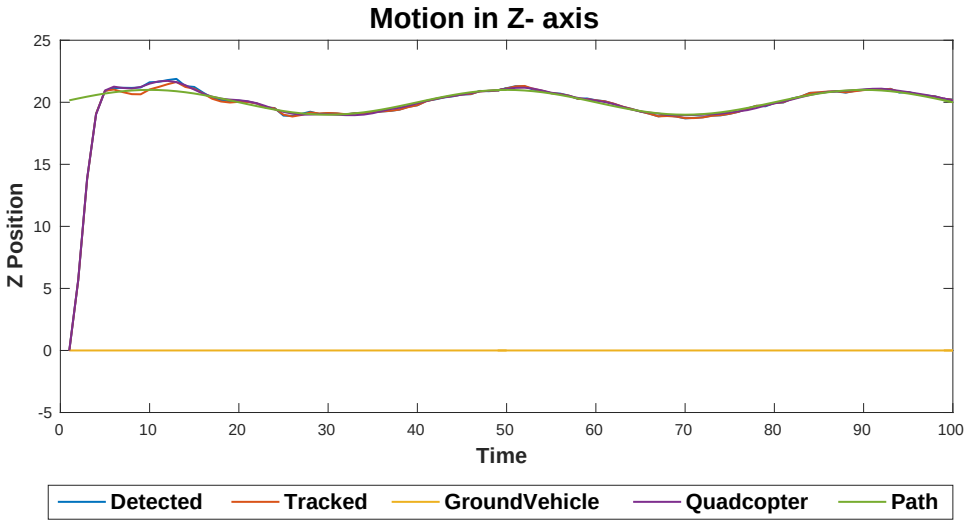

Figure 7.42: Case 6.1: Results for Closed Loop when z varies

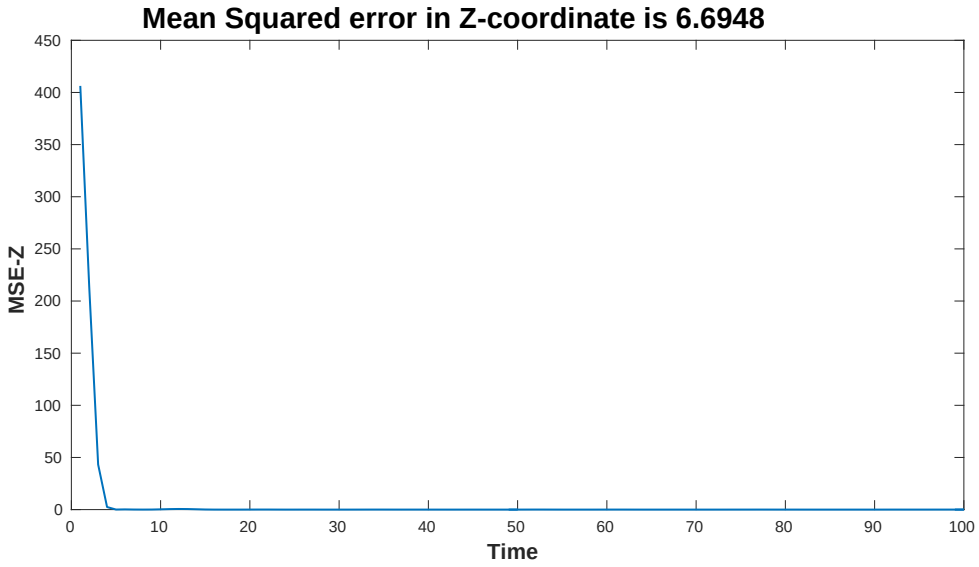

Figure 7.43: Case 6.1: Mean Squared Error for z-axis

7.7.2 Reduced Focal Length

For this case the focal length of the camera is reduced to $f = 25mm$. For a camera with smaller focal length as expected the error for z axis increases while for the x and y axes error remains approximately same. The results obtained are shown in Figures $7.44 - 7.49$

7.7.3 Results for Changing Focal Length

The results obtained by changing the focal length of the camera can be compressed down into the following tabular form to be discussed.

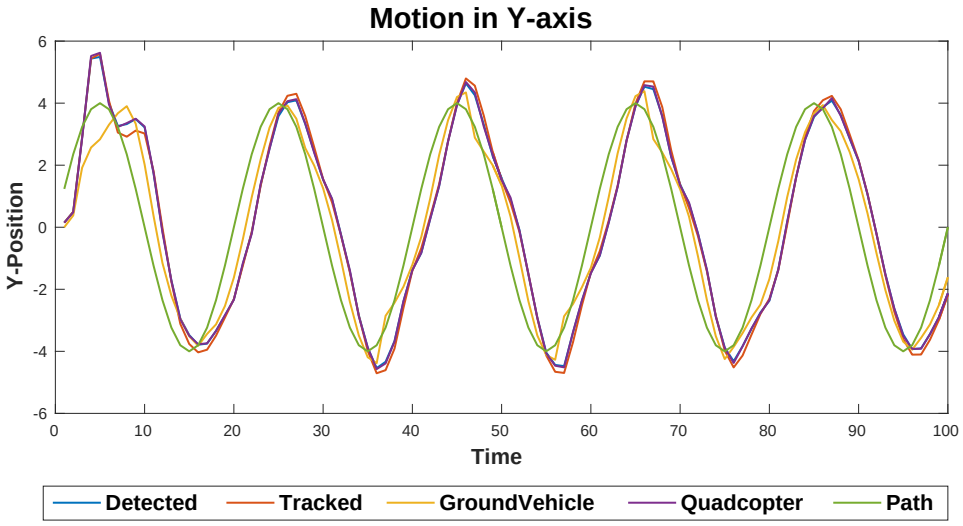

Figure 7.44: Case 6.2: Results for Closed Loop when y varies

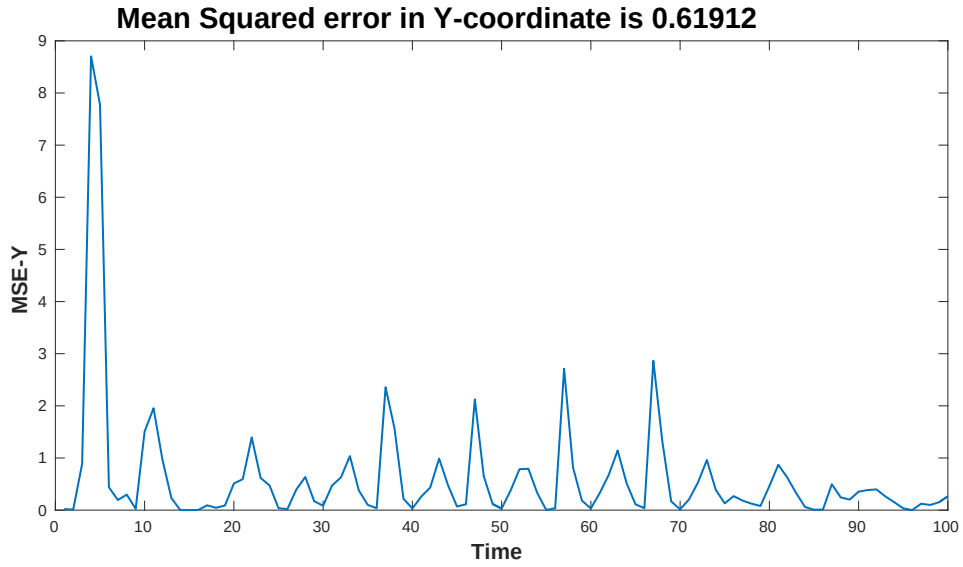

Figure 7.45: Case 6.2: Mean Squared Error for y-axis

From Table 7.1 we can see that as expected the error increases when focal length is decreased and the error is marginally reduced when the focal length is increased. With longer lens the depth of the field is reduced hence isolating the quadcopter from

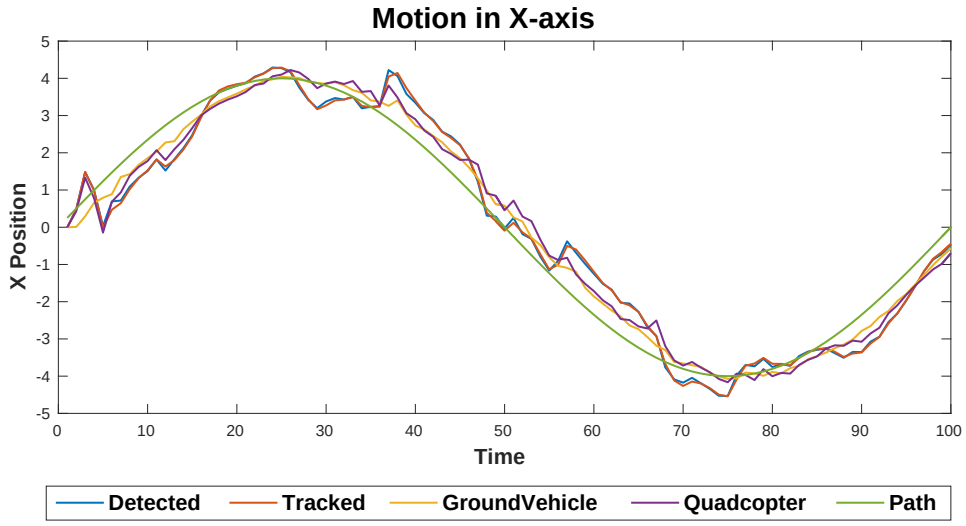

Figure 7.46: Case 6.2: Results for Closed Loop when x varies

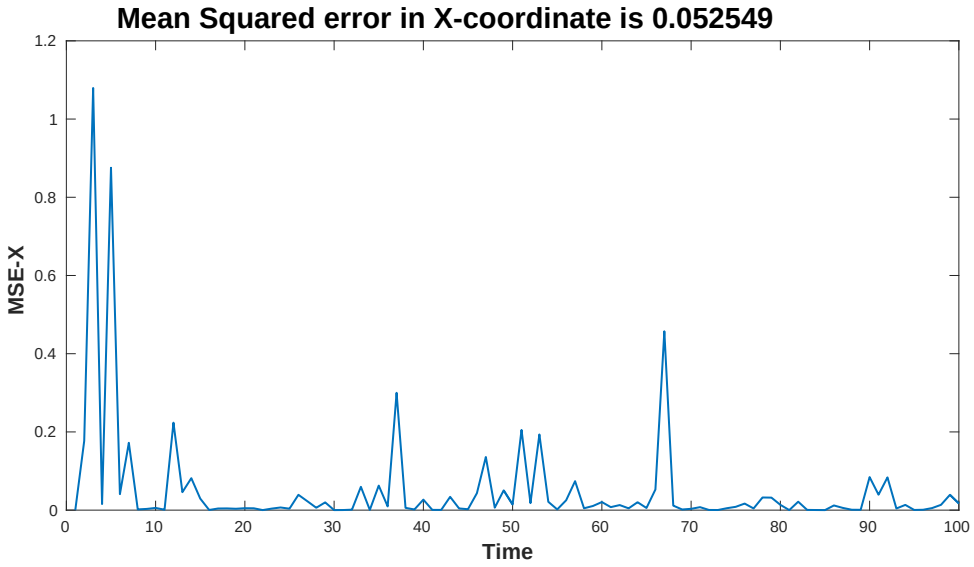

Figure 7.47: Case 6.2: Mean Squared Error for x-axis

the background and hence improving the results.

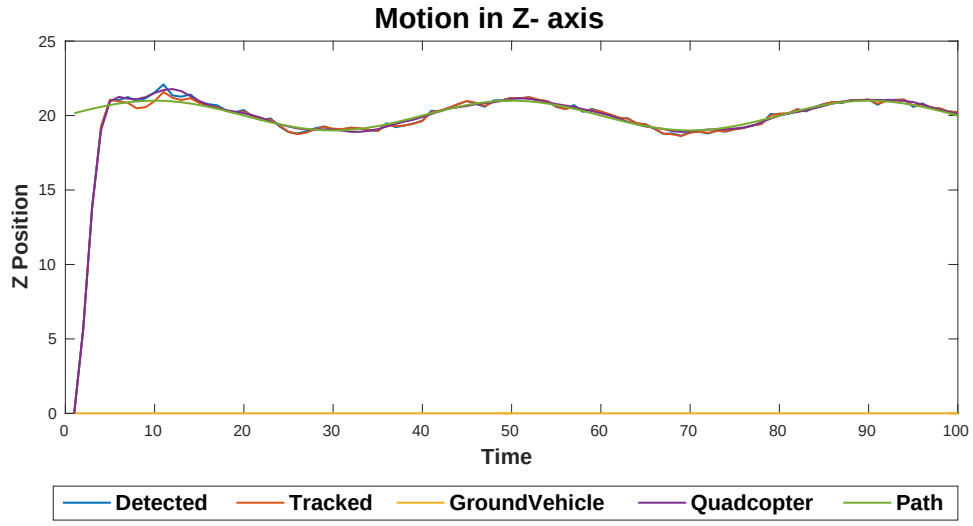

Figure 7.48: Case 6.2: Results for Closed Loop when z varies

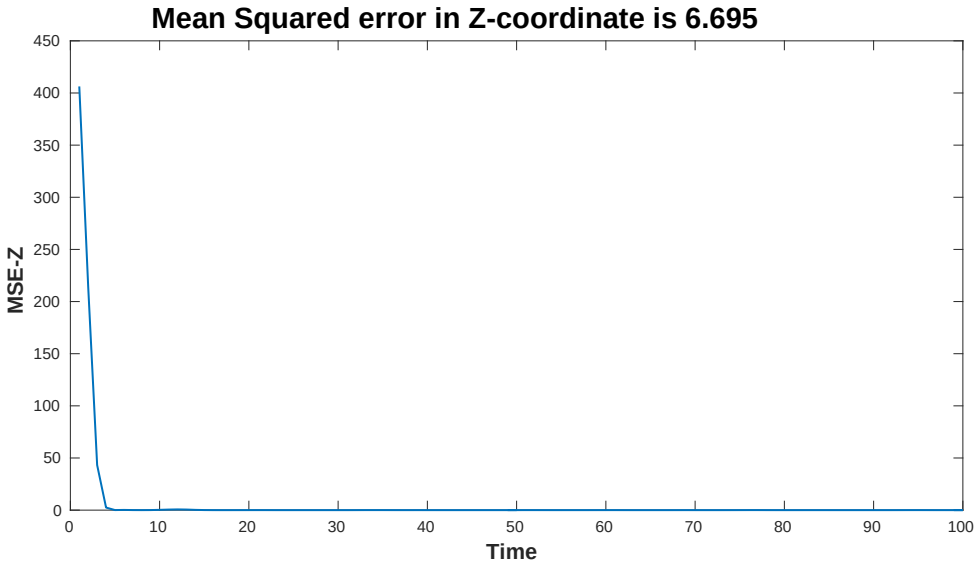

Figure 7.49: Case 6.2: Mean Squared Error for z-axis

Table 7.1
Error Matrix for Changing Focal Length

Mean Squared Error	**f = 45mm**	**f = 25mm**	**f = 65mm**
x Position Error	0.0477	0.0525	0.0457
y Position Error	0.5780	0.6191	0.5670
z Position Error	6.6963	6.6980	6.6948
x Position Error after T_s	0.0363	0.0445	0.0415
y Position Error after T_s	0.5864	0.6288	0.5781
z Position Error aftre T_s	0.0649	0.0666	0.0633
$(x_{Detected} - x_{Tracked})$	0.0038	0.0036	0.0037
$(y_{Detected} - y_{Tracked})$	0.0226	0.0239	0.0246
$(z_{Detected} - z_{Tracked})$	0.0166	0.0170	0.0150

7.8 Changing Resolution of Camera

Resolution of the camera determines its capability to see small objects with distinct boundaries. A pixel is used as a unit for the digital images. Clearly the resolution and the pixel size are closely related. Resolution of a camera indicates the pixel count in the digital imaging. For a given lens, if the pixel size is smaller, the resolution for the camera will be higher and hence giving us a more clear image. Whereas bigger the size of the pixel , resolution drops and we obtain a blurred image of the same scene. Now to determine the effects of changing resolution on our tracking and pose estimation algorithm we first increase the resolution by decreasing pixel size and then resolution is decreased by increasing the pixel size.The results of the simulation are discussed in the following sections.

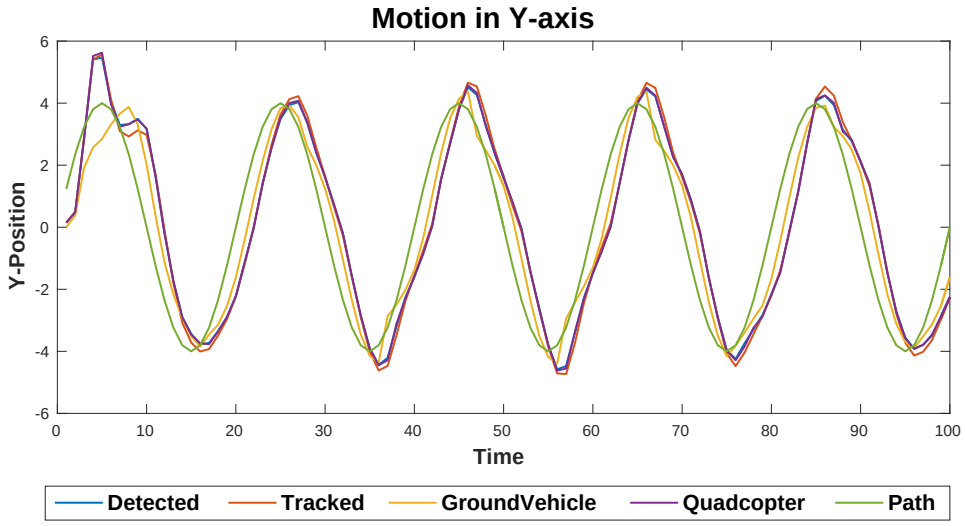

Figure 7.50: Case 7.1: Results for Closed Loop when y varies

7.8.1 Lower Resolution

For this case we have increased the pixel size to double its initial value of 0.024. The results for our closed loop feedback system are shown in Figures 7.50-7.55.

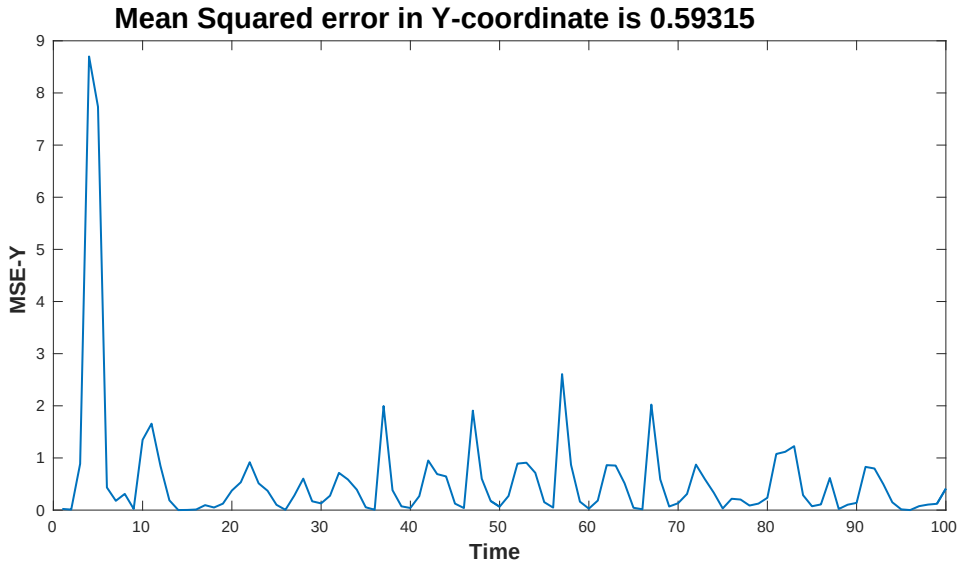

Figure 7.51: Case 7.1: Mean Squared Error for y-axis

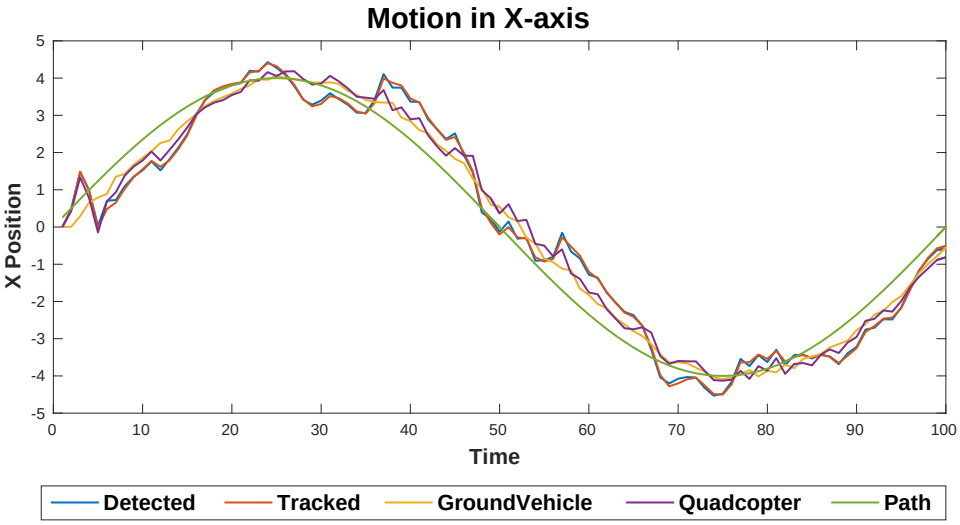

Figure 7.52: Case 7.1: Results for Closed Loop when x varies

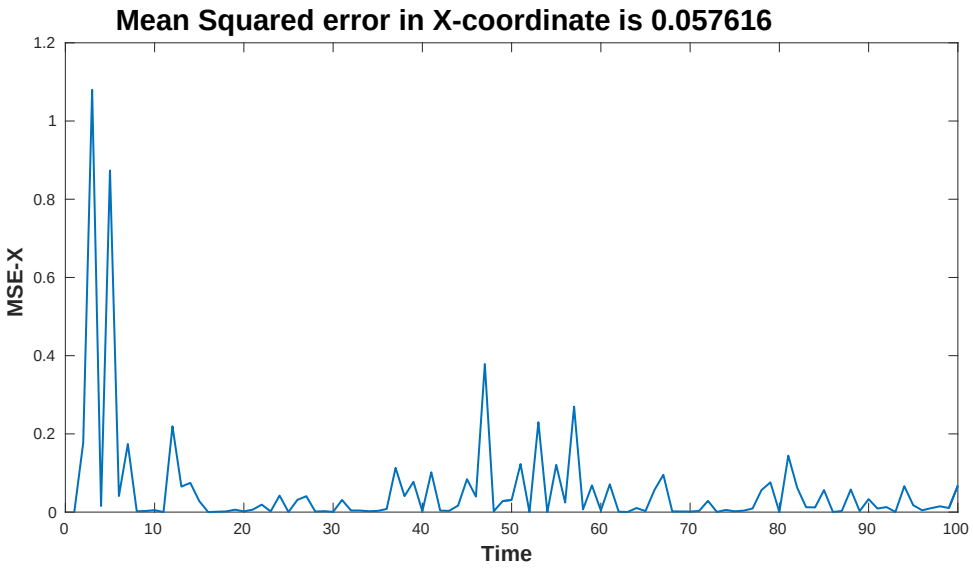

Figure 7.53: Case 7.1: Mean Squared Error for x-axis

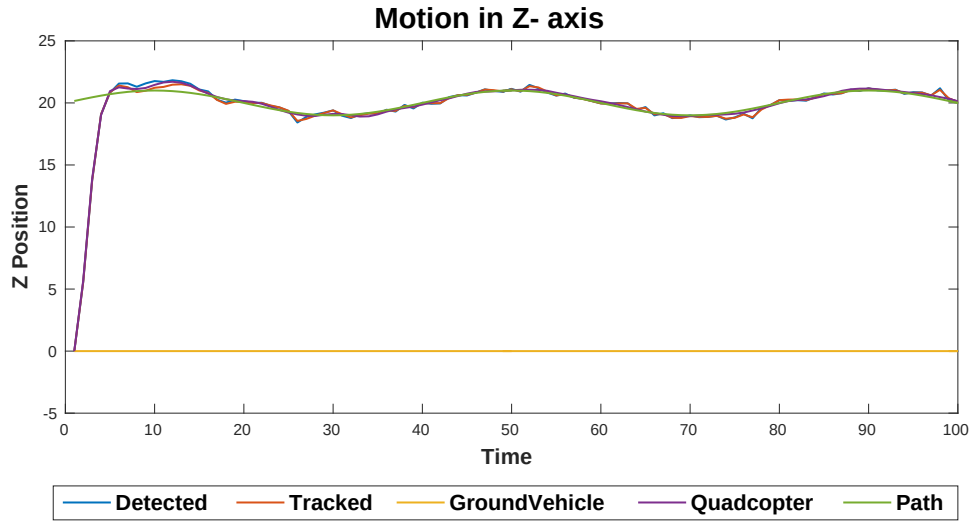

Figure 7.54: Case 7.1: Results for Closed Loop when z varies

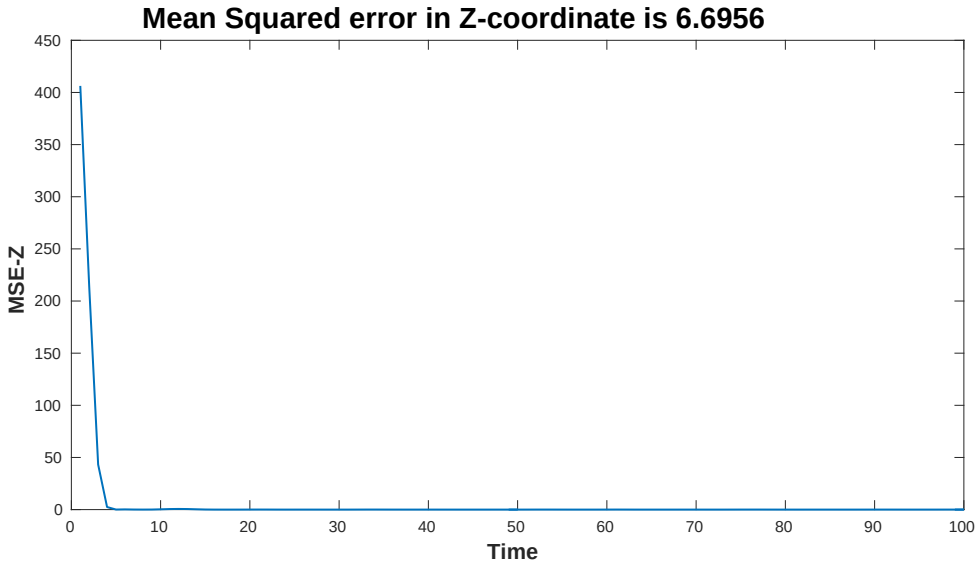

Figure 7.55: Case 7.1: Mean Squared Error for z-axis

7.8.2 Higher Resolution

For this case we have decreased the pixel size to 0.018. The results for our closed loop feedback system are shown in Figures 7.56-7.61.

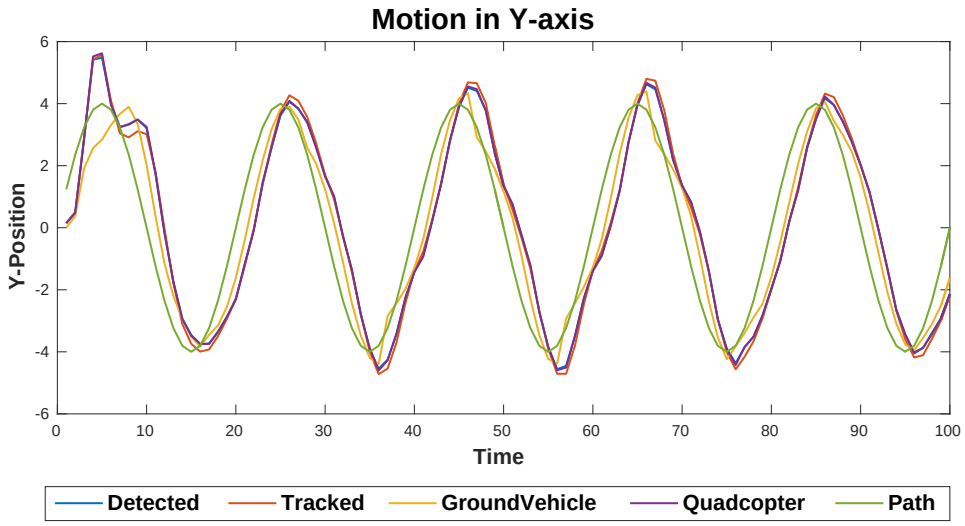

Figure 7.56: Case 7.2: Results for Closed Loop when y varies

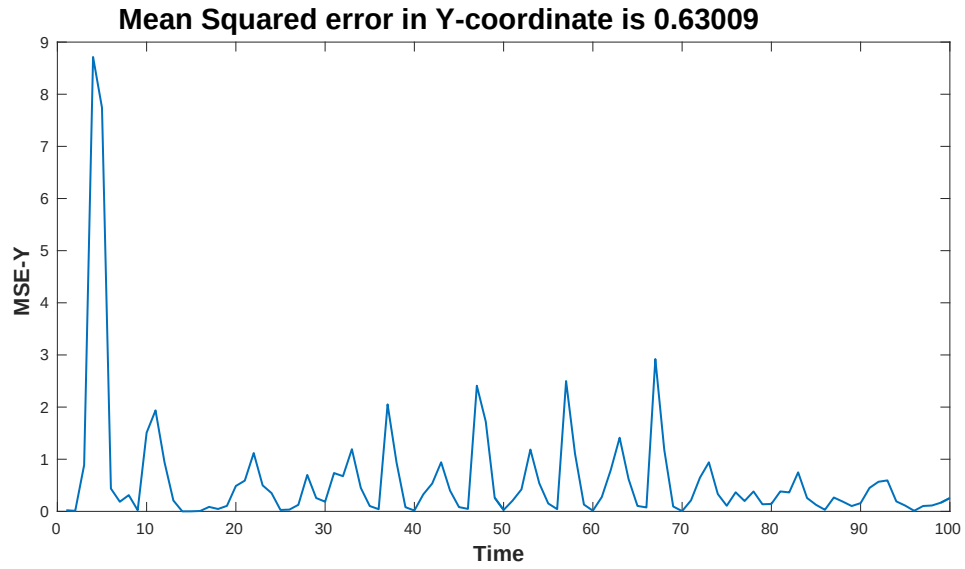

Figure 7.57: Case 7.2: Mean Squared Error for y-axis

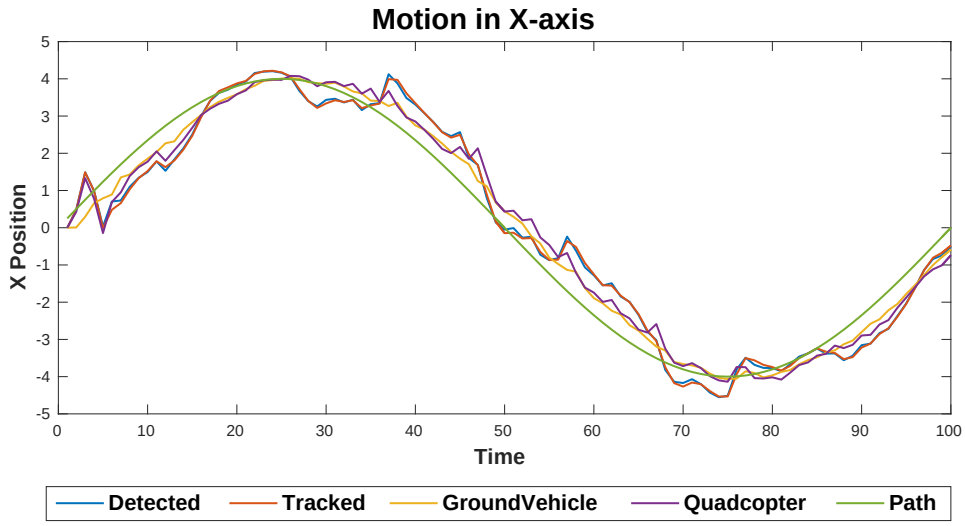

Figure 7.58: Case 7.2: Results for Closed Loop when x varies

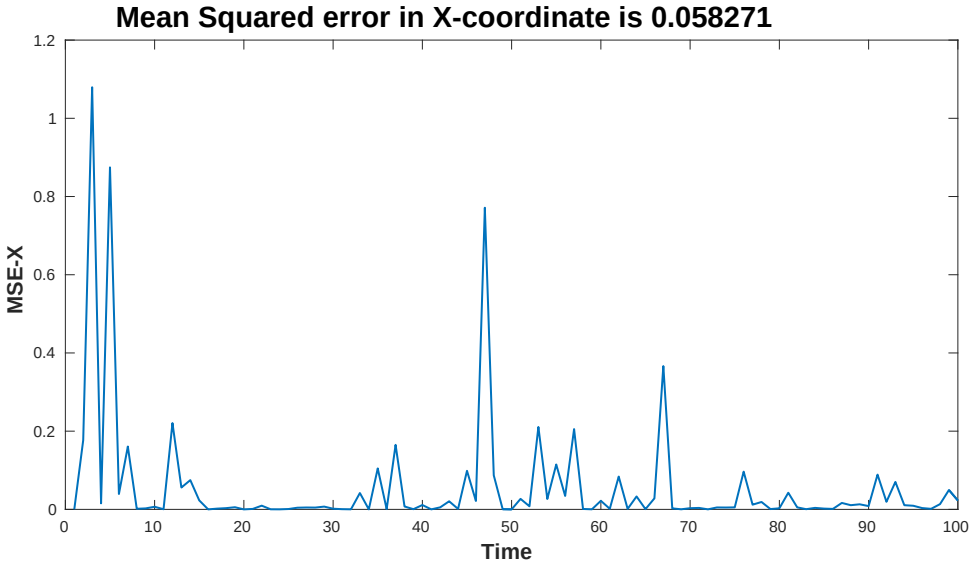

Figure 7.59: Case 7.2: Mean Squared Error for x-axis

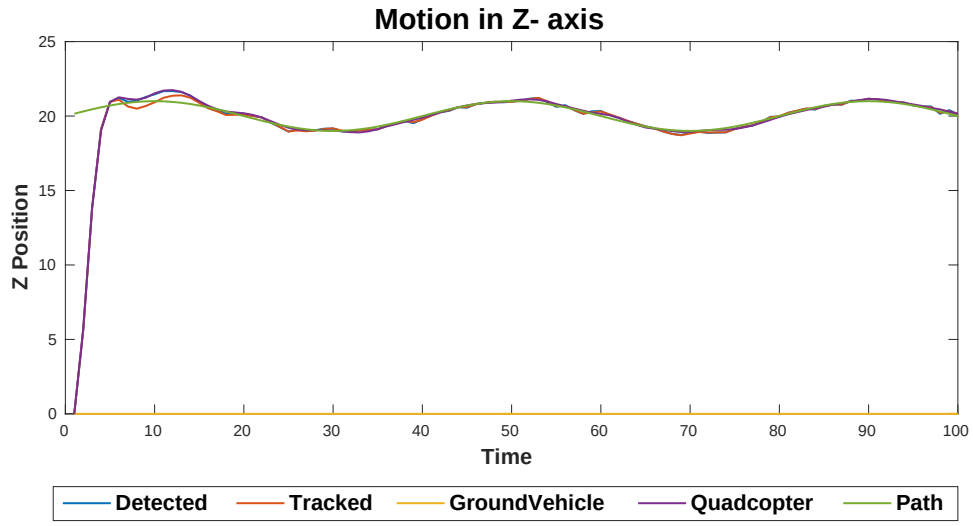

Figure 7.60: Case 7.2: Results for Closed Loop when z varies

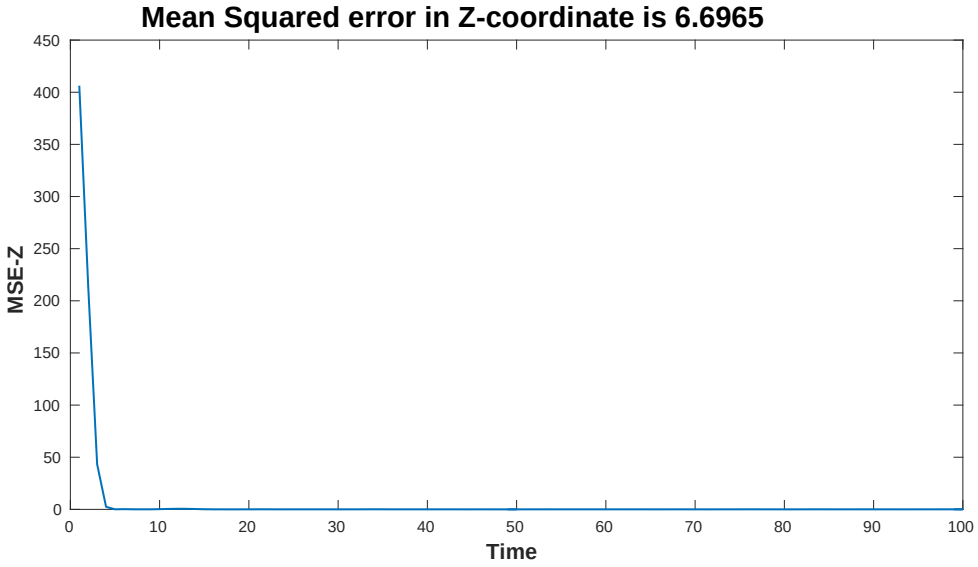

Figure 7.61: Case 7.2: Mean Squared Error for z-axis

7.8.3 Results for Changing Resolution

We now discuss the effect of different camera resolutions the simulation result obtained in the sections 7.8.1 and 7.8.2, we form an error matrix to compare the effect of different camera resolution on our closed loop feedback system. This error matrix indicates the position error in all three direction and the error in detection and tracking results obtained for vision based tracking. From Table 7.2 we can infer that

Table 7.2
Error Matrix with Changing Resolution

Mean Squared Error	**PP = 24**	**PP = 18**	**PP = 48**
x Position Error	0.0477	0.0583	0.0576
y Position Error	0.5780	0.6301	0.5931
z Position Error	6.6963	6.6956	6.6965
x Position Error after T_s	0.0363	0.0471	0.0330
y Position Error after T_s	0.5864	0.6401	0.6020
z Position Error aftre T_s	0.0649	0.0651	0.0641
$(x_{Detected} - x_{Tracked})$	0.0038	0.0040	0.0034
$(y_{Detected} - y_{Tracked})$	0.0226	0.0237	0.0240
$(z_{Detected} - z_{Tracked})$	0.0166	0.0169	0.0155

as the resolution for the camera is decreased the tracking becomes worse and similarly becomes better if resolution is increased. Mostly the effect can be seen only for the y-axis which is as expected because the pose estimation from blurred LED images obtained is more difficult. Although there is not a huge difference so we can access the trade-off between the amount of improvement and the increase in cost and computational requirements.

7.9 Overall Results for Simualtion

The results obtained from all the simulations are shown in Table 7.3. By examining the position error in the x, y, and z directions we can infer that the closed loop architecture works efficiently and the quadcopter follows the ground vehicle and the specified path. Note that when the quadcopter is kept at a constant position in the x or y direction, while maintaining hover position, the error increases because the ground vehicle is moving and the relative position central PD controller action is adversely affected. This can be attributed to the fact that our ground vehicle moves at a constant velocity in the forward x and y direction. The relative position determined by the joint controller uses the tracked position and the current position of the ground vehicle. The controller parameters are tuned in a way so that any random motion can be tracked by including external noises. These noises affect the controller action when the quadcopter position for two axes are expected to be constant. Although in such cases also the controller tries to follow the desired path by several spikes in the plots are observed when steep position changes are observed for the third axis. When the quadcopter moves in all the three directions the error in the measured position and desired position is considerably small. Simulations that study the effects of the camera parameters show expected results. With longer focal length, the depth of field is reduced and thus localization of the quadcopter in the z direction is improved. Change in camera resolution affects the error only marginally. As a

result it can be concluded that there is a trade-off between slight error improvement and computational power required for higher resolution cameras. It can be easily concluded that the closed loop feedback system developed to remotely navigate the quadcopter from a ground vehicle using only vision sensing performs well.

Table 7.3
Overall Simulation Error Matrix

Simulation Case	**Parameters Values**	$\mathbf{x}_{error}$	$\mathbf{y}_{error}$	$\mathbf{z}_{error}$
Case 1	$x = 4$, $z = 20$, $f = 45mm$, $PP = 24$ & y varies	0.0436	0.1376	2.5050
Case 2	$y = 4$, $z = 20$, $f = 45mm$, $PP = 24$ & x varies	0.0260	0.1847	2.5272
Case 3	$z = 20$, $f = 45mm$, $PP = 24$ & x & y vary	0.1241	2.2258	2.5231
Case 4	$z = 20$, $f = 45mm$, $PP = 24$ & x & y vary with FFC	0.0567	0.7586	2.5231
Case 5	$z = 20$, $f = 45mm$, $PP = 24$ & x & y vary wrt GV	0.1313	0.1126	6.4917
Case 6	$f = 45mm$, $PP = 24$ & x, y & z vary with FFC	0.0477	0.5780	6.6963
Case 7	$f = 25mm$, $PP = 24$ & x, y & z vary with FFC	0.0557	0.6670	6.6950
Case 8	$f = 65mm$, $PP = 24$ & x, y & z vary with FFC	0.0525	0.6191	6.6950
Case 9	$f = 45mm$, $PP = 18$ & x, y & z vary with FFC	0.0576	0.5931	6.6965
Case 10	$f = 45mm$, $PP = 48$ & x, y & z vary with FFC	0.0543	0.6301	6.6956

Chapter 8

Conclusion

The main objective of this project was to build a closed loop feedback system for navigating a quadcopter from a moving ground vehicle and to then validate the proposed control approaches. Both the vehicles, the unmanned aerial vehicle and the ground vehicle are moving and for pose estimation of the quadcopter only vision based sensing was used. The ground vehicle motion was controlled using the pure pursuit algorithm navigating a predefined path. At each time step the ground vehicle was given a new position to which to move. The ground vehicle was simulated to move around the space with a constant velocity and to follow a desired path.

A Proportional-Integral-Derivative (PID) controller was implemented for position control and attitude stabilization of the UAV. It was augmented with feedforward

control for better tracking and position control of the quadcopter, which reduced the position error substantially. The quadcopter followed a desired path sent directly to the quadcopter dynamic model. The quadcopter was simulated to start from any random initial position and to reach the expected position following the path thereafter.

Once a standalone navigation control for the quadcopter was developed, the next step was to develop a closed loop architecture and use to vision-based estimation to remotely fly the quadcopter using inputs from the ground vehicle. A single camera mounted on the moving ground vehicle looking upwards was used for the vision-based tracking. This camera maintained a constant feed by taking images of the moving quadcopter. Through the images obtained from the camera, a vision based estimation algorithm coupled with a Kalman filter was implemented to get the three dimensional position of the quadcopter. To validate the vision-based estimator and tune the Kalman filter parameters, a simulated video was developed with a randomly moving quadcopter with a given dynamic model.

The quadcopter gets its position updates from the ground vehicle. Once the quadcopter attained an initial pose the closed loop feedback system is initialized. The relative pose with respect to the moving ground vehicle was fed in for an updated navigation point. This was then communicated to the quadcopter using the control output from a joint centralized controller for the closed loop system. Multiple simulations to check the efficiency of the controller with varied inputs with external noise

were done to validate the suggested control approach.

For the simulation, all x, y and z directions were varied. Initially, the quadcopter mostly maintained a constant hover position from the center of mass of the camera to remain in the field of view. The quadcopter followed the ground vehicle and the error was extremely low. The results obtained when the quadcopter was at a constant hover position ensured that the control architecture developed was extremely robust, i.e. the system had the ability to withstand adverse conditions of testing of randomly varying input levels. This robustness was achieved by including external noise for aerial vehicle motion and also tuning the Kalman filter parameters by incorporating measurement and observation noise. Due to this even when the position for z-axis was varied the feedback system worked with low position error.

To further test the vision-based algorithm, camera parameters were varied and the simulation results obtained were as expected. With a longer focal length lens the depth of the field is reduced hence improving z direction tracking. Similarly, the resolution of the camera was also altered. Since, we process the image before detection and tracking the resolution change doesn't have huge impact on the result. If the results from resolution variations are observed closely we infer that with higher resolution the tracking becomes more precise and error reduces.

Possible extensions of this work include implementing this algorithm on an experimental setup to obtain and verify the results in real life scenarios. The experiments

presented in this work are best suited for indoors and do not deal with camera limitations for outdoors such as field of view restriction or environmental hindrances. With higher UAV altitude relative to the ground vehicle the field of view will be a major issue. Also, for the vision based estimation, other algorithms such as color-based tracking could be implemented for better yaw and attitude estimates. Delays due to the communication lapse should also be studied. Other control laws as discussed in [37] that deal with model uncertainty and disturbance rejection could also improve on the proposed system for real-world implementation.

References

[1] P.Henry M. Krainin-D. MAturana D. Fox] A. S. Huang, A. Bachrach and N. Roy. Visual odometry and mapping for autonomous flight using an rgb-d camera. *International Symposium on Robotics Research (ISRR)*, 100:235–252, 2011.

[2] Markus Achtelik, Abraham Bachrach, Ruijie He, Samuel Prentice, and Nicholas Roy. Autonomous navigation and exploration of a quadrotor helicopter in gps-denied indoor environments. In *First Symposium on Indoor Flight*, number 2009, 2009.

[3] Gyula Mester Aleksander Rodic. The modeling and simulation of an autonomous quad-rotor microcopter in a virtual outdoor scenario. *Acta Polytechnica Hungarica*, 8(4):107–124, 2011.

[4] Nishaad N. Salvapantula & Karl A. Stol Alex G. Kendall. On-board object tracking control of a quadcopter with monocular vision. *International conference on unmanned aircraft system (ICUAS)*, pages 404–411, 2014.

[5] Adrien Angeli, David Filliat, Stéphane Doncieux, and Jean-Arcady Meyer. 2D Simultaneous Localization And Mapping for Micro Air Vehicles. In *European Micro Aerial Vehicles (EMAV 2006)*, Braunschweig, Germany, July 2006.

[6] A. Zul Azfar and D. Hazry. Simple approach on implementing imu sensor fusion in pid controller for stabilizing quadrotor flight control. *EEE 7th International Colloquium on Signal Processing and its Applications, Penang*, pages 28–32, 2011.

[7] D. Bohdanov and H.H.T. Liu. Vision-based quadrotor micro-uav position and yaw estimation and control. *AIAA Guidance, Navigation, and Control Conference*, 2012-5048.

[8] Denys Bohdanov. Quadcopter uav control for vision- based moving target tracking task. Master's thesis, University of Toronto, 2012.

[9] C. Yousheng C. Hui and L. Xiaokun. Autonomous takeoff, tracking and landing of a uav on amoving ugv using onboard monocular vision. *IEEE 32nd Chinese Control Conference (CCC)*, pages 5895–5901, 2013.

[10] Alejandro Enrique Dzul Lopez Rogelio Lozano Carillo, Luis Rodolfo Garcia and Claude Pegard. Combining stereo vision and inertial navigation system for a quad-rotor uav. *Journal of Inetlligent and Robotic Systems*, 65:373–387, 2012.

[11] Koray Celik and Arun K. Somani. Monocular vision slam for indoor aerial

vehicles. *IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 1566–1573, 2013.

[12] J-Y Choi and S-G Kim. Collaborative tracking control of uav-ugv. *11th International Conference on Ubiquitous Robots and Ambient Intelligence (URAI), Kuala Lumpur*, pages 641–645, 2014.

[13] N.M. Fonseca Ferreira C.Lebres, V.Santos and J.A. Tenreiro Machado. Application of fractional controllers for quad rotor, nonlinear science and complexity. *Nonlinear Science and Complexity,Springer, Netherlands*, 6:303–309, 2011.

[14] L. Mirea D. Gheorghita, L. Vintu and C. Braescu. Quadcopter control system. *19th International Conference on System theory, Control and Computing (ICSTCC)*, pages 421–430, 2015.

[15] Dustin Deneault. Tracking ground targets with measurements obtained from a single monocular camera mounted on an unmanned aerial vehicle. Master's thesis, Kansas State University, 2007.

[16] M. Mehrer S. Moreno D.Hartman, K. Landis and J. Kim. *Quadcopter Dynamic Modeling and Simulation.* MATLAB and Simulink Student Design Challenge, 2014.

[17] L. Mihaylova E. Semerdjiev and X. Li. An adaptive imm estimator for aircraft tracking. *International Conference on Information Fusion, Sunnyvale, CA*, 1999.

[18] Frederic Guinand Jean-Francois Brethe Herve Pelvillian Jean-Yves Parede El Houssein Chouaib Harik, Francois Guerin. Fuzzy logic controller for predictive vision-based target tracking with an unmanned aerial vehicle. *Advanced Robotics*, 31.

[19] S. Esmailifar and F. Saghafi. Autonomous unmanned helicopter landing system design for safe touchdown on 6dof moving platform. *Fifth International Conference on Autonomic and Autonomous Systems, Valencia*, pages 245–250, 2009.

[20] I. Fantoni F. Kendoul, K. Nonami and R.Lozano. An adaptive vision-based autopilot for mini flying guidance, navigation and control. *Journal of Autonomous Robots*, 27:165–174, 2009.

[21] Farbod Fahimi and Karansingh Thakur. An alternative closed-loop vision-based control approach for unmanned aircraft systems with application to a quadrotor. *International Conference on Unmanned Aircraft Systems (ICUAS),Atlanta,GA*, pages 353–358, 2013.

[22] S.L. Waslander G.M. Hoffmann, H. Huang and C.J. Tomlin. Precision flight control for a mutli-vehicle quadrotor helicopter test-bed. *Control Engineering Practice*, 9:1023–1036, 2011.

[23] H.H. Bulthoff Grabe, Volker and Paolo Robuffo Giordano. On-board velocity estimation and closed-loop control of a quadrotor uav based on optical flow. *IEEE*

International Conference on Robotics and Automation (ICRA), pages 491–497, 2012.

[24] F-X. Russotto Tarek Hamel Herisse, Bruno and Robert Mahony. Hovering flight and vertical landing control of a vtol unmanned aerial vehicle using optical flow. *IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS),Nice*, pages 801–806, 2008.

[25] Y. Zhang I. Sadaghzadeh, A. Mehta and C. Rabbath. Fault-tolerant trajectory tracking control of a quadrotor helicopter using gain scheduled pid and model reference adaptive control. *Report for Defense Research and Development*, 2, 2011.

[26] C.Lebres V.Santos J. Coelho, R. Neto. Application of fractional algorithms in control of a quad rotor flight. *Conference on Nonlinear Science and Complexity*, 2008.

[27] H. Jung J-K Lee and H. Hu. Collaborative control of uav/ugv. *11th International Conference on Ubiquitous Robotics and Ambient Intelligence (URAI), Kuala Lumpur*, pages 641–645, 2014.

[28] Steven L.Waslander John M. Daly, Yan Ma. Coordinated landing of a quadcopter on a skid-steered ground vehicle in the presence of time delays. *IEEE International Conference on Intelligent Robots and Systems*, 38:179–191, 2011.

[29] J. Kim and S. Sukkarieh. Airborne simultaneous localization and map building.

IEEE International Conference on Robotics and Automation (ICRA), 1:406–411, 2003.

[30] B. Grocholsky L. Chaimowicz and J.F. Keller. Experiments in multirobot air-ground coordination. *IEEE International Conference on Robotics and Automation (ICRA)*, 4:4053–4058, 2004.

[31] J. Li and Y. Li. Dynamic analysis and pid control for a quadrotor. *IEEE International Conference on Mechatronics and Automation (ICMA),Beijing*, pages 573–578, 2011.

[32] Kolja Kuhnlenz M. Achtelik, Tianguang Zhang and Martin Buss. Visual tracking and control of a quadcopter using a stereo camera and inertial sensors. *International Conference on Mechatronics and Automation (ICMA),Changchun*, pages 2863–2869, 2009.

[33] Ruijie He Sameul Prentice M. Achtelik, Abraham Bachrach and Nicholas Roy. Stereo vision and laser odometry for autonomous helicopters in gps-denied indoor environments. *SPIE Defense, Security, and Sensing. International Society for Optics and Photonics*, 2009.

[34] E. Mueggler M. Faessler and K. Schwabe. A monocular pose estimation system based on infrared leds. *IEEE International Conference on Robotics and Automation (ICRA), Hong Kong*, pages 907–913, 2014.

[35] Y. Bar-Shalom M. Yeddanapudi and K. Pattipati. imm estimation for multitarget-multisensor air traffic surveillance. *IEEE*, 85:80–96, 1997.

[36] D. Bersak P. Pounds and A. Dollar. Stability of small-scale uav helicopters and quadrotors with added payload mass under pid control. *Journal of Autonomous Robots*, 33:129–142, 2012.

[37] S. Agarwal H. Pota S. Oh, K. Pathak and M. Garratt. Autonomous helicopter landing on a moving platform using a tether. *IEEE International Conference on Robotics and Automation (ICRA)*, pages 3960–3965, 2005.

[38] S. Saripalli and G. Sukhatme. Landing on a moving target using an autonomous helicopter. *Field and Service Robotics, Springer*, 24:277–286, 2006.

[39] K. Pattipati T. Kirubarajan, Y. Bar-Shalom and I. Kadar. Ground target tracking with variable structure imm estimator. *IEEE Transactions on Aerospace and Electronic Systems*, 36:26–46, 2000.

[40] M. Nitsche T. Krajnik and J. Faigl. A practical multirobot localization system. *Journal of Intelligent Robotic System*, 76:539–562, 2014.

[41] K. Khnlenz T. Zhang, W. Li and M. Buss. Multi-sensory motion estimation and control of an autonomous quadrotor. *Journal of Autonomous Robots*, pages 1493–1514, 2011.

[42] H. Voos and H. Bou-Ammar. Nonlinear tracking and landing controller for

quadrotor aerial robots. *IEEE International Conference on Control Aoolications (CCA), Yokohama*, pages 2136–2141, 2010.

[43] T. Zhang W. Li and K. Klihnlenz. A vision-guided autonomous quadrotor in an air-ground multi-robot system. *IEEE International Conference on Robotics and Automation (ICRA)*, pages 2980–2985, 2011.

[44] Cihat Bora Yigit and Erdinc Altug. Visual attitude stabilization of a unmanned helicopter in unknown environments with an embedded single-board computer. *IEEE International Symposium on Robotic and Sensors Environments (ROSE), Magdeburg*, pages 49–54, 2012.

[45] R. Zhang and H.H.T. Liu. Vision-based relative altitude estimation of small unmanned aerial vehicles in target localization. *American Control Conference,San Francisco, CA*, pages 4622–4627, 2011.