Quadrotor UAV Control for Vision-based Moving Target Tracking Task

by

Denys Bohdanov

A thesis submitted in conformity with the requirements for the degree of Masters of Applied Science
Graduate Department of Aerospace Engineering
University of Toronto

© Copyright 2012 by Denys Bohdanov
The problem of stand-off tracking of a moving target using a quadrotor unmanned aerial vehicle (UAV) based on vision-sensing is investigated. A PID (Proportional-Integral-Derivative) controller is implemented for attitude stabilization of the quadrotor. An LQG-based (Linear-Quadratic-Gaussian) control law is designed and implemented for position control of the quadrotor for a moving target tracking task. A novel vision-based estimation algorithm is developed, enabling estimation of quadrotor’s position, altitude and yaw relative to the target based on limited information about the target. Two image processing algorithms are implemented and compared for the task of feature detection and feature tracking in a series of images. Image processing algorithms are integrated with quadrotor control and experiments are performed to validate proposed control and estimation approaches.
# Contents

1 Introduction 1
  1.1 Literature Review .................................................. 2
  1.1.1 Quadrotor Control ................................................... 2
  1.1.2 Vision-based Tracking ............................................ 4
  1.2 Motivation and Research Objective ................................. 5
  1.3 Thesis Outline ....................................................... 6

2 Quadrotor Model 8
  2.1 Quadrotor Dynamics ................................................... 8
  2.2 Linearized System Model ............................................. 14
    2.2.1 Model Validation .................................................. 16

3 Quadrotor Control 20
  3.1 Attitude Stabilization ............................................... 21
  3.2 Position Tracking Control ........................................... 24
    3.2.1 Way-point Following ............................................ 24
    3.2.2 Target Tracking .................................................. 26

4 Vision-based Estimation 29
  4.1 Two Point Estimation ................................................ 29
    4.1.1 Sensitivity Analysis ............................................. 35
  4.2 Image Processing .................................................... 38
    4.2.1 Optic Flow Feature Tracking ................................... 39
    4.2.2 Colour-based Feature Tracking ................................. 42
  4.3 Kalman Filter ....................................................... 44

5 Simulations 46
  5.1 Stationary Target .................................................... 47
  5.2 Moving Target ...................................................... 51
6 Experiments

6.1 Experimental Setup ........................................ 56
6.2 Way-point Following ........................................ 59
6.3 Stationary Target ........................................... 60
   6.3.1 Optic Flow-based Hovering Results ..................... 60
   6.3.2 Colour-based Hovering Results ........................ 62
6.4 Moving Target ............................................. 63
   6.4.1 Attitude Control ....................................... 64
   6.4.2 Linear Path Experiments ............................... 64
   6.4.3 Circular Path Experiments ............................. 65

7 Conclusions .................................................. 71

Bibliography .................................................. 73
List of Tables

4.1 Parameter variation. .................................................. 35
4.2 Change of estimates with respect to changes in parameters. .......... 38
5.1 Standard deviation of errors for Kalman filter estimates. ............... 55
6.1 Standard deviation of estimation errors for colour-based hovering. ... 63
6.2 Standard deviation of tracking errors for Linear and Circular path experiments. .................................................. 65
List of Figures

2.1 Quadrotor motion principle .............................................. 9
2.2 Quadrotor structure and coordinate frame definition .................. 14
2.3 Block diagram for x direction. ............................................. 15
2.4 Response of non-linear and linearized models (0rad roll). ............... 17
2.5 Response of non-linear and linearized models (0.01rad roll). ............. 18
2.6 Altitude predicted by non-linear and augmented linearized models. ...... 19

3.1 Control structure. .......................................................... 20
3.2 PID control for pitch and roll angles. .................................... 22
3.3 Kalman Estimates .......................................................... 23

4.1 Definition of reference frames. .............................................. 30
4.2 Estimation errors due to errors in target length, roll and pitch angles. .... 36
4.3 Estimation errors due errors in focal lengths and image processing. .... 37
4.4 Optic flow feature detection and tracking. .................................. 40
4.5 Colour-based image processing. ............................................ 43

5.1 Simulation setup for stationary target simulations ......................... 46
5.2 Simulation of pitch and roll angle control using PID ..................... 47
5.3 Errors in relative position and yaw angle estimation using the linear observer 48
5.4 Relative velocity estimation using the linear observer .................... 49
5.5 Simulation results of vision-based hovering ............................... 50
5.6 Simulation setup for moving target tracking simulations .................. 51
5.7 PID control for pitch and roll angles. ...................................... 52
5.8 Position and orientation estimates from Kalman filter for piece-wise constant velocity target. ............................................ 53
5.9 Velocity estimates from Kalman filter for piece-wise constant velocity target. 54
5.10 Position and orientation tracking errors. ................................... 55

6.1 Quadrotor UAV and ground robot used in the experiments ............... 56
<table>
<thead>
<tr>
<th>Section</th>
<th>Title</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>6.2</td>
<td>Laboratory setup used for experiments</td>
<td>57</td>
</tr>
<tr>
<td>6.3</td>
<td>Implementation outline</td>
<td>58</td>
</tr>
<tr>
<td>6.4</td>
<td>LQG waypoint following</td>
<td>59</td>
</tr>
<tr>
<td>6.5</td>
<td>Quadrotor relative position and heading estimation for optic flow-based hovering</td>
<td>61</td>
</tr>
<tr>
<td>6.6</td>
<td>Experiment results of relative position and yaw angle estimation for color-based hovering</td>
<td>66</td>
</tr>
<tr>
<td>6.7</td>
<td>PID control for pitch and roll angles during target tracking task</td>
<td>67</td>
</tr>
<tr>
<td>6.8</td>
<td>Raw camera data compared to Kalman filter estimates for Linear Path experiments</td>
<td>67</td>
</tr>
<tr>
<td>6.9</td>
<td>Quadrotor and GMT positions for Linear Path experiments.</td>
<td>68</td>
</tr>
<tr>
<td>6.10</td>
<td>Raw camera data compared to Kalman filter estimates for Circular Path experiments</td>
<td>69</td>
</tr>
<tr>
<td>6.11</td>
<td>Quadrotor and GMT positions for Circular Path experiments.</td>
<td>70</td>
</tr>
</tbody>
</table>
Nomenclature

$\alpha$ Torque coefficient

$\bar{u}_{1,2}, \bar{v}_{1,2}$ Guessed image coordinates of the two features (pixels)

$\zeta$ Relative position of the quadrotor with respect to target (m)

$\lambda$ Update rate, positive constant

$A, B, C$ State space system matrices

$C_{1,2}$ Transformation matrices

$C_{oq}$ Rotation matrix from quadrotor body frame to $F_o$

$I$ Moment of inertia matrix ($kg \cdot m^2$)

$K$ Feedback gain matrix

$Q, R$ LQR gain matrices

$w$ Zero mean Gaussian noise

$x, y, u$ State, measurement and control inputs respectively

$\mu_{i,j}$ Central moments

$\omega_x, \omega_y, \omega_z$ Rotational velocities (rad/sec)

$\phi, \theta, \psi$ Roll, Pitch, Yaw Euler angles (rad)

$\tau$ System delay constant (s)

$\bar{a}, \bar{b}, \bar{c}$ Position vectors (normalized)

$c_{x,y}$ Center of coordinates in the image plane (pixels)
\( E \)  Error function

\( F_i \)  Reference frame \( i \)

\( f_{x,y} \)  Focal lengths of the camera (pixels)

\( g \)  Acceleration due to gravity (\( m/s^2 \))

\( I \)  Image intensity (normalized)

\( J \)  LQR cost function

\( K \)  Window size for autocorrelation matrix (pixels)

\( k_p, k_d, k_i \)  PID gains

\( l \)  Quadrotor moment arm (m)

\( m \)  Mass of the quadrotor (kg)

\( M(x, y) \)  Autocorrelation matrix

\( M_{i,j} \)  Spatial moment of intensity distribution

\( M_x, M_y, M_z \)  Moments about corresponding axis of the quadrotor (\( N \cdot m \))

\( T_{1-4} \)  Thrust force from each of the four motors (N)

\( u_{1,2}, v_{1,2} \)  Coordinates of features 1 and 2 in the image plane (pixels)

\( v_x, v_y, v_z \)  Translational velocities (m/s)

\( w_{i,j} \)  Weights for autocorrelation matrix (normalized)

\( z_c^{D,E} \)  Z coordinate of points D and E in \( F_c \)
Chapter 1

Introduction

Vision-based control of micro unmanned aerial vehicles (micro-UAVs) is an exciting area of research with a wide range of practical applications. UAVs are very versatile and they are quickly finding new uses. Quadrotor UAVs are of particular interest in research and field applications due to their small size, great manoeuvrability and hovering capability. Omni-directional flight capability also extends the range of possible applications. Quadrotors can get close to targets of interest and can remain undetected at lower altitudes than planes and larger UAVs. However, these advantages come at a cost. Quadrotors have a limited payload capacity because they require four motors for operation, which consumes more power than conventional UAVs. This reduces the number of sensors and mission critical equipment they are able to carry. Due to this limitation researchers have turned to vision-based techniques for navigation of UAVs. Vision-based navigation provides a large number of useful features that extend quadrotors capabilities. Vision sensors are passive, low in cost and weight and provide a great source of data for various missions where UAVs would get deployed. These include such diverse applications as search and rescue missions, traffic and pipeline inspection, forest fire monitoring and a wide range of military applications. Usage of vision sensors becomes even more essential when other sensing data are not available (e.g. GPS-denied environment). One
of the applications where UAVs are being utilized with increasing frequency is target
tracking. UAVs equipped with vision sensors are well suited for this kind of task and are
much safer and cheaper to operate than manned vehicles. Work presented in this thesis
deals primarily with a task of tracking a moving target using a quadrotor UAV. The
primary contribution of the thesis is performing a tracking task based on vision sensing
and image processing alone. A novel image processing algorithm is developed in order to
perform tracking reliably. It allows for accurate estimation of position, altitude and yaw
of the UAV relative to the target using limited information about the target of interest.

1.1 Literature Review

1.1.1 Quadrotor Control

Quadrotor control for various applications have been studied for a long time and
significant progress has been made in the field. Although quadrotors are now capable
of autonomous flight and aggressive manoeuvring, a number of key issues remain open,
with no single accepted solution. Research presented in References [18], [6] presents a
PID controller for attitude stabilization of a quadrotor UAV. This research successfully
demonstrated through experiments that a simple PID control is sufficient for stable flight.
Other research work done in the field moves away from PID and investigates more so-
phisticated controls for improved performance for a variety of particular scenarios such
as a sliding mode controller or backstepping control schemes [31, 24, 15]. Recently there
has been a lot of interest in applying adaptive control schemes for quadrotor control. It
is believed that these controls can be used to alleviate model uncertainties, varying pay-
loads and disturbance rejection. In References [3], [2] switching model predictive control
was used for quadrotor control in case of wind-gusts and other environmental distur-
bances. Model reference adaptive control was also implemented and compared with gain
scheduled PID control for fault tolerance. Experiments presented in Reference [23] show
that both PID and model reference adaptive controls perform acceptably.

More involved control schemes for quadrotor UAVs typically aim to solve a particular design problem. They often prove harder to design and implement and provide little or no improvement in nominal close-to-hover flight conditions. Experiments presented in Reference [13] show that linear controllers based on linearized equations of motion provide great performance around hover conditions. PD and LQR control schemes were also successfully used for quadrotor control for the task of trajectory tracking with visual feedback [28, 10]. Experiments presented in Reference [22] prove quadrotor stability with PID control even during flight with varying payload. Research in Reference [30] demonstrates feasibility of using a PID controller for target tracking task for a quadrotor UAV. Previous research focuses primarily on the quadrotor control and overlooks its integration with various image processing techniques. Work presented in this thesis aims to bridge this gap by using an observer-based controller to integrate the image processing estimation and controller design. Based on this body of previous research linear controllers were chosen for integration with image processing algorithms. Once the vision-based estimation is successfully integrated with proposed control structure it can be replaced with other control methods for improved performance if required. While there is a variety of approaches dealing with the problem of quadrotor attitude stabilization and position control, work presented here is primarily concerned with integration of control with vision-based estimation algorithms. For this purpose a PID control law is chosen for attitude stabilization of the quadrotor and LQG augmented with integral control is used for position control and target tracking. PID and LQG controls were chosen because previous work in the field proves reliability of both approaches and shows them to be suitable for most applications [17, 21].
1.1.2 Vision-based Tracking

In this work target tracking is achieved by image processing and localization of the UAV with respect to the target one frame at a time. There is a wide range of algorithms utilizing multiple frames for UAV localization but they are computationally expensive and are difficult to implement on the low-power computers typically found on-board of smaller UAVs. One of the main challenges in tracking is estimation of relative altitude between the UAV and the target. Estimation of quadrotor’s altitude is challenging based on monocular camera measurements due to the projection limitations. Other sensors such as barometers, GPS or laser-based sensors are usually used for estimation. GPS is not always suitable for altitude estimation at low altitudes due to low update rates and low accuracy of estimates. Kendoul and et al. [16] proposed a vision based autopilot relying on optic flow. This technique as well as others relying on optic flow provides accurate altitude estimates only when there is significant motion of the field of view between frames, which is not the case during hover for example. Therefore, pressure sensors are often required by quadrotors for hovering and rotational motion. Pressure sensors are rather unreliable and their performance depends largely on weather conditions. Alternatively, active sensing such as sonar has also been investigated for altitude estimation [5]. These were proven to work indoors with some limitations but their accuracy decreases for higher altitudes. Various vision-based algorithms other than optic flow have also been used for target tracking as well [27]. These algorithms rely on color matching or image templates which assume a priori knowledge of target’s appearance [7, 12]. Field tests show that data provided by these algorithms is too noisy for altitude hold tasks. Even indoors at low altitudes other active sensors such as sonar or range finders are required for stable flight. Work presented here aims to achieve target tracking based on vision sensing alone. This requires accurate altitude and yaw estimation based on image processing and it is one of the issues addressed in this work.
1.2 Motivation and Research Objective

The literature review outlined above provides a summary of recent developments in quadrotor control and vision-based estimation. The issue this research is addressing is achieving reliable target tracking for a quadrotor UAV based on monocular camera sensing alone. Position, altitude and yaw of the quadrotor UAV relative to the target of interest are to be estimated based on images obtained through vision-based tracking alone. Proposed algorithms and control design is to be implemented and tested on a quadrotor UAV and a ground robot.

Research done in this work assumes that one dimension of the target of interest is known. It could be a width of the road or length of a vehicle or another object of interest. A novel algorithm presented in this work enables efficient and accurate altitude and yaw estimation of the quadrotor with respect to a ground target [8]. It relaxes the assumption made by Zhang and et al.[29] that the yaw of the quadrotor is known, and instead estimates it along with position of the UAV. The main contributions of this work are the development and implementation of a simple and reliable quadrotor controller for attitude stabilization and target tracking, a novel algorithm for estimating relative position, altitude and yaw of a quadrotor relative to a target based on limited knowledge of the target, and applying Kalman filtering to provide tracking capability in the case of partial target occlusion or temporary target loss.

Conventionally, unmanned aircraft have downward facing cameras mounted on them as one of the primary payload sensors. These are used for teleoperation, reconnaissance, surveillance and target tracking. Work presented here utilizes a camera mounted on the moving target rather than on the UAV. The camera is facing upwards and provides images of the UAV instead of the target itself. This does not change the tracking task or the geometry of the problem in terms of algorithm and control formulation, however, it simplifies experimental setup by eliminating the need to mount a camera on the quadrotor. In practical applications this setup is being considered for a number of different purposes.
Having a camera on the ground was shown to aid in quadrotor pose estimation and improve quadrotor control [4]. Having the camera on the ground vehicle is also useful in the case of ground-air cooperation of heterogeneous systems. For example, coordinated landing of a quadrotor on a moving ground vehicle was achieved using the camera on the ground approach [11]. This is also being considered for reliable and robust tethering of quadrotors to ground vehicles in rescue missions where UAVs have to fly low over the ground vehicles. Quadrotor UAVs are finding applications in the field of simultaneous localization and mapping. They are being mounted with stereo camera systems [1] or Kinect sensors [14]. Due to limited payload capability it becomes problematic to carry a downwards facing camera in addition to stereo cameras. In this case placing the camera on the unmanned ground vehicle (UGV) is advantageous.

1.3 Thesis Outline

This thesis is divided into 7 chapters. The first chapter is an introduction which describes the motivation behind the project and provides a brief summary of previous work done in the fields of quadrotor control and vision-based target tracking. Chapter 2 discusses the model of the quadrotor used in the design of control laws. A derivation of the non-linear equations of motion is provided along with the linearization of the model. Chapter 3 presents the control law design for attitude stabilization of the UAV, and the position control and target tracking control design. Image processing and estimation algorithm is outlined in Chapter 4. It also discusses two image processing techniques used in the project: optical flows and colour-based image processing. Results of simulations performed to validate estimation and control strategies are presented in Chapter 5. This chapter also outlines the overall system structure and provides a discussion of the simulation results. Chapter 6 discusses experiments performed to verify how well the system performs during actual tracking tasks. Experimental setup and hardware used during
the experiments are described. Discussion of experimental results and their comparison to simulation results is provided. Finally, Chapter 7 provides a summary of the project and briefly discusses possible extensions for future research.
Chapter 2

Quadrotor Model

2.1 Quadrotor Dynamics

Quadrotors are under-actuated systems in which 4 inputs are used to control motions of 6 degrees of freedom. Figure 2.1 demonstrates the motion principle for a quadrotor UAV. The movement in x and y directions is controlled by roll and pitch respectively. In order to increase roll, thrust on motor 1 is decreased and thrust on motor 3 is increased. This creates a moment which tilts the quadrotor and creates a component of thrust pointing in the x direction. Control for the pitch and y direction works similarly with motors 2 and 4. In order to control altitude, thrust on all motors is added or decreased equally. Decreasing thrust on a pair of motors and increasing it on the counter-rotating pair of motors by the same amount creates a moment about z axis of the quadrotor and causes it to yaw.

Dynamic equations of motion of the quadrotor UAV need to be derived in two reference frames: the inertial frame and the body-fixed frame. Position and velocity of the quadrotor is easily expressed and measured in an inertial reference frame. Orientation and rates of roll, pitch and yaw are usually measured in the body-fixed frame using gyroscopes, accelerometers or magnetometers so the dynamic equations of motion have to be
Chapter 2. Quadrotor Model

Figure 2.1: Quadrotor motion principle

expressed in that frame. In order to derive equations of motion we define the following state vectors:

\[
\mathbf{x}_1 = [x, y, z]^T \quad \mathbf{x}_2 = [\phi, \theta, \psi]^T \\
\mathbf{v} = [v_x, v_y, v_z]^T \quad \boldsymbol{\omega} = [\omega_x, \omega_y, \omega_z]^T
\]  

\hspace{1cm} (2.1)

\hspace{1cm} (2.2)

where \( \mathbf{x}_1 \) is a position vector defined relative to the inertial reference frame with each of the components expressed in the inertial frame, and linear and angular velocities as well as orientation are defined relative to the body fixed reference frame. \( \theta, \phi, \psi \) are the standard Euler angles, being pitch, roll and yaw of a quadrotor respectively; \( v_x, v_y, v_z \) and \( \omega_x, \omega_y, \omega_z \) are the translational and rotational velocities of the quadrotor.

Relationship between linear velocities and angle rates in inertial and body-fixed frames can be expressed using the following two transformation matrices:

\[
\begin{bmatrix}
\dot{x}_1 \\
\dot{x}_2
\end{bmatrix} =
\begin{bmatrix}
C_1(x_2) & 0 \\
0 & C_2(x_2)
\end{bmatrix}
\begin{bmatrix}
v \\
\omega
\end{bmatrix}
\]  

\hspace{1cm} (2.3)
Chapter 2. Quadrotor Model

\[
C_1 = \begin{bmatrix}
c\psi c\theta & -s\psi c\phi + c\psi s\theta s\phi & s\psi s\phi + c\psi c\phi s\theta \\
s\psi c\theta & c\psi c\phi + s\psi s\theta s\phi & -c\psi s\phi + s\theta s\psi c\phi \\
-s\theta & c\theta s\phi & c\theta c\phi 
\end{bmatrix}
\] (2.4)

\[
C_2 = \begin{bmatrix}
1 & s\phi t\theta & c\phi t\theta \\
0 & c\phi & -s\phi \\
0 & \frac{s\phi}{c\phi} & \frac{c\phi}{c\phi}
\end{bmatrix}
\] (2.5)

where \(c, s, t\) denote cosine, sine and tangent respectively. It is more convenient to write equations of motion of the quadrotor in terms of body-fixed velocities and angular rates. In order to do so we apply the Lagrangian approach that is extended to quasi-coordinates [20]. Quasi-coordinates refer to certain variables that can not be strictly defined. In this case angular velocities in the body-fixed frame can not be integrated directly to obtain absolute orientation of the quadrotor. In quasi-coordinate formulation, the components of angular velocity vector are derivatives of quasi-coordinates defined only in terms of their differentials with respect to time.

\[
L = T - V
\] (2.6)

\[
L = \frac{1}{2}m v^T v + \frac{1}{2}\omega^T \mathbf{I} \omega + mgz
\] (2.7)

\[
\mathbf{I} = \begin{bmatrix}
I_{xx} & 0 & 0 \\
0 & I_{yy} & 0 \\
0 & 0 & I_{zz}
\end{bmatrix}
\] (2.8)

where \(m\) and \(\mathbf{I}\) are the mass and the moment of inertia of the quadrotor respectively. It is assumed that the quadrotor is symmetric and the moment of inertia matrix is diagonal. Furthermore \(I_{xx} = I_{yy}\) due to symmetry. The quasi-coordinate Lagrangian formulation
then becomes [20]:

$$\frac{d}{dt} \left( \frac{\partial L}{\partial \dot{\mathbf{v}}} \right)^T + \mathbf{\omega} \times \frac{\partial L}{\partial \mathbf{v}} - \mathbf{T}_1^T \frac{\partial L}{\partial \mathbf{x}_1} = \mathbf{\tau}_1$$

(2.9)

$$\frac{d}{dt} \left( \frac{\partial L}{\partial \mathbf{\omega}} \right)^T + \mathbf{v} \times \frac{\partial L}{\partial \mathbf{v}} + \mathbf{\omega} \times \frac{\partial L}{\partial \mathbf{\omega}} - \mathbf{T}_2^T \frac{\partial L}{\partial \mathbf{x}_2} = \mathbf{\tau}_2$$

Taking partial derivatives and performing cross-product operations the following six equations of motion are obtained:

$$m[\dot{v}_x - v_y \dot{\omega}_z + v_z \dot{\omega}_y - g \theta] = 0 \quad (2.10)$$

$$m[\dot{v}_y - v_z \dot{\omega}_x + v_x \dot{\omega}_z + g c \theta \phi] = 0 \quad (2.11)$$

$$m[\dot{v}_z - v_x \dot{\omega}_y + v_y \dot{\omega}_x + g c \theta \phi] = u_1 \quad (2.12)$$

$$I_{xx} \dot{\omega}_x + (I_{zz} - I_{yy}) \omega_y \omega_z = u_2 \quad (2.13)$$

$$I_{yy} \dot{\omega}_y + (I_{xx} - I_{zz}) \omega_z \omega_x = u_3 \quad (2.14)$$

$$I_{zz} \dot{\omega}_z = u_4 \quad (2.15)$$

$$\mathbf{\tau} = \begin{bmatrix} 0 \\ 0 \\ u_1 \\ u_2 \\ u_3 \\ u_4 \end{bmatrix} = \begin{bmatrix} 0 \\ 0 \\ T_1 + T_2 + T_3 + T_4 \\ (T_4 - T_2)l \\ (T_3 - T_1)l \\ (T_2 + T_4 - T_1 - T_3)\alpha \end{bmatrix} \quad (2.16)$$

$T_{1-4}$ are the forces applied by each of the four motors, $l$ is the length of the moment arm from the center of gravity to each of the motors and $\alpha$ is the torque coefficient which is determined experimentally. It depends on a number of factors including the size of propeller blades, their pitch angle, angular velocity, and air density [26]. Equations of motion derived above are formulated in the body-fixed frame. They can be converted to the inertial frame as follows. In order to express equations of motion in the inertial
frame of reference we need to express $[v; \omega]^T$ and its derivatives using $[x_1; x_2]$. Using transformations:

$$
\begin{bmatrix}
C_1^{-1} & 0 \\
0 & C_2^{-1}
\end{bmatrix}
\begin{bmatrix}
\dot{x}_1 \\
\dot{x}_2
\end{bmatrix} = \begin{bmatrix} v \\
\omega
\end{bmatrix} 
$$

(2.17)

$$
\begin{bmatrix}
C_1^{-1} & 0 \\
0 & C_2^{-1}
\end{bmatrix}
\begin{bmatrix}
\ddot{x}_1 \\
\ddot{x}_2
\end{bmatrix} - \begin{bmatrix} \dot{C}_1 & 0 \\
0 & \dot{C}_2
\end{bmatrix}
\begin{bmatrix}
v \\
\omega
\end{bmatrix} = \begin{bmatrix} \dot{v} \\
\dot{\omega}
\end{bmatrix} 
$$

(2.18)

the following equations of motion in the inertial reference frame can be obtained:

$$
m\ddot{x} = (s\psi s\phi + c\psi c\phi s\theta)u_1 
$$

(2.19)

$$
m\ddot{y} = (-c\psi s\phi + c\phi s\theta s\psi)u_1 
$$

(2.20)

$$
m\ddot{z} = (c\theta c\phi)u_1 - mg 
$$

(2.21)

$$
M_{x_2}\ddot{x}_2 + \frac{1}{2} M_{x_2}\dddot{x}_2 = 
\begin{bmatrix}
u_2 \\
u_3 c\phi - u_4 s\phi \\
-u_2 s\theta + u_3 c\theta s\phi + u_4 c\theta c\phi
\end{bmatrix} 
$$

(2.23)

where:

$$
M_{x_2} = 
\begin{bmatrix}
I_{xx} & 0 & -I_{xx} s\theta \\
0 & I_{yy}s^2\phi + I_{zz}s^2\phi & (I_{yy} - I_{zz})c\phi c\theta s\phi \\
-I_{xx} s\theta & (I_{yy} - I_{zz})c\phi c\theta s\psi & I_{xx}s^2\theta + I_{yy}c^2\theta s^2\phi + I_{zz}c^2\theta c^2\phi
\end{bmatrix} 
$$

(2.24)

As can be seen the dynamic equations of motion of translational degrees of freedom of the quadrotor are simple in the earth-fixed reference frame. The rotational degrees of freedom are expressed more easily in the body-fixed frame. Therefore, the quadrotor
dynamics can be described by the following equations of motion:

\[
\begin{align*}
    m\ddot{x} &= (\sin\psi\sin\phi + \cos\psi\cos\phi\sin\theta)T \\
    m\ddot{y} &= (-\cos\psi\sin\phi + \sin\theta\sin\psi\cos\phi)T \\
    m(\ddot{z} + g) &= (\cos\theta\cos\phi)T \\
    I_{xx}\dot{\omega}_x &= M_x - (I_{zz} - I_{yy})\omega_y\omega_z \\
    I_{yy}\dot{\omega}_y &= M_y - (I_{xx} - I_{zz})\omega_z\omega_x \\
    I_{zz}\dot{\omega}_z &= M_z
\end{align*}
\]

(2.25)

where \((x, y, z)\) is quadrotor’s position in an inertial frame; \(\phi, \theta\) and \(\psi\) are roll, pitch and yaw angles, respectively; \(T\) denotes the total thrust generated by the four motors; \(M_x\), \(M_y\) and \(M_z\) are moments about corresponding axis generated by thrust differences from opposing motors; \(m\) is Quadrotor’s mass; \(I_{xx}, I_{yy}\) and \(I_{zz}\) are the quadrotor’s inertial moments about corresponding axis. Note that the position dynamics in (2.25) is derived in the inertial frame, while the attitude dynamics is defined in the quadrotor’s body frame.

The total force \(T\) and the moments \(M_{x,y,z}\) are given as follows

\[
\begin{bmatrix}
    T \\
    M_x \\
    M_y \\
    M_z
\end{bmatrix} =
\begin{bmatrix}
    1 & 1 & 1 & 1 \\
    0 & -l & 0 & l \\
    -l & 0 & l & 0 \\
    -\alpha & \alpha & -\alpha & \alpha
\end{bmatrix}
\begin{bmatrix}
    T_1 \\
    T_2 \\
    T_3 \\
    T_4
\end{bmatrix}
\]

(2.26)

where \(T_i\) \((i = 1, 2, 3, 4)\) denotes the thrust generated by the \(i\)th motor of the quadrotor, \(l\) is the distance between a motor and the center of the quadrotor.
2.2 Linearized System Model

In order to stabilize quadrotor’s attitude, the attitude dynamic model is linearized around hover (i.e, $\phi = 0$ and $\theta = 0$). The simplified model is given by

\[
\begin{align*}
I_{xx} \ddot{\phi} &= M_x - (I_{zz} - I_{yy}) \dot{\theta} \dot{\psi} \\
I_{yy} \ddot{\theta} &= M_y - (I_{xx} - I_{zz}) \dot{\psi} \dot{\phi} \\
I_{zz} \ddot{\psi} &= M_z
\end{align*}
\] (2.27)

Attitude of the quadrotor will be stabilized by a PID control law at desired roll, pitch and yaw angles. This enables us to perform attitude stabilization separately from position control and tracking tasks. Quadrotor’s motion on the X-Y plane can be linearized at the hovering state (i.e, $\phi = 0$, $\theta = 0$ and $T = mg$) based on Equations 2.25 and simplified as follows:

\[
\begin{align*}
\ddot{x} &= g \theta \\
\ddot{y} &= -g \phi
\end{align*}
\] (2.28)

where $\theta$ and $\phi$ are pitch and roll angles of the quadrotor. This linearized model assumes that attitude dynamics are controlled independently, so the rate of yaw and the X-Y
position can be decoupled from each other. This assumption is valid for small rates of yaw. In this work, a Linear Quadratic Gaussian (LQG) control method is applied to formulate three separate controllers for the relative yaw angle, and x and y positions.

The linearized model in (2.28) does not take into account dynamics of the actuators. In order to model dynamics of motors used on the quadrotor and include delays into system’s model a first order transfer function is used:

\[ u_T = \frac{1}{\tau s + 1} u \]  

where \( \tau \) is the system delay constant.

After including the actuator dynamics into the system, an open loop state space model of quadrotor’s motion in x and y directions is obtained as follows:

\[ \dot{x}_{x,y} = Ax_{x,y} + Bu_{x,y} \]  

where \( u_x \) and \( u_y \) are the two inputs for the x and y directions. Figure 2.3 shows the block diagram for a translational degree of freedom in the x direction. The system matrices and the system states are given by:

\[
\begin{align*}
A &= \begin{bmatrix}
0 & 1 & 0 \\
0 & 0 & 1 \\
0 & 0 & -1/\tau
\end{bmatrix}, \\
B &= \begin{bmatrix}
0 \\
0 \\
g/\tau
\end{bmatrix}, \\
x_x &= \begin{bmatrix}
x \\
\dot{x} \\
\ddot{x}
\end{bmatrix}, \\
x_y &= \begin{bmatrix}
y \\
\dot{y} \\
\ddot{y}
\end{bmatrix}
\end{align*}
\]  

(2.31)
Similarly, the equations of motion for the yaw degree of freedom can be linearized as follows after taking delays into account:

\[
\dot{\Psi} = A_\psi \Psi + B_\psi M_z, \quad \zeta_\psi = C_\psi \Psi
\]  

(2.32)

where

\[
A_\psi = \begin{bmatrix} 0 & 1 & 0 \\ 0 & 0 & 1 \\ 0 & 0 & -1/\tau \end{bmatrix}, \quad B_\psi = \begin{bmatrix} 0 \\ 0 \end{bmatrix}, \quad C_\psi = \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \end{bmatrix}, \quad \Psi = \begin{bmatrix} \dot{\psi} \\ \ddot{\psi} \end{bmatrix}
\]  

(2.33)

As well, the altitude motion given in Eq.(2.25) can be linearized as follows

\[
m \ddot{z} = T
\]  

(2.34)

Note that in Eq.(2.34) \( z \) actually denotes the deviation of the altitude from a fixed value.

Taking system delays into account altitude dynamics can be represented in state space as follows:

\[
A_z = \begin{bmatrix} 0 & 1 & 0 \\ 0 & 0 & 1 \\ 0 & 0 & -1/\tau \end{bmatrix}, \quad B_z = \begin{bmatrix} 0 \\ 0 \end{bmatrix}, \quad C_z = \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \end{bmatrix}, \quad z = \begin{bmatrix} z \\ \dot{z} \end{bmatrix}
\]  

(2.35)

2.2.1 Model Validation

In order to assess validity of the linearized model, its response to various inputs was compared to the response of the full non-linear model. During the simulation the full non-linear model and the linearized model with uncoupled equations of motion were simulated with identical initial conditions with position, altitude and yaw all set to zero. They were then given inputs: 0rad for roll; sinusoidal with frequency 5rad/sec and
amplitude of 0.08rad for pitch; thrust of 0.01N (additional to the nominal thrust required to hover); constant rate of yaw of 0.00015rad/sec. This particular simulation provides representative results, however, simulations were run for a variety of other inputs and initial conditions. During these runs the quadrotor is assumed to have a stabilization control which stabilizes it at required pitch and roll angles. It does not have a position or yaw control. Figure 2.4 shows the open loop response of the system. As can be seen on Figure 2.4 the response of the linearized model matches closely to that of the full non-linear system for x direction and yaw. For y direction the full model is oscillating due to sinusoidal input for pitch. The full model is able to capture cross-coupling between the x
and y directions where the linearized model is not. It should be noted that cross-coupling
effects are small and quadrotor’s oscillations in y direction are small (0.01m) even though
variation in pitch input is large (±0.08rad). Effects of cross-coupling become larger as
the quadrotor moves away from the hover condition. Figure 2.5 shows response of the

\begin{figure}[h]
\centering
\includegraphics[width=\textwidth]{fig25.png}
\caption{X direction response.}
\end{figure}

\begin{figure}[h]
\centering
\includegraphics[width=\textwidth]{fig25b.png}
\caption{Y direction response.}
\end{figure}

Figure 2.5: Response of non-linear and linearized models (0.01 rad roll).

full non-linear model and the linearized model in x and y directions when roll is set at a
constant value of 0.01 rad. In this case the quadrotor accelerates in y direction. It can be
seen that as velocity of the quadrotor increases responses of the two models diverge. This
is expected as the model was linearized about hover and assumes that various degrees
of freedom of quadrotor are decoupled. Simulation showed that coupling between the
x and y directions is small and can be ignored. This, however, is not the case for the
altitude of the quadrotor. The linearized model assumes that pitch and roll angles have
no effect on quadrotor’s acceleration in vertical direction. As shown in Fig.2.4 that is
a poor approximation. The linearized model predicts that the quadrotor will accelerate
upwards but in fact it loses altitude due to the coupling effects from roll and pitch. In
order to provide better model for vertical movement of the quadrotor in z direction input
to the model was augmented as follows:

\begin{equation}
{u}_{z,\text{aug}} = u_r + g(\cos \theta \cos \phi - 1)
\end{equation}
Figure 2.6: Altitude predicted by non-linear and augmented linearized models.

Fig.2.6 shows that response predicted by the linearized model with augmented input is close to that of the full non-linear model. As will be shown in later chapters altitude is controlled by a PID controller due to the fact that the altitude dynamics are time-varying because the thrust of the quadrotor changes with time as the battery voltage drops during operation. Therefore, the linearized system with augmented input was used in the design of the controller.
Chapter 3

Quadrotor Control

This chapter describes the development and implementation of two types of control for the quadrotor UAV. The attitude stabilization control is designed for stabilizing roll and pitch angles of the quadrotor as well as its altitude. Once that is achieved, quadrotor’s dynamics can be simplified as discussed in the previous chapter. Linear-Quadratic-Gaussian (LQG) control is then applied for stabilization of quadrotor’s position in x and y directions and yaw. Figure 3.1 shows the control architecture implemented on the quadrotor UAV.

![Control structure](image)

Figure 3.1: Control structure.
3.1 Attitude Stabilization

The lower level PID control is responsible for stabilizing the pitch and roll angles at desired values dictated by the higher level position control. The roll and pitch angles and their rates of change are obtained from the on-board magnetometers and gyroscopes on the quadrotor UAV. Control laws for attitude stabilization are given by the following equations:

\[
M_x = k_p(\phi_d - \phi) + k_d(\dot{\phi_d} - \dot{\phi}) + k_i \int (\phi_d - \phi) dt 
\]  \hspace{1cm} (3.1)
\[
M_y = k_p(\theta_d - \theta) + k_d(\dot{\theta_d} - \dot{\theta}) + k_i \int (\theta_d - \theta) dt 
\]  \hspace{1cm} (3.2)

Constant control gains \( k_p, k_d \) and \( k_i \) are taken to be the same for roll and pitch due to the assumption that the quadrotor is axisymmetric. The PID control is able to track the commanded roll and pitch angles best if they are kept small. As will be discussed later, position control is designed such that the commanded roll and pitch angles are small in order to keep target in the field of view and make image processing and estimation simpler and more accurate. The commanded pitch and roll angles are allowed to vary \( \pm 10 \) degrees. As will be shown later in the experiments, this is sufficient to allow the quadrotor to move fast enough for tracking tasks. It should also be noted that placing limits on the pitch and roll angles restricts quadrotor’s acceleration rather than velocity. Figure 3.2 shows performance of the PID control for pitch and roll of the quadrotor for one of the test flights. The reference signal is generated by the higher level control for \( x \) and \( y \) positions described in the next section. As can be seen PID is able to follow the reference signal. There is a delay in the response of the PID. It comes from the actuator dynamics and is taken into account in quadrotor’s model. It was experimentally determined to be 0.2 seconds.

PID control was also used to control the altitude of the quadrotor UAV relative to the ground target. This is done because in our controller implementation the thrust force
Chapter 3. Quadrotor Control

22

(a) Pitch PID control.

(b) Roll PID control.

Figure 3.2: PID control for pitch and roll angles.

of each of the four motors is controlled by a throttle number which ranges from 0 to 1. Throttle number represents a ratio of the voltage applied to each motor over the total voltage in the batteries. The voltage in the batteries declines over the course of a flight making the throttle number a time-dependent variable which is difficult to model. Since it’s variation with time is slow it can be dealt with efficiently by a PID control with dominant integral gain. The PID control law for the altitude is given as:

\[
T = k_{p,z}(z_d - z) + k_{d,z}(\dot{z}_d - \dot{z}) + k_{i,z}\int (z_d - z)dt
\]  

(3.3)

A discrete time Kalman filter is designed in order to fuse the measurements of position from image processing and acceleration measurements from the on-board sensors in order to estimate position and velocity in z-direction for the PID control. Equation 3.4 shows the system model used to estimate quadrotor’s altitude. It is independent of quadrotor’s dynamics. Quadrotor’s acceleration is taken as input to the system with position and velocity in z direction being the state to be estimated. In order to design a Kalman filter, the input and measurements variances must be selected. In this case the acceleration of the quadrotor is the input and the altitude of the quadrotor is the only measurement. Variance in acceleration was determined experimentally. Variance of the position measurement depends on the image processing method and was also determined.
experimentally.

\[
\begin{bmatrix}
\dot{\hat{z}}_n \\
\ddot{\hat{z}}_n
\end{bmatrix} =
\begin{bmatrix}
1 & \Delta t \\
0 & 1
\end{bmatrix}
\begin{bmatrix}
\dot{\hat{z}}_{(n-1)} \\
\ddot{\hat{z}}_{(n-1)}
\end{bmatrix} +
\frac{\Delta t^2}{2}
\begin{bmatrix}
\ddot{\hat{z}}_{(n-1)} \\
\dot{\hat{z}}_{(n-1)}
\end{bmatrix}
\begin{bmatrix}
1 & 0 \\
0 & 1
\end{bmatrix}
\begin{bmatrix}
\dot{z}_n \\
\ddot{z}_n
\end{bmatrix}
\]

Figure 3.3 shows the position and velocity estimates for the altitude of the quadrotor.

Figure 3.3: Kalman Estimates

For velocity estimation another approach aside from Kalman filtering was tested. It involves calculating acceleration from velocity data and omitting estimates of velocity with unreasonably large acceleration. It ensures that only valid velocity estimates are passed on to the control law by putting a limit on change in velocity and omitting all velocity measurements which produce unreasonably large accelerations. This approach, shown on Figure 3.3 as "Processed Derivative", does provide an improvement over raw data. The Kalman filter, however, performs better and it is used in altitude control.
3.2 Position Tracking Control

A position control was developed incrementally. It was first designed for hovering and way-point following tasks for the quadrotor and it was then extended to the task of target following.

3.2.1 Way-point Following

The way-point following control was designed by assuming that the quadrotor is following a constant reference signal. In this case the way-points are modelled as a target with instantaneously changing position and zero velocity. As discussed in the previous chapter a state-space model of quadrotor’s motion on the X-Y plane can be obtained as follows:

\[
\dot{x}_{x,y} = A x_{x,y} + B u_{x,y} \tag{3.5}
\]

Defining \( X_t \triangleq [x_t \ y_t \ z_t]^T \) as the inertial position of the target. Assume the target’s motion on the X-Y plane can be expressed as follows:

\[
\dot{x}_{xt,yt} = A x_{xt,yt} + B_t w_{x,y} \tag{3.6}
\]

where \( x_{xt} = [x_t \ \dot{x}_t \ \ddot{x}_t]^T \), \( x_{yt} = [y_t \ \dot{y}_t \ \ddot{y}_t]^T \), \( w_{x,y} \) denotes zero mean Gaussian noises, and the system matrix \( B_t = [0 \ 1 \ 0]^T \).

Define \( x_{xr} = x_{xt} - x_x \) and \( x_{yr} = x_{yt} - x_y \). The motion of the quadrotor relative to the target is then obtained as:

\[
\dot{x}_{xr} = A x_{xr} + B u_x + B_t w_x, \quad \zeta_x = C x_{xr} \tag{3.7}
\]

\[
\dot{x}_{yr} = A x_{yr} + B u_y + B_t w_y, \quad \zeta_y = C x_{yr} \tag{3.8}
\]
where

\[ C = \begin{bmatrix} 1 & 0 & 0 \\ 0 & 0 & 1 \end{bmatrix} \] (3.9)

The outputs \( \zeta_x \) and \( \zeta_y \) denote the relative position and acceleration in x and y directions, respectively. However, in the case of the waypoint following, the target (a given waypoint) does not accelerate and the outputs of \( \zeta_x \) and \( \zeta_y \) provide quadrotor’s accelerations in respective directions. There are two outputs being observed. They are position of the quadrotor relative to the target and acceleration of the quadrotor given by inertial measurement units (IMUs) on-board of the UAV. Therefore, the target tracking problem can be converted into a regulation problem with output feedback. In this work, two linear observers are implemented to provide full state feedback for the LQG controllers. The observers are given by:

\[
\dot{x}_{xr} = A \dot{x}_{xr} + Bu_x + L(x - C \dot{x}_{xr}) \quad (3.10)
\]

\[
\dot{x}_{yr} = A \dot{x}_{yr} + Bu_y + L(y - C \dot{x}_{yr}) \quad (3.11)
\]

where the gain matrix \( L \) is obtained by solving the algebraic Ricatti equation. A cost function for the optimal position control of the quadrotor is defined as follows:

\[
J = \int (x_r^T Q x_r + u^T R u) dt \quad (3.12)
\]

where the weighting matrices \( Q \) and \( R \) are chosen experimentally such that the tracking error is minimized while keeping control inputs (roll and pitch angles) small.

The LQG control law which minimizes this cost function is defined as \( u = -K x_r \), and the feedback gain matrix \( K \) is the solution of the following algebraic Ricatti equation:

\[
K = R^{-1} B^T P \quad (3.13)
\]

\[
A^T P + PA - PB R^{-1} B^T P + Q = 0 \quad (3.14)
\]
The same control scheme and feedback gain matrix can be directly applied to the y direction. In order to improve the performance of the PID controllers for attitude stabilization and simplify the image processing task, the desired roll and pitch angles must be kept less than ±10 deg.

Similar design can be applied to control of the yaw of the quadrotor with the state-space system described as:

$$\dot{\Psi} = A_\psi \Psi + B_\psi M_z, \quad \zeta_\psi = C_\psi \Psi$$  \hspace{1cm} (3.15)

The relative yaw angle is provided by the estimation algorithm introduced in the next chapter and the rate of yaw is obtained from the on-board sensors. The LQG design for the yaw controller follows the same procedure as the x and y direction motion controllers.

### 3.2.2 Target Tracking

For a target tracking task it is assumed that the target is moving with piece-wise constant velocity and it’s altitude is piece-wise continuous. In order for the quadrotor to be able to track the moving target it’s maximum velocity has to be larger than that of the moving target. In the case of a target with piece-wise constant velocity smaller than the velocity of the quadrotor, the reference position for the quadrotor to follow can be considered as a slowly changing constant reference signal. In order to track a moving target we use tracking via coordinate translation augmented with integral control. Consider the following coordinate translation:

$$\tilde{x}(t) = x(t) - x_d$$  \hspace{1cm} (3.16)

$$C_y x_d = r$$  \hspace{1cm} (3.17)

$$u(t) = -K(x(t) - x_d)$$  \hspace{1cm} (3.18)
where

\[
\begin{bmatrix}
  x_t \\ v_t \\ 0
\end{bmatrix}, \quad C_y = \begin{bmatrix}
  1 & 0 & 0
\end{bmatrix}
\]

Equation 3.19

Applying this coordinate translation to the original state space system model with control gives:

\[
\dot{x}(t) = (A - BK)x + Ax_d
\]

Equation 3.20

At the steady state the error for the system becomes:

\[
A x_d = \begin{bmatrix}
  v_t \\ 0 \\ 0
\end{bmatrix}
\]

Equation 3.21

In case of a way-point following \( v_t = 0 \) and the steady-state error is 0. In order to eliminate this steady state error an integral control is used by appending an integrator to the plant. Integral of the error between reference input and output and the resulting augmented system are defined as follows:

\[
\begin{bmatrix}
  \dot{x} \\ \dot{e}_I
\end{bmatrix} = \begin{bmatrix}
  A & 0 \\ -C_y & 0
\end{bmatrix} \begin{bmatrix}
  x \\ e_I
\end{bmatrix} + \begin{bmatrix}
  B & 0 \\ 0 & I
\end{bmatrix} \begin{bmatrix}
  u \\ r
\end{bmatrix}
\]

Equation 3.22

A state feedback is then generated for the augmented plant. A cost function used to penalize the integral of the error is:

\[
J = \int_{t_0}^{t_f} \begin{bmatrix}
  x^T & e_I^T
\end{bmatrix} \begin{bmatrix}
  0 & 0 \\ 0 & I
\end{bmatrix} \begin{bmatrix}
  x \\ e_I
\end{bmatrix} + u^T R u \, dt
\]

Equation 3.24
where the control weighting matrix is generated by trial and error. It is chosen such that the desired pitch and roll angles are small. The optimal control law then becomes:

$$
\mathbf{u} = -[\mathbf{K}_x \mathbf{K}_I] \begin{bmatrix} \mathbf{x} \\ \mathbf{e}_f \end{bmatrix} = -\mathbf{K}_x \mathbf{x} - \mathbf{K}_I \int_0^t (r - \mathbf{C}_y \mathbf{x}) dt
$$

(3.25)

This control law is used together with the vision-based estimation in order to track the moving target.

The two control strategies for hovering and target tracking tasks for the quadrotor were verified by simulations and then implemented on the actual quadrotor UAV for experiments. Both of the LQG methods take the data obtained by the image processing as feedback and generate the desired roll and pitch angles, thrust and desired rate of yaw. The next chapter describes design and implementation of the image processing techniques that were used to estimate position and yaw of the quadrotor relative to the target.
Chapter 4

Vision-based Estimation

4.1 Two Point Estimation

In this section an algorithm for estimation of quadrotor’s position and heading relative to the target is presented. In the experimental setup used during the project the camera is mounted on the ground vehicle and it is facing upwards, capturing images of the quadrotor above it. Mathematically, the problem of estimating pose of the target in the air (the quadrotor) and the camera on the ground is equivalent to estimating pose of the target on the ground with the camera being in the air (on-board of the quadrotor). In one case the camera frame is allowed to rotate while the target frame stays on the level ground; in another the camera frame is fixed to the level plane and target frame is allowed to rotate. Instead of using an on-board vision of the UAV, in this experiment we investigate the usage of on-board camera of the ground target, because camera on the ground is better suited for keeping the air vehicle within its field of view than the other way around. In addition, constraints on the data processing and the weight of the camera are removed by having the camera on the ground target. Formulation for this setup is derived below.

A quadrotor UAV hovering over the camera on the ground robot has a marker attached
Chapter 4. Vision-based Estimation

Figure 4.1: Definition of reference frames.

Figure 4.1: Definition of reference frames.

to it for image processing. The length of the target marker is known. The roll and pitch of
the quadrotor are also known from quadrotor’s on-board sensors. The camera is assumed
to be on flat ground. Figure 4.1 illustrates the problem setup. Frame $F_c$ is the camera
frame located on the ground. $F_q$ is quadrotor’s body frame. $F_O$ is frame $F_c$ translated
to the quadrotor’s center of mass. The relative position of the quadrotor with respect to
the camera is measured from the origin of $F_O$ to the origin of $F_c$. In order to calculate
quadrotor’s position and heading relative to the camera on the ground we define the
following angles and vectors:

$$\alpha = \angle AOC \quad \beta = \angle BOC \quad \delta = \angle AOB$$

$$\vec{a} = \overrightarrow{OA} \quad \vec{b} = \overrightarrow{OB} \quad \vec{c} = \overrightarrow{OC}$$

$$\cos \alpha = \frac{\vec{a} \cdot \vec{c}}{||\vec{a}|| \ ||\vec{c}||} \quad \cos \beta = \frac{\vec{b} \cdot \vec{c}}{||\vec{b}|| \ ||\vec{c}||} \quad \cos \delta = \frac{\vec{a} \cdot \vec{b}}{||\vec{a}|| \ ||\vec{b}||}$$
In order to relate coordinates of the points on the marker in the image frame to the camera frame the following camera model is used:

\[
\begin{align*}
  u &= f_x \frac{x_c}{z_c} + c_x \\
  v &= f_y \frac{y_c}{z_c} + c_y
\end{align*}
\]  

(4.4)

where \(f_{x,y}\) are the focal length in \(x\) and \(y\) directions and \(c_{x,y}\) are the centers of coordinates in the image frame in respective directions. The camera parameters were calibrated and determined using OpenCV software package. A more conventional pinhole camera model was tested as well. The errors in position and yaw estimates were larger using the pinhole model due to the fact that the pixels on inexpensive imagers such as those used in web-cameras have rectangular pixels rather than square which creates different focal lengths in \(x\) and \(y\) directions. The pinhole model does not take this into account.

Vectors \(\bar{a} = [u_1/f_x; v_1/f_y; 1]\) and \(\bar{b} = [u_2/f_x; v_2/f_y; 1]\) are known from the image processing and camera calibration described above. Vector \(\bar{c}\) is not known but can be obtained as follows. Since \(\vec{HO}\) is parallel to \(\vec{z}_q\) (\(z\)-axis of quadrotor’s body frame) by definition \(\vec{OC}\) can be expressed in \(F_q\) as \(\mathcal{F}_q \bar{c} = [0 \ 0 \ 1]^T\). To obtain this vector in frame \(F_o\) we use a rotation matrix:

\[
\mathbf{C}_{oq} = \begin{bmatrix}
  c\psi_r c\theta_r & c\psi_r s\theta_r s\phi_r - s\psi_r c\phi_r & c\psi_r s\theta_r c\phi_r + s\psi_r s\phi_r \\
  s\psi_r c\theta_r & s\psi_r s\theta_r s\phi_r + c\psi_r c\phi_r & s\psi_r s\theta_r c\phi_r - c\psi_r s\phi_r \\
  -s\theta_r & c\theta_r s\phi_r & c\theta_r c\phi_r
\end{bmatrix}
\]  

(4.5)

where \(\phi_r, \theta_r, \psi_r\) are roll, pitch and yaw angles relative to the ground target. Knowledge of rotation of the quadrotor in pitch and roll in quadrotor’s body fixed frame is sufficient for this calculation. Assuming that target is located on flat ground, the roll and pitch angles of the quadrotor in body-fixed frame will be the same in target’s body fixed frame. Using \(\mathbf{C}_{oq}\) we obtain vector \(\bar{c}\) in the camera frame:
\[ F_c \vec{c} = F_o \vec{c} = C_{cq} F_q \vec{c} \] (4.6)

Knowing the angles defined in (4.1) magnitude of \( \overrightarrow{OH} \) can be obtained using the cosine law:

\[ ||\overrightarrow{OH}|| = ||\overrightarrow{OD}|| \cos \alpha = ||\overrightarrow{OE}|| \cos \beta \] (4.7)

\[ L^2 = ||\overrightarrow{OD}||^2 + ||\overrightarrow{OE}||^2 - 2||\overrightarrow{OD}||||\overrightarrow{OE}|| \cos \delta \] (4.8)

Plugging Eq.(4.7) into Eq.(4.8) and rearranging for \( ||\overrightarrow{OH}|| \):

\[ ||\overrightarrow{OH}|| = \cos \alpha \sqrt{\frac{L^2}{1 + \left(\frac{\cos \alpha}{\cos \beta}\right)^2 - 2 \frac{\cos \alpha}{\cos \beta} \cos \delta}} \] (4.9)

Magnitude and direction of \( \overrightarrow{OH} \) in the camera frame are now known. Relative altitude between quadrotor and the ground target is still yet to be determined. Relative altitude can be found as follows:

\[ z_c^D = \frac{||\overrightarrow{OH}||}{||\overrightarrow{a}|| \cos \alpha} \quad z_c^E = \frac{||\overrightarrow{OH}||}{||\overrightarrow{b}|| \cos \beta} \] (4.10)

Once z-coordinates of points E and D are known in the camera frame we can find the relative altitude in \( F_C \) and \( F_O \) as:

\[ z_c = \frac{z_c^D + z_c^E}{2} \] (4.11)

Knowing the relative altitude, the relative displacement of the quadrotor and the ground target in x and y can now be found using relationship in Eq.(4.4):

\[ x_c = \frac{u_1 + u_2 - 2c_x}{2f_x} \quad y_c = \frac{v_1 + v_2 - 2c_y}{2f_y} \] (4.12)
Solution described above assumes that the yaw angle of the quadrotor relative to the ground target is known. The roll and pitch angles can be obtained from quadrotor’s gyroscopes, but it is not possible to obtain relative yaw from quadrotor’s on-board sensors. Dead-reckoning yaw by integrating the rate of yaw over time does not work for a number of reasons. Firstly it was found to drift and rapidly diverge from quadrotor’s actual yaw. Also the quadrotor does not necessarily start tracking the ground target with zero relative yaw so dead-reckoning is not applicable. Yaw is crucial for quadrotor’s position and heading estimation and control and can be estimated as follows. This algorithms uses Gauss-Newton iterative optimization for estimation of yaw.

Initially the yaw is guessed to be:

$$\bar{\psi} = \text{atan}2((v_2 - v_1), (u_2 - u_1))$$

(4.13)

Based on this guess the relative position of the quadrotor with respect to the camera is calculated as described above. Then using this initial estimate of the relative position \([\bar{x}_c \ \bar{y}_c \ \bar{z}_c]\), the coordinates of points D and E in the image plane can be predicted by rearranging the following equation:

$$\begin{bmatrix} \bar{x}_c \\ \bar{y}_c \\ \bar{z}_c \end{bmatrix} + \mathbf{C}_{oq} \begin{bmatrix} \mp L/2 \\ 0 \\ 0 \end{bmatrix} = \begin{bmatrix} (\bar{u}_{1,2} - c_x)/f_x \\ (\bar{v}_{1,2} - c_y)/f_y \\ 1 \end{bmatrix} \bar{z}_{c,D,E}$$

(4.14)

Predicted values \([\bar{u}_{1,2} \ \bar{v}_{1,2}]\) will differ from the actual values obtained from image processing because the initial yaw guess is inaccurate. Gauss-Newton iterative algorithm is utilized to correct the initial yaw estimate. The error function must be defined as a
sum of squares. It will be minimized iteratively and it is defined as:

\[ E(\psi) = \frac{1}{2} \sum_{i=1}^{2} r_i^T r_i \]  

(4.15)

where \( r_{1,2} = [\bar{u}_{1,2} - u_{1,2}, \bar{v}_{1,2} - v_{1,2}]^T \) is the difference between the actual image coordinates of points D and E and their estimate based on the initial yaw guess \( \bar{\psi} \). The first and second derivatives of the error function with respect to the design parameter, relative yaw, is:

\[ \frac{\partial E}{\partial \psi} = \sum_{i=1}^{2} r_i^T \frac{dr_i}{d\psi} \]  

(4.16)

\[ \frac{\partial^2 E}{\partial \psi^2} = \sum_{i=1}^{2} \left( r_i^T \frac{\partial^2 r_i}{\partial \psi^2} + \left( \frac{\partial r_i}{\partial \psi} \right)^T \left( \frac{\partial r_i}{\partial \psi} \right) \right) \]  

(4.17)

Assuming that the term \( \frac{\partial^2 r_i}{\partial \psi^2} \) is negligible we can write a formula for updating estimates of yaw:

\[ \psi = \bar{\psi} - \lambda \left( \sum_{i=1}^{2} \left( \frac{\partial r_i}{\partial \psi} \right)^T \left( \frac{\partial r_i}{\partial \psi} \right) \right)^{-1} \frac{\partial E}{\partial \psi} \]  

(4.18)

where \( \lambda \) is a positive constant \( 0 < \lambda < 1 \) used to tune the update rate. The algorithm terminates when the error in equation (4.15) converges to a small value defined by the user and determined experimentally. A simulation was written to verify performance of the algorithm. It was found to converge in 2-7 iterations depending on the accuracy of the initial yaw guess, which depends on roll and pitch of the quadrotor. In practice the roll and pitch angles are kept small by design which greatly improves accuracy of initial yaw estimate. This allows the algorithm to converge in 2-3 iterations. It is crucial to choose appropriate error bounds for algorithm’s convergence. The error bounds will depend largely on the quality of the image obtained from the camera. If images obtained have low resolution then a large error bound will be appropriate. If it is chosen to be small the algorithm will oscillate around the correct solution but will not converge. Levenberg-Marquardt method can be used for improved convergence. It will slow down convergence
rate but improve robustness by taking smaller steps towards the minimum of the error function reducing chance of overshooting the minimum.

### 4.1.1 Sensitivity Analysis

A sensitivity analysis for the two-point estimation algorithm was performed in order to establish the main sources of error in position, altitude and yaw estimation. The method of parameter variation was used for the sensitivity analysis. There are a number of parameters that can significantly affect the accuracy of the estimation algorithm. From derivation in the previous section it is clear that the error in the length measurement of the object of interest will propagate and cause inaccuracy in estimation. Roll and pitch angles of the quadrotor relative to the target are also important. Accuracy of the camera calibration parameters such as focal length and accuracy of image processing algorithm will also play a role in the final estimation result. In order to determine which parameters effect the final estimate the most the estimation algorithm was simulated with various errors in those parameters.

First an image was generated based on the camera model described above. Then the errors were added to various parameters and the two-point algorithm was run to estimate the relative position and yaw. Results from the estimation were then compared with the ground truth and the error in estimation was plotted against the error in parameters. In the simulation the camera is taking an image of a target that is 0.125m in length.

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Nominal Value</th>
<th>Error Range</th>
</tr>
</thead>
<tbody>
<tr>
<td>Object Length (m)</td>
<td>0.125</td>
<td>0-0.03</td>
</tr>
<tr>
<td>Focal Length $f_x$ (pixels)</td>
<td>800.19</td>
<td>-20-20</td>
</tr>
<tr>
<td>Focal Length $f_y$ (pixels)</td>
<td>804.55</td>
<td>-20-20</td>
</tr>
<tr>
<td>Coordinates $u_{1,2}$ (pixels)</td>
<td>(66.7, 132.5)</td>
<td>-10-10</td>
</tr>
<tr>
<td>Coordinates $v_{1,2}$ (pixels)</td>
<td>(-114.5, -102.5)</td>
<td>-10-10</td>
</tr>
<tr>
<td>Pitch Angle (rad)</td>
<td>-0.035</td>
<td>-0.035-0.035</td>
</tr>
<tr>
<td>Roll Angle (rad)</td>
<td>0.0872</td>
<td>-0.035-0.035</td>
</tr>
</tbody>
</table>

Table 4.1: Parameter variation.
Target’s position relative to the camera is 0.2m in x, -0.1m in y and 1.5m in z. Rotation of the camera with respect to the target is 5 degrees in roll, -2 degrees in pitch and 10 degrees in yaw. These simulation parameters were chosen because they are representative of a case of a quadrotor tracking a ground robot indoor. A number of other tests were performed and data shown here is representative of general trends obtained during the other simulation runs. Table 4.1 outlines nominal values of all parameters and ranges of their variation.

Figures 4.3 and 4.2 show the change in estimation errors with respect to the change in various parameter uncertainties. Table 4.2 shows the sensitivity derivatives obtained
through parameter variation. These are not normalized. Actual values of changes in estimation provide clearer picture of the estimation errors due to the uncertainty in parameter values. The most significant errors in estimation come from bolded terms. The error in length of a target or an object of interest significantly effects altitude estimation. Errors in the focal length of the camera do not significantly effect estimation. Errors in the image coordinates $u_{1,2}$ have significant effect on altitude estimation. This is due to the fact that change in those coordinates effects the length of the target as it appears on an image plane and the length of the target is directly proportional to altitude. Error in the coordinates $v_{1,2}$ does not impact altitude because it does not change the length of
Table 4.2: Change of estimates with respect to changes in parameters.

<table>
<thead>
<tr>
<th>Parameter</th>
<th>X(m)</th>
<th>Y(m)</th>
<th>Z(m)</th>
<th>Yaw(rad)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Object Length (m)</td>
<td>1.16</td>
<td>-0.77</td>
<td>11.98</td>
<td>-0.26</td>
</tr>
<tr>
<td>Focal Length $f_x$ (pixels)</td>
<td>1.25e-5</td>
<td>-9.38e-5</td>
<td>1.82e-3</td>
<td>0</td>
</tr>
<tr>
<td>Focal Length $f_y$ (pixels)</td>
<td>5.29e-5</td>
<td>2.5e-4</td>
<td>3.7e-5</td>
<td>0</td>
</tr>
<tr>
<td>Coordinates $u_{1,2}$ (pixels)</td>
<td>4.73e-3</td>
<td>-4.21e-3</td>
<td>0.05</td>
<td>5.89e-3</td>
</tr>
<tr>
<td>Coordinates $v_{1,2}$ (pixels)</td>
<td>7.13e-3</td>
<td>3.62e-3</td>
<td>6.29e-3</td>
<td>-0.03</td>
</tr>
<tr>
<td>Pitch Angle (rad)</td>
<td>1.50e-4</td>
<td>1.49</td>
<td>0.10</td>
<td>0</td>
</tr>
<tr>
<td>Roll Angle (rad)</td>
<td>-1.46</td>
<td>-5.63e-3</td>
<td>0.40</td>
<td>0</td>
</tr>
</tbody>
</table>

a target in pixels significantly. However, it has a large effect on yaw estimation. Errors in the pitch and roll angles of the camera relative to the target decrease the accuracy of relative position estimate.

Results of the sensitivity analysis suggest that the most important factors for accurate position, altitude and yaw estimation are the length of the target or object of interest and the accuracy of image processing algorithm used to extract and track the two feature points between the frames. Errors in the roll and pitch angles are small in practice since they come from magnetometers on-board of the quadrotor and they do not have any significant effects on estimation.

### 4.2 Image Processing

An image processing algorithm is required for feature identification and extraction as well as for continuously tracking features between the frames. The main criteria for selection of a suitable algorithm include the algorithm’s sensitivity to changing environmental conditions, such as lightning, and it's ability to deal with images of low quality due to the image distortion or movement of the camera. Feature extraction has to be accurate since we aim to use a monocular camera as the only sensor to estimate the relative pose and velocity of the quadrotor and the ground moving target (GMT). Computational complexity of the algorithm is also very important. Low update rates could introduce
delays into the control feedback loop which will degrade the tracking performance. A number of tests were conducted in order to determine lowest acceptable update rate for stable hovering and flight of the quadrotor. It was determined to be approximately 10Hz. If a quadrotor receives updates about it’s position and altitude at lower frequency it will become unstable. Ideally a higher update rate would be preferred. Additionally if computational complexity of the algorithm is low and the update rate is high the frames can be processed partially. Quadrotor’s position in the current frame could be used as an initial guess for UAV’s position in the next frame. This can significantly reduce processing times and increase the update rate even further. This is less useful if the update rate is 10Hz or lower. A quadrotor moves rapidly and its position may change significantly from one frame to the next making it difficult to predict it’s position in one frame based on previous frames. In order to satisfy these design requirements two image processing algorithms were implemented and tested.

4.2.1 Optic Flow Feature Tracking

For preliminary testing of tracking a quadrotor using optic flows the UAV was fitted with a specially designed marker. A chessboard pattern was chosen for the marker since it is easily identifiable on an image and is one of the simplest markers to track. A view from the camera looking at the marker on the quadrotor is shown in Figure 4.4. Initial feature extraction is achieved using a Shi and Tomasi [25] corner detection algorithm. Corners are defined as any trackable feature on an image. They can be extracted from an image by generating an autocorrelation matrix which contains the second derivatives of the image intensity values. If the second derivative at a point on a given image is significantly large in two orthogonal directions it means that this point is unique and it is suitable for tracking. An autocorrelation matrix for a given image can be formulated
as follows [9]:

\[ M(x, y) = \begin{bmatrix}
\sum_{-K \leq i, j \leq K} w_{i,j} \frac{\partial^2 I}{\partial x^2}(x + i, y + j) & \sum_{-K \leq i, j \leq K} w_{i,j} \frac{\partial^2 I}{\partial x \partial y}(x + i, y + j) \\
\sum_{-K \leq i, j \leq K} w_{i,j} \frac{\partial^2 I}{\partial x \partial y}(x + i, y + j) & \sum_{-K \leq i, j \leq K} w_{i,j} \frac{\partial^2 I}{\partial y^2}(x + i, y + j)
\end{bmatrix} \]

(4.19)

\( I \) is the image intensity and \( K \) is a window around any given point \((x, y)\) on an image for which second derivatives are calculated. \( w_{i,j} \) are weights optionally used to provide more weight to derivatives closer to the point of interest when calculating second order derivatives. Corners on an image can be identified by examining two eigenvalues of autocorrelation matrix at any given point. For point \((x, y)\) if the lesser of the two eigenvalues is less than some predefined threshold then that point is a corner. Eigenvalues of the autocorrelation matrix are also used as a signature for identifying and tracking corners from frame to frame. This feature detection algorithm is computationally expensive and it is run only once to initialize the tracking. Tracking could be achieved by continuously running this algorithm on each frame, however, that would be a rather inefficient solution. In order to track the two features identified by this algorithm (indicated on Fig.4.4) between frames an optical flow algorithm is used. As discussed in the previous section,
the estimation algorithm requires two points on the target to be identified. In the case of the chessboard pattern marker a total of 12 corners are identified (the inside corners of the pattern). This is done using the constraint that the corners have to be aligned in a chessboard pattern. All other corners detected on the image are omitted. Corners are numbered from 1 to 12 starting from the top left corner. Corners 5 and 8 are then identified as the two features that should be tracked between the frames.

In order to track the 2 points identified as features of interest a sparse pyramidal Lukas-Kanade (LK) algorithm was implemented [19]. Sparse techniques assume that some information about points being tracked is known. This allows for processing of only some part of the image instead of processing of the entire frame. Since some information about features of interest is known from initial feature extraction a sparse algorithm is suitable. Lukas-Kanade algorithm works based on three assumptions about images and motion of features. Brightness of features is assumed to remain relatively constant through different frames. This means that the image processing rate should be faster than change of lightning conditions so this assumption is satisfied in most cases. Second assumption is that of spatial coherence. Points close together on an image do not move around relative to each other between frames. Since chessboard marker is solid, points on it do not move relative to each other. Lastly in order for any sparse algorithm to work, features in different frames have to be moving slowly as compared to the image update frequency. This is not an issue if image update rates are high or if the features being tracked are moving slowly. This assumption was found to be fairly restrictive. During target tracking tasks the quadrotor can be moving with speeds of up to 0.5m/s. In that case an update rate of 10Hz will not be enough to reliably track the two features from frame to frame. In order to alleviate this problem a pyramidal LK algorithm was used instead. It relaxes the last assumption by first estimating position of the marker coarsely on larger patch of the image and then iteratively refining its initial guess. Therefore, it can efficiently locate the two features on the frame and still provide good accuracy.
Figure 4.4 shows the algorithm identifying and tracking two points of interest. Points identified in green are the two features tracked by the algorithm. Windows of size 24 by 24 pixels drawn around each feature are shown in yellow. They indicate where the algorithm expects to find the two features of interest on the next frame.

4.2.2 Colour-based Feature Tracking

Another algorithm which was implemented and evaluated was based on the color matching feature extraction. This algorithm is robust enough to deal with constant movement of an object of interest without losing the target. This is important since the quadrotor will be constantly moving relative to the ground target. It is also computationally cheap and can reliably provide high update rates. As in the case with the optic flow image processing, the quadrotor was fitted with a special marker consisting of two rectangles of different colour. The distance between the two rectangles is assumed to be known. It represents the length of the target of interest.

The algorithm starts by taking an image of the marker and then converting it from RGB (Red-Grean-Blue) colour space to a binary image. That is done by converting the image into HSV (Hue-Saturation-Value) colour space and thresholding the image twice based on two different colours of the marker. HSV color space is a cylindrical coordinate representation of points of RGB colour space. It is based on hue, saturation and brightness (value) of each point and is more suitable for colour-based image processing. Thresholding then reduces the number of channels in the image from 3 to 1 with each pixel having a value of either 1 or 0. Coordinates of the centroid of both rectangles are then calculated using the moments and areas of the binary pixel distribution. This way of identifying two features on the quadrotor is robust to rapid motion of the marker as well as to the noise from color-based thresholding.
Spatial moments of intensity distributions for binary images are calculated as follows:

\[ M_{i,j} = \sum_{x,y} (I(x,y)x^i y^j) \]  \hspace{1cm} (4.20)

Central moments are used to calculate the total area of the intensity distribution:

\[ \mu_{i,j} = \sum_{x,y} (I(x,y)(x-x_c)^i(y-y_c)^j) \]  \hspace{1cm} (4.21)

where \( x_c \) and \( y_c \) are the coordinates of the center of the distribution:

\[ x_c = \frac{M_{1,0}}{M_{0,0}} \quad y_c = \frac{M_{0,1}}{M_{0,0}} \]  \hspace{1cm} (4.22)

Coordinates of the feature points in the image plane are then calculated:

\[ u = \frac{M_{1,0}}{\mu_{0,0}} - c_x \quad v = \frac{M_{0,1}}{\mu_{0,0}} - c_y \]  \hspace{1cm} (4.23)

This algorithm has been shown to be accurate, computationally inexpensive and robust with respect to the relative motion of the quadrotor and the ground target. However,
the color based features are susceptible to changing lighting conditions and changing background features. This can be solved using adaptive color-based tracking methods such as Mean-shift algorithm, but that would increase computational complexity and decrease the update rate.

4.3 Kalman Filter

Measurements of the relative position and orientation between the quadrotor UAV and the moving target obtained from the image processing and estimation are noisy. Those measurements are added to noisy signals from the GPS (if it is available) in order to estimate target’s position and velocity in the inertial frame. A Kalman filter is designed to deal with noise as well as the prediction of target’s position in case of a temporary target loss or occlusion. It is assumed that the target is moving with piece-wise constant velocity and that target’s acceleration is negligible. In this case acceleration is taken to have a form of Gaussian noise. A discrete Kalman filter is formulated as follows. A state vector for target’s position, orientation and velocity to be estimated is defined as $x = [X^T, \dot{X}^T, \psi^T, \dot{\psi}]^T$. The motion and measurement models for the motion of the ground moving target are defined as follows:

$$x_{k+1} = H_{x,k}x_k + H_{w,k}w_k$$ (4.24)
$$y_k = G_kx_k + n_k$$ (4.25)

where $y_k$ is the measurement of position, altitude and yaw of the quadrotor relative to the GMT obtained from image processing, $w_k$ is acceleration assumed to be the zero-mean Gaussian process noise $v_k \sim \mathcal{N}(0, Q)$, and $n_k$ is measurement noise from the GPS and the image processing $w_k \sim \mathcal{N}(0, R)$. The state transition matrices $H_{x,k}$ and $H_{w,k}$
in the motion model are defined as:

\[
H_{x,k} = \begin{bmatrix} I_4 & \Delta t I_4 \\ 0_4 & I_4 \end{bmatrix}, \quad H_{w,k} = \begin{bmatrix} \frac{\Delta t^2}{2} I_4 \\ \Delta t I_4 \end{bmatrix}, \quad H_k = \begin{bmatrix} I_4 & 0_4 \end{bmatrix}
\]

(4.26)

where \( \Delta t \) is the sampling time interval. Prediction, correction and Kalman gain are formulated as follows:

\[
\hat{x}_{k|k-1} = H_{x,k-1} \hat{x}_{k-1}
\]

(4.27)

\[
P_{k|k-1} = H_{x,k-1} P_{k-1} H_{x,k-1}^T + H_{w,k-1} Q H_{w,k-1}^T
\]

(4.28)

\[
K_k = P_{k|k-1} G_k (G_k P_{k|k-1} G_k^T + R)^{-1}
\]

(4.29)

\[
\hat{x}_{k|k} = \hat{x}_{k|k-1} + s K_k (y_k - G_k \hat{x}_{k|k-1})
\]

(4.30)

\[
P_{k|k} = (I - K_k G_k) P_{k|k-1}
\]

(4.31)

where \( s \) in the state correction step is a scalar prediction function which is set to 1 if the target is in the field of view and zero otherwise. Since the target is assumed to be moving with a piece-wise constant velocity, dead reckoning is used to predict target’s position during an occlusion or temporary target loss.
Chapter 5

Simulations

Prior to implementation of the control laws and the vision-based estimation algorithm on a quadrotor UAV, a set of simulations was performed to verify feasibility of the proposed approach and predict performance of the system. It consisted of two scenarios: 1. A quadrotor hovering over a stationary target or marker based on the 2-point estimation algorithm alone (no GPS data available) and 2. A quadrotor tracking a moving target based on image processing with GPS data available. In the second case when GPS data is available a temporary target loss event was simulated. Results of these simulations are presented in this section.

Figure 5.1: Simulation setup for stationary target simulations
5.1 Stationary Target

In the simulation, images are generated from true values of the relative positions between the quadrotor and the target. The 2-point estimation algorithm is applied to obtain the relative position and yaw angle. Quadrotor’s parameters are selected to represent the actual hardware setup available in the lab and used for the experiments later: \( m = 0.8kg \), \( I_{x,y} = 0.00676kg.m^2 \) and \( I_z = 0.0158kg.m^2 \). Delay due to actuation was chosen to be 0.2s and was obtained through experiments. The noise added to the results obtained from the image processing and the 2-point optimization algorithm was a Gaussian noise with zero mean and 0.1m standard deviation for the x and y directions,
0.15m for z direction and 0.05rad for yaw. This was selected as worst possible scenario for the estimation based on the experiments with the image processing techniques discussed in previous chapters. Camera parameters are as follows: $f_x = 800.2$, $f_y = 804.5$, $c_x = 306.6$, $c_y = 217.3$. Camera images were generated from the ground truth data at 640 by 480 pixels resolution. The pitch and roll angle responses obtained in simulation using

![Graphs showing errors in X, Y, Z positions and Yaw angle estimation.](image)

Figure 5.3: Errors in relative position and yaw angle estimation using the linear observer the PID controller are shown in Figure 5.2. The PID controller performs adequately if the desired roll and pitch angles are kept small. This condition is not restrictive because the attitude angles are kept small by design of the position control. This is done primarily to improve estimation. As well, the linearized model of the quadrotor based on which the position and attitude controls were design are only valid for small pitch and roll angles. Restricting input to the quadrotor limits its acceleration but not its velocity and, as will be shown later, the quadrotor is capable of tracking stationary and moving targets.
Graphs also show the presence of a delay due to actuation, image processing and data transfer between the quadrotor and the ground station.

Figure 5.3 shows results of the relative position and yaw angle estimation obtained by the observer based on the linear Kalman filter. The observer provides an accurate estimation of the relative position and yaw angle given the noise in raw image data. Errors in the x and y directions have standard deviations of 0.01m and 0.02m respectively and a mean of 0m. Errors for the altitude and yaw estimates have standard deviations of 0.01m and 0.005rad respectively with 0 mean. These errors are acceptable considering the fact that the noisy measurements provided by the camera imaging are only accurate to within tens of centimetres.

The linear Kalman filter also estimates the relative velocity of the quadrotor with respect to the target. Estimation results are presented in Figure 5.4. The velocity estimation is not as good as the relative position estimation due to the fact that there
are no measurements of the relative velocity and it is estimated by fusing position and acceleration measurements together using the linear system model.

The estimated relative position, velocity and acceleration from the observer are used as state feedback for the position controller. In the simulation, the control goal is to keep the relative position and orientation of 0.2 m in x direction, 0 m in y direction, 2 m in z direction and 0 rad in yaw. The quadrotor begins hover at 5s and it is located at \((x, y, z)_q = (0.5, 1, 2.5)m\) with yaw of 0.1rad in the beginning of the simulation. Target was located at coordinates \((x, y, z)_t = (0, 0, 0)m\) and yaw of 0rad. Simulation results of the position control are shown in Figure 5.5. As can be seen, the quadrotor converges to the desired position relative to the target of interest. Rise times were 2.4s for x and
y positions, 3.7s for altitude and 1.1s for yaw. Standard deviations of errors at steady state were 0.046cm and 0.054cm in x and y directions respectively, 0.075cm in altitude and 0.0031rad in yaw.

5.2 Moving Target

Figure 5.6: Simulation setup for moving target tracking simulations

Figure 5.6 shows the system overview for the moving target simulation. In this case a Kalman filter is added for two purposes. Firstly it estimates the position and velocity of the target based on quadrotor’s GPS position and the relative position of the target relative to the quadrotor obtained from the image processing. Secondly the Kalman filter provides predicted position of the target in a case of target occlusion or temporary target loss. The simulation parameters were again chosen to match those of the quadrotor, the ground robot and the camera used during experiments and they are the same as for the stationary target experiments. Ground robot’s speed is varied from 5 to 20cm per second which reflects the maximum capabilities of the ground robot in the lab. During the simulation the target starts moving with a constant velocity and a constant rate of yaw. At the time \( t = 25\sec \) the target changes its velocity in both x and y directions. At the
time \( t = 37\, \text{sec} \) the target changes its rate of yaw. Target occlusion is simulated during times \( t = 15 - 20\, \text{sec} \) and \( t = 35 - 40\, \text{sec} \). During those time periods no images from the camera are generated and the estimator relies on target’s velocity estimates for prediction of target’s position. Figure 5.7 shows performance of the PID control for attitude stabilization. The control is able to track high frequency input generated by the LQG control for position accurately. The roll and pitch angles of the quadrotor do not exceed 2 degrees. This is done by design by choosing low gains for LQG. Figures 5.9 and 5.8 demonstrate Kalman filter’s ability to estimate target’s velocity and position respectively. Standard deviations of errors for velocity and position estimates are shown in Table 5.1. Due to errors in velocity estimation it can be seen that position estimate starts to drift.
immediately after the target gets out of the field of view of the camera. However, since error in velocity estimation is small, the drift is slow and the quadrotor continues tracking the target successfully. This method of dealing with the target occlusion or target loss events works well for relatively short periods of time. If the target is not detected by the camera for extended periods of time then the estimation will diverge from target’s actual position and the quadrotor will lose the target permanently. The length of the time window during which the quadrotor is capable of reacquiring the target is a function of target’s velocity and trajectory, size of the field of view of the camera and quadrotor’s altitude. Figure 5.10 shows errors in the relative position and yaw during the task of tracking the ground target. As shown the quadrotor converges to the desired relative
Figure 5.9: Velocity estimates from Kalman filter for piece-wise constant velocity target.

As demonstrated by simulations above, the proposed two-point estimation algorithm and control and estimation structure work well for cases of hover over stationary target as well as target tracking tasks. Next control and estimation were implemented on a set of hardware to experimentally verify proposed system and evaluate it’s performance.
Chapter 5. Simulations

Table 5.1: Standard deviation of errors for Kalman filter estimates.

<table>
<thead>
<tr>
<th></th>
<th>Std. Dev. Position</th>
<th>Std. Dev. Velocity</th>
</tr>
</thead>
<tbody>
<tr>
<td>X</td>
<td>0.003m</td>
<td>0.005m/s</td>
</tr>
<tr>
<td>Y</td>
<td>0.003m</td>
<td>0.005m/s</td>
</tr>
<tr>
<td>Z</td>
<td>0.005m</td>
<td>0.0006m/s</td>
</tr>
<tr>
<td>Yaw</td>
<td>0.01rad</td>
<td>0.017rad/s</td>
</tr>
</tbody>
</table>

Figure 5.10: Position and orientation tracking errors.
Chapter 6

Experiments

The proposed vision-based estimation and control strategies were implemented on an actual quadrotor UAV and a ground robot. Experiments were performed for hovering over a stationary target based on vision-sensing alone and for tracking of a moving target in a GPS-enabled environment with a temporary target occlusion. This chapter describes the experimental setup and demonstrates results obtained during the experiments.

6.1 Experimental Setup

Figure 6.1: Quadrotor UAV and ground robot used in the experiments
Chapter 6. Experiments

The quadrotor used during the experiments is a Quanser UAV which has 3 gyroscopes, 3 magnetometers and 3 accelerometers on-board. These sensors accurately provide linear accelerations, roll and pitch angles and rates of change of roll, pitch and yaw. The quadrotor is fitted with a Gumstix™ computer on-board running a Linux based operating system at 400MHz. All control code is generated as a Simulink™ diagram and then compiled into C code for embedded systems. Compiled control code is run on Gumstix™ on-board of the quadrotor. The quadrotor has a wireless card on-board which is used to receive position data from an off-board laptop computer. Figure 6.3 shows an outline of the implementation structure.

A Logitech™ web-camera is used for obtaining images for image processing. It runs at a frame rate of 25 frames per second. The resolution of images used during experiments was 640 by 480. Camera’s field of view is 65 degrees which is suitable for tracking tasks.

Optitrack™ motion capture system provides sub-millimetre accuracy position information with an update rate of up to 100Hz. Optitrack is connected to a ground station.
computer which records all position and orientation data and relays it to the quadrotor UAV. Optitrack™ system served two purposes during the experiments. It was used as a simulated GPS signal which provided the quadrotor with its position and orientation in the inertial frame. The ground truth measurements including position and orientation of the quadrotor as well as the ground target were also collected using Optitrack.

Roomba Create™ robot was used as a ground moving target. Roomba robot is fitted with a Netbook running Ubuntu operating system that uses ROS (Robot Operating System) software framework to interface with Roomba’s hardware. Using ROS, velocity and rate of yaw of the robot can be specified directly as inputs and robot’s dynamics can then be described using the following unicycle model:

\[
\begin{align*}
\dot{x}_t &= v_t \cos \psi_t \\
\dot{y}_t &= v_t \sin \psi_t \\
v_t &= u_1 \\
\dot{\psi}_t &= u_2
\end{align*}
\]

The ground robot is capable of moving with a speed of 0.4m/s and change direction of motion at 1.5rad/sec.
6.2 Way-point Following

Proposed control scheme was first implemented for a simple way-point following task with Optitrack system used for position and orientation feedback. Initially no image processing was done and the quadrotor was receiving accurate pose information with minimal delay. This was done as a benchmark to evaluate quadrotor’s performance with minimal errors in estimation. Figure 6.4 shows results for a waypoint following task for a quadrotor UAV. Points where the error increases instantaneously represent a new waypoint command sent to the quadrotor. Waypoints used in the experiment illustrated in Figure 6.4 were: \((0, 0, 1)_{t=0}, (0.5, 0, 1)_{t=60}, (-0.1, 1, 1)_{t=70}, (0.1, 0.4, 1)_{t=95}\). As can be seen the pitch and roll inputs remain between \(\pm 8\text{deg}\) with standard deviation of \(2.3\text{deg}\). Standard deviation of error is \(0.047\text{m}\) for \(x\) and \(y\) position, \(0.03\text{m}\) in altitude and \(2.5\) degrees in yaw.

Figure 6.4: LQG waypoint following
6.3 Stationary Target

For experiments with a quadrotor UAV hovering over a stationary target the two image processing algorithms described earlier were tested. Results for hovering based on the optic flows and the colour-based image processing are presented. Both sets of experiments are done in a GPS-denied environment. The quadrotor is relying solely on data obtained through image processing for feedback.

6.3.1 Optic Flow-based Hovering Results

A large set of experiments were done to appropriately measure performance of the system. Figure 6.5 shows a set of results from one of the flight tests which are representative of the overall system performance. Results demonstrate that the quadrotor is capable of maintaining a constant relative distance to the ground camera. Hovering performance of the quadrotor became worse in comparison to hovering with the ground truth data (Optitrack) used for feedback. Standard deviation of errors for x and y position increased from 0.047m to 0.069m. Similarly it increased for altitude from 0.03m to 0.04m. Standard deviation of error for yaw remained the same at 2.5 degrees. The difference in performance comes from the introduction of another delay due to the image processing and associated errors in estimation. Extracting features from an image and running the estimation algorithm creates a delay which is not present when Optitrack data is used for feedback directly. In the case of optic flow-based estimation two images must be taken, compared to each other and then the yaw and position optimization algorithm needs to iterate to a solution. Optitrack cameras use infrared light and specially designed reflective markers for image processing. Each trackable object is required to have at least three reflective markers attached to it during tracking. This enables Optitrack to determine position and orientation of the quadrotor accurately, at high update rates and without any iteration. Data obtained from the estimation algorithm is also
noisier and less accurate than the Optitrack data as discussed in the previous section. The overall increase in position and heading error does not destabilize the system and it is capable of stable flight and stand-off tracking.

Figure 6.5: Quadrotor relative position and heading estimation for optic flow-based hovering.

Although optic flow-based image processing worked in general, a few difficulties were encountered when using this algorithm. The main drawback of the optic flows is its inability to deal with blurry or distorted images. When the quadrotor moves rapidly the marker attached to it ends up being blurred and the optic flow algorithm fails to locate the two features of interest. When this happens frames are dropped and the pyramidal
Lukas-Kanade algorithm has to be run repeatedly on entire frames until features are captured again. While this does not happen often during hovering tasks it becomes problematic for the task of following a moving target. Frame rate achieved using optic flows was 17Hz on average over 10 test flights. This is well above the minimum 10Hz update rate required for stable hover.

6.3.2 Colour-based Hovering Results

Another set of experiments was conducted in order to compare the colour-based image processing with the optic flows. As in the previous experiment the quadrotor is required to hover directly over a target. In this case the quadrotor performs a stand-off tracking task which requires it to maintain constant position, altitude and orientation relative to the target. Estimated relative position and yaw angle and their errors are shown in Figure 6.6. As in the case with the optic flow-based image processing the quadrotor was able to hover stably over the target. Standard deviation of errors in x and y directions was 0.07m and 0.035m in altitude; standard deviation of yaw was 2.3 degrees. These errors provide a slight improvement over hovering using the optic flows. Average frame rate achieved with the colour-based image processing was 23Hz during 10 test flights performed. It is a significant improvement over the optic flows and it reflects computational simplicity of the algorithm.

Based on the results of experiments described above the color-based image processing was selected for further experimentation. The color-based feature detection is less computationally expensive and can, therefore, be done at faster frame rates. It is also more reliable because it is able to provide estimates of the relative pose of the quadrotor even when images are blurry or distorted. Optic flow algorithm loses two feature points if image is of low quality.

For experiments with the color-based image processing a Kalman filter was implemented to improve estimation. The estimation errors of the relative position and yaw
Table 6.1: Standard deviation of estimation errors for colour-based hovering.

<table>
<thead>
<tr>
<th></th>
<th>Std. Vision Alone</th>
<th>Std. Kalman Filter</th>
</tr>
</thead>
<tbody>
<tr>
<td>X(m)</td>
<td>0.0110</td>
<td>0.0057</td>
</tr>
<tr>
<td>Y(m)</td>
<td>0.0130</td>
<td>0.0051</td>
</tr>
<tr>
<td>Z(m)</td>
<td>0.0056</td>
<td>0.0031</td>
</tr>
<tr>
<td>Yaw(rad)</td>
<td>0.0056</td>
<td>0.0036</td>
</tr>
</tbody>
</table>

angle are shown in Figure 6.6. The Kalman filter used to remove noise from raw image data improves estimation. Standard deviation of estimation errors decreased as shown in Table 6.1.

\section{6.4 Moving Target}

In order to test the ground target tracking performance and the ability to deal with a temporary target loss a number of experiments were done. In the moving target experiments the quadrotor is flying in a GPS-enabled environment. It is aware of it’s location and yaw in the inertial frame defined by Optitrack. Based on the proposed control and estimation structure the quadrotor is capable of tracking a ground target in a GPS-denied environment as well. However, in that case it is unable to estimate target’s velocity; only velocity of the target relative to the quadrotor can be estimated. This makes it impossible to predict target’s position in the case of occlusion or target loss. When global positioning data is available the quadrotor is capable of dealing with target loss and continue tracking the target.

During the moving target experiments the ground robot is pre-programmed to follow a path independently of quadrotor’s position and velocity. A number of different ground target trajectories were tested during the experiments. It was found that the quadrotor performs the tracking task with similar accuracy in all cases independent of the trajectory. Results for linear and arced trajectories are presented here.
6.4.1 Attitude Control

Figure 6.7 shows performance of the PID control for pitch and roll of the quadrotor. Reference signal is generated by the LQG control for x and y position. There is a delay of 0.2 seconds in response of the PID. It comes from delays due to image processing and actuator dynamics and is taken into account in quadrotor’s model for the LQG design.

6.4.2 Linear Path Experiments

During the linear path experiments ground robot was programmed to move in a straight line with velocity of 10cm/sec for 20 seconds, rotate to change yaw angle with angular rate of 0.07 rad/sec for 10 seconds and move backwards in another straight line with velocity of 5cm/sec for 15 seconds. These commands are given as input to the Roomba robot. The quadrotor was required to perform a stand-off tracking task keeping relative position to ground target at 0m in x and y direction, 1m altitude and 0 rad yaw. Figure 6.8 demonstrates Kalman filter’s performance for these experiments. On Figure 6.8 position of the ground moving target was predicted using dead reckoning during two intervals which are illustrated using prediction data function, discusses at the end of Chapter 4. When prediction function is 0, dead reckoning was used. As can be seen it works well for a target with piece-wise constant velocity. As predicted by numerical simulation, the errors in velocity estimates are small and drift of position estimates during occlusion is slow. This allows the quadrotor to continue tracking the target even when the image data is unavailable.

Figure 6.9 shows quadrotor’s altitude, position and yaw relative to the moving target. Largest errors in tracking occur when target stops and starts changing orientation with constant rate of yaw. The magnitude of error during that period depends on the speed of rotation of the ground robot. This is due to the fact that the quadrotor has limits set on maximum pitch and roll angles as well as on its rate of yaw. This limits quadrotor’s ability to react to fast-moving targets. If rate of yaw of GMT is decreased the tracking
error also decreases significantly. As shown on figure 6.9 (d), errors stay within 10cm for majority of the tracking task.

6.4.3 Circular Path Experiments

For the circular path experiments the ground target was moving in an arc, with position and yaw changing at the same time all throughout the path. Forward velocity of the ground robot was 7cm/sec for 25 seconds and rate of yaw was 0.03rad/sec for 25 seconds. After that the robot stopped for 5 seconds and moved backwards with same velocity and rate of yaw for another 10 seconds. These trajectories were mostly limited by the space available for experimentation. As can be seen from Figures 6.10 and 6.11 the Kalman filter is able to estimate target’s position and yaw, even during a tracking loss events. The quadrotor successfully tracks the target throughout the course with errors within 10cm. The largest error occurs at time 30s and corresponds to the target stopping abruptly and the quadrotor overshooting the target. The quadrotor still remains in the field of view of the target for the entire duration of the tracking task. Table 6.2 shows the standard deviations of errors for both linear and circular path experiments. The errors during experiments with a moving target are comparable with errors in tracking during a stand-off hover using vision-based approach.

<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>X(m)</td>
<td>0.093</td>
<td>0.086</td>
<td>0.072</td>
</tr>
<tr>
<td>Y(m)</td>
<td>0.096</td>
<td>0.079</td>
<td>0.071</td>
</tr>
<tr>
<td>Z(m)</td>
<td>0.043</td>
<td>0.037</td>
<td>0.035</td>
</tr>
<tr>
<td>Yaw(rad)</td>
<td>0.044</td>
<td>0.036</td>
<td>0.040</td>
</tr>
</tbody>
</table>

Table 6.2: Standard deviation of tracking errors for Linear and Circular path experiments.
Figure 6.6: Experiment results of relative position and yaw angle estimation for color-based hovering.
Chapter 6. Experiments

(a) Pitch PID control.

(b) Roll PID control.

Figure 6.7: PID control for pitch and roll angles during target tracking task.

Figure 6.8: Raw camera data compared to Kalman filter estimates for Linear Path experiments.
Figure 6.9: Quadrotor and GMT positions for Linear Path experiments.
Figure 6.10: Raw camera data compared to Kalman filter estimates for Circular Path experiments.
Figure 6.11: Quadrotor and GMT positions for Circular Path experiments.
Chapter 7

Conclusions

In this thesis design and implementation of a controller for a quadrotor UAV were investigated. The primary focus was on a task of a vision-based target tracking. The goal of the project was to complete implementation of a system capable of tracking a moving target based on a limited information about the target and vision-sensing alone. Proportional-Integral-Derivative control was used for stabilization of attitude and altitude of a quadrotor UAV. A Linear Quadratic Gaussian control law was developed for the way-point following and position control of the quadrotor. It was augmented with an integral control specifically for a task of tracking a moving target. Control laws were integrated with a novel position and yaw estimation algorithm which relies on vision-sensing alone. Two image processing algorithms were compared for position and yaw estimation. Optic flow-based processing provides highly accurate estimates but is heavily reliant on distinct features and good image quality which are not available in a common target tracking scenario. Optic flow algorithm was also computationally expensive and provided estimation at low update rates which effects control results. Colour-based estimation proved to be more reliable and less sensitive to image quality. It also enables high update rates of estimation reducing delays in the feedback. Although it does not provide the sub-pixel accuracy of optic flow processing it is still accurate enough for reli-
able performance. Simulations were performed to verify feasibility of proposed estimation and control structure. Control and estimation algorithms proposed were implemented on a quadrotor UAV and a ground robot and performance of the system was assessed through a number of experiments. The quadrotor was capable of tracking the target in both GPS-denied and GPS-enabled environments based on vision sensing alone. In case where the quadrotor was operating with GPS it was capable of dealing with target occlusion and temporary target loss cases successfully.

Possible extensions for this work would be investigation of more sophisticated image processing and feature extraction algorithms. The experiments presented in this work deal primarily with indoor scenarios and do not deal with the limitations of the camera used on-board of the quadrotor. The question of higher altitudes and field of view limitations can also be investigated further. As mentioned in the Introduction, the quadrotor and the ground vehicle can be tethered in order to complete more complicated missions and provide data collection for various tasks. Implementation of control laws which deal with model uncertainty and disturbance rejection will also aid in moving the proposed system from the lab to the field.
Bibliography


Colloquium on Signal Processing and its Applications, Penang, Malaysia, March 2011.


[23] I. Sadeghzadeh, A. Mehta, Y. Zhang, and C. Rabbath. Fault-tolerant trajectory tracking control of a quadrotor helicopter using gain scheduled PID and model ref-


