2013-04-29

Modeling, Parameter Estimation, and Navigation of Indoor Quadrotor Robots

Stephen C. Quebe
Brigham Young University - Provo

Modeling, Parameter Estimation, and Navigation of Indoor Quadrotor Robots

Stephen C. Quebe

A thesis submitted to the faculty of
Brigham Young University
in partial fulfillment of the requirements for the degree of

Master of Science

Randal W. Beard, Chair
D. J. Lee
Robert P. Burton

Department of Electrical and Computer Engineering

Brigham Young University

April 2013

ABSTRACT

Modeling, Parameter Estimation, and Navigation of Indoor Quadrotor Robots

Stephen C. Quebe
Department of Electrical and Computer Engineering
Master of Science

This thesis discusses topics relevant to indoor unmanned quadrotor navigation and control. These topics include: quadrotor modeling, sensor modeling, quadrotor parameter estimation, sensor calibration, quadrotor state estimation using onboard sensors, and cooperative GPS navigation.

Modeling the quadrotor, sensor modeling, and parameter estimation are essential components for quadrotor navigation and control. This thesis investigates prior work and organizes a wide variety of models and calibration methods that enable indoor unmanned quadrotor flight. Quadrotor parameter estimation using a particle filter is a contribution that extends current research in the area. This contribution is novel in that it applies the particle filter specifically to quadrotor parameter estimation as opposed to quadrotor state estimation. The advantages and disadvantages of such an approach are explained.

Quadrotor state estimation using onboard sensors and without the aid of GPS is also discussed, as well as quadrotor pose estimation using the Extended Kalman Filter with an inertial measurement unit and simulated 3D camera updates. This is done using two measurement updates: one from the inertial measurement unit and one from the simulated 3D camera.

Finally, we demonstrate that when GPS lock cannot be obtained by an unmanned vehicle individually. A group of cooperative robots with pose estimates to one anther can exploit partial GPS information to improve global position estimates for individuals in the group. This method is advantageous for robots that need to navigate in environments where signals from GPS satellites are partially obscured or jammed.

Keywords: quadrotor, gps denied navigation, parameter estimation

ACKNOWLEDGMENTS

Special thanks to Dr. Randy Beard, Dr. Robert Burton, Dr. DJ Lee, John MacDonald, Dr. Clark Taylor, and Dr. Rajnikant Sharma. A very special thanks to Darcee.

Table of Contents

List of Figures

Chapter 1

Introduction

This thesis reports on a number of topics relevant to navigation and control of small indoor unmanned aerial vehicles. Each topic discussed is a building block for achieving autonomous navigation and control of an unmanned quadrotor aircraft. This thesis discusses: modeling quadrotor aircraft, modeling onboard sensors, estimating quadrotor model parameters, estimating sensor model parameters, and cooperative GPS navigation.

An unmanned aerial vehicle (UAV) needs to be able to control its position and orientation in relation to a map or objects of interest to accomplish most meaningful tasks. For example, in order for a small aircraft to travel in an indoor environment, it must first be able to control it's position in relation to the obstacles in the indoor environment. It is likely that the robot or vehicle will have some task relating to a target or object of interest. For the vehicle to interact with an object of interest, it must know and be able to estimate its position in relation to the object and the environment. This process is called navigation. Fundamental to control of a dynamic system is system ID and state estimation. For each of the following topics relating to the navigation and control of a small quadrotor in an unknown environment, this introduction discusses the motivation for each topic, an overview, prior work, and contributions to existing literature.

1.1 Modeling the Quadrotor

The most fundamental step to navigation and state estimation of a quadrotor aircraft is designing a model that represents the quadrotor mathematically. This is necessary for effective estimation and control. There has been a great deal of research in the area of mathematically modeling a quadrotor aircraft.

The fundamental kinematics and dynamics of the quadrotor are discussed in a number of papers [1, 2, 3]. Some authors include a complete explanation of reference frames in addition to the construction of the simple kinematics and dynamics [4, 5, 6]. Researchers at Stanford University and other authors model secondary effects such as rotor dynamics, air foil design, blade flap, and air flow disruption [7, 8, 9, 10]. This thesis outlines a complete view of the coordinate frames, kinematics, and dynamics necessary for basic simulation and estimation of a small quadrotor aircraft.

1.2 Modeling the Sensors

For a quadrotor to estimate its pose and move in an unknown environment, it must be able to obtain information about its external environment. For this discussion, it is assumed that external systems are not available to provide the UAV with pose or other information and that all information must be collected by onboard sensors. In order to interpret and use these sensors for navigation and control, the UAV must model the sensors mathematically. There are many useful onboard sensors. Some of the more common and inexpensive sensors include MEMS accelerometers, MEMS gyroscopes and 3D cameras such as the Microsoft Kinect.

How these sensors are modeled and used with respect to quadrotor estimation and control has been discussed by many authors. Much research has been conducted on the modeling and interpretation of MEMS accelerometers and gyroscopes (when combined, also called an inertial measurement unit or IMU) [11, 12, 6, 13]. IMU models and how they relate specifically to quadrotor aircraft are discussed by a number of authors[7, 14]. Members of the BYU MAGICC lab have looked into improved accelerometer models that aid in quadrotor state estimation [2, 15, 16]. Trucco and Verri give a very detailed explanation of modeling a 2D or 3D camera [17]. The Microsoft Kinect is investigated in great detail by Smisek [18]. This thesis provides a detailed explanation of a MEMS IMU model and a simplified 3D camera model as it relates specifically to state estimation and control of the quadrotor aircraft. These sensors are essential for indoor navigation because other commonly used navigation systems such as GPS are largely inaccessible inside buildings and other structures.

Also, we note some practical subtleties and important strengths and weaknesses of these sensors when being used onboard a micro UAV.

1.3 Estimating the Model and Sensor Parameters

In order to use these mathematical models of the quadrotor and onboard sensors accurately, there are a number of platform and sensor specific parameters that must be estimated. This is called parameter estimation or sometimes system ID when discussing the quadrotor model. Estimation of parameters for sensors is often called sensor calibration.

There are many ways to estimate parameters for the quadrotor model. A number of authors suggest methods for calibrating or estimating MEMS IMU parameters [2, 6, 19]. Cross axis sensor calibration for the MEMS IMU is discussed in great detail by two application papers [20, 21]. Camera calibration and parameter estimation are dealt with in great depth by [17]. For specific research regarding Microsoft Kinect calibration see [18]. In regard to quadrotor parameter estimation, some authors measure model parameters for use in estimation and quadrotor simulation [10]. Research has also shown that it is possible to estimate some sensor parameters, specifically gyroscope bias, onboard during flight [11, 16, 22].

This thesis contributes an explanation of the math behind and how to implement cross axis MEMS IMU calibration. We also discuss a novel method for quadrotor parameter estimation using a particle filter. The particle filter is a Monte Carlo estimator and is described in [23, 24]. Given truth data from a test flight, this thesis shows that it is possible to estimate quadrotor model parameters using the particle filter. This contribution enables parameter estimation without tedious measurement techniques by using known state information gathered from an accurate motion capture system. The theory behind this method, as well as how to practically implement it, will also be explained. This thesis also gives experimental results and some discussion of the methods advantages and disadvantages.

1.4 Aircraft State Estimation

The quadrotor is an inherently unstable platform and requires feedback for stable control [8]. Many methods of feedback control rely on an estimate of the quadrotor's pose

in 3D space [1, 2, 7, 25, 9]. In the scenario where the quadrotor is operating in an unknown environment, without the aid of external sensors, it must estimate its pose using onboard sensors. MEMS IMU and 3D cameras are often used for this purpose.

For outdoor navigation, many researchers rely on pose estimation using an IMU with periodic corrections from GPS, incorporated using an Extended Kalman Filter or a particle filter [24, 26, 12]. Veth also discusses IMU sensors in tight integration with GPS [19]. Because GPS is unreliable indoors, many groups place known visual landmarks and use vision systems to aid in robot state estimation and navigation [27, 28]. These systems are not useful in unknown environments because they require the prior placement of artificial landmarks. A number of authors show that it is possible to obtain consistent large scale vision measurements using 3D visual odometry, but these groups use stable platforms that do not require vision updates for stability control [29, 30]. Research conducted at Brigham Young University looks specifically at quadrotor aircraft pose estimation [16, 15, 31]. These papers show improved methods for estimating a quadrotor's state using an inertial measurement unit. Huang et al. directly discusses quadrotor pose estimation and control using 3D vision and IMU in an unknown environment [32].

In this thesis a similar approach to [32] is discussed using the Microsoft Kinect. The Kinect is used to provide periodic estimates of the quadrotors 3D pose. Fundamental principles for pose estimation from a set of matching 3D features is described by [33, 34, 35, 36, 37]. The method described here uses these Kinect measurements to correct IMU drift and for quadrotor feedback and control using an Extended Kalman Filter. Navigation in this thesis is derived frame based navigation as discussed in [38, 39], but with respect to a single local object of interest instead of navigating relative to a global map. This enables the quadrotor to prevent failure due to both global map inconsistencies and slow vision estimates in large environments.

1.5 Cooperative GPS Navigation

Robots and unmanned aircraft sometimes work in situations where partial GPS information is available. This can occur when signals from GPS satellites are partially obscured. This is common because of the low signal power of GPS transmissions[40]. Access to GPS

information can also be obscured from active jamming and unintentional signal interference [41]. When an unmanned vehicle does not have access to enough GPS information to estimate its position the information is often discarded. However, we show in this thesis that with multiple systems interacting with one another, it is possible and often useful to share information with one another so the group can collectively navigate. In a great deal of previous work, it has been shown that cooperative navigation is possible and can be very effective in GPS denied areas [42, 43, 44, 45, 46]. We show that when robots or unmanned vehicles can communicate and have relative pose estimates relating their poses to one another, partial GPS information available can be exploited to improve global position estimates of unmanned vehicles that operative cooperatively.

Chapter 2

Quadrotor Models

2.1 Overview

A quadrotor aircraft can be mathematically represented by a collection of differential equations, state variables, input variables, and constant model parameters. The state variables represent the quadrotor's position, orientation, and velocities in space. The input variables represent things such as forces and moments generated by the aircraft and the environment. The model parameters are values that include the quadrotor's mass, moments of inertia, and other constant quantities that influence how the quadrotor state changes. The differential equations, current state, and inputs are used in conjunction with gravity and other aerodynamic forces to mathematically model how the quadrotor state changes through time. This state space model is valuable for navigation, stabilization of the aircraft. In this chapter, frames of reference, transformations between frames, and two different state space models for the quadrotor are discussed.

2.2 Model State Variables

In the quadrotor state space model used in the thesis, there are twelve states. These are the same states desribed in [4, 1]. They represent the quadrotor's position and orientation as well as the quadrotor's linear and angular velocities. The first three values x, y, and z are the position of the quadrotor's center of mass in the world frame. The second set of three values u, v, and w are the instantaneous linear velocities of the quadrotor measured in the body frame. The next three values ϕ, θ, and ψ are the Euler angles (roll, pitch, and yaw) that define the quadrotor's orientation. These are not all defined in the same reference frame; they are applied one after another and are defined in different frames. Yaw is defined as a rotation about the vehicle frame's z axis. Pitch is defined as the rotation about the

yaw frame's y axis. Roll is defined as the rotation about the pitch frames' x axis. So yaw is applied first, then pitch is applied after yawing, then roll is applied after pitching. The last set of values p, q, and r are the instantaneous angular velocities measured in the body frame. The complete state can be written

$$\mathbf{x} = \begin{bmatrix} x & y & z & u & v & w & \phi & \theta & \psi & p & q & r \end{bmatrix}^\intercal \tag{2.1}$$

where each element in the array is a state variable.

2.3 Frames of Reference

In order to fully discuss the quadrotor state space models, it is important to understand different frames of reference used to describe and relate quantities affecting and measuring the aircraft's state. For further reading on reference frames see [4, 6]. There are a number of important frames of reference used for this model. These include the inertial or world frame, the vehicle frame, the Euler frames, and the body frame.

2.3.1 The World Frame

The world reference frame $\mathcal{F}_w$ is a non-accelerating reference frame defined at an arbitrary position in the quadrotor's environment. It is oriented with the x axis pointing forward, the y axis pointing to the right, and the z axis point down, aligned with gravity.

2.3.2 The Vehicle Frame

The vehicle frame $\mathcal{F}_v$ is positioned at the center of mass of the quadrotor with the axes aligned with the world frame axes. Translating between the world and vehicle frame is considered in the next section. The vehicle frame is used as an intermediate frame of reference for transformations between the world frame and the body frame.

2.3.3 The Body Frame

The body frame $\mathcal{F}_b$ is positioned at the center of mass of the quadrotor. The x axis points forward out the front of the quadrotor. The y axis points out the right of the

quadrotor. The z axis points down through the bottom of the quadrotor. The quadrotor's sensors measure a number of quantities in the body frame.

2.3.4 The Euler Frames

The Euler frames are intermediate frames that move from the vehicle frame to the body frame. All of the Euler frames are positioned at the center of mass of the quadrotor. The yaw frame $\mathcal{F}_{ey}$ is the same as the vehicle frame. The pitch frame $\mathcal{F}_{ep}$ is the same as the yaw frame except it is rotated about the yaw frame's z axis by ϕ. The roll frame $\mathcal{F}_{er}$ is the same as the Euler pitch frame except it is rotated about the Euler pitch frame's y axis by θ.

2.4 Transformations

Different quantities relating to the quadrotor are measured and modeled in different reference frames. To model the quadrotor, it is essential to be able to translate values between these reference frames. One way to do this is by posing these transformations as matrix and vector operations.

2.4.1 World To Vehicle Frame

The following transformation moves a vector from the world frame to the vehicle frame. Both the world frame and the vehicle frames axes are aligned and no rotation is necessary. In this thesis, subscripts on vectors are used to indicate the reference frame these vectors are defined in. For example: $\mathbf{v}_w$ is a vector defined in the world frame. Transformation matrices follow a similar convention where a transformation's subscripts indicate the domain and range spaces of the transformation. For example: R_{b_v} transforms from the vehicle frame to the body frame. This equation is

$$\mathbf{v}_v = \mathbf{v}_w - \begin{bmatrix} x \\ y \\ z \end{bmatrix} \tag{2.2}$$

and transforms a vector from the world frame to the vehicle frame.

2.4.2 Vehicle to World Frame

This transformation moves a vector from the vehicle frame to the world frame and is the inverse of the preceding transformation. The equation

$$\mathbf{v}_w = \mathbf{v}_v + \begin{bmatrix} x \\ y \\ z \end{bmatrix} \tag{2.3}$$

defines this transformation.

2.4.3 Vehicle to Body Frame

The vehicle frame and body frame are both positioned at the center of mass of the quadrotor. To transform a vector from the vehicle frame to the body frame is a 3D rotation. The equations

$$R_{b\_v} = R_{b\_er} R_{er\_ep} R_{ep\_v}, \tag{2.4}$$

$$R_{b\_v} = \begin{bmatrix} 1 & 0 & 0 \\ 0 & \cos\phi & \sin\phi \\ 0 & -\sin\phi & \cos\phi \end{bmatrix} \begin{bmatrix} \cos\theta & 0 & -\sin\theta \\ 0 & 1 & 0 \\ \sin\theta & 0 & \cos\theta \end{bmatrix} \begin{bmatrix} \cos\psi & \sin\psi & 0 \\ -\sin\psi & \cos\psi & 0 \\ 0 & 0 & 1 \end{bmatrix}, \tag{2.5}$$

$$R_{b\_v} = \begin{bmatrix} \cos\theta\cos\psi & \cos\theta\sin\psi & -\sin\theta \\ \sin\phi\sin\theta\cos\psi - \cos\phi\sin\psi & \sin\phi\sin\theta\sin\psi + \cos\phi\cos\psi & \sin\phi\cos\theta \\ \cos\phi\sin\theta\cos\psi + \sin\phi\sin\psi & \cos\phi\sin\theta\sin\psi - \sin\phi\cos\psi & \cos\phi\cos\theta \end{bmatrix} \tag{2.6}$$

describe this rotation as a rotation matrix and are built by applying all three Euler rotations in their respective coordinate frames. The transformation

$$\mathbf{v}_b = R_{b\_v} \mathbf{v}_v \tag{2.7}$$

can be written more concisely using the rotation matrix.

2.4.4 Body to Vehicle Frame

Transforming from the body frame to vehicle frame is an inverse of the transformation from the vehicle frame to the body frame. The inverse of a rotation matrix is simply its transpose. The equations

$$R_{v\_b} = R_{b\_v}^{\intercal}, \tag{2.8}$$

$$\mathbf{v}_v = R_{v\_b}\mathbf{v}_b \tag{2.9}$$

represent this transformation.

2.4.5 World to Body Frame

Some quantities are measured in the inertial frame and some are measured in the body frame. Combining the transformation from world to vehicle frame with the transformation from the vehicle to body frame yields the following transformation. This can also be written for homogenous coordinates in a single matrix, but the method described by the equations

$$\mathbf{v}_b = R_{b\_v}\mathbf{v}_v, \tag{2.10}$$

$$\mathbf{v}_b = R_{b\_v}(\mathbf{v}_w - \begin{bmatrix} x \\ y \\ z \end{bmatrix}) \tag{2.11}$$

is functionally equivalent.

2.4.6 Body to World Frame

Transforming from the body to world frame is the inverse of the preceding transformation. The equations

$$\mathbf{v}_w = \mathbf{v}_v + \begin{bmatrix} x \\ y \\ z \end{bmatrix}, \tag{2.12}$$

$$\mathbf{v}_w = (R_{v\_b}\mathbf{v}_b) + \begin{bmatrix} x \\ y \\ z \end{bmatrix} \tag{2.13}$$

transform a vector from the world frame to the body frame.

2.4.7 Euler Frames to Body Frame

Instantaneous angular velocity is measured in the body frame, but it affects the angles in their respective Euler frames. This derivation has a small catch; because each vector from the different Euler frames has a single value with zeros in every other row, it is possible to factor out the Euler frame vector and create a single transformation matrix. This transformation is derived by:

$$\begin{bmatrix} x \\ y \\ z \end{bmatrix} = R_{b\_er} \begin{bmatrix} a \\ 0 \\ 0 \end{bmatrix} + R_{b\_er}R_{er\_ep} \begin{bmatrix} 0 \\ b \\ 0 \end{bmatrix} + R_{b\_er}R_{er\_ep}R_{ep\_ey(v)} \begin{bmatrix} 0 \\ 0 \\ c \end{bmatrix}, \tag{2.14}$$

$$\begin{bmatrix} x \\ y \\ z \end{bmatrix} = \begin{bmatrix} a \\ 0 \\ 0 \end{bmatrix} + \begin{bmatrix} 0 \\ \cos(\phi)b \\ -\sin(\phi)b \end{bmatrix} + \begin{bmatrix} -\sin(\theta)c \\ \sin(\phi)\cos(\theta)c \\ \cos(\phi)\cos(\theta)c \end{bmatrix}, \tag{2.15}$$

$$\begin{bmatrix} x \\ y \\ z \end{bmatrix} = \begin{bmatrix} 1 \\ 0 \\ 0 \end{bmatrix} a + \begin{bmatrix} 0 \\ \cos(\phi) \\ -\sin(\phi) \end{bmatrix} b + \begin{bmatrix} -\sin(\theta) \\ \sin(\phi)\cos(\theta) \\ \cos(\phi)\cos(\theta) \end{bmatrix} c, \tag{2.16}$$

$$\begin{bmatrix} x \\ y \\ z \end{bmatrix} = \begin{bmatrix} 1 & 0 & -\sin(\theta) \\ 0 & \cos(\phi) & \sin(\phi)\cos(\theta) \\ 0 & -\sin(\phi) & \cos(\phi)\cos(\theta) \end{bmatrix} \begin{bmatrix} a \\ b \\ c \end{bmatrix}, \tag{2.17}$$

$$T_{b\_e} = \begin{bmatrix} 1 & 0 & -\sin(\theta) \\ 0 & \cos(\phi) & \sin(\phi)\cos(\theta) \\ 0 & -\sin(\phi) & \cos(\phi)\cos(\theta) \end{bmatrix} \tag{2.18}$$

where the scaler values a, b, and c represent scalar values of the input vector.

2.4.8 Body to Euler Frames

Transforming the body to Euler frame is done by taking an inverse of the T_{b_e} transformation matrix. The equations

$$T_{e\_b} = T_{b\_e}^{-1}, \tag{2.19}$$

$$T_{e\_b} = \begin{bmatrix} 1 & \sin(\phi)\tan(\theta) & \cos(\phi)\tan(\theta) \\ 0 & \cos(\phi) & -\sin(\phi) \\ 0 & \frac{\sin(\phi)}{\cos(\theta)} & \frac{\cos(\phi)}{\cos(\theta)} \end{bmatrix} \tag{2.20}$$

show this.

2.5 Rotorcraft Quadrotor Model Overview

There are two quadrotor models used in this thesis. One is a lower level rotorcraft model similar to the models explained in [7, 5]. Another is a higher level similar to the model described in [31]. The lower level model is described in this section. These models represent two ways of modeling a quadrotor aircraft for estimation. The lower level model is used to simulate the quadrotor and for parameter estimation.

2.5.1 Model Inputs

The rotorcraft model used in this thesis abstracts out the motors and blades to provide an input of forces and torques about the airframe. Each value: τ_x, τ_y, and τ_z describes a torque about one of the rotorcraft's axes in the body frame. The force F_z represents the total thrust of the motors, which in a quadrotor is assumed to be pointed in the body z or down direction. The model input vector

$$\mathbf{u} = \begin{bmatrix} \tau_x \\ \tau_y \\ \tau_z \\ F_z \end{bmatrix} \tag{2.21}$$

represents these input values.

2.5.2 Equations of Motion

The following model dynamics show how the parameters and inputs are used to update the state of the quadrotor. These transitions are described using a series of first order differential equations relating each quantity using its derivative.

Position Differential Equations

The equations

$$\begin{bmatrix} \dot{x} \\ \dot{y} \\ \dot{z} \end{bmatrix} = \mathbf{v}_w = \mathbf{v}_v = R_{v\_b} \begin{bmatrix} u \\ v \\ w \end{bmatrix} \tag{2.22}$$

show how position in the world frame is a function of the body frame velocities and the quadrotor's orientation.

Velocity Differential Equations

The velocity differential equations relate the force from gravity, thrust, drag, and the Coriolis force to the acceleration of the robot's body frame. These accelerations are used to update the velocity state. The Coriolis force is a result of defining the quadrotor's velocities in the body frame, which is an accelerating reference frame. The equations

$$F_{thrust} = \begin{bmatrix} 0 \\ 0 \\ F_z \end{bmatrix}, \tag{2.23}$$

$$F_{drag} = \begin{bmatrix} k_{drag}u \\ k_{drag}v \\ 0 \end{bmatrix}, \tag{2.24}$$

$$F_{gravity} = \begin{bmatrix} 0 \\ 0 \\ mg \end{bmatrix}, \tag{2.25}$$

$$\mathbf{p} = m \begin{bmatrix} u \\ v \\ w \end{bmatrix}, \tag{2.26}$$

$$F_{coriolis} = -(\begin{bmatrix} p \\ q \\ r \end{bmatrix} \times \mathbf{p}), \tag{2.27}$$

$$F_{total} = F_{thrust} + F_{drag} + R_{b\_v} F_{gravity} + F_{coriolis} \tag{2.28}$$

are used to find the total force from thrust, drag, gravity, and the Coriolis force. The body accelerations are a direct function of the forces and the quadrotor's mass. This next equation,

$$\begin{bmatrix} \dot{u} \\ \dot{v} \\ \dot{w} \end{bmatrix} = \mathbf{a}_b = \frac{1}{m} * F_{total} \tag{2.29}$$

relates the total force to accelerations in the body frame.

Euler Angle Differential Equations

The angular velocities must also be related to the Euler angles in a similar way to the linear velocities and position values, except it is necessary to use the transformation from the body frame to the Euler frames. The equations

$$\begin{bmatrix} \dot{\phi} \\ \dot{\theta} \\ \dot{\psi} \end{bmatrix} = \omega_e = T_{e\_b} \begin{bmatrix} p \\ q \\ r \end{bmatrix} \tag{2.30}$$

apply this transformation to the angular rates.

Angular Velocities Differential Equations

Torques are related to angular accelerations in an analogous way to forces and linear acceleration. There is also a torque that is a function of the angular velocities and moments

of inertia. It is analogous to the Coriolis force, and is a result of defining the angular velocities in the body frame, which is accelerating. The equations

$$\mathbf{J} = \begin{bmatrix} J_x & 0 & 0 \\ 0 & J_y & 0 \\ 0 & 0 & J_z \end{bmatrix}, \tag{2.31}$$

$$\mathbf{L} = \mathbf{J} \begin{bmatrix} p \\ q \\ r \end{bmatrix}, \tag{2.32}$$

$$\tau_c = -(\begin{bmatrix} p \\ q \\ r \end{bmatrix} \times \mathbf{L}), \tag{2.33}$$

$$\tau_{motors} = \begin{bmatrix} \tau_x \\ \tau_y \\ \tau_z \end{bmatrix}, \tag{2.34}$$

$$\tau_{total} = \tau_{total} + \tau_c \tag{2.35}$$

represent the angular velocity equations. Angular velocities are a function of the total torque and the quadrotor's moments of inertia. The equation

$$\begin{bmatrix} \dot{p} \\ \dot{q} \\ \dot{r} \end{bmatrix} = \alpha_b = \mathbf{J}^{-1}\tau_{total} \tag{2.36}$$

shows how to obtain angular accelerations from torques in the body frame. This completes the differential equations that define the rotorcraft model state transitions. The equations of motion define how the states change through time as functions of the model inputs and parameters.

2.5.3 Model Parameters

This model has a number of constant parameters that represent physical properties of nature and constant properties of the quadrotor. The first quantity is the quadrotor's mass m. This value can be easily measured. The second parameter is the gravity constant g. The next three values represent the quadrotor's moments of inertia. Because the quadrotor is radially symmetric, the important moments of inertia are along the airframes primary axes. These moments are represented by J_x, J_y, J_z. The final parameter is the drag coefficient k_{drag}. This value relates the linear velocity of the quadrotor to a deceleration in the opposite direction. The model parameter array

$$K = \{m, g, J_x, J_y, J_z, k_{drag}\} \tag{2.37}$$

contains these parameters.

2.6 High Level Quadrotor Model Overview

The high level quadrotor model is designed to represent a quadrotor aircraft with an onboard controller. This controller is completely abstracted and the system is considered as a black box. It takes values for commanded roll angle, pitch angle, yaw rate, and total thrust. These input values can be related to the UAV's true orientation and total thrust using a simplified version of the rotorcraft model. The high level model is a pared down version of the rotorcraft model, where the roll, pitch, yaw rate and total thrust are assumed to be be controlled directly. This model works if the relationship between commanded values and true values of the airframe do not change over time and if the delay between when a value is commanded and when the state truly changes is small. In reality, there is a delay between the commanded values and realized values, and the constant parameters for thrust change as a function of the quadrotor's battery level. This model is very convenient when working with an off the shelf quadrotor, however these violated assumptions have an effect on estimation as will be discussed later in the thesis.

2.6.1 Model Inputs

The following inputs are directly commanded in the high level model. These inputs,

$$\mathbf{u} = \begin{bmatrix} \phi_{command} \\ \theta_{command} \\ \dot{\psi}_{command} \\ F_{zcommand} \end{bmatrix} \tag{2.38}$$

are taken by the aircraft's onboard controller that uses onboard sensors and functions to estimate the quadrotor's pitch, roll, yaw, and thrust.

2.6.2 Equations of Motion

The differential equations defining the high level model are very similar to the rotorcraft model. As stated before, the critical differences are the assumption that the pitch, roll, yaw rate and thrust are commanded directly. This leads to the elimination of a few differential equations. The following equation

$$\begin{bmatrix} \dot{x} \\ \dot{y} \\ \dot{z} \end{bmatrix} = \mathbf{v}_v = R_{v\_b} \begin{bmatrix} u \\ v \\ w \end{bmatrix} \tag{2.39}$$

is the equation for change in world position using body velocities.

The equations

$$F_z = k_{thrust} F_{zcommand}, \tag{2.40}$$

$$F_{thrust} = \begin{bmatrix} 0 \\ 0 \\ F_z \end{bmatrix}, \tag{2.41}$$

$$F_{drag} = \begin{bmatrix} k_{drag} u \\ k_{drag} v \\ 0 \end{bmatrix}, \tag{2.42}$$

$$F_{gravity} = \begin{bmatrix} 0 \\ 0 \\ mg \end{bmatrix}, \tag{2.43}$$

$$\mathbf{p} = m \begin{bmatrix} u \\ v \\ w \end{bmatrix}, \tag{2.44}$$

$$F_{coriolis} = -(\begin{bmatrix} p \\ q \\ r \end{bmatrix} \times \mathbf{p}), \tag{2.45}$$

$$F_{total} = F_{thrust} + F_{drag} + R_{b\_v} F_{gravity} + F_{coriolis}, \tag{2.46}$$

$$\begin{bmatrix} \ddot{x}_b \\ \ddot{y}_b \\ \ddot{z}_b \end{bmatrix} = \mathbf{a}_b = \frac{1}{m} * F_{total} \tag{2.47}$$

represent the changes in body velocities given the commanded thrust and other forces. These relationships are similar to the rotorcraft model.

Changes in the aircraft's yaw is related to angular rates like in the rotorcraft model. However, changes in pitch and roll are direct functions of the commanded values. The differential equations

$$\begin{bmatrix} 0 \\ 0 \\ \dot{\psi} \end{bmatrix} = T_{e\_b} \begin{bmatrix} 0 \\ 0 \\ r \end{bmatrix}, \tag{2.48}$$

$$\phi = k_{angle} \phi_{command}, \tag{2.49}$$

$$\theta = k_{angle} \theta_{command} \tag{2.50}$$

show these relationships, where k_{angle} is necessary to the commanded angles are defined by the onboard black box system in non-standard units and must be converted to radians.

The pitch are roll rates are assumed to be zero and the change in yaw rate is also a function of the commanded value. The equation

$$r = k_{angle}\dot{\psi}_{command} \tag{2.51}$$

defines this relationship. These equations of motion are simplified and define how the inputs to the onboard black box controller effects the state vector as time moves forward.

2.6.3 Model Parameters

The model parameters for the high level model are the quadrotor's mass and moments of inertia just like the rotorcraft model. There are also three constants that relate input values to physical states in the model. The parameter k_{drag} is the same as the rotorcraft model. The parameter k_{thrust} relates the thrust input value to the total force in the body z direction, and the k_{angle} value relates the inputs for the commanded roll and pitch values to their true Euler angle values in radians. The parameter array

$$K = \{m, g, J_x, J_y, J_z, k_{drag}, k_{thrust}, k_{angle}\} \tag{2.52}$$

encapsulates these quantities.

Chapter 3

Sensor Models

For accurate simulation and estimation of the UAV's state, it is useful to have accurate sensor models. In this section, models for the following are described: MEMS gyroscopes, MEMS accelerometers, and a simplified model of the Microsoft Kinect. A more detailed description of MEMS gyroscope and accelerometer models can be found in [5, 2]. In order for good estimation, good sensor models must be used. In this section the sensor models used in this thesis for estimation are described. For further reading see [4, 2, 8].

In this thesis, it is assumed that the IMU sensor is aligned with the body frame axes and center at the quadrotor's center of mass. It is also assumed that the 3D camera is positioned at the origin of the body frame. However, measurements from the 3D camera are taken using a different set of axes. The necessary reference frames and transformations used for sensing are also described.

3.1 Reference Frames

The first of the frames of reference described below is used for translating sensor readings into the reference frames used in the quadrotor models. The other reference frame, the object frame, will be used later in the discussion of aircraft stabilization.

3.1.1 The Object Frame of Reference

The object frame $\mathcal{F}_o$ is positioned at the center of mass of an object of interest. We assume the object is not moving and the axes are aligned with the world frame. This relative frame is used by the 3D camera to provide periodic pose updates relative to the object of interest.

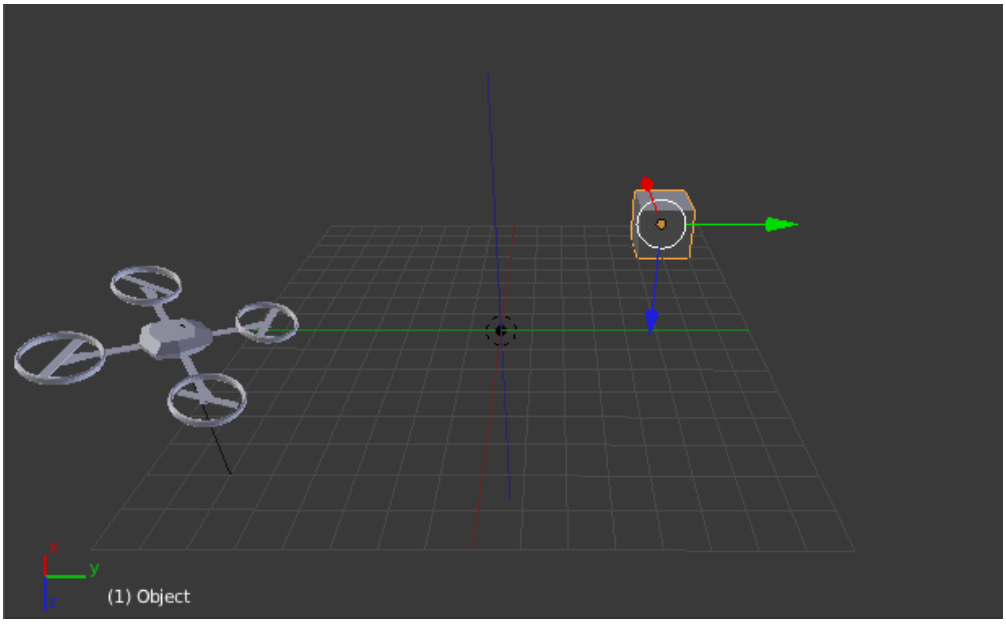

Figure 3.1: Object Frame

3.1.2 The Camera Frame

The camera frame $\mathcal{F}_c$ is different from many of the other reference frames. It is positioned at the optical center for the camera. The x axis points out the right side of the camera. The y axis points down out the bottom of the camera. The z axis points forward from the optical center along the optical axis. The assumption is made that the camera's optical center is located at the same position as the body frame (the center of mass of the quadrotor). It is also assumed that the camera frame z axis (optical axis) is aligned with the body frame x axis, the camera frame y axis is aligned with the body frame z axis and the camera frame x axis is a aligned with the body frame y axis. Camera measurements are received in the camera frame and must be transformed into the body frame to be used by the estimator.

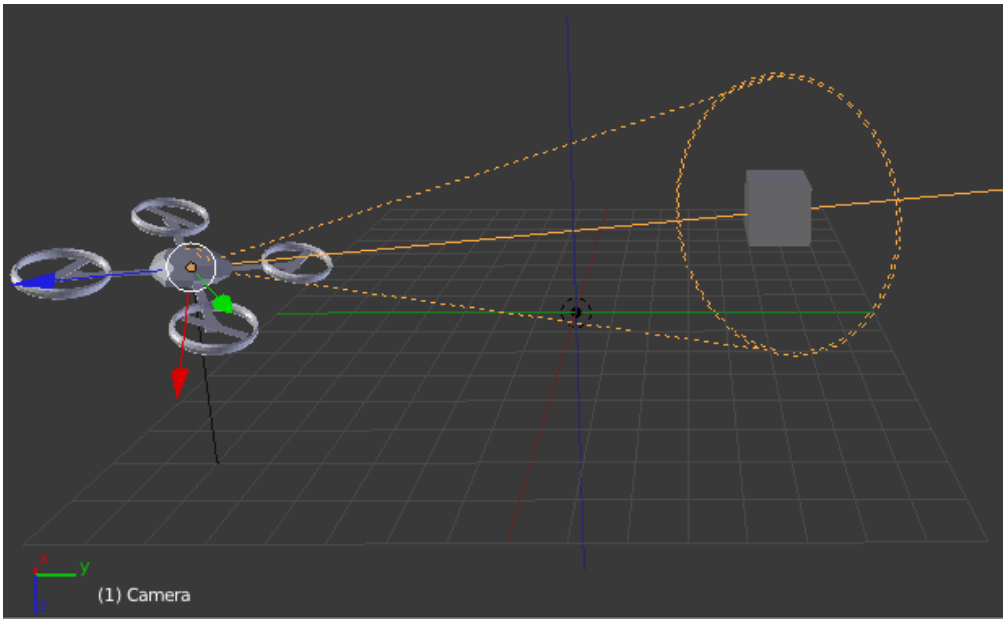

Figure 3.2: Camera Frame

3.2 Transformations

The following transformations are used to utilize measurements from the camera and estimate pose relative to an object in the quadrotor's environment.

3.2.1 Object Frame to Body Frame

It is important to be able to transform measurements in the object frame to measurements in the body frame. This is described by the equations

$$\mathbf{v}_w = \mathbf{v}_o + \begin{bmatrix} x_{object} \\ y_{object} \\ z_{object} \end{bmatrix}, \tag{3.1}$$

$$\mathbf{v}_b = R_{b\_v}(\mathbf{v}_w - \begin{bmatrix} x \\ y \\ z \end{bmatrix}) \tag{3.2}$$

and is done in a similar manner to transforming from the world frame to the body frame.

3.2.2 Body Frame to Object Frame

Moving a vector from the body frame to the object frame is the inverse of the above transformation

$$\mathbf{v}_w = (R_{v\_b}\mathbf{v}_b) + \begin{bmatrix} x \\ y \\ z \end{bmatrix}, \tag{3.3}$$

$$\mathbf{v}_o = \mathbf{v}_w - \begin{bmatrix} x_{object} \\ y_{object} \\ z_{object} \end{bmatrix} \tag{3.4}$$

and is analogous to translating from the body frame to the world frame.

3.2.3 Camera Frame to Body Frame

The camera frame is aligned with the body frame, but the axes are labeled differently. This difference in convention must be considered to use measurements made by the 3D camera. The matrix

$$T_{b\_c} = \begin{bmatrix} 0 & 0 & 1 \\ 1 & 0 & 0 \\ 0 & 1 & 0 \end{bmatrix}, \tag{3.5}$$

$$\mathbf{v}_b = T_{b\_c}\mathbf{v}_c \tag{3.6}$$

changes values in the camera's frame to the body frame. This is assuming the camera's optical center (usually the center of the camera's lens) is also the aircraft's center of mass. This isn't always true, but can be approximated by deliberately positioning the camera when constructing the airframe.

3.2.4 Body Frame to Camera Frame

This inverse transformation

$$T_{c\_b} = \begin{bmatrix} 0 & 1 & 0 \\ 0 & 0 & 1 \\ 1 & 0 & 0 \end{bmatrix}, \tag{3.7}$$

$$\mathbf{v}_c = T_{c\_b}\mathbf{v}_b \tag{3.8}$$

is just the transpose of the transformation from the camera frame to body frame.

3.3 Accelerometer Model

The MEMS accelerometer can be simply modeled as a device that measures the deflection of a proof mass attached inside a container by a lever arm [5]. When the sensor moves, the proof mass generates a deflection of the lever arm that is directly related to acceleration. However, because acceleration due to gravity operates both on the proof mass and on the external container the proof mass is attached to, acceleration due to gravity is not measured by the accelerometer. This yields the mathematical model

$$\mathbf{a_b} = \frac{1}{m}(\mathbf{F_{total}} - \mathbf{F_{gravity}}) \tag{3.9}$$

where both $\mathbf{F_{total}}$ and $\mathbf{F_{gravity}}$ are measured in the body frame.

One reason this sensor is useful is because when the sensor is not accelerating and $F_{total} = 0$ the sensor measures $-F_{gravity}$. This gravity vector can be used to estimate the sensor's orientation. For a quadrotor, unless the aircraft is in a controlled hover, it is in a state of acceleration. This leads to the accelerometer sensor on the quadrotor giving unexpected angle estimates from assuming $F_{total} = 0$. This can be compensated for using techniques described in [2, 16]

3.3.1 Measurement Model

The true accelerometer measurement is modeled as a raw measurement with a misalignment matrix, a scale factor, and bias factor applied to it. Without knowing these factors it is not possible to get accurate acceleration readings. Estimating the bias and scale parameters is discussed in the next chapter. The matrix **M**, or

$$\begin{bmatrix} a_{xmeasured} \\ a_{ymeasured} \\ a_{zmeasured} \end{bmatrix} = \mathbf{M} \begin{bmatrix} 1/k_{xscalea} & 0 & 0 \\ 0 & 1/k_{yscalea} & 0 \\ 0 & 0 & 1/k_{zscalea} \end{bmatrix} \begin{bmatrix} a_{xraw} - k_{xbiasa} \\ a_{yraw} - k_{ybiasa} \\ a_{zraw} - k_{zbiasa} \end{bmatrix} \tag{3.10}$$

is a misalignment matrix that compensates for the accelerometer's axes being misaligned and non-orthogonal. Solving for this matrix is also a subject discussed in the next chapter.

3.3.2 Noise Model

A description of the sensor noise,

$$\mathbf{a_b} = \begin{bmatrix} a_{xmeasured} \\ a_{ymeasured} \\ a_{zmeasured} \end{bmatrix} + \begin{bmatrix} \mathcal{N}_x \\ \mathcal{N}_y \\ \mathcal{N}_z \end{bmatrix} \tag{3.11}$$

can be used to filter accelerometer readings or for better measurements. For each axis, sensor noise can be modeled as a zero mean normally distributed random variable with a standard deviation of σ.

3.4 Gyroscope Model

The MEMS gyroscope gives noisy measurements of the vehicles's angular rotation rates p, q, and r. It works by measuring the coriolis acceleration of a vibrating proof mass. For further explanation see [4].

3.4.1 Measurement Model

Similar to the accelerometer, measurements from the gyroscope can be modeled as a raw measurement with a misalignment matrix, a scale factor, and bias factor. Generally the bias factor on a gyroscope is less significant than the bias factor on the accelerometer [20]. The matrix **M**, or

$$\begin{bmatrix} \omega_{xmeasured} \\ \omega_{ymeasured} \\ \omega_{zmeasured} \end{bmatrix} = \mathbf{M} \begin{bmatrix} 1/k_{xscale\omega} & 0 & 0 \\ 0 & 1/k_{yscale\omega} & 0 \\ 0 & 0 & 1/k_{zscale\omega} \end{bmatrix} \begin{bmatrix} \omega_{xraw} - k_{xbias\omega} \\ \omega_{yraw} - k_{ybias\omega} \\ \omega_{zraw} - k_{zbias\omega} \end{bmatrix} \tag{3.12}$$

is also misalignment matrix.

3.4.2 Noise Model

The noise characteristics of measurements from each axis of the gyroscope are also well represented by a zero mean Gaussian distribution. This is shown in the equation

$$\mathbf{a_b} = \begin{bmatrix} \omega_{xmeasured} \\ \omega_{ymeasured} \\ \omega_{zmeasured} \end{bmatrix} + \begin{bmatrix} \mathcal{N}_x \\ \mathcal{N}_y \\ \mathcal{N}_z \end{bmatrix} \tag{3.13}$$

where σ standard deviation of the distribution.

3.5 3D Camera Model

The 3D camera can be simply modeled when using the Microsoft Kinect. The Microsoft Kinect works by projecting a known infrared pattern onto the scene and senses that pattern to extrapolate depth. These depth values are then corresponded to color values taken by a regular color camera. There are calibration parameters that can be solved for as explained by [18] but these parameters do not make a significant difference in Kinect color and depth readings. The Kinect driver gives a set of two values for each pixel p_{ij} and d_{ij}. The value p_{ij} represents the color information for that pixel. This color information corresponds to a physical 3D point located at d_{ij} in the camera reference frame. These measurements

can be transformed into the body frame using (3.5). Noise on the Kinect 3d measurements can also be approximated by a zero mean Gaussian distribution

$$p_{ij} = \begin{bmatrix} b \\ g \\ r \end{bmatrix}, \tag{3.14}$$

$$d_{ij} = \begin{bmatrix} x \\ y \\ z \end{bmatrix} + \begin{bmatrix} \mathcal{N}_x \\ \mathcal{N}_y \\ \mathcal{N}_z \end{bmatrix} \tag{3.15}$$

with a standard deviation of σ. This simple model is relatable to the radial/axial Gaussian noise model described in [47]. These values maintain their linearity when (3.5) is applied to transform these quantities into the body frame.

Chapter 4

Estimating Model and Sensor Parameters

Accurate simulation and estimation is highly dependent on good models and consequently good model parameter estimates. As discussed in the introduction, there are a number of methods to measure or estimate values for model and sensor parameters, either through measurement or estimation. In this chapter, a novel approach for quadrotor parameter estimation is contributed using the sequential importance resampling (SIR) particle filter [23, 48]. The SIR particle filter has been successfully applied to parameter and state estimation problems in a number of different fields [24, 48]. Also, methods for calibrating (or estimating parameters) for the MEMS accelerometer and gyroscope are described. These methods use least squares to estimate the bias, scale, and misalignment parameters for each sensor [21, 20].

4.1 Quadrotor Parameter Estimation

4.1.1 Overview

The particle filter method for parameter estimation is possible when given a set of true state values measured directly from a trial flight of the UAV. These values give true state estimates that enable the particle filter to adjust its distribution and hone in on the true parameters.

These truth values were obtained with a series of trial flights where state measurements were acquired using a motion analysis system identical to the one described in [2]. This system uses a number of specialized infrared cameras that flood the room with infrared light and track small reflective balls attached to the UAV's airframe. It gives accurate fast pose data that can be differentiated to obtain both linear and angular velocities.

The filter works by using these measured states from a trial flight and by searching for parameters that will enable a simulated version of the quadrotor models to match the real state measurements as closely as possible. As the model iterates, the difference between the simulated state and the true state shrinks as the parameters converge to more accurate values. This technique is only effective if the models are well designed and reasonably model the quadrotor and if the trial flight excites each parameter being estimated.

4.1.2 Particle Filter

The particle filter is a Monte Carlo estimation method that can be derived as local stochastic search or an approximated Bayes filter. The filter is turned by adjusting noise parameters on both the state transition model and the measurement model. For a description of the particle filter within the Bayes framework see [48].

Thinking about the particle filter as stochastic local search, it generates a number of hypotheses for parameter estimates called particles. Each particle represents a specific guess as to the true parameters. For each iteration of the filter, each particle is simulated forward using a state transition function g with the addition of some process noise $\mathcal{N}(0, \sigma_g)$. After the state transition, the particle is used to generate a hypothetical measurement. If the true parameters were x_i, the measurement is estimated to be $h(x_i) = z_i$. This measurement is then compared to a true measurement z. The closer the hypothetical measurement is to the true measurement, the more likely that this particle is good or close to the true parameters. The likelihood of this hypothesis being true is a function of the probability distribution function of some known normal measurement noise $f_{\mathcal{N}}(\hat{z}_i, \mu_h, \sigma_h)$. These particles are then resampled where the higher weighted particles are more likely to be kept for the next iteration of the particle filter. The more particles in the filter, the more effective the estimate is. This particle filter algorithm is explored below.

```
// apply state transition to particles
foreach particle x_i do
    x̂_i = g(x_i) + N(0, σ_g)
end
// calculate particle weights
foreach particle x_i do
    ẑ_i = h(x_i)
    μ_h = z;
    w_i = f_N(ẑ_i, μ_h, σ_h)
end
x_next = resample(x, w)
```

4.1.3 Estimating Rotorcraft Model Parameters

In order to estimate the rotorcraft parameters the particle filter is designed so that the measurements are quadrotor states measured by the motion capture system. The equation

$$z = \begin{bmatrix} x & y & z & u & v & w & \phi & \theta & \psi & p & q & r \end{bmatrix}^\intercal \tag{4.1}$$

shows an example measurement.

Each particle represents a hypothesis set of parameter estimates. The equation

$$x_i = \begin{bmatrix} J_x & J_y & J_z & k_{drag} & k_{thrust} \end{bmatrix}^\intercal \tag{4.2}$$

shows what a single particle represents.

The particle state transition function $g(x_i)$ is the identity and simply returns the current state. In other words, the model parameters are expected to change as a function of time. Although state transition is trivial, it is still critical to add process noise σ_g to enable each particle to diversify and move closer to the target parameter or distribution. The lower the process noise, the more precise the estimate the particle filter can converge to. The higher the process noise, the larger the local search area for each particle.

The measurement model function $h(x_i)$ is a complete simulation of the rotorcraft model given a set of parameters. This simulation is run for a period of time given the hypothetical parameters and the resulting state is compared to the true measured state

values. This is then used to generate a particle weight. The lower the variance on the measurement noise σ_h, the faster the filter converges and the more confident the filter is in rejecting distant hypotheses.

4.1.4 Experimentation and Estimation Results

In order to test the effectiveness of this estimation technique, we designed an experiment using a real quadrotor aircraft, motion capture system, and complete implementation of the filter. This section considers the experiment.

Measuring the Expected Model Parameters

Before running the filter, the moments of inertia were measured using the compound pendulum method and the assumptions described in [4]. These measured values were then used to evaluate the effectiveness of the estimator. The expected value for the thrust coefficient was measured by attaching the quadrotor to a stand attached above a digital scale. The thrust coefficient was then estimated by recording command thrust and force measured on the scale, similar to the method described in [7]. The expected value for the draw coefficient was measured using a series of carefully controlled experimental flights and simulated flight data solving for the coefficient by least squares fitting to the model described in the previous section. Using these methods, the values $x = \begin{bmatrix} .0872 & .143 & .154 & .0955 & 0.632 \end{bmatrix}^\intercal$ were used to evaluate the effectiveness of the estimator.

Estimating the Model Parameters

The first step to estimating the quadrotor model's parameters was to run a test flight with aggressive maneuvering in order to excite all modes of the model. During this test flight, the 12 state variables of the quadrotor were measured using the motion capture system. After this flight had been recorded, the stochastic search or particle filter estimator was run with 3000 particles. The particles were distributed with an initial distribution of

$\mathcal{N}(\mu_{init}, \sigma_{init})$. These values

$$\mu_{init} = \begin{bmatrix} 0 & 0 & 0 & 0 & 0 \end{bmatrix}^\intercal, \tag{4.3}$$

$$\sigma_{init} = \begin{bmatrix} .2 & .2 & .2 & .1 & .1 \end{bmatrix}^\intercal \tag{4.4}$$

were selected to be reasonably far from the measured values. This was done to be representative of someone using the filter with little knowledge of the true values.

The process noise

$$\sigma_{process} = \begin{bmatrix} .01 & .01 & .01 & .001 & .001 \end{bmatrix}^\intercal \tag{4.5}$$

was selected on the order of the expected values to provide a small amount of sample diversity and to allow the filter to move around false local minima. The measurement noise

$$\sigma_{measurement} = \begin{bmatrix} .1 & .1 & .1 & .01 & .01 \end{bmatrix}^\intercal \tag{4.6}$$

was similarly selected in an effort to control the confidence and convergence speed of the estimator.

Estimator Results

Three methods are used to evaluate the effectiveness of the estimator. First we show how the sum squared error between the predicted state using the estimated parameters and the true state measured by the motion capture system decreases as time increases. In other words, given the true position of the quadrotor at any time, the simulation is able to use the estimated parameters to more accurately predict the next true state of the quadrotor at each time step. Eventually this prediction error using the high level quadrotor model and estimated parameters decreases significantly. This can be seen in the figure below.

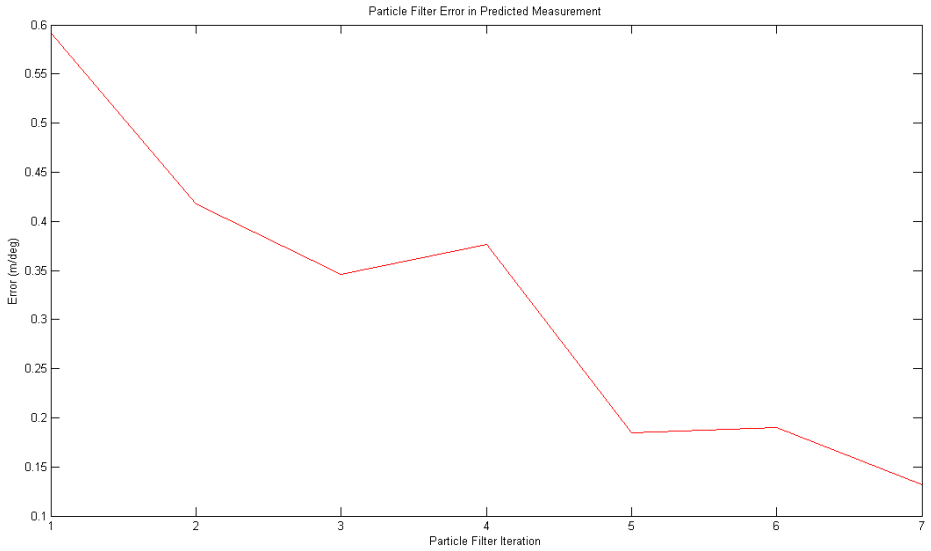

Figure 4.1: Quadrotor Parameter Estimation: Particle Filter Prediction Error

The next method we used to show the effectiveness of the filter in estimating model parameters is by simulating the motion of the quadrotor given true inputs and comparing the simulator's output using the estimated parameters to the true output of the system. The following figure shows error between the predicted state variables and the simulated state variables given the same inputs.

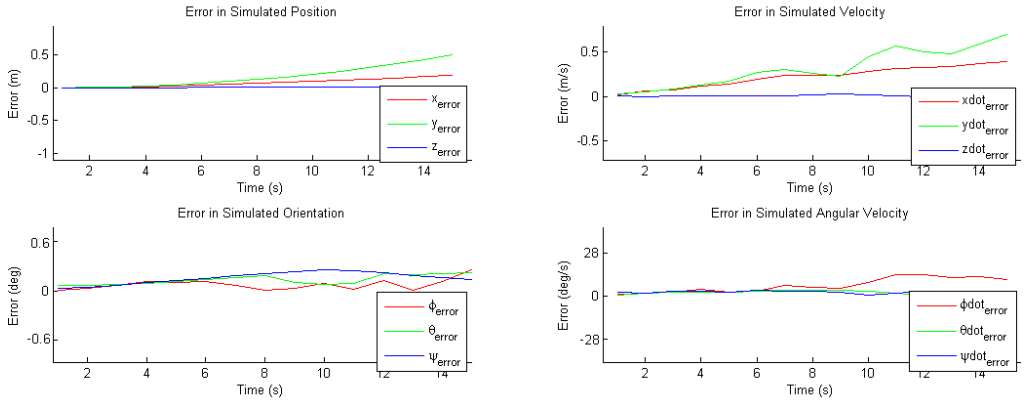

Figure 4.2: Quadrotor Simulation Results Using Estimated Parameters

It can be seen that over the 1.5 seconds of simulation, the error in estimated values slowly accumulates and eventually begins to diverge. However, given that the model is imperfect, the parameter estimates are very usable, considering that multiple vision updates a second from vision will be used to correct for prediction drift.

Finally, we compare the estimated parameters with the measured model parameters

$$x = \begin{bmatrix} .0872 & .143 & .154 & .0955 & 0.632 \end{bmatrix}^\intercal, \tag{4.7}$$

$$\hat{x} = \begin{bmatrix} .0772 & .129 & .131 & 0.0965 & 0.466 \end{bmatrix}^\intercal \tag{4.8}$$

and show that the estimated parameters are indeed close to the measured parameters.

Looking at all three metrics it is clear that the particle filter is capable of making useful estimates of the quadrotor model parameters. It is especially motivating that the system is simple to implement and design if a motion capture system is available, and may provide advantages over hand measurement if the parameters are changing rapidly due to hardware or airframe changes.

4.1.5 Notes

This parameter estimation method performed similarly in a number of trials on different data sets. Occasionally the filter failed to converge on a value close to measured parameters or converge with a low measurement error. Increasing the number of particles in the filter tended to greatly improve estimate quality. Also of note is that the filter may have performed better if each of the iterations calculated the error over the entire data set rather than each iteration using a different element of the data set. However, even with these drawbacks, the particle filter seems to be an effective and convenient method for quadrotor parameter estimation when truth data is readily available. The particle filter is also especially useful when the state transition or measurement model is complicated.

4.2 Accelerometer Cross Axis Calibration

4.2.1 Problem

The cross axis calibration or parameter estimation was also done using known quadrotor states from information collected by the motion capture system. Due to quadrotor's tendency to be constantly accelerating when in flight, the airframe was moved slowly through the space by hand and the orientation of the aircraft was recorded at each step of the process and used to calculate the position of the gravity vector in the body frame of the aircraft. At the same time, the raw accelerometer measurements in the body frame were recorded and paired up with the measured gravity. These two measurements were used to estimated the accelerometer's cross calibration matrix. This method uses least squares to find the parameters that minimize the sum squared error between the estimated gravity vector using accelerometer measurements and the measured gravity vector from the motion capture system. The estimation method is discussed in [20, 21].

4.2.2 Method

The following system represents the relationship between the raw accelerometer measurements, the model parameters and the measured gravity vector for each data point from a the data collection. In the equations,

$$\mathbf{Ax} = \mathbf{b}, \tag{4.9}$$

$$\begin{bmatrix} F_{accel\_1} \\ F_{accel\_2} \\ F_{accel\_3} \\ \vdots \\ F_{accel\_N} \end{bmatrix} \begin{bmatrix} k_{xscalea} & m_{xy} & m_{xz} \\ m_{yx} & k_{yscalea} & m_{yz} \\ m_{zx} & m_{zy} & k_{zscalea} \\ k_{xbiasa} & k_{ybiasa} & k_{zbiasa} \end{bmatrix} = \begin{bmatrix} F_{mocap\_1} \\ F_{mocap\_2} \\ F_{mocap\_3} \\ \vdots \\ F_{mocap\_N} \end{bmatrix} \tag{4.10}$$

the accelerometer measurements are stored as homogenous row coordinate vectors $F_{accel\_N} = \begin{bmatrix} ax & ay & az & 1 \end{bmatrix}$. The gravity vectors or acceleration measured from the motion capture system are stored as column vectors $F_{mocap\_N} = \begin{bmatrix} ax & ay & az \end{bmatrix}$. The cross calibration matrix is a transformation matrix where $k_{xscalea}$, $k_{yscalea}$, and $k_{zscalea}$ represent the accelerometer

gains, k_{xbiasa}, k_{ybiasa}, and k_{zbiasa} represent the accelerometer biases, and the off diagonal elements m represent cross-axis relationships.

The vectors and matrices

$$\hat{\mathbf{x}} = (\mathbf{A}^\intercal \mathbf{A})^{-1} \mathbf{A}^\intercal \mathbf{b}, \tag{4.11}$$

$$\begin{bmatrix} \hat{k}_{xscalea} & \hat{m}_{xy} & \hat{m}_{xz} \\ \hat{m}_{yx} & \hat{k}_{yscalea} & \hat{m}_{yz} \\ \hat{m}_{zx} & \hat{m}_{zy} & \hat{k}_{zscalea} \\ \hat{k}_{xbiasa} & \hat{k}_{ybiasa} & \hat{k}_{zbiasa} \end{bmatrix} = \left(\begin{bmatrix} F_{accel\_1} \\ F_{accel\_2} \\ F_{accel\_3} \\ \vdots \\ F_{accel\_N} \end{bmatrix}^\intercal \begin{bmatrix} F_{accel\_1} \\ F_{accel\_2} \\ F_{accel\_3} \\ \vdots \\ F_{accel\_N} \end{bmatrix} \right)^{-1} \begin{bmatrix} F_{accel\_1} \\ F_{accel\_2} \\ F_{accel\_3} \\ \vdots \\ F_{accel\_N} \end{bmatrix}^\intercal \begin{bmatrix} F_{mocap\_1} \\ F_{mocap\_2} \\ F_{mocap\_3} \\ \vdots \\ F_{mocap\_N} \end{bmatrix} \tag{4.12}$$

define the least squares solution. The solution matrix $\mathbf{x}$ represents the off diagonal misalignment terms, the scale terms, and the accelerometer bias for each axis.

4.2.3 Results

Results from the least squares cross axis calibration show that the accelerometer can be calibrated quickly and effectively using known state information from a motion capture data collection. The figure below shows two lines representing the error between angles measured by the accelerometer and the angles measured by the motion capture system in a real trial flight. The trial flight was different from the data collection used to calibrate the sensor. It can be seen that in general the calibrated sensor measurements trend well with the motion capture measurements when the quadrotor is in calm flight. The large spikes of error are probably do to accelerations in the quadrotor violating the model assumptions used to estimate roll and pitch.

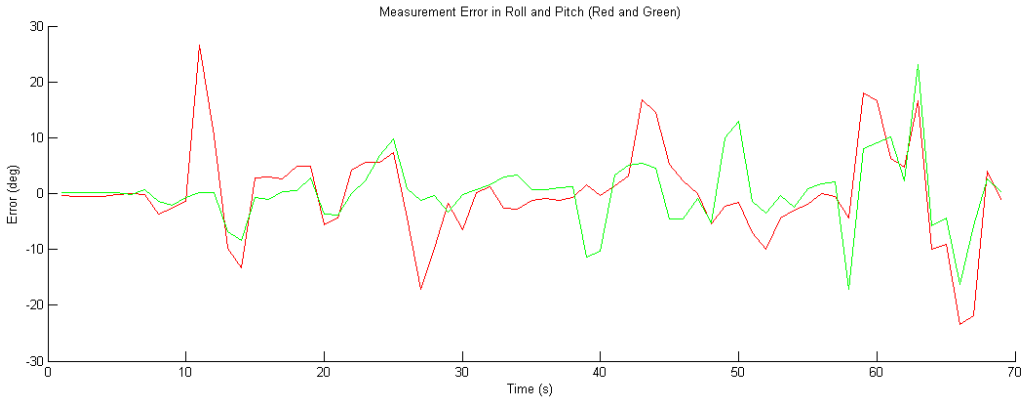

Figure 4.3: Cross Axis Accelerometer Calibration Results: Measurement Error in Roll and Pitch

4.3 Gyroscope Cross Axis Calibration

4.3.1 Problem

Calibration of the gyroscope using the least square method is almost identical to calibration of the accelerometer. The notation and mathematics are analogous.

4.3.2 Method

Again, the equations

$$\mathbf{Ax} = \mathbf{b}, \tag{4.13}$$

$$\begin{bmatrix} \omega_{gyro\_1} \\ \omega_{gyro\_2} \\ \omega_{gyro\_3} \\ \vdots \\ \omega_{gyro\_N} \end{bmatrix} \begin{bmatrix} k_{xscale\omega} & m_{xy} & m_{xz} \\ m_{yx} & k_{yscale\omega} & m_{yz} \\ m_{zx} & m_{zy} & k_{zscale\omega} \\ k_{xbias\omega} & k_{ybias\omega} & k_{zbias\omega} \end{bmatrix} = \begin{bmatrix} \omega_{mocap\_1} \\ \omega_{mocap\_2} \\ \omega_{mocap\_3} \\ \vdots \\ \omega_{mocap\_N} \end{bmatrix} \tag{4.14}$$

set up the system of equations where the measured velocities are functions of the raw velocities and MEMS gyroscope parameters.

The equations

$$\hat{\mathbf{x}} = (\mathbf{A}^\intercal \mathbf{A})^{-1} \mathbf{A}^\intercal \mathbf{b}, \tag{4.15}$$

$$\begin{bmatrix} \hat{k}_{xscale\omega} & \hat{m}_{xy} & \hat{m}_{xz} \\ \hat{m}_{yx} & \hat{k}_{yscale\omega} & \hat{m}_{yz} \\ \hat{m}_{zx} & \hat{m}_{zy} & \hat{k}_{zscale\omega} \\ \hat{k}_{xbias\omega} & \hat{k}_{ybias\omega} & \hat{k}_{zbias\omega} \end{bmatrix} = \left(\begin{bmatrix} F_{gyro\_1} \\ F_{gyro\_2} \\ F_{gyro\_3} \\ \vdots \\ F_{gyro\_N} \end{bmatrix}^\intercal \begin{bmatrix} F_{gyro\_1} \\ F_{gyro\_2} \\ F_{gyro\_3} \\ \vdots \\ F_{gyro\_N} \end{bmatrix} \right)^{-1} \begin{bmatrix} F_{gyro\_1} \\ F_{gyro\_3} \\ F_{gyro\_3} \\ \vdots \\ F_{gyro\_N} \end{bmatrix}^\intercal \begin{bmatrix} \omega_{mocap\_1} \\ \omega_{mocap\_2} \\ \omega_{mocap\_3} \\ \vdots \\ \omega_{mocap\_N} \end{bmatrix} \tag{4.16}$$

are the derivation of the least squares solution to the gyroscope equations. The matrix $\mathbf{x}$ represents the off diagonal misalignment terms, the scale terms, and the accelerometer bias for each axis.

4.3.3 Results

Results from the cross-axis gyroscope calibration were also reasonable. The figure below shows error in rates measured by the gyroscope and rates measured by the motion capture system.

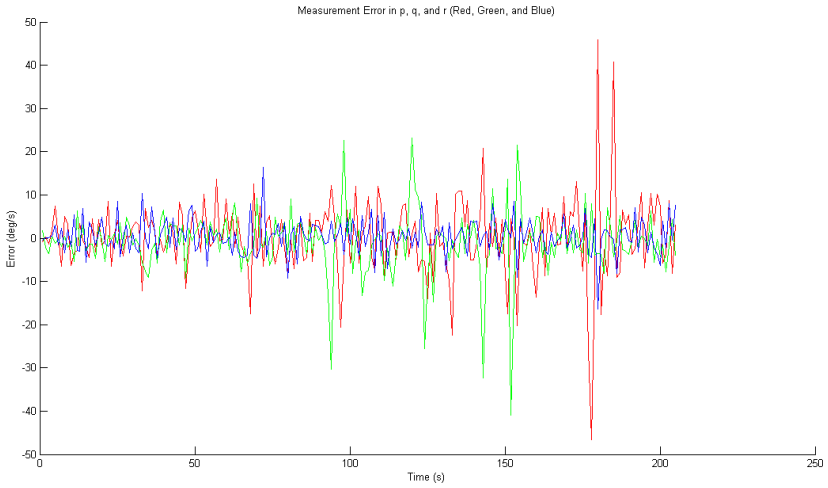

Figure 4.4: Cross Axis Gyroscope Calibration Results: Measurement Error in Angular Rates

The estimated angular velocities clearly model the true angular velocities measured by the motion captures system. There are a few measurement errors, when the aircraft moved more aggressively.

Chapter 5

Quadrotor State Estimation

State estimation is foundational for successful feedback control. Feedback control is necessary because the quadrotor is an inherently unstable platform [8]. Autonomous indoor navigation is generally unable to depend on GPS measurement for state estimation and must in turn use onboard sensors such as the inertial guidance unit and a 3D camera. Although many authors depend on known landmarks or artificial markers [27, 28], this is not an option in a real unknown operating environment.

There are two core parts to quadrotor state estimation using onboard sensors. The first component is high resolution estimate updates using the onboard IMU. These estimates drift over time, but provide fast feedback necessary for effective control. The second component is global updates, that when outdoors might be provided by GPS but indoors must be provided by an onboard sensor such as a 3D vision system.

Although some quadrotors have been successfully flown using only onboard sensors such as an IMU and 3D camera, they rely on globally consistent maps developed by techniques such as EKF SLAM and FastSLAM [32]. These mapping paradigms have been proven to become inconsistent as time passes [49, 50]. Other systems such as view-based mapping are able to remain locally consistent for the entire life span of an autonomous robot [51, 38].

5.1 Overview

The core algorithm used here for incorporating IMU and 3D pose information is the Extended Kalman Filter. The Extended Kalman Filter has the advantage of being reliable and robust but is not guaranteed to be optimal or consistent. It is also considerably faster than many other alternatives such as Monte Carlo estimation methods [48]. The Kalman Filtering approach described here uses a single prediction step driven by a constant

velocity model and two update steps - one for inertial unit measurement updates and one driven by 3D pose updates from the vision system. This is in contrast to the more common quadrotor estimation method where the prediction system is driven solely by the IMU and the measurement update is driven by the global transformation measurements[19, 16]. Below is an outline of the Extended Kalman Filter estimation algorithm.

In the Extended Kalman Filter the state belief, process noise, and measurement noise are all assumed to be normally distributed. The linear model restriction is relaxed by using the functions Jacobian matrices $\mathbf{F}$, $\mathbf{F_c}$, $\mathbf{H}$ to transform the covariance and calculate the Kalman gain $\mathbf{K}$. The vector $\mathbf{x}$ represents the current state estimate with the covariance matrix $\mathbf{P}$. The vector $\hat{\mathbf{x}}$ represents the predicted belief and $\hat{\mathbf{P}}$ represents the covariance of the predicted belief. If the functions were linear and the Kalman Filter were used, this estimator would be the minimum mean square-error estimator.

5.2 Extended Kalman Filter Algorithm

```
// prediction
x̂ = f(x, u, d_dt)
P̂ = F P F^T + F_c U F_c^T
// update
ẑ = h(x̂)
y = z − ẑ
S = H P̂ H^T + R
K = P̂ H^T S^−1
x = x̂ + K y
P = (I − K H) P̂
```

In order to use this algorithm, it is essential to calculate valid Jacobians and maintain filter consistency with realistic values for process and measurement noise. Due to the effects of non-linearities and violations of the Gaussian assumption, the process noise is artificially inflated.

5.3 Prediction Model

The prediction model is a constant velocity model that models how the quadrotor's twelve state variables change as a function of time, it's previous state, and current inputs. The state, x, being estimated by this estimator is the twelve state vector described previously in the models section. The input's to the system, u, are the inputs described in the high level model, but are ignored in the prediction function because the values tended to be inaccurate and the constant velocity model tended to be more accurate in experimentation. This prediction model could possibly could be enhanced using the improved accelerometer model published by [16]. The quadrotor's twelve state prediction model is

$$f(\mathbf{x}, \mathbf{u}, d_{dt}) = \begin{bmatrix} x + t_{dt}\dot{x} \\ y + t_{dt}\dot{y} \\ z + t_{dt}\dot{z} \\ u \\ v \\ w \\ \phi + t_{dt}\dot{\phi} \\ \theta + t_{dt}\dot{\theta} \\ \psi + t_{dt}\dot{\psi} \\ p \\ q \\ r \end{bmatrix} \tag{5.1}$$

where the quadrotor's linear and angular instantaneous velocities are assumed to remain constant. These velocities:

$$\begin{bmatrix} \dot{x} \\ \dot{y} \\ \dot{z} \end{bmatrix} = (R_{v\_b} \begin{bmatrix} u \\ v \\ w \end{bmatrix}) + \begin{bmatrix} x \\ y \\ z \end{bmatrix}, \tag{5.2}$$

$$\begin{bmatrix} \dot{\phi} \\ \dot{\theta} \\ \dot{\psi} \end{bmatrix} = T_{e\_b} \begin{bmatrix} p \\ q \\ r \end{bmatrix} \tag{5.3}$$

are derived by transforming u, v, p, q, and r into the vehicle frame.

The prediction Jacobians $\mathbf{F}$ and $\mathbf{F_c}$ are calculated by taking the derivative of the prediction function $f(\mathbf{x}, \mathbf{u}, d_{dt})$ with respect to the state vector x to produce a linear estimate for how the state changes as time progresses. Also, the second Jacobian is found by taking the derivative of the prediction function $f(\mathbf{x}, \mathbf{u}, d_{dt})$ with respect to the model's inputs u. This jacobian is all zeros due to the fact that we ignore the models inputs in this estimator's prediction function. The two equations:

$$\mathbf{F} = \frac{\partial f}{\partial \mathbf{x}}, \tag{5.4}$$

$$\mathbf{F_c} = \frac{\partial f}{\partial \mathbf{u}} \tag{5.5}$$

are the prediction Jacobians.

5.4 IMU Measurement Update Model

The IMU measurement update step incorporates information from accelerometer and gyroscope measurements into the state estimation. These values are measurements of p, q, r, ϕ, and psi provided by the calibrated IMU and are described in the previous section. However, the estimator can still filter these values given their known noise characteristics, and for that reason this measurement update is still valuable in the filter. This function is denoted h_{imu}. The function

$$h_{imu}(\mathbf{x}) = \begin{bmatrix} p_{imu} \\ q_{imu} \\ r_{imu} \\ \phi_{imu} \\ \theta_{imu} \end{bmatrix} \tag{5.6}$$

represents this measurement update model.

The Jacobian matrix for the IMU measurement update is calculated in the same way as the prediction model Jacobian. The matrix

$$\mathbf{H_{imu}} = \frac{\partial h_{imu}}{\partial \mathbf{x}} \tag{5.7}$$

defines the IMU measurement Jacobian. In this case the Jacobian is a trivial identity matrix due to the simplicity of them IMU measurement update model.

5.5 3D Vision Measurement Update Model

The 3D vision measurement update provides an estimate of the change in pose, linear velocity, and yaw. With IMU estimates of roll and pitch it forms a transformation estimate from the world reference frame to the UAV's body reference frame. When the quadrotor is navigating in reference to a single object like in this case, for simplicity we treat the object frame as a world reference frame. This assumption makes sense as long as the quadrotor navigates with respect to only one object, as in the case of the experiment and flight described later in this thesis. However, if the aircraft uses a frame based navigation technique, this assumption is still useful for control if the object frame is inertial and the down axis is still aligned with the world frame's down axis. This step incorporates information from the visual odometry into the state estimation. This function is denoted h_{vision}. Considering the quadrotor's navigation with respect to a single world-oriented reference frame at time make the vision measurement update trivial. This is a tremendous advantage of the Frame to Frame matching navigation technique for consistent realtime navigation by [52]. As long as the robot is estimating with respect to a single frame, the position and velocity of the robot can be directly measured with respect to an inertial reference frame in the visual scene is unchanging [52]. This gives consistent navigation with respect to single "world" reference frame and as long as those frames can be linked, the robot can navigate consistently over long periods of time. It is important to note however that with a changing inertial reference frame,

although valuable for navigation, dilutes the concept of a global position. The equation

$$h_v(\mathbf{x}) = \begin{bmatrix} x_{vision} \\ y_{vision} \\ z_{vision} \\ u_{vision} \\ v_{vision} \\ w_{vision} \\ \psi_{vision} + t_{dt}\dot{\psi} \end{bmatrix} \tag{5.8}$$

is a complete description of the trivial vision update function.

The Jacobian matrix for the vision measurement update is calculated in the same way as the prediction and IMU model Jacobian matrices. To define the Jacobian we have

$$\mathbf{H_{vision}} = \frac{\partial h_{vision}}{\partial \mathbf{x}} \tag{5.9}$$

where again this jacobian is similar to the identity matrix.

5.6 Results

Estimation results with rapid simulated vision updates were very accurate, as can be seen in the figure below. These simulated vision updates were recorded from the motion capture system then transformed into the quadrotor's camera frame. After being transformed into the camera frame, the measurements were corrupted with Gaussian noise with a standard deviation of .05 meters on the measurement's x, y and z elements, and 3 degrees on the camera measurement's ϕ, θ, and ψ elements. The measurements were then delayed by .20 seconds. These estimates were effective on multiple runs. The estimator also performed similarly with the quadrotor moving aggressively.

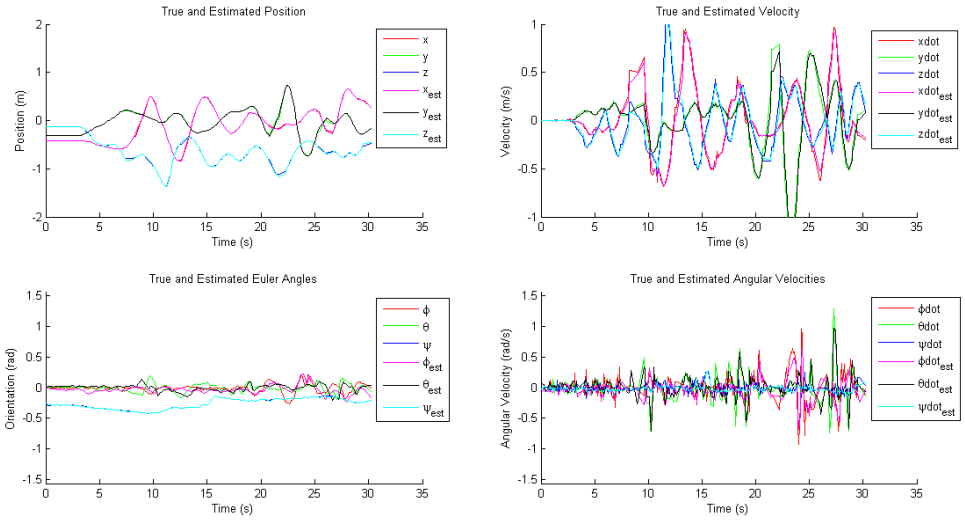

Figure 5.1: State Estimation Error With .20 Second Delayed Simulated Vision Updates

The next figure shows estimation results with vision further delayed by 1.0 second. The weakness of the constant velocity model is apparent in this figure. Notice the jagged edges when the quadrotor changes directions.

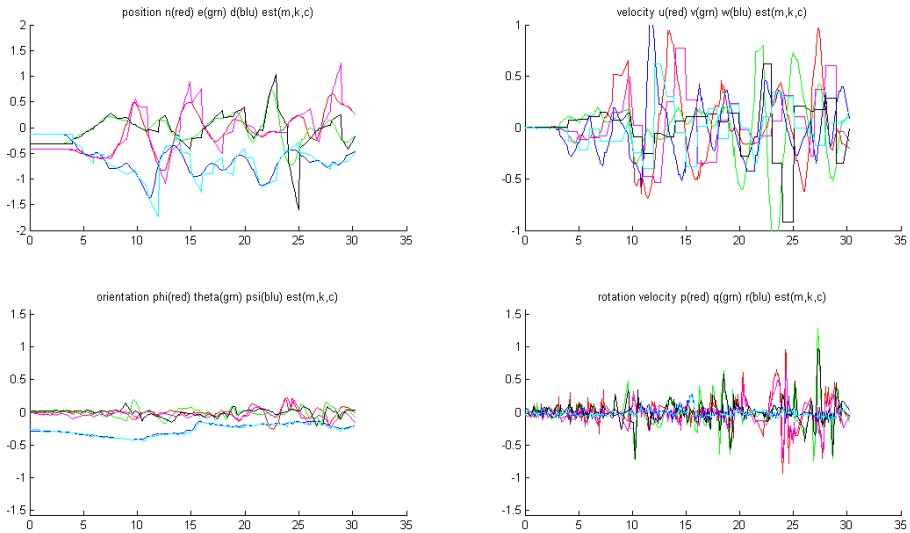

Figure 5.2: State Estimation Error With 1.0 Second Delayed Simulated Vision Updates

These estimates could be improved using out of sequence measurement updates. However, when visual processing takes around 20ms, the previous estimator was very functional. As a benchmark, Konolige showed that estimating the rigid body transformation given images from a 3D camera can be done in 16ms on a 2Ghz Pentium M processor [52].

It can be seen that when navigating with respect to a single reference frame, this estimation method is useful and allows for decent estimates. It may seem useless to make the assumptions that enable the estimator to treat a single frame as an inertial reference, but if the robot is able to switch views as it traverses world oriented inertial frames, the robot is able to travel great distances relying on locally consistent navigation with respect to a previously identified view. However, although this enables long term navigation and visual mapping it does not imply consistent and non-drifting metric "global" estimates" [51].

Chapter 6

Cooperative GPS Navigation

This chapter is derived from a collaborative paper written primarily by the thesis author [53]. Recently, the wide availability of GPS signals has enabled both aerial and ground-based autonomous vehicles to navigate in a variety of circumstances. Unfortunately, because the received GPS signal is extremely low power, the satellite signals can be easily obstructed. According to Misra and Enge, "The low signal power is the Achilles' heel of GPS, especially in military use. Even in civilian use, there is concern about the vulnerability of the signals." [40]. For this same reason, GPS receivers are also susceptible to jamming and unintentional interference [41]. An individual agent that relies on accurate navigation estimates has a substantial possibility for failure when operating in an environment where interference and occlusions are common.

Many future scenarios envision the use of multiple autonomous vehicles in collaboration to fulfill system objectives. In this thesis, we attempt to answer the question of whether the utilization of multiple vehicles can help to overcome the weak signal problem inherent in GPS.

In prior work, Mourikis and Roumeliotis [44] demonstrate in their analysis of multi-robot cooperative localization that, "if one robot has access to absolute position measurements, the positioning uncertainty of all the robots in the group remains bounded and converges to a constant value." Note, however, that this work specifically assumes that one robot in the system is able to localize itself without any information from other agents. In addition, the localization accuracy of the entire system will be bounded by the accuracy of that individual agent's localization estimate. In other prior works [42, 43, 44, 45, 46], it was demonstrated that in the absence of a global positioning system, the drift in navigation error can be significantly reduced by jointly estimating the navigation state of vehicles in the sys-

tem. Not that the final result of all these papers was not a bound on overall navigation error, but rather a decrease in the growth rate of navigational error. In [54] a system for ensuring bounded navigation error over time was introduced. This system, however, assumed the presence of visual observations between agents in the network and at least two landmarks of known location.

Note that in contrast to prior works, this thesis presents a solution that enables drift-free navigation. In addition, if multiple vehicles within the network can view different GPS satellites, the accuracy of the system estimate will be better than any individual location estimate would provide. Finally, our system enables global location estimates in situations when no individual agent is able to estimate location. This increased performance is accomplished by sharing low-level pseudo-range information about GPS signals among the system's vehicles, rather than high-level estimates. In the following sections, we describe the mathematical algorithm used to enable cooperative GPS navigation, followed by the experimental results achieved.

6.1 Cooperative GPS Algorithm

6.1.1 Overview

For our algorithm to improve navigational accuracy, three conditions must be met. First, agents within the system must be able to observe the relative positions of other agents. Second, agents must be able to communicate both relative position estimates and pseudorange measurements from visible GPS satellites with other agents. Lastly, the set of independent measurements in the system must be greater than or equal to the set of unknowns in the system. Under these conditions our algorithm can form a "cooperative GPS receiver."

6.1.2 Detailed Description

Our cooperative GPS algorithm uses a least-means squared estimation routine based on the Newton-Raphson method. The inputs to the cooperative receiver are the relative position measurements between agents and the pseudorange measurements of all agents connected in the group. The estimation algorithm solves for the location and clock bias of

each agent that best fits the measurements received and the relative position of the agents. Following is a detailed description of the algorithm.

The information from a local agent l and the information from n useful neighboring agents form a system of equations and unknowns. If the set of independent measurements $\boldsymbol{M}$ has greater cardinality than the set of unknowns $\boldsymbol{U}$, a solution to the system can found.

The set of measurements $\boldsymbol{M}$ is made up of independent pseudorange measurements from the local agent (P_{local}), pseudorange measurements from neighboring agents ($\mathrm{P}_{neighbors}$), as well as relative measurements between each neighbor and the local agent ($\mathbf{R}$). All relative measurements are considered independent. A satellite pseudorange measurement is independent if it is from a satellite not already observed by the system. The new satellite also needs to form a good geometry with the user and the satellite constellation [40].

The set of unknowns $\boldsymbol{U}$ is made up of the local agent's state and the useful neighboring agent's states. Each state consists of four values: the agent's GPS receiver position in three dimensions and the GPS receiver's clock bias. A solution to the system can be estimated if:

$$\mathbf{U} = \mathbf{X}_{local} \cup \mathbf{X}_{remote}, \tag{6.1}$$

$$\mathbf{M} = \mathrm{P}_{local} \cup \mathrm{P}_{neighbors} \cup \mathbf{R}, \tag{6.2}$$

$$|\mathbf{M}| \geq |\mathbf{U}| \tag{6.3}$$

where ($\mathbf{X}_{local}$) is the local agent's state estimate and ($\mathbf{X}_{remote}$) is the estimate of all the other agents' locations. If this restriction is satisfied, an estimation of the local agent's position can be calculated simultaneously with an estimation of each useful neighbor's position. Below, we define the basic equations used to solve for the locations of each agent in the system.

Let $\mathbf{x}^{(s)}$be the true position of a satellite s. Let $\mathbf{x}^{(l)}$ represent the local agent position and $b^{(l)}$ the local agent's GPS receiver clock bias. Also let $\mathbf{x}^{(n)}$ and $b^{(n)}$ be a neighboring agent's position and clock bias respectively. The term $\rho_c^{(s,*)}$ denotes a corrected pseudorange measurement from a satellite s to an agent (either the local agent or a neighbor agent)

$$\rho_c^{(s,*)} = \left\|\mathbf{x}^{(s)} - \mathbf{x}^{(*)}\right\| + b^{(*)}, \tag{6.4}$$

$$\rho_c^{(s,*)} = \sqrt{\left(x^{(s)} - x^{(*)}\right)^2 + \left(y^{(s)} - y^{(*)}\right)^2 + \left(z^{(s)} - z^{(*)}\right)^2} + b^{(*)}. \tag{6.5}$$

The term $\mathbf{p}^{(l,n)}$ is the measured relative position between the local agent l and a neighbor agent n. This measurement,

$$\mathbf{p}^{(l,n)} = \mathbf{x}^{(l)} - \mathbf{x}^{(r)}, \tag{6.6}$$

$$\mathbf{p}^{(l,r)} = \begin{bmatrix} x^{(l)} \\ y^{(l)} \\ z^{(l)} \end{bmatrix} - \begin{bmatrix} x^{(r)} \\ y^{(r)} \\ z^{(r)} \end{bmatrix} \tag{6.7}$$

must be taken in the global reference frame.

Due to the square root term in the pseudo-range equation, the estimates of agent locations and clock bias must be performed using a non-linear optimization procedure. For our algorithm, we have used a Gauss-Newton based optimization procedure. To initialize Gauss-Newton, an initial estimate is required. This initial estimate should be the best estimate of the agent's positions. Typically, if a previous estimate of the location is known, that is used. Otherwise, the center of the earth is used.

Given an initial estimate, the Gauss-Newton solver will repeatedly linearize the equations, solve for the best linear solution to obtain a correction, and add these corrections to the current estimate. The corrections can be expressed as:

$$\delta\mathbf{x}^{(*)} = \begin{bmatrix} \delta x^{(*)} \\ \delta y^{(*)} \\ \delta z^{(*)} \end{bmatrix} = \mathbf{x}^{(*)} - \mathbf{x}_0^{(*)}, \tag{6.8}$$

$$\delta b^{(*)} = b^{(*)} - b_0^{(*)} \tag{6.9}$$

where x_0 is the current estimate.

To solve for the corrections, the change in computed pseudorange must be found. The expressions

$$\delta\rho^{(s,*)}=\rho_c^{(s,*)}-\rho_0^{(s,*)}, \tag{6.10}$$

$$\delta\rho^{(s,*)} = \left\|\mathbf{x}^{(s)}-\mathbf{x}^{(*)}\right\| - \left\|\mathbf{x}^{(s)}-\mathbf{x}_0^{(*)}\right\| + \delta b^{(*)} \tag{6.11}$$

express a change in pseudorange. These equations

$$\delta\rho^{(s,*)} \approx \frac{\left(\mathbf{x}^{(s)}-\mathbf{x}_0^{(*)}\right)}{\left\|\mathbf{x}^{(s)}-\mathbf{x}_0^{(*)}\right\|}\delta\mathbf{x}^{(*)} + \delta b^{(*)} \tag{6.12}$$

are linearized about the current estimate.

Note that the linearized pseudorange equation contains the linear approximation to a vector norm directed from a satellite s to an estimated agent position multiplied by the correction term $\delta\mathbf{x}^{(*)}$. The derivation:

$$\delta\mathbf{p}^{(l,n)}=\mathbf{p}^{(l,r)}-\mathbf{p}_0^{(l,n)}, \tag{6.13}$$

$$\delta\mathbf{p}^{(l,n)}=\begin{bmatrix}\delta x^{(l,n)}\\ \delta y^{(l,n)}\\ \delta z^{(l,n)}\end{bmatrix}=\delta\mathbf{x}^{(l)}-\delta\mathbf{x}^{(n)} \tag{6.14}$$

expresses the already linear relative measurement equation.

The system of linear equations can be written in matrix form and solved using least squares. For purposes of explanation, we will used a simple example involving the local agent with pseudorange measurements to three satellites and a single neighboring agent with pseudorange measurements to two satellites. This example can be generalized to include any number of neighboring agents. The equations:

$$\delta\rho = \begin{bmatrix} -\mathbf{1}^{(\mathbf{s,l})^{\mathbf{T}}} & 1 & \mathbf{0} & 0 \\ \mathbf{0} & 0 & -\mathbf{1}^{(\mathbf{s,r})^{\mathbf{T}}} & 1 \\ \mathbf{I} & 0 & -\mathbf{I} & 0 \end{bmatrix} \begin{bmatrix} \delta\mathbf{x}^{(l)} \\ \delta b^{(l)} \\ \delta\mathbf{x}^{(r)} \\ \delta b^{(r)} \end{bmatrix}, \tag{6.15}$$

$$where\ -\mathbf{1}^{(\mathbf{s,r})} = -\frac{\left(\mathbf{x}^{(s)} - \mathbf{x}_0^{(r)}\right)}{\left\|\mathbf{x}^{(s)} - \mathbf{x}_0^{(r)}\right\|}, \tag{6.16}$$

$$\mathbf{G} = \begin{bmatrix} -\mathbf{1}^{(\mathbf{s,l})^{\mathbf{T}}} & 1 & \mathbf{0} & 0 \\ \mathbf{0} & 0 & -\mathbf{1}^{(\mathbf{s,r})^{\mathbf{T}}} & 1 \\ \mathbf{I} & 0 & -\mathbf{I} & 0 \end{bmatrix} \tag{6.17}$$

represent these values linearized.

Using the matrix $\mathbf{G}$, the derivation

$$\begin{bmatrix} \delta\rho \\ \delta\mathbf{p} \end{bmatrix} = \mathbf{G} \begin{bmatrix} \delta\mathbf{x}^{(l)} \\ \delta b^{(l)} \\ \delta\mathbf{x}^{(n)} \\ \delta b^{(n)} \end{bmatrix}, \tag{6.18}$$

$$\begin{bmatrix} \delta\mathbf{x}^{(l)} \\ \delta b^{(l)} \\ \delta\mathbf{x}^{(n)} \\ \delta b^{(n)} \end{bmatrix} = \left(\mathbf{G}^T\mathbf{G}\right)^{-1}\mathbf{G}^T \begin{bmatrix} \delta\rho \\ \delta\mathbf{p} \end{bmatrix}, \tag{6.19}$$

$$\begin{bmatrix} \delta\widehat{\mathbf{x}}^{(l)} \\ \delta\hat{b}^{(l)} \\ \delta\widehat{\mathbf{x}}^{(n)} \\ \delta\hat{b}^{(n)} \end{bmatrix} = \begin{bmatrix} \mathbf{x}_0^{(l)} \\ b_0^{(l)} \\ \mathbf{x}_0^{(n)} \\ b_0^{(n)} \end{bmatrix} + \begin{bmatrix} \delta\mathbf{x}^{(l)} \\ \delta b^{(l)} \\ \delta\mathbf{x}^{(n)} \\ \delta b^{(n)} \end{bmatrix} \tag{6.20}$$

finds the correction term. The process is repeated until the length of $\delta\mathbf{x}^{(l)}$ is within a desired distance threshold.

6.2 Experimental Results

To demonstrate the feasibility of the cooperative GPS receiver, we utilized two computers equipped with a Novatel GPS receiver. Each receiver provided raw pseudorange and ephemeris measurements to an agent. The systems were placed in surveyed locations where the view of the GPS constellation was naturally or artificially obscured. Each system, or agent, was at least 100 meters apart and remained stationary during the experiments. Relative position measurements were created artificially from the known locations of both receivers.

The following three figures show that the cooperative solution consistently outperformed the non-cooperative solution. Each figure compares the performance of the cooperative and non-cooperative algorithms using estimate position error in the earth-centered, earth-fixed reference frame. The non-cooperative algorithm estimated position using Gauss-Newton and Least Squares, but did not use any information about or from other agents. The cooperative agent used the algorithm above in collaboration with one other connected agent.

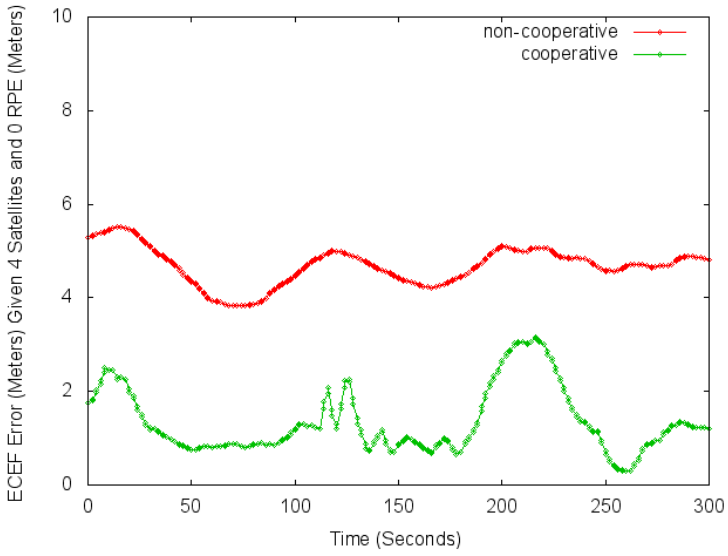

Figure 6.1: Cooperative and Non-Cooperative Estimation Error Four Satellites

Figure 1 shows the results of an experiment where each agent is only able to take measurements from four satellites in the GPS constellation. In the experiment the cooperative algorithm is on average 3.33 meters more accurate than the non-cooperative algorithm.

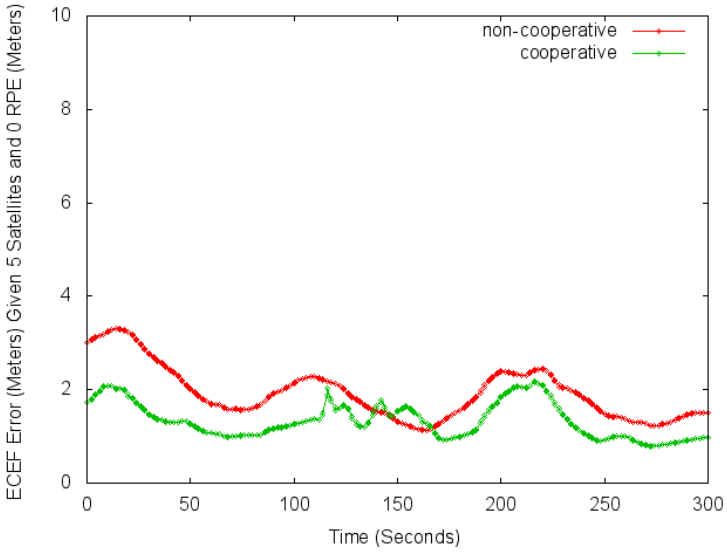

Figure 6.2: Cooperative and Non-Cooperative Estimation Error Five Satellites

Figure 2 shows results from a similar experiment where each agent was able to receive measurements from a maximum of five satellites. The difference between average error of the cooperative solution and the average error of the non-cooperative solution was 1.34 meters. With more satellites the non-cooperative solution was able to form a better estimate.

The last figure, Figure 3, shows results from an experiment where each agent was able to see up to eight satellites. Figure 3 shows that the cooperative algorithm performs only slightly better than the non-cooperative solution when signals from most satellites are unobstructed. The average error between the two algorithms was only 0.11 meters. Further experiments showed that this trend continues as each agent obtains a full view of the satellite constellation.

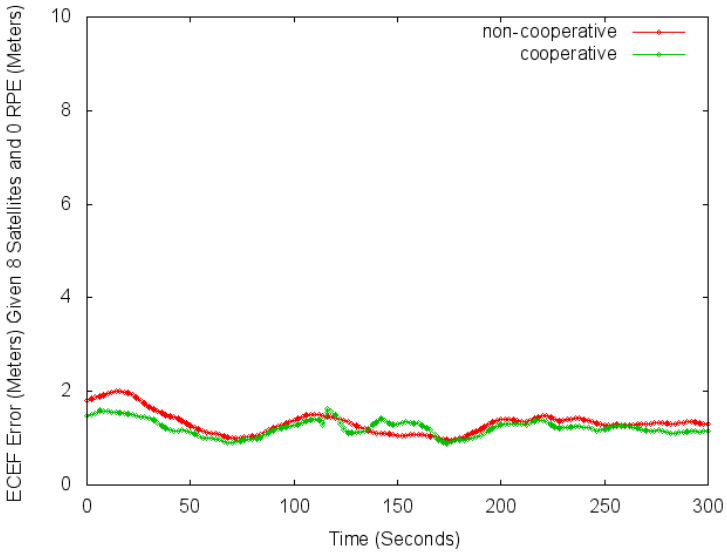

Figure 6.3: Cooperative and Non-Cooperative Estimation Error Eight Satellites

We also conducted an experiment where one agent was only able to receive measurements from two satellites, and a neighboring agent was able to receive measurements from three satellites. When either agent operated alone, it was unable to form position estimates. However, by forming a cooperative GPS receiver with both agents, a location estimate was obtained from the GPS signals. On average, both agents estimated their position with an error of less than five meters. Estimates were comparable to non-cooperative solutions where an agent received measurements from only four satellites. This significant result from our experiment demonstrated that cooperative navigation provided a mechanism for agents that would have otherwise been lost to localize and continue mobile operation.

6.3 Analysis

In each experiment, cooperative estimation yielded more accurate estimates than individual estimation routines. Of particular interest was the case where the agents were operating in an environment with multiple and varied occlusions to the GPS constellation. This caused each agent to observe a different set of satellites. If each agent has unique information to contribute to the system, the cooperative solution was significantly more accurate than an individual solution. In some cases, agents that were unable to estimate

position individually were able to accurately estimate positions when working cooperatively. In this case, the cooperative GPS receiver allowed for a global position estimate that was unachievable using individual receivers. This demonstrates that a cooperative GPS solution can overcome many of the weaknesses found in conventional GPS navigation.

In the future, we would like to relax several of the underlying assumptions of our work in this thesis. Our algorithm currently assumes direct connections between all agents in the system and good relative position estimates. In the future, we would like to determine the proper method to integrate inaccurate relative position estimates with distributed pseudo-range measurements.

Chapter 7

Conclusions and Future Work

7.1 Conclusions

This thesis explored topics relevant to navigation and control of a small indoor unmanned aerial vehicle. The topics discussed included building blocks for autonomous navigation and view control for a quadrotor robot. Two quadrotor models, a high level and lower level model were discussed in detail. Sensor models for onboard sensors relevant to indoor navigation were also described in detail. Extensions to current work in the area of quadrotor model parameter estimation were discussed. A specific contribution is work on estimation quadrotor model parameters using an SIR particle filter. State estimation, a prerequisite to feedback control was also covered. The observer or estimator was designed using an Extended Kalman Filter. The final chapter covered work in cooperative GPS navigation. This was demonstrated to be useful and at times more accurate than non-cooperative nagivation. One other noteworthy value of this thesis is the literature review and inclusion of major topics relating to indoor quadrotor simulation, navigation, and estimation.

7.2 Future Work

Future work would include the refinement of quadrotor parameter estimation through optimization and consideration of an entire trial flight in each iteration of the estimator. A stable and reliable quadrotor platform would enable further development by providing reliability constraints and guarantees. Refinement of hardware and software systems has the potential to smooth out estimator development and error. A final area of future work would be to extend the estimator to operate using a large map with multiple views instead of maintaining view on a single object of interest. This would enable long term navigation in GPS denied areas.

Bibliography

[1] S. Bouabdallah, P. Murrieri, and R. Siegwart, "Design and control of an indoor micro quadrotor," *IEEE International Conference on Robotics and Automation, 2004. Proceedings. ICRA '04. 2004*, pp. 4393–4398 Vol.5, 2004. [Online]. Available: http://ieeexplore.ieee.org/lpdocs/epic03/wrapper.htm?arnumber=1302409 2, 4, 6

[2] C. Chamberlain, "System Identification, State Estimation, and Control of Unmanned Aerial Robots," Ph.D. dissertation, Brigham Young University, 2011. 2, 3, 4, 20, 24, 28

[3] M. Schmidt, "Simulation and Control of a Quadrotor Unmanned Aerial Vehicle," Ph.D. dissertation, 2011. [Online]. Available: http://uknowledge.uky.edu/gradschool_theses/93/ 2

[4] R. Beard, "Quadrotor dynamics and control," 2008. [Online]. Available: http://quad08topgun.groups.et.byu.net/documents/quadrotor_2_20_2008.pdf 2, 6, 7, 20, 25, 31

[5] R. W. Beard and T. W. Mclain, *Small unmanned aircraft: Theory and practice*. Princeton University Press, 2012. 2, 12, 20, 24

[6] E. Shin and N. El-Sheimy, *Accuracy improvement of low cost INS/GPS for land applications*, 2001, no. 20156. [Online]. Available: http://www.ucalgary.ca/engo_webdocs/NES/01.20156.EHShin.pdf 2, 3, 7

[7] G. Hoffmann, H. Huang, S. Waslander, and C. Tomlin, "Quadrotor helicopter flight dynamics and control: Theory and experiment," in *Proc. of the AIAA Guidance, Navigation, and Control Conference*, no. August, 2007, pp. 1–20. [Online]. Available: http://www.mastermindcn.com/papers/Hoffmann,Tomlin_2007_QuadrotorHelicopterFlightDynamicsandControlTheoryandExperiment.pdf 2, 4, 12, 31

[8] P. Pounds, R. Mahony, J. Gresham, P. Corke, and J. Roberts, "Towards dynamically-favourable quad-rotor aerial robots," in *Proceedings of the 2004 Australasian Conference on Robotics \& Automation*, 2004. [Online]. Available: http://eprints.qut.edu.au/33833/1/33833.pdf 2, 3, 20, 40

[9] P. Pounds, R. Mahony, and P. Corke, "Modelling and control of a quad-rotor robot," in *Proceedings Australasian Conference on Robotics and Automation 2006*, 2006. [Online]. Available: http://eprints.qut.edu.au/33767 2, 4

[10] R. Goela, S. Shahb, N. Guptac, and N. Ananthkrishnanc, "Modeling, Simulation and Flight Testing of an Autonomous Quadrotor," *Proceedings of ICEAE*, 2009. [Online]. Available: https://multi-rotor-uav.googlecode.com/files/A7D40060d01.pdf 2, 3

[11] E. Shin and N. El-Sheimy, "An unscented Kalman filter for in-motion alignment of low-cost IMUs," in *Position Location and Navigation Symposium, 2004. PLANS 2004*, 2004, pp. 273—-279. [Online]. Available: http://ieeexplore.ieee.org/xpls/abs_all.jsp?arnumber=1309005 2, 3

[12] P. Zhang, J. Gu, E. Milios, and P. Huynh, "Navigation with IMU/GPS/digital compass with unscented Kalman filter," in *Mechatronics and Automation, 2005 IEEE International Conference*, no. July, 2005, pp. 1497–1502. [Online]. Available: http://ieeexplore.ieee.org/xpls/abs_all.jsp?arnumber=1626777 2, 4

[13] S. Sukkarieh, E. Nebot, and H. F. Durrant-Whyte, "A high integrity IMU/GPS navigation loop for autonomous land vehicle applications," *Robotics and Automation, IEEE Transactions on*, vol. 15, no. 3, pp. 572–578, 1999. [Online]. Available: http://ieeexplore.ieee.org/xpls/abs_all.jsp?arnumber=768189 2

[14] M. Achtelik, A. Bachrach, R. He, S. Prentice, and N. Roy, "Autonomous navigation and exploration of a quadrotor helicopter in gps-denied indoor environments," *IEEE ICRA. http://iarc. angel-strike. com/oldauvs/5th_mission/2009SymposiumPapers/2009MIT. pdf*, 2009. 2

[15] R. Leishman, J. Macdonald, R. W. Beard, and T. W. Mclain, "Accelerometers & Observers : Improving Quadrotor State Estimation," *Robotics and Automation Magazine*, p. Submitted. 2, 4

[16] J. Macdonald, R. Leishman, R. Beard, and T. Mclain, "Improved IMU-Based Estimation To Enable Indoor Flight." 2, 3, 4, 24, 41, 42

[17] E. Trucco and A. Verri, *Introductory techniques for 3-D computer vision*, 1998. [Online]. Available: http://cseweb.ucsd.edu/classes/fa09/cse252a/trucco_verri_pp178-194.pdf 2, 3

[18] J. Smisek, M. Jancosek, and T. Pajdla, "3D with Kinect," *2011 IEEE International Conference on Computer Vision Workshops (ICCV Workshops)*, pp. 1154–1160, Nov. 2011. [Online]. Available: http://ieeexplore.ieee.org/lpdocs/epic03/wrapper.htm?arnumber=6130380 2, 3, 26

[19] M. Veth and J. Raquet, "Two-Dimensional Stochastic Projections for Tight Integration of Optical and Inertial Sensors for Navigation," *Proceedings of the Institute of Navigation National Technical Meeting*, pp. 587–596, 2006. [Online]. Available: http://oai.dtic.mil/oai/oai?verb=getRecord&metadataPrefix=html&identifier=ADA462963 3, 4, 41

[20] STElectronics, "Application note Tilt measurement using a low-g 3-axis accelerometer," 2010. 3, 26, 28, 35

[21] ——, "Technical article Everything about STMicroelectronics 3-axis digital MEMS," no. July, 2011. 3, 28, 35

[22] J. Rios and E. White, "Fusion filter algorithm enhancements for a MEMS GPS/IMU," *ION NTM*, 2002. [Online]. Available: http://www.devisersoftware.com/down/20114229.pdf 3

[23] J. Carpenter, P. Clifford, and P. Fearnhead, "Improved particle filter for nonlinear problems," *Radar, Sonar and Navigation, IEE Proceedings-*, vol. 146, no. 1, pp. 2–7, 1999. [Online]. Available: http://ieeexplore.ieee.org/xpls/abs_all.jsp?arnumber=768732 3, 28

[24] A. Giremus, A. Doucet, V. Calmettes, and J. Y. Tourneret, "A Rao-Blackwellized particle filter for INS/GPS integration," in *Acoustics, Speech, and Signal Processing, 2004. Proceedings.(ICASSP'04). IEEE International Conference on*, vol. 3, 2004, pp. iii—-964. [Online]. Available: http://ieeexplore.ieee.org/xpls/abs_all.jsp?arnumber=1326707 3, 4, 28

[25] D. Mellinger, N. Michael, and V. Kumar, "Trajectory generation and control for precise aggressive maneuvers with quadrotors," *The International Journal of Robotics Research*, Jan. 2012. [Online]. Available: http://ijr.sagepub.com/cgi/doi/10.1177/0278364911434236 4

[26] A. Mohamed and K. Schwarz, "Adaptive Kalman filtering for INS/GPS," *Journal of Geodesy*, no. September 1998, pp. 193–203, 1999. [Online]. Available: http://www.springerlink.com/index/D78FT4M70HQ7E49R.pdf 4

[27] S. Ahrens, D. Levine, G. Andrews, and J. P. How, "Vision-based guidance and control of a hovering vehicle in unknown, GPS-denied environments," *Proceedings of the IEEE International Conference on Robotics and Automation (2009)*, pp. 2643–2648, 2009. [Online]. Available: http://ieeexplore.ieee.org/lpdocs/epic03/wrapper.htm?arnumber=5152680 4, 40

[28] G. Tournier, M. Valenti, J. How, and J. Feron, "Estimation and control of a quadrotor vehicle using monocular vision and moire patterns," in *AIAA Guidance, Navigation and Control Conference and Exhibit*, no. August, 2006, pp. 21—-24. [Online]. Available: http://citeseerx.ist.psu.edu/viewdoc/download?doi=10.1.1.86.517&rep=rep1&type=pdf 4, 40

[29] L. M. Paz, P. Pinies, J. D. Tardos, and J. Neira, "Large-Scale 6-DOF SLAM With Stereo-in-Hand," *IEEE Transactions on Robotics*, vol. 24, no. 5, pp. 946–957, 2008. [Online]. Available: http://ieeexplore.ieee.org/lpdocs/epic03/wrapper.htm?arnumber=4633683 4

[30] P. Lamon and R. Siegwart, "Inertial and 3d-odometry fusion in rough terrain-towards real 3d navigation," in *Intelligent Robots and Systems, 2004.(IROS 2004). Proceedings. 2004 IEEE/RSJ International Conference on*, 2004, pp. 1716—-1721. [Online]. Available: http://ieeexplore.ieee.org/xpls/abs_all.jsp?arnumber=1389643 4

[31] R. Leishman, J. Macdonald, S. Quebe, J. Ferrin, R. Beard, and T. Mclain, "Utilizing an Improved Rotorcraft Dynamic Model in State Estimation," *IEEERSJ International Conference on Intelligent Robots and Systems*, 2011. 4, 12

[32] A. Huang and A. Bachrach, "Visual odometry and mapping for autonomous flight using an RGB-D camera," in *International Symposium on Robotics Research (ISRR)*, 2011. [Online]. Available: http://www.cs.washington.edu/research/projects/aiweb/media/papers/Huang-ISRR-2011.pdf 4, 40

[33] H. Bay, A. Ess, T. Tuytelaars, and L. Vangool, "Speeded-Up Robust Features (SURF)," *Computer Vision and Image Understanding*, vol. 110, no. 3, pp. 346–359, 2008. [Online]. Available: http://linkinghub.elsevier.com/retrieve/pii/S1077314207001555 4

[34] M. Fischler and R. Bolles, "Random sample consensus: a paradigm for model fitting with applications to image analysis and automated cartography," *Communications of the ACM*, vol. 24, no. 6, 1981. [Online]. Available: http://dl.acm.org/citation.cfm?id=358692 4

[35] B. K. P. Horn, H. M. Hilden, and S. Negahdaripour, "Closed-form solution of absolute orientation using orthonormal matrices," *Journal of the Optical Society of America A*, vol. 5, no. 7, p. 1127, 1988. [Online]. Available: http://www.opticsinfobase.org/abstract.cfm?URI=josaa-5-7-1127 4

[36] A. Lorusso, D. W. Eggert, and R. B. Fisher, "A Comparison of Four Algorithms for 1 Estimating 3-D Rigid Transformations," *Proceedings of the 1995 British conference on Machine vision Vol 1*, pp. 237–246, 1995. [Online]. Available: http://citeseerx.ist.psu.edu/viewdoc/download?doi=10.1.1.56.9872&rep=rep1&type=pdf 4

[37] D. Nistér, O. Naroditsky, and J. Bergen, "Visual odometry," in *Computer Vision and Pattern Recognition, 2004. CVPR 2004. Proceedings of the 2004 IEEE Computer Society Conference on*, 2004, pp. I—-652. [Online]. Available: http://ieeexplore.ieee.org/xpls/abs_all.jsp?arnumber=1315094 4

[38] K. Konolige, J. Bowman, J. D. Chen, P. Mihelich, M. Calonder, V. Lepetit, and P. Fua, "View-based Maps," *The International Journal of Robotics Research*, vol. 29, no. 8, pp. 941–957, 2010. [Online]. Available: http://ijr.sagepub.com/cgi/doi/10.1177/0278364910370376 4, 40

[39] K. Konolige and M. Agrawal, "FrameSLAM: From Bundle Adjustment to Real-Time Visual Mapping," *IEEE Transactions on Robotics*, vol. 24, no. 5, pp. 1066–1077, 2008. [Online]. Available: http://ieeexplore.ieee.org/lpdocs/epic03/wrapper.htm?arnumber=4648456 4

[40] P. Misra and P. Enge, *Global Positioning System: Signals, Measurements, and Performance*. Lincoln, MA: Ganga-Jamuna Press, 2001. 4, 48, 50

[41] E. D. Kaplan and C. Hegarty, *Understanding GPS: Principles and Applications, Second Edition*, E. D. Kaplan and C. Hegarty, Eds. Artech House Publishers, 2005. [Online]. Available: http://www.amazon.com/dp/1580538940 5, 48

[42] A. Bahr, J. J. Leonard, and M. F. Fallon, "Cooperative Localization for Autonomous Underwater Vehicles," *The International Journal of Robotics Research*, vol. 28, no. 6, pp. 714–728, 2009. [Online]. Available: http://ijr.sagepub.com/cgi/doi/10.1177/0278364908100561 5, 48

[43] J. W. Fenwick, P. M. Newman, and J. J. Leonard, "Cooperative concurrent mapping and localization," pp. 1810–1817, 2002. [Online]. Available: http://ieeexplore.ieee.org/lpdocs/epic03/wrapper.htm?arnumber=1014804 5, 48

[44] A. I. Mourikis and S. I. Roumeliotis, "Performance analysis of multirobot Cooperative localization," pp. 666–681, 2006. [Online]. Available: http://ieeexplore.ieee.org/lpdocs/epic03/wrapper.htm?arnumber=1668252 5, 48

[45] A. C. Sanderson, "A distributed algorithm for cooperative navigation among multiple mobile robots," *Advanced Robotics*, vol. 12, no. 4, pp. 335–349, 1998. 5, 48

[46] R. Sharma and C. N. Taylor, "Vision Based Distributed Cooperative Navigation for MAVs in GPS denied areas," *AIAA Infotech*, 2009. 5, 48

[47] C. V. Nguyen, S. Izadi, and D. Lovell, "Modeling Kinect Sensor Noise for Improved 3D Reconstruction and Tracking," *2012 Second International Conference on 3D Imaging, Modeling, Processing, Visualization & Transmission*, pp. 524–530, Oct. 2012. [Online]. Available: http://ieeexplore.ieee.org/lpdocs/epic03/wrapper.htm?arnumber=6375037 27

[48] S. Thrun, W. Burgard, D. Fox, and Others, *Probabilistic robotics*. MIT press Cambridge, MA, 2005, vol. 1. 28, 29, 40

[49] T. Bailey, J. Nieto, and E. Nebot, "Consistency of the FastSLAM algorithm," *Proceedings 2006 IEEE International Conference on Robotics and Automation 2006 ICRA 2006*, no. 1, pp. 424–429, 2006. [Online]. Available: http://ieeexplore.ieee.org/lpdocs/epic03/wrapper.htm?arnumber=1641748 40

[50] S. J. Julier and J. K. Uhlmann, "A counter example to the theory of simultaneous localization and map building," *Proceedings 2001 ICRA IEEE International Conference on Robotics and Automation Cat No01CH37164*, vol. 4, no. 1, pp. 4238–4243, 2001. [Online]. Available: http://ieeexplore.ieee.org/lpdocs/epic03/wrapper.htm?arnumber=933280 40

[51] K. Konolige and J. Bowman, "Towards lifelong visual maps," *2009 IEEERSJ International Conference on Intelligent Robots and Systems*, pp. 1156–1163, 2009. [Online]. Available: http://ieeexplore.ieee.org/lpdocs/epic03/wrapper.htm?arnumber=5354121 40, 47

[52] K. Konolige and M. Agrawal, "Frame-Frame Matching for Realtime Consistent Visual Mapping," *Proceedings 2007 IEEE International Conference on Robotics and Automation*, no. April, pp. 2803–2810, 2007. [Online]. Available: http://ieeexplore.ieee.org/lpdocs/epic03/wrapper.htm?arnumber=4209514 44, 47

[53] S. Quebe, J. Campbell, S. DeVilbiss, and C. Taylor, "Cooperative GPS navigation," in *Position Location and Navigation Symposium (PLANS), 2010 IEEEION*, May 2010, pp. 834–837. 48

[54] R. Sharma and C. Taylor, "Cooperative navigation of MAVs In GPS denied areas," pp. 481–486, 2008. [Online]. Available: http://ieeexplore.ieee.org/lpdocs/epic03/wrapper.htm?arnumber=4648041 49