EMBEDDED SYSTEMS AND NETWORKING APPROACH TO GUIDANCE AND ESTIMATION

A Thesis

by

DANIEL GONÇALVES GHAN

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

MASTER OF SCIENCE

Chair of Committee,	Manoranjan Majji
Committee Members,	John E. Hurtado
	Daren B. H. Cline
Head of Department,	Rodney Bowersox

August 2019

Major Subject: Aerospace Engineering

ABSTRACT

Networked embedded systems are valuable tools for guidance and navigation, with applications ranging from UAV swarms to self-driving cars. LASR Lab is working on projects that require networked embedded systems running Kalman filters, such as a smart tower that will send an alert over the Internet if its embedded system detects structural changes in the tower. To demonstrate the architecture that will support these projects, a demonstration has been made consisting of a microcomputer and an Inertial Measurement Unit contained in a football-shaped shell. This football is designed to stream its position and orientation over a network in real time. Like other LASR Lab projects, the football utilizes a Tinker Board microcomputer and a VectorNav IMU. Data are streamed in real time to a controller laptop over a Wi-Fi connection using the Open MPI protocol. An Extended Kalman Filter and an Unscented Kalman Filter for estimating position, velocity, orientation, and gyroscope biases were developed and implemented in C++, along with calibration routines for estimating initial conditions and noise parameters. Due to a lack of reliable measurements and mathematical bugs, the filters were not successful in estimating position, velocity, or attitude; but the hardware, software, and networking architecture was demonstrated successfully and can be used in future projects.

ACKNOWLEDGMENTS

Thank you to my advisor, Dr. Majji; and my committee members, Dr. Hurtado and Dr. Cline, for their guidance and support of this research.

Thank you to Lisa Brown and Dr. White at the Low-Speed Wind Tunnel for allowing me to use their 3D printer at a reduced cost.

Thank you to all my friends who provided moral support and helped me enjoy my time as a graduate student. In particular, thank you to Robert, for helping me build the shell of the demonstration described in this thesis; to Nathan, for being the one friend who showed up to my defense; and to my girlfriend, Teresa, for being my constant companion and listening to my frustrations.

Thank you, Mom and Dad, for your love, support, encouragement, and guidance; during my time as a graduate student and before. Your efforts laid the groundwork for my present and future success.

CONTRIBUTORS AND FUNDING SOURCES

Contributors

This work was supervised by a thesis committee consisting of Dr. Manoranjan Majji (advisor) and Dr. John Hurtado of the Department of Aerospace Engineering and Dr. Darren Cline of the Department of Statistics. Andrew B. Simon designed the demonstration's power supply. All other work conducted for the thesis was completed by the student independently.

Funding Sources

Graduate study was supported by a fellowship from Texas A&M University. This work was also made possible in part by funding from Smart Tower Systems and by sensors provided by VectorNav Technologies.

NOMENCLATURE

DCM	Direction Cosine Matrix
EKF	Extended Kalman Filter
IMU	Inertial Measurement Unit
LASR Lab	Land, Air, and Space Robotics Laboratory
MRP	Modified Rodrigues Parameter
NED	North-East-Down
UAV	Unmanned Aerial Vehicle
UKF	Unscented Kalman Filter

TABLE OF CONTENTS

Page

LIST OF FIGURES

Page

LIST OF TABLES

Page

CHAPTER I

INTRODUCTION

This thesis describes the development and testing of a networked embedded system used for attitude and position estimation.

Motivation

The immediate motivation for this project was to develop a networked embedded system and Kalman filters for the Land, Air, and Space Robotics (LASR) Laboratory, a research lab at Texas A&M that primarily focusses on guidance and estimation for aerospace systems. LASR Lab is using this architecture to develop a smart tower, and may use similar architecture in future projects. The smart tower is a radio or cell-phone tower in which an embedded system will use a Kalman filter to determine if the structural properties of the tower have changed, and if so to send an alert. Networked embedded systems for estimation and navigation have many potential applications, including Unmanned Aerial Vehicle (UAV) swarms, [1] self-driving cars, [2] and even docking spacecraft.

Project Architecture

A vehicle or a smart tower equipped with networked embedded systems will have the architecture shown in Figure 1. The vehicle or tower contains a microcomputer connected to some sensors. Software on the microcomputer uses sensor data to estimate some states—in the case of the smart tower, the natural frequency and damping ratio of the tower—and communicates wirelessly with other devices.

The demonstration's architecture, shown in Figure 2, is analogous to the smart tower architecture. The sensor is a VectorNav VN-100, an IMU that provides acceleration, magnetic field, and angular velocity measurements; and the microcomputer is a Tinker Board. These are

enclosed in a 3D-printed football-shaped shell. The Tinker Board uses Kalman filters to estimate the football's position, velocity, and orientation. A Wi-Fi router passes these data from the Tinker Board to a laptop and commands from the laptop to the Tinker Board. These messages are handled by the Open MPI software library.

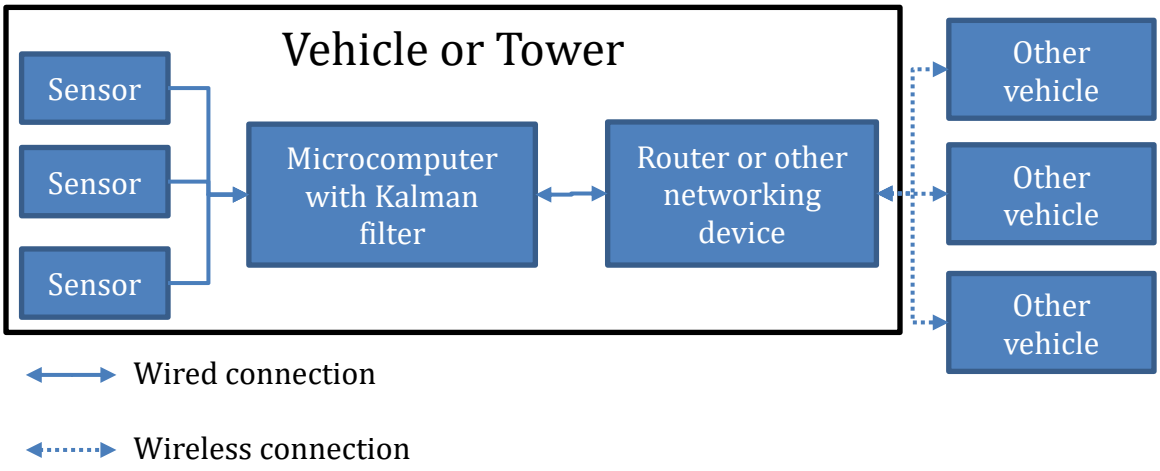

Figure 1: General Architecture

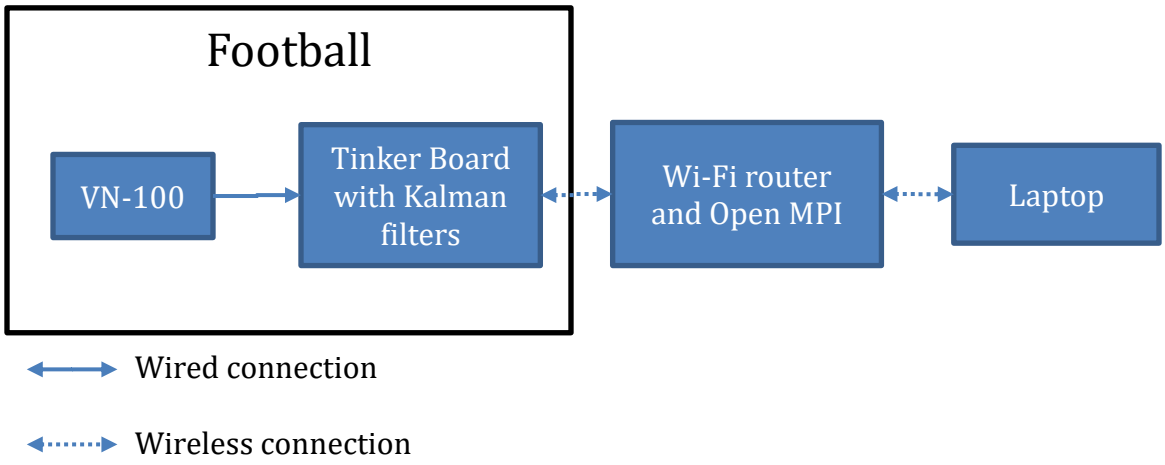

Figure 2: Demonstration Architecture

Scope and Structure of Thesis

In this thesis, the word "chapter" refers to a numbered chapter (e.g. "Introduction"). "Section" refers to a first-level division of a chapter (e.g. "Scope and Structure of Thesis") and "subsection" refers to a second-level division (e.g. "Installing Open MPI on Ubuntu"). This section provides an overview of structure and contents of the following chapters in this thesis, which explain how the football was made, how it was tested, and how it performed.

Chapter II, Demonstration Design, contains detailed descriptions of the football's hard components, the demonstration's software, and how everything was put together.

Chapter III, Kalman Filter Theory, explains what a Kalman filter is and provides the algorithms of the two Kalman filters developed for the demonstration.

Chapter IV, Position and Attitude Estimation, develops sensor models, dynamic models, and noise models and converts them into a form that can be used by the Kalman filters. It also details the calibration routines and initial conditions used by the demonstration.

Chapter V, Results, details the performance of the demonstration architecture, then describes the tests used to evaluate the Kalman filters' and the results of those tests.

Variable Name Conventions

Some quantities in this thesis have several different variations. For example, there is a true acceleration, a measured acceleration, and an estimated acceleration associated with each timestep. Related quantities are denoted by the same letter and are distinguished from each other by various modifiers. Table 1 lists the common modifiers and their meanings.

Table 1: Common Variable Modifiers

Modifier	Location	Example	Meaning
Bolded		$\boldsymbol{g}$	Vector quantity
^	Accent	$\hat{\boldsymbol{x}}$	Estimated quantity
~	Accent	$\tilde{\boldsymbol{a}}$	Measured quantity
(none)	Accent	$\boldsymbol{a}$	True quantity, without sensor or estimation error (does not apply to covariances such a *P*)
(number)	Subscript	ω_1	Vector component: $\boldsymbol{\omega} = \begin{bmatrix} \omega_1 \\ \omega_2 \\ \omega_3 \end{bmatrix}$
k	Subscript	$\boldsymbol{u}_k$	Quantity corresponding to the k^{th} timestep
×	Subscript	$\boldsymbol{a}_\times$	Cross product matrix: $\boldsymbol{a}_\times \equiv \begin{bmatrix} 0 & -a_3 & a_2 \\ a_3 & 0 & -a_1 \\ -a_2 & a_1 & 0 \end{bmatrix}$ $\boldsymbol{a}_\times \boldsymbol{b} = \boldsymbol{a} \times \boldsymbol{b}$
–	Superscript	P_k^-	Quantity before filtering at timestep *k* (but after propagating to timestep *k*)
+	Superscript	P_k^+	Quantity after filtering at timestep *k* (but before propagating to timestep *k* + 1)
T	Superscript	H_k^{T}	Matrix transpose
d	Before	$\mathrm{d}\boldsymbol{q}$	True value minus expected value: $\mathrm{d}\boldsymbol{q} \equiv \boldsymbol{q} - \hat{\boldsymbol{q}}$

CHAPTER II

DEMONSTRATION DESIGN

Sensors

The football is designed to estimate its position, velocity, and orientation using acceleration, angular velocity, and magnetic field measurements. These measurements are provided by a VectorNav VN-100 IMU. The VN-100 has a 3-axis magnetometer, accelerometer, and gyroscope and includes its own filters for estimating position, velocity, and orientation. [3] However, the football makes use only of the uncompensated magnetic, acceleration, angular velocity, and GPS position; because one purpose of the demonstration is to test the Kalman filters developed in this thesis, which should not rely on VectorNav's filtering.

Electrical Power System

The Tinker Board and the VN-100 use electrical power. The VN-100 is powered through its USB connection to the Tinker Board, and the Tinker Board is powered by a battery. A DC transformer steps down the voltage from the battery's 12.6V to the Tinker Board's 5V, and a 1.5A fuse protects the Tinker Board from excessive current. Figure 3 is a diagram of this setup.

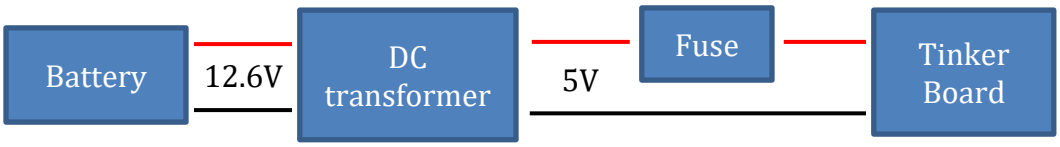

Figure 3: Electrical Power Diagram

The battery has a switch and two cables—one for charging and one for power output. To turn on the football, the switch must be in the on position and the Tinker Board connected to the battery. To charge the battery, the switch must be in the on position, the Tinker Board must be powered off or disconnected from the battery (the Tinker Board draws more power than the charger

provides), and the battery must be connected to the charger. If the battery is not powering the Tinker Board or being charged, the switch should be in the off position.

Physical Layout

The football shell is two 3D-printed pieces. Four tabs in the top piece (Figure 4) fit into slots in the bottom piece (Figure 5) and the two are held together by four #8 screws, one going through each tab. The screws are held in place by threaded inserts hammered into the sides of the bottom piece (Figure 6).

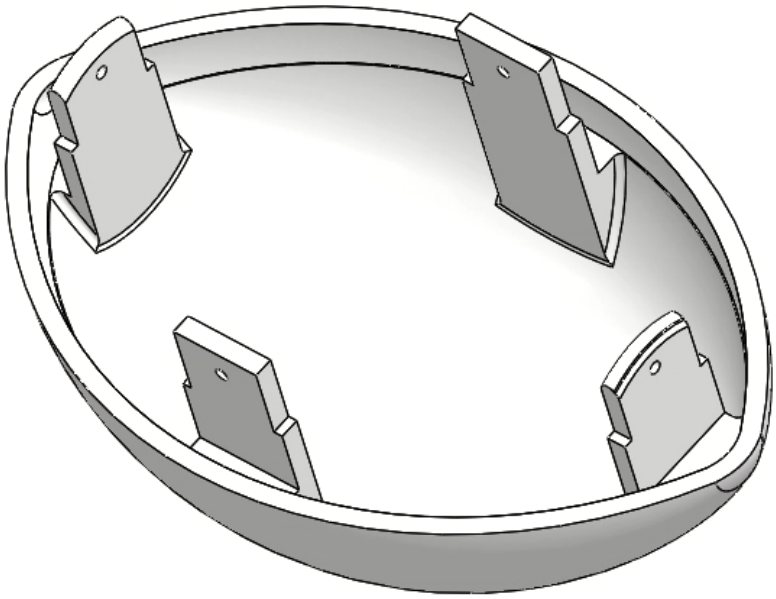

Figure 4: Top Piece of Shell

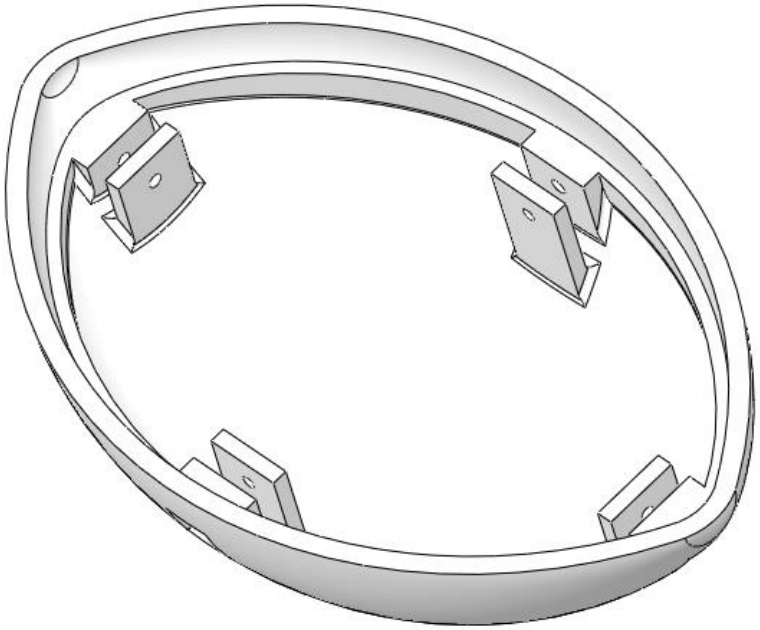

Figure 5: Bottom Piece of Shell

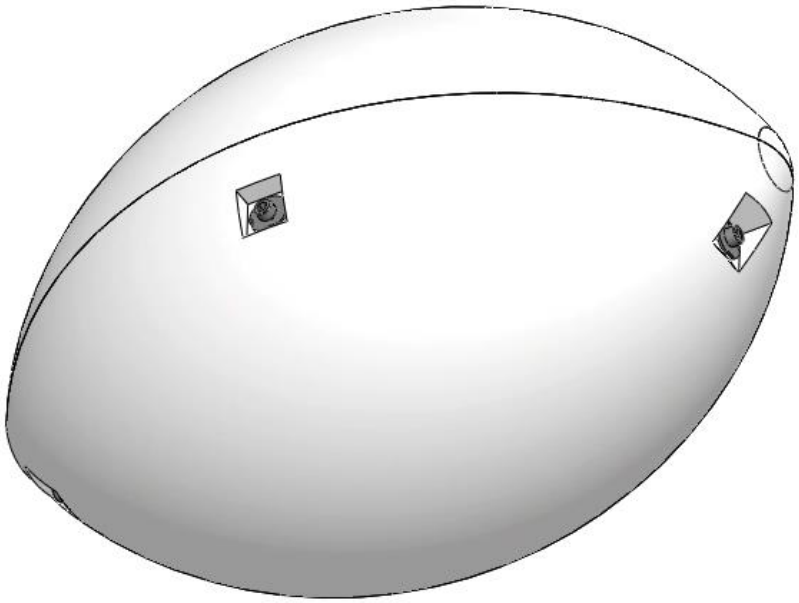

Figure 6: Bottom View of Football

All the components inside the shell are fastened to a plywood board that was laser-cut to fit between a ledge around the rim of the bottom piece (Figure 5) and ledges on the tabs of the top

piece (Figure 4). The Tinker Board is fastened to the board by four #4 screws threaded into inserts mounted on the bottom side of the board. The battery, the transformer, and the VN-100 are attached using a snap-together fastener similar to Velcro. Figure 7 shows the layout of these components.

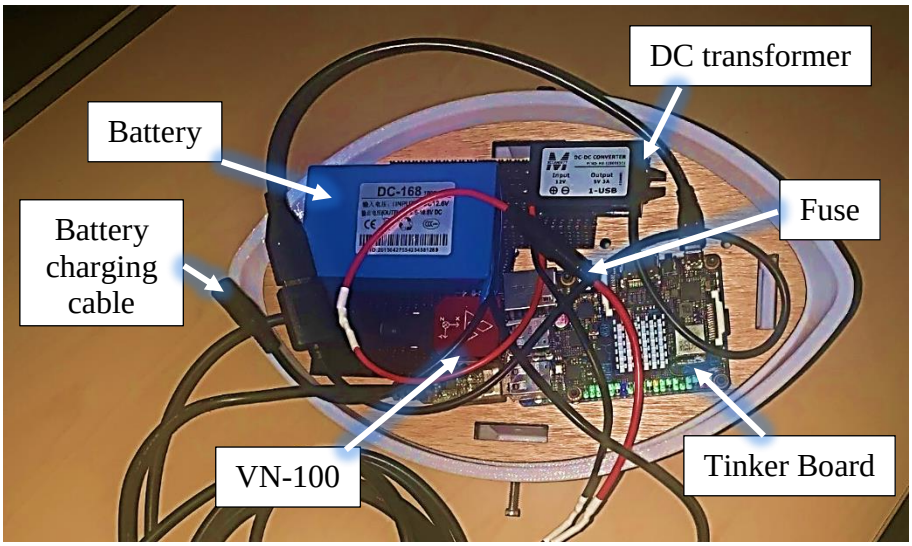

Figure 7: Component Layout

Software

The Tinker Board runs Ubuntu, a popular Linux distribution, and has Open MPI installed. The controller used for the experiments in this thesis was a Dell Inspiron laptop also running Ubuntu. The subsections at the end of this section contain instructions for installing Open MPI and setting up a controller. The router used for these experiments was configured to assign static IP addresses; the Tinker Board was always given the address 10.0.0.20. The football and the controller do not have to be connected to the same router, as long as there is a path between them and the football's IP address is known. The demonstration was successfully tested over an Internet connection.

The demonstration software is written in C++ and utilizes three non-standard libraries: the Open MPI library [4] for communications, the VectorNav library [5] for reading sensor data, and

a library called Eigen [6] for linear algebra. An Extended Kalman Filter (EKF) and an Unscented Kalman Filter (UKF) were written as classes in a header file. One program, called FootballProgram, resides on the Tinker Board; and another, called FootballReceiver, is on the controller. Most of the computing is done by FootballProgram; FootballReceiver simply logs and prints data.

To start the demonstration, the user connects the controller to the router and then runs the following two commands in the command line (or runs a shell script containing these commands):

```
ssh 10.0.0.20 "stty -F /dev/ttyUSB0 115200"
```

which sets the baud rate of communication between the VN-100 to the Tinker Board, and

```
mpirun --host 10.0.0.20 /home/tinker/Documents/Football/
FootballProgram : --host localhost /home/daniel/Football/
FootballReceiver
```

which runs the programs. It is important that the Tinker Board's process is listed first so that it has access to user input. The program then proceeds as shown in Figure 8.

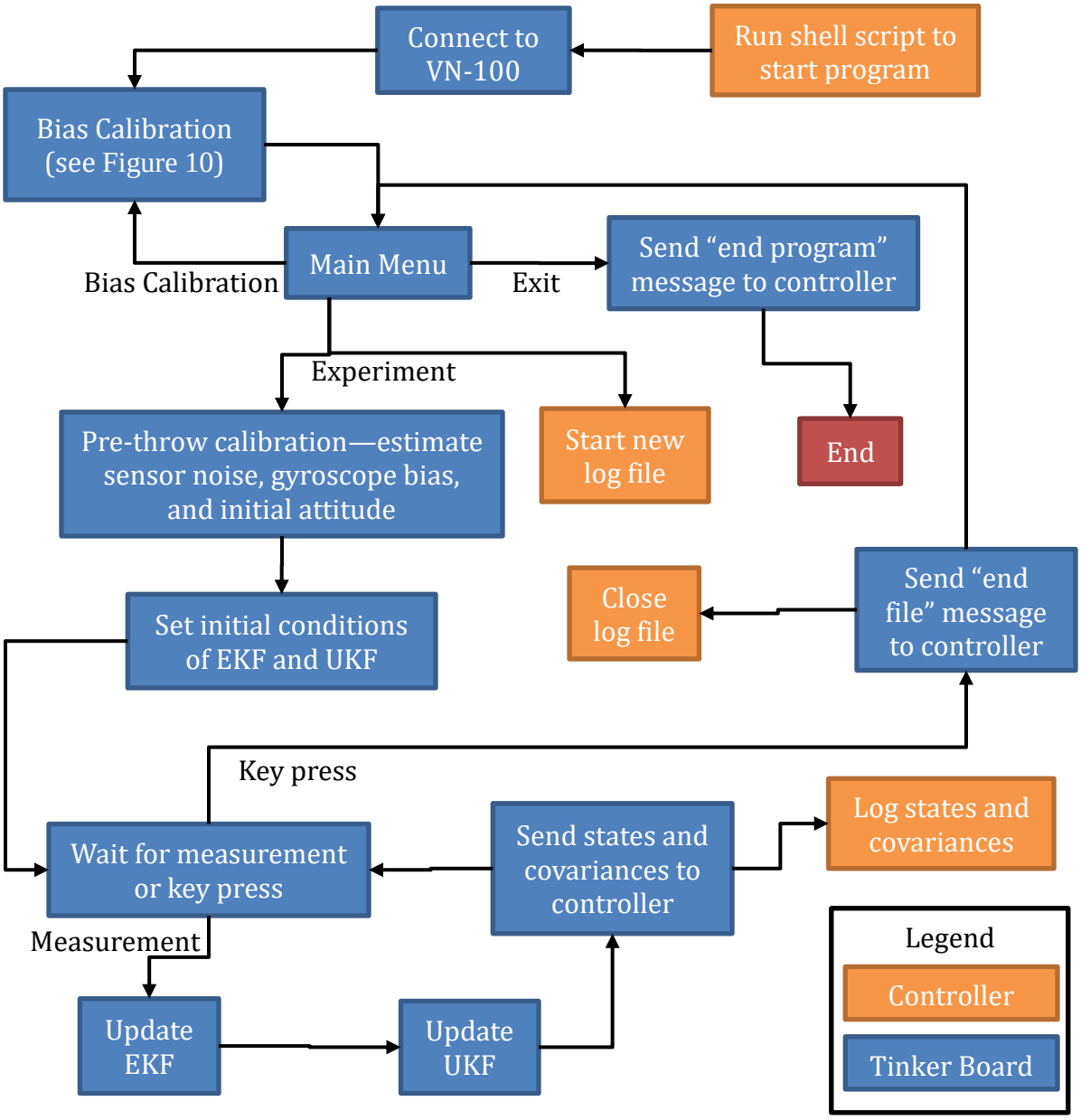

Figure 8: Software Flowchart

Installing Open MPI on Ubuntu

A Tinker Board can be used like a desktop computer if attached to a monitor, keyboard, and mouse. The following steps were used to install Open MPI on both the football's Tinker Board and the laptop used as the controller.

1. **Connect to the Internet**

This is required for Steps 2 and 3. It should be straightforward, but experience has shown that the Tinker Board's clock might need to be corrected before connecting (whenever the Tinker Board is powered off, its clock resets to January 2018).

2. **Download Open MPI**

Version 4.0.0 was used for the demonstration, but as of the time of writing another version (4.0.1) has been released. This should not make any difference, but the latest version can be found at https://www.open-mpi.org/software/ompi/ . Version 4.0.1 can be downloaded by running the following command:

```
wget https://www.open-mpi.org/software/ompi/v4.0/downloads/
openmpi-4.0.1.tar.gz
```

3. **Install prerequisite software**

Open MPI requires a C++ compiler (e.g. g++), and "make" and a decompressor are required for the installation. Experience has shown that Ubuntu itself should be up-to-date to ensure the installation will be successful. Running the following commands will ensure that these requirements are met and install any missing software:

```
sudo apt-get update
sudo apt-get install g++
sudo apt-get install make
sudo apt-get install libibnetdisc-dev
```

4. **Install Open MPI**

The following commands will decompress and install Open MPI. Installation will take several minutes and produce much output. Errors during the `configure` command are to be expected and do not mean that the installation is failing.

```
tar -xvf openmpi-4.0.1.tar.gz
cd openmpi-4.0.1
./configure --prefix="/home/$USER/.openmpi"
make
sudo make install
cd ..
```

5. Set path variables

This step is necessary for Linux to be able to find Open MPI commands. Open the file ~/.bashrc and add the following two lines to the beginning:

```
export PATH="$PATH:/home/$USER/.openmpi/bin"
export LD_LIBRARY_PATH="$LD_LIBRARY_PATH:/home/$USER/.openmpi/lib/"
```

To use Open MPI, it will be necessary to open a new terminal window or to run the above commands in the terminal.

6. Verify Installation

To verify that Open MPI is installed, enter

```
mpirun --version
```

This should return a message containing the version number just installed.

Configuring a Controller

The controller needs to have Open MPI installed and Open MPI needs to be able to access the Tinker Board without a username or password. The following steps were used to configure the laptop used as the controller for this thesis.

1. Install Open MPI

Follow the steps in the previous subsection.

2. **Connect the controller and Tinker Board to the same network and know the Tinker Board's IP address**

Ideally, the Tinker Board should have a static IP address. The router used for this thesis always assigns the Tinker Board the address 10.0.0.20. If Tinker Board's IP address is unknown, it may be necessary to run `ifconfig` in the Tinker Board's terminal. The IP address is listed next to `inet`, generally in the second block of output text. Use this IP address instead of 10.0.0.20 in the following steps.

3. **Set username for remote access**

This step will allow the controller to access the Tinker Board without specifying a username. If the Tinker Board's IP address changes, this step (only) will have to be repeated. Open (or create) the file ~/.ssh/config on the controller and add the following lines:

```
Host 10.0.0.20
User tinker
```

4. **Share keys**

This step will allow the controller to access to the Tinker Board without a password. Run the following commands on the controller. The first command will have several options; keep pressing Enter to use the default settings.

```
ssh-keygen
ssh-copy-id tinker@10.0.0.20
```

5. **Verify**

Run the command

```
mpirun --host 10.0.0.20 hostname
```

It should return `ELAR-Systems`.

CHAPTER III

KALMAN FILTER THEORY

This chapter provides an overview of the Kalman filter theory. The first section explains what a Kalman filter is and how it is applied to linear systems. The second and third sections explain how Extended Kalman Filters and Unscented Kalman Filters, respectively, use the same principles but apply them to nonlinear systems. The formulas given in this chapter are not specific to any system; the values of the matrices defined in this chapter, which are specific to the system, are derived in Chapter IV.

Kalman Filters

Kalman filters estimate a state vector $\boldsymbol{x}$ and a covariance matrix

$$P \equiv E[(\boldsymbol{x} - \hat{\boldsymbol{x}})(\boldsymbol{x} - \hat{\boldsymbol{x}})^{\mathrm{T}}] \tag{1}$$

At generally regular intervals, the filter receives a measurement vector $\boldsymbol{z}$ and updates the state and covariance estimates, first by propagating them forward in time and then filtering them with the measurements. These steps utilize linear dynamic and measurement models, respectively.

Propagation

The dynamics of the system are written in discrete form as

$$\boldsymbol{x}_{k+1} = \Phi \boldsymbol{x}_k + \Gamma \boldsymbol{u}_k + \Upsilon \boldsymbol{w}_k \tag{2}$$

where $\boldsymbol{u}_k$ is a disturbance vector and $\boldsymbol{w}_k$ is a Gaussian process noise vector. The estimates after propagation are

$$\hat{\boldsymbol{x}}_{k+1}^{-} = \Phi_{\mathrm{k}} \hat{\boldsymbol{x}}_k^{+} + \Gamma_{\mathrm{k}} \boldsymbol{u}_k \tag{3}$$

$$P_{k+1}^{-} = \Phi_k P_k^{+} \Phi_k^{\mathrm{T}} + \Upsilon_k Q_k \Upsilon_k^{\mathrm{T}} \tag{4}$$

where

$$Q_k \equiv E\left[\boldsymbol{w}_k \boldsymbol{w}_k^{\mathrm{T}}\right] \tag{5}$$

Filtering

The measurements are modelled as a linear function of the state plus some zero-mean Gaussian white noise:

$$\boldsymbol{y}_k = H_k \boldsymbol{x}_k + \boldsymbol{\eta}_{y_k} \tag{6}$$

$$E\left(\boldsymbol{\eta}_y \boldsymbol{\eta}_y^{\mathrm{T}}\right) = R \tag{7}$$

The Kalman gain is defined as

$$K_k = P_k^- H_k^{\mathrm{T}} \left(H_k P_k^- H_k^{\mathrm{T}} + R\right)^{-1} \tag{8}$$

and the expected measurement is

$$\hat{\boldsymbol{y}}_k = H_k \hat{\boldsymbol{x}}_k^- \tag{9}$$

The filtered state and covariance estimates are

$$\hat{\boldsymbol{x}}_k^+ = \hat{\boldsymbol{x}}_k^- + K_k(\boldsymbol{y}_k - \hat{\boldsymbol{y}}_k) \tag{10}$$

$$P_k^+ = (I - K_k H_k) P_k^- \tag{11}$$

Extended Kalman Filters

An EKF is similar to a standard Kalman filter except that the state transition matrix can be a function of the state and the measurement model can be nonlinear:

$$\boldsymbol{y}_k = \boldsymbol{h}(\boldsymbol{x}_k) + \boldsymbol{\eta}_{y_k} \tag{12}$$

The measurement sensitivity matrix from Equation (6) is approximated as

$$H_k = \left.\frac{\partial \boldsymbol{h}(\boldsymbol{x})}{\partial \boldsymbol{x}}\right|_{\hat{\boldsymbol{x}}_k^-} \tag{13}$$

Unscented Kalman Filters

A UKF handles nonlinear propagation and measurement models differently from an EKF. A UKF uses considerably more computing power than an EKF but captures some of the nonlinearity of the original equations. Despite requiring more computing power, UKFs are viable on small embedded systems; Fico et al. were able to implement a 20-state UKF on a 10€ microcontroller at 35Hz. [7] The UKF used in the demonstration follows the procedure laid out by Crassidis and Markley in [8].

Propagation

This version of the UKF uses a trapezoidal approximation for the process noise—half is added to the covariance before propagation (in Equation (15)) and the other half after propagation (in Equation (21)). For this purpose it is useful to define the following quantity:

$$\bar{Q}_k = \frac{1}{2}\Upsilon_k Q_k \Upsilon_k^{\mathrm{T}} \tag{14}$$

Let n be the number of states in the filter (not to be confused with $\boldsymbol{n}$, the normalized magnetic measurement). Each timestep, a set of $2n + 1$ vectors are generated with a mean of the state vector and a covariance proportional to the covariance matrix. First, a set of $2n$ vectors is computed from the columns of Equation (15):

$$\pm\sqrt{(n+\lambda)(P_k^+ + \bar{Q}_k)} \rightarrow \boldsymbol{\sigma}_k(i) \tag{15}$$

where λ is a pre-determined weighting factor and the square root indicates a matrix such that

$$\sqrt{X}\left(\sqrt{X}\right)^{\mathrm{T}} = X \tag{16}$$

These vectors are added to the state vector to create the sigma points:

$$\boldsymbol{\chi}_k^+(0) = \hat{\boldsymbol{x}}_k^+ \tag{17}$$

$$\boldsymbol{\chi}_k^+(i) = \hat{\boldsymbol{x}}_k^+ + \boldsymbol{\sigma}_k(i) \tag{18}$$

Each sigma point is propagated the same way as a state vector in a standard Kalman filter:

$$\boldsymbol{\chi}_{k+1}^-(i) = \Phi_k \boldsymbol{\chi}_k^+(i) + \Gamma_k \boldsymbol{u} \tag{19}$$

The post-propagation state estimate is a weighted average of these propagated sigma points:

$$\hat{\boldsymbol{x}}_{k+1}^- = \frac{1}{n+\lambda}\left(\lambda \boldsymbol{\chi}_{k+1}^-(0) + \frac{1}{2}\sum_{i=1}^{2n} \boldsymbol{\chi}_{k+1}^-(i)\right) \tag{20}$$

The post-propagation covariance matrix is calculated using Equation (21):

$$P_{k+1}^- = \frac{1}{n+\lambda}\Bigg(\lambda(\boldsymbol{\chi}_{k+1}^-(0) - \hat{\boldsymbol{x}}_{k+1}^-)(\boldsymbol{\chi}_{k+1}^-(0) - \hat{\boldsymbol{x}}_{k+1}^-)^{\mathrm{T}} + \frac{1}{2}\sum_{i=1}^{2n}(\boldsymbol{\chi}_{k+1}^-(i) - \hat{\boldsymbol{x}}_{k+1}^-)(\boldsymbol{\chi}_{k+1}^-(i) - \hat{\boldsymbol{x}}_{k+1}^-)^{\mathrm{T}}\Bigg) + \bar{Q}_k \tag{21}$$

Figure 9 represents a propagation step of a two-state UKF.

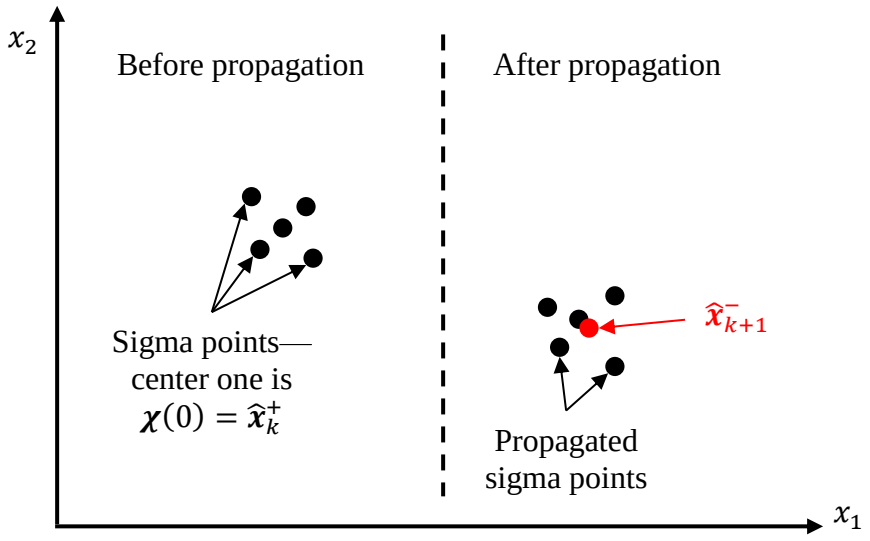

Figure 9: Two-State UKF Propagation

Filtering

The UKF has the same measurement model as the EKF (Equation (12)). The expected measurement is

$$\hat{\boldsymbol{y}}_k = \frac{1}{n+\lambda}\left(\lambda \boldsymbol{\gamma}_k(0) + \frac{1}{2}\sum_{i=1}^{2n}\gamma_k(i)\right) \tag{22}$$

where

$$\boldsymbol{\gamma}_k(i) = \boldsymbol{h}(\boldsymbol{\chi}_k^-(i)) \tag{23}$$

The Kalman gain is calculated using Equation (24); then the state is filtered using Equation (10) and the covariance using Equation (25).

$$K_k = P_k^{xy}(P_k^{vv})^{-1} \tag{24}$$

$$P_k^+ = P_k^- - K_k P_k^{vv} K_k^{\mathrm{T}} \tag{25}$$

where

$$P_k^{vv} = \frac{1}{n+\lambda}\left(\lambda(\boldsymbol{\gamma}_k(0) - \hat{\boldsymbol{y}}_k)(\boldsymbol{\gamma}_k(0) - \hat{\boldsymbol{y}}_k)^{\mathrm{T}} + \frac{1}{2}\sum_{i=1}^{2n}(\boldsymbol{\gamma}_k(i) - \hat{\boldsymbol{y}}_k)(\boldsymbol{\gamma}_k(i) - \hat{\boldsymbol{y}}_k)^{\mathrm{T}}\right) + R_k \tag{26}$$

is the innovation covariance and

$$P_k^{xy} = \frac{1}{n+\lambda}\left(\lambda(\boldsymbol{\chi}_k^-(0) - \boldsymbol{x}_k^-)(\boldsymbol{\gamma}_k(0) - \hat{\boldsymbol{y}}_k)^{\mathrm{T}} + \frac{1}{2}\sum_{i=1}^{2n}(\boldsymbol{\chi}_k^-(i) - \boldsymbol{x}_k^-)(\boldsymbol{\gamma}_k(i) - \hat{\boldsymbol{y}}_k)^{\mathrm{T}}\right) \tag{27}$$

is the cross-correlation.

CHAPTER IV

POSITION AND ATTIDUTE ESTIMATION

As mentioned previously, the football's position and attitude are estimated using an EKF and a UKF. This chapter develops the sensor and dynamic models used for this estimation, describes the procedures for estimating initial conditions, derives the values of all matrices used by the filters, and explains how the filters were modified to use quaternions (an overparameterization) to represent attitude.

Quaternion Kinematics

Quaternion are commonly used to represent rotations. A rotation by an angle α about a unit axis $\boldsymbol{e}$ is represented as [9]

$$\boldsymbol{q} = \begin{bmatrix} q_0 \\ q_1 \\ q_2 \\ q_3 \end{bmatrix} = \begin{bmatrix} q_0 \\ \underline{q} \end{bmatrix} = \begin{bmatrix} \cos\dfrac{\alpha}{2} \\ \boldsymbol{e}\sin\dfrac{\alpha}{2} \end{bmatrix} \tag{28}$$

(This thesis uses the scalar-first convention.) Because they are more computationally efficient than direction cosine matrices (DCMs) and, unlike even more efficient methods like Rodrigues parameters, have no mathematical singularities, the Kalman filters developed for the demonstration use quaternions to represent attitude. This section provides some quaternion kinematic equations relevant to the filters.

A quaternion can be converted into a DCM by Equation (29):

$$C(\boldsymbol{q}) = (2q_0^2 - 1)I_{3\times 3} + 2\underline{q}\underline{q}^{\mathrm{T}} - 2q_0\underline{q}_{\times} \tag{29}$$

The time derivative of the quaternion as a function of the angular velocity vector $\boldsymbol{\omega}$ is given by [10]

$$\frac{d\boldsymbol{q}}{dt} = \frac{1}{2}\Omega(\boldsymbol{\omega})\boldsymbol{q} = \frac{1}{2}\Xi(\boldsymbol{q})\boldsymbol{\omega} \tag{30}$$

where

$$\Omega(\boldsymbol{\omega}) = \begin{bmatrix} 0 & -\omega_1 & -\omega_2 & -\omega_3 \\ \omega_1 & 0 & \omega_3 & -\omega_2 \\ \omega_2 & -\omega_3 & 0 & \omega_1 \\ \omega_3 & \omega_2 & -\omega_1 & 0 \end{bmatrix} \tag{31}$$

$$\Xi(\boldsymbol{q}) = \begin{bmatrix} -q_1 & -q_2 & -q_3 \\ q_0 & -q_3 & q_2 \\ q_3 & q_0 & -q_1 \\ -q_2 & q_1 & q_0 \end{bmatrix} \tag{32}$$

Sensor Models

Three sensors are used in the demonstration, each with three axes: the accelerometer, gyroscope, and magnetometer. Each of these sensors assumed to be corrupted by Gaussian white noise and by biases. The gyroscope bias changes according to the random walk model, but the accelerometer and magnetometer biases are assumed to be constant. These biases are estimated during the calibration routines described in the Calibration section.

Mathematically, the gyroscope measurement is modelled as

$$\tilde{\boldsymbol{\omega}} = \boldsymbol{\omega} + \boldsymbol{b}_\omega + \boldsymbol{\eta}_\omega \tag{33}$$

Where $\boldsymbol{\omega}$ is the true angular velocity, $\boldsymbol{b}_\omega$ is the gyroscope biases and $\boldsymbol{\eta}_\omega$ is a Gaussian noise vector such that

$$Q_\omega \equiv E(\boldsymbol{\eta}_\omega \boldsymbol{\eta}_\omega{}^{\mathrm{T}}) = \begin{bmatrix} \sigma_{\omega_1}^2 & 0 & 0 \\ 0 & \sigma_{\omega_2}^2 & 0 \\ 0 & 0 & \sigma_{\omega_3}^2 \end{bmatrix} \tag{34}$$

The bias changes according to the random walk model:

$$\frac{d\boldsymbol{b}_\omega}{dt} = \boldsymbol{\eta}_b \tag{35}$$

$$Q_b \equiv E\left(\boldsymbol{\eta}_b \boldsymbol{\eta}_b^{\mathrm{T}}\right) = \begin{bmatrix} \sigma_b^2 & 0 & 0 \\ 0 & \sigma_b^2 & 0 \\ 0 & 0 & \sigma_b^2 \end{bmatrix} \tag{36}$$

The value of σ_b has been arbitrarily set to 0.001°/s.

The accelerometer measurement is modelled as follows:

$$\tilde{\boldsymbol{a}} = C(\boldsymbol{q})(\boldsymbol{a}^N - \boldsymbol{g}) + \boldsymbol{b}_a + \boldsymbol{\eta}_a \tag{37}$$

where C is given by Equation (29); $\boldsymbol{a}$ is the true acceleration (the superscript N indicates the NED (North-East-Down) frame); $\boldsymbol{g}$ is the gravitational acceleration; $\boldsymbol{b}_a$ is the accelerometer biases, which are assumed to be constant; and $\boldsymbol{\eta}_a$ is the accelerometer white noise such that

$$E(\boldsymbol{\eta}_a \boldsymbol{\eta}_a{}^{\mathrm{T}}) = \begin{bmatrix} \sigma_{a_1}^2 & 0 & 0 \\ 0 & \sigma_{a_2}^2 & 0 \\ 0 & 0 & \sigma_{a_3}^2 \end{bmatrix} \tag{38}$$

The angular velocity is not a function of the Kalman filter state; and although acceleration is a function of the state, it cannot be used to recover the state. Therefore, those measurements are used in the propagation part of the filter rather than filtering portion. Acceleration is treated as a disturbance and angular velocity is used to determine part of the state transition matrix.

The magnetometer measurement is modelled as

$$\tilde{\boldsymbol{m}} = C(\boldsymbol{q})\boldsymbol{m}^N + \boldsymbol{b}_m + \boldsymbol{\eta}_m \tag{39}$$

where $\boldsymbol{m}$ is the true magnetic field and $\boldsymbol{\eta}_m$ is a Gaussian noise vector such that

$$E(\boldsymbol{\eta}_m \boldsymbol{\eta}_m{}^{\mathrm{T}}) = \begin{bmatrix} \sigma_{m_1}^2 & 0 & 0 \\ 0 & \sigma_{m_2}^2 & 0 \\ 0 & 0 & \sigma_{m_3}^2 \end{bmatrix} \tag{40}$$

This measurement is normalized before being passed to the Kalman filters:

$$\boldsymbol{y} = \tilde{\boldsymbol{n}} = \frac{\tilde{\boldsymbol{m}} - \hat{\boldsymbol{b}}_m}{\sqrt{(\tilde{\boldsymbol{m}} - \hat{\boldsymbol{b}}_m) \cdot (\tilde{\boldsymbol{m}} - \hat{\boldsymbol{b}}_m)}} = C(\boldsymbol{q})\boldsymbol{n}^N + \boldsymbol{\eta}_n \tag{41}$$

$\boldsymbol{n}^N$ is estimated during the pre-throw calibration and the magnetometer bias during the Bias calibration (see the Calibration section). The noise of the normalized vector is such that

$$R = E(\boldsymbol{\eta}_n \boldsymbol{\eta}_n^{\mathrm{T}}) = \begin{bmatrix} \sigma_{n_1}^2 & 0 & 0 \\ 0 & \sigma_{n_2}^2 & 0 \\ 0 & 0 & \sigma_{n_3}^2 \end{bmatrix} \tag{42}$$

$$\sigma_{n_i}^2 = \frac{\sigma_{m_i}^2}{\boldsymbol{m} \cdot \boldsymbol{m} + \sigma_{m_1}^2 + \sigma_{m_2}^2 + \sigma_{m_3}^2} \tag{43}$$

Dynamics

The kinematic equations for position and velocity are

$$\frac{d\boldsymbol{r}}{dt} = \boldsymbol{v} \tag{44}$$

$$\frac{d\boldsymbol{v}}{dt} = \boldsymbol{a} \tag{45}$$

Applying a zero-order hold assumption and integrating Equations (44), (45), and (35) yields these discrete kinematic equations:

$$\boldsymbol{r}_{k+1} = \boldsymbol{r}_k + \Delta t_k \boldsymbol{v}_k + \frac{\Delta t^2}{2} \boldsymbol{a}_k \tag{46}$$

$$\boldsymbol{v}_{k+1} = \boldsymbol{v}_k + \Delta t_k \boldsymbol{a}_k \tag{47}$$

$$\boldsymbol{b}_{\omega_{k+1}} = \boldsymbol{b}_{\omega_k} + \Delta t \boldsymbol{\eta}_{b_k} \tag{48}$$

The approach to propagating the quaternion is slightly different, as its normality must be maintained. The result is [10]

$$\boldsymbol{q}_{k+1} = \left(\cos\frac{\theta_k}{2} I_{4\times 4} + \frac{\sin\frac{\theta_k}{2}}{\theta_k} \Omega(\Delta t_k \boldsymbol{\omega}_k) \right) \boldsymbol{q}_k \tag{49}$$

where Ω is defined in Equation (31) and

$$\theta_k = \Delta t_k \sqrt{\boldsymbol{\omega}_k \cdot \boldsymbol{\omega}_k} \tag{50}$$

Substituting the sensor models and estimates into Equations (46), (47), and (49) yields

$$\begin{aligned} \boldsymbol{r}_{k+1} = \boldsymbol{r}_k + \Delta t_k \hat{\boldsymbol{v}}_k + \frac{\Delta t_k^2}{2} [C^{\mathrm{T}}(\hat{\boldsymbol{q}}_k)(\tilde{\boldsymbol{a}}_k - \boldsymbol{b}_a) + \boldsymbol{g}] + \Delta t_k \mathrm{d}\boldsymbol{v}_k + \frac{\Delta t_k^2}{2} \mathrm{d}\boldsymbol{a}_k \\ + \boldsymbol{\varepsilon}_{r_k} \end{aligned} \tag{51}$$

$$\boldsymbol{v}_{k+1} = \boldsymbol{v}_k + \Delta t_k [C^{\mathrm{T}}(\hat{\boldsymbol{q}}_k)(\tilde{\boldsymbol{a}}_k - \boldsymbol{b}_a) + \boldsymbol{g} + \mathrm{d}\boldsymbol{a}] + \boldsymbol{\varepsilon}_{v_k} \tag{52}$$

$$\boldsymbol{q}_{k+1} = \Phi_k^{qq} \boldsymbol{q}_k - \frac{\sin\frac{\hat{\theta}_k}{2}}{\hat{\theta}_k} \Delta t_k \Xi(\boldsymbol{q}_k) \left(\hat{\boldsymbol{b}}_{\omega_k} + \mathrm{d}\hat{\boldsymbol{b}}_{\omega_k} + \boldsymbol{\eta}_{\omega_k} \right) + \boldsymbol{\varepsilon}_{q_k} \tag{53}$$

where

$$\Delta t_k \equiv t_{k+1} - t_k \tag{54}$$

$$\hat{\boldsymbol{a}}_k = C^{\mathrm{T}}(\hat{\boldsymbol{q}})(\tilde{\boldsymbol{a}}_k - \boldsymbol{b}_a) + \boldsymbol{g} \tag{55}$$

$$\hat{\boldsymbol{\omega}}_k = \tilde{\boldsymbol{\omega}}_k - \hat{\boldsymbol{b}}_{\omega_k} \tag{56}$$

$$\Phi_k^{qq} = \cos\frac{\hat{\theta}_k}{2}\mathrm{I}_{4\times4} + \frac{\sin\frac{\hat{\theta}_k}{2}}{\hat{\theta}_k}\Omega(\Delta t_k\tilde{\boldsymbol{\omega}}_k) \tag{57}$$

$$\hat{\theta}_k = \Delta t_k\sqrt{\hat{\boldsymbol{\omega}}_k\cdot\hat{\boldsymbol{\omega}}_k} \tag{58}$$

the d operator is defined in Table 1, and the $\boldsymbol{\varepsilon}$ terms represent errors introduced by the zero-order hold assumption.

Derivation of Acceleration Process Noise

The calculations in this section take place each timestep before propagation, so nearly all quantities should have the subscript *k*; and all quaternion quantities should have the superscript +. For the sake of notational sanity, these subscripts and superscripts have been dropped in this section.

d$\boldsymbol{a}$ is one of the process noise terms in both Kalman filters. This section derives the value of

$$Q_a \equiv E(\mathrm{d}\boldsymbol{a}\mathrm{d}\boldsymbol{a}^{\mathrm{T}}) \tag{59}$$

which is part of the process noise covariance *Q*, in terms of the quaternion covariance

$$O \equiv E(\mathrm{d}\boldsymbol{q}\mathrm{d}\boldsymbol{q}^{\mathrm{T}}) \tag{60}$$

which can be obtained from the Kalman filter's covariance matrix. The convention used in this section is that the row and column indices of *O* start at 0, so that

$$O_{ij} = E\left(\mathrm{d}q_i\mathrm{d}q_j\right) \tag{61}$$

From Equations (37) and (55),

$$\mathrm{d}\boldsymbol{a} = [C^{\mathrm{T}}(\boldsymbol{q}) - C^{\mathrm{T}}(\hat{\boldsymbol{q}})](\boldsymbol{a} - \boldsymbol{b}_a) + C^{\mathrm{T}}(\boldsymbol{q})\boldsymbol{\eta}_a \tag{62}$$

Substituting Equation (62) into Equation (59),

$$Q_a = E([\mathrm{d}C^{\mathrm{T}}\breve{\boldsymbol{a}} + C^{\mathrm{T}}(\boldsymbol{q})\boldsymbol{\eta}_a][\mathrm{d}C^{\mathrm{T}}\breve{\boldsymbol{a}} + C^{\mathrm{T}}(\boldsymbol{q})\boldsymbol{\eta}_a]^{\mathrm{T}}) \tag{63}$$

where

$$\mathrm{d}C \equiv C(\boldsymbol{q}) - C(\hat{\boldsymbol{q}}) \tag{64}$$

$$\breve{\boldsymbol{a}} \equiv \tilde{\boldsymbol{a}} - \boldsymbol{b}_a \tag{65}$$

Rearranging some terms and recognizing that the sensor noise is uncorrelated to the attitude estimate,

$$Q_a = E(\mathrm{d}C^{\mathrm{T}}\breve{\boldsymbol{a}}\breve{\boldsymbol{a}}^{\mathrm{T}}\mathrm{d}C) + C^{\mathrm{T}}(\boldsymbol{q})E(\boldsymbol{\eta}_a\boldsymbol{\eta}_a^{\mathrm{T}})C(\boldsymbol{q}) \tag{66}$$

The first expected value in Equation (66) is shown in Equation (68). This was obtained by linearizing Equation (64) about the estimated attitude, resulting in Equation (67), and substituting that into the expression.

$$\frac{1}{2}\mathrm{d}C \approx 2\hat{q}_0\mathrm{d}q_0 I_{3\times 3} + \mathrm{d}\underline{q}\underline{\hat{q}}^{\mathrm{T}} + \underline{\hat{q}}\,\mathrm{d}\underline{q}^{\mathrm{T}} - \hat{q}_0\mathrm{d}\underline{q}_{\times} - \mathrm{d}q_0\underline{\hat{q}}_{\times} \tag{67}$$

$$\begin{aligned}\frac{1}{4}E(\mathrm{d}C^{\mathrm{T}}\breve{\boldsymbol{a}}\breve{\boldsymbol{a}}^{\mathrm{T}}\mathrm{d}C) \\ \approx 4\hat{q}_0^2 O_{0,0}\breve{\boldsymbol{a}}\breve{\boldsymbol{a}}^{\mathrm{T}} + \left(\underline{\hat{q}}\cdot\breve{\boldsymbol{a}}\right)^2 O_{1-3,1-3} + \underline{\hat{q}}\breve{\boldsymbol{a}}^{\mathrm{T}}O_{1-3,1-3}\breve{\boldsymbol{a}}\underline{\hat{q}}^{\mathrm{T}} \\ - \hat{q}_0^2\breve{\boldsymbol{a}}_{\times}O_{1-3,1-3}\breve{\boldsymbol{a}}_{\times} - O_{0,0}\underline{\hat{q}}_{\times}\breve{\boldsymbol{a}}\breve{\boldsymbol{a}}^{\mathrm{T}}\underline{\hat{q}}_{\times} + X + X^{\mathrm{T}}\end{aligned} \tag{68}$$

where

$$\begin{aligned}X = \Big(\left(O_{1-3,0}\cdot\breve{\boldsymbol{a}}\right)\left(2\hat{q}_0 I_{3\times 3} + \underline{\hat{q}}_{\times}\right) + \left(\left(\underline{\hat{q}}\cdot\breve{\boldsymbol{a}}\right)I_{3\times 3} - \hat{q}_0\breve{\boldsymbol{a}}_{\times}\right)O_{1-3,1-3}\Big)\breve{\boldsymbol{a}}\underline{\hat{q}}^{\mathrm{T}} \\ + 2\hat{q}_0\left(\left(\underline{\hat{q}}\cdot\breve{\boldsymbol{a}}\right)O_{1-3,0} - \breve{\boldsymbol{a}}_{\times}\left(\hat{q}_0 O_{1-3,0} + O_{0,0}\underline{\hat{q}}\right)\right)\breve{\boldsymbol{a}}^{\mathrm{T}} \\ - \left(\underline{\hat{q}}\cdot\breve{\boldsymbol{a}}\right)\hat{q}_0\breve{\boldsymbol{a}}_{\times}O_{1-3,1-3} + \left(\hat{q}_0\breve{\boldsymbol{a}}_{\times} - \left(\underline{\hat{q}}\cdot\breve{\boldsymbol{a}}\right)I_{3\times 3}\right)O_{1-3,0}\breve{\boldsymbol{a}}^{\mathrm{T}}\underline{\hat{q}}_{\times}\end{aligned} \tag{69}$$

The second expected value in Equation (66) is estimated simply by using the estimated attitude and the measurement covariance from Equation (38). Thus,

$$Q_a \approx 4\hat{q}_0^2 O_{0,0}\breve{\boldsymbol{a}}\breve{\boldsymbol{a}}^{\mathrm{T}} + \left(\underline{\hat{q}} \cdot \breve{\boldsymbol{a}}\right)^2 O_{1-3,1-3} + \underline{\hat{q}}\breve{\boldsymbol{a}}^{\mathrm{T}} O_{1-3,1-3}\breve{\boldsymbol{a}}\underline{\hat{q}}^{\mathrm{T}} - \hat{q}_0^2\breve{\boldsymbol{a}}_{\times} O_{1-3,1-3}\breve{\boldsymbol{a}}_{\times}$$
$$- O_{0,0}\underline{\hat{q}}_{\times}\breve{\boldsymbol{a}}\breve{\boldsymbol{a}}^{\mathrm{T}}\underline{\hat{q}}_{\times} + X + X^{\mathrm{T}} + C^{\mathrm{T}}(\hat{\boldsymbol{q}}) \begin{bmatrix} \sigma_{a_1}^2 & 0 & 0 \\ 0 & \sigma_{a_2}^2 & 0 \\ 0 & 0 & \sigma_{a_3}^2 \end{bmatrix} C(\hat{\boldsymbol{q}}) \tag{70}$$

Calibration

The demonstration has two calibration routines. This section first describes the attitude estimation routine used in both calibrations and then the purpose and methods used for each calibration.

Attitude Determination

Both calibration routines use accelerometer and magnetometer measurements to determine the football's orientation in the NED coordinate system. The reference vectors are gravity (which defines the third axis of the NED frame) and a magnetic field vector based on the World Magnetic Model (which is used to fix the azimuth but not the elevation). The procedure is shown in Equations (71) through (78).

$$\cos\theta_a = \hat{a}_3 \tag{71}$$

$$\boldsymbol{q}_a = \begin{bmatrix} \cos\dfrac{\theta_a}{2} \\ \dfrac{\hat{a}_2}{\sqrt{\hat{a}_1^2 + \hat{a}_2^2}}\sin\dfrac{\theta_a}{2} \\ \dfrac{-\hat{a}_1}{\sqrt{\hat{a}_1^2 + \hat{a}_2^2}}\sin\dfrac{\theta_a}{2} \\ 0 \end{bmatrix} \tag{72}$$

$$\boldsymbol{n}^I = \boldsymbol{q}_a \otimes \hat{\boldsymbol{n}} \tag{73}$$

$$\boldsymbol{o}^I \equiv \frac{1}{\sqrt{(n_1^I)^2 + (n_2^I)^2}} \begin{bmatrix} n_1^I \\ n_2^I \end{bmatrix} \tag{74}$$

$$\boldsymbol{o}^N \equiv \frac{1}{\sqrt{(n_1^N)^2 + (n_2^N)^2}} \begin{bmatrix} n_1^N \\ n_2^N \end{bmatrix} \tag{75}$$

$$\cos\theta_m = \boldsymbol{o}^I \cdot \boldsymbol{o}^N \tag{76}$$

$$\boldsymbol{q}_m = \begin{bmatrix} \cos\frac{\theta_m}{2} \\ 0 \\ 0 \\ \pm\sin\frac{\theta_m}{2} \end{bmatrix} \tag{77}$$

$$\boldsymbol{q} = \boldsymbol{q}_a^* \otimes \boldsymbol{q}_m^* \tag{78}$$

The sign of the ± is the sign of $\boldsymbol{o}^I \times \boldsymbol{o}^N$, $\otimes$ represents quaternion multiplication, and the superscript * represents a quaternion's conjugate. Half-angle formulas are used to avoid inverse trigonometry.

Bias Calibration

The bias calibration estimates the accelerometer and magnetometer biases and the magnitude of the gravitational acceleration. It is performed automatically after the football is powered on. The calibration has three stages, with the demonstration asking the user to orient the football in different ways between stages.

The measurements from each stage are averaged and the algorithm enters a loop, as shown in Figure 10. Attitude estimation follows the steps laid out in the Attitude Determination subsection, and the least squares estimation uses Equations (79) and (80).

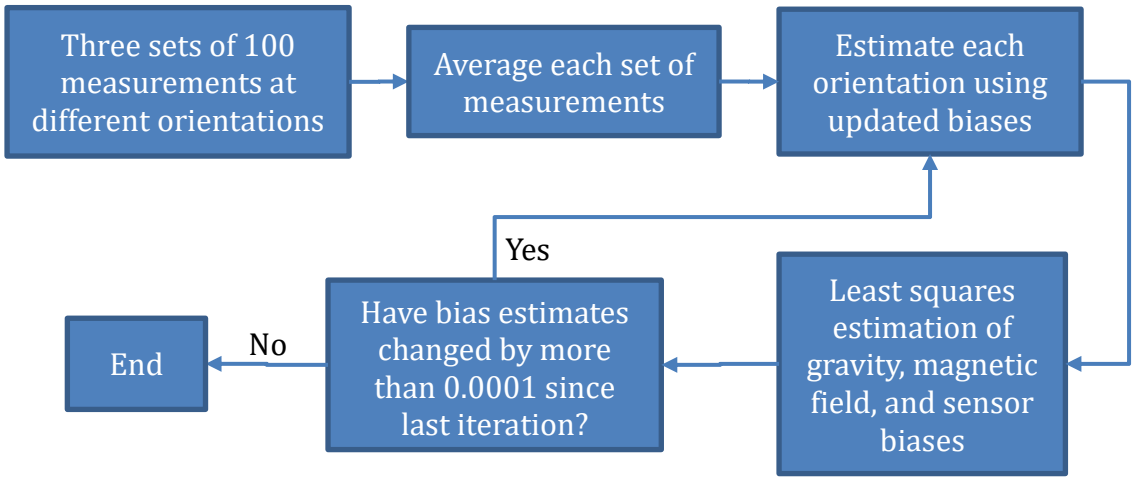

Figure 10: Bias Calibration

$$\begin{bmatrix} \hat{\boldsymbol{g}} \\ \hat{\boldsymbol{b}}_a \end{bmatrix} = \left(A^{\mathrm{T}}A\right)^{-1}A^{\mathrm{T}}\begin{bmatrix} \tilde{\boldsymbol{a}}_1 \\ \tilde{\boldsymbol{a}}_2 \\ \tilde{\boldsymbol{a}}_3 \end{bmatrix} \tag{79}$$

$$\begin{bmatrix} \hat{\boldsymbol{m}} \\ \hat{\boldsymbol{b}}_m \end{bmatrix} = \left(A^{\mathrm{T}}A\right)^{-1}A^{\mathrm{T}}\begin{bmatrix} \tilde{\boldsymbol{m}}_1 \\ \tilde{\boldsymbol{m}}_2 \\ \tilde{\boldsymbol{m}}_3 \end{bmatrix} \tag{80}$$

where

$$A = \begin{bmatrix} C(\boldsymbol{q}_1) & I_{3\times3} \\ C(\boldsymbol{q}_2) & I_{3\times3} \\ C(\boldsymbol{q}_3) & I_{3\times3} \end{bmatrix} \tag{81}$$

To maintain the definition of the coordinate system (z-axis is down), the estimated gravity vector is adjusted:

$$\hat{\boldsymbol{g}}(new) = \begin{bmatrix} 0 \\ 0 \\ \sqrt{\hat{\boldsymbol{g}}(old)\cdot\hat{\boldsymbol{g}}(old)} \end{bmatrix} \tag{82}$$

Calibration Before a Throw

Immediately before each throw, the football must be left stationary for about one second. 100 measurements taken during this period are used to perform the magnetic reference calibration and estimate the angular rate biases, sensor noise, and initial orientation.

Because the football should be stationary during the calibration, the true angular rates should be zero and the true acceleration and magnetic field should be constant. The initial angular rate biases are estimated as the mean of the measured angular rates during the calibration. The variances in Equations (34), (38), and (40) (of each axis of the gyroscope, accelerometer, and magnetometer) are estimated using the variances of the measurements taken during the calibration. The variance of the normalized magnetometer measurements is then calculated using Equation (43), with the magnetic field estimated during the Bias calibration being substituted for $\boldsymbol{m}$.

The magnetic reference vector is then updated. By default, its value is

$$\boldsymbol{n}^N = \begin{bmatrix} 0.5071102631985 63 \\ 0.026081527295963 \\ 0.861486468200513 \end{bmatrix} \tag{83}$$

which is the World Magnetic Model's estimate for College Station, Texas (30°35'27"N, 96°21'42"W) normalized. At calibration, this vector is updated so that its elevation matches the angle between the mean accelerometer and magnetometer measurements (which estimates the angle between gravity and Earth's magnetic field). The azimuth is preserved.

$$\boldsymbol{n}_3^N(new) = \frac{\widetilde{\boldsymbol{m}} \cdot \widetilde{\boldsymbol{a}}}{\sqrt{(\widetilde{\boldsymbol{m}} \cdot \widetilde{\boldsymbol{m}})(\widetilde{\boldsymbol{a}} \cdot \widetilde{\boldsymbol{a}})}} \tag{84}$$

$$\begin{bmatrix} \boldsymbol{n}_1^N \\ \boldsymbol{n}_2^N \end{bmatrix}(new) = \sqrt{\frac{1-\left(\boldsymbol{n}_3^N(new)\right)^2}{\left(\boldsymbol{n}_1^N(old)\right)^2+\left(\boldsymbol{n}_2^N(old)\right)^2}} \begin{bmatrix} \boldsymbol{n}_1^N \\ \boldsymbol{n}_2^N \end{bmatrix}(old) \tag{85}$$

Finally, the initial attitude quaternion is estimated as described in the Attitude Determination subsection.

Initial Conditions

This section gives the initial state and covariance estimates used by the Kalman filters. The EKF and UKF define the state differently, but both contain the football's position, velocity, attitude, and gyroscope biases.

The position is estimated relative to the initial position on any given run. Therefore, by definition, the initial position and its covariance are zero.

The football is assumed to be stationary at the beginning of the run (the pre-throw calibration relies on that assumption). Therefore, the initial velocity and its covariance are also zero.

The initial attitude is estimated during the pre-throw calibration. Its covariance was arbitrarily set at

$$E(\mathrm{d}\boldsymbol{q}_0\mathrm{d}\boldsymbol{q}_0^{\mathrm{T}}) = \begin{bmatrix} 0.000001 & 0 & 0 & 0 \\ 0 & 0.000001 & 0 & 0 \\ 0 & 0 & 0.000001 & 0 \\ 0 & 0 & 0 & 0.000001 \end{bmatrix} \tag{86}$$

This was converted to the relevant portion of the covariance matrix by Equation (90) for the EKF and Equation (118) for the UKF.

The initial gyroscope biases are also estimated during the pre-throw calibration. Their initial covariance is estimated as the covariance of the gyroscope measurements taken during that calibration.

All other cross-covariances are initially zero.

Extended Kalman Filter

State Vector Definition and Covariance Reduction

The EKF's state is the position, velocity (both in the NED frame), orientation (quaternion representing the rotation from the NED coordinate system to the sensor's coordinate system), and gyroscope biases, in that order:

$$\boldsymbol{x} = \begin{bmatrix} \boldsymbol{r} \\ \boldsymbol{v} \\ \boldsymbol{q} \\ \boldsymbol{b}_{\omega} \end{bmatrix} \tag{87}$$

The quaternion poses a problem: due to the quaternion constraint $\boldsymbol{q}^{\mathrm{T}}\boldsymbol{q} = 1$, which differentiates to $\boldsymbol{q}^{\mathrm{T}}\Delta\boldsymbol{q} = 0$, the covariance matrix P must be singular. Due to the accumulation of numerical errors, P is unlikely to remain singular for long. To get around this issue, P is reduced to a 12×12 matrix (losing one row and one column associated with the quaternion) and propagated and filtered following the methodology laid out by Lefferts et al. in [10]. The reduced covariance matrix is

$$\bar{P} = S^{\mathrm{T}}(\hat{\boldsymbol{q}})PS(\hat{\boldsymbol{q}}) \tag{88}$$

where

$$S(\boldsymbol{q}) = \begin{bmatrix} I_{6\times6} & 0_{6\times3} & 0_{6\times3} \\ 0_{4\times6} & \Xi(\boldsymbol{q}) & 0_{4\times3} \\ 0_{3\times6} & 0_{3\times3} & I_{3\times3} \end{bmatrix} \tag{89}$$

The full covariance matrix can be recovered using Equation (90):

$$P = S(\hat{\boldsymbol{q}})\bar{P}S^{\mathrm{T}}(\hat{\boldsymbol{q}}) \tag{90}$$

The quaternion covariance (defined in Equation (60)) is

$$O = \Xi(\hat{\boldsymbol{q}})\bar{P}_{7-9,7-9}\Xi^{\mathrm{T}}(\hat{\boldsymbol{q}}) \tag{91}$$

Propagation

The disturbance is defined as

$$\boldsymbol{u}_k = \hat{\boldsymbol{a}}_k \tag{92}$$

and the process noise as

$$\boldsymbol{w}_k = \begin{bmatrix} \mathrm{d}\boldsymbol{v}_k \\ \mathrm{d}\boldsymbol{a}_k \\ \boldsymbol{\eta}_{\omega_k} \\ \boldsymbol{\eta}_{b_k} \end{bmatrix} \tag{93}$$

From Equations (51), (52), (53), and (48), it can be shown that:

$$\Phi = \begin{bmatrix} I_{3\times3} & \Delta t_k I_{3\times3} & 0_{3\times4} & 0_{3\times3} \\ 0_{3\times3} & I_{3\times3} & 0_{3\times4} & 0_{3\times3} \\ 0_{4\times3} & 0_{4\times3} & \Phi_k^{qq} & -\dfrac{\sin\dfrac{\hat{\theta}_k}{2}}{\hat{\theta}_k}\Delta t\Xi(\hat{\boldsymbol{q}}_k^+) \\ 0_{3\times3} & 0_{3\times3} & 0_{3\times4} & I_{3\times3} \end{bmatrix} \tag{94}$$

$$\Gamma_k = \begin{bmatrix} \dfrac{\Delta t_k^2}{2} I_{3\times3} \\ \Delta t_k I_{3\times3} \\ 0_{7\times3} \end{bmatrix} \tag{95}$$

$$\Upsilon_k = \begin{bmatrix} \Delta t_k I_{3\times3} & \dfrac{\Delta t_k^2}{2} I_{3\times3} & 0_{3\times3} & 0_{3\times3} \\ 0_{3\times3} & \Delta t_k I_{3\times3} & 0_{3\times3} & 0_{3\times3} \\ 0_{4\times3} & 0_{4\times3} & -\dfrac{\sin\dfrac{\hat{\theta}_k}{2}}{\hat{\theta}_k}\Delta t_k\Xi(\hat{\boldsymbol{q}}_k^+) & 0_{4\times3} \\ 0_{3\times3} & 0_{3\times3} & 0_{3\times3} & \Delta t_k I_{3\times3} \end{bmatrix} \tag{96}$$

As the noise of each sensor is assumed to uncorrelated, the process noise covariance is

$$Q_k = \begin{bmatrix} P_{k_{4-6,4-6}} + 10^{-6} I_{3\times3} & 0 & 0 & 0 \\ 0 & Q_{a_k} + 10^{-8} I_{3\times3} & 0 & 0 \\ 0 & 0 & Q_\omega + 10^{-8} I_{3\times3} & 0 \\ 0 & 0 & 0 & Q_b \end{bmatrix} \tag{97}$$

where $P_{k_{4-6,4-6}}$ is the portion of the covariance matrix corresponding to velocity (the fourth through sixth rows of the fourth through sixth columns of P) and Q_ω, Q_b, and Q_a are given by Equations (34), (36), and (59) respectively. The identity matrix terms are an attempt to capture the noise introduced by the zero-order hold assumption; their values were chosen arbitrarily.

The covariance matrix is propagated using modified versions of the state transition matrix and the matrix multiplying the process noise:

$$\bar{\Phi}_k = \begin{bmatrix} I_{3\times3} & \Delta t_k I_{3\times3} & 0_{3\times3} & 0_{3\times3} \\ 0_{3\times3} & I_{3\times3} & 0_{3\times3} & 0_{3\times3} \\ 0_{3\times3} & 0_{3\times3} & C(\hat{\boldsymbol{q}}_{k+1}^-) C^{\mathrm{T}}(\hat{\boldsymbol{q}}_k^+) & -\frac{\Delta t}{2} I_{3\times3} \\ 0_{3\times3} & 0_{3\times3} & 0_{3\times3} & I_{3\times3} \end{bmatrix} \tag{98}$$

$$\bar{\Upsilon}_k = S^{\mathrm{T}}(\hat{\boldsymbol{q}}_k^+)\Upsilon_k = \begin{bmatrix} \Delta t_k I_{3\times3} & \frac{\Delta t_k^2}{2} I_{3\times3} & 0_{3\times3} & 0_{3\times3} \\ 0_{3\times3} & \Delta t_k I_{3\times3} & 0_{3\times3} & 0_{3\times3} \\ 0_{3\times3} & 0_{3\times3} & -\frac{\sin\frac{\hat{\theta}_k}{2}}{\hat{\theta}_k} \Delta t_k I_{3\times3} & 0_{3\times3} \\ 0_{3\times3} & 0_{3\times3} & 0_{3\times3} & I_{3\times3} \end{bmatrix} \tag{99}$$

$$\bar{P}_{k+1}^- = \bar{\Phi}_k \bar{P}_k^+ \bar{\Phi}_k^{\mathrm{T}} + \bar{\Upsilon}_k Q_k \bar{\Upsilon}_k^{\mathrm{T}} \tag{100}$$

Filtering

The measurement model is given by Equation (39). Applying Equations (41) and (12),

$$\boldsymbol{h}(\boldsymbol{x}) = C(\boldsymbol{q})\boldsymbol{n}^N \tag{101}$$

so

$$H_k = \left[0_{3\times 6} \quad \frac{\partial C(\boldsymbol{q})}{\partial q_0}\boldsymbol{n}^N \quad \frac{\partial C(\boldsymbol{q})}{\partial q_1}\boldsymbol{n}^N \quad \frac{\partial C(\boldsymbol{q})}{\partial q_2}\boldsymbol{n}^N \quad \frac{\partial C(\boldsymbol{q})}{\partial q_3}\boldsymbol{n}^N \quad 0_{3\times 3}\right]_{\widehat{\boldsymbol{q}}_k^-} \tag{102}$$

$$\frac{\partial C(\boldsymbol{q})}{\partial q_0} = 2q_0 I_{3\times 3} + 2\underline{q}_\times \tag{103}$$

$$\frac{\partial C(\boldsymbol{q})}{\partial q_1} = 2\begin{bmatrix} q_1 & q_2 & q_3 \\ q_2 & -q_1 & q_0 \\ q_3 & -q_0 & -q_1 \end{bmatrix} \tag{104}$$

$$\frac{\partial C(\boldsymbol{q})}{\partial q_2} = 2\begin{bmatrix} -q_2 & q_1 & -q_0 \\ q_1 & q_2 & q_3 \\ q_0 & q_3 & -q_2 \end{bmatrix} \tag{105}$$

$$\frac{\partial C(\boldsymbol{q})}{\partial q_3} = 2\begin{bmatrix} -q_3 & q_0 & q_1 \\ -q_0 & -q_3 & q_2 \\ q_1 & q_2 & q_3 \end{bmatrix} \tag{106}$$

The measurement noise covariance *R* is given by Equation (42).

After propagating the state and covariance, the filter calculates a modified measurement sensitivity matrix using Equation (107), then the modified Kalman gain using Equation (108). The standard Kalman gain is recovered using Equation (109), then the state is filtered using Equation (10) and the covariance using Equation (110).

$$\bar{H}_k = H_k S(\widehat{\boldsymbol{q}}_k^-) \tag{107}$$

$$\bar{K}_k = \bar{P}_k^- \bar{H}_k^{\mathrm{T}} \left(\bar{H}_k \bar{P}_k^- \bar{H}_k^{\mathrm{T}} + R_k\right)^{-1} \tag{108}$$

$$K_k = S(\widehat{\boldsymbol{q}}_k^-)\bar{K}_k \tag{109}$$

$$\bar{P}_k^+ = (I_{9\times 9} - \bar{K}_k \bar{H}_k)\bar{P}_k^- \tag{110}$$

Unscented Kalman Filter

State Vector Definition

The state vector is defined as

$$\boldsymbol{x}_k \equiv \begin{bmatrix} \boldsymbol{r}_k \\ \boldsymbol{v}_k \\ \delta\boldsymbol{p}_k \\ \boldsymbol{b}_\omega \end{bmatrix} \tag{111}$$

where $\delta\boldsymbol{p}_k$ is a Modified Rodrigues Parameter (MRP) representation of the attitude error. The attitude quaternion itself is a separate variable; after each timestep, the quaternion is updated and the attitude error reset to zero. The relationship between a set of MRPs and a quaternion is given by

$$\boldsymbol{p} = \frac{\underline{q}}{1 + q_0} \tag{112}$$

or, inversely,

$$q_0 = \frac{1 - \boldsymbol{p} \cdot \boldsymbol{p}}{1 + \boldsymbol{p} \cdot \boldsymbol{p}} \tag{113}$$

$$\underline{q} = (1 + q_0)\boldsymbol{p} \tag{114}$$

The MRPs have a singularity when the quaternion represents a rotation of 180°, which is why it is used to represent an attitude error and not the attitude itself.

The covariance of the quaternion, which is required to calculate Q_a, is calculated by approximating the quaternion error as

$$\mathrm{d}\boldsymbol{q} \approx M\Delta\boldsymbol{p} \tag{115}$$

where

$$M \equiv \frac{\partial \mathrm{d}\boldsymbol{q}}{\partial \Delta \boldsymbol{p}}\Big|_{\Delta p = 0_{3\times 1}} = 2\Xi(\widehat{\boldsymbol{q}}) \tag{116}$$

Thus, the quaternion covariance is

$$O = E(\mathrm{d}\boldsymbol{q}_0 \mathrm{d}\boldsymbol{q}_0^{\mathrm{T}}) \approx 4\Xi(\widehat{\boldsymbol{q}}) E(\Delta \boldsymbol{p} \Delta \boldsymbol{p}^{\mathrm{T}}) \Xi^{\mathrm{T}}(\widehat{\boldsymbol{q}}) \tag{117}$$

Inversely,

$$P_{7-9,7-9} = E(\Delta \boldsymbol{p} \Delta \boldsymbol{p}^{\mathrm{T}}) = \frac{1}{4} \Xi^{\mathrm{T}}(\widehat{\boldsymbol{q}}) E(\mathrm{d}\boldsymbol{q}_0 \mathrm{d}\boldsymbol{q}_0^{\mathrm{T}}) \Xi(\widehat{\boldsymbol{q}}) \tag{118}$$

Propagation

The lower triangular matrix from the Cholesky decomposition is used as the square root in Equation (16).

The propagation matrices are:

$$\Phi = \begin{bmatrix} I_{3\times 3} & \Delta t_k I_{3\times 3} & 0_{3\times 3} & 0_{3\times 3} \\ 0_{3\times 3} & I_{3\times 3} & 0_{3\times 3} & 0_{3\times 3} \\ 0_{3\times 3} & 0_{3\times 3} & I_{3\times 3} & -\Delta t I_{3\times 3} \\ 0_{3\times 3} & 0_{3\times 3} & 0_{3\times 3} & I_{3\times 3} \end{bmatrix} \tag{119}$$

$$\Gamma_k = \begin{bmatrix} \frac{\Delta t_k^2}{2} I_{3\times 3} \\ \Delta t_k I_{3\times 3} \\ 0_{6\times 3} \end{bmatrix} \tag{120}$$

$$\Upsilon_k = \begin{bmatrix} \Delta t_k I_{3\times 3} & \frac{\Delta t_k^2}{2} I_{3\times 3} & 0_{3\times 3} & 0_{3\times 3} \\ 0_{3\times 3} & \Delta t_k I_{3\times 3} & 0_{3\times 3} & 0_{3\times 3} \\ 0_{3\times 3} & 0_{3\times 3} & -\Delta t_k \mathrm{I}_{3\times 3} & 0_{3\times 3} \\ 0_{3\times 3} & 0_{3\times 3} & 0_{3\times 3} & \Delta t_k I_{3\times 3} \end{bmatrix} \tag{121}$$

$\boldsymbol{u}$, $\boldsymbol{w}$, and Q are the same as those used in the EKF. They are given by Equations (92), (93), and (97) respectively.

Filtering

The measurement model and the measurement covariance are the same as those used by the EKF. They are given by Equations (101) and (42), respectively.

CHAPTER V

RESULTS

Hardware and Software Performance

Overall, the demonstration works. The physical design of the football is robust, and the EKF and UKF run in parallel at an update frequency of 100Hz without any problems. The networking works smoothly; data can be printed to the command line in real time and logged to a file. There is a bug that sometimes causes an extra character to be inserted at the end of each message; this is a minor problem that can easily be worked around when post-processing the data, but one that should be addressed in the future.

There is one issue that impacts the performance of the Kalman filters: the magnetometer does not provide reliable measurements. This may be because VectorNav sensors correct the magnetometer measurement based on location, but the VN-100 has no GPS and therefore no way of knowing its location. In any case, the measurements do not match the model (Equation (39)), which affects the attitude determination algorithm, the Bias calibration, and attitude filtering in both Kalman filters. The rest of this section describes the steps that were taken to mitigate the magnetometer issue.

A new routine was implemented to estimate the accelerometer biases. It assumes that the VN-100 is stationary and perfectly flat and assumes a gravitational acceleration of $9.793404 m/s^2$, which was calculated based on the latitude and altitude (82m) of College Station. 100 accelerometer measurements are taken and the biases are estimated as their average plus $9.793404 m/s^2$ in the z-axis.

The magnetometer measurement is the only one used in the filtering phase of the Kalman filters. Its effect on the filters was eliminated by multiplying the Kalman gains by zero, so the

Kalman filters could estimate their position attitude only by propagating the accelerometer and gyroscope measurements. Without any filtering the covariance can only increase with time.

The attitude determination algorithm was left as it was. The algorithm uses the accelerometer to determine the elevation and the magnetometer to determine the heading, so the elevation was considered to be reliable but the heading was not. For this reason, the trajectories in the Vicon Tests section are plotted in two dimensions (vertical and horizontal) instead of three.

The rest of this chapter evaluates the Kalman filters' performance based on three tests: one run during which the football was stationary, two during which it was swung on a pendulum, and three during which it was thrown.

Stationary Test

One test was performed during which the football was kept stationary. It was expected that the attitude estimates would drift chaotically over time due to the gyroscope noise but remain close to the original attitude. This was the case for the UKF (Figure 12) but the EKF showed a steady drift of about 2° per second (Figure 11). Such a constant drift suggests an error in the EKF that has not been found as of the writing of this thesis, possibly related to the gyroscope bias.

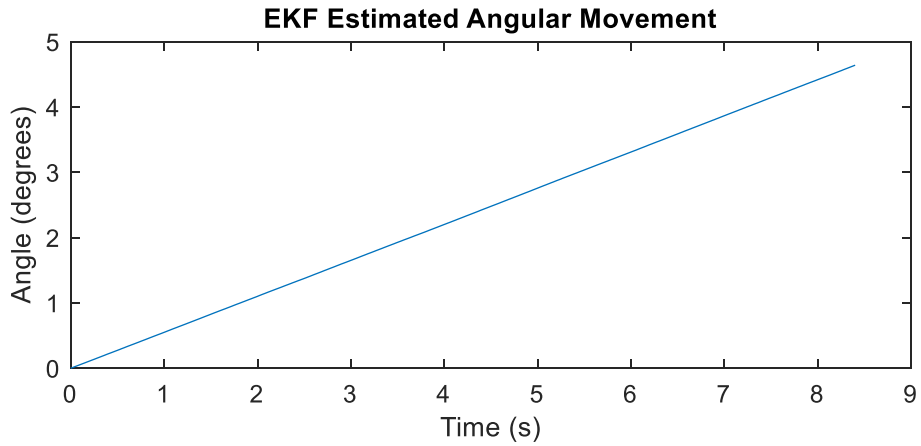

Figure 11: EKF Attitude Drift

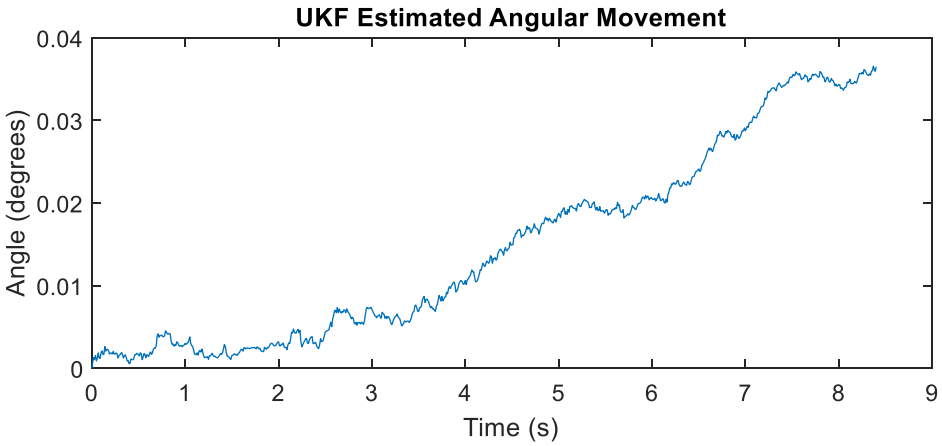

Figure 12: UKF Attitude Drift

Figure 13 and Figure 14 shows how the position and velocity estimates, respectively, from the EKF and UKF drifted during the test. Over such a short period of time, the drift was expected to be similar for both filters, but the EKF had much more drift in the x and y components. The cause of this is unknown and would require further investigation. The UKF drift and EKF z-component drift showed a linear drift in velocity and parabolic drift in position, which suggests that the accelerometer bias estimate was incorrect. Given the makeshift nature of the bias calibration used, this is no surprise.

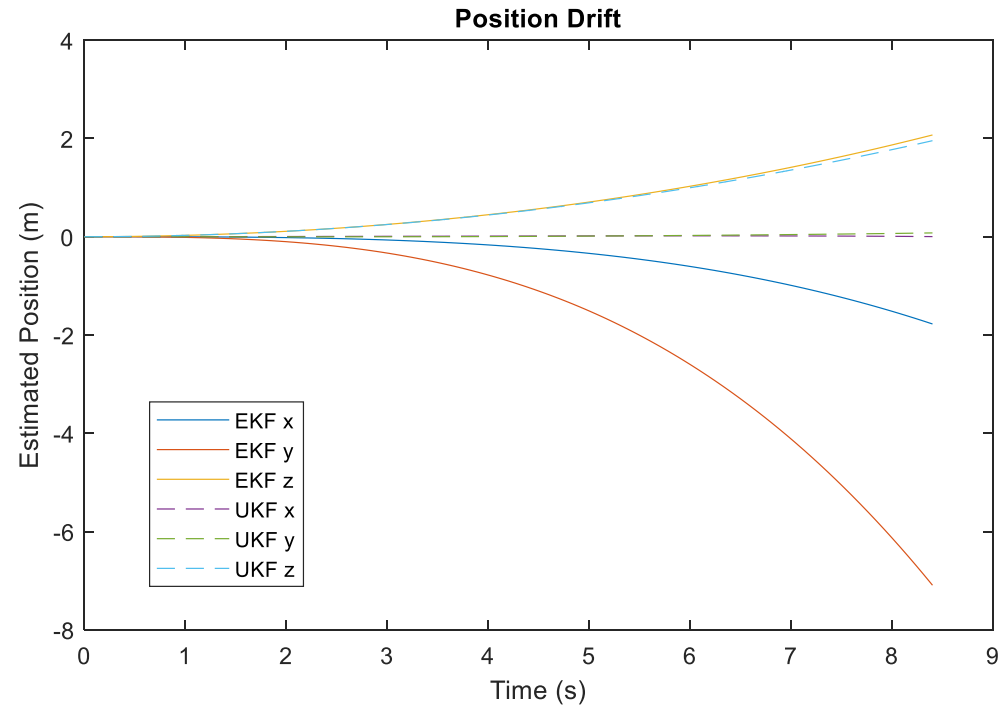

Figure 13: Position Drift

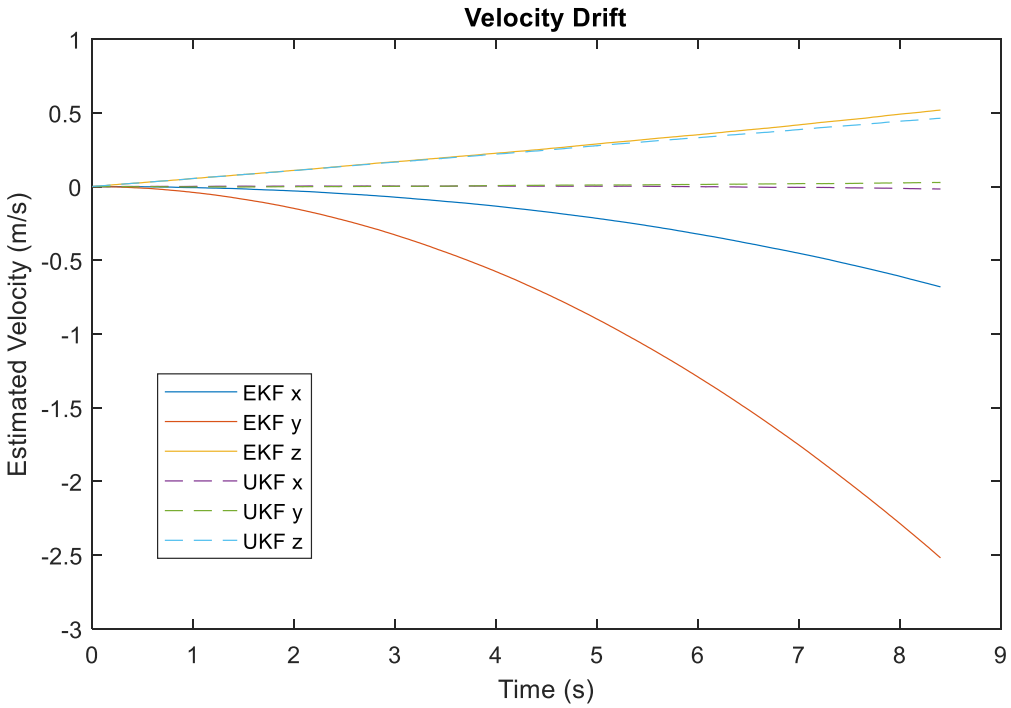

Figure 14: Velocity Drift

Figure 15 shows the uncertainties from the stationary test. The position, velocity, and bias uncertainties are simply the corresponding diagonal elements of the covariance matrix; the angular uncertainties are defined as

$$\sigma_\alpha \equiv \sqrt{E(\mathrm{d}\alpha^2)} \tag{122}$$

where α is defined on Page 19. From Equation (28), it can be shown that

$$\sigma_\alpha^2 \approx E(\mathrm{d}q_1^2) + E(\mathrm{d}q_2^2) + E(\mathrm{d}q_3^2) \tag{123}$$

These expected values are taken from the quaternion covariance O, which is calculated using Equation (91) for the EKF and Equation (117) for the UKF.

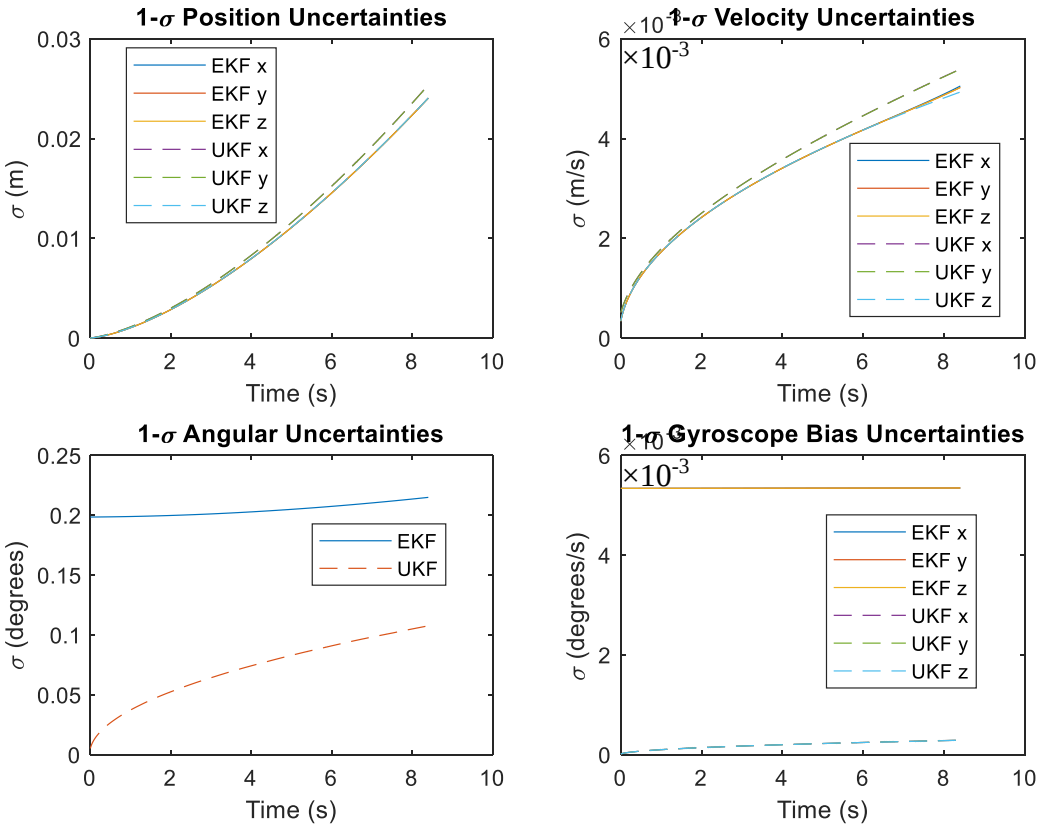

Figure 15: Stationary Test Uncertainties

As expected, the covariances always increase because the estimates are not being filtered. The position and velocity uncertainties are much smaller than the actual errors, most likely because of the error in the accelerometer bias estimates (and the potential bug in the EKF). Without improving the accelerometer bias estimates, this issue could be fixed by adding terms into the process noise covariance that take these bias estimation errors into account. A more major issue is that the UKF has a bug that causes the covariance matrix to drop almost to zero on the first timestep, as can be seen in the angular and gyroscope bias uncertainties. Despite extensive searching, this bug has not yet been found as of the time of writing.

Pendulum Tests

Two runs were performed during which the football was swung on a pendulum. The primary purpose of this test was to evaluate the attitude estimates' drift when the football is in motion. Two of the bolts holding the two halves of the football shell together were threaded through a rope, and the rope was tied to a bar near the ceiling of the LASR Lab. The distance from the bar to the center of the football was about 377cm. During each test, the football was placed on a chair to keep it still during the pre-throw calibration and set in motion after the calibration's completion.

Accelerometer measurements were used to determine which vector in the football's coordinate system pointed down when the pendulum was in its equilibrium state. The pendulum angle is defined as the angle between this equilibrium down vector and down at any given time. Even though the rope was free to twist, the pendulum angle is expected to oscillate sinusoidally with a period of about 3.9s; its local minimum in any given oscillation should be zero. Figure 16 and Figure 17 show the pendulum angles estimated during the two pendulum tests.

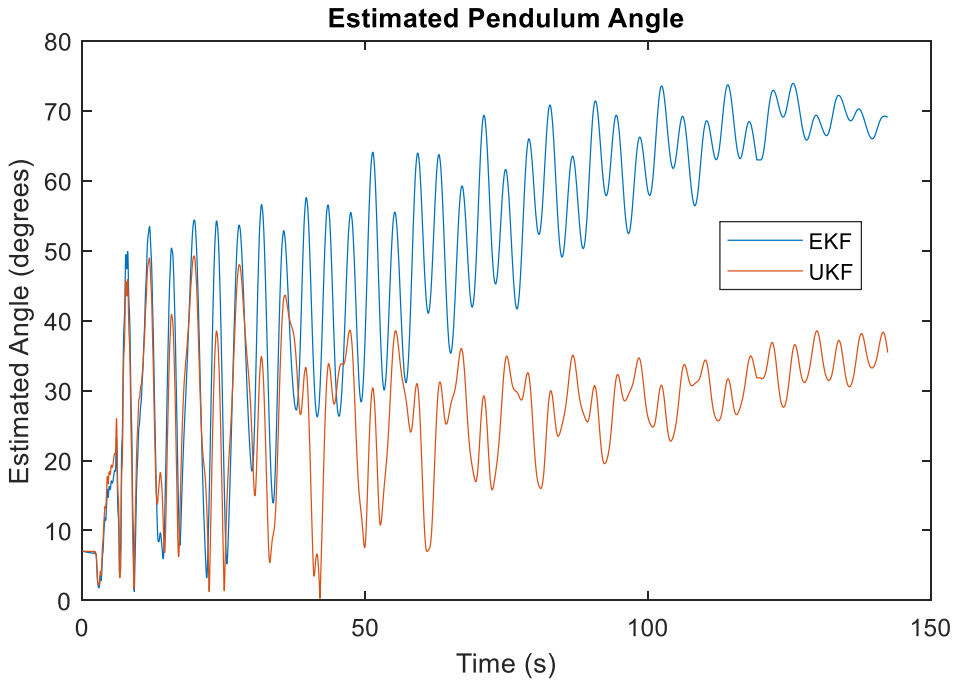

Figure 16: Pendulum Test #1

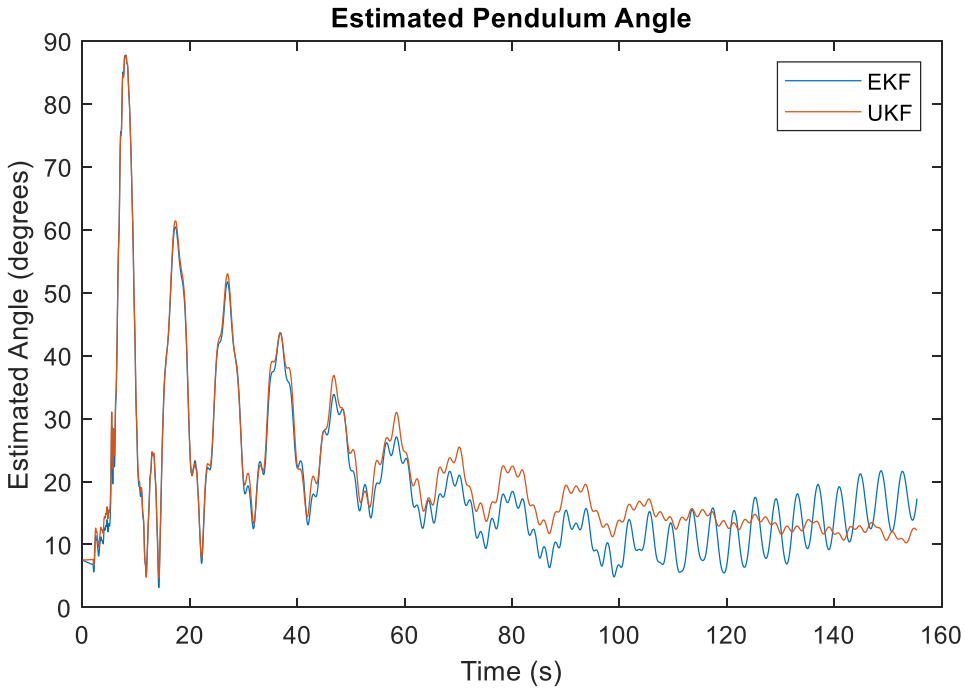

Figure 17: Pendulum Test #2

The estimated pendulum angles oscillate with approximately the expected period of 3.9s, but the oscillation is not exactly sinusoidal, especially during the second test. This may be due to unintended oscillations between the football and the rope; a rigid pendulum with a rigid attachment to the football would probably solve that problem. The minimum estimated pendulum angle, which would ideally remain near zero, drifts considerably, especially during the first test. The EKF performed much worse than the UKF during the first test, possibly because of whatever error caused the drift in the stationary test.

Figure 18 shows the angular certainties (as defined in Equation (122)) from the pendulum tests. The uncertainties are much less than the drift observed in the estimated pendulum angles. One possible cause of this discrepancy is underestimation of the process noise associated with the zero-order hold assumption.

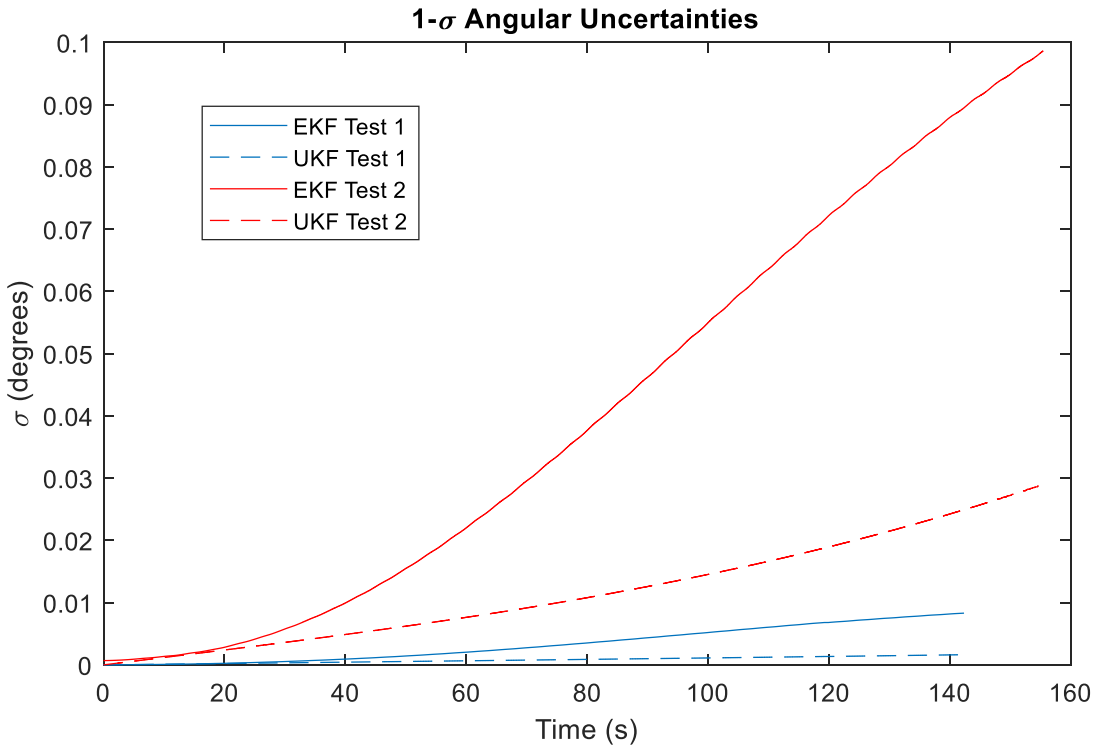

Figure 18: Pendulum Test Uncertainties

Vicon Tests

Three runs were performed during which the football was thrown and its position tracked by the LASR Lab's Vicon system. The primary purpose of this test was to evaluate the filters' position propagation while the football is in motion. The Vicon system consists of a series of cameras that tracks a set of plastic balls covered in reflective tape. The software, assuming the balls are attached to a rigid object, can determine the object's position and attitude if enough balls are visible to enough cameras. Five balls were attached to a small piece of plywood that was attached to the football by one of the screws holding the two halves of the shell together.

On the following pages, four figures are shown for each of the three throws. The first figure in each set (Figure 19, Figure 23, and Figure 27) shows the acceleration measured during the throw. The second in each set (Figure 20, Figure 24, and Figure 28) and third in each set (Figure 21, Figure 25, and Figure 29) show the trajectories estimated by the EKF and UKF and measured by the Vicon system—the third is zoomed in to show the measured trajectory more clearly. The fourth figure in each set (Figure 22, Figure 26, and Figure 30) shows the uncertainties during the throw (the same ones shown for the stationary test).

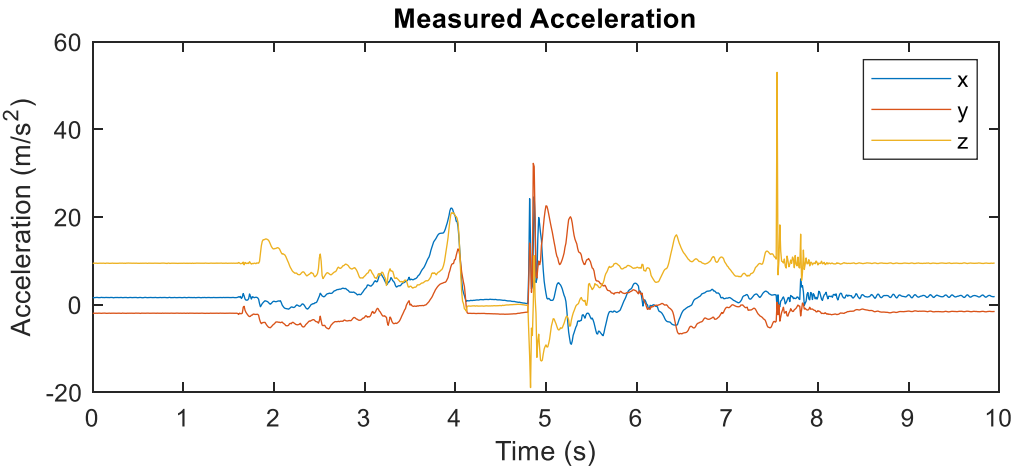

Figure 19: Throw 1 Acceleration

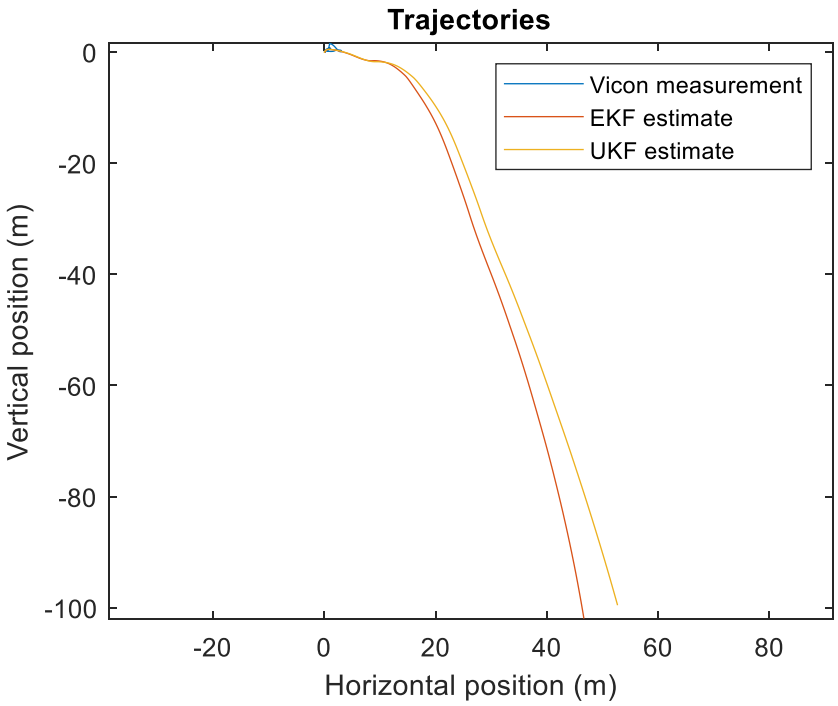

Figure 20: Throw 1 Estimated Trajectories

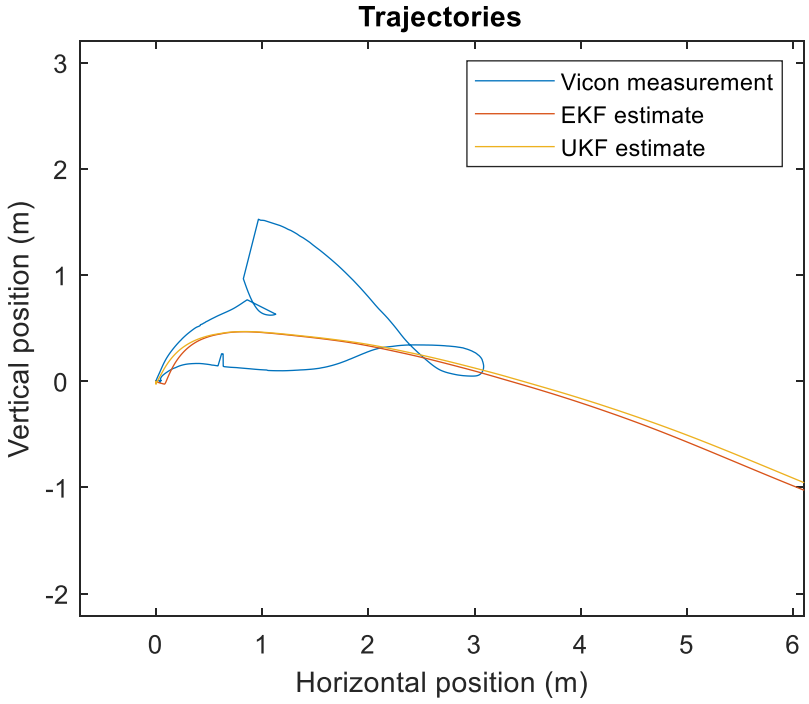

Figure 21: Throw 1 Vicon Trajectory

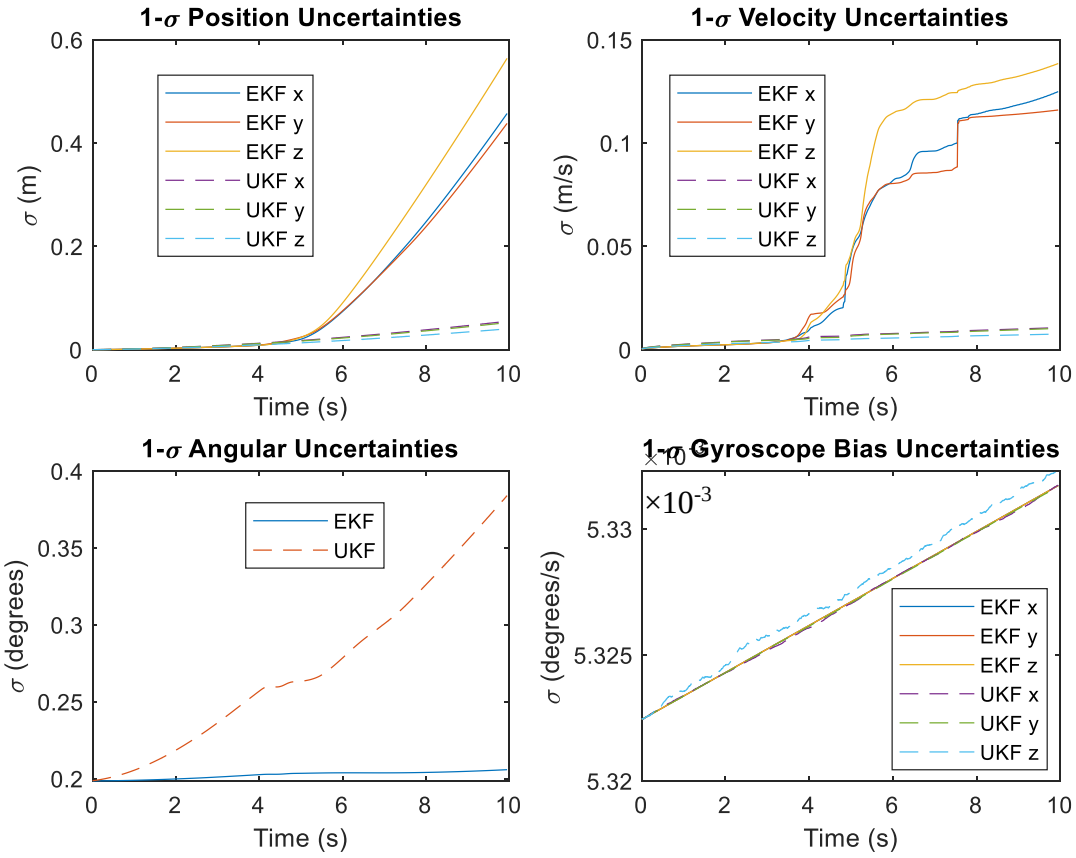

Figure 22: Throw 1 Uncertainties

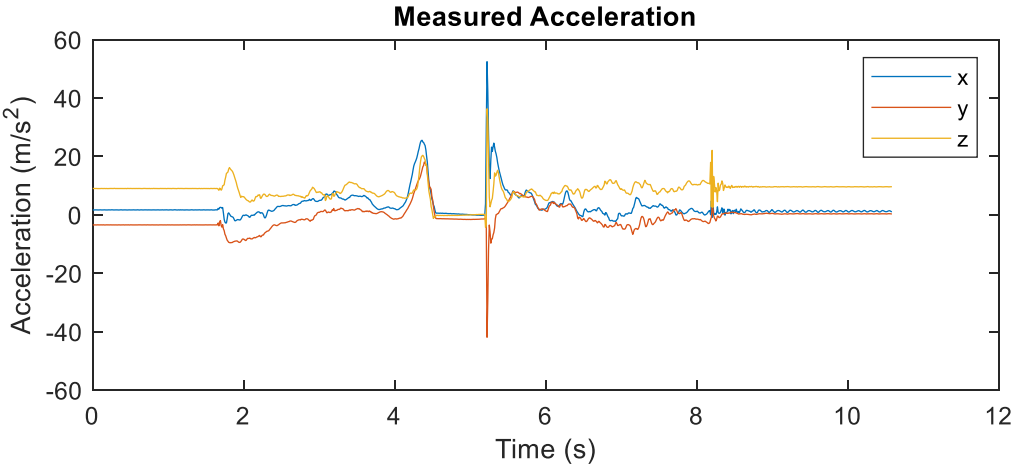

Figure 23: Throw 2 Acceleration

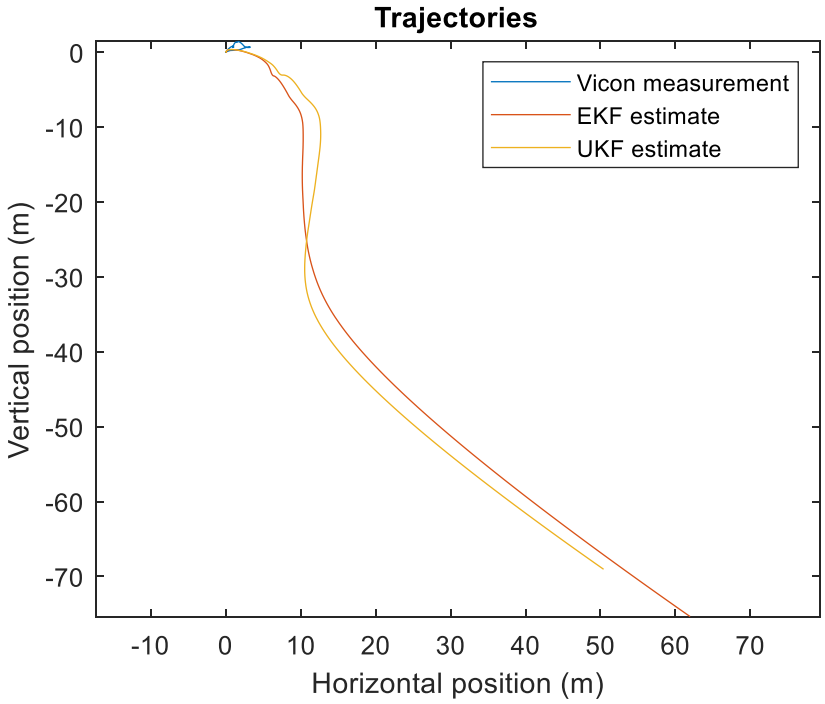

Figure 24: Throw 2 Estimated Trajectories

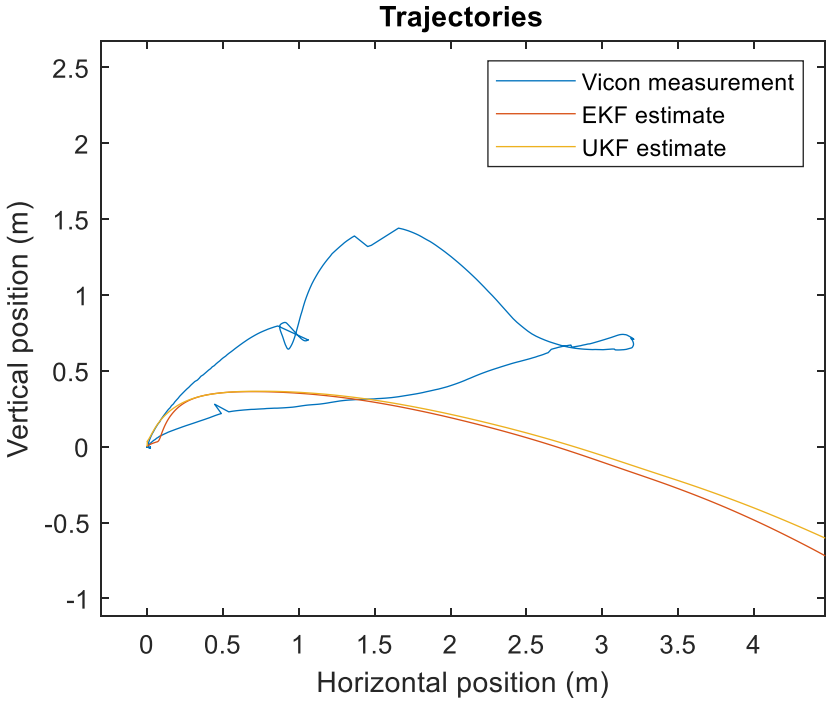

Figure 25: Throw 2 Vicon Trajectory

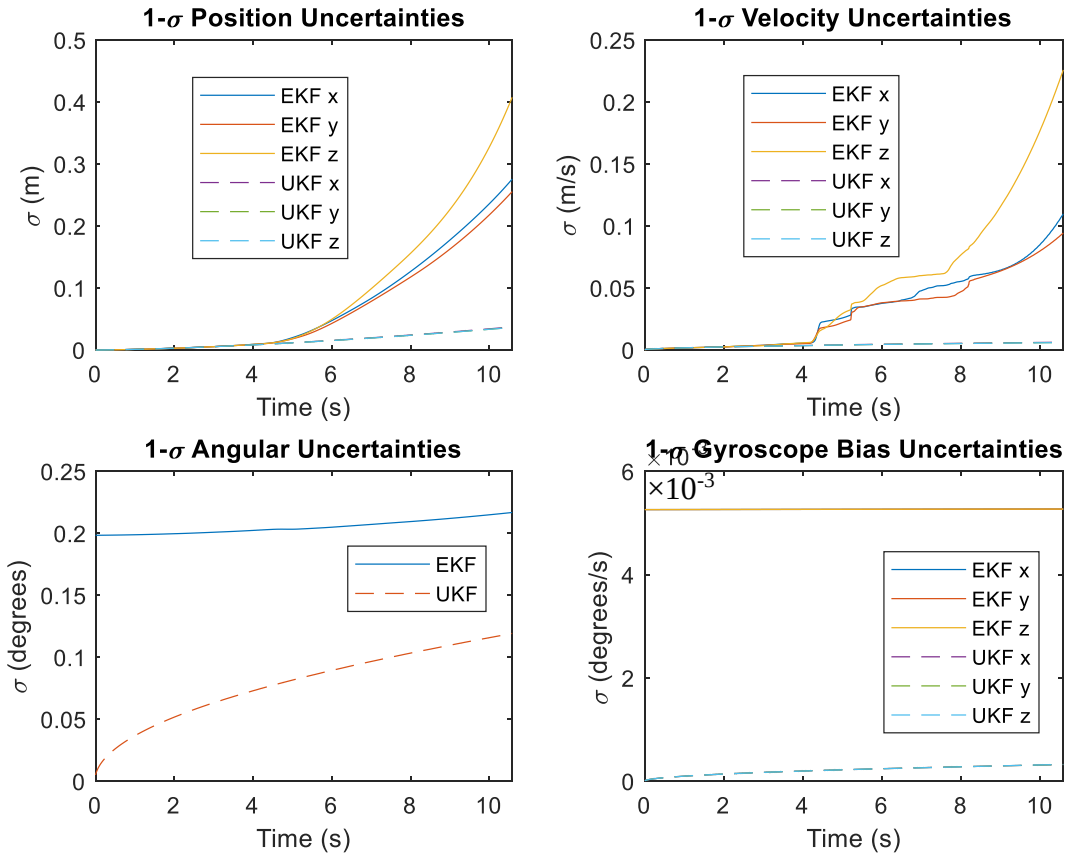

Figure 26: Throw 2 Uncertainties

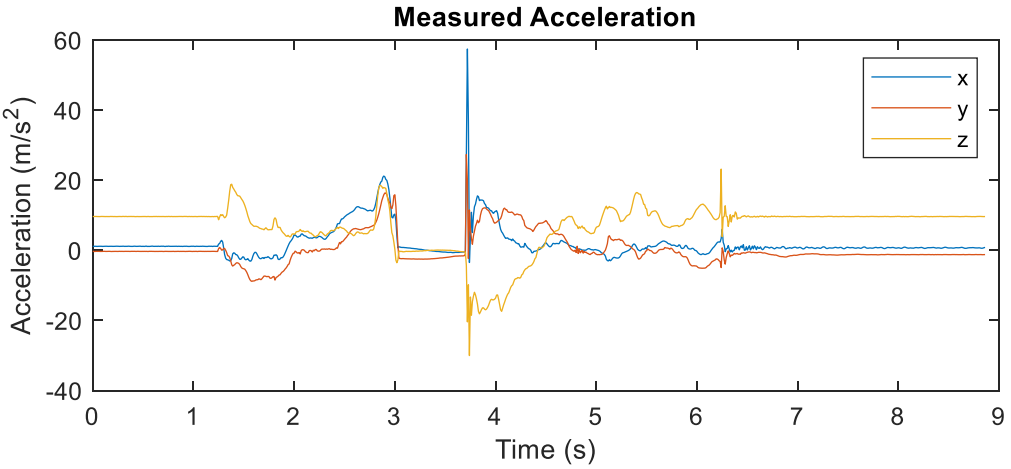

Figure 27: Throw 3 Acceleration

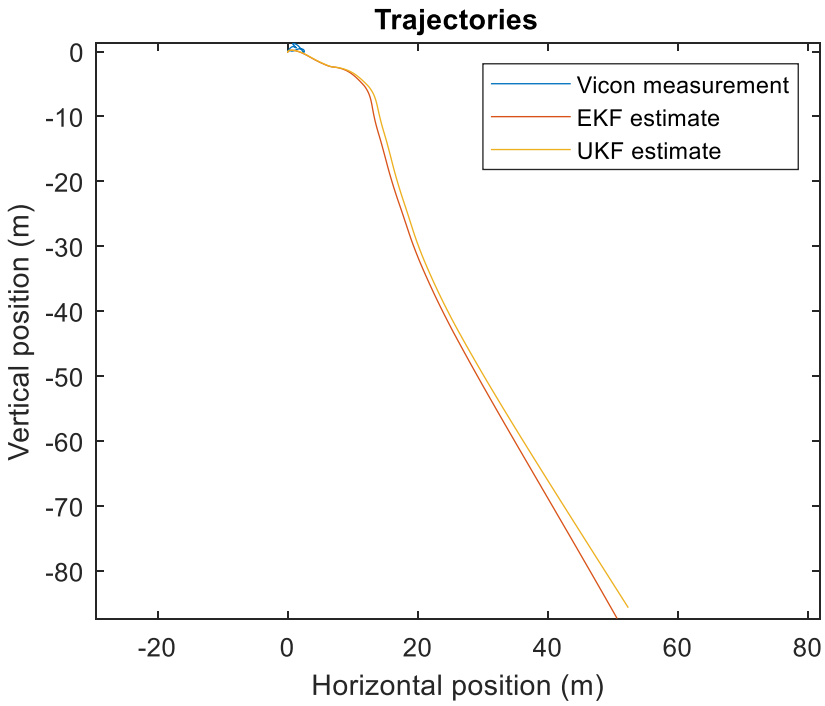

Figure 28: Throw 3 Estimated Trajectories

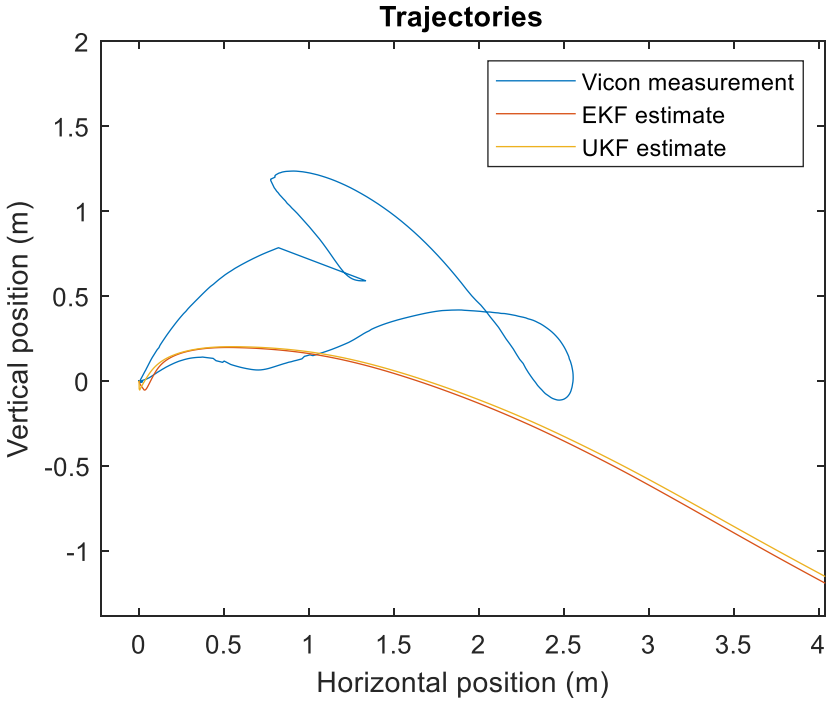

Figure 29: Throw 3 Vicon Trajectory

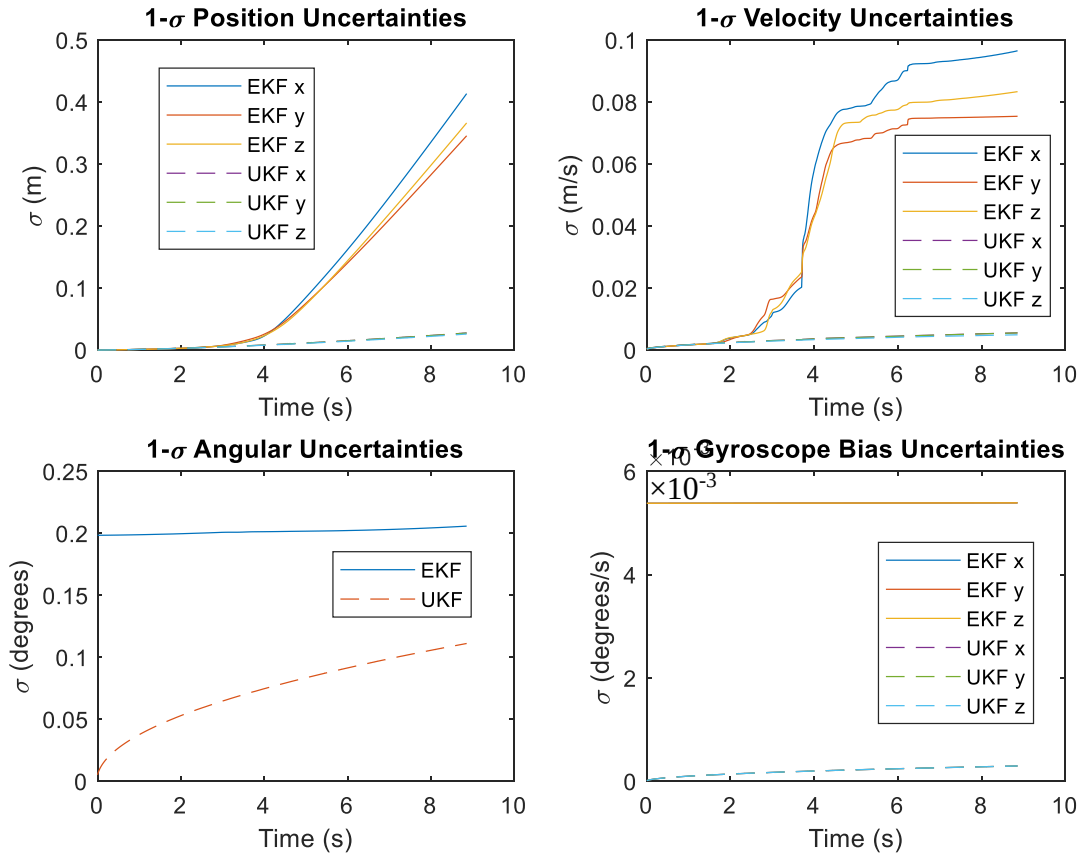

Figure 30: Throw 3 Uncertainties

The sharp rises in the EKF velocity covariances, which causes the sharp rise in the EKF position covariances, correspond with sharp spikes in the measured acceleration. This correlation makes sense because high accelerations increase the acceleration portion of the process noise covariance, which directly affects the velocity covariance. The UKF should have similar rises in its covariances; their absence is another sign that something is wrong with the UKF covariance implementation.

The position estimates degrade so fast that they are absolutely useless after the first few seconds of motion. The attitude drift observed in the stationary and pendulum tests may explain

why the position estimate is so bad: when the attitude estimate is not accurate and the football is not in free fall, the measured acceleration due to the force holding it up does not line up with gravity, causing the estimated position to fall rapidly. The position and velocity covariances are much smaller than the errors because, as the stationary and pendulum tests showed, the attitude error is underestimated.

Throw 1 had a pleasant surprise: unlike in every other run analyzed for this thesis, the UKF's covariance did not set itself near zero at the beginning of the run. Without knowing what causes that issue, it is impossible to say why it did not occur in that run.

CHAPTER VI

CONCLUSIONS

The networked embedded system architecture was demonstrated successfully. Other than a minor issue of extra characters being added to the end of messages, the software and networking capabilities work without any problems and can be used in future projects or to test new algorithms. The hardware also works, except that the VN-100's magnetometer is not reliable. Additionally, the physical design of the football has two minor shortcomings. Firstly, the battery switch and cables are only accessible by opening the football. The demonstration would be easier to use if the battery charging cable and a power switch were accessible through holes in the shell. Secondly, the cables inside the football, especially the VN-100 cable, are much longer than necessary and add noticeable weight to the device. This could be remedied by splicing shorter cables and/or obtaining a custom cable from VectorNav.

The Kalman filters were demonstrated less successfully. In addition to them being handicapped by the lack of any reliable measurements, multiple bugs are apparent from the tests performed—one or two that cause the EKF's attitude and velocity to drift and another that resets the UKF's covariance matrix on the first timestep. After a few seconds, both filters' position and velocity estimates and the EKF's attitude estimates are useless. However, other than these mathematical errors, the Kalman filters work—they do not crash and they were successfully integrated with the VectorNav software library and Open MPI. Therefore, they provide templates on which future Extended and Unscented Kalman filters can be built.

The bias calibration routine could not be fully tested because of the magnetometer issue, but it appeared to work and did not crash. The other calibration routines worked and, like the Kalman filters, can be used as templates for future calibration routines.

Some work would have to be done to be able to demonstrate the Kalman filters successfully. First, the bugs would have to be found and fixed. Then, the filters would need some tuning—some of the initial conditions and noise parameters used to generate the data in this thesis were chosen based on educated guesses rather than any data. Finally, the filters will only be reliable over a long period of time if they have access to a position measurement and attitude measurements with at least three degrees of freedom (e.g. two vectors or three angles).

REFERENCES

[1] C. Ruiz, X. Chen, L. Zhang, and P. Zhang, "Demo Abstract: Collaborative localization and navigation in heterogeneous UAV swarms," in *14th ACM Conference on Embedded Networked Sensor Systems, SenSys 2016, November 14, 2016 - November 16, 2016*, Stanford, CA, United States, 2016: Association for Computing Machinery, Inc, in Proceedings of the 14th ACM Conference on Embedded Networked Sensor Systems, SenSys 2016, pp. 324-325, doi: 10.1145/2994551.2996544. [Online]. Available: http://dx.doi.org/10.1145/2994551.2996544

[2] J. Wang and J. Dang, "Modeling and simulation of traffic characteristics of vehicular ad-hoc network," *Journal of Computational Methods in Sciences and Engineering*, vol. 15, no. 3, pp. 507-513, 2015, doi: 10.3233/JCM-150563.

[3] "VN-100 User Manual." VectorNav Technologies. https://www.vectornav.com/docs/default-source/documentation/vn-100-documentation/vn-100-user-manual-(um001).pdf?sfvrsn=b49fe6b9_32 (accessed 16 June, 2019).

[4] "Open MPI: Open Source High Performance Computing." https://www.open-mpi.org/ (accessed 16 June, 2019).

[5] "Downloads." VectorNav Technologies. https://www.vectornav.com/support/downloads (accessed 16 June, 2019).

[6] B. G. Jacob, Gaël. "Eigen." http://eigen.tuxfamily.org (accessed 16 June, 2019).

[7] V. M. Fico *et al.*, "Implementing the Unscented Kalman Filter on an embedded system: A lesson learnt," in *2015 IEEE International Conference on Industrial Technology, ICIT*

2015, March 17, 2015 - March 19, 2015, Seville, Spain, 2015, vol. 2015-June: Institute of Electrical and Electronics Engineers Inc., in Proceedings of the IEEE International Conference on Industrial Technology, June ed., pp. 2010-2014, doi: 10.1109/ICIT.2015.7125391. [Online]. Available: http://dx.doi.org/10.1109/ICIT.2015.7125391

[8] J. L. Crassidis and F. L. Markley, "Unscented filtering for spacecraft attitude estimation," *Journal of Guidance, Control, and Dynamics*, vol. 26, no. 4, pp. 536-542, 2003, doi: 10.2514/2.5102.

[9] J. E. Hurtado, *Kinematic and Kinetic Principles*. 2016.

[10] E. J. Lefferts, F. L. Markley, and M. D. Shuster, "Kalman Filtering for Spacecraft Attitude Estimation," *Journal of Guidance, Control, and Dynamics*, vol. 5, no. 5, pp. 417-429, 1982.