A Hardware-Minimal Unscented Kalman Filtering Framework for Visual-Inertial Navigation of Small Unmanned Aircraft

Virginia Tech

Joshua Galen Eddy

April 20, 2017

Abstract

This thesis presents the formulation and implementation of an Unscented Kalman Filter (UKF) framework for fusion of visual and inertial sensors in unmanned aircraft navigation. Specifically, we fuse sensor readings from a 3-axis accelerometer, 3-axis gyroscope, and a Simultaneous Localization and Mapping (SLAM) algorithm to estimate the pose of an aircraft, such as a quadcopter, capable of hovering flight. We discuss the formulation of our UKF fusion algorithm, the development of a Robot Operating System (ROS) software package implementing the algorithm, and several experiments in which this algorithm was used to track the motion of a physical vehicle simulating hovering flight in an indoor environment. We verify the filter's effectiveness by comparing its output to that of a Vicon motion capture system. We then discuss possible applications of this system and future work which may build upon the results developed herein.

Acknowledgments

This work would not have been possible were it not for my graduate adviser and committee chair, Dr. Kevin Kochersberger. He has been a constant source of support since October 2012 when he provided my friends and me with lab space in which to design our first competition robots. Over the ensuing five years, Dr. Kochersberger has played a pivotal role in my development as an engineer and researcher. He demonstrates his commitment to nurturing students every day by giving tirelessly of his time and energy.

It is also my pleasure to thank Dr. Danette Allen of the NASA Langley Autonomy Incubator, who provided me with two summer internships in her lab and taught me much of what I know today about Kalman filtering. Dr. Allen supplied me with both funding and equipment in order to perform this research, even going so far as to allow me to use her lab's motion capture facilities for my experiments.

I would also like to thank Dr. Loc Tran and Mr. James Neilan of NASA Langley and Ralph Williams of Analytical Mechanics Associates, who collectively provided me with hours of programming guidance during my time at the Autonomy Incubator.

Special thanks also go to Samuel Rubenking, who stuck with me to the end. Thanks, Sam.

Contents

List of Figures

Acronyms

AR Augmented Reality

EKF Extended Kalman Filter

GPS Global Positioning System

KF Kalman Filter

IMU Inertial Measurement Unit

MSF-EKF Multi-Sensor Fusion Extended Kalman Filter

PTAM Parallel Tracking and Mapping

SLAM Simultaneous Localization and Mapping

ROS Robot Operating System

UAS Unmanned Aircraft System

UAV Unmanned Aerial Vehicle

UKF Unscented Kalman Filter

UT Unscented Transform

UTM Unmanned Traffic Management

VIN Visual-Inertial Navigation

Chapter 1

Introduction

This document contains several sections with titles placed in parentheses. These sections are included to give added context to this thesis, but are not strictly necessary to the reader's understanding of the technical material being presented.

1.1 (Personal Motivation)

I first took an interest in unmanned aircraft in the fall of 2012, my sophomore year of college. In search of an exciting engineering challenge, several of my friends and I founded the Cooperative Autonomous Robotics Design (CARD) team at Virginia Tech. Our core team consisted of a dozen students devoted to designing and competing with drones and other robotic vehicles. Our team, guided by my future graduate adviser Dr. Kevin Kochersberger, entered a number of design competitions and brought home several awards for the university. My early experiences with the team brought me into contact with microcontroller programming, Proportional-Integral-Derivative (PID) controller design, mechatronics, and computer-aided design (CAD) modeling.

After two years of involvement with the CARD team, I applied for an internship at the National Institute of Aerospace[1] (NIA) in Hampton, Virginia. In the summer of 2014,

[1] http://www.nianet.org

I was part of a team of NIA researchers working on the Flying Donkey Challenge[2], an international engineering competition centered around the idea of "flying donkeys," full-sized autonomous airplanes capable of quickly carrying cargo between small airports in rural Africa. This competition, unfortunately now defunct, was divided into a number of sub-challenges focusing on different technical objectives such as precision landing and collision avoidance. Our team's goal was to design an inexpensive navigation system that could reliably guide unmanned aircraft during a Global Positioning System (GPS) blackout. This project introduced me to many of the technologies and techniques that would later become my major research interests, particularly the Robot Operating System[3] (ROS), Kalman Filtering, and sensor fusion.

Through my internship at the NIA, I met Dr. Danette Allen, head of the NASA Langley Autonomy Incubator. During my 2014–15 academic year, Dr. Allen sponsored the CARD team to design and build two autonomous multirotor delivery drones. These aircraft were capable of delivering 5-lb packages to distances of up to 2.5 miles (or 5 miles, round trip). In addition, these vehicles were able to land precisely on 1 m^2 April tags such as that found in Figure 4.2[4]. Following the completion of this project, I worked as a summer intern at the Autonomy Incubator.

During the summer of 2015, I began the research that eventually evolved into my thesis project, studying Visual-Inertial Navigation (VIN) and the Unscented Kalman Filter (UKF). As I read more and more on these subjects, I became interested in the design of the algorithm underlying the UKF. Unlike many other formulations of the Kalman Filter, the UKF has a notably limited dependence on information about the system under scrutiny (this *system agnosticism* is discussed in more detail in Chapter 3). The more I learned about the UKF, the more I became excited about the idea of taking advantage of this limited dependence trait to build a minimalistic software interface by which a wide variety of disparate systems could be tracked and studied in a ROS framework. I envisioned a

[2] http://www.flyingdonkey.org

[3] http://wiki.ros.org

[4] http://wiki.ros.org/apriltags_ros

"one-stop shopping" experience for massively reusable and customizable filtering profiles that could fulfill the needs of researchers and roboticists who may have little knowledge of state estimation techniques. This vision eventually drove my development of the `kalman_sense` ROS package, cementing my interest in unmanned aerial vehicle (UAV) state estimation processes and controls.

1.2 Organization of this Document

Prior Work

In Prior Work, we explore recent contributions to loosely coupled filter-based navigation and state estimation processes. We focus primarily on a number of impactful publications coming from ETH Zurich's Autonomous Systems Lab (ASL) and the University of Pennsylvania's GRASP Lab. We define the current state of the art in filter-based navigation and establish the research context in which my thesis exists.

Algorithm Design and Implementation

Because of the algorithmic nature of state estimation processes, we explore in detail the design and implementation of the `kalman_sense` ROS package. We discuss plant model abstraction as well as code organization and data flow and then summarize the process by which one could extend `kalman_sense`'s functionality and the advantages of system-agnostic algorithm design.

Experimental Design

In this section, we first establish the goals of the testing regimen and then discuss the real-world execution of these goals. We discuss important statistical methods for characterizing the system's effectiveness as well as data collection procedures and post-processing. The system's physical testing infrastructure is explored in detail.

Experimental Results

In Experimental Results, we evaluate the system's performance during testing and seek out any limiting factors that influence estimation accuracy. We probe for possible improvements to the algorithm and provide a notional understanding of the system's theoretical effectiveness in real-world scenarios.

Conclusions

We briefly summarize the contributions made in this thesis, the effectiveness of the `kalman_sense` package, and any insights acquired during programming and testing.

Future Work

In Future Work, we expand upon the possible improvements proposed in Experimental Results and also offer a number of applications for the algorithms and processes developed herein. Specific examples of heterogeneous fleet management and unmanned traffic management (UTM) are explored.

Chapter 2

Prior Work

2.1 Development of Unscented Kalman Filter

Since the late 1990s, the Unscented Kalman Filter (UKF) has been a frequent topic of interest in the field of guidance, navigation, and controls. This extension of the Kalman Filter (explored in detail in Chapter 3) appeals to roboticists and controls engineers because it allows the preservation of nonlinear behavior both in the state propagation and measurement prediction steps. Through its use of the unscented transform (discussed further below), the UKF retains not only the first and second moments of the state distribution—the mean and covariance—but also the third moment, the skew. It is this retention of information, coupled with true nonlinear behavior, that makes the Unscented Kalman Filter such a desirable tool for motion tracking and localization.

In [1], Julier and Uhlmann presented a nonlinear estimation approach for the Kalman Filter, originally developed in Uhlmann's doctoral dissertation. Recognizing that most applications for autonomous navigation are fundamentally nonlinear in both their dynamics and their observation models, Julier and Uhlmann proposed the use of a set of discretely sampled "sigma points" to determine the mean and covariance of a probability distribution. By recasting the prediction and correction steps of the Kalman Filter in the form of unscented transforms (UTs), this new filter eliminates the need to calculate

Jacobian matrices. Julier and Uhlmann argued that for this reason their formulation was easier to implement than the EKF and went on to suggest that its use could supplant the EKF in virtually all applications, linear or nonlinear.

In [2], Julier acknowledges that the (linear) Kalman Filter has been used successfully in many nonlinear scenarios, but notes that the use of only the first two moments of the state estimate sigma points results in the neglect of all higher order information (that is, third-order moments, or "skew"), a potentially rich source of new and useful information relating to symmetry of the state estimate. By extending the sigma point selection scheme of the conventional unscented transform, Julier was able to present a tractable but computationally complex extension of the Kalman Filter that could predict not only the first two moments of a sigma point distribution but also the skew. Though formulated initially for unimodal distributions, Julier stated that the approach could, with additional mathematical considerations, be generalized for use with multimodal distributions. Julier's contention was that the use of higher order information could promote better performance levels in autonomous vehicle navigation. The utility of maintaining and utilizing higher order information through the use of skewed filtering was assessed in a realistic tracking scenario. However, the results were somewhat disappointing as the change in performance was only marginal, presumably due to the linearity of the filter's update rule. Accordingly, research in this area continues, including examination of the use of nonlinear update rules in the filtering process.

In [3], Julier describes a novel approach to modifying the unscented transformation state estimation method. In the new approach, Julier takes the additional step of introducing a framework for scaling sigma points as part of the state estimation process. The general framework of the new methodology allows preservation of the first two moments of any set of sigma points, thus providing a construct for limiting values to either the conventional unscented transform or the modified (scaled) transform. Providing detailed mathematical validations, Julier demonstrates that the new scaling algorithm is computationally manageable in that it is, in essence, little more than the conventional unscented

transformation algorithm with the addition of a simple post-processing step, the only difference being the inclusion of an extra correction term. Thus, the new algorithm's computational and storage costs are similar to that of the non-scaled transformation. The performance level of the scaled UT is thus demonstrably superior to the unscaled UT for propagating the two lower-order moments of a sigma point distribution.

In [4] Julier and Uhlmann discuss the application of the EKF as an estimation algorithm and the associated difficulties in doing so. Because the EKF is fundamentally a linearizing approach to estimation, its effectiveness is thus tied to the veracity of the local linearity assumption for the system under scrutiny. These limitations led to the development of the UT for nonlinear applications. In this paper, Julier and Uhlmann describe the UT and its benefits, including easier implementation and improved accuracy. The UT offers greater accuracy and reliability by applying higher order information, using sigma point sampling, to the traditional mean and covariance information associated with linear applications. The authors provide examples, which may be tailored to various process and observation models, that show how the UT overcomes the limitations of the EKF.

2.2 Visual-Inertial Navigation

Recent advancements in computing power have given rise to a plethora of once-inaccessible algorithms for vision and localization. Simultaneous Localization and Mapping (SLAM) algorithms have been one of the chief beneficiaries of this technological bounty. For example, the rise of Graphics Processing Units (GPUs) has allowed for tremendous increases in the speed of SLAM as well as Visual Odometry (VO) programs. Improved state estimation methods, such as the UKF, have grown in popularity at the same time. The convergence of these research avenues lies in Visual-Inertial Navigation (VIN), an approach to navigation using any number of the various approaches to fusing either raw camera data or the outputs of vision algorithms with inertial sensor readings. Donavanik et al. provide a concise survey of the state of the art at the time of this writing in [5]. In this

article the researchers discuss many of the current challenges in robotic VIN, including many stemming from the use of particular SLAM algorithms and EKF-based frameworks such as consistency of state estimates over time. From the article,

> This problem [of consistency] concerns the EKF as it uses linearization of a non-linear model, which introduces errors that in turn will lead to inconsistency.
>
> Over time, this error can accumulate. A possible mitigation is to use the Unscented Kalman Filter (UKF).

From here on, we will briefly explore recent advancements in VIN, paying particular attention to filter-based methods and their accompanying sensor frameworks.

In [6], Georg Klein and David Murray proposed a method for tracking a handheld camera in unknown environments for use in small augmented reality (AR) workspaces. In contrast to many previous SLAM-based approaches to camera tracking, Klein and Murray split the tracking and mapping functions into two separate computational tasks. They performed these tasks on a dual-core computer utilizing parallel threads, with one thread directly tracking erratic motion of the handheld camera and the other thread constructing a 3D map of the environment. Through the use of this Parallel Tracking and Mapping (PTAM) algorithm, Klein and Murray were able to take advantage of computationally expensive batch-optimization techniques for map reconstruction, techniques which were rarely used in real-time applications previously. This, in turn, allowed Klein and Murray to forego the common approach of creating a sparse map of high quality features in favor of a much denser map containing features that could vary widely in quality. The resulting system could produce detailed maps tracking thousands of features at frame-rate and could recover gracefully from a variety of intermittent tracking failures. That being said, the researchers made certain relaxing assumptions regarding the scenes to be tracked. PTAM, by nature of its orientation toward AR applications, operates best in small, static, planar environments (such as on the surface of a desk or the floor of an office). PTAM's value to the robotics community quickly became obvious due to its independence of

a priori knowledge of the scene and its minimal initialization procedure (explored in Chapter 4) .

write up PTAM init procedure in Exp Design

Four years later, in [7], Stephan Weiss et al. presented a visual-inertial navigation system for autonomous UAV navigation which employed PTAM. The researchers presented the results of several experiments in which a UAV equipped with a single monocular camera and an IMU navigated through unknown environments without the aid of GPS satellites or other external sensing infrastructure. All calculations were performed in real time using an EKF framework, proving that this minimalist combination of sensors could be employed in real-world GPS-compromised flight scenarios to great effect. At approximately the same time, Shen et al. conducted experiments in [8] at the University of Pennsylvania GRASP Lab aimed at stable indoor flight and GPS-denied localization in constrained multi-floor environments with a similarly limited suite of purely onboard sensors. The research distinguishes itself by emphasizing the use of onboard sensors only, as well as fully autonomous, real-time internal computational capabilities, with no hands-on user interaction beyond basic high-level commands. The research extends to multi-floor UAV navigation with loop closure. It also addresses specially designed controllers to help compensate for sudden changes in wind velocity and air flow as the UAV traverses constrained low-clearance areas with potentially strong aerodynamic disturbances.

In [9], Weiss and Siegwart went on to tackle the problem of metric scale in monocular VIN systems. The researchers developed a general algorithm that provides metric scale to monocular visual odometry and monocular SLAM systems using inertial measurement unit (IMU) data. The authors accomplished the development of the metric scale by the addition of a 3-axis accelerometer and 3-axis gyroscope to track 3D motion and to correct visual scale drift. Weiss and Siegwart created a modular solution based on their existing EKF framework and provided both simulated and empirical results. Weiss and Seigwart provided in-depth analysis of their system's applications, versatility, and reliability for real-time visual odometry and SLAM.

In 2012, Weiss et al. built upon this metric scale algorithm to present a versatile sensor

fusion framework for autonomous flight [10]. Due to latency, noise, and arbitrary scaling within the output of a UAV's sensors, it is both impractical and ill-advised to incorporate this sensor output for position control without calibration or post-processing. They address these problems using an EKF-SLAM formulation which fuses pose measurements with inertial sensors. The researchers not only estimate pose and velocity of the UAV, but also estimate the sensor biases, correct the scale of position measurements, and perform inter-sensor self-calibration in real time. Their research demonstrates that the proposed framework is capable of running entirely onboard a UAV and performing state prediction at a rate of 1 kHz. Their results illustrate that this approach is able to handle measurement delays (up to 500 ms), sensor noise (with positional standard deviation up to 20 cm), and slow update rates (as low as 1 Hz) while still allowing dynamic maneuvers. Weiss et al. present a detailed quantitative performance evaluation of the system under the influence of different disturbance parameters and different sensor setups to highlight the versatility of their approach. That same year, Weiss et al. further developed their system in [11], adding a speed-estimation module to the framework which fuses IMU and vision data to turn the monocular camera into a metric body-speed sensor. They then demonstrated how this module could be used for self-calibration of the UAV's onboard sensor suite in real time.

Shortly thereafter, Huang et al. presented solutions in [12] to two UKF limitations that exist in current state-of-the-art SLAM systems. Specifically, the researchers addressed the problems of cubic complexity in the number of state pose estimates, and the inconsistencies in those estimates caused by a mismatch between the observability properties of statistically-linearized UKF systems and the observability properties of nonlinear systems. To address the problem of cubic complexity, they introduced a novel sampling strategy which produces a constant computational cost. This sampling method, while linear in the prediction phase, is quadratic in the update phase. Although this new sampling strategy was primarily proposed for resolving the cubic complexity SLAM problem, the researchers stressed that it has potential usefulness in other nonlinear estimation applications. To

address the problem of inconsistency in state estimations, Huang et al. proposed a new UKF algorithm which, due to the imposition of observability constraints, ensures that the linear regression computations of the modified UKF system produce results similar to those of nonlinear SLAM systems and, in the process, provide improved accuracy and consistency in state estimations. Importantly, the researchers validated their results with both real-world and simulation experiments.

In 2013, Lynen et al. presented a generic framework in [13] based on the EKF-SLAM system developed in [10] which was shown to be more robust during sensor blackouts and to be self-correcting in scale. The researchers demonstrated that their Multi-Sensor-Fusion EKF (MSF-EKF) framework was capable of processing measurements from an unlimited number of sensors, as well as sensor types, while simultaneously performing automatic self-calibrations of the overall sensor suite. It was the design of this software framework, which the researchers released as open source software shortly after publication, that inspired many of the design decisions behind the `kalman_sense` package.

In [14], Rogers et al. presented a methodology for overcoming some of the constraining conditions encountered in a GPS-guided autonomous robotic system, such as occlusion (blocking of GPS signals) and multipath (reception of indirect signals due to environmental reflections) and potentially to ameliorate the effects of jamming or spoofing resulting from adversarial activities. Specifically, the methodology incorporated GPS measurements into a feature-based mapping system, thus providing geo-referenced coordinates allowing for better execution of high-level missions and providing the ability to correct accumulated mapping errors over the course of long-term operations in both indoor and outdoor environments.

In [15], Faessler et al. reported on the development and demonstration of a low-cost, low-weight, vision-based quadrotor UAV with onboard sensing, computation, and control capabilities. These onboard capabilities eliminated reliance on external positioning systems such as GPS or motion capture systems. This development moved the UAV from its current line-of-sight control state to wireless communications with the ability

to execute intricate processes autonomously and to transmit live feedback to a user. Reporting on both indoor and outdoor experiments, the researchers believed that such a vehicle potentially would be a great enhancement in search-and-rescue missions, disaster response, and remote inspection of terrain.

Chapter 3

Algorithm Design and Implementation

3.1 UKF Formulation

3.1.1 Prediction Step

We begin by defining the following quantities:

$$\mathbf{p} = \{x, y, z\}^T \tag{3.1}$$

$$\mathbf{q} = \{q_x, q_y, q_z, q_w\}^T \tag{3.2}$$

$$\mathbf{v} = \{\dot{x}, \dot{y}, \dot{z}\}^T \tag{3.3}$$

$$\mathbf{\Omega} = \{\omega_x, \omega_y, \omega_z\}^T \tag{3.4}$$

$$\mathbf{a} = \{\ddot{x}, \ddot{y}, \ddot{z}\}^T \tag{3.5}$$

The vectors **p**, **v**, and **a** represent the vehicle's position, velocity, and acceleration, respectively. The quaternion[1] **q** represents the vehicle's orientation and the vector $\mathbf{\Omega}$ represents

[1] From here on, all quaternion quantities are represented according to the convention $\mathbf{q} = \{q_x, q_y, q_z, q_w\}^T$, where q_w is the scalar component and is always placed in the last position. This convention was chosen to maintain consistency with the Eigen library's internal representation of quaternions.

the vehicle's angular rates. We now define the state vector $\mathbf{x}$ as

$$\mathbf{x} = \left\{\mathbf{p}^T, \mathbf{q}^T, \mathbf{v}^T, \mathbf{\Omega}^T, \mathbf{a}^T\right\}^T \tag{3.6}$$

and let $n = 16$ represent the number of state variables.

We now define the following constants:

$$\begin{aligned} \alpha &= 10^{-3} \\ \kappa &= 0 \\ \beta &= 2 \\ \lambda &= \alpha^2 (n + \kappa) - n \end{aligned} \tag{3.7}$$

The constants α and κ control the spread of sigma points chosen within the filter. The value of β governs what distribution is assumed for the states $\mathbf{x}$. Setting $\beta = 2$ is optimal for a Gaussian state distribution, which we assume throughout. With these constants defined, we can now describe the selection of sigma points.

Let $\mathbf{P} \in \mathbb{R}^{n \times n}$ be the covariance matrix associated with the state $\mathbf{x}$. A set of $2n+1$ sigma points is then derived from $\mathbf{x}$ and $\mathbf{P}$ as

$$\begin{aligned} \boldsymbol{\chi}^0_{k-1|k-1} &= \mathbf{x}_{k-1|k-1} \\ \boldsymbol{\chi}^i_{k-1|k-1} &= \mathbf{x}_{k-1|k-1} + \left(\sqrt{(n+\lambda)\,\mathbf{P}_{k-1|k-1}}\right)_i, && i = 1, \ldots, n \\ \boldsymbol{\chi}^i_{k-1|k-1} &= \mathbf{x}_{k-1|k-1} - \left(\sqrt{(n+\lambda)\,\mathbf{P}_{k-1|k-1}}\right)_{i-n}, && i = n+1, \ldots, 2n, \end{aligned} \tag{3.8}$$

where $\boldsymbol{\chi}^i$ is the i-th sigma point and $\left(\sqrt{(n+\lambda)\,\mathbf{P}_{k-1|k-1}}\right)_i$ is the i-th column of the matrix $\sqrt{(n+\lambda)\,\mathbf{P}_{k-1|k-1}}$. The matrix square root $\mathbf{A}$ of a matrix $\mathbf{B}$ is defined here as

$$\mathbf{A}\mathbf{A}^T = \mathbf{B} \tag{3.9}$$

and is computed via Cholesky decomposition.

To predict the next state, these sigma points are propagated through the nonlinear process function f (defined in Section 3.1.3):

$$\boldsymbol{\chi}^i_{k|k-1} = f\left(\boldsymbol{\chi}^i_{k-1|k-1}\right), \quad i = 0, \ldots, 2n. \tag{3.10}$$

These new sigma points are then used to predict the next state and covariance:

$$\hat{\mathbf{x}}_{k|k-1} = \sum_{i=0}^{2n} W_s^i \boldsymbol{\chi}_{k|k-1}^i \tag{3.11}$$

$$\mathbf{P}_{k|k-1} = \left(\sum_{i=0}^{2n} W_c^i \left(\boldsymbol{\chi}_{k|k-1}^i - \hat{\mathbf{x}}_{k|k-1} \right) \left(\boldsymbol{\chi}_{k|k-1}^i - \hat{\mathbf{x}}_{k|k-1} \right)^T \right) + \mathbf{Q}_k, \tag{3.12}$$

where the state weights W_s and covariance weights W_c are defined as

$$\begin{aligned} W_s^0 &= \frac{\lambda}{n+\lambda} \\ W_c^0 &= \frac{\lambda}{n+\lambda} + \left(1 - \alpha^2 + \beta\right) \\ W_s^i &= W_c^i = \frac{1}{2\left(n+\lambda\right)}, \quad i = 1, \ldots, 2n \end{aligned} \tag{3.13}$$

and the process noise covariance matrix $\mathbf{Q}_k$ is defined in Section 3.1.5.

3.1.2 Correction Step

Given the belief defined by $\mathbf{x}_{k|k-1}$ and $\mathbf{P}_{k|k-1}$, we compute $2n+1$ sigma points again as

$$\begin{aligned} \boldsymbol{\chi}_{k|k-1}^0 &= \mathbf{x}_{k|k-1} \\ \boldsymbol{\chi}_{k|k-1}^i &= \mathbf{x}_{k|k-1} + \left(\sqrt{(n+\lambda)\,\mathbf{P}_{k|k-1}} \right)_i, && i = 1, \ldots, n \\ \boldsymbol{\chi}_{k|k-1}^i &= \mathbf{x}_{k|k-1} - \left(\sqrt{(n+\lambda)\,\mathbf{P}_{k|k-1}} \right)_{i-n}, && i = n+1, \ldots, 2n. \end{aligned} \tag{3.14}$$

Next, the sigma points are projected through the observation function h (defined in Section 3.1.4):

$$\boldsymbol{\gamma}_k^i = h\left(\boldsymbol{\chi}_{k|k-1}^i \right), \quad i = 0, \ldots, 2n. \tag{3.15}$$

Let $m = 7$ represent the number of measurements taken from each PTAM message. Each of these messages is read into `kalman_sense` as an m-dimensional measurement vector $\mathbf{z}_k$ of the form

$$\mathbf{z}_k = \left\{ x, y, z, q_x, q_y, q_z, q_w \right\}_{meas.}^T. \tag{3.16}$$

The predicted measurement vector $\hat{\mathbf{z}}_k$ and measurement noise covariance $\mathbf{P}_{zz}$ are then

computed as

$$\hat{\mathbf{z}}_k = \sum_{i=0}^{2n} W_s^i \boldsymbol{\gamma}_k^i \tag{3.17}$$

$$\mathbf{P}_{zz} = \left(\sum_{i=0}^{2n} W_c^i \left(\boldsymbol{\gamma}_k^i - \hat{\mathbf{z}}_k \right) \left(\boldsymbol{\gamma}_k^i - \hat{\mathbf{z}}_k \right)^T \right) + \mathbf{R}, \tag{3.18}$$

where $\mathbf{R}$ is the measurement noise covariance matrix defined in Section 3.1.6. The state-measurement cross-covariance $\mathbf{P}_{xz}$ is then defined as

$$\mathbf{P}_{xz} = \sum_{i=0}^{2n} W_c^i \left(\boldsymbol{\chi}_{k|k-1}^i - \hat{\mathbf{x}}_{k|k-1} \right) \left(\boldsymbol{\gamma}_k^i - \hat{\mathbf{z}}_k \right)^T. \tag{3.19}$$

Next, we compute the Kalman gain $\mathbf{K}_k$ per the definition

$$\mathbf{K}_k = \mathbf{P}_{xz} \mathbf{P}_{zz}^{-1}. \tag{3.20}$$

The corrected state $\hat{\mathbf{x}}_{k|k}$ is then the sum of the predicted state and the innovation, weighted by $\mathbf{K}_k$:

$$\hat{\mathbf{x}}_{k|k} = \hat{\mathbf{x}}_{k|k-1} + \mathbf{K}_k \left(\hat{\mathbf{z}}_k - \mathbf{z}_k \right). \tag{3.21}$$

The corrected covariance $\mathbf{P}_{k|k}$ is the difference between the predicted covariance $\mathbf{P}_{k|k-1}$ and the predicted measurement covariance, weighted by the Kalman gain:

$$\mathbf{P}_{k|k} = \mathbf{P}_{k|k-1} - \mathbf{K}_k \mathbf{P}_{zz} \mathbf{K}_k^T. \tag{3.22}$$

3.1.3 Process Model

To propagate the vehicle's state forward in time, we perform a number of integral operations on the linear accelerations and angular rates measured by the IMU. To determine the orientation of the vehicle at some time k, we integrate the measured angular rate vector $\boldsymbol{\Omega}_{meas.}$ per the relationship

$$\mathbf{q}_k = \mathbf{q}_{k-1} + \frac{1}{2} \Theta \left(\boldsymbol{\Omega}_k \right) \mathbf{q}_{k-1} \Delta t, \tag{3.23}$$

where $\Theta(\mathbf{\Omega}_k) \in \mathbb{R}^{4\times4}$ is the angular rate integration matrix defined by

$$\Theta(\mathbf{\Omega}_k) = \begin{bmatrix} 0 & \omega_z & -\omega_y & \omega_x \\ -\omega_z & 0 & \omega_x & \omega_y \\ \omega_y & -\omega_x & 0 & \omega_z \\ -\omega_x & -\omega_y & -\omega_z & 0 \end{bmatrix} \tag{3.24}$$

and the vehicle's current angular velocity vector is measured directly by the gyroscope:

$$\mathbf{\Omega}_k = \mathbf{\Omega}_{meas.}. \tag{3.25}$$

To calculate the vehicle's current acceleration, we first subtract gravity from the measured acceleration vector $\mathbf{a}_{meas.}$ and then rotate the difference into the inertial frame:

$$\mathbf{a}_k = R_b^g\left(\mathbf{q}_{k-1}\right)\left(\mathbf{a}_{meas.} - \mathbf{g}\right). \tag{3.26}$$

Have Sam look over this. Are subscripts right?

Here $R_b^g\left(\mathbf{q}_{k-1}\right)$ is the rotation matrix representation of the previous orientation quaternion $\mathbf{q}_{k-1}$ and $\mathbf{g}$ is the vector representation of gravity in the global frame.

What is R(q)?

With the vehicle's current acceleration known, we can compute the current velocity of the vehicle by integrating the vehicle's average acceleration between the current and previous time steps:

$$\mathbf{v}_k = \mathbf{v}_{k-1} + \frac{1}{2}\left(\mathbf{a}_k + \mathbf{a}_{k-1}\right)\Delta t. \tag{3.27}$$

Similarly, we compute the vehicle's current position using the vehicle's average velocity:

$$\mathbf{p}_k = \mathbf{p}_{k-1} + \frac{1}{2}\left(\mathbf{v}_k + \mathbf{v}_{k-1}\right)\Delta t. \tag{3.28}$$

Together, Equations 3.23, 3.25, 3.26, 3.27, and 3.28 constitute the nonlinear process function f:

write out f explicitly

3.1.4 Observation Model

Quaternion Continuity Correction

Pseudovelocity Correction

Of the vehicle's states, the acceleration vector $\mathbf{a}$ and angular rate vector $\mathbf{\Omega}$ are measured directly by the IMU:

$$\mathbf{x} = \left\{ \mathbf{p}^T, \mathbf{q}^T, \mathbf{v}^T, \underbrace{\mathbf{\Omega}^T, \mathbf{a}^T}_{IMU} \right\}^T. \tag{3.29}$$

The vehicle's position $\mathbf{p}$ and orientation $\mathbf{q}$ are measured directly by PTAM:

$$\mathbf{x} = \left\{ \underbrace{\mathbf{p}^T, \mathbf{q}^T}_{PTAM}, \mathbf{v}^T, \mathbf{\Omega}^T, \mathbf{a}^T \right\}^T. \tag{3.30}$$

The velocity $\mathbf{v}$, however, is not measured by either. There is no sensor in place by which $\mathbf{v}$ can be directly measured. Velocity must thus be determined through a derived measurement.

Because velocity is predicted via integration of the vehicle's acceleration (Section 3.1.3) and velocity is not directly measured by PTAM, noise in the accelerometer can cause unbounded drift in the velocity. The erroneous velocity is integrated during each succeeding prediction step, causing drift in the estimated position *regardless of corrective position measurements from PTAM*. In this situation, the filter produces position estimates which drift at a constantly increasing rate between PTAM messages. The filter corrects the position each time that a PTAM message arrives only to drift wildly a fraction of a second later when the next IMU message arrives.

To prevent this catastrophic drift, velocity is estimated by numerical differentiation of the vehicle's position during each correction step. When a PTAM message is received, the position reading from the last PTAM message is subtracted from the current position reading. This change in position is then divided by the time step between the two PTAM messages, Δt_P:

$$\mathbf{v}_k = \frac{\mathbf{p}_{k,\,meas.} - \mathbf{p}_{k-1,\,meas.}}{\Delta t_P}. \tag{3.31}$$

This pseudovelocity correction prevents unbounded velocity drift, which in turns makes the filter numerical stable.

3.1.5 Process Noise Covariance Matrix

The process noise covariance matrix $\mathbf{Q}_k \in \mathbb{R}^{n \times n}$ is symmetric, time-dependent, and positive definite. We assume that its nonzero coefficients are a function of multiple integrations between the linear accelerations and angular velocities measured by the vehicle's 3-axis accelerometer and 3-axis gyroscope.

We assume that all axes of the accelerometer and gyroscope exhibit white noise with known standard deviations σ_a and σ_ω, respectively[2]. We further assume that no cross-axis covariance exists between any two accelerometer axes or any two gyroscope axes.

For the sake of convenient estimation, we assume that the accelerometer and gyroscope variance terms grow linearly over each discrete time step Δt. Thus, in the case of the x-axis accelerometer and the x-axis gyroscope, the corresponding terms of $\mathbf{Q}_k$ would be

$$Q_{a_x,a_x} = \sigma_a \Delta t \tag{3.32}$$

and

$$Q_{\omega_x,\omega_x} = \sigma_\omega \Delta t, \tag{3.33}$$

respectively. We can now estimate the other nonzero terms of $\mathbf{Q}_k$ according to their integral relationships with the acceleration and angular velocity measurements.

integral equations

We assume no coupling between the angular quantities ($\mathbf{q}$ and $\mathbf{\Omega}$) and the linear quantities ($\mathbf{p}$, $\mathbf{v}$, and $\mathbf{a}$), making the process noise covariance matrix largely sparse. This assumption logically stems from the small angle assumption. Because the IMU runs at 250 Hz, we assume that the vehicle makes only small rotations between IMU measurements.

full Q matrix

[2]This claim of equal standard deviations between axes is supported by the IMU data sheet.

3.1.6 Measurement Noise Covariance Matrix

The matrix $\mathbf{R} \in \mathbb{R}^{m \times m}$ models the covariance between measurements made by the pose sensor, PTAM. Like the process noise covariance matrix $\mathbf{Q}_k$, the measurement noise covariance matrix $\mathbf{R}$ is symmetric and positive definite. However, for the purposes of this experiment, $\mathbf{R}$ is assumed to be time-invariant due to nearly constant feature density in the experimental scene. Because the vehicle camera moves across the scene at an almost constant height (that is, at a nearly constant distance to the features it observes on the floor), the measurement noise covariance of the pose sensor is assumed to be approximately constant in time. The coefficients of $\mathbf{R}$ were estimated by numerically computing the covariances between raw measurements from PTAM and the "ground truth" supplied by a Vicon motion capture system[3].

The resulting $\mathbf{R}$ matrix is as follows: R Matrix

3.2 Software Design Considerations

Much of the impetus for creating the `kalman_sense` package came from a desire to create a generic UKF framework for estimating the state of an arbitrary system using any number of relative and absolute sensors. To achieve this, the `kalman_sense` package is organized in an object-oriented manner around an overarching abstract class called `UnscentedKf`. This abstract class contains a number of methods performing the different mathematical operations defined in Section 3.1. These methods have been written in a generic manner to enable easy extension of `UnscentedKf` by subclasses containing concrete implementations of various systems. Currently, the package contains only one subclass, known as `QuadUkf`. This subclass contains methods and data structures related directly to estimating the state of a quadcopter or other rotorcraft UAV.

The `UnscentedKf` class encapsulates the generic mathematics of the UKF without knowledge of particular system constraints. This class does little other than matrix math-

[3] https://www.vicon.com/

ematics and is designed to take as input the number of a system's states n and its number of sensors m. With this information, `UnscentedKf` is able to populate a set of mean and covariance weights and intelligently perform all of the requisite linear algebra for the UKF formulation. All other knowledge of particular states, sensors, vehicle geometry, and other metrics is hidden within subclasses such as `QuadUkf`.

`UnscentedKf` behaves in a manner similar to a Java interface in that it requires the extending class to supply functions codifying a process model and an observation model for the system under scrutiny. These two functions, along with n and m, form the entirety of what `UnscentedKf` "knows" about the vehicle. All other details, including the fact that the class is being used in a ROS environment, are hidden from `UnscentedKf`. It is worth noting that `UnscentedKf`'s only dependency is on the Eigen C++ linear algebra library[4].

The subclass (`QuadUkf` for the remainder of this thesis) handles all of the ROS communications for the given system. Specifically, this class has callback functions for receiving sensor data and is responsible for publishing state and covariance estimates.

[4] www.eigen.tuxfamily.org

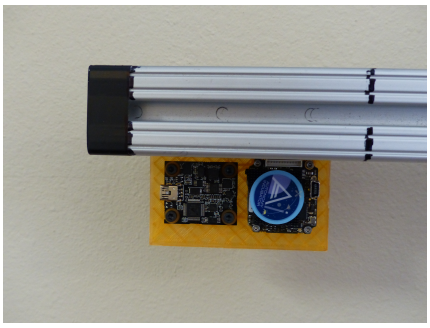
Chapter 4

Experimental Design

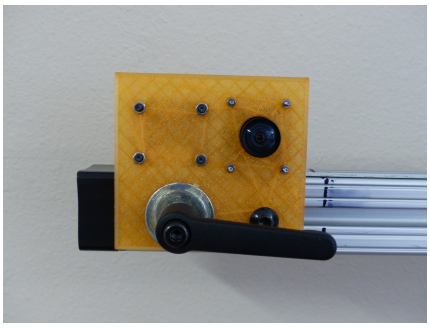
4.1 Testing Considerations

Before venturing further, we should summarize the goals of the UKF framework described previously, paying particular attention to unmanned aircraft system (UAS) operations. This ROS package was designed with the express intent of producing estimates of the position vector **p** and orientation quaternion **q** of a rotorcraft UAV in real time. Thus, the experiments testing `kalman_sense`'s efficacy compare the filter's estimates of position and orientation to the ground truth measured by Vicon.

This system depends upon two sensors: a global-shutter camera and an IMU. The IMU used in this experiment contains a 3-axis accelerometer and 3-axis gyroscope. To simulate both sensors moving through the scene in a manner reminiscent of hovering rotorcraft flight, a rolling test stand was constructed to carry the sensors safely throughout a large motion capture environment. Mounting the sensor suite on a large, steady, level platform allows for a high degree of control over the accelerations and angular velocities felt by the IMU, as well as the motion seen by the ventral camera. In order to validate the UKF framework's effectiveness under ideal conditions, a modern laptop computer with an Intel i7 processor and 16 GB of RAM was used for all computations. The floor of the motion capture environment was strewn with a mixture of April tags and modified Quick



(a) Top view of sensor mount.



(b) Bottom view of sensor mount.

Figure 4.1: The 3D-printed sensor mount. In (a), the Phidgets IMU is on the left and the mvBlueFOX camera is on the right. In (b), the linear braking handle is visible.

Response[1] (QR) codes in order to provide sufficient visual features for PTAM to track.

4.2 Materials

4.2.1 Computation and Sensing

1. One (1) MatrixVision mvBlueFOX-MLC Camera[2]
2. One (1) 1044_0 PhidgetSpatial Precision 3/3/3 High Resolution IMU[3]
3. One (1) Hewlett-Packard Spectre x360 Convertible Laptop 13-ac076nr[4]
4. Two (2) male Mini USB 2.0 to male USB Type A cables

4.2.2 Mobile Test Stand

1. One (1) Oklahoma Sound PRC200 Premium Presentation Cart[5]

[1] https://en.wikipedia.org/wiki/QR_code
[2] https://www.matrix-vision.com/USB2.0-single-board-camera-mvbluefox-mlc.html
[3] http://www.phidgets.com/products.php?product_id=1044
[4] http://store.hp.com/us/en/pdp/hp-spectre-x360---13-ac076nr
[5] http://www.oklahomasound.com/products/product-category/single/?prod=9

2. One (1) 3D-printed Sensor Mount (see Figure 4.1)
3. Two (2) 4" C-Clamps
4. One (1) 1.2-meter 80/20 1515 Rail[6]
5. One (1) 15 Series "L" Handle Linear Bearing Brake Kit[7]
6. One (1) $\frac{5}{16}$-18 × 0.687" Black FBHSCS (Screw)[8]
7. Two (2) Slide-In Economy T-Nuts[9]
8. Three (3) 1" Vicon Infrared Retroreflector Balls
9. Two (2) $\frac{1}{2}$" Vicon Infrared Retroreflector Balls
10. One (1) 0.7-meter Length of $\frac{1}{8}$"-thick Carbon Fiber Rod

4.3 The Experiments

A series of three experiments were designed to characterize the UKF framework's effectiveness in various regimes of motion. The first two experiments consisted of "long walks" in the x-direction and y-direction in order to characterize the accuracy of the filter in lengthy, unidimensional translations. These experiments were meant to determine changes in estimate accuracy over large, planar translations (for example, to uncover the

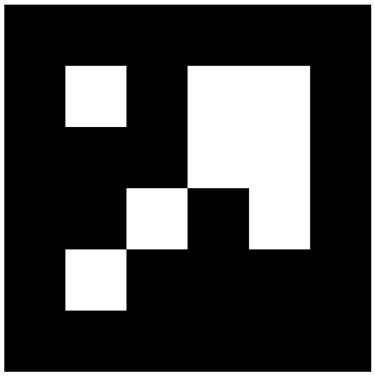


Figure 4.2: Example of an April tag from the `apriltags_ros` ROS package.

[6] https://8020.net/1515.html
[7] https://8020.net/6800.html
[8] https://8020.net/shop/3320.html
[9] Also available at https://8020.net/shop/3320.html

evolution of error in the system over time while effectively manipulating only one state variable). For each long walk, the test stand was translated without rotation along the positive x- and y-axes over distances of approximately seven meters, then returned to the starting location via the same path.

The third experiment was a rectangular translation designed to characterize the system's effectiveness when translated along two axes. Again, the cart was translated without rotation around the corners of a nearly square rectangle having sides approximately four meters in length.

Three data streams were collected for analysis using the `rosbag`[10] ROS data recording utility.

more explanation of post processing

[10] http://wiki.ros.org/rosbag

Chapter 5

Experimental Results

Get some results

Chapter 6

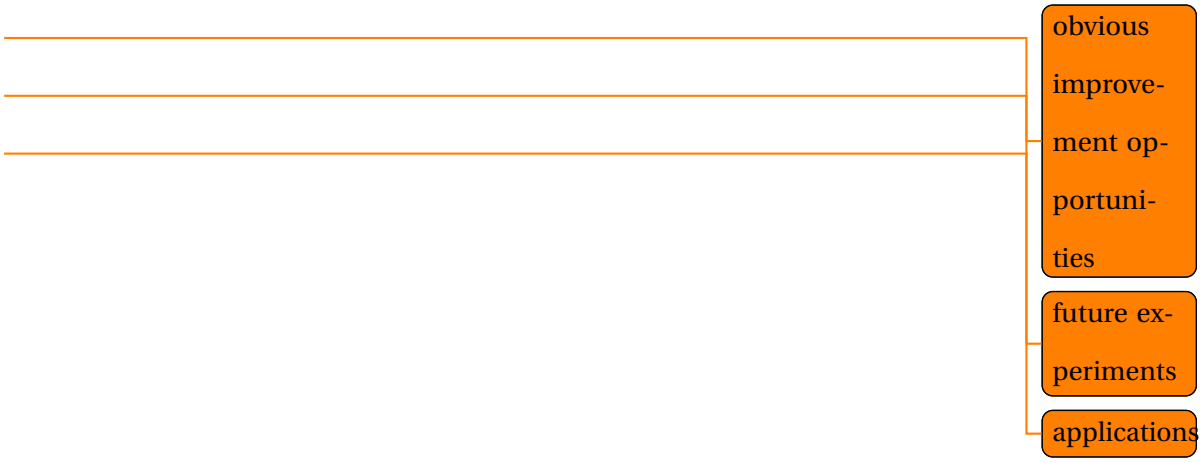
Conclusions

draw conclusions

Chapter 7

Future Work

Further development of this work would likely center on integration of more sensors and refinement of the noise models governing the process noise and the measurement noise in each sensor.



Bibliography

[1] S. J. Julier and J. K. Uhlmann, "A New Extension of the Kalman Filter to Nonlinear Systems," in *Proc. SPIE*, vol. 3068, pp. 182–193, 1997.

[2] S. J. Julier, "A Skewed Approach to Filtering," in *Proc. SPIE*, vol. 3373, pp. 271–282, 1998.

[3] S. J. Julier, "The Scaled Unscented Transformation," in *Proceedings of the 2002 American Control Conference (IEEE Cat. No.CH37301)*, vol. 6, pp. 4555–4559, May 2002.

[4] S. J. Julier and J. K. Uhlmann, "Unscented Filtering and Nonlinear Estimation," in *Invited Paper Proceedings of the IEEE*, vol. 92, pp. 401–422, March 2004.

[5] D. Donavanik, A. Hardt-Stremayr, G. Gremillion, S. Weiss, and W. Nothwang, "Multi-Sensor Fusion Techniques for State Estimation of Micro Air Vehicles," 2016.

[6] G. Klein and D. Murray, "Parallel Tracking and Mapping for Small AR Workspaces," in *Proc. Sixth IEEE and ACM International Symposium on Mixed and Augmented Reality (ISMAR'07)*, (Nara, Japan), November 2007.

[7] S. Weiss, D. Scaramuzza, and R. Siegwart, "Monocular-SLAM-Based Navigation for Autonomous Micro Helicopters in GPS-denied Environments," *Journal of Field Robotics*, vol. 28, pp. 854–874, Nov. 2011.

[8] S. Shen, N. Michael, and V. Kumar, "Autonomous Multi-Floor Navigation with a Computationally Constrained MAV," in *IEEE International Conference on Robotics and Automation*, pp. 20–25, 2011.

[9] S. Weiss and R. Siegwart, "Real-Time Metric State Estimation for Modular Vision-Inertial Systems," in *2011 IEEE International Conference on Robotics and Automation*, pp. 4531–4537, May 2011.

[10] S. Weiss, M. W. Achtelik, M. Chli, and R. Siegwart, "Versatile Distributed Pose Estimation and Sensor Self-Calibration for an Autonomous MAV," in *2012 IEEE International Conference on Robotics and Automation*, pp. 31–38, May 2012.

[11] S. Weiss, M. W. Achtelik, S. Lynen, M. Chli, and R. Siegwart, "Real-Time Onboard Visual-Inertial State Estimation and Self-Calibration of MAVs in Unknown Environments," in *2012 IEEE International Conference on Robotics and Automation*, pp. 957–964, May 2012.

[12] G. P. Huang, A. I. Mourikis, and S. I. Roumeliotis, "A Quadratic-Complexity Observability-Constrained Unscented Kalman Filter for SLAM," *IEEE Transactions on Robotics*, vol. 29, pp. 1226–1243, Oct 2013.

[13] S. Lynen, M. W. Achtelik, S. Weiss, M. Chli, and R. Siegwart, "A Robust and Modular Multi-Sensor Fusion Approach Applied to MAV Navigation," in *2013 IEEE/RSJ International Conference on Intelligent Robots and Systems*, pp. 3923–3929, Nov 2013.

[14] J. G. Rogers, J. R. Fink, and E. A. Stump, "Mapping with a Ground Robot in GPS Denied and Degraded Environments," in *2014 American Control Conference*, pp. 1880–1885, June 2014.

[15] M. Faessler, F. Fontana, C. Forster, E. Mueggler, M. Pizzoli, and D. Scaramuzza, "Autonomous, Vision-Based Flight and Live Dense 3D Mapping with a Quadrotor Micro Aerial Vehicle," *Journal of Field Robotics*, vol. 33, no. 4, pp. 431–450, 2016.