

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

Joshua Galen Eddy

Thesis submitted to the Faculty of the
Virginia Polytechnic Institute and State University
in partial fulfillment of the requirements for the degree of

Master of Science
in
Aerospace Engineering

Kevin Kochersberger, Chair
Mazen Farhood
Craig Woolsey

May 5, 2017
Blacksburg, Virginia

Abstract

This thesis presents the formulation and implementation of an Unscented Kalman Filter (UKF) framework for fusion of visual and inertial sensor data 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 that 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

1	Introduction	1
1.1	(Personal Motivation)	1
1.2	Organization of this Document	3
2	Prior Work	5
2.1	Development of the Unscented Kalman Filter	5
2.2	Visual-Inertial Navigation	7
3	Algorithm Design and Implementation	13
3.1	UKF Formulation	13
3.1.1	Prediction Step	13
3.1.2	Correction Step	16
3.1.3	Process Model	17
3.1.4	Observation Model	19
3.1.5	Process Noise Covariance Matrix	20
3.1.6	Measurement Noise Covariance Matrix	21
3.2	Software Design Considerations	22
4	Experimental Design	24
4.1	Testing Considerations	24
4.2	Materials	26
4.2.1	Computation and Sensing	26

4.2.2	Mobile Test Stand	27
4.3	The Experiments	28
5	Experimental Results	29
6	Conclusions	33
7	Future Work	34
7.1	System Improvements	34
7.1.1	Integration of New Sensors	34
7.1.2	Improvements for Current Sensors	37
7.1.3	Refinements to the Underlying Algorithm	37
7.2	Future Experiments	38
7.3	System Applications	38
	Bibliography	40

List of Figures

1.1 Example April tag	2
4.1 3D-printed sensor mount	25
5.1 Box Pattern Trial 1 Trajectory	30
5.2 Box Pattern Trial 2 Trajectory	30
5.3 Box Pattern Trial 3 Trajectory	31
5.4 Box Pattern Trial 4 Trajectory	31
5.5 Box Pattern Trial 5 Trajectory	32

Acronyms

AR Augmented Reality

EKF Extended Kalman Filter

GPS Global Positioning System

IMU Inertial Measurement Unit

KF Kalman Filter

MSF-EKF Multi-Sensor Fusion Extended Kalman Filter

PTAM Parallel Tracking and Mapping

ROS Robot Operating System

SLAM Simultaneous Localization and Mapping

UAS Unmanned Aircraft System

UAV Unmanned Aerial Vehicle

UKF Unscented Kalman Filter

UT Unscented Transform

UTM Unmanned Traffic Management

VIN Visual-Inertial Navigation

VO Visual Odometry

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.

Historically, man-portable drones capable of indoor reconnaissance have been computationally constrained due to restrictions on both size and weight. Recent advances in miniaturized desktop computers, as well as low-power Graphics Processing Units (GPUs), have enabled a new class of computationally intensive algorithms to run aboard small aircraft without degrading flight time or other performance metrics. Specifically, small drones now present a viable platform for high-fidelity Simultaneous Localization and Mapping (SLAM), visual object recognition, and state estimation algorithms employing numerous heterogeneous sensors.

finish in-
tro

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 Aerospace¹ (NIA) in Hampton, Virginia. In the summer of 2014, I was part of a team of NIA researchers working on the Flying Donkey Challenge², 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 System³ (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

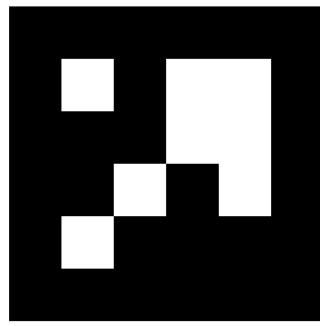


Figure 1.1: An example of an April tag.

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 m² April tags such as that found in Figure 1.1⁴. Following the completion of this project, I worked as a summer intern at the Autonomy Incubator.

¹<http://www.nianet.org>

²<http://www.flyingdonkey.org>

³<http://wiki.ros.org>

⁴http://wiki.ros.org/apriltags_ros

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 “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 the Autonomous Systems Lab⁵ (ASL) at ETH Zurich⁶ and the University of Pennsylvania’s GRASP Lab⁷. We define the current state of the art in filter-based navigation and establish the research context in which my thesis exists.

⁵www.asl.ethz.ch

⁶*Eidgenössische Technische Hochschule Zürich*, the Swiss Federal Institute of Technology in Zurich.

⁷www.grasp.upenn.edu

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` ROS 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 the 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 visual localization and state estimation. 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 various approaches to fuse either raw camera data or the outputs of vision-based algorithms with inertial sensor readings. In [5], Donavanik et al. provide a concise survey of the state of the art in VIN at the time

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

At this juncture, 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).

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. They also provided in-depth analysis of their system's applications, versatility, and reliability for real-time visual odometry and SLAM.

write up
PTAM init
procedure
in Exp De-
sign

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. The researchers address these problems using an EKF-SLAM formulation which fuses pose measurements with inertial sensor data. In doing so, they 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. The researchers also 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 the VIN system in [11], adding a speed-estimation module 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` ROS 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.

add Direct
IMU in-
tegration
section

Chapter 3

Algorithm Design and Implementation

3.1 UKF Formulation

3.1.1 Prediction Step

We begin by defining the following quantities:

$$\mathbf{p} = \{x, y, z\}^T \quad (3.1)$$

$$\mathbf{q} = \{q_x, q_y, q_z, q_w\}^T \quad (3.2)$$

$$\mathbf{v} = \{\dot{x}, \dot{y}, \dot{z}\}^T \quad (3.3)$$

$$\boldsymbol{\Omega} = \{\omega_x, \omega_y, \omega_z\}^T \quad (3.4)$$

$$\mathbf{a} = \{\ddot{x}, \ddot{y}, \ddot{z}\}^T \quad (3.5)$$

The vectors \mathbf{p} , \mathbf{v} , and \mathbf{a} represent the vehicle's position, velocity, and acceleration, respectively. The quaternion¹ \mathbf{q} represents the vehicle's orientation, and the vector $\boldsymbol{\Omega}$ represents

¹From 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

$$\mathbf{x} = \left\{ \mathbf{p}^T, \mathbf{q}^T, \mathbf{v}^T, \boldsymbol{\Omega}^T, \mathbf{a}^T \right\}^T \quad (3.6)$$

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

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

$$\begin{aligned} \chi_{k-1|k-1}^0 &= \mathbf{x}_{k-1|k-1} \\ \chi_{k-1|k-1}^i &= \mathbf{x}_{k-1|k-1} + \left(\sqrt{(n + \lambda) \mathbf{P}_{k-1|k-1}} \right)_i, \quad i = 1, \dots, n \\ \chi_{k-1|k-1}^i &= \mathbf{x}_{k-1|k-1} - \left(\sqrt{(n + \lambda) \mathbf{P}_{k-1|k-1}} \right)_{i-n}, \quad i = n + 1, \dots, 2n, \end{aligned} \quad (3.7)$$

where χ^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 square root matrix $\sqrt{(n + \lambda) \mathbf{P}_{k-1|k-1}}$. The constants α , β , κ , and λ are defined as

$$\alpha = 0.75 \quad (3.8)$$

$$\kappa = 0 \quad (3.9)$$

$$\beta = 2 \quad (3.10)$$

$$\lambda = \alpha^2(n + \kappa) - n = -7 \quad (3.11)$$

These constants control the spread of the sigma points chosen within the filter. According to Wan et al. in [16], setting $\beta = 2$ is optimal for a Gaussian state distribution and κ is commonly set to zero. The value of α is usually small and positive. Here, we chose α by trial and error after observing the rate at which the filter converged in a number of preliminary experiments. We found that choosing $0 < \alpha \leq 0.01$ made the filter slow to converge. The filter's estimates would lag behind the pose sensor's readings by an interval that was sometimes several seconds in length and in steady state would frequently undershoot the pose sensor. Conversely, choosing $\alpha \geq 1$ caused the filter to diverge. At $\alpha \approx 0.75$, the filter had no distinguishable lag and was highly stable.

The square root matrix \mathbf{A} of a matrix \mathbf{B} is defined here as

$$\mathbf{AA}^T = \mathbf{B} \quad (3.12)$$

and is computed via Cholesky decomposition².

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

$$\boldsymbol{\chi}_{k|k-1}^i = f(\boldsymbol{\chi}_{k-1|k-1}^i), \quad i = 0, \dots, 2n. \quad (3.13)$$

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

$$\hat{\mathbf{x}}_{k|k-1} = \sum_{i=0}^{2n} W_s^i \boldsymbol{\chi}_{k|k-1}^i \quad (3.14)$$

$$\mathbf{P}_{k|k-1} = \left(\sum_{i=0}^{2n} W_c^i (\boldsymbol{\chi}_{k|k-1}^i - \hat{\mathbf{x}}_{k|k-1}) (\boldsymbol{\chi}_{k|k-1}^i - \hat{\mathbf{x}}_{k|k-1})^T \right) + \mathbf{Q}_k, \quad (3.15)$$

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

$$\begin{aligned} W_s^0 &= \frac{\lambda}{n+\lambda} \\ W_c^0 &= \frac{\lambda}{n+\lambda} + (1-\alpha^2 + \beta) \\ W_s^i &= W_c^i = \frac{1}{2(n+\lambda)}, \quad i = 1, \dots, 2n \end{aligned} \quad (3.16)$$

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

²https://en.wikipedia.org/wiki/Cholesky_decomposition#The_Cholesky_algorithm

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

$$\begin{aligned}\boldsymbol{\chi}_{k|k-1}^0 &= \mathbf{x}_{k|k-1} \\ \boldsymbol{\chi}_{k|k-1}^i &= \mathbf{x}_{k|k-1} + \left(\sqrt{(n+\lambda) \mathbf{P}_{k|k-1}} \right)_i, \quad i = 1, \dots, n \\ \boldsymbol{\chi}_{k|k-1}^i &= \mathbf{x}_{k|k-1} - \left(\sqrt{(n+\lambda) \mathbf{P}_{k|k-1}} \right)_{i-n}, \quad i = n+1, \dots, 2n.\end{aligned}\tag{3.17}$$

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

$$\boldsymbol{\gamma}_k^i = h(\boldsymbol{\chi}_{k|k-1}^i), \quad i = 0, \dots, 2n.\tag{3.18}$$

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

$$\mathbf{z}_k = \{x, y, z, q_x, q_y, q_z, q_w\}_{meas.}^T.\tag{3.19}$$

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

$$\hat{\mathbf{z}}_k = \sum_{i=0}^{2n} W_s^i \boldsymbol{\gamma}_k^i\tag{3.20}$$

$$\mathbf{P}_{zz} = \left(\sum_{i=0}^{2n} W_c^i (\boldsymbol{\gamma}_k^i - \hat{\mathbf{z}}_k) (\boldsymbol{\gamma}_k^i - \hat{\mathbf{z}}_k)^T \right) + \mathbf{R},\tag{3.21}$$

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

$$\mathbf{P}_{xz} = \sum_{i=0}^{2n} W_c^i (\boldsymbol{\chi}_{k|k-1}^i - \hat{\mathbf{x}}_{k|k-1}) (\boldsymbol{\gamma}_k^i - \hat{\mathbf{z}}_k)^T.\tag{3.22}$$

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

$$\mathbf{K}_k = \mathbf{P}_{xz} \mathbf{P}_{zz}^{-1}. \quad (3.23)$$

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

$$\hat{\mathbf{x}}_{k|k} = \hat{\mathbf{x}}_{k|k-1} + \mathbf{K}_k (\hat{\mathbf{z}}_k - \mathbf{z}_k). \quad (3.24)$$

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:

$$\mathbf{P}_{k|k} = \mathbf{P}_{k|k-1} - \mathbf{K}_k \mathbf{P}_{zz} \mathbf{K}_k^T. \quad (3.25)$$

3.1.3 Process Model

To propagate the vehicle's state forward in time, we perform a number of integration 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

$$\mathbf{q}_k = \mathbf{q}_{k-1} + \frac{1}{2} \Theta(\boldsymbol{\Omega}_k) \mathbf{q}_{k-1} \Delta t, \quad (3.26)$$

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

$$\Theta(\boldsymbol{\Omega}_k) = \begin{bmatrix} 0 & \omega_z & -\omega_y & \omega_x \\ -\omega_z & 0 & \omega_x & \omega_y \\ \omega_y & -\omega_x & 0 & \omega_z \\ -\omega_x & -\omega_y & -\omega_z & 0 \end{bmatrix} \quad (3.27)$$

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

$$\boldsymbol{\Omega}_k = \boldsymbol{\Omega}_{meas..} \quad (3.28)$$

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:

$$\mathbf{a}_k = R_b^g [\mathbf{q}_{k-1}] (\mathbf{a}_{meas.} - \mathbf{g}). \quad (3.29)$$

Here $R_b^g [\mathbf{q}_{k-1}]$ 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.

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:

$$\mathbf{v}_k = \mathbf{v}_{k-1} + \frac{1}{2} (\mathbf{a}_k + \mathbf{a}_{k-1}) \Delta t. \quad (3.30)$$

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

$$\mathbf{p}_k = \mathbf{p}_{k-1} + \frac{1}{2} (\mathbf{v}_k + \mathbf{v}_{k-1}) \Delta t. \quad (3.31)$$

Together, equations 3.26, 3.28, 3.29, 3.30, and 3.31 constitute the nonlinear process function f :

$$f(\mathbf{x}_k) = \begin{Bmatrix} \mathbf{p}_{k-1} + \frac{1}{2} (\mathbf{v}_k + \mathbf{v}_{k-1}) \Delta t \\ \mathbf{q}_{k-1} + \frac{1}{2} \Theta(\mathbf{\Omega}_k) \mathbf{q}_{k-1} \Delta t \\ \mathbf{v}_{k-1} + \frac{1}{2} (\mathbf{a}_k + \mathbf{a}_{k-1}) \Delta t \\ \mathbf{\Omega}_{meas.} \\ R_b^g [\mathbf{q}_{k-1}] (\mathbf{a}_{meas.} - \mathbf{g}) \end{Bmatrix}. \quad (3.32)$$

Have Sam look over this. Are subscripts right?

What is $R(\mathbf{q})$?

3.1.4 Observation Model

Quaternion Continuity Correction

Due to a defect in the ROS PTAM implementation, quaternion estimates switch sign without warning after rotations of approximately 270 degrees. To prevent loss of continuity in quaternion estimates, we keep the previous quaternion estimate \mathbf{q}_{k-1} in memory and employ the following algorithm to correct the incoming quaternion estimate if a sign-flip has occurred:

fill this
out more?

Algorithm 1 Check for continuity between quaternion estimates

```

1: function CHECKQUATERNIONCONTINUITY( $\mathbf{q}_{k-1}$ ,  $\mathbf{q}_k$ )
2:   sum =  $\|\mathbf{q}_{k-1} + \mathbf{q}_k\|$ 
3:   diff =  $\|\mathbf{q}_{k-1} - \mathbf{q}_k\|$ 
4:   if diff  $\leq$  sum then
5:      $\mathbf{q}_k = -\mathbf{q}_k$ 
6:   return  $\mathbf{q}_k$ 
```

Pseudovelocity Correction

Of the various states being measured, only the acceleration vector \mathbf{a} and angular rate vector $\boldsymbol{\Omega}$ are measured directly by the IMU:

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

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

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

The vehicle's velocity \mathbf{v} , however, is not measured by either the IMU or PTAM. There is no sensor in place by which \mathbf{v} can be directly measured. Therefore, velocity must be determined through a derived measurement.

Because velocity is predicted by integrating the vehicle's acceleration (Section 3.1.3),

noise in the accelerometer can cause unbounded drift in the velocity. The resulting erroneous velocity then propagates through each successive 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 Kalman Filter corrects the position each time that a PTAM message arrives, only to drift again a fraction of a second later when the next IMU message arrives. This kind of intermittent drift produces sharp “bounces” in the vehicle’s estimated trajectory and makes the filter numerically unstable.

To prevent this potentially 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 :

$$\mathbf{v}_k = \frac{\mathbf{p}_{k, \text{meas.}} - \mathbf{p}_{k-1, \text{meas.}}}{\Delta t_P}. \quad (3.33)$$

This pseudovelocity correction prevents unbounded velocity drift, which in turn makes the filter’s output numerically 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 , respectively³. 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

³This claim of equal standard deviations between axes is supported by the IMU data sheet.

the x -axis accelerometer and the x -axis gyroscope, the corresponding terms of \mathbf{Q}_k would be

$$Q_{a_x, a_x} = \sigma_a \Delta t \quad (3.34)$$

and

$$Q_{\omega_x, \omega_x} = \sigma_\omega \Delta t, \quad (3.35)$$

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.

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.

integral
equations

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 system⁴.

full Q ma-
trix

diagonality?
Q-R ratio?

⁴<https://www.vicon.com/>

The resulting **R** matrix is as follows:

R Matrix

3.2 Software Design Considerations

Much of the impetus for creating the `kalman_sense` ROS 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 mathematics 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 then 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 library⁵.

The subclass (`QuadUkf` for the remainder of this thesis) handles all of the ROS commu-

⁵www.eigen.tuxfamily.org

nlications for the given system. Specifically, this class has callback functions for receiving sensor data and is responsible for publishing state and covariance estimates.

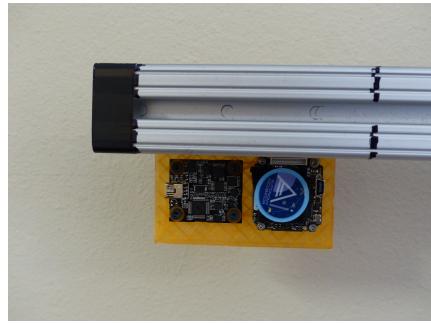
Chapter 4

Experimental Design

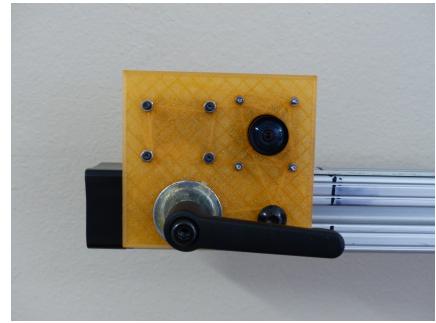
4.1 Testing Considerations

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 \mathbf{p} and orientation quaternion \mathbf{q} of a rotorcraft UAV in real time. Thus, the experiments testing the system'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 single 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 containing an Intel i7 processor with 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



(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.

modified Quick Response¹ (QR) codes in order to provide sufficient visual features for PTAM to track.

¹https://en.wikipedia.org/wiki/QR_code

4.2 Materials

4.2.1 Computation and Sensing

1. One (1) MatrixVision mvBlueFOX-MLC Camera²
2. One (1) 1044_0 PhidgetSpatial Precision 3/3/3 High Resolution IMU³
3. One (1) Hewlett-Packard Spectre x360 Convertible Laptop 13-ac076nr⁴
4. Two (2) male Mini USB 2.0 to male USB Type A cables

²<https://www.matrix-vision.com/USB2.0-single-board-camera-mvbluefox-mlc.html>

³http://www.phidgets.com/products.php?product_id=1044

⁴<http://store.hp.com/us/en/pdp/hp-spectre-x360---13-ac076nr>

4.2.2 Mobile Test Stand

1. One (1) Oklahoma Sound PRC200 Premium Presentation Cart⁵
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® Inc. 1515 Rail⁶
5. One (1) 15 Series "L" Handle Linear Bearing Brake Kit⁷
6. One (1) $\frac{5}{16}$ -18 × 0.687" Black FBHSCS (Screw)⁸
7. Two (2) Slide-In Economy T-Nuts⁹
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 Tube

⁵<http://www.oklahomasound.com/products/product-category/single/?prod=9>

⁶<https://8020.net/1515.html>

⁷<https://8020.net/6800.html>

⁸<https://8020.net/shop/3320.html>

⁹Also available at <https://8020.net/shop/3320.html>

4.3 The Experiments

Two 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 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 data recording utility.

more explanation
of post
processing

¹⁰<http://wiki.ros.org/rosbag>

Chapter 5

Experimental Results

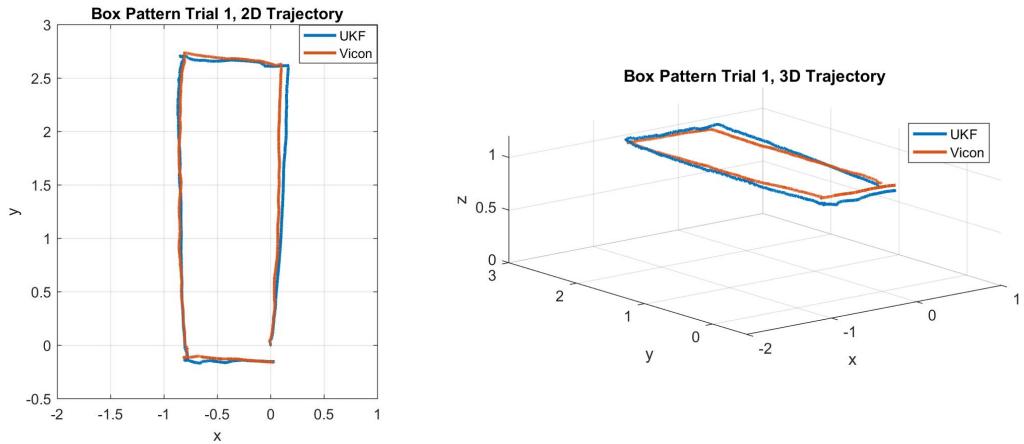


Figure 5.1: 2D and 3D trajectory plots from Box Pattern Trial 1.

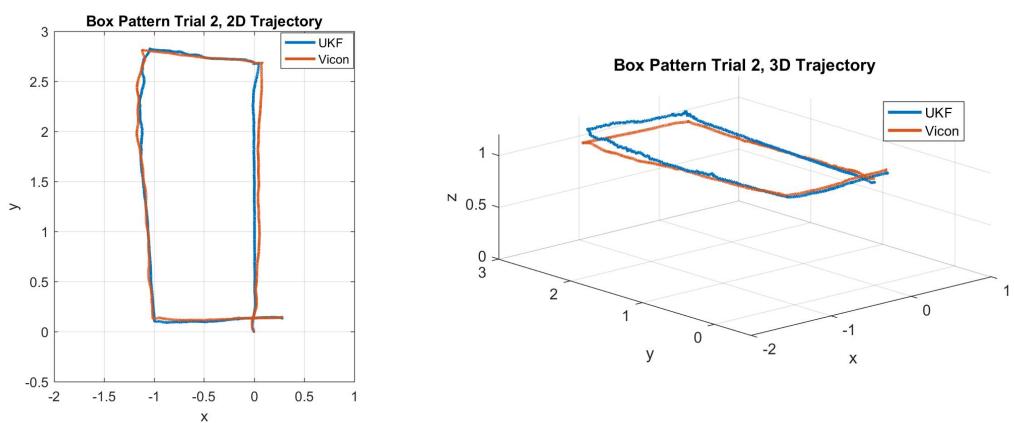


Figure 5.2: 2D and 3D trajectory plots from Box Pattern Trial 2.

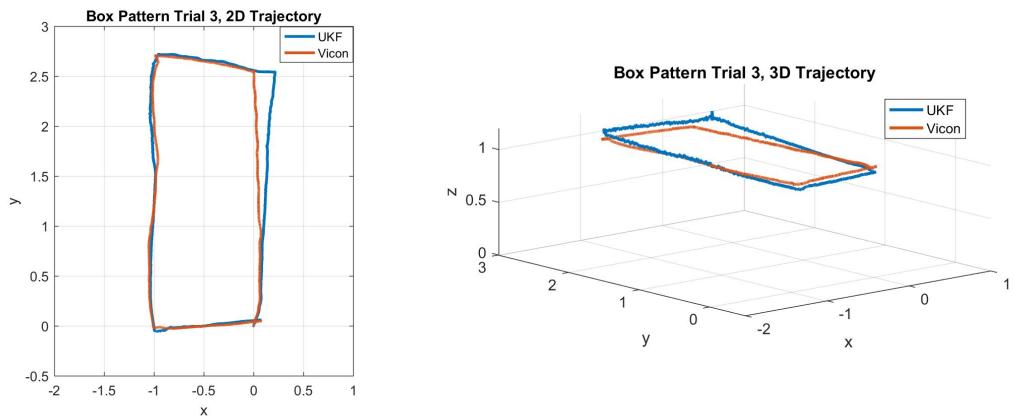


Figure 5.3: 2D and 3D trajectory plots from Box Pattern Trial 3.

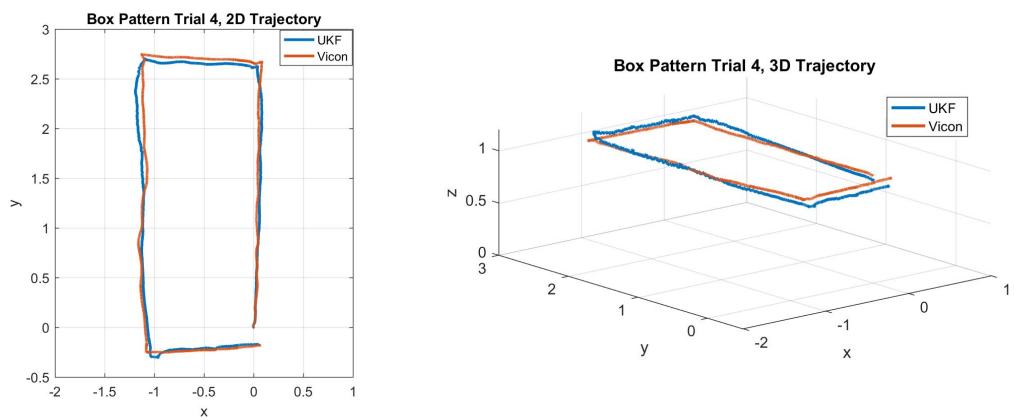


Figure 5.4: 2D and 3D trajectory plots from Box Pattern Trial 4.

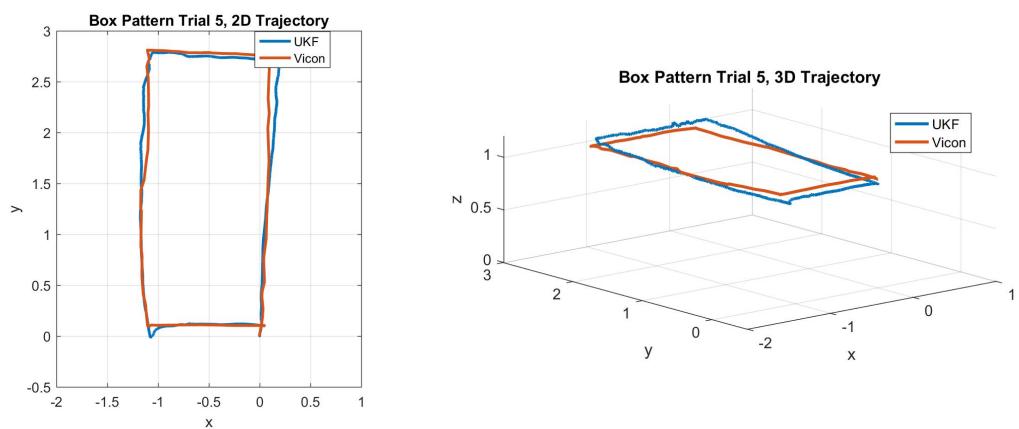


Figure 5.5: 2D and 3D trajectory plots from Box Pattern Trial 5.

Chapter 6

Conclusions

draw con-
clusions

Chapter 7

Future Work

7.1 System Improvements

Experimentation with `kalman_sense` has revealed a number of areas for improvement. Speaking generally, the system needs enhancements to become more accurate and more robust to sensor dropout. Further development of this work would likely center on three main areas:

1. Integration of new sensors,
2. Improvements for current sensors, and
3. Refinements to the underlying algorithm.

Each of these could produce a marked improvement to the system in accuracy, robustness, or both. Particular use cases, Concepts of Operations (CONOPS), and desired performance metrics would likely drive the selection of which of these areas to tackle.

7.1.1 Integration of New Sensors

The accuracy and robustness of the system could be augmented by the integration of additional sensors, including but not limited to

1. One or more GPS receivers (particularly Real Time Kinematic¹ GPS),
2. A pressure altimeter,
3. A 3-axis magnetometer,
4. One or more laser rangefinders,
5. One or more forward-facing cameras (preferably depth cameras, such as RGB-D),
6. One-dimensional, multi-dimensional, or scanning LIDAR².

We have previously mentioned work from the GRASP Lab ([8]) that has demonstrated the efficacy of similar UKF frameworks in indoor-outdoor transitions and in navigating confined spaces using an expanded suite of sensors. In the case where more than one sensor can be used to observe a given vehicle state variable, those multiple sensors can be used to check one another and even perform real-time, in-air calibration. The system developed in this thesis has the advantage of being hardware-minimal in that it depends on only two sensors (the IMU and ventral camera), but this convenience comes at the cost of robustness as either sensor presents a single point of failure. Moreover, the sensors have no overlap in that neither can observe any of the states observed by the other. This eliminates the possibility of checking one sensor against the other with high confidence. As an example, introducing a ventral laser rangefinder or pressure altimeter would allow for much more precise measurement of the vehicle's altitude and would allow for in-flight re-initialization of any metric SLAM/VO algorithm.

With the exception of certain sensors such as scanning LIDAR, which can cost tens of thousands of dollars, and RTK-GPS, which often falls in the same price range, these proposed new sensors would be inexpensive additions to the vehicle. Given the sophistication of current miniaturized sensors, it is not unreasonable to expect sub-meter or centimeter-level position accuracy from vehicles weighing fewer than fifteen pounds (including a substantial sensor payload) and costing less than 20,000 USD. It has been

¹https://en.wikipedia.org/wiki/Real_Time_Kinematic

²"Light Detection and Ranging," <https://en.wikipedia.org/wiki/Lidar>

the case for a number of years now that meter-level accuracy could be taken for granted with “toy” quadcopters using only MEMS³ IMUs and sub-\$100 GPS receivers. The trend of miniaturization has made more and more sensors of better and better quality increasingly available in the weight- and cost-constrained world of aerial robotics. A variety of inexpensive off-the-shelf autopilots, such as the Pixhawk 2.1⁴, already boast triple-redundant IMUs and support for multiple GPS modules. Many of the hardware integration challenges that once plagued hobbyists and researchers have been eliminated by popular demand for user-friendly drones.

The organization of `kalman_sense`’s code base is such that the integration of new sensors would require only the addition of new callback functions and ROS subscribers, as opposed to a full redesign or any major edits to the existing code. By a similar token, a new SLAM or VO algorithm could also be used in place of PTAM, allowing an “apples-to-apples” comparison of different vision algorithms. For example, Donavanik et al. ([5]) have identified a new algorithm known as ORB-SLAM ([17]) as a promising candidate for robust SLAM, already implemented in ROS. Because of ROS’s sophisticated ecosystem of interoperable packages and message formats, any pose sensor employing the same message type as PTAM could plug in to `kalman_sense` in a matter of seconds.

Much of the design thinking that went into the `kalman_sense` package was focused on this particular possibility. The aircraft of the future will be vehicles laden with a vast suite of heterogeneous sensors, the outputs of which would, at any given time, be used not only for inner-loop and outer-loop flight control but also for self-calibration, real-time diagnostics, and operator telemetry. Future UAVs will be akin to floating laboratories, carrying numerous means of measurement for all manner of self-sensing as well as scientific applications.

³https://en.wikipedia.org/wiki/Microelectromechanical_systems

⁴<http://www.proficnc.com/>

7.1.2 Improvements for Current Sensors

For improved pose measurements, a more robust implementation of PTAM could be written and the same experiments could be repeated for comparison. PTAM's rotation errors and inability to eliminate lens distortion pose severe limitations to its effectiveness in small UAS navigation. Moreover, if a rewritten implementation were able to interface to a downward-facing rangefinder, the initialization process could be eliminated altogether. The problem with this is that making these changes would likely require a major rewrite, a project that could only feasibly be undertaken by an experienced team of computer vision researchers and software developers. The mere cost in terms of both time and labor would likely make this an unappealing proposition, especially if other ready-made SLAM/VO algorithms were available to serve as PTAM replacements.

Processing power will also be central to the system's effectiveness during deployment on a real aircraft. This research is in many ways the product of recent advances in computing power. The advent of miniature desktop computers, such as the Intel NUC⁵, as well as power-efficient Graphics Processing Units⁶ (GPUs) has made a new frontier of estimation and localization algorithms accessible for the first time in the UAV world. A simple and generally inexpensive approach to augmenting the kalman_sense system would thus be the inclusion of top-of-the-line lightweight computing hardware. With modern onboard computers, even the need for cross-platform code compilation (so-called "transpiling") is eliminated because these computers employ the same x86 architecture used by most developers.

7.1.3 Refinements to the Underlying Algorithm

The UKF formulation itself presents another set of opportunities for improvement. In particular, better noise modeling would allow for increased accuracy. As described previously, the noise covariance matrices used during the experiments were only rough, static

⁵"Next Unit of Computing," https://en.wikipedia.org/wiki/Next_Unit_of_Computing

⁶https://en.wikipedia.org/wiki/Graphics_processing_unit

models meant to overestimate the noise in the system. In practice, it would be better to replace these with time-varying matrices whose terms more closely matched the intrinsic parameters of the system. Due to the high degree of coupling between state variables and the fundamental nonlinearity of the process model, this work was considered to be outside the scope of this thesis. In the future, however, a better **Q** matrix could be determined, perhaps by application of autocovariance and system identification methods. Moreover, any new sensor integrated into the system would require its own **R** matrix detailing its particular idiosyncrasies. With this done, attention could then be turned to tuning the **Q/R** ratio of each sensor, thus making the system even more faithful to known sensor intrinsics. With more sensors, self-calibration techniques such as those employed in [10] could also be implemented in order to avoid having to reinitialize one or more sensors in mid-air.

7.2 Future Experiments

Using a better SLAM/VO algorithm would be perhaps the easiest and cheapest way to improve the system. Testing out different vision algorithms would thus be an appealing avenue of inquiry. It would behoove the researcher to characterize the system's behavior under the same conditions which caused PTAM to fail. In particular, test flights that include pure rotations, longer translations, and motion over non-flat terrain would be of interest.

finish this

7.3 System Applications

Applications for this system are wide-ranging, spanning almost every conceivable use case for a small drone. The following is a non-comprehensive list of possible applications:

1. Mapping and 3D reconstruction of structures and topography,
2. Emergency response,

3. Infrastructure inspection,
4. Autonomous package delivery, and
5. Military/defense solutions.

What all of these applications have in common is the need for robust localization. In any of the above scenarios, the possibility of GPS signal loss presents a real threat not only to the aircraft but to the health and safety of bystanders and property.

Mapping and 3D Reconstruction

Growing demand for so-called “precision agriculture” has brought with it the need to map large areas of cropland at a high speed and low cost point.

Emergency Response

Infrastructure Inspection

1. Power lines,
2. Gas pipelines,
3. Railroads, and
4. Buildings

Military/Defense Solutions

1. Robust tracking of friendly and hostile elements,
2. Remote observation of roads and vehicle-related hazards,
3. Autonomous deliveries of materiel in theater, and
4. Human-machine teaming.

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.
- [16] E. A. Wan and R. V. D. Merwe, “The Unscented Kalman Filter for Nonlinear Estimation,” in *Proceedings of the IEEE 2000 Adaptive Systems for Signal Processing, Communications, and Control Symposium (Cat. No.00EX373)*, pp. 153–158, 2000.

- [17] R. Mur-Artal, J. M. M. Montiel, and J. D. Tardós, “ORB-SLAM: A Versatile and Accurate Monocular SLAM System,” *IEEE Transactions on Robotics*, vol. 31, pp. 1147–1163, Oct 2015.