IMPROVING THE ROBUSTNESS OF MONOCULAR VISION-AIDED NAVIGATION FOR MULTIROTORS THROUGH INTEGRATED ESTIMATION AND GUIDANCE

A Thesis

by

WILLIAM DANIEL WHITTEN

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

MASTER OF SCIENCE

Chair of Committee,	John Hurtado
Co-Chair of Committee,	John Junkins
Committee Member,	Sivakumar Rathinam
Head of Department,	Rodney Bowersox

May 2017

Major Subject: Aerospace Engineering

ABSTRACT

Multirotors could be used to autonomously perform tasks in search-and-rescue, reconnaissance, or infrastructure-monitoring applications. In these environments, the vehicle may have limited or degraded GPS access. Researchers have investigated methods for simultaneous localization and mapping (SLAM) using on-board vision sensors, allowing vehicles to navigate in GPS-denied environments. In particular, SLAM solutions based on a monocular camera offer low-cost, low-weight, and accurate navigation indoors and outdoors without explicit range limitations. However, a monocular camera is a bearing-only sensor. Additional sensors are required to achieve metric pose estimation, and the structure of a scene can only be recovered through camera motion. Because of these challenges, the performance of monocular-based navigation solutions is typically very sensitive to the environment and the vehicle's trajectory. This work proposes an integrated estimation and guidance approach for improving the robustness of monocular SLAM to environmental uncertainty. It is specifically intended for a multirotor carrying a monocular camera, downward-facing rangefinder, and inertial measurement unit (IMU). A guidance maneuver is proposed that takes advantage of the metric rangefinder measurements. When the environmental uncertainty is high, the vehicle simply moves up and down, initializing features with a confident and accurate baseline. In order to demonstrate this technique, a vision-aided navigation solution is implemented which includes a unique approach to feature covariance initialization that is based on consider least squares. Features are only initialized if there is enough information to accurately triangulate their position, providing an indirect metric of environmental uncertainty that could be used to signal the guidance maneuver. The navigation filter is validated using hardware and simulated data. Finally, simulations show that the proposed initialization maneuver is a simple, practical, and ef-

fective way to improve the robustness of monocular-vision-aided-navigation and could increase the amount of autonomy that GPS-denied multirotors are capable of achieving.

ACKNOWLEDGMENTS

I first heard about the field of vision-aided navigation in the fall of 2014 and immediately knew that I wanted to gain expertise in this area. However, it has taken several years to develop the skills and knowledge necessary to achieve this goal. Many individuals and organizations have helped me to obtain these requirements. I have greatly enjoyed and benefited from the company and knowledge of the faculty and students associated with the Land, Air, and Space Robotics (LASR) lab. I am particular grateful for Dr. Hurtado's AERO 622 class where I developed a strong understanding of and appreciation for attitude parameterizations. I also benefited greatly from the study of Dr. Junkin's book "Optimal Estimation of Dynamic Systems." In addition I am grateful for my summer experience at the Autonomous Vehicles Lab at the University of Florida Research and Education Facility (REEF). I am particularly thankful for conversations with Dr. Brink. From Dr. Brink I gleaned a strong appreciation for the covariance matrix and the idea of using vertical motion to establish an accurate baseline for feature initialization. More generally I am thankful to my family for supporting me in endless ways throughout my education and to God for giving me the resources and talents to pursue a subject that I am passionate about.

CONTRIBUTORS AND FUNDING SOURCES

Contributors

This work was supported by a thesis committee consisting of Professors John Hurtado and John Junkins of the department of Aerospace Engineering and Professor Sivakumar Rathinam of the department of Mechanical Engineering.

Dr. Kevin Brink, an Air Force Research Lab engineer, originally suggested the idea of using a downward-facing rangefinder and vertical motion to improve the quality of monocular-vision-aided navigation. Dr. Brink also provided advice for the implementation and demonstration of the work. All other work conducted for the thesis was completed by the student independently.

Funding Sources

This material is based upon work supported by the National Science Foundation Graduate Research Fellowship Program under Grant No. DGE-1252521. Any opinions, findings, and conclusions or recommendations expressed in this material are those of the author(s) and do not necessarily reflect the views of the National Science Foundation.

NOMENCLATURE

IMU	Inertial Measurement Unit
SLAM	Simultaneous Localization and Mapping
GPS	Global Positioning System
EKF	Extended Kalman Filter
DOF	Degree of Freedom

TABLE OF CONTENTS

Page

LIST OF FIGURES

1. INTRODUCTION AND LITERATURE REVIEW

1.1 GPS-Denied Navigation

Multirotors are helicopters with more than one rotor. Thanks to the recent proliferation of small exteroceptive sensors (e.g. cameras) and microelectromechanical systems (MEMS), small multirotors have become commonplace and increasingly sophisticated. These helicopters range in size from those that can fit in the palm of a hand to larger units that weigh several kilograms [1]. Researchers have taken advantage of these vehicles' agility and small size while demonstrating multi-vehicle swarm flight [2], high speed maneuvers [3], and even cooperative ball throwing and catching [4].

The capabilities of these vehicles make them ideal for inspection, surveillance, search and rescue, and surveying applications in numerous industries. Currently, such applications require highly trained human pilots. Fully autonomous multirotors could significantly decrease the cost associated with deployment in these applications. In addition, the helicopters could operate in environments that are particularly dangerous or difficult to reach for humans. In these environments, the vehicle may not have access to GPS (e.g. building interiors, urban canyons) or GPS may lack accuracy or consistency.

Ideally, a GPS-denied vehicle would have the ability to navigate using only on-board sensors and computation in an a priori unknown and unstructured environment. The navigation solution should ideally have negligible drift over long trajectories and the algorithm's computational requirements should remain constant. Achieving these design objectives remains challenging for small multicopters. The limited payload and electrical power greatly constrain the sensor selection and computational resources available. A ground robot can, if necessary, stop and process batch measurements to update the state estimate. An airborne multirotor, on the other hand, has fast and unstable dynamics that

require real-time state estimates.

Despite the difficulties, researchers have investigated the use of onboard sensors and computation to provide GPS-denied state estimation. Early work required a priori scene knowledge [5] and/or offboard computation [6]. Blösch et al. demonstrated a micro aerial vehicle capable of control and navigation using exclusively onboard exteroceptive and proprioceptive sensors [7]. Their system used an inertial measurement unit (IMU) and monocular camera for sensing. The Parallel Tracking and Mapping (PTAM) SLAM algorithm [8] provided an accurate up-to-scale pose estimate. This vision-based pose estimate was fused with IMU measurements in a separate extended Kalman filter (EKF) to provide a full 6 degree of freedom (DOF) pose estimate. Extended versions of the work were used for a European Union-funded vision-controlled swarm project [1] and autonomous waypoint navigation in a simulated disaster area [9]. This so-called loosely-coupled vision-aided navigation approach has remained popular. NASA researchers fused PTAM estimates with IMU measurements to demonstrate autonomous landing [10]. This system has also been successfully implemented on the low-cost commercial AR Drone [11]. More recently, an alternative monocular SLAM algorithm known as semi-direct visual odometry (SVO) has also been effectively used in a loosely-coupled navigation scheme [12].

Researchers have considered sensors other than monocular cameras. Bachrach achieved impressive GPS-denied autonomy indooors using 2D laser scan-matching [13]. This platform was capable of autonomous navigation and flight through windows and doors. Laser scans were also used to enable onboard navigation, mapping, control, and planning through multi-level buildings [14]. With the addition of an RGB-D camera, this system successfully mapped sections of an earthquake-damaged building in Japan [15].

The use of RGB-D sensors for multirotor navigation has also been investigated. Huang, et. al. stabilized and controlled a quadcopter in a cluttered indoor environment using a Microsoft Kinect sensor [16]. Researchers have also used the RGB-D sensor in a relative

navigation and control framework [17].

Stereo cameras are also a viable sensor for quadcopter navigation. Heng, et. al. demonstrated indoor obstacle avoidance and path planning based on a stereo camera [18]. The camera pose was taken from a motion camera system. However, [19] uses a forward-facing stereo and downward facing monocular camera for autonomous mapping and navigation without the need for an external motion capture reference. More recently, Schmid, et. al demonstrated extensive indoor and outdoor flights using a forward-facing stereo camera [20]. The stereo algorithm was implemented on a field programmable gate array (FPGA).

Each sensor type has strengths and weaknesses for multicopter navigation. It is not possible to reconstruct the scale of the estimated position and map coming from monocular SLAM without secondary sensors. Laser-scan matching algorithms typically require environmental assumptions such as flat walls or floors. These assumptions are generally applicable only to indoor applications. RGB-D sensors currently detect patterns in infrared structured light or the time of flight of infrared light. For this reason, they are of limited use outdoors or in bright sunlight. Stereo reconstruction is computationally expensive. Real-time operation on a computationally constrained vehicle may require a secondary processor or FPGA devoted to stereo processing. All sensors and associated algorithms, except for monocular SLAM, have explicit range limitations. All vision-based approaches require adequate lighting. Finally, many odometry/SLAM algorithms require a "cluttered" environment with a sufficient set of distinct visual features or depth gradients.

Due to the many restrictions and limitations of exteroceptive perception, it is unlikely that navigation using a single sensor will be robust to the environmental changes encountered during realistic missions [21] [22]. However, monocular cameras have a number of features that make them highly suitable as general purpose primary or complementary navigation sensors. Monocular cameras are inexpensive, have a high information to weight ratio, do not have explicit range limitations, and work well in both indoor and out-

door settings. Most unmanned vehicles already have cameras onboard. Even if a laser rangefinder, stereo camera, or RGB-D sensor is used as the primary exteroceptive sensor, monocular visual odometry can serve as a complementary navigation solution or act as the main input when needed. Due to the practical advantages of cameras, developing monocular SLAM solutions that are accurate and efficient is of critical importance for achieving robust autonomous GPS-denied flight.

1.2 Monocular Simultaneous Localization and Mapping

Monocular SLAM has been studied extensively. The first real-time implementations were based on filtering methods. Davison introduced one of the first successful real-time algorithms [23] [24]. This implementation relies on an EKF. The EKF state vector consists of the vehicle state and the map. The vehicle state vector includes the current pose and velocity while the map state consists of the Cartesian coordinates of distinct features that are tracked between camera frames. The global positions of the features are initially unknown. Using a process model and camera measurement model, the vehicle state and map are simultaneously estimated. The covariance matrix accounts for uncertainty as well as the correlations between the vehicle state and the feature locations and also the correlations between different feature position estimates. In [25], the Cartesian feature coordinates were replaced with the so-called inverse depth parameterization. The inverse depth parameterization has a more linear measurement model for small feature baselines, leading to better convergence from initial condition errors. However, inverse depth has six parameters, leading to a higher computational burden.

In the Kalman filter, computational complexity scales quadratically with the number of feature states. This limits the number of features that can be included in the map. The multi-state constraint Kalman (MSKF) attempts to address this problem by estimating a sliding window of past camera poses rather than actively estimating the global positions

of features [26]. The complexity of the MSKF is linear in the number of features. Using a novel observability rule [27], the MSKF has shown excellent accuracy and consistency. However, this filter is not reliable if the camera has little motion with respect to the observed environment. The hybrid MSCKF/SLAM visual-inertial odometry described in [28] attempts to overcome this limitation by combining the MSKF with an EKF-SLAM implementation that uses an anchor point feature parameterization.

Monocular SLAM can also be formulated as a bundle adjustment (i.e. batch) problem. Once thought to only be useful for offline processing, bundle adjustment (BA) uses global optimization to reconstruct the motion of a camera and the environmental structure from a sequence of images [29]. PTAM was the first real-time BA monocular SLAM implementation [8]. It achieved real-time performance by heuristically selecting certain images as keyframes. In one thread, the pose of each new image with respect to the most recent keyframe is determined. This tracking thread operates at a high frame rate. A separate thread executes a more computationally expensive process that optimizes the global pose of each keyframe with respect to the sparse feature map. PTAM also includes loop closure. Loop closure is recognizing when a current image includes features that have been mapped in the past and updating the current estimated global pose to reflect this constraint. ORB-SLAM implements a number of enhancements to the feature-based BA concept introduced by PTAM [30]. As the name suggests, ORB-SLAM uses the ORB features and detectors [31] rather than the patches used in PTAM. The ORB-features and additional modifications enable loop closing that is more invariant to viewpoint and more robust feature initialization.

The previously discussed monocular SLAM implementations rely on features that are identified and tracked from monocular camera images. However, features only make up a small portion of the amount of information in an image; the remaining information is discarded. In addition, some environments do not contain a sufficient amount of readily-

identifiable features. Edge landmarks have been used instead of features in man-made environments where tracking edges is more robust [32]. The so-called direct monocular SLAM methods attempt to use all of the information contained in an image by estimating the camera pose using the image intensity (i.e. gradient). Large-Scale Direct Monocular SLAM (LSD-SLAM) was one of the first real-time direct monocular SLAM implementations [33]. The authors demonstrate accurate localization and semi-dense mapping on long indoor and outdoor trajectories. Semi-direct visual odometry (SVO) utilizes both direct image alignment and feature-based techniques [34].

1.3 Challenges of Monocular SLAM for Multirotor Navigation

A monocular camera image only contains information about the bearing of a point feature in the camera frame. Using a camera projection model, the azimuth and elevation angles of the ray that connects the camera origin to the point feature can be recovered. A simple model is shown in fig. 1.1 . This model, which does not account for distortion or pixel aspect ratio, requires knowledge of the lens focal length f and the principle point (c_x, c_y). These parameters can be determined by taking images of a known calibration target [35] or tracking distinct features [36].

Because monocular cameras are bearing-only sensors, monocular SLAM has two key challenges: scale ambiguity and feature initialization. The result of scale ambiguity is that the position estimates and the structure of the scene are only accurate up to a scale factor. If the camera is moving and the originally observed features are no longer in view then the scale factor estimate is likely to drift over time. Feature initialization is the process of estimating the relative position of a feature to other features and/or to the camera pose. From a single camera image, only the bearing to the feature can be determined. Fig 1.2 illustrates this challenge. Thus, two images of the feature, taken at two poses, are required to initialize a feature into the map. These two positions are separated by a distance known

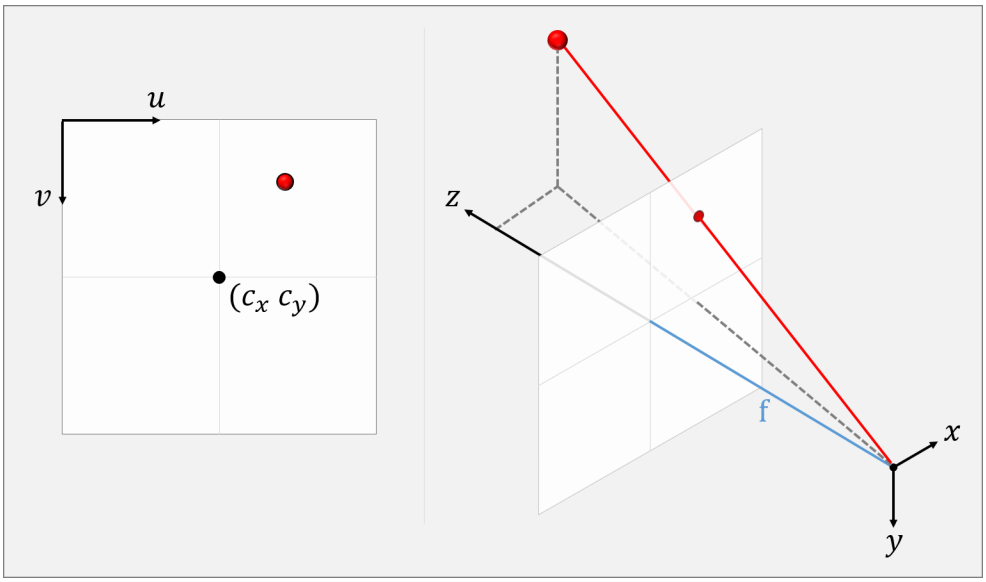

Figure 1.1: Pinhole projection model. The pixel coordinates u and v encode the azimuth and elevation angle of the ray (red line) that connects the camera origin to the point feature. The camera model relates the pixel coordinates of a projected feature to these angles.

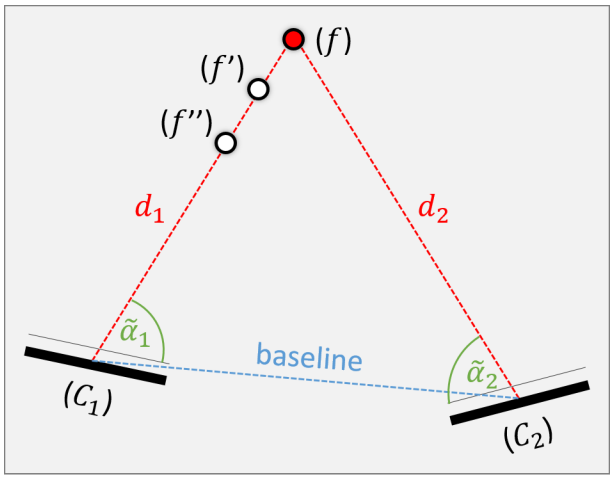

Figure 1.2: 2D representation of feature initialization. From a single camera image at pose C_1, only the bearing to f is observed. From this one image, f' or f'' are equally valid solutions. However, by taking another image at C_2, only f satisfies the constraints posed by the two images.

as the baseline. The baseline must be large enough that the difference between the two feature projections is measurable. The required baseline depends on the physical distance of the feature to the imaging plane and the focal length of the lens.

The scale ambiguity must be accounted for if monocular odometry will be used to close the loop on multirotor control. One option is to initialize SLAM using a priori information such as an initialization target [24]. This fixes the initial scale but does not prevent scale drift unless the target is always in view. The use of an initialization target may not be practical for autonomous multirotor flight.

An alternative solution to the scale ambiguity is the use of additional sensors. An IMU provides metric state estimates and has been used extensively as a complement to monocular SLAM. However, due to the integration of noise and errors in the estimated biases, the use of an IMU will not entirely eliminate scale drift, particularly if features are observed for short periods of time. The use of sensors that measure altitude can be used to resolve the scale with proper motion [37].

Feature initialization is also a serious concern for autonomous quadrotor navigation, particularly if there are no features currently initialized. Without manual control or a guidance law, the vehicle will not "know" that it needs to translate or the magnitude and direction of the required translation. If an IMU is the only navigation aid, the position of the vehicle will be highly uncertain. In this scenario, even if the vehicle knows that it needs to translate, it will have almost no idea about its own position. Thus even with sufficient translation to form a baseline, the amount of uncertainty in the baseline will result in feature initialization with an unusable amount of uncertainty and accuracy.

The challenges of scale ambiguity and initialization are particularly severe in three scenarios:

1. The quadrotor has just been powered on and is required to begin an autonomous

mission. It has no information about the environment. The vehicle must take off and begin the mission despite not having any position reference other than double integration from the IMU.

2. The vehicle has been flying for enough time that the original features it observed are no longer visible. Due to accumulated errors, the position and velocity states are drifting and becoming increasingly uncertain. New features are being initialized with increasing uncertainty and inaccuracy.

3. The vehicle pose suddenly changes relative to the features that it was observing such that none of these features are now visible. This is possible if, for instance, a vehicle with a forward-facing camera suddenly yaws 90 degrees. Now the multirotor has no information about the environment.

These three scenarios are examples of "significant environmental uncertainty." For vehicles without additional exteroceptive sensors, researchers have handled the first scenario by manually or remotely translating the vehicle and relying on the scale to converge from integrated accelerometer measurements [37]. This provides the baseline necessary to initialize features into the map but convergence is slow. The authors of [11] used a rangefinder and commanded a vertical motion. The scale is calculated using a closed-form solution and assumed to remain constant.

The second scenario is typically handled by using a downward-facing camera, using a wide-field-of-view lens, and limiting the trajectory to relatively small distances compared to the size of the initial map. This minimizes the drift of states, but limits the environments in which monocular SLAM can be used as the primary or complementary navigation solution.

Researchers recently addressed the third scenario in [38]. The authors demonstrate initialization after aggressive flight by stabilizing attitude and altitude (thanks to a downward

facing rangefinder) and allowing the subsequent position drift to provide the required baseline for feature initialization. While good results were obtained, this technique is limited to downward-facing cameras over flat ground.

1.4 An Integrated Monocular-Vision-Aided-Navigation and Guidance Solution

This work builds on these results and presents a solution that could provide improved robustness in these scenarios of significant environmental uncertainty. The robustness of monocular-vision-aided navigation to environmental uncertainty is improved through an altitude measurement and an integrated estimation and guidance solution. The specific methods and contributions of this work are:

1. A filter-based SLAM approach is presented. This implementation differs from previous work through the use of a unique feature initialization method that emphasizes the importance of not assuming a prior distribution for newly initialized feature states. The consider least squares algorithm [39] is used to estimate a covariance matrix that includes correlation with all current states. The feature is initialized if the largest eigenvalue of the feature covariance matrix is less than some threshold.

2. The system is tested using an experimental platform that includes a forward-facing monocular camera with wide field of view, downward-facing rangefinder, and inertial measurement unit. The platform is moved throughout a room in a trajectory and environment that is challenging for monocular-vision-aided navigation. A motion capture system tracks the global position and orientation for reference. The hardware results are compared to a simulation closely modeled after the hardware experiment.

3. A guidance law designed to reduce scale drift is introduced. Using the proposed feature initialization technique, features should not be initialized if there is not enough

information to confidently create a depth prior. Thus, features will not be initialized if metric information is needed. If the number of initialized features drops below a threshold, the vehicle is commanded to first stabilize attitude. This first step is similar to [38]. After stabilization, the vehicle is instructed to increase thrust. This results in increasing altitude and some inevitable drift parallel to the ground. A confident measurement of altitude, which could come from a rangefinder or other exteroceptive odometry, provides the scale necessary to initialize features. A sonic altimeter is considered in this work.

4. The proposed guidance law is analyzed using two simulated trajectories. The performance of the estimator with and without the altitude change is compared.

1.5 Overview of Sensor Package and SLAM Implementation

The full navigation sensor package consists of a MEMS IMU, forward-facing monocular camera with wide field of view (FOV) lens, and downward-facing rangefinder. Fig. 1.3 shows an example of such a system. This platform will be used for the hardware experiments in ch. 4.

A forward-facing camera was chosen because it represents a more general and challenging case than a downward-facing camera. Although a downward-facing camera has been used for monocular-vision-aided navigation for quadrotors, many ground surfaces lack a significant number of identifiable features. In addition, initialized features from a downward-facing camera cannot easily be used for obstacle avoidance unless the vehicle is moving downward. For this work, a downward-facing camera would be advantageous because the downward-facing rangefinder would directly provide metric information about the most uncertain (i.e. depth) direction. Thus the methods proposed in this work could also be applied to a downward-facing camera.

In this navigation solution, the 6 DOF pose of the IMU must be estimated. In order

Figure 1.3: Sensors mounted on experimental setup.

to take advantage of features in the camera images, the 3D locations of these features must be simultaneously estimated. At any point in time the SLAM algorithm contains estimates of the IMU pose and the global position of m features. This information is depicted graphically in fig. 1.4 . Estimates for the IMU velocity, gyroscope bias, and accelerometer bias must also be maintained in order to use the IMU measurements.

Camera images are received every so often. In this work, the first received image will be designated as the "keyframe image". Some number of features will be identified in the keyframe image (pink circles in fig. 1.4). The initial IMU pose estimate is recorded and designated as the keyframe pose.

Some time later a new image will become available. However, before the arrival of the new image, IMU and/or rangefinder measurements will have been received. These measurements will be used to propagate and/or update, respectively, the filter state. Thus by the time the new image is received, there exists a prior estimate of the current IMU pose. In this new image, features are again detected. The feature descriptors are matched with some of the feature descriptors previously found in the keyframe image (pink lines in fig. 1.4). Using the current IMU pose estimate, the keyframe pose estimate, and the measured pixel locations of the feature in the current and keyframe images, the global

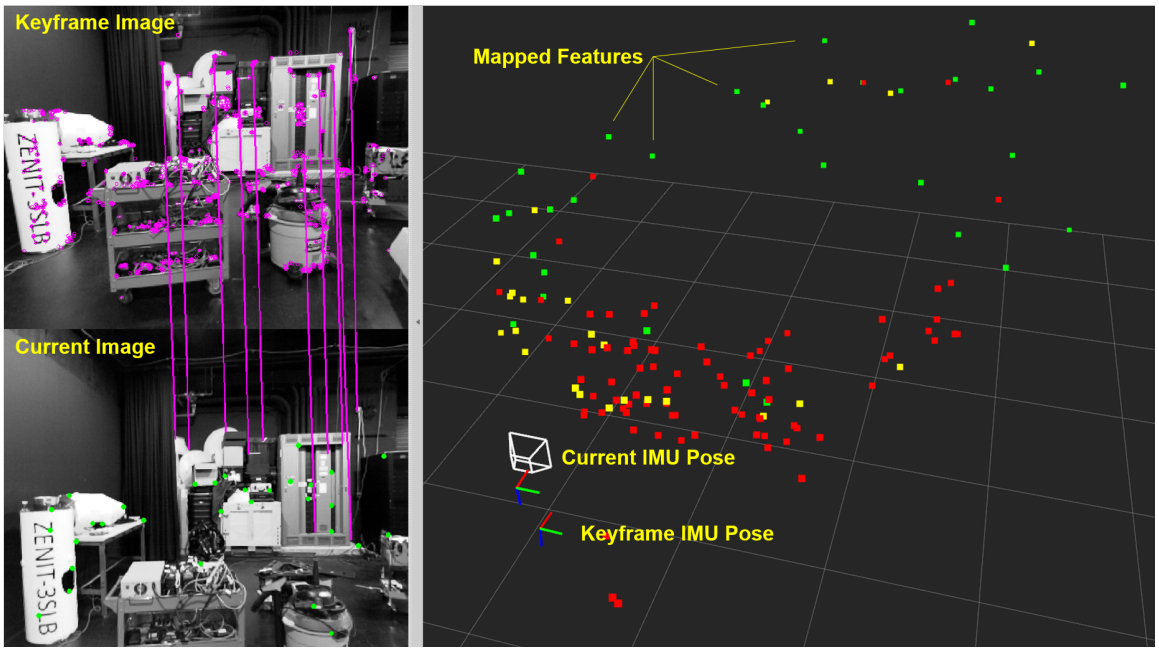

Figure 1.4: The pose and images corresponding to the keyframe and current IMU poses. At each pose, the camera observes the 2D projection of features. New features are matched with features that were previously observed in the keyframe image (pink lines and circles). The observations of mapped features (green circles) are used to update the filter state. Each mapped feature has an associated 3D position (green, yellow, and red squares).

position of the feature is triangulated. Using the consider least squares approach, the uncertainty of this triangulated feature position is estimated. If the uncertainty of the depth of the feature is below a threshold, the feature descriptor and associated position estimate are added to the "map" by appending it to the state vector.

Soon another image will become available. Feature matches will once again be made between this image and the keyframe image. In addition, features matches should be made to the mapped feature descriptors. These observations of mapped features are used to update the filter state.

This process is repeated as new IMU data, rangefinder measurements, and images are received. As the camera moves, there will become increasingly fewer matches between

the current image and the keyframe image. Thus, the keyframe image and pose will need to be changed every so often.

Ch. 2 describes how IMU, rangefinder, and camera measurements are used to update the filter state. Ch. 3 details the process for matching features and delayed feature initialization. It also explains how the state is dynamically augmented with the keyframe pose and new features. The monocular-vision-aided navigation system is tested using experimental and simulated data. The results are presented in ch. 4. Using this system, the guidance law is proposed and tested. This is explained in ch. 5. Conclusions are provided in ch. 6.

Though the vision-aided navigation solution provided here produces accurate estimates with simulated and hardware experimental data, it is not intended to be a "complete" SLAM solution that would work in every environment. There are many improvements that could be made, many of which are described throughout this work. Rather, the present implementation contains the elements necessary to demonstrate and complement the proposed guidance law on a multirotor platform with a specific sensor package.

2. MONOCULAR VISION-AIDED NAVIGATION IN AN EXTENDED KALMAN FILTER FRAMEWORK

As described in ch. 1, there is an extensive body of work on monocular SLAM. There are dozens of methods that can be used. For this work, an extended Kalman filter framework was chosen. As opposed to other classes of monocular SLAM solutions (e.g. sparse bundle adjustment methods), the EKF-framework readily allows multiple sensor measurements to be "fused" together without the need for a separate filter as in [9] and [34]. In addition, using the IMU directly helps the odometry to "distinguish" between a rotation and a translation (specifically transformations with significant components in directions perpendicular to the camera's optical axis). Finally, many bundle adjustment methods do not provide a covariance or pose uncertainty metric. Having a confidence metric is important in many robotic and aerospace applications. For these reasons, an EKF-SLAM solution was designed and implemented for this work.

2.1 Background and Notation

This section describes the notational conventions used in this work. Various attitude parameterizations and approximations are presented. Finally, a careful distinction is made between the true state, estimated state, and error states of the Kalman filter.

2.1.1 General Variable Conventions

In this work, vectors and matrices are written as bold Roman characters. Vectors are lowercases letters (e.g. $\mathbf{v}$). Matrices are uppercase letters (e.g. $\mathbf{A}$). The element of the r^{th} row and c^{th} column of matrix $\mathbf{A}$ is denoted by A_{rc}. Scalars are written as lowercase italic letters or symbolic characters (e.g. α). Reference frames are written using uppercase italic characters (e.g. F). The characters i and j are reserved to represent elements of a set or

rows/columns of a matrix.

2.1.2 Attitude and Reference Frames

The quaternion is used to express the rotation from one frame to another. It consists of an imaginary vector component $\mathbf{q}$ and a scalar component q_4.

$$\bar{q} = \begin{bmatrix} \mathbf{q} \\ q_4 \end{bmatrix} = \begin{bmatrix} q_1 & q_2 & q_3 & q_4 \end{bmatrix}^{\mathrm{T}} \tag{2.1}$$

The rotation between two frames, A and B, is represented by ${}^{B}_{A}\bar{q}$. A vector expressed in the A frame can be coordinatized in the B frame using the direction cosine matrix ${}^{B}_{A}\mathbf{C} = \mathbf{C}({}^{B}_{A}\bar{q})$.

$${}^{B}\mathbf{v} = {}^{B}_{A}\mathbf{C}\ {}^{A}\mathbf{v} \tag{2.2}$$

$$\mathbf{C}(\bar{q}) = \mathbf{I}_{3\times 3} - 2q_4\lfloor \mathbf{q} \times \rfloor + 2\lfloor \mathbf{q} \times \rfloor \lfloor \mathbf{q} \times \rfloor \tag{2.3}$$

Where $\lfloor \mathbf{q} \times \rfloor$ is the skew symmetric matrix composed of the elements of $\mathbf{q}$. In general, the skew symmetric matrix $\lfloor \mathbf{a} \times \rfloor$ formed from the matrix $\mathbf{a} = \begin{bmatrix} a_1 & a_2 & a_3 \end{bmatrix}^{\mathrm{T}}$ has the property that $\lfloor \mathbf{a} \times \rfloor = -\lfloor \mathbf{a} \times \rfloor^{\mathrm{T}}$ and

$$\lfloor \mathbf{a} \times \rfloor = \begin{bmatrix} 0 & -a_3 & a_2 \\ a_3 & 0 & -a_1 \\ -a_2 & a_1 & 0 \end{bmatrix} \tag{2.4}$$

The rotation matrix ${}^{B}_{A}\mathbf{R} = {}^{B}_{A}\mathbf{C}^{\mathrm{T}}$ and

$${}^{A}\mathbf{v} = {}^{B}_{A}\mathbf{R}\ {}^{B}\mathbf{v} = \mathbf{R}({}^{B}_{A}\bar{q})\ {}^{B}\mathbf{v} \tag{2.5}$$

The quaternion is closely related to the axis-angle attitude representation. The axis-

angle parameterization consists of a rotation θ about the unit vector $\hat{\mathbf{k}}$.

$$\bar{q} = \begin{bmatrix} \mathbf{q} \\ q_4 \end{bmatrix} = \begin{bmatrix} \sin(\theta/2)\ \hat{\mathbf{k}} \\ \cos(\theta/2) \end{bmatrix} \tag{2.6}$$

Additionally,

$$\mathbf{C}(\hat{\mathbf{k}}, \theta) = \cos(\theta)\mathbf{I}_{3\times 3} - \sin(\theta)\lfloor \hat{\mathbf{k}} \times \rfloor + (1 - \cos(\theta)\ \hat{\mathbf{k}}\hat{\mathbf{k}}^{\mathrm{T}}) \tag{2.7}$$

In the case of a small rotation $\delta\bar{q}$, the rotation angle θ is also small. The small angle approximation can be used to rewrite eq. (2.6) as

$$\delta\bar{q} = \begin{bmatrix} \sin(\theta/2)\hat{\mathbf{k}} \\ \cos(\theta/2) \end{bmatrix} \approx \begin{bmatrix} \frac{1}{2}\theta\hat{\mathbf{k}} \\ 1 \end{bmatrix} = \begin{bmatrix} \frac{1}{2}\delta\boldsymbol{\theta} \\ 1 \end{bmatrix} \tag{2.8}$$

where $\delta\boldsymbol{\theta} = \theta\hat{\mathbf{k}}$.

Similarly, eq. (2.7) can be approximated as

$$\mathbf{C} \approx \mathbf{I}_{3\times 3} - \lfloor \delta\boldsymbol{\theta} \times \rfloor \qquad \mathbf{C}^{\mathrm{T}} \approx \mathbf{I}_{3\times 3} + \lfloor \delta\boldsymbol{\theta} \times \rfloor \tag{2.9}$$

Eq. (2.9) can also be found by substituting eq. (2.8) into eq. (2.3) and neglecting the small $\lfloor \mathbf{q} \times \rfloor \lfloor \mathbf{q} \times \rfloor$ term.

A vector between the origin of frame A and the origin of frame B is written as $\mathbf{v}_A^B$. If this vector is coordinatized in the A frame it can be written as ${}^A\mathbf{v}_A^B$. This will usually be simplified to ${}^A\mathbf{v}_B$

2.1.3 True States, Estimated States, and Error States

The Extended Kalman filter (EKF) extends the results of the Kalman filter to nonlinear system models. The filter is driven by a system model and a linearized approximation of

the associated error model.

A nonlinear system model with continuous dynamics and discrete measurements is considered:

$$\dot{\mathbf{x}}(t) = \mathbf{f}(\mathbf{x}(t), \mathbf{u}(t), \mathbf{w}(t), t) \tag{2.10}$$

$$E[\mathbf{w}(t)] = \mathbf{0} \tag{2.11}$$

$$E[\mathbf{w}(t+\tau)\mathbf{w}^{\mathrm{T}}(t)] = \mathbf{Q}(t)\delta(\tau) \tag{2.12}$$

$$\tilde{\mathbf{y}}_k = \mathbf{h}(\mathbf{x}_k) + \mathbf{v}_k \tag{2.13}$$

$$E[\mathbf{v}_k] = \mathbf{0} \tag{2.14}$$

$$E[\mathbf{v}_k \mathbf{v}_k^{\mathrm{T}}] = \mathbf{R}_k \tag{2.15}$$

Where $\mathbf{w}(t)$ and $\mathbf{v}_k$ are the process and measurement noise, respectively. $E[g]$ is the expected value of a random variable g.

The true state $\mathbf{x}(t)$ is unknown. The EKF algorithm uses the system model to provide a state estimate $\hat{\mathbf{x}}(t)$ and measurement estimate $\hat{\mathbf{y}}_k$. The error state $\breve{\mathbf{x}}(t)$ encodes the error between the truth and estimates. Similarly, the measurement error $\breve{\mathbf{y}}_k$ represents the error between the actual measurement and the estimated measurement. Though the error state and measurement error are unknown (except for simulated cases), it is possible to develop a mathematical model for the errors. In this work, two types of errors will be considered: additive and multiplicative. Additive error is denoted with Δ. The additive error between a vector $\mathbf{v}$ and its estimate $\hat{\mathbf{v}}$ is

$$\Delta\mathbf{v} = \mathbf{v} - \hat{\mathbf{v}} \tag{2.16}$$

The error in a quaternion is better represented as multiplicative error. Multiplicative

error is defined using δ. The quaternion error $\delta\bar{q}$ is defined as

$$\delta\bar{q}(t) = \bar{q}(t) \otimes \hat{\bar{q}}^{-1}(t) \tag{2.17}$$

Where $\otimes$ indicates quaternion multiplication and $\hat{\bar{q}}^{-1}$ is the quaternion inverse of $\hat{\bar{q}}$.

Similarly, the error in an attitude matrix is defined as

$$\delta\mathbf{C} = \mathbf{C}\hat{\mathbf{C}}^{\mathrm{T}} \tag{2.18}$$

$\breve{\mathbf{x}}(t)$ and $\breve{\mathbf{y}}_k$ may, in general, contain states and measurements, respectively, that are modeled with additive or multiplicative error.

The EKF requires a linearized approximation of the error state and measurement error:

$$\dot{\breve{\mathbf{x}}}(t) \approx \mathbf{F}\breve{\mathbf{x}}(t) + \mathbf{G}\mathbf{w}(t) = \mathbf{F}(\hat{\mathbf{x}}(t), \mathbf{u}(t), t)\ \breve{\mathbf{x}}(t) + \mathbf{G}(\hat{\mathbf{x}}(t), \mathbf{u}(t), t)\ \mathbf{w}(t) \tag{2.19}$$

$$\breve{\mathbf{y}}_k \approx \mathbf{H}\breve{\mathbf{x}} + \mathbf{v}_k = \mathbf{H}(\hat{\mathbf{x}}_k)\ \breve{\mathbf{x}} + \mathbf{v}_k \tag{2.20}$$

The $\mathbf{F}$, $\mathbf{G}$, and $\mathbf{H}$ matrices are used in the Kalman filter to approximate the covariance $\mathbf{P}$ of the error states, where

$$\mathbf{P}(t) = E[\breve{\mathbf{x}}(t)\breve{\mathbf{x}}^{\mathrm{T}}(t)] \tag{2.21}$$

2.2 EKF SLAM System State Models

In this work, the state of the system is composed of the IMU state $\mathbf{x}_I$, the kefyrame state $\mathbf{x}_L$, and the feature state $\mathbf{x}_F$. Some of the pose states are illustrated in fig. 2.1 .

$$\mathbf{x} = \begin{bmatrix} \mathbf{x}_I \\ \mathbf{x}_L \\ \mathbf{x}_F \end{bmatrix} \tag{2.22}$$

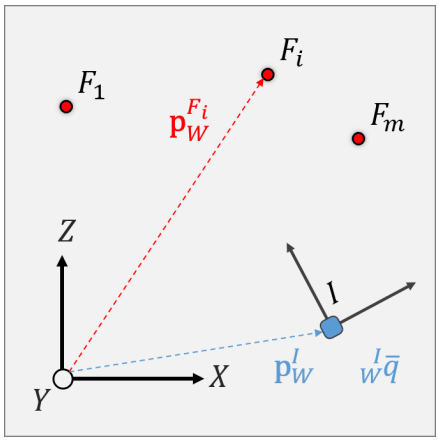

Figure 2.1: 2D diagram of the current IMU pose and feature positions.

$\mathbf{x}_I$ consists of the quaternion representing the attitude of the IMU in the world frame, the position of the IMU in the world frame, the velocity of the IMU in the world frame, the gyroscope bias, and the accelerometer bias. The third axis of the world frame is aligned with gravity.

$$\mathbf{x}_I = \begin{bmatrix} {}_W^I\bar{q}^{\mathrm{T}} & {}^W\mathbf{p}_I^{\mathrm{T}} & {}^W\mathbf{v}_I^{\mathrm{T}} & \mathbf{b}_g^{\mathrm{T}} & \mathbf{b}_a^{\mathrm{T}} \end{bmatrix}^{\mathrm{T}} \tag{2.23}$$

The feature map consists of m features $F_1, ...F_i, ...F_m$. $\mathbf{x}_F$ contains the position of each mapped feature in the world frame.

$$\mathbf{x}_F = \begin{bmatrix} {}^W\mathbf{p}_{F_1}^{\mathrm{T}} & \dots & {}^W\mathbf{p}_{F_i}^{\mathrm{T}} & \dots & {}^W\mathbf{p}_{F_m}^{\mathrm{T}} \end{bmatrix}^{\mathrm{T}} \tag{2.24}$$

In order to initialize features, it is necessary to observe the features in two or more images and to have an estimate of the baseline between the images. Features are first observed in the keyframe image. The pose of the IMU when the keyframe image was recorded is the keyframe pose. As the camera moves, features are detected and matched with features in the keyframe image. The position and orientation of the IMU in the world frame when the keyframe image was taken make up the keyframe state $\mathbf{x}_L$. The purpose

and usage of the keyframe state is discussed in more detail in sec. 3.2.

$$\mathbf{x}_L = \begin{bmatrix} {}^{I_L}_{W}\bar{q}^{\mathrm{T}} & {}^{W}\mathbf{p}^{\mathrm{T}}_{I_L} \end{bmatrix}^{\mathrm{T}} \tag{2.25}$$

The Kalman filter uses the process model to estimate the state and covariance of the system at some time t_{k+1} given the estimates at some previous time t_k. In this work, the process model is driven by IMU measurements as the inputs. The expected value of the process model is used to provide estimates for the rate of change of the state estimates. Using the process model, estimates, and the previously defined error definitions, the error state equations are linearly approximated.

2.2.1 System Model

The gyroscope provides a measurement of the angular velocity $\tilde{\boldsymbol{\omega}}(t)$. $\tilde{\boldsymbol{\omega}}(t)$ relates to the true angular velocity $\boldsymbol{\omega}(t)$ according to the following model:

$$\boldsymbol{\omega}(t) = \tilde{\boldsymbol{\omega}}(t) - \mathbf{b}_g(t) - \mathbf{n}_g(t) \tag{2.26}$$

$$\dot{\mathbf{b}}_g(t) = \mathbf{n}_{wg}(t) \tag{2.27}$$

$\mathbf{n}_g(t)$ and $\mathbf{n}_{wg}(t)$ are white noise processes.

The rotation from the world frame to the current IMU frame, ${}^{I}_{W}\bar{q}(t)$, is desired. However, between t_k and t_{k+1}, integration of the gyroscope measurements provides ${}^{I}_{I_k}\bar{q}$, the rotation from the IMU frame at time t_k to the IMU frame at time t. These poses are illustrated in fig. 2.2 . ${}^{I}_{W}\bar{q}(t)$ can be recovered using the relationship

$${}^{I}_{W}\bar{q}(t) = {}^{I}_{I_k}\bar{q}(t) \otimes {}^{I_k}_{W}\bar{q} \tag{2.28}$$

Where $\otimes$ indicates quaternion multiplication.

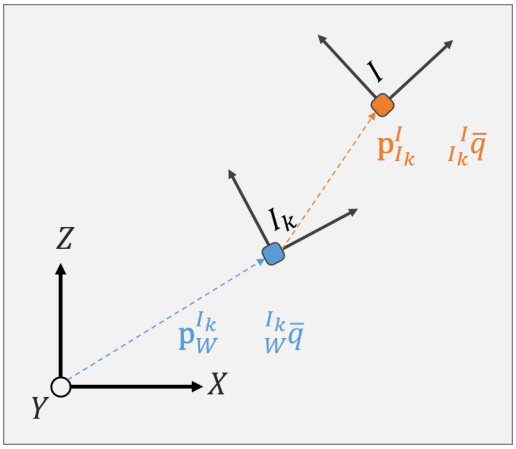

Figure 2.2: Between time t_k and t_{k+1}, the IMU translates according to $\mathbf{p}^I_{I_k}$ and undergoes a rotation parameterized by ${}^I_{I_k}\bar{q}$.

The angular velocity of the IMU is related to ${}^I_{I_k}\dot{\bar{q}}(t)$:

$$ {}^I_{I_k}\dot{\bar{q}}(t) = \frac{1}{2}\begin{bmatrix} -\lfloor \boldsymbol{\omega}(t) \times \rfloor & \boldsymbol{\omega}(t) \\ -\boldsymbol{\omega}^{\mathrm{T}}(t) & 0 \end{bmatrix} {}^I_{I_k}\bar{q}(t) \tag{2.29} $$

The accelerometer provides a measurement of the specific force $\tilde{\mathbf{s}}(t)$. $\tilde{\mathbf{s}}(t)$ relates to the true specific force according to the following model:

$$ \mathbf{s}(t) = \tilde{\mathbf{s}}(t) - \mathbf{b}_a(t) - \mathbf{n}_a(t) \tag{2.30} $$

$$ \dot{\mathbf{b}}_a(t) = \mathbf{n}_{wa}(t) \tag{2.31} $$

$\mathbf{n}_a(t)$ and $\mathbf{n}_{wa}(t)$ are white noise processes.

The acceleration of the IMU is modeled as

$$ {}^W\dot{\mathbf{v}}_I(t) = {}^W\mathbf{a}_I(t) = ({}^I_{I_k}\mathbf{C}\ {}^{I_k}_W\mathbf{C})^{\mathrm{T}}\mathbf{s}(t) + {}^W\mathbf{g} \tag{2.32} $$

$$ = {}^{I_k}_W\mathbf{C}^{\mathrm{T}}\ {}^I_{I_k}\mathbf{C}^{\mathrm{T}}\mathbf{s}(t) + {}^W\mathbf{g} \tag{2.33} $$

where

$$ {}^{I}_{I_k}\mathbf{C} \equiv \mathbf{C}\left({}^{I}_{I_k}\bar{q}(t)\right), \quad {}^{I_k}_{W}\mathbf{C} \equiv \mathbf{C}\left({}^{I_k}_{W}\bar{q}\right) \tag{2.34} $$

The rate of change of the IMU position is equivalent to the IMU velocity

$$ {}^{W}\dot{\mathbf{p}}_I = {}^{W}\mathbf{v}_I \tag{2.35} $$

The keyframe pose of the IMU does not change over time:

$$ \dot{\mathbf{x}}_L = \mathbf{0} \tag{2.36} $$

Similarly, the feature positions are static and do not change:

$$ \dot{\mathbf{x}}_F = \mathbf{0} \tag{2.37} $$

2.2.2 State Estimates

The EKF will use IMU measurements as inputs with which to evaluate the rates of the IMU state estimates. IMU measurements are only available at discrete points in time, t_k and t_{k+1}. At times t_k and t_{k+1}, the IMU provides measurements of the angular velocity $\tilde{\boldsymbol{\omega}}_k$ and $\tilde{\boldsymbol{\omega}}_{k+1}$, respectively. The IMU also provides measurements of the specific forces $\tilde{\mathbf{s}}_k$ and $\tilde{\mathbf{s}}_{k+1}$ at t_k and t_{k+1}, respectively.

The IMU state estimates $\hat{\mathbf{x}}_I$ are considered first

$$ \hat{\mathbf{x}}_I = \begin{bmatrix} {}^{I}_{W}\hat{\bar{q}}^{\mathrm{T}} & {}^{W}\hat{\mathbf{p}}_I^{\mathrm{T}} & {}^{W}\hat{\mathbf{v}}_I^{\mathrm{T}} & \hat{\mathbf{b}}_g^{\mathrm{T}} & \hat{\mathbf{b}}_a^{\mathrm{T}} \end{bmatrix}^{\mathrm{T}} \tag{2.38} $$

In order to develop equations for the rate of change of the attitude, the measured angu-

lar velocity is first linearly interpolated:

$$\tilde{\boldsymbol{\omega}}(t) = \tilde{\boldsymbol{\omega}}_k + (\tilde{\boldsymbol{\omega}}_{k+1} - \tilde{\boldsymbol{\omega}}_k)\frac{t - t_k}{t_{k+1} - t_k}, \qquad t_k \leq t \leq t_{k+1} \tag{2.39}$$

Taking the expected value of eq. (2.26) and eq. (2.27) , the estimates $\hat{\boldsymbol{\omega}}(t)$ and $\hat{\mathbf{b}}_g(t)$ are

$$\dot{\hat{\mathbf{b}}}_g(t) = \mathbf{0} \rightarrow \hat{\mathbf{b}}_g(t) = \hat{\mathbf{b}}_{g_k} \tag{2.40}$$

$$\hat{\boldsymbol{\omega}}(t) = \tilde{\boldsymbol{\omega}}(t) - \hat{\mathbf{b}}_g(t) = \tilde{\boldsymbol{\omega}}(t) - \hat{\mathbf{b}}_{g_k} \tag{2.41}$$

Using $\hat{\boldsymbol{\omega}}(t)$ and ${}^{I}_{I_k}\hat{\bar{q}}(t)$ in eq. (2.29) provides the estimate ${}^{I}_{I_k}\dot{\hat{\bar{q}}}(t)$:

$${}^{I}_{I_k}\dot{\hat{\bar{q}}}(t) = \frac{1}{2}\begin{bmatrix} -\lfloor \hat{\boldsymbol{\omega}}(t) \times \rfloor & \hat{\boldsymbol{\omega}}(t) \\ -\hat{\boldsymbol{\omega}}^{\mathrm{T}}(t) & 0 \end{bmatrix} {}^{I}_{I_k}\hat{\bar{q}}(t) \tag{2.42}$$

In order to develop equations for the rate of change of the velocity, the measured specific force is linearly interpolated:

$$\tilde{\mathbf{s}}(t) = \tilde{\mathbf{s}}_k + (\tilde{\mathbf{s}}_{k+1} - \tilde{\mathbf{s}}_k)\frac{t - t_k}{t_{k+1} - t_k}, \qquad t_k \leq t \leq t_{k+1} \tag{2.43}$$

Taking the expected value of eq. (2.30) and eq. (2.31) , the estimates $\hat{\mathbf{s}}(t)$ and $\hat{\mathbf{b}}_a(t)$ are

$$\dot{\hat{\mathbf{b}}}_a(t) = \mathbf{0} \rightarrow \hat{\mathbf{b}}_a(t) = \hat{\mathbf{b}}_{a_k} \tag{2.44}$$

$$\hat{\mathbf{s}}(t) = \tilde{\mathbf{s}}(t) - \hat{\mathbf{b}}_a(t) = \tilde{\mathbf{s}}(t) - \hat{\mathbf{b}}_{a_k} \tag{2.45}$$

Using the state estimates in eq. (2.32) yields the estimate ${}^{W}\dot{\hat{\mathbf{v}}}_I(t)$

$${}^{W}\dot{\hat{\mathbf{v}}}_I(t) = {}^{I_k}_{W}\hat{\mathbf{C}}^{\mathrm{T}}\,{}^{I}_{I_k}\hat{\mathbf{C}}^{\mathrm{T}}\hat{\mathbf{s}}(t) + {}^{W}\mathbf{g} \tag{2.46}$$

The keyframe state estimate follows from eq. (2.36)

$$\dot{\hat{\mathbf{x}}}_L = \mathbf{0} \tag{2.47}$$

Similarly, the feature state estimate follows from eq. (2.37)

$$\dot{\hat{\mathbf{x}}}_F = \mathbf{0} \tag{2.48}$$

2.2.3 Error Model

For the covariance propagation, it is necessary to find the $\mathbf{F}$ and $\mathbf{G}$ matrices as defined in eq. (2.19) . The IMU error states will be considered first. To work in the EKF form, it is necessary to find an equation for $\dot{\breve{\mathbf{x}}}_I$ that is a linear combination of the error states and noise vector:

$$\dot{\breve{\mathbf{x}}}_I \approx \mathbf{F}_I \breve{\mathbf{x}}_I + \mathbf{G}_I \mathbf{w} \tag{2.49}$$

The noise vector $\mathbf{w}$ is composed of the noise states:

$$\mathbf{w} = \begin{bmatrix} \mathbf{n}_g^{\mathrm{T}} & \mathbf{n}_a^{\mathrm{T}} & \mathbf{n}_{wg}^{\mathrm{T}} & \mathbf{n}_{wa}^{T} \end{bmatrix}^T \tag{2.50}$$

The IMU error state $\breve{\mathbf{x}}_I$ is composed of the individual error states:

$$\breve{\mathbf{x}}_I = \begin{bmatrix} \delta\boldsymbol{\theta}_I^{\mathrm{T}} & \Delta\mathbf{p}_I^{\mathrm{T}} & \Delta\mathbf{v}_I^{\mathrm{T}} & \Delta\mathbf{b}_g^{\mathrm{T}} & \Delta\mathbf{b}_a^{\mathrm{T}} \end{bmatrix}^{\mathrm{T}}, \tag{2.51}$$

The position, velocity, and bias error terms use additive error. The attitude error ($\delta\bar{q}$) is modeled as a multiplicative error:

$${}^{I}_{W}\bar{q} = \delta\bar{q} \otimes {}^{I}_{W}\hat{\bar{q}} \tag{2.52}$$

$$ {}^{I}_{W}\mathbf{C} = \mathbf{C}(\delta\bar{q})\,{}^{I}_{W}\hat{\mathbf{C}} \tag{2.53} $$

The error is assumed to be small and the small angle approximation of eq. (2.9) is used to define the attitude angle error $\delta\boldsymbol{\theta}_I$:

$$ \mathbf{C}(\delta\bar{q}) \approx \mathbf{C}(\delta\boldsymbol{\theta}_I) = \mathbf{I}_{3\times 3} - \lfloor \delta\boldsymbol{\theta}_I \times \rfloor \tag{2.54} $$

It can be shown (see [40], sec. 2.4), that the rate of change of the attitude error $\delta\dot{\boldsymbol{\theta}}_I$ can be approximated as a linear combination of the error states and noise:

$$ \delta\dot{\boldsymbol{\theta}}_I = -\hat{\boldsymbol{\omega}} \times \delta\boldsymbol{\theta}_I - \Delta\mathbf{b}_a - \mathbf{n}_g \tag{2.55} $$

Using eqs. (2.32), (2.53), and (2.54):

$$ \begin{aligned} {}^{W}\dot{\mathbf{v}}_I &= {}^{I}_{W}\mathbf{C}^{\mathrm{T}}(\mathbf{s}) + {}^{W}\mathbf{g} \\ &= {}^{I}_{W}\mathbf{C}^{\mathrm{T}}(\tilde{\mathbf{s}} - \mathbf{b}_a - \mathbf{n}_a) + {}^{W}\mathbf{g} \\ &\approx {}^{I}_{W}\hat{\mathbf{C}}^{\mathrm{T}}(\mathbf{I}_{3\times 3} + \lfloor \delta\boldsymbol{\theta}_I \times \rfloor)(\tilde{\mathbf{s}} - \mathbf{b}_a - \mathbf{n}_a) + {}^{W}\mathbf{g} \\ &= {}^{I}_{W}\hat{\mathbf{C}}^{\mathrm{T}}(\mathbf{I}_{3\times 3} + \lfloor \delta\boldsymbol{\theta}_I \times \rfloor)(\tilde{\mathbf{s}} - \hat{\mathbf{b}}_a - \Delta\mathbf{b}_a - \mathbf{n}_a) + {}^{W}\mathbf{g} \\ &= {}^{I}_{W}\hat{\mathbf{C}}^{\mathrm{T}}(\tilde{\mathbf{s}} - \hat{\mathbf{b}}_a - \Delta\mathbf{b}_a - \mathbf{n}_a) + {}^{I}_{W}\hat{\mathbf{C}}^{\mathrm{T}}\lfloor \delta\boldsymbol{\theta}_I \times \rfloor(\tilde{\mathbf{s}} - \hat{\mathbf{b}}_a - \Delta\mathbf{b}_a - \mathbf{n}_a) + {}^{W}\mathbf{g} \end{aligned} \tag{2.56} $$

Using eqs. (2.46) and (2.56), $\Delta\dot{\mathbf{v}}_I$ can be written as

$$ \begin{aligned} \Delta\dot{\mathbf{v}}_I &= {}^{W}\dot{\mathbf{v}}_I - {}^{W}\dot{\hat{\mathbf{v}}}_I \\ &= {}^{I}_{W}\hat{\mathbf{C}}^{\mathrm{T}}\big(- \Delta\mathbf{b}_a - \mathbf{n}_a + \lfloor \delta\boldsymbol{\theta}_I \times \rfloor(\tilde{\mathbf{s}} - \hat{\mathbf{b}}_a - \Delta\mathbf{b}_a - \mathbf{n}_a)\big) \end{aligned} \tag{2.57} $$

Some interesting properties of eq. (2.57) will be discussed in sec. 5.1. For now,

eq. (2.57) can be approximated as

$$\Delta\dot{\mathbf{v}}_I \approx -{}^I_W\hat{\mathbf{C}}^{\mathrm{T}}(\Delta\mathbf{b}_a + \mathbf{n}_a + \lfloor(\hat{\mathbf{s}} - \hat{\mathbf{b}}_a)\times\rfloor\delta\boldsymbol{\theta}_I) \tag{2.58}$$

Using the additive error definition, equations for the position and bias error rates are easily derived:

$$\Delta\dot{\mathbf{p}}_I = {}^W\dot{\mathbf{p}}_I - {}^W\dot{\hat{\mathbf{p}}}_I = \Delta\mathbf{v}_I \tag{2.59}$$

$$\Delta\dot{\mathbf{b}}_g = \mathbf{b}_g - \hat{\mathbf{b}}_g = \mathbf{n}_{wg} \tag{2.60}$$

$$\Delta\dot{\mathbf{b}}_a = \mathbf{b}_a - \hat{\mathbf{b}}_a = \mathbf{n}_{wa} \tag{2.61}$$

Using eqs. (2.55)-(2.61), $\dot{\breve{\mathbf{x}}}_I$ can be written in the form of eq. (2.49) , with

$$\mathbf{F}_I = \begin{bmatrix} -\lfloor\hat{\boldsymbol{\omega}}\times\rfloor & \mathbf{0}_{3\times3} & \mathbf{0}_{3\times3} & -\mathbf{I}_{3\times3} & \mathbf{0}_{3\times3} \\ \mathbf{0}_{3\times3} & \mathbf{0}_{3\times3} & \mathbf{I}_{3\times3} & \mathbf{0}_{3\times3} & \mathbf{0}_{3\times3} \\ -{}^I_W\hat{\mathbf{C}}\lfloor(\tilde{\mathbf{s}} - \hat{\mathbf{b}}_a)\times\rfloor & \mathbf{0}_{3\times3} & \mathbf{0}_{3\times3} & \mathbf{0}_{3\times3} & -{}^I_W\hat{\mathbf{C}} \\ \mathbf{0}_{3\times3} & \mathbf{0}_{3\times3} & \mathbf{0}_{3\times3} & \mathbf{0}_{3\times3} & \mathbf{0}_{3\times3} \\ \mathbf{0}_{3\times3} & \mathbf{0}_{3\times3} & \mathbf{0}_{3\times3} & \mathbf{0}_{3\times3} & \mathbf{0}_{3\times3} \end{bmatrix} \tag{2.62}$$

$$\mathbf{G}_I = \begin{bmatrix} -\mathbf{I}_{3\times3} & \mathbf{0}_{3\times3} & \mathbf{0}_{3\times3} & \mathbf{0}_{3\times3} \\ \mathbf{0}_{3\times3} & \mathbf{0}_{3\times3} & \mathbf{0}_{3\times3} & \mathbf{0}_{3\times3} \\ \mathbf{0}_{3\times3} & -{}^I_W\hat{\mathbf{C}} & \mathbf{0}_{3\times3} & \mathbf{0}_{3\times3} \\ \mathbf{0}_{3\times3} & \mathbf{0}_{3\times3} & \mathbf{I}_{3\times3} & \mathbf{0}_{3\times3} \\ \mathbf{0}_{3\times3} & \mathbf{0}_{3\times3} & \mathbf{0}_{3\times3} & \mathbf{I}_{3\times3} \end{bmatrix} \tag{2.63}$$

Next the keyframe state error is considered. Using eqs. (2.36) and (2.47):

$$\dot{\breve{\mathbf{x}}}_L = \dot{\mathbf{x}}_L - \dot{\hat{\mathbf{x}}}_L = \mathbf{F}_L\breve{\mathbf{x}}_L + \mathbf{G}_L\mathbf{w} = \mathbf{0}_{6\times1} \tag{2.64}$$

Then

$$\mathbf{F}_L = \mathbf{0}_{6\times6} \tag{2.65}$$

$$\mathbf{G}_L = \mathbf{0}_{6\times 6} \tag{2.66}$$

Now the feature error state is considered. Using eqs. (2.37) and (2.48):

$$\dot{\breve{\mathbf{x}}}_F = \dot{\mathbf{x}}_F - \dot{\hat{\mathbf{x}}}_F = \mathbf{F}_F \breve{\mathbf{x}}_F + \mathbf{G}_F \mathbf{w} = \mathbf{0}_{3m\times 1} \tag{2.67}$$

Then

$$\mathbf{F}_F = \mathbf{0}_{3m\times 3m} \tag{2.68}$$

$$\mathbf{G}_F = \mathbf{0}_{3m\times 12} \tag{2.69}$$

The linearized error state equation for the full state can be written:

$$\mathbf{F} = \begin{bmatrix} \mathbf{F}_I & \mathbf{0}_{15\times 6} & \mathbf{0}_{15\times 3m} \\ \mathbf{0}_{6\times 15} & \mathbf{F}_L & \mathbf{0}_{6\times 3m} \\ \mathbf{0}_{3m\times 15} & \mathbf{0}_{3m\times 6} & \mathbf{F}_F \end{bmatrix} = \begin{bmatrix} \mathbf{F}_I & \mathbf{0}_{15\times 6} & \mathbf{0}_{15\times 3m} \\ \mathbf{0}_{6\times 15} & \mathbf{0}_{6\times 6} & \mathbf{0}_{6\times 3m} \\ \mathbf{0}_{3m\times 15} & \mathbf{0}_{3m\times 6} & \mathbf{0}_{3m\times 3m} \end{bmatrix} \tag{2.70}$$

$$\mathbf{G} = \begin{bmatrix} \mathbf{G}_I \\ \mathbf{G}_L \\ \mathbf{G}_F \end{bmatrix} = \begin{bmatrix} \mathbf{G}_I \\ \mathbf{0}_{6\times 12} \\ \mathbf{0}_{3m\times 12} \end{bmatrix} \tag{2.71}$$

2.3 Time Update

Using the models developed in sec. 2.2, the state and covariance can be propagated forward in time from t_k to t_{k+1}. The particular form of the integration of the IMU states and the distinction between ${}^{I}_{I_k}\bar{q}$ and ${}^{I}_{W}\bar{q}$ is based largely on [27] and [41]. [41] uses this form to derive a filter that is more robust to initial attitude uncertainty. [27] takes advantage of this form to find an analytical solution for the state transition matrix. Neither of these extensions are used here but could be incorporated.

Integration of eq. (2.46) yields an expression for ${}^{W}\hat{\mathbf{v}}_{I}(t)$:

$$\begin{aligned} {}^{W}\hat{\mathbf{v}}_{I}(t) = {}^{W}\dot{\hat{\mathbf{p}}}_{I} &= {}^{W}\hat{\mathbf{v}}_{I_k} + {}^{W}\mathbf{g}\Delta t + {}^{I_k}_{W}\mathbf{C}^{\mathrm{T}} \int_{t_k}^{t} {}^{I}_{I_k}\mathbf{C}^{\mathrm{T}}\hat{\mathbf{s}}(\tau)d\tau \\ &= {}^{W}\hat{\mathbf{v}}_{I_k} + {}^{W}\mathbf{g}\Delta t + {}^{I_k}_{W}\mathbf{C}^{\mathrm{T}}\hat{\boldsymbol{\nu}}(t) \end{aligned} \tag{2.72}$$

where $\dot{\hat{\boldsymbol{\nu}}}(t) = {}^{I}_{I_k}\mathbf{C}^{\mathrm{T}}\hat{\mathbf{s}}(t)$ and $\Delta t = t - t_k$. Integration of eq. (2.72) yields an expression for ${}^{W}\hat{\mathbf{p}}_{I}(t)$:

$$\begin{aligned} {}^{W}\hat{\mathbf{p}}_{I}(t) &= {}^{W}\hat{\mathbf{p}}_{I_k} + {}^{W}\hat{\mathbf{v}}_{I_k}\Delta t + {}^{W}\mathbf{g}(\Delta t)^2 + {}^{I_k}_{W}\mathbf{C}^{\mathrm{T}} \int_{t_k}^{t} \hat{\boldsymbol{\nu}}(\tau)d\tau \\ &= {}^{W}\hat{\mathbf{p}}_{I_k} + {}^{W}\hat{\mathbf{v}}_{I_k}\Delta t + {}^{W}\mathbf{g}(\Delta t)^2 + {}^{I_k}_{W}\mathbf{C}^{\mathrm{T}}\hat{\boldsymbol{\rho}}(t) \\ &= {}^{W}\hat{\mathbf{p}}_{I_k} + {}^{W}\hat{\mathbf{p}}^{I}_{I_k} \end{aligned} \tag{2.73}$$

where $\dot{\hat{\boldsymbol{\rho}}}(t) = \hat{\boldsymbol{\nu}}(t)$ and ${}^{W}\hat{\mathbf{p}}^{I}_{I_k} = {}^{W}\hat{\mathbf{v}}_{I_k}\Delta t + {}^{W}\mathbf{g}(\Delta t)^2 + {}^{I_k}_{W}\mathbf{C}^{\mathrm{T}}\hat{\boldsymbol{\rho}}(t)$.

The covariance $\mathbf{P}(t)$ evolves as a function of $\mathbf{P}(t)$, $\mathbf{G}\big(\hat{\mathbf{x}}(t)\big)$, and $\mathbf{F}\big(\hat{\mathbf{x}}(t)\big)$ [42]:

$$\dot{\mathbf{P}}(t) = \mathbf{F}\big(\hat{\mathbf{x}}(t)\big)\mathbf{P}(t) + \mathbf{P}(t)\Big(\mathbf{F}\big(\hat{\mathbf{x}}(t)\big)\Big)^{\mathrm{T}} + \mathbf{G}\big(\hat{\mathbf{x}}(t)\big)\mathbf{Q}(t)\Big(\mathbf{G}\big(\hat{\mathbf{x}}(t)\big)\Big)^{\mathrm{T}} \tag{2.74}$$

$\mathbf{P}(t)$ can be partitioned into submatrices. To conserve space, the notation of each element as a function of time has been dropped.

$$\mathbf{P} = \begin{bmatrix} \mathbf{P}_{II} & \mathbf{P}_{IL} & \mathbf{P}_{IF} \\ \mathbf{P}_{IL}^{\mathrm{T}} & \mathbf{P}_{LL} & \mathbf{P}_{LF} \\ \mathbf{P}_{IF}^{\mathrm{T}} & \mathbf{P}_{LF}^{\mathrm{T}} & \mathbf{P}_{FF} \end{bmatrix} \tag{2.75}$$

Using the sparse structure of $\mathbf{F}(\hat{\mathbf{x}})$ and $\mathbf{G}(\hat{\mathbf{x}})$, eq. (2.74) can be rewritten to improve computational time. To conserve space and improve clarity, the notation of $\mathbf{F}$ and $\mathbf{G}$ as a

function of $\hat{\mathbf{x}}$ has been dropped.

$$\dot{\mathbf{P}} = \begin{bmatrix} \mathbf{F}_I\mathbf{P}_{II} + \mathbf{P}_{II}\mathbf{F}_I^{\mathrm{T}} + \mathbf{G}_I\mathbf{Q}\mathbf{G}_I^{\mathrm{T}} & \mathbf{F}_I\mathbf{P}_{IL} & \mathbf{F}_I\mathbf{P}_{IF} \\ \mathbf{P}_{IL}^{\mathrm{T}}\mathbf{F}_I^{\mathrm{T}} & \mathbf{0}_{6\times 6} & \mathbf{0}_{6\times 3m} \\ \mathbf{P}_{IF}^{\mathrm{T}}\mathbf{F}_I^{\mathrm{T}} & \mathbf{0}_{3m\times 6} & \mathbf{0}_{3m\times 3m} \end{bmatrix} \tag{2.76}$$

In order to solve for $\mathbf{P}(t_{k+1})$ and the remaining elements of $\hat{\mathbf{x}}(t_{k+1})$, it is necessary to numerically solve the following system of first order differential equations for $t_k \leq t \leq t_{k+1}$ using 4th order Runge-Kutta:

$$\begin{aligned} {}^{I}_{I_k}\dot{\hat{q}}(t) \qquad & {}^{I}_{I_k}\hat{q}(t_k) = \begin{bmatrix} 0 & 0 & 0 & 1 \end{bmatrix}^{\mathrm{T}} \\ \dot{\hat{\boldsymbol{\nu}}}(t) \qquad & \hat{\boldsymbol{\nu}}(t_k) = \mathbf{0} \\ \dot{\hat{\boldsymbol{\rho}}}(t) \qquad & \hat{\boldsymbol{\rho}}(t_k) = \mathbf{0} \\ \dot{\mathbf{P}}(t) \qquad & \mathbf{P}(t_k) = \mathbf{P}_k \end{aligned} \tag{2.77}$$

Now ${}^{I}_{W}\hat{q}(t_{k+1})$ can be solved using the relationship ${}^{I}_{W}\hat{q}(t_{k+1}) = {}^{I}_{I_k}\hat{q}(t_{k+1}){}^{I}_{W}\hat{q}(t_k)$. Similarly, ${}^{W}\hat{\mathbf{v}}_I(t_{k+1})$ and ${}^{W}\hat{\mathbf{p}}_I(t_{k+1})$ can be found by using $\hat{\boldsymbol{\nu}}(t_{k+1})$ and $\hat{\boldsymbol{\rho}}(t_{k+1})$ in eqs. (2.72) and (2.73), respectively.

2.4 Feature Models

The monocular camera provides images at some rate. Each time an image is received, unique features are identified in the scene. As previously discussed, only the bearing of a feature is observable in a single camera image. In this work, only initialized features are processed by the EKF camera update. Initialized features have an associated global position estimate. Uninitialized features are processed separately until they are initialized. This is known as delayed initialization. Feature matching and initialization are described in detail in ch. 3.

This section is concerned with using observations of initialized features to update the

filter state. The feature matcher will provide a measurement vector

$$\tilde{\mathbf{y}}_F = \begin{bmatrix} \tilde{\mathbf{y}}_{F_1} \\ \vdots \\ \tilde{\mathbf{y}}_{F_i} \\ \vdots \\ \tilde{\mathbf{y}}_{F_m} \end{bmatrix} \tag{2.78}$$

$$\tilde{\mathbf{y}}_{F_i} = \begin{bmatrix} \tilde{u}_i \\ \tilde{v}_i \end{bmatrix} \tag{2.79}$$

where $\tilde{u}_i$ and $\tilde{v}_i$ are the pixel coordinates of feature F_i. F_i is located at some position ${}^{W}\mathbf{p}_{F_i}$.

2.4.1 Measurement Model

The pinhole projection model relates the position of a feature in the camera frame to its projection on the image plane (see fig. 1.1):

$$\tilde{\mathbf{y}}_{F_i} = \begin{bmatrix} \tilde{u}_i \\ \tilde{v}_i \end{bmatrix} = \begin{bmatrix} f_x(h_x/h_z) + c_x \\ f_y(h_y/h_z) + c_y \end{bmatrix} + \mathbf{v}_{F_i} \tag{2.80}$$

where $\mathbf{v}_{F_i}$ is a white noise process with variance $\mathbf{R}_{F_i}$ and

$$\begin{bmatrix} h_x & h_y & h_z \end{bmatrix}^{\mathrm{T}} = {}^{C}\mathbf{p}_{F_i} = {}^{C}_{I}\mathbf{C}\left({}^{I}_{W}\mathbf{C}({}^{W}\mathbf{p}_{F_i} - {}^{W}\mathbf{p}_{I}) - {}^{I}\mathbf{p}_{C}\right) \tag{2.81}$$

where ${}^{C}_{I}\mathbf{C}$ represents the attitude of the IMU with respect to the camera and ${}^{I}\mathbf{p}_{C}$ is the position of the camera in the IMU frame (see fig. 2.3). These are assumed to be known without uncertainty.

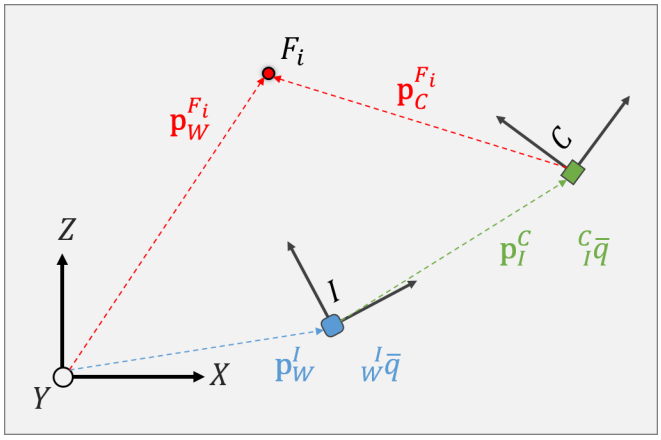

Figure 2.3: The position of the feature in the camera frame can be constructed from the global pose of the camera and the global position of the feature.

2.4.2 Measurement Estimate

The estimated measurement $\hat{\mathbf{y}}_{F_i}$ is found by using the state estimates in eqs. (2.81) and (2.80)

$$\begin{bmatrix}\hat{h}_x & \hat{h}_y & \hat{h}_z\end{bmatrix}^{\mathrm{T}} = {}^C\hat{\mathbf{p}}_{F_i} = {}_I^C\mathbf{C}\left({}_W^I\hat{\mathbf{C}}({}^W\hat{\mathbf{p}}_{F_i} - {}^W\hat{\mathbf{p}}_I) - {}^I\mathbf{p}_C\right) \tag{2.82}$$

$$\hat{\mathbf{y}}_{F_i} = \begin{bmatrix} f_x(\hat{h}_x/\hat{h}_z) + c_x \\ f_y(\hat{h}_y/\hat{h}_z) + c_y \end{bmatrix} \tag{2.83}$$

2.4.3 Measurement Error Model

It is necessary to obtain a model for the measurement error $\breve{\mathbf{y}}_{F_i}$ in the form of eq. (2.20)

.

$$\breve{\mathbf{y}}_{F_i} \approx \mathbf{H}_{F_i}\breve{\mathbf{x}} + \mathbf{v}_{F_i} \tag{2.84}$$

$$\mathbf{H}_{F_i} = \frac{\delta\tilde{\mathbf{y}}_{F_i}}{\delta^C\mathbf{p}_{F_i}}\begin{bmatrix}\frac{\delta^C\mathbf{p}_{F_i}}{\delta\breve{\mathbf{x}}_I} & \mathbf{0}_{2\times 6} & \frac{\delta^C\mathbf{p}_{F_i}}{\delta\breve{\mathbf{x}}_F}\end{bmatrix} \tag{2.85}$$

$$\frac{\delta\tilde{\mathbf{y}}_{F_i}}{\delta^C\mathbf{p}_{F_i}} = \frac{1}{\hat{h}_z}\begin{bmatrix} f_x & 0 & -f_x\hat{h}_x/\hat{h}_z \\ 0 & f_y & -f_y\hat{h}_y/\hat{h}_z \end{bmatrix} \tag{2.86}$$

$$\frac{\delta^{C}\mathbf{p}_{F_i}}{\delta\breve{\mathbf{x}}_I} = {}^{C}_{I}\mathbf{C}\begin{bmatrix}\lfloor\left({}^{I}_{W}\hat{\mathbf{C}}({}^{C}\hat{\mathbf{p}}_{F_i} - {}^{W}\hat{\mathbf{p}}_I)\right)\times\rfloor & -{}^{I}_{W}\hat{\mathbf{C}} & \mathbf{0}_{3\times3} & \mathbf{0}_{3\times3} & \mathbf{0}_{3\times3}\end{bmatrix} \tag{2.87}$$

$\frac{\delta^{C}\mathbf{p}_{F_i}}{\delta\breve{\mathbf{x}}_F}$ is a $1 \times m$ block matrix. The i^{th} block column is ${}^{I}_{W}\hat{\mathbf{C}}$. All other block columns contain $\mathbf{0}_{3\times3}$.

The total measurement Jacobian, $\mathbf{H}_F$, is a block column matrix composed of matrices $\mathbf{H}_{F_1} \ldots \mathbf{H}_{F_i} \ldots \mathbf{H}_{F_m}$:

$$\mathbf{H}_F = \begin{bmatrix}\mathbf{H}_{F_1}\\ \vdots \\ \mathbf{H}_{F_i}\\ \vdots \\ \mathbf{H}_{F_m}\end{bmatrix} \tag{2.88}$$

2.5 Rangefinder Models

The downward facing rangefinder provides a scalar distance measurement. Assuming a flat floor, the returned measurement $\tilde{y}_R$ is the distance from the rangefinder emitter to the ground along the axis of the rangefinder.

2.5.1 Measurement Model

The rangefinder measurement axis is aligned with the z-axis of a frame R. In the global frame, the attitude of R is expressed as the product of the rotation of the IMU and the rotation of the rangefinder:

$${}^{R}_{W}\mathbf{R} = {}^{R}_{W}\mathbf{C}^{\mathrm{T}} = ({}^{R}_{I}\mathbf{C}\,{}^{I}_{W}\mathbf{C})^{\mathrm{T}} = {}^{I}_{W}\mathbf{R}\,{}^{R}_{I}\mathbf{R} \tag{2.89}$$

The position of the rangefinder can be expressed in the world frame as

$${}^{W}\mathbf{p}_R = {}^{W}\mathbf{p}_I + {}^{I}_{W}\mathbf{R}\,{}^{I}\mathbf{p}_R \tag{2.90}$$

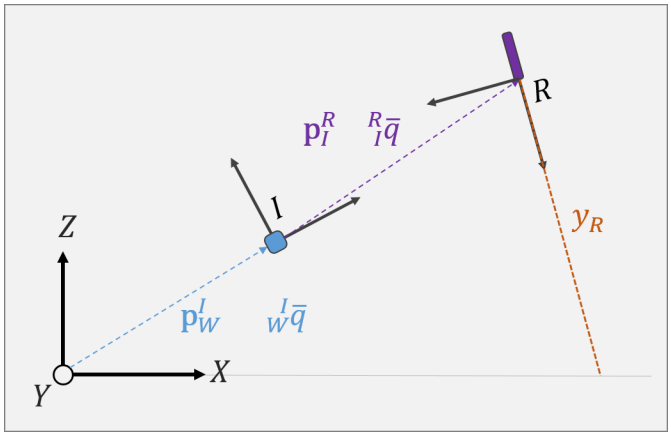

Figure 2.4: The distance of the rangefinder to the ground along the rangefinder z-axis can be constructed from the global pose of the rangefinder.

where ${}^I\mathbf{p}_R$ is the position of the rangefinder emitter in the IMU frame (see fig. 2.4). ${}_I^R\mathbf{R}$ and ${}^I\mathbf{p}_R$ are assumed to be known without uncertainty.

Using these definitions, the measurement equation for $\tilde{y}_R$ can be expressed as

$$\tilde{y}_R = -\frac{1}{{}_W^R R_{33}} {}^W p_{R_{31}} + v_R = -\frac{1}{{}_W^R R_{33}} ({}^W p_{I_{31}} + {}_W^I\mathbf{R}_{3j} {}^I\mathbf{p}_R) + v_R \tag{2.91}$$

where v_R is a white noise process with variance r_R. ${}_W^R R_{33}$ is the element at the third column and third row of ${}_W^R\mathbf{R}$. ${}^W p_{I_{31}}$ is the third element of ${}^W\mathbf{p}_I$. ${}_W^I\mathbf{R}_{3j}$ is the third row of ${}_W^I\mathbf{R}$.

2.5.2 Measurement Estimate

From eq. (2.91) , the estimated measurement is

$${}^W\hat{p}_{R_{31}} = {}^W\hat{p}_{I_{31}} + {}_W^I\hat{\mathbf{R}}_{3j} {}^I\mathbf{p}_R \tag{2.92}$$

$$\hat{y}_R = -\frac{1}{{}_W^R\hat{R}_{33}} {}^W\hat{p}_{R_{31}} \tag{2.93}$$

2.5.3 Measurement Error Model

It is necessary to obtain a model for the measurement error $\breve{y}_R$ in the form of eq. (2.20)

.

$$\breve{y}_R \approx \mathbf{H}_R \breve{\mathbf{x}} + v_R \tag{2.94}$$

$$\mathbf{H}_R = \begin{bmatrix} \frac{\delta \tilde{y}_R}{\delta(\delta \boldsymbol{\theta}_I)} & \mathbf{0}_{1\times 2} & \frac{\delta \tilde{y}_R}{\delta(\Delta p_{I_{31}})} & \mathbf{0}_{1\times 9} & \mathbf{0}_{1\times 3m} \end{bmatrix} \tag{2.95}$$

$$\frac{\delta \tilde{y}_R}{\delta(\delta \boldsymbol{\theta}_I)} = \frac{1}{({}^R_W\hat{R}_{33})^2} {}^W\hat{p}_{R_{31}} (-{}^I_W\hat{\mathbf{R}}_{3j} \lfloor {}^R_I\mathbf{R}_{i3} \times \rfloor) - \frac{1}{{}^R_W\hat{R}_{33}} {}^I\mathbf{p}_R{}^{\mathrm{T}} \lfloor {}^I_W\hat{\mathbf{R}}_{3j} \times \rfloor \tag{2.96}$$

$$\frac{\delta \tilde{y}_R}{\delta(\Delta p_{I_{31}})} = -\frac{1}{{}^R_W\hat{R}_{33}} \tag{2.97}$$

where ${}^R_I\mathbf{R}_{i3}$ is the third column of ${}^R_I\mathbf{R}$.

2.6 Measurement Update

Using the models in secs. 2.4 and 2.5, the state can be updated according to the normal EKF procedure. The prior state and covariance, $\hat{\mathbf{x}}^-$ and $\mathbf{P}^-$, respectively, are the state and covariance found using the propagated process model. The values for $\mathbf{H}_k$, $\mathbf{R}_k$, $\tilde{\mathbf{y}}_k$, and $\hat{\mathbf{y}}_k$ should be selected based on whether the state is to be updated using a feature or rangefinder measurement.

$$\mathbf{K}_k = \mathbf{P}_k^- \mathbf{H}_k^{\mathrm{T}} (\mathbf{H}_k \mathbf{P}_k^- \mathbf{H}_k^{\mathrm{T}} + \mathbf{R}_k)^{-1} \tag{2.98}$$

$$\mathbf{P}_k^+ = (\mathbf{I} - \mathbf{K_k} \mathbf{H}_k^{\mathrm{T}}) \mathbf{P}_k^- \tag{2.99}$$

$$\hat{\check{\mathbf{x}}}_k^+ = \mathbf{K}_k (\tilde{\mathbf{y}}_k - \hat{\mathbf{y}}_k) \tag{2.100}$$

$\hat{\check{\mathbf{x}}}_k^+$ is the estimated state error that must be "applied" to update the state. The additive error states are updated using the additive error definition. For example, the estimate ${}^W\hat{\mathbf{p}}_I$

is updated using

$$ {}^{W}\hat{\mathbf{p}}_I{}^{+} = {}^{W}\hat{\mathbf{p}}_I{}^{-} + \Delta\mathbf{p}_I^{+} \tag{2.101} $$

The attitude states are updated using the small quaternion error approximation

$$ \bar{q}^{+} = \begin{bmatrix} \frac{1}{2}\delta\boldsymbol{\theta}^{+} \\ 1 \end{bmatrix} \otimes \bar{q}^{-} \tag{2.102} $$

The quaternions are re-normalized after this update.

2.7 Discussion

There are several ways in which this filter could be extended to improve performance. It is well known that the global pose of monocular SLAM is not fully observable [27] [43] [44] [45]. The global position and yaw states are expected to become inconsistent over time. There have been many techniques proposed to improve the consistency of filter-based SLAM. Civera showed improvements using a robo-centric/camera-centric state [45]. The authors of [43] offer an "observability-constrained" measurement Jacobian that allows the linearized filter to have the same observability properties as the full non-linear system. [27] shows that using an analytical state transition matrix and only using propagated positions and velocities in the measurement Jacobians also results in a more consistent filter. One of these techniques could be applied to improve the consistency of this filter.

The current implementation assumes that the inter-sensor parameters and camera calibration parameters are known without uncertainty. In many cases this is not a valid assumption. Many researchers have seen improvements in the consistency of monocular SLAM by including such parameters in the state vector [46] [27]. Another option is to use least squares or another batch optimization technique to estimate the parameters and associated covariance beforehand. This uncertainty could be accounted for using the consider

EKF [39].

The EKF is not guaranteed to converge, particularly in the presence of large errors. For this reason, the filter may not converge if there are significant initial attitude errors. The initial pitch and roll angles of the vehicle can be accurately estimated by sampling a few accelerometer measurements when the system is at rest. When the system has no acceleration, the accelerometer measurements provide the direction of the gravity vector if the magnitude of the acceleration due to gravity is well-known.

3. FEATURE TRACKING AND DELAYED INITIALIZATION

The models described in ch. 2 assume that features are extracted from monocular camera images and do not explain the use of the keyframe state. These models also assume that each observed feature has an associated global position estimate. This chapter describes how features are observed and initialized. It also explains the usage of the keyframe state.

3.1 Feature Detection and Matching

Each time a new image is received, ORB [31] keypoints and descriptors are detected and computed, respectively, using the OpenCV ORB classes. The descriptors uniquely "describe" the feature. These new descriptors are matched with the descriptors of initialized features. If certain criteria are met (see 3.2), the image is declared as the current keyframe image. Otherwise, the new descriptors are matched with keyframe descriptors.

3.1.1 Matching New Features

The feature tracking module matches new image descriptors with keyframe image descriptors. First, the OpenCV "brute force" search algorithm tries to match each keyframe descriptor with two of the new image descriptors. The fast approximate nearest neighbor (FLANN) search algorithm [47] was also considered. However, FLANN is not deterministic and the resulting filter performance varied widely from run to run even with the same data. When adding new features, it is advantageous to choose features that cover as much of the image as possible and do not overlap. For this reason, matches are only made to features that are outside of some radius to observations of any mapped features. Some keyframe descriptors have multiple likely matches. These ambiguous descriptors cannot be matched reliably. The ratio test [48] is applied to remove matches made between an ambiguous keyframe descriptor and new image descriptors. This eliminates many of the

matching outliers. If at least 8 matches pass the ratio test, the remaining matches are used to find a fundamental matrix by way of the eight-point algorithm [49]. Finally, the keyframe and new keypoints are reprojected using this fundamental matrix. The variance of the differences between the measured and ideal keypoint locations is calculated and used as the measurement variance.

The matches are further pruned such that no two features are within a certain radius of one another. The remaining matches are passed on to the feature initialization module.

3.1.2 Matching Mapped Features

The "brute force" matcher is used to match each mapped feature descriptor with a keyframe image descriptor. Because an estimate of the global position of each feature exists, the location of the feature in the image can be predicted using eqs. 2.82 and 2.83. If the mapped feature is matched to a location that is within some radius of the predicted location, the match is accepted. This greatly eliminates faulty matches in environments with repeated texture. Observations of mapped features are used to update the estimates in the manner described in ch. 2.

3.2 Changing the Keyframe State

The keyframe image is replaced if the average number of new feature matches in the last w images is less than some value. The number of feature matches will decline over time because the camera will move away from the scene described by the keyframe image. In addition, as previously described, only matches that are "far" from the observations of mapped features are accepted. Eventually it may not be possible to match any features that are "far enough" from mapped feature observations. In these cases, a new keyframe image is necessary.

When a new keyframe is declared, the keyframe state is cloned from the current state in a manner similar to [50]. This is an application of the stochastic cloning technique

[51]. However, unlike [50], in this work only one keyframe state is kept in the state vector. The declaration of a new keyframe is declaring that the current IMU pose should replace the existing keyframe pose. It is an entirely different keyframe; it is not "updating" the existing keyframe. Rather the existing keyframe is deleted from the state and the newly declared keyframe is added to the state vector.

When a new keyframe is declared, the keyframe state estimate $\hat{\mathbf{x}}_L(t_k)$ at time t_k takes on the value of the current IMU pose:

$$\hat{\mathbf{x}}_L(t_k) = \begin{bmatrix} {}^I_W\hat{\bar{q}}(t_k)^{\mathrm{T}} & {}^W\hat{\mathbf{p}}_I(t_k)^{\mathrm{T}} \end{bmatrix}^{\mathrm{T}} \tag{3.1}$$

The error state equation can be partitioned into 4 subvectors:

$$\breve{\mathbf{x}} = \begin{bmatrix} \breve{\mathbf{x}}_{I_P}^{\mathrm{T}} & \breve{\mathbf{x}}_{I_O}^{\mathrm{T}} & \breve{\mathbf{x}}_L^{\mathrm{T}} & \breve{\mathbf{x}}_F^{T} \end{bmatrix}^T \tag{3.2}$$

$\breve{\mathbf{x}}_{I_P}$ contains the IMU pose states.

$$\breve{\mathbf{x}}_{I_P} = \begin{bmatrix} \delta\boldsymbol{\theta}_I^{\mathrm{T}} & \Delta\mathbf{p}_I^{\mathrm{T}} \end{bmatrix}^{\mathrm{T}} \tag{3.3}$$

$\breve{\mathbf{x}}_{I_O}$ contains the other IMU states.

$$\breve{\mathbf{x}}_{I_O} = \begin{bmatrix} \Delta\mathbf{v}_I^{\mathrm{T}} & \Delta\mathbf{b}_g^{\mathrm{T}} & \Delta\mathbf{b}_a^{\mathrm{T}} \end{bmatrix}^{\mathrm{T}} \tag{3.4}$$

When the existing keyframe is deleted, the keyframe covariance and all cross-correlation terms are removed such that the covariance can be partitioned into the following block

columns and rows:

$$\mathbf{P}(t_k) = \begin{bmatrix} \mathbf{P}_{I_P I_P}(t_k) & \mathbf{P}_{I_P I_O}(t_k) & \mathbf{P}_{I_P F}(t_k) \\ \mathbf{P}^{\mathrm{T}}_{I_P I_O}(t_k) & \mathbf{P}_{I_O I_O}(t_k) & \mathbf{P}_{I_O F}(t_k) \\ \mathbf{P}^{\mathrm{T}}_{I_P F}(t_k) & \mathbf{P}^{\mathrm{T}}_{I_O F}(t_k) & \mathbf{P}_{FF}(t_k) \end{bmatrix} \tag{3.5}$$

When a new keyframe is declared, the covariance and cross-correlation terms associated with the IMU pose are cloned and augmented into the covariance matrix:

$$\mathbf{P}(t_k) = \begin{bmatrix} \mathbf{P}_{I_P I_P}(t_k) & \mathbf{P}_{I_P I_O}(t_k) & \mathbf{P}_{I_P I_P}(t_k) & \mathbf{P}_{I_P F}(t_k) \\ \mathbf{P}^{\mathrm{T}}_{I_P I_O}(t_k) & \mathbf{P}_{I_O I_O}(t_k) & \mathbf{P}^{\mathrm{T}}_{I_P I_O}(t_k) & \mathbf{P}_{I_O F}(t_k) \\ \mathbf{P}_{I_P I_P}(t_k) & \mathbf{P}_{I_P I_O}(t_k) & \mathbf{P}_{I_P I_P}(t_k) & \mathbf{P}_{I_P F}(t_k) \\ \mathbf{P}^{\mathrm{T}}_{I_P F}(t_k) & \mathbf{P}^{\mathrm{T}}_{I_O F}(t_k) & \mathbf{P}^{\mathrm{T}}_{I_P F}(t_k) & \mathbf{P}_{FF}(t_k) \end{bmatrix} \tag{3.6}$$

3.3 Feature Initialization

Many techniques have been suggested in the literature for feature initialization. In filter-based SLAM, the techniques can be broadly divided into two classes: delayed and undelayed initialization. In undelayed initialization, a feature is immediately added to the map upon first observation. Because no depth information is available from the single observation, the feature covariance must be large enough to include the true (but unknown) position. Features using a Euclidean parameterization in undelayed initialization are unlikely to converge. The authors of [25] showed that a 6 parameter inverse depth parameterization could successfully be used for undelayed initialization. Other parameterizations, including 4 and 7 parameter sets, have also been studied [52].

A problem with undelayed initialization techniques is that the features have to be initialized with some prior covariance. Usually each feature is initialized with the same covariance and is assumed to be uncorrelated with the existing filter states. Neither of these assumptions are true in general and can contribute to filter inconsistency.

Other work has focused on delayed initialization techniques. In delayed initialization,

the feature position is not added to the EKF state until the feature position and covariance have been estimated using a separate process. Undelayed initialization methods vary widely, but most record observations of a feature while the camera moves throughout a scene. The feature position can be recursively updated (e.g. with a particle filter [53]), in a single batch update (e.g [28]), or even batch EKF updates (e.g. [50]).

In practice, these techniques have the disadvantage of having to track and store features over a potentially long period of time. This increases the amount of computer memory required. If the camera is still, numerous redundant measurements will be acquired. In addition, many features that are observed once may never "move" enough to be intialized (e.g. points that are very far from the camera).

In this work, a delayed feature initialization technique is applied. Every time a new image is received, the feature matching module finds new feature matches. Each of these matches is a candidate for initialization. Features are initialized if their global position can be determined with a confidence that exceeds a specified threshold. If the candidate is rejected, the new measurement is discarded.

3.3.1 Inverse Depth Feature Position Estimate

Each candidate is passed to the feature initialization module. The first step is finding an estimate for the position of the feature in the world frame. Nonlinear least squares is used to find the position ${}^{W}\hat{\mathbf{p}}_{F_n}$ of feature F_n.

For initialization, an inverse depth parameterization is used rather than the Euclidean position for the iterative solver. The inverse depth parameterization is more linear and has better numerical properties for small baselines. The feature initialization procedure presented here closely follows [27]. In this work, however, only two camera poses are used.

Consider a feature F_n first observed when the camera had pose C_1 (see fig. 3.1). In

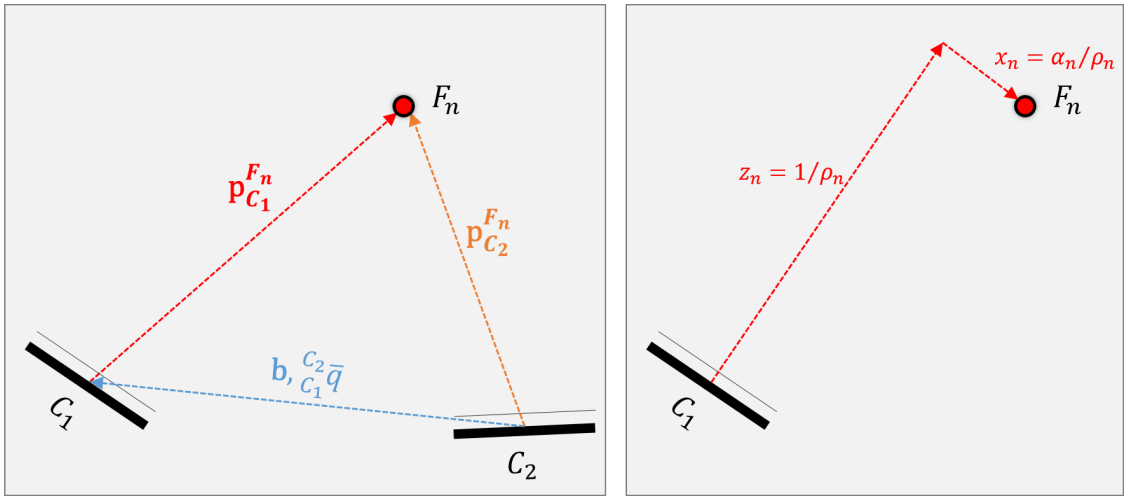

Figure 3.1: 2D diagram of feature initialization geometry.

this inverse depth parameterization, ${}^{C_1}\mathbf{p}_{F_n}$ is re-parameterized

$$\begin{bmatrix} h_{1,x} \\ h_{1,y} \\ h_{1,z} \end{bmatrix} = {}^{C_1}\mathbf{p}_{F_n} = \begin{bmatrix} x_n \\ y_n \\ z_n \end{bmatrix} = z_n \begin{bmatrix} x_n/z_n \\ y_n/z_n \\ 1 \end{bmatrix} = 1/\rho_n \begin{bmatrix} \alpha_n \\ \beta_n \\ 1 \end{bmatrix} \tag{3.7}$$

$$\rho_n = 1/z_n \qquad \alpha_n = x_n/z_n \qquad \beta_n = y_n/z_n \tag{3.8}$$

Some time later the camera has pose C_2. The baseline vector ${}^{C_2}\mathbf{b}$ points from the origin of the C_2 frame to the origin of the C_1 frame (see fig. 3.1). The vector ${}^{C_2}\mathbf{p}_{F_n}$ points from the origin of C_2 to F_n and is the concatenation of the baseline vector with the vector pointing from C_1 to F_n.

$$\begin{bmatrix} h_{2,x} \\ h_{2,y} \\ h_{2,z} \end{bmatrix} = {}^{C_2}\mathbf{p}_{F_n} = {}^{C_2}\mathbf{b} + ({}^{C_2}_{C_1}\mathbf{C})\,{}^{C_1}\mathbf{p}_{F_n} \tag{3.9}$$

If C_1 is the keyframe pose of the camera and C_2 is the current pose of the camera, the

baseline vector can be determined:

$$
\begin{aligned}
{}^{C_2}\mathbf{b} &= {}^{C_2}_{W}\mathbf{C}({}^{W}\mathbf{p}_{C_1} - {}^{W}\mathbf{p}_{C_2}) \\
&= {}^{C_2}_{W}\mathbf{C}\big(({}^{W}\mathbf{p}_{I_L} + {}^{I_L}_{W}\mathbf{C}^{\mathrm{T}}\,{}^{I}\mathbf{p}_{C}) - ({}^{W}\mathbf{p}_{I} + {}^{I}_{W}\mathbf{C}^{\mathrm{T}}\,{}^{I}\mathbf{p}_{C})\big) \\
&= {}^{C}_{I}\mathbf{C}\,{}^{I}_{W}\mathbf{C}\big({}^{W}\mathbf{p}_{I_L} - {}^{W}\mathbf{p}_{I} + ({}^{I_L}_{W}\mathbf{C} - {}^{I}_{W}\mathbf{C})^{\mathrm{T}}\,{}^{I}\mathbf{p}_{C}\big)
\end{aligned} \tag{3.10}
$$

Similarly, the rotation between C_1 and C_2 can be written as

$$
{}^{C_2}_{C_1}\mathbf{C} = ({}^{C}_{I}\mathbf{C}\,{}^{I}_{W}\mathbf{C})({}^{C}_{I}\mathbf{C}\,{}^{I_L}_{W}\mathbf{C})^{\mathrm{T}} \tag{3.11}
$$

The candidate match contains pixel coordinates of the feature when the camera was at poses C_1 and C_2. These can be stacked into $\tilde{\mathbf{y}}_{F_n}$

$$
\tilde{\mathbf{y}}_{F_n} = \begin{bmatrix} \tilde{u}_{1,n} & \tilde{v}_{1,n} & \tilde{u}_{2,n} & \tilde{v}_{2,n} \end{bmatrix}^{\mathrm{T}} \tag{3.12}
$$

where

$$
\begin{bmatrix} \tilde{u}_{j,n} \\ \tilde{v}_{j,n} \end{bmatrix} = \begin{bmatrix} f_x(h_{j,x}/h_{j,z}) + c_x \\ f_y(h_{j,y}/h_{j,z}) + c_y \end{bmatrix} + \mathbf{v}_{F_n} \quad j \in 1,2 \tag{3.13}
$$

The Gauss-Newton algorithm [42] can be used to iteratively determine the estimates of α_n, β_n, and ρ_n that best satisfy the measurement model. This algorithm requires the Jacobian $\mathbf{J}_{F_n}$ of eq. (3.12) with respect to $\bar{\mathbf{x}} = \begin{bmatrix} \alpha_n & \beta_n & \rho_n \end{bmatrix}^{\mathrm{T}}$.

$$
\mathbf{J}_{F_n} = \begin{bmatrix} \dfrac{\delta\tilde{\mathbf{y}}_{F_n}}{\delta^{C_1}\mathbf{p}_{F_n}} \dfrac{\delta^{C_1}\mathbf{p}_{F_n}}{\delta\bar{\mathbf{x}}} \\ \dfrac{\delta\tilde{\mathbf{y}}_{F_n}}{\delta^{C_2}\mathbf{p}_{F_n}} \dfrac{\delta^{C_2}\mathbf{p}_{F_n}}{\delta\bar{\mathbf{x}}} \end{bmatrix} \tag{3.14}
$$

$$
\frac{\delta\tilde{\mathbf{y}}_{F_n}}{\delta^{C_j}\mathbf{p}_{F_n}} = \frac{1}{h_{j,z}} \begin{bmatrix} f_x & 0 & -f_x h_{j,x}/h_{j,z} \\ 0 & f_y & -f_y h_{j,y}/h_{j,z} \end{bmatrix} \quad j \in 1,2 \tag{3.15}
$$

$$\frac{\delta^{C_j}\mathbf{p}_{F_n}}{\delta\bar{\mathbf{x}}} = \frac{1}{\rho_n}\begin{bmatrix}\mathbf{C}_{j_{i1}} & \mathbf{C}_{j_{i2}} & -\frac{1}{\rho_n}\mathbf{C}_j\begin{bmatrix}\alpha_n\\ \beta_n\\ 1\end{bmatrix}\end{bmatrix} \quad j \in 1,2 \tag{3.16}$$

$$\mathbf{C}_1 = \mathbf{I}_{3\times 3} \tag{3.17}$$

$$\mathbf{C}_2 = {}^{C_2}_{C_1}\mathbf{C} \tag{3.18}$$

$\mathbf{C}_{j_{i1}}$ and $\mathbf{C}_{j_{i2}}$ are the first and second columns, respectively, of matrix $\mathbf{C}_j$.

After several iterations, the nonlinear least squares algorithm converges to estimates $\hat{\alpha}_n$, $\hat{\beta}_n$, and $\hat{\rho}_n$. Using $\hat{\alpha}_n$, $\hat{\beta}_n$, and $\hat{\rho}_n$, the position of the feature in Euclidean coordinates can be constructed using eq. (3.7) and

$${}^{W}\hat{\mathbf{p}}_{F_n} = {}^{W}\hat{\mathbf{p}}_{I_L} + ({}^{I_L}_{W}\hat{\mathbf{C}}^{\mathrm{T}})^{C_1}\mathbf{p}_{F_n} \tag{3.19}$$

3.3.2 Covariance Estimate

In addition to the state estimate, the new feature state must have an associated covariance. Rather than assume a prior covariance, the initial covariance will be approximated using the linear consider least squares approach [39]. This unique approach to feature initialization considers the uncertainty of the keyframe and current IMU poses in addition to the measurement uncertainty. This is important because the uncertainty due to measurement noise will usually be small and approximately constant. In reality, feature initialization accuracy and uncertainty is heavily dependent on accurate knowledge of the poses from which the images were taken. The technique presented here takes both uncertainties into account. If only the measurement uncertainty was considered, the feature initialization criteria described in sec. 3.3.3 would not be robust to pose uncertainty. This technique also immediately provides cross-correlation terms with all of the EKF states.

First, the measurement error $\breve{\mathbf{y}}_{F_n}$ is approximated as

$$\breve{\mathbf{y}}_{F_n} \approx \mathbf{H}_{F_n} \Delta\mathbf{p}_{F_n} + \mathbf{H}_X \breve{\mathbf{x}} + \mathbf{v}_{F_n} \tag{3.20}$$

$$\mathbf{H}_{F_n} = \begin{bmatrix} \dfrac{\delta\tilde{\mathbf{y}}_{F_n}}{\delta^{C_1}\mathbf{p}_{F_n}} {}^{C}_{I}\mathbf{C}\,{}^{I_L}_{W}\mathbf{C} \\ \dfrac{\delta\tilde{\mathbf{y}}_{F_n}}{\delta^{C_2}\mathbf{p}_{F_n}} {}^{C}_{I}\mathbf{C}\,{}^{I}_{W}\mathbf{C} \end{bmatrix} \tag{3.21}$$

$$\mathbf{H}_X = \begin{bmatrix} \mathbf{H}_{X_1} \\ \mathbf{H}_{X_2} \end{bmatrix} \tag{3.22}$$

$$\mathbf{H}_{X_1} = {}^{C}_{I}\mathbf{C} \begin{bmatrix} \mathbf{0}_{2\times 15} & \lfloor \left({}^{I_L}_{W}\mathbf{C}({}^{W}\mathbf{p}_{F_n} - {}^{W}\mathbf{p}_{I_K})\right) \times \rfloor & -{}^{I_L}_{W}\mathbf{C} & \mathbf{0}_{2\times 3m} \end{bmatrix} \tag{3.23}$$

$$\mathbf{H}_{X_2} = {}^{C}_{I}\mathbf{C} \begin{bmatrix} \lfloor \left({}^{I}_{W}\mathbf{C}({}^{W}\mathbf{p}_{F_n} - {}^{W}\mathbf{p}_{I})\right) \times \rfloor & -{}^{I}_{W}\mathbf{C} & \mathbf{0}_{2\times 15} & \mathbf{0}_{2\times 3m} \end{bmatrix} \tag{3.24}$$

$\mathbf{H}_{F_n}$ and $\mathbf{H}_X$ are matrices that indicate how sensitive the measurement errors are to errors in the feature position and EKF state, respectively. Using these matrices and the state estimate covariance $\mathbf{P}$, the covariance of the errors in ${}^{W}\hat{\mathbf{p}}_{F_n}$ can be estimated using consider analysis.

$$\mathbf{P}_{F_nF_n} = (\mathbf{H}_{F_n}^{\mathrm{T}} {\mathbf{R}_{F_n}}^{-1} \mathbf{H}_{F_n} - \mathbf{H}_{F_n}^{\mathrm{T}} {\mathbf{R}_{F_n}}^{-1} \mathbf{H}_X (\mathbf{H}_X^{\mathrm{T}} {\mathbf{R}_{F_n}}^{-1} \mathbf{H}_X + \mathbf{P}^{-1})^{-1} \mathbf{H}_X^{\mathrm{T}} {\mathbf{R}_{F_n}}^{-1} \mathbf{H}_{F_n})^{-1} \tag{3.25}$$

$$\mathbf{P}_{F_nX} = -\mathbf{P}_{F_nF_n} \mathbf{H}_{F_n}^{\mathrm{T}} {\mathbf{R}_{F_n}}^{-1} \mathbf{H}_X (\mathbf{H}_X^{\mathrm{T}} {\mathbf{R}_{F_n}}^{-1} \mathbf{H}_X)^{-1} \tag{3.26}$$

$\mathbf{P}_{F_nF_n}$ is the covariance of ${}^{W}\hat{\mathbf{p}}_{F_n}$. $\mathbf{P}_{F_nX}$ contains the cross-correlation terms with the EKF error state.

This technique has an important disadvantage of requiring the inverse of two potentially large matrices: $\mathbf{P}^{-1}$ and $(\mathbf{H}_X^{\mathrm{T}} {\mathbf{R}_{F_n}}^{-1} \mathbf{H}_X + \mathbf{P}^{-1})^{-1}$. There are several ways that this computational burden could be reduced. One is to only use the consider update if $\mathbf{P}$ is "large". For example, the trace or determinant of $\mathbf{P}$ could be used as a metric for the total

uncertainty of $\mathbf{P}$. If the uncertainty is small, the traditional linear least squares covariance that only considers measurement uncertainty would be used.

A second option would be to use the following approximation for $\mathbf{P}^{-1} \gg \mathbf{H}_X^{\mathrm{T}} {\mathbf{R}_{F_n}}^{-1} \mathbf{H}_X$:

$$(\mathbf{H}_X^{\mathrm{T}} {\mathbf{R}_{F_n}}^{-1} \mathbf{H}_X + \mathbf{P}^{-1})^{-1} \approx (\mathbf{P}^{-1})^{-1} = \mathbf{P} \tag{3.27}$$

A third option is to only consider a portion of $\mathbf{P}$. For example, only the portion of $\mathbf{P}$ corresponding to the pose states could be considered. In this case, the feature would be initialized without correlations to the other states.

3.3.3 Adding a Feature to the Map

In general the most uncertain direction of the feature position lies along the line of sight of the feature to the camera. The variance of this direction dominates compared with any orthogonal directions. Thus it is expected that the largest eigenvalue of $\mathbf{P}_{F_n F_n}$ will correspond to the depth direction. This eigenvalue, denoted as σ_d^2, is used as a threshold for feature initialization. Only features with a σ_d^2 less than some value are added to the map.

To add a feature to the state, it is simply appended to the state vector:

$$\hat{\mathbf{x}} = \begin{bmatrix} \hat{\mathbf{x}} \\ {}^{W}\hat{\mathbf{p}}_{F_n} \end{bmatrix} \tag{3.28}$$

The covariance is augmented using $\mathbf{P}_{F_n F_n}$ and $\mathbf{P}_{F_n X}$:

$$\mathbf{P} = \begin{bmatrix} \mathbf{P} & \mathbf{P}_{F_n X}^{\mathrm{T}} \\ \mathbf{P}_{F_n X} & \mathbf{P}_{F_n F_n} \end{bmatrix} \tag{3.29}$$

4. PERFORMANCE ON EXPERIMENTAL AND SIMULATED DATA

The data fusion system described in chs. 2 and 3 was tested using experimental and simulated data. This section describes the results and performance. The hardware dataset demonstrates that the proposed system is capable of accurate navigation under realistic conditions. Next, the motion capture data from this experiment is used to generate a realistic simulation. This simulation was designed to mimic the hardware test conditions. It indicates how well the system would perform with less parameter/calibration error and more robust feature tracking.

4.1 Hardware Setup

Although the proposed EKF-SLAM method is designed specifically for multirotors, in this work the algorithm is validated using the hand-held platform shown in fig. 1.3 . This is a highly modular platform that allows different sensors and single-board-computers to easily be mounted. For this work, the platform is outfitted with a forward-facing Pointgrey BFLY-PGE-13E4C-CS camera, Vectornav VN-100 IMU, Pixhawk flight controller, and PX4FLOW optical flow sensor. Only the rangefinder on the PX4FLOW is used in this work. The Pixhawk is only used as an interface between the PX4FLOW sensor and the computer recording the data. All of these components were designed for or fit the payload requirements of a medium-sized multirotor. In this experiment, the data was taken and recorded using the Robot Operating System (ROS) distributed computing framework.

The measurements were run through the previously described SLAM system offline. The measurement rates were maintained using the measurement timestamps. Monochrome images were recorded at 20 Hz with 640x512 pixel resolution using a copy of the Edmund Optics 4.5mm C-mount lens. Shutter speed, aperture, gain, and focus were held constant during the test. Pixel binning improved the signal-to-noise ratio in the dark lab

environment. The IMU accelerometer and gyroscope were sampled at 200 Hz while the rangefinder operated at 10 Hz.

The tests were conducted at the Texas A&M Land Air and Space Robotics (LASR) lab. LASR has a Vicon motion capture system that provides inertial pose measurements with millimeter accuracy. The motion capture data serves as the ground truth for these experiments. The motion capture space is limited (approximately 3x3 meters) so experiments must stay within this region.

Figure 4.1: Sample image from hardware experiment.

The sensor package was moved through the motion capture space. It was desirable to find a realistic scene that would show the strengths and weaknesses of the SLAM system. The camera faced a scene consisting of features at depths ranging from 1-6 meters. A sample image from the sequence is shown in fig. 4.1 . The camera moves far enough through the scene that few, if any, of the features seen at the beginning of the scene are visible throughout the entire sequence. For this reason the estimates were expected to drift.

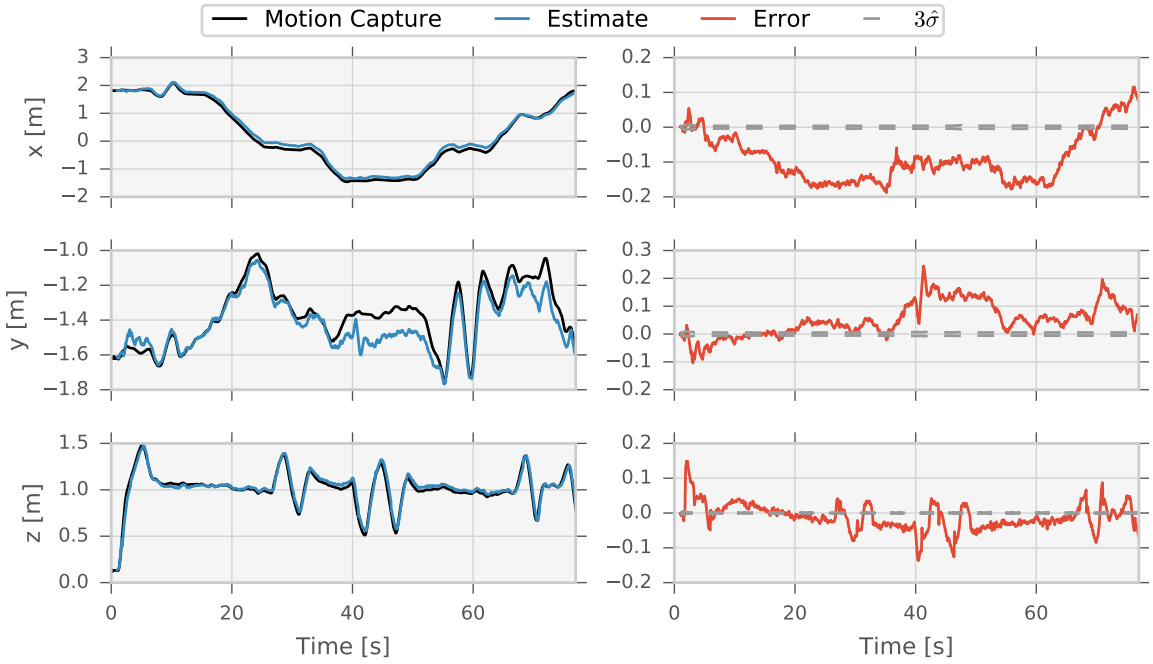

Figure 4.2: Position estimates and errors between the estimates and motion capture data. All units are in meters.

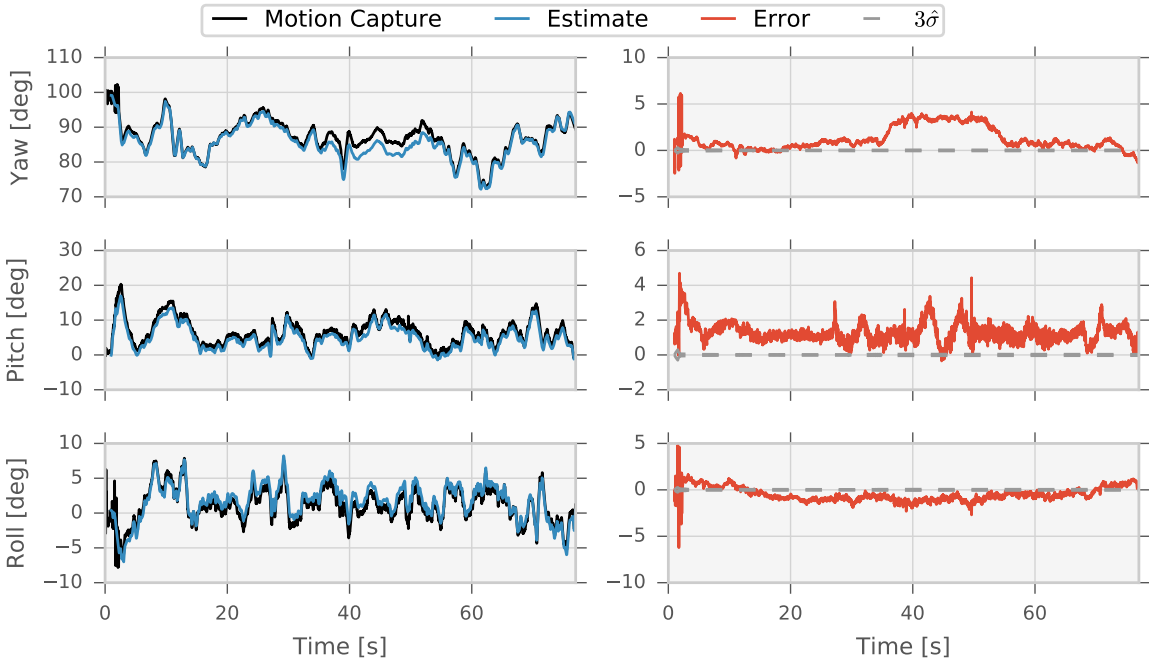

Figure 4.3: Attitude estimates and errors between the estimates and motion capture data. All units are in degrees. The biases in the roll and pitch errors are likely partially due to misalignment between the motion capture frame and the gravity-aligned navigation frame.

There is no loop closure so although the camera "revisits" some areas of the environment it cannot benefit from this prior information. Another challenging aspect of the sequence is that the majority of each image in the sequence consists of a black floor, wall, or ceiling with no discernible features. The lighting is poor and highly directional, leading to deep shadows.

4.2 Hardware Experimental Results

Figs. 4.2 and 4.3 show the motion capture and estimated IMU position during the sequence. The filter is clearly not consistent; the reported errors are significantly outside the estimated covariance bounds. For this experimental case, much of the inconsistency comes from not taking into account the uncertainty in the camera calibration parameters and the relative position and orientation of the rangefinder and camera with respect to the IMU. Having better estimates of these parameters would improve filter consistency.

There is also uncertainty in the origin and attitude of the motion capture body with respect to the IMU. This particular experimental error may account for some of the bias seen particularly in the attitude and altitude (position: z) results.

Despite the inconsistency, the visual odometry solution provides an accurate pose estimate. It is apparent that there is a scaling error in the x and y position estimates. This is most pronounced during the period between 20-60 seconds. The altitude estimate (z) is also affected by scale drift during this time. However, the metric rangefinder measurement prevented the scale from drifting significantly in any axis. The largest relative errors occur in the y-direction. The primary component of the depth of most of the mapped features lies along the y-axis (see fig. 4.4). For this reason it is understandably the most inaccurate direction. Overall, fig. 4.5 shows that the position errors are relatively small compared with the linear distance traveled.

The pitch and roll angles are observable states [27]. Although inconsistent, the errors

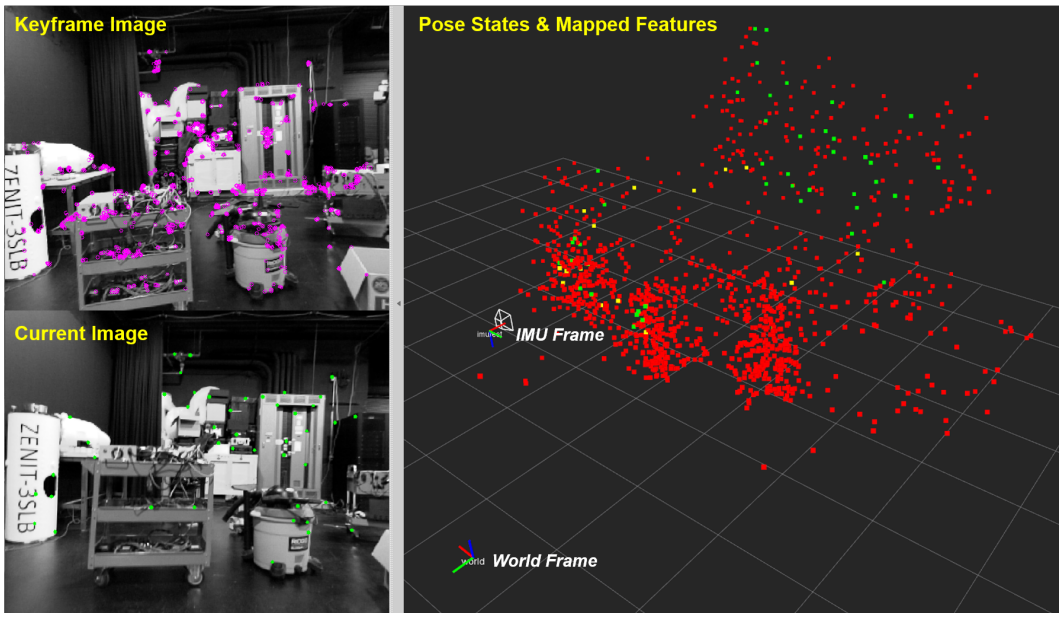

Figure 4.4: SLAM visualization. The camera translates primarily along the x-axis (red arrow on triad). The primary component of the depth of most of the mapped features (green, yellow, and red squares) lies along the world y-axis (green arrow on triad). Note that the red features are features that were once in the map but have been "forgotten" because they moved out of view or no new observations were recorded.

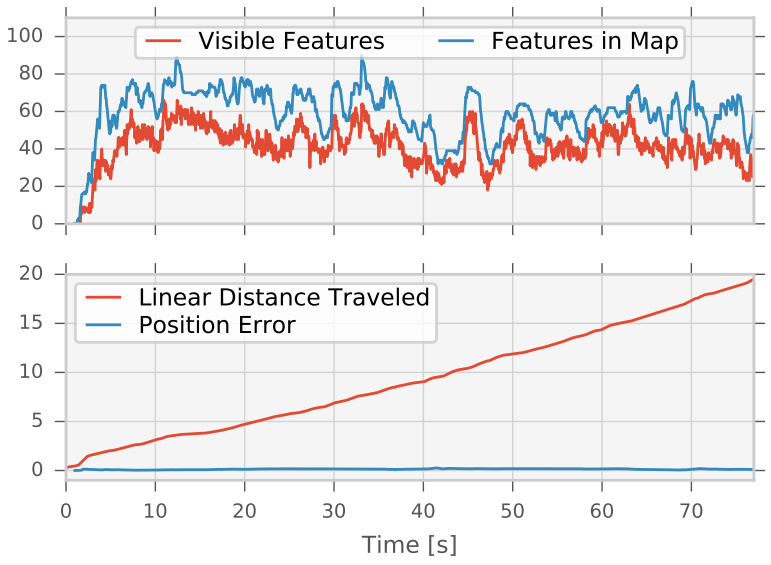

Figure 4.5: Top: the filter performs well when there are 30-50 feature observations available. Due to limitations with the feature tracker and camera motion, not all mapped features will always be associated with an observation. Bottom: the total length of the trajectory was 20 meters. The error at any point in time is small compared with the total distance traveled.

do not drift considerably. Yaw is not an observable state and is expected to drift over time.

4.3 Simulated Experimental Setup

In order to study how well the proposed EKF-SLAM solution might perform without the current limitations of the parameter uncertainties and feature tracking module, a simulation that mimics the hardware test was created. In this simulation, the hardware was modeled to match the setup described in sec. 4.1 as closely as possible. This includes camera calibration parameters, intersensor parameters, and sensor noise parameters.

Next, the second derivative of the motion capture position and first derivative of the motion capture attitude were numerically evaluated using the 5-point-stencil. These derivatives were assumed to be piecewise-linear and approximate the acceleration and rate of change of the attitude of the IMU, respectively. The trapezoidal rule and Simpson's rule were then used to integrate the numerical acceleration to find exact time histories for velocity and position given the piecewise-linear acceleration. The first order quaternion integrator from [40] was used to find exact time histories for attitude given the piecewise-linear attitude rate. The "re-integrated" pose is not identical to the original motion capture pose but is nearly so. There is no reason that they must match precisely; the goal is just to mimic the hardware experiment.

These kinematic time histories were used to generate IMU and rangefinder measurements using the sensor models of ch. 2. Noise terms were sampled from the appropriate normal distributions at each time step.

120 feature positions were randomly generated to generally match the distribution of features seen in the hardware experiment. 40 features were distributed along the virtual "back wall". The remaining 80 features were distributed in front of the wall. These feature positions and the kinematic time histories were used to create a time history of feature observations. Simulating feature measurements in this way allows vision-based estima-

tors to be evaluated separately from their feature tracking and matching implementation. However, the Guassian independent white noise samples used for the simulated measurements are not likely to accurately reflect the performance of most feature detectors and matchers under real lighting conditions. In addition, the simulated features are able to be observed in every frame while inside the camera's field of view. Due to camera motion blur and noise, a real feature matcher will not necessarily return observations to features in every frame. Finally, the simulated features are never occluded. This is another significant "advantage" of the simulated feature measurements.

Under these conditions, the simulated camera could "see" 40-50 features at a time. As shown in fig. 4.5 , this is approximately the same amount of features that the feature-tracking-and-matching module returned in each frame of the hardware experiment. The performance of the system is greatly dependent on the number of feature observations that can be made in each frame. Thus if a comparison will be made between the hardware and simulated results it is necessary that each case have access to approximately the same number of features.

4.4 Simulated Experimental Results

Figs. 4.6 and 4.7 compare the ground truth to the EKF estimates. Unlike the hardware case, most of the state estimates are relatively consistent for the first 20 seconds. This is most likely because there are no calibration or intersensor parameter uncertainties. However, like the hardware experimental results, most of the states begin to drift and become inconsistent after 20 seconds.

Because of the efforts to mimic the hardware testing conditions, the estimates are still not "perfect". By adding more features, reducing sensor noise, etc., a simulation that returns essentially perfect estimates could easily be constructed if desired. The performance of the simulated case shows improvement over the hardware case but, due to the attempted

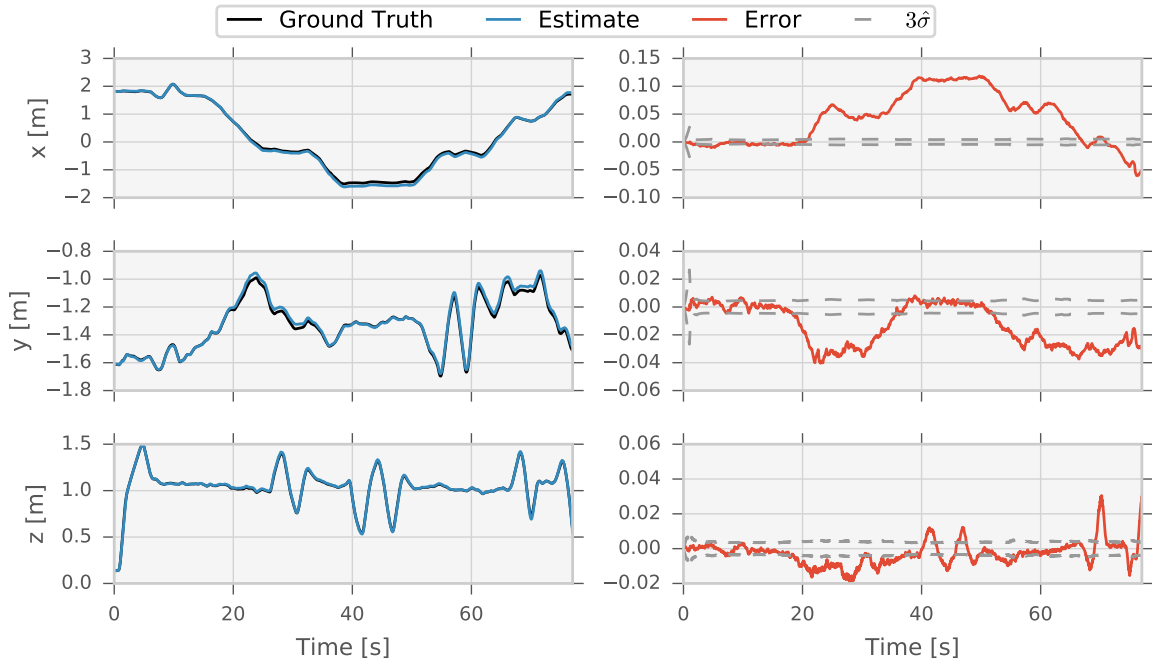

Figure 4.6: Position estimates and errors between the estimates and ground truth. All units are in meters.

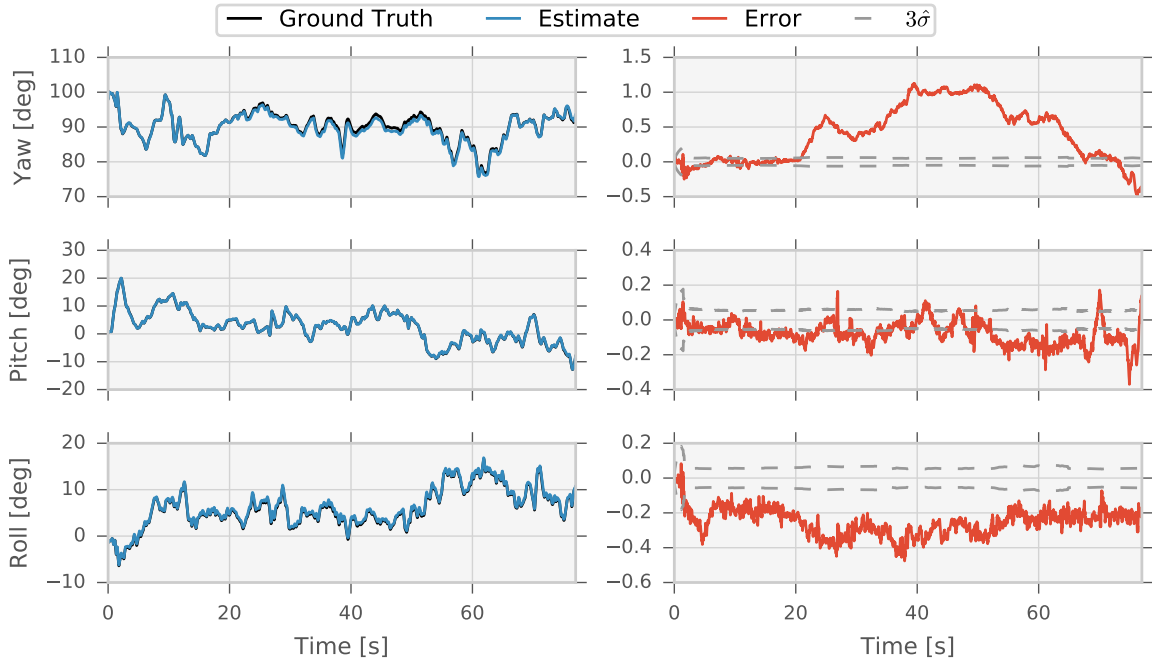

Figure 4.7: Attitude estimates and errors between the estimates and ground truth. All units are in degrees.

realism, demonstrates some of the same trends.

4.5 Discussion and Conclusions

The hardware and simulated test cases validate the navigation solution described in chs. 2 and 3. Many modifications have been previously suggested that could improve the overall filter performance. It is expected that these additions could improve performance over long trajectories. However, both the hardware and simulated test cases demonstrated that the proposed system produced estimates that are accurate compared with the distance traveled.

5. A GUIDANCE LAW TO IMPROVE ROBUSTNESS

The hardware and simulated experiments in ch. 5 demonstrate that the proposed sensor package and visual odometry can provide accurate pose estimates. Critically, these experiments also show that the rangefinder provides enough information to render estimates and maps with minimal scale ambiguity.

To minimize scale drift and inconsistency, a guidance law is proposed that takes advantage of the downward-facing rangefinder. Thanks to the rangefinder, changes in altitude are fully observable and produce a well-defined stereo baseline. For this reason, vertical motion allows new features to be initialized with greater accuracy than motion in another direction would permit. Strategically changing altitude could increase the performance of the estimator.

5.1 Altitude: A Confident Baseline

The use of a downward-facing rangefinder allows changes in altitude to be estimated directly. If the rangefinder has low measurement noise, it is intuitively clear that confident and accurate altitude measurements should be possible.

Interestingly, for small accelerations, the expected error in altitude estimates from purely integrating IMU measurements is less than the error in the x and y axes. This is independent of the use of a rangefinder. Consider again eq. (2.57) . This is the error model for acceleration assuming small attitude errors. The equation is rewritten here:

$$\Delta\dot{\mathbf{v}}_I = {}^I_W\hat{\mathbf{C}}^{\mathrm{T}}\big(- \Delta\mathbf{b}_a - \mathbf{n}_a + \lfloor\delta\boldsymbol{\theta}_I \times\rfloor(\tilde{\mathbf{s}} - \hat{\mathbf{b}}_a - \Delta\mathbf{b}_a - \mathbf{n}_a)\big)$$

The term $\Delta\dot{\mathbf{v}}_{I_s} = {}^I_W\hat{\mathbf{C}}^{\mathrm{T}}\lfloor\delta\boldsymbol{\theta}_I \times\rfloor(\tilde{\mathbf{s}} - \hat{\mathbf{b}}_a)$ is of interest. For a small acceleration ϵ and

gravity aligned world frame:

$$(\tilde{\mathbf{s}} - \hat{\mathbf{b}}_a) = {}^{I}_{W}\hat{\mathbf{C}} \begin{bmatrix} \epsilon_1 \\ \epsilon_2 \\ g + \epsilon_3 \end{bmatrix} \tag{5.1}$$

For small ϵ and $\delta\boldsymbol{\theta}_I$:

$$\Delta\dot{\mathbf{v}}_{I_s} = {}^{I}_{W}\hat{\mathbf{C}}^{\mathrm{T}} \lfloor \delta\boldsymbol{\theta}_I \times \rfloor (\tilde{\mathbf{s}} - \hat{\mathbf{b}}_a) \approx g \, ({}^{I}_{W}\hat{\mathbf{C}}^{\mathrm{T}} \lfloor \delta\boldsymbol{\theta}_I \times \rfloor {}^{I}_{W}\hat{\mathbf{C}}_{i3}) = \begin{bmatrix} \Delta\dot{\mathbf{v}}_{I_{s1}} \\ \Delta\dot{\mathbf{v}}_{I_{s2}} \\ 0 \end{bmatrix} \tag{5.2}$$

where ${}^{I}_{W}\hat{\mathbf{C}}_{i3}$ is the third column of ${}^{I}_{W}\hat{\mathbf{C}}$ and $g \approx 9.81 \; m/s^2$.

Because of the $\Delta\dot{\mathbf{v}}_{I_s}$ term, attitude errors have nearly 10 times the impact on acceleration errors in the x and y directions as they do in the z direction.

Fig. 5.1 shows an experimental case. An IMU was carried for 40 seconds while the Vicon motion capture system recorded the pose. The filter described in ch. 2 propagated the model without any measurement updates. It is clear that the altitude (z) estimate has less error and estimated uncertainty than the other two directions. For short time periods, the IMU can provide locally accurate estimates of the change in altitude relative to translations in the other two axes.

5.2 Guidance Law

Thanks to the rangefinder and the role of gravity, changes in altitude can be estimated metrically and accurate. This motivates the idea of deliberately initializing features using motion dominated by changes in altitude. Features initialized while changing altitude should be initialized accurately because the baseline between measurements is well known.

There are several ways that vertical motion could be used in a guidance law with the

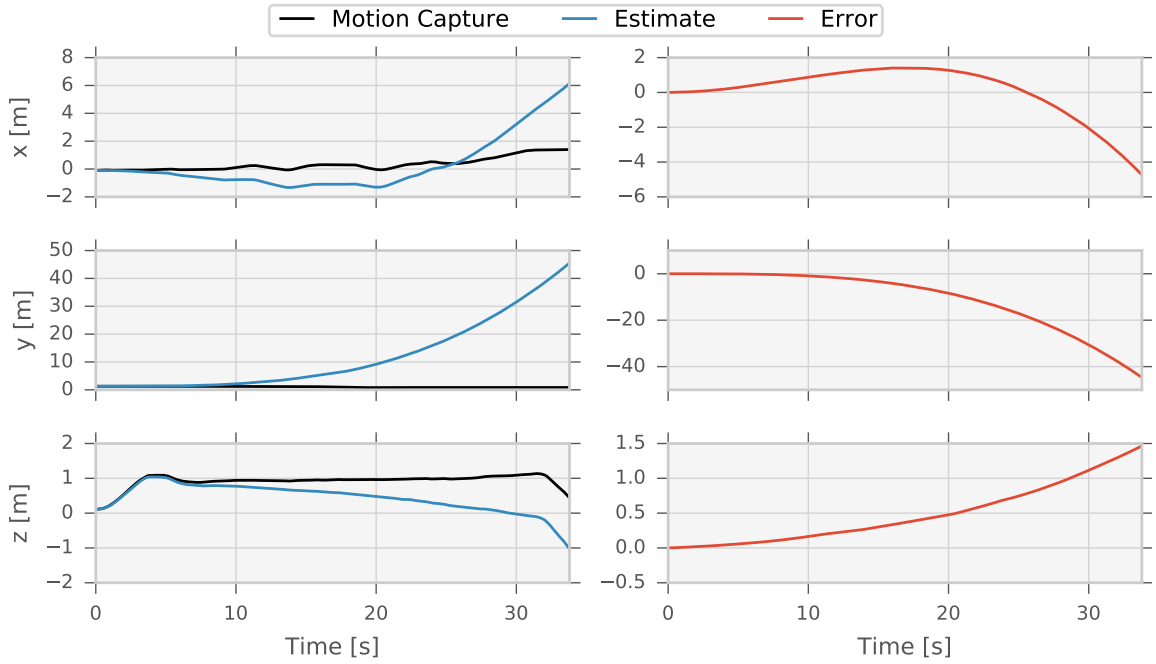

Figure 5.1: "IMU-only" inertial navigation. The error of the position estimate in the z (altitude) direction is much lower than the errors in the other two directions.

intention of improving state estimates. An autonomous vehicle will have some type of guidance or path planning associated with it's mission objective. Potential missions include mapping a room, picking up an object, or going from point A to point B while avoiding obstacles. The path planning associated with these missions could be augmented to include strategic vertical motion.

However, multirotor vehicles have stringent restrictions on battery power. Increasing altitude requires additional thrust. This reduces the amount of energy that could be spent on the primary mission objective. A more desirable approach is an integrated navigation and guidance system. The navigation system would provide a metric that indicates when the estimation performance needs the information provided by the altitude change. Only then would the primary mission be halted or augmented to include the vertical motion. This basic idea is capture in algorithm 1.

Algorithm 1 Guidance Law 1

Input: γ, where γ is a pose uncertainty metric
Input: Γ, where Γ is a preset threshold on the allowable pose uncertainty metric
Input: h, where h is the altitude
  if $\gamma < \Gamma$ **then**
    $h_0 \Leftarrow h$
    while $\gamma < \Gamma$ **do**
      Command altitude increase
    end while
    while $h_k > h_0$ **do**
      Command altitude decrease
    end while
  else
    Command input to achieve nominal mission objective (e.g. map a room)
  end if

Algorithm 1 is intended to be generally applicable to any monocular-vision-aided navigation solution with a suitable criteria γ. The monocular SLAM solution in this work is well-suited for this integrated estimation and guidance approach due to the feature initialization criteria specified in sec. 3.3.3. Features are only added to the map if there is enough information to confidently triangulate it's position. Thus, the number of features in the map can be used as the threshold γ.

5.3 Guidance Law Demonstration

The impact of the proposed guidance law is demonstrated via simulation. Throughout the course of this work it has been clear that moving up and down greatly reduces scale drift in hardware test cases. However, the performance of monocular visual odometry is very sensitive to the environment and the amount of features that are visible. Because the current implementation does not have local-loop closure, feature information is routinely "forgotten" if the camera yaws away from the current scene. This reduces filter performance.

In order to demonstrate the impact of the proposed guidance law on the accuracy of the estimates, it is necessary to show the state estimates for two trajectories: one with the guidance law implemented and one without. It essential to eliminate as many variables between test cases as possible. This naturally leads to the use of simulation, where most extraneous variables can be controlled. Ch. 4 demonstrated that a realistic simulation could be constructed and that the EKF-SLAM system had similar, though improved, performance between simulated and hardware data.

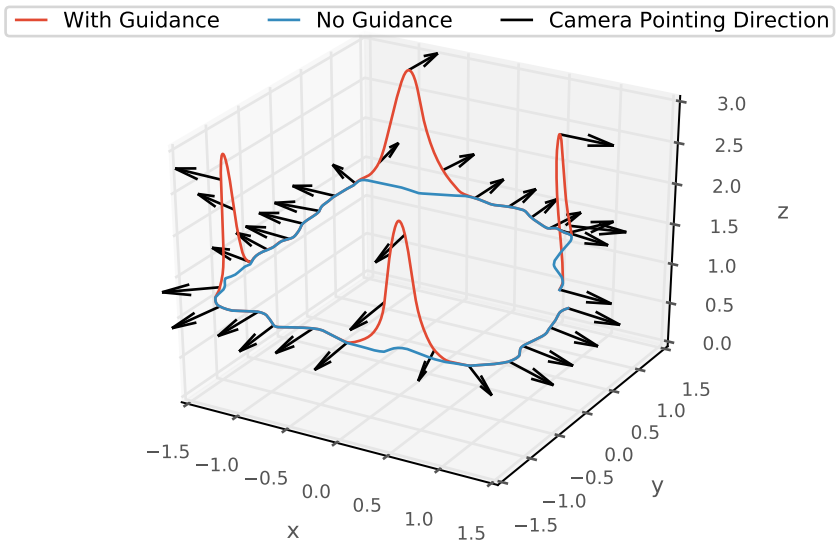

Figure 5.2: Two trajectories are analyzed. The nominal path without the proposed guidance law is shown in blue. This path was augmented per the proposed guidance law to produce the red path.

To demonstrate the effect of the guidance law, a highly challenging scenario for monocular SLAM was constructed. In this scenario, a quadrotor is asked to map a virtual room with four walls. Two paths were considered. First, a nominal path was planned to achieve the mapping objective. In this path (blue line, fig. 5.2), the camera translates parallel to

each wall. At the end of each wall, the vehicle quickly yaws approximately 90 degrees to view the next wall. Since yaw is not globally observable, and rotation does not add information about features, this is a very "dangerous" maneuver for a vehicle using monocular SLAM. After the yaw rotation, the camera can no longer observe the vast majority of the features in the current map. The kinematics from this trajectory were generated from motion capture data using the methods of sec. 4.3. In the second path, the nominal path was augmented with 1.5 meter altitude changes (red line, fig. 5.2) immediately after the yaw rotations began.

95 features were distributed along each virtual wall. The feature positions were identical in both test cases. All other simulation parameters are the same as in sec. 4.3. It should be noted that because the sensor noise is randomly sampled, the sensor measurements in each test case are not identical. Otherwise, all extraneous variables are held constant between the two trajectories.

5.4 Estimator Performance: Without Guidance Law

Fig. 5.3 compares the true trajectory to the estimated pose for the nominal path without the guidance law. The estimator diverges dramatically after the second corner turn around $t = 40s$. After this turn, all mapped features fell outside the camera's field of view and the subsequent translation along the y-axis did not provide sufficient information to initialize new features because the "IMU-only" navigation quickly became inaccurate and unconfident.

One may ask whether or not the feature initialization criteria is too "stringent" and whether features should be allowed to be added to the map with less confidence. This was considered and the resulting estimates diverged even before the second turn was reached. It is possible that using the inverse depth parameterization proposed by Civera [25] rather than Euclidean coordinates would make this a viable alternative. This 6-parameter set has

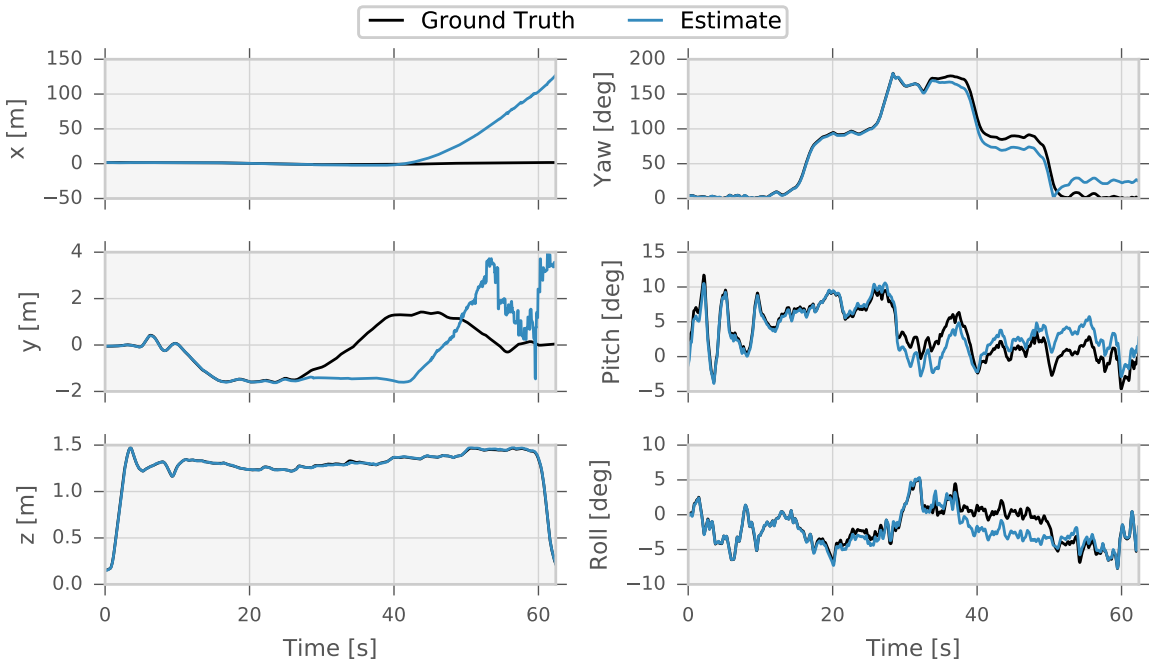

Figure 5.3: True and estimated pose on trajectory without guidance solution.

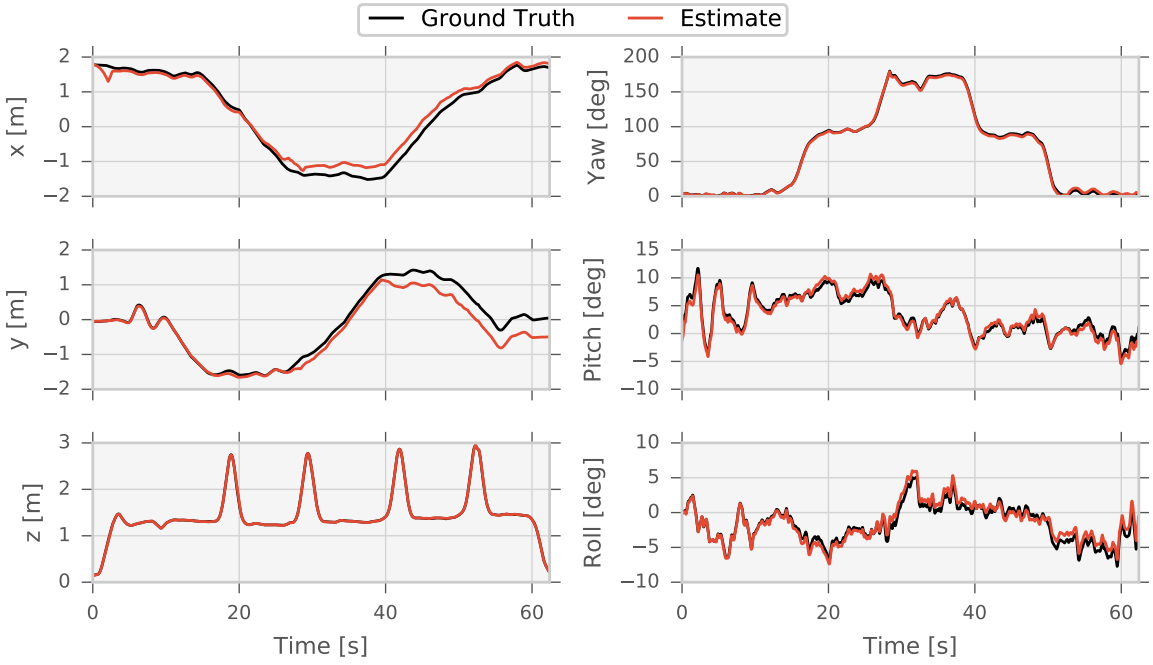

Figure 5.4: True and estimated pose on trajectory with guidance solution.

been shown to have better convergence with poor initial conditions.

5.5 Estimator Performance: With Guidance Law

Fig. 5.4 compares the true trajectory to the estimated pose for the path that utilizes the guidance solution. Though there is non-negligible drift, the estimator is able to provide stable pose estimates during the entire length of the difficult trajectory. Fig. 5.5 shows the number of mapped features over time. During each turn, existing features quickly leave the field of view of the camera and are removed from the map. No depth information is gained for features during the yaw rotation. The feature tracking uncertainty criteria correctly and "automatically" prevents new features from being admitted to the map during the rotation. However, as the vehicle translates vertically, the metric information allows features to confidently be initialized and added to the map. The "troughs" of the red line in fig. 5.5 are indicative of this portion of the trajectory. The blue line of fig. 5.5 is a threshold on the number of features that could have been used to determine when to execute the altitude change.

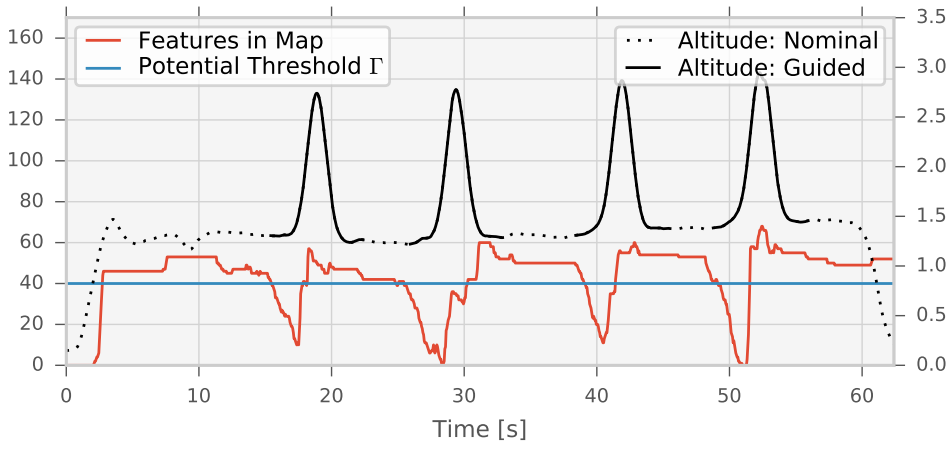

Figure 5.5: Immediately after the yaw rotations there are few mapped features available. Moving up and down establishes a baseline for more features to be initialized. The number of mapped features could be used as a threshold for when to execute the altitude changes.

5.6 Discussion

In addition to or instead of relying on an uncertainty metric, the vertical motion could be added in the path planning stage to maneuvers that are considered “high risk”. In the simulated scenario, for instance, increasing altitude during the yaw rotation might have limited some of the drift present in the present solution. However, in general the need for the vertical motion to initialize new features may not be so easily predicted.

6. SUMMARY AND CONCLUSIONS

A filter-based monocular SLAM solution was described in detail. The filter uses IMU and rangefinder measurements in addition to camera images. This navigation solution uses a consider least squares approach to estimate the feature covariance and avoid the use of an arbitrary prior. The consider initialization approach is simple to implement and immediately provides cross-correlation covariance terms with other states. Only features that can be initialized with a confident depth estimate are added to the map. This minimizes the likelihood of adding features to the map which will contribute little useful information. The navigation solution is validated using experimental data and a simulated trajectory derived from motion capture data. A simple guidance law was proposed that takes advantage of the downward-facing rangefinder. Vertical motion creates a metric and accurate baseline that can be used to confidently initialize new features. A simulated scenario shows that changing altitude dramatically improves the performance of the estimator over a challenging trajectory.

Due to the feature initialization criteria of the filter, the number of features in the map serves as a strong indication for when the guidance maneuverer is required. However, there are many alternative integrated estimation and guidance approaches that could be used. The guidance law is not limited to a particular state estimation scheme. Thus the key concepts of this work could be applied to a variety monocular-vision-aided navigation systems.

REFERENCES

[1] D. Scaramuzza, M. C. Achtelik, L. Doitsidis, F. Friedrich, E. Kosmatopoulos, A. Martinelli, M. W. Achtelik, M. Chli, S. Chatzichristofis, L. Kneip, *et al.*, "Vision-controlled micro flying robots: from system design to autonomous navigation and mapping in GPS-denied environments," *IEEE Robotics & Automation Magazine*, vol. 21, no. 3, pp. 26–40, 2014.

[2] G. Vásárhelyi, C. Virágh, G. Somorjai, N. Tarcai, T. Szörényi, T. Nepusz, and T. Vicsek, "Outdoor flocking and formation flight with autonomous aerial robots," in *2014 IEEE/RSJ International Conference on Intelligent Robots and Systems*, pp. 3866–3873, IEEE, 2014.

[3] S. Lupashin, A. Schöllig, M. Sherback, and R. D'Andrea, "A simple learning strategy for high-speed quadrocopter multi-flips," in *Robotics and Automation (ICRA), 2010 IEEE International Conference on*, pp. 1642–1648, IEEE, 2010.

[4] R. Ritz, M. W. Müller, M. Hehn, and R. D'Andrea, "Cooperative quadrocopter ball throwing and catching," in *2012 IEEE/RSJ International Conference on Intelligent Robots and Systems*, pp. 4972–4978, IEEE, 2012.

[5] J. Courbon, Y. Mezouar, N. Guénard, and P. Martinet, "Vision-based navigation of unmanned aerial vehicles," *Control Engineering Practice*, vol. 18, no. 7, pp. 789–799, 2010.

[6] S. Ahrens, D. Levine, G. Andrews, and J. P. How, "Vision-based guidance and control of a hovering vehicle in unknown, GPS-denied environments," in *Robotics and Automation, 2009. ICRA'09. IEEE International Conference on*, pp. 2643–2648, IEEE, 2009.

[7] M. Blösch, S. Weiss, D. Scaramuzza, and R. Siegwart, "Vision based MAV navigation in unknown and unstructured environments," in *Robotics and Automation (ICRA), 2010 IEEE International Conference on*, pp. 21–28, IEEE, 2010.

[8] G. Klein and D. Murray, "Parallel tracking and mapping on a camera phone," in *Mixed and Augmented Reality, 2009. ISMAR 2009. 8th IEEE International Symposium on*, pp. 83–86, IEEE, 2009.

[9] S. Weiss, M. W. Achtelik, S. Lynen, M. C. Achtelik, L. Kneip, M. Chli, and R. Siegwart, "Monocular vision for long-term micro aerial vehicle state estimation: A compendium," *Journal of Field Robotics*, vol. 30, no. 5, pp. 803–831, 2013.

[10] R. Brockers, S. Susca, D. Zhu, and L. Matthies, "Fully self-contained vision-aided navigation and landing of a micro air vehicle independent from external sensor inputs," in *SPIE Defense, Security, and Sensing*, pp. 83870Q–83870Q, International Society for Optics and Photonics, 2012.

[11] J. Engel, J. Sturm, and D. Cremers, "Camera-based navigation of a low-cost quadrocopter," in *2012 IEEE/RSJ International Conference on Intelligent Robots and Systems*, pp. 2815–2821, IEEE, 2012.

[12] 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. 1, 2015.

[13] A. Bachrach, S. Prentice, R. He, and N. Roy, "RANGE–robust autonomous navigation in GPS-denied environments," *Journal of Field Robotics*, vol. 28, no. 5, pp. 644–666, 2011.

[14] S. Shen, N. Michael, and V. Kumar, "Autonomous multi-floor indoor navigation with a computationally constrained MAV," in *Robotics and Automation (ICRA), 2011*

IEEE International Conference on, pp. 20–25, IEEE, 2011.

[15] N. Michael, S. Shen, K. Mohta, Y. Mulgaonkar, V. Kumar, K. Nagatani, Y. Okada, S. Kiribayashi, K. Otake, K. Yoshida, *et al.*, "Collaborative mapping of an earthquake-damaged building via ground and aerial robots," *Journal of Field Robotics*, vol. 29, no. 5, pp. 832–841, 2012.

[16] A. S. Huang, A. Bachrach, P. Henry, M. Krainin, D. Maturana, D. Fox, and N. Roy, "Visual odometry and mapping for autonomous flight using an RGB-D camera," in *International Symposium on Robotics Research (ISRR)*, vol. 2, 2011.

[17] R. Leishman, J. Macdonald, T. McLain, and R. Beard, "Relative navigation and control of a hexacopter," in *Robotics and Automation (ICRA), 2012 IEEE International Conference on*, pp. 4937–4942, IEEE, 2012.

[18] L. Heng, L. Meier, P. Tanskanen, F. Fraundorfer, and M. Pollefeys, "Autonomous obstacle avoidance and maneuvering on a vision-guided MAV using on-board processing," in *Robotics and automation (ICRA), 2011 IEEE International Conference on*, pp. 2472–2477, IEEE, 2011.

[19] F. Fraundorfer, L. Heng, D. Honegger, G. H. Lee, L. Meier, P. Tanskanen, and M. Pollefeys, "Vision-based autonomous mapping and exploration using a quadrotor MAV," in *2012 IEEE/RSJ International Conference on Intelligent Robots and Systems*, pp. 4557–4564, IEEE, 2012.

[20] K. Schmid, P. Lutz, T. Tomić, E. Mair, and H. Hirschmüller, "Autonomous vision-based micro air vehicle for indoor and outdoor navigation," *Journal of Field Robotics*, vol. 31, no. 4, pp. 537–570, 2014.

[21] S. Shen, Y. Mulgaonkar, N. Michael, and V. Kumar, "Multi-sensor fusion for robust autonomous flight in indoor and outdoor environments with a rotorcraft MAV," in

2014 IEEE International Conference on Robotics and Automation (ICRA), pp. 4974–4981, IEEE, 2014.

[22] D. P. Koch, T. W. McLain, and K. M. Brink, "Multi-sensor robust relative estimation framework for GPS-denied multirotor aircraft," in *Unmanned Aircraft Systems (ICUAS), 2016 International Conference on*, pp. 589–597, IEEE, 2016.

[23] A. J. Davison, "Real-time simultaneous localisation and mapping with a single camera," in *Computer Vision, 2003. Proceedings. Ninth IEEE International Conference on*, pp. 1403–1410, IEEE, 2003.

[24] A. J. Davison, I. D. Reid, N. D. Molton, and O. Stasse, "MonoSLAM: Real-time single camera SLAM," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 29, no. 6, pp. 1052–1067, 2007.

[25] J. Civera, A. J. Davison, and J. M. Montiel, "Inverse depth parametrization for monocular SLAM," *IEEE Transactions on Robotics*, vol. 24, no. 5, pp. 932–945, 2008.

[26] A. I. Mourikis and S. I. Roumeliotis, "A multi-state constraint Kalman filter for vision-aided inertial navigation," in *Proceedings 2007 IEEE International Conference on Robotics and Automation*, pp. 3565–3572, IEEE, 2007.

[27] M. Li and A. I. Mourikis, "High-precision, consistent EKF-based visual–inertial odometry," *The International Journal of Robotics Research*, vol. 32, no. 6, pp. 690–711, 2013.

[28] M. Li, "Visual-inertial odometry on resource-constrained systems," 2014.

[29] B. Triggs, P. F. McLauchlan, R. I. Hartley, and A. W. Fitzgibbon, "Bundle adjustment – a modern synthesis," in *International Workshop on Vision Algorithms*, pp. 298–372, Springer, 1999.

[30] R. Mur-Artal, J. Montiel, and J. D. Tardós, "ORB-SLAM: a versatile and accurate monocular SLAM system," *IEEE Transactions on Robotics*, vol. 31, no. 5, pp. 1147–1163, 2015.

[31] E. Rublee, V. Rabaud, K. Konolige, and G. Bradski, "ORB: An efficient alternative to SIFT or SURF," in *2011 International Conference on Computer Vision*, pp. 2564–2571, IEEE, 2011.

[32] E. Eade and T. Drummond, "Edge landmarks in monocular SLAM," *Image and Vision Computing*, vol. 27, no. 5, pp. 588–596, 2009.

[33] J. Engel, T. Schöps, and D. Cremers, "LSD-SLAM: Large-scale direct monocular SLAM," in *European Conference on Computer Vision*, pp. 834–849, Springer, 2014.

[34] C. Forster, M. Pizzoli, and D. Scaramuzza, "SVO: Fast semi-direct monocular visual odometry," in *2014 IEEE International Conference on Robotics and Automation (ICRA)*, pp. 15–22, IEEE, 2014.

[35] Z. Zhang, "A flexible new technique for camera calibration," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 22, no. 11, pp. 1330–1334, 2000.

[36] O. D. Faugeras, Q.-T. Luong, and S. J. Maybank, "Camera self-calibration: Theory and experiments," in *European Conference on Computer Vision*, pp. 321–334, Springer, 1992.

[37] 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, IEEE, 2013.

[38] M. Faessler, F. Fontana, C. Forster, and D. Scaramuzza, "Automatic re-initialization and failure recovery for aggressive flight with a monocular vision-based quadro-

tor," in *2015 IEEE International Conference on Robotics and Automation (ICRA)*, pp. 1722–1729, IEEE, 2015.

[39] D. P. Woodbury, M. Majji, and J. L. Junkins, "Considering measurement model parameter errors in static and dynamic systems," *The Journal of the Astronautical Sciences*, vol. 58, no. 3, pp. 461–478, 2011.

[40] N. Trawny and S. I. Roumeliotis, "Indirect Kalman filter for 3D attitude estimation," *University of Minnesota, Dept. of Comp. Sci. & Eng., Tech. Rep*, vol. 2, p. 2005, 2005.

[41] T. Lupton and S. Sukkarieh, "Visual-inertial-aided navigation for high-dynamic motion in built environments without initial conditions," *IEEE Transactions on Robotics*, vol. 28, no. 1, pp. 61–76, 2012.

[42] J. L. Crassidis and J. L. Junkins, *Optimal estimation of dynamic systems*. CRC press, 2011.

[43] J. A. Hesch, D. G. Kottas, S. L. Bowman, and S. I. Roumeliotis, "Camera-IMU-based localization: Observability analysis and consistency improvement," *The International Journal of Robotics Research*, vol. 33, no. 1, pp. 182–201, 2014.

[44] G. P. Huang, A. I. Mourikis, and S. I. Roumeliotis, "Observability-based rules for designing consistent EKF SLAM estimators," *The International Journal of Robotics Research*, vol. 29, no. 5, pp. 502–528, 2010.

[45] J. Civera, O. G. Grasa, A. J. Davison, and J. Montiel, "1-point RANSAC for EKF-based structure from motion," in *Intelligent Robots and Systems, 2009. IROS 2009. IEEE/RSJ International Conference on*, pp. 3498–3504, IEEE, 2009.

[46] M. Bloesch, S. Omari, M. Hutter, and R. Siegwart, "Robust visual inertial odometry using a direct EKF-based approach," in *Intelligent Robots and Systems (IROS), 2015*

IEEE/RSJ International Conference on, pp. 298–304, IEEE, 2015.

[47] M. Muja and D. G. Lowe, "Fast approximate nearest neighbors with automatic algorithm configuration.," *VISAPP (1)*, vol. 2, no. 331-340, p. 2, 2009.

[48] D. G. Lowe, "Distinctive image features from scale-invariant keypoints," *International Journal of Computer Vision*, vol. 60, no. 2, pp. 91–110, 2004.

[49] R. I. Hartley, "In defense of the eight-point algorithm," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 19, no. 6, pp. 580–593, 1997.

[50] M. Bryson and S. Sukkarieh, "Building a robust implementation of bearing-only inertial SLAM for a UAV," *Journal of Field Robotics*, vol. 24, no. 1-2, pp. 113–143, 2007.

[51] S. I. Roumeliotis and J. W. Burdick, "Stochastic cloning: A generalized framework for processing relative state measurements," in *Robotics and Automation, 2002. Proceedings. ICRA'02. IEEE International Conference on*, vol. 2, pp. 1788–1795, IEEE, 2002.

[52] J. Sola, "Consistency of the monocular EKF-SLAM algorithm for three different landmark parametrizations," in *Robotics and Automation (ICRA), 2010 IEEE International Conference on*, pp. 3513–3518, IEEE, 2010.

[53] A. J. Davison, I. D. Reid, N. D. Molton, and O. Stasse, "MonoSLAM: Real-time single camera SLAM," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 29, no. 6, pp. 1052–1067, 2007.