MONOCULAR VISION-BASED GUIDANCE AND NAVIGATION

A Thesis

by

ANDREW VERRAS

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

August 2021

Major Subject: Aerospace Engineering

ABSTRACT

This thesis develops mathematical methods for utilizing monocular camera measurements for the guidance and navigation of various aerospace vehicles. For each of the three projects discussed, a nonlinear estimation algorithm is developed and then applied for testing in the field, in the laboratory, and/or in simulation. The first project is an Entry, Descent, and Landing (EDL) application in the presence of *a priori* unknown terrain. The proposed algorithm extracts features from the image, and integrates the camera measurements with onboard inertial sensors to orient a landing vehicle with respect to the terrain. The filter is tested using model terrain in a laboratory setting as well as simulated terrain. The second project concerns automated aerial refueling and develops a filter that uses images of a known target along with inertial sensors and GPS to provide accurate estimates of the relative position and orientation of two airborne vehicles. The target vehicle is marked by LED beacons which allow for fast image processing. Results are obtained from field testing using two automobiles and from laboratory testing using robotic platforms from the Land, Air, and Space Robotics (LASR) Laboratory at Texas A&M University. Finally, the third project is for a space debris removal application. The target is a depleted rocket body in Earth orbit. It has a known shape and size but has no onboard sensors or beacons. The image projection of the target's rocket nozzle is detected by the computer vision software, and an ellipse is fitted to the projection. The parameters of this ellipse are used to estimate the target's position and orientation, which then guides an onboard capture system to secure the target. This process is tested using robots from the LASR lab.

ACKNOWLEDGMENTS

I would like to thank my advisor, Dr. Manoranjan Majji, for always having so many interesting projects to work on, and for his guidance and expertise. His classes and directed study sessions were illuminating, and his patience when I had too many questions was impressive. His professionalism and constant enthusiasm are contagious and have made him a great role model for me.

I appreciate my colleagues who endured long hours in the lab with me while I fine-tuned my code to work in tandem with their systems, especially Davis Adams, Caleb Peck, and James McElreath. Additionally, I thank Dr. Carolina Restrepo for setting aside time with me to provide technical advice and mentoring.

My love and thanks go to Jacalynn, who was always willing to hear me rant about my work, to sympathize with me when I was stressed, and to celebrate with me when I finally reached a milestone. I also want to thank my parents for teaching me to have an analytical mind and to follow my dreams.

CONTRIBUTORS AND FUNDING SOURCES

Contributors

This work was supported by a thesis committee consisting of Dr. Manoranjan Majji and Dr. John Junkins of the Department of Aerospace Engineering and Dr. Zixiang Xiong of the Department of Electrical and Computer Engineering.

The Entry Descent and Landing filter in Chapter 3 was developed under the guidance of Carolina Restrepo from National Aeronautics and Space Administration (NASA) Goddard Space Flight Center and Ronney Lovelace from NASA Johnson Space Center (JSC). The implementation of the aerial refueling tracking software in Chapter 4 was done in collaboration with industry partner, VectorNav Technologies. The proportional integral derivative (PID) controller used for the laboratory testing in section 4.6 was designed by graduate student Davis Adams from Texas A&M University.

All other work conducted for the thesis was completed by the student independently.

Funding Sources

Graduate study was supported by a graduate research assistantship from Texas A&M University and the following sources.

1. Support from National Geospatial Intelligence Agency (NGA) through Grant No. HM0476-19-1-2015 is gratefully acknowledged. Dr. Scott A. True is thanked for serving as a technical monitor and research advocate for this project.

2. Office of Naval Research Grant Number N00014-19-1-2435 is acknowledged for their support. The author is thankful to Dr. Brian Holm-Hansen and Dr. David Gonzales for their support.

3. DOD STTR Phase 2 - Reconstruction and Pose Estimation of Space Objects, with Vision Systems Inc. (VSI) as the prime organization produced a rendering pipeline

used to verify some of the methods proposed here-in. Our collaborators, Drs. Joe Mundy and Daniel Crispell are gratefully acknowledged for the opportunity to work together.

4. NASA's Johnson Space Center (JSC) Cooperative Agreement entitled "Onboard Adaptive Safe Site Identification Systems (OASSIS)" is acknowledged for partially supporting the research work carried out in this thesis. Technical interactions with Drs. Carolina Restrepo and John Carson are gratefully acknowledged. The author is also grateful for technical assistance and expertise of Mr. Ronney Lovelace on this project.

5. DOD STTR Phase 2 - LASR Support of Flexible Body Dynamics Testing of Air-to-Air Refueling Concepts, with VectorNav Technologies Inc., as the prime organization. The robotic testing of the refueling project was supported by this work. Dr. Jeremy Davis is acknowledged for his technical mentoring. The author is grateful to Dr. Michael D. Pilkenton for his kind help.

6. DOD STTR Phase 2 - Low Thrust Trajectory Optimization is acknowledged for partially supporting the work involving the rocket body capture demonstration experiment. The author appreciates Dr. Alok Das for his conceptual design of the rocket body capture idea and Dr. Moble Benedict for suggesting the pressurized projectile to demonstrate the concept. AFRL Rapid Innovations Office (Dr. Micheal Stallard) is acknowledged for their support.

TABLE OF CONTENTS

LIST OF FIGURES

LIST OF TABLES

1. INTRODUCTION

The improvements to camera technology over the past few decades has made computer vision into an integral part of many guidance, navigation, and control (GNC) systems. Modern cameras are able to take hundreds of images per second, with each image containing millions of pixels. Typical estimators use an inertial measurement unit (IMU), which contains devices such as an accelerometer and gyroscope to track a body's position and orientation. Using an IMU on its own, however, exhibits some issues. Most prominently, an IMU experiences a continual drift from the truth which builds up over time. The drift can be modeled as a stochastic process and estimated as part of the system's state vector. However this method requires additional perception capabilities that can assess the biases accumulated by the inertial sensors. The inertial localization information in the perception system can be provided by a camera or a light detection and ranging (LIDAR) system.

A camera is an attractive option for this additional sensor, because it is inexpensive compared to more complex sensors such as the LIDAR. While cameras constitute an inexpensive solution to acquire data and measurements related to the environment in which an autonomous vehicle operates, the image processing of unstructured images of natural and man-made scenes is a considerable challenge. The area of automated understanding and characterization of such scenes has emerged as an exciting subject called computer vision [2, 3].

Typically, autonomous guidance and control of robotic vehicles necessitates the estimation of the pose (location and orientation) of the sensor platform, in addition to a model of the environment being imaged. Computer vision systems can be sensitive to changes in targets' positions that occur in the plane of the camera. This is attributable to the mechanics of image transformation. Thus, applications where a target is expected to mostly move in that plane are ideal.

Estimation of position along the axis of the camera is still possible, albeit more difficult.

This limitation can be mitigated by using a stereo camera setup which uses multiple cameras at known displacements to produce depth information, although this is more expensive than a monocular camera. Stereo cameras are also more difficult to characterize due to the potential for misalignment, since the relative position and orientation of the two cameras must be known precisely at setup. Even with a single monocular camera, however, there are ways to estimate the full position and orientation of the target. This thesis will examine a few ways to implement a GNC system using a monocular camera as the sensor.

A computer vision system must take an image and convert it into data for use in an algorithm. The systems discussed in this thesis require the extraction and tracking of features, i.e., parts of the image that stand out and appear similarly over multiple images.

The kinds of features extracted vary based on the type of problem being solved. Computer vision tracking problems can be separated into two broad classes: structured and unstructured feature data. Structured feature data refers to a target with known geometry. This simplifies the feature extraction, because only features that resemble the target are sought. Unstructured feature data refers to unknown geometries that must be tracked. In this case, the decision on what to use as a feature is based not on a known object's properties, but rather on general considerations on what makes a feature easiest to pinpoint in an image. Normally, these features will be corners where two edges meet. This is because while edges contain many identical points than cannot be disambiguated, corners are unique. There are many robust algorithms for the general problem of feature extraction and tracking such as Scale Invariant Feature Transform (SIFT) [4], Speeded-Up Robust Features (SURF) [5], and Binary Robust Invariant Scalable Keypoints (BRISK) [6].

The case of unstructured feature data is studied in a terrain relative navigation (TRN) application. In this situation, an aerospace vehicle is moving relative to terrain, and an estimate is obtained for the vehicle's position, orientation, and velocity relative to the terrain. The first step in this problem is extracting features from the images. These features will be distinct parts of the terrain, such as sharp peaks or corners of rocks. Since the layout of the terrain is not known *a priori*, the relative locations of the detected features must be estimated by the navigation system. Obtaining the relative feature locations gives a map of the terrain, and the vehicle's state is then estimated relative to this map.

A TRN algorithm for unstructured data is useful in applications where a space vehicle needs to navigate an unfamiliar environment. One example is proximity operations in the vicinity of an asteroid, such as a sample-return or mining mission. The application discussed in this thesis is entry, descent, and landing (EDL) operations in an environment with unknown topography. This would be the case for a lander on Europa or another moon in the outer solar system, since high-resolution images of the surface do not exist. Such a lander would need to adapt on-the-fly to unexpected terrain features [7]. In particular, the vehicle should avoid craft-sized hazards for safe landing in planetary environments. Also, such vehicles require a landing site that falls within specific tolerances for slope of the terrain, roughness of the terrain, and other metrics. Thus the precision of the vehicle's state estimate is important, as is the detection of hazards.

This thesis develops a new camera-based Multiplicative Extended Kalman Filter (MEKF) for this TRN application. This is different from the typical star-tracker application because the features are considered not to be at infinity, but rather at finite depth. The filter is tested in TRN simulations and the results are shown in this thesis. The software suite is currently being implemented in a medium fidelity EDL test bed at NASA's Johnson Space Center under the Safe and Precise Landing – Integrated Capabilities Evolution (SPLICE) project [8].

In the case of structured data, an object is present that has a known shape and size, and the goal is to determine the position and orientation of the object relative to the camera. Knowledge of the target object's geometry simplifies the problem in several ways. For one, the known shape reduces the search-space for feature extraction, since only objects of a certain shape need to be detected. For example, in one application discussed here, the target is marked by LED beacons. The beacon modulation enables accurate and reliable object recognition. Another benefit of structured data is the knowledge of scale. This allows for a much more accurate determination of the distance to the target along the line of the camera. This quantity is typically the most difficult for a vision system to estimate, but in this case the size of the object's projection in the image provides a valuable clue. However, to establish the structure, the target of interest needs to be cooperative and accessible.

This thesis explores two applications that involve structured data and cooperative targets. One is the case of automated aerial refueling. In aerial refueling, a tanker aircraft unloads fuel into a receiver aircraft, while both are airborne. Current aerial refueling methods require a pilot in either the tanker or receiver, and a fully automated refueling system would likely require a vision system to correct for inaccuracies in GPS systems at very close range. The refueling problem in this thesis is of the probe-and-drogue type, in which the tanker has a flexible hose attached to a drogue, which stabilizes the hose. The receiver has a probe which it seeks to insert into the hose to receive fuel. In the implementation tested here, there is a ring of LED beacons placed on the drogue. The camera on the receiver detects the beacons, and the vision system uses them in an MEKF to estimate the relative position. The vision system is tested in the presence and absence of GPS signals in a ground test environment. The results of the MEKF are fed into a GNC controller. The controller uses a proportional integral derivative (PID) loop to drive a robotic probe to track the target.

The other application discussed here is a proximity operations problem to capture a rotating spacecraft. A solution to this problem could assist in cleaning up space debris from earth orbit, something which becomes increasingly necessary as more debris piles up in low earth orbit every year. In this problem, the sensing spacecraft seeks to deposit a payload within the rocket nozzle of the target spacecraft. In the lab emulation tests for this problem, the payload is a projectile that is launched from an air cannon on the sensing spacecraft. The projectile carries a tether and performs a soft capture of the target without piercing it. This method is meant to ensure no additional debris is created in the process. The vision system makes use of the known circular shape and size of the target nozzle to estimate the relative position and angular velocity. Once these are known to a certain degree of accuracy, the system sends a signal to deposit the payload. The algorithm is carried out in a lab emulation performed at the Land, Air, and Space Robotics (LASR) Lab at Texas A&M University, and the payload is deposited using a test air cannon. In this experiment, the camera is the only sensing device.

The thesis is organized as follows. Chapter 2 discusses the mathematical models of image formation using a monocular camera. The next three chapters correspond to the three projects discussed. Chapter 3 discusses the problem of planetary EDL with unknown terrain, chapter 4 discusses automated aerial refueling, and chapter 5 discusses spacecraft capture operations. Each of these sections discusses the problem, demonstrates the mathematical formulation of the solution, and concludes with applications and corresponding results.

2. IMAGE FORMATION MODEL

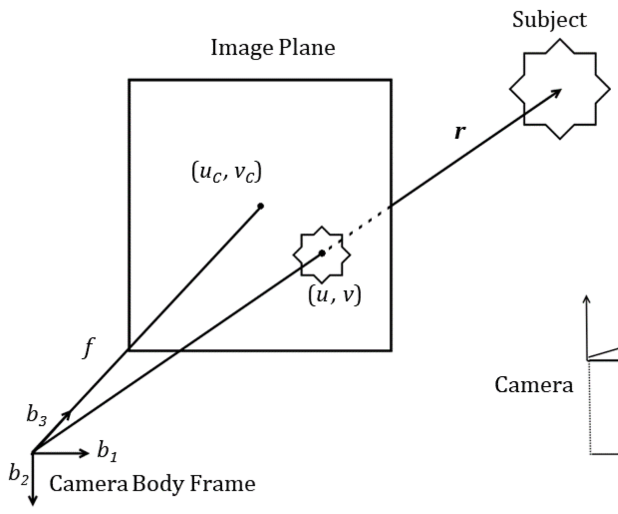
2.1 Pinhole Camera Model

The analysis in this thesis utilizes a pinhole camera model. This means the camera aperture is approximated as an infinitesimal point, and the lens is neglected. In reality, cameras use a lens to focus the light to form an image. This allows more light to enter the camera than would be the case in a pinhole camera, but it has the detrimental effect of causing distortion of the image, especially near the edges. This distortion will be ignored, which is an adequate approximation of the real situation.

The pinhole camera model is illustrated in Fig. 2.1. In this model, a ray of light from the photographed object passes through the pinhole, which is located at the center of the camera's reference frame. The figure shows a focal plane in front of the camera for simplicity, but in reality the plane is behind the camera. The geometry makes either representation equivalent, except that putting the plane in front of the camera results in an image that is right-side-up, in contrast to the upside-down image on the actual focal plane. The ray of light intersects the focal plane at a certain point and the image is captured. The distance from the pinhole to the image plane is called the focal length, and is denoted f.

The camera's body axes are chosen for consistency with the convention for the labeling of pixels in an image. Consequently, as shown in Fig. 2.1, the x-axis points to the right of the image plane, the y-axis points to the bottom of the image plane, and the z-axis completes the triad by pointing along the camera's line-of-sight. Another convention is that the pixels are numbered from the top-left of the image, rather than the center. Thus the optical center $\boldsymbol{u_c} = \begin{bmatrix} u_c & v_c \end{bmatrix}^T$ must be introduced. Together, the focal length f and the optical center $\boldsymbol{u_c}$ are referred to as the camera intrinsics. Both quantities are in units of pixels.

Consider a point in object-space, $\boldsymbol{r} = \begin{bmatrix} x & y & z \end{bmatrix}^T$, and its image-space projection $\boldsymbol{u} = \begin{bmatrix} u & v \end{bmatrix}^T$. By examining the similar-triangles shown in the planar view of Fig. 2.1, as well as



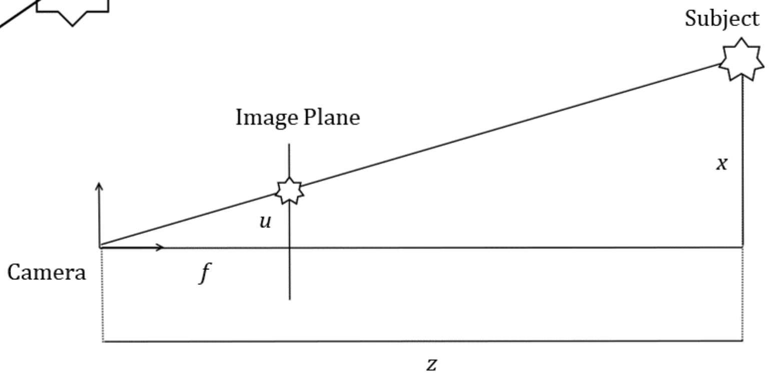


Figure 2.1: (Left): diagram of pinhole camera model, showing the projection of the subject onto the image plane. The image plane is actually behind the camera, but it is shown in front for simplicity. (Right): a side view of the pinhole camera model, which reveals the relationship between the projection coordinates u, f and the true coordinates x, z. A similar relationship exists for the y-axis.

a corresponding view for the y-axis, the following transformation can be derived:

$$\begin{bmatrix} u \\ v \end{bmatrix} = \frac{f}{z} \begin{bmatrix} x \\ y \end{bmatrix} + \begin{bmatrix} u_c \\ v_c \end{bmatrix} \tag{2.1}$$

Here, $\boldsymbol{r}$ is componentiated in the camera's body axes. This nonlinear transformation from object-space to image-space will be denoted by the function $\boldsymbol{h} : \mathbb{R}^{3\times1} \rightarrow \mathbb{R}^{2\times1}$, so that (2.1) can be written

$$\boldsymbol{u} = \boldsymbol{h}(\boldsymbol{r}) \tag{2.2}$$

2.2 Vector and Matrix Notation

Throughout the remainder of the thesis, it will be important to note which reference frame each vector is componentiated in. We will use the following convention. Position vectors are represented in the form $\boldsymbol{r}_{A/B}$, where the subscript A/B denotes the position of the A frame relative to the B frame. All vectors used represent column matrices that are componentiated

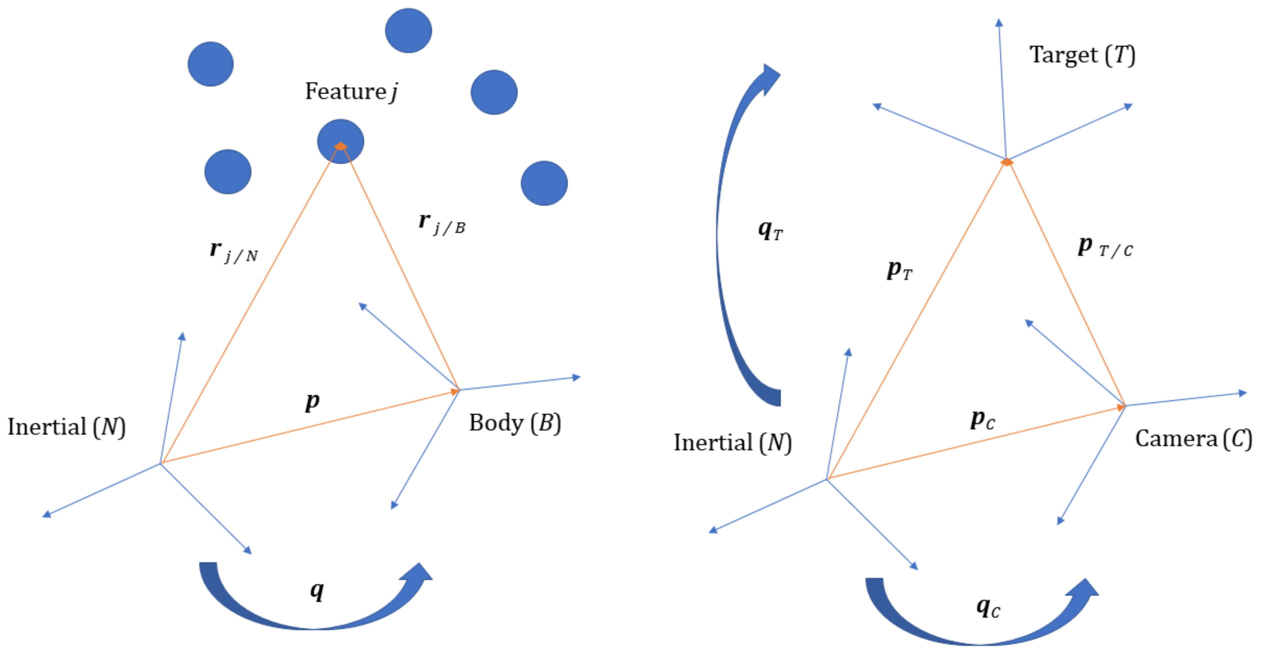


Figure 2.2: Two naming conventions used in this thesis. Note that sometimes the subscripts are dropped for quantities measured with respect to the inertial frame N. (Left): treating features as individual items labeled 1 through m. Used in Chapter 3. (Right): treating features in the aggregate as a target T. Used in Chapters 4 and 5.

in the basis of the latter frame. In this example, then, $\boldsymbol{p}_{A/B}$ is componentiated in the B frame. Similarly, the transformation matrix $C_{A/B}$ denotes the direction cosine matrix (DCM) that transforms vectors into the A frame from the B frame.

In most projects from this thesis, rotations are recorded as quaternions. The quaternion representing the rotation to A from B is written $\boldsymbol{q}_{A/B}$. When the DCM corresponding to this quaternion is needed, it can be written as $C(\boldsymbol{q}_{A/B})$. This DCM is the same as $C_{A/B}$, but using the functional notation emphasizes that it is determined by the quaternion.

The typical model used in this thesis' projects is illustrated in Fig. 2.2. On the left, the features are considered individually and referred to with indices $1, \ldots, m$. This is used for the EDL project in chapter 3, where the features are extracted automatically from images of the terrain. For that project, the camera is mounted on a landing vehicle whose body axes B correspond to the camera's axes. On the right, the features are points on a target object with body frame T. The camera is mounted on a separate body whose body frame is

Project	Known	Unknown
Planetary EDL	Acceleration of B Angular velocity of B	Structure of features Feature positions Pose of B
Aerial Refueling	Acceleration of C and T Angular velocity of C and T Structure of features (LED ring)	Pose of T relative to C
Spacecraft Capture	Structure of features (circular nozzle)	Pose of T relative to C Angular velocity of T

Table 2.1: Summary of projects. The acceleration and angular velocity are considered "known" when IMU measurements are available.

equivalent to the camera frame C. As shown in the figure, if no starting reference frame is specified, quantities are assumed to be measured from the inertial frame N.

The basic known and unknown quantities of the three problems are tabulated in Table 2.1.

3. PLANETARY EDL WITH UNKNOWN TERRAIN[*]

The goal of this formulation is to estimate the position and orientation history of a vehicle's body frame, B, as it moves relative to a fixed terrain frame, N. This arrangement is shown in Fig. 3.1. There are two sources of information that will be used to achieve this goal. An inertial measurement unit (IMU) that consists of a 3-axis accelerometer and a 3-axis rate gyroscope and a monocular camera. Both the IMU and the camera will be considered to be attached to the vehicle. Since these sensors are calibrated with respect to the vehicle frame, their axes are equivalent to the axes of B. The integration of the inertial sensor measurements with the monocular camera feature based processing will be carried out by an EKF. The IMU measurements will be used for propagation using a standard 3D kinematics model of the vehicle, and the features obtained from the camera images will be used for a Kalman update. The IMU measurements are considered to be from a 3-axis accelerometer and a 3-axis gyroscope. These will both be modeled with a stochastic bias, which is a Markov process. In discrete time, the bias process is an independent increment process that can be modeled using a Gaussian random variable with a zero mean and a prescribed standard deviation [9]. Since the biases are modeled as a first order Markov process, it is required that the estimates for the IMU biases also be tracked and updated by the EKF.

The use of camera images in the estimator necessitates having a list of features detected in the image which represent objects at known 3D positions. Since these feature positions are not known *a priori*, they must be estimated before performing any Kalman updates. The estimator developed in this paper has two distinct sub-problems which will be solved separately. The solutions are merged to realize an EKF formulation. The first problem is the estimation of 3D feature positions, assuming that the inertial sensors have not drifted

[*]Part of this chapter reprinted with permission from "Vision and inertial sensor fusion for terrain relative navigation" by A. Verras, R. Eapen, A. Simon, M. Majji, R. Bhaskara, C. Restrepo, and R. Lovelace, 2021. *2021 AIAA SciTech Forum*, Copyright 2021 by the authors of the paper.

significantly (phase 1). The second problem is estimation of the vehicle's position and orientation using an EKF (phase 2). Since the camera is monocular, multiple images are required in order to estimate 3D feature positions, making use of the baseline incurred by the motion of the camera. This is known as motion stereo [10]. However, since the Kalman update step cannot be performed until after the feature 3D positions have been estimated, the estimated states will become less and less accurate the longer the estimator takes to estimate 3D feature positions due to the accumulation of biases in the inertial sensors. In particular, there will be no updates to the IMU biases, so the measurements will become less accurate with time. Thus, a balance must be struck in how much time to spend estimating the 3D feature positions. The stopping point for this part of the problem (phase 1) will be scene dependent. Possible criteria to use as a stopping condition include the processing of a certain number of images, or detecting a certain displacement threshold between the first and last images. The latter consideration ensures that there is enough variation for the monocular camera to obtain a depth fix.

The main loop of the estimation algorithm is estimated in this work as follows. A feature extraction algorithm is run on the first camera image, which is termed the keyframe. The detected features are tracked over the next few images until the enough images have been obtained for phase 1. The body states are propagated over this set of images, and the estimated states are then used along with the image-space feature positions to estimate the 3D feature positions, which completes phase 1. After that, the feature tracking will continue as the full EKF provides estimates for the state vector. Once the stopping point for phase 2 is reached, which may be a threshold for the number of feature tracks remaining, a new keyframe is chosen and the process is repeated.

3.1 Tracked States

The states used by the estimator to propagate the kinematics between any two image updates are written as

$$\boldsymbol{x} = \begin{bmatrix} \boldsymbol{q}^T & \boldsymbol{p}^T & \boldsymbol{v}^T & \boldsymbol{\beta}_g^T & \boldsymbol{\beta}_a^T \end{bmatrix}^T \tag{3.1}$$

The unit-norm quaternion $\boldsymbol{q}$ is the orientation of the body frame relative to the terrain frame, so that the corresponding DCM, $C(\boldsymbol{q})$, transforms to N from B. The position $\boldsymbol{p}$ and velocity $\boldsymbol{v}$ are reported in the terrain frame, and the gyroscope and accelerometer biases, $\boldsymbol{\beta}_g$ and $\boldsymbol{\beta}_a$ respectively, are in the body frame. Thus $\boldsymbol{x} \in \mathbb{R}^{16\times 1}$.

A multiplicative extended Kalman filter (MEKF) is implemented in this paper. The filter is "multiplicative" because the orientation state is tracked as a quaternion with a multiplicative error factor $\boldsymbol{\delta q}$. Namely, the true quaternion $\boldsymbol{q}$ is related to the estimated quaternion $\hat{\boldsymbol{q}}$ by

$$\boldsymbol{q} = \boldsymbol{\delta q} \otimes \hat{\boldsymbol{q}} \tag{3.2}$$

Note that $\otimes$ represents quaternion multiplication using the argument order convention common to multiplicative EKF attitude estimators, which is contrary to the Hamiltonian convention [11, 9]. Using a multipliciative error permits the error quaternion itself to be normalized. In contrast, an additive error quaternion would be close to zero, and thus would not satisfy the unit-norm constraint.

Since the quaternion is not a minimal representation of the orientation, its components are interdependent. This leads to a non-minimal state space [9] (recall minimality implies both controllability and observability). If the 16-by-16 state covariance corresponding to the 16 element state vector $\boldsymbol{x}$ was tracked, this interpendence would lead to a singular covariance matrix. To alleviate this concern, a 15-by-15 state covariance matrix P will be tracked instead. Fortunately, the error quaternion $\boldsymbol{\delta q}$ lends itself to a simple 3 element representation when a small-angle approximation is applied. The error quaternion is made up of a vector part $\boldsymbol{\rho}$ and a scalar part q_4. Namely,

$$\boldsymbol{\delta q} = \begin{bmatrix} \boldsymbol{\rho} \\ q_4 \end{bmatrix} \tag{3.3}$$

Assuming this quaternion is close to the identity, its components can be approximated as

$$\boldsymbol{\delta\rho} \approx \frac{\boldsymbol{\alpha}}{2} \qquad \text{and} \qquad \delta q_4 \approx 1 \tag{3.4}$$

where $\boldsymbol{\alpha}$ is the vector of small roll-pitch-yaw Euler angles $\boldsymbol{\alpha} = \begin{bmatrix} \phi & \theta & \psi \end{bmatrix}^T$ associated with the error quaternion $\boldsymbol{\delta q}$. Since $\boldsymbol{\alpha}$ is a minimal representation of the orientation error, it is a good candidate to replace $\boldsymbol{\delta q}$ in the error state vector $\boldsymbol{\Delta x}$. Thus the error state will be

$$\boldsymbol{\Delta x} = \begin{bmatrix} \boldsymbol{\alpha}^T & \boldsymbol{\Delta p}^T & \boldsymbol{\Delta v}^T & \boldsymbol{\Delta\beta}_g^T & \boldsymbol{\Delta\beta}_a^T \end{bmatrix}^T \tag{3.5}$$

where the additive errors are $\boldsymbol{\Delta p} = \boldsymbol{p} - \hat{\boldsymbol{p}}$, $\boldsymbol{\Delta v} = \boldsymbol{v} - \hat{\boldsymbol{v}}$, $\boldsymbol{\Delta\beta}_g = \boldsymbol{\beta}_g - \hat{\boldsymbol{\beta}}_g$, and $\boldsymbol{\Delta\beta}_a = \boldsymbol{\beta}_a - \hat{\boldsymbol{\beta}}_a$. Then $\boldsymbol{\Delta x} \in \mathbb{R}^{15\times 1}$. It can be seen that the mean of $\boldsymbol{\Delta x}$ is zero. I.e.,

$$E\left[\boldsymbol{\Delta x}\right] = \mathbf{0} \tag{3.6}$$

where $E[\cdot]$ denotes the probabilistic expectation operator. Furthermore, the covariance matrix $P \in \mathbb{R}^{15\times 15}$ can be defined as

$$P = E\left[\boldsymbol{\Delta x}\boldsymbol{\Delta x}^T\right] \tag{3.7}$$

In addition to alleviating the singularity issues associated with a nonminimal covariance, defining P this way also has the benefit of providing a physically meaningful orientation covariance, since its orientation components can be interpreted as errors in roll-pitch-yaw Euler angles.

The 16 element state vector $\boldsymbol{x}$, defined in (3.1), will be utilized in the propagation step of the MEKF, while the 15 element error state vector $\boldsymbol{\Delta x}$, defined in (3.5) will be utilized

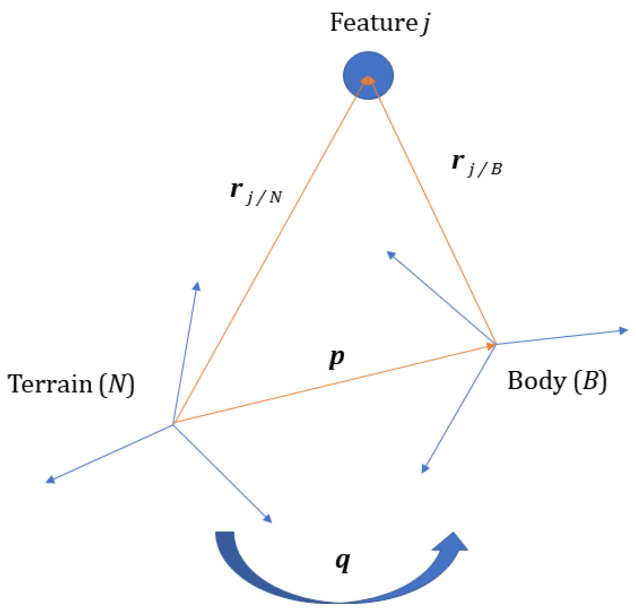


Figure 3.1: Position of feature j relative to terrain frame ($\boldsymbol{r}_{j/N}$) and relative to body frame ($\boldsymbol{r}_{j/B}$). The position and orientation of the body relative to the terrain are given by $\boldsymbol{p}$ and $\boldsymbol{q}$, respectively.

in the update step, along with its covariance P.

One more small-angle approximation that will be applied to the error quaternion involves approximating the covariance matrix $C(\boldsymbol{\delta q})$ as

$$C(\boldsymbol{\delta q}) \approx I - [\boldsymbol{\alpha}\times] \tag{3.8}$$

where $[\boldsymbol{\alpha}\times]$ denotes the skew-symmetric cross-product matrix of $\boldsymbol{\alpha}$.

3.2 Measurement Model

3.2.1 Camera Measurement Model

Since the camera is attached to the body frame, the camera measurement model will depend on the feature's position relative to the body frame. Consider the j^{th} feature. From Fig. 3.1, its body frame position is given by

$$\boldsymbol{r}_{j/B} = C(\boldsymbol{q})(\boldsymbol{r}_{j/N} - \boldsymbol{p}) \tag{3.9}$$

This relative position vector is componentiated as

$$\boldsymbol{r}_{j/B} = \begin{bmatrix} x_{j/B} \\ y_{j/B} \\ z_{j/B} \end{bmatrix} \tag{3.10}$$

The pinhole camera model from (2.2) is used to give:

$$\begin{bmatrix} u_j \\ v_j \end{bmatrix} = \boldsymbol{h}\left(\boldsymbol{r}_{j/B}\right) \tag{3.11}$$

The camera measurement model is

$$\begin{bmatrix} \tilde{u}_j \\ \tilde{v}_j \end{bmatrix} = \begin{bmatrix} u_j \\ v_j \end{bmatrix} + \begin{bmatrix} \nu_{u_j} \\ \nu_{v_j} \end{bmatrix} \tag{3.12}$$

where ν_{u_j} and ν_{v_j} are zero-mean Gaussian random variables with variances $\sigma^2_{u_j}$ and $\sigma^2_{v_j}$, respectively.

The estimation model is

$$\begin{bmatrix} \hat{u}_j \\ \hat{v}_j \end{bmatrix} = \boldsymbol{h}\left(\hat{\boldsymbol{r}}_{j/B}\right) \tag{3.13}$$

$$\hat{\boldsymbol{r}}_{j/B} = C(\hat{\boldsymbol{q}})(\hat{\boldsymbol{r}}_{j/N} - \hat{\boldsymbol{p}}) \tag{3.14}$$

and the image space position errors are denoted $\Delta u_j = \tilde{u}_j - \hat{u}_j$ and $\Delta v_j = \tilde{v}_j - \hat{v}_j$.

3.2.1.1 Camera Measurement Covariance Analysis

In addition to the additive camera measurement errors in (3.12), additional errors occur due to the inaccurate knowledge of the state and feature positions. The error in the estimate of the j^{th} feature will be denoted $\Delta\boldsymbol{r}_{j/N} = \boldsymbol{r}_{j/N} - \hat{\boldsymbol{r}}_{j/N}$. Expanding (3.9) in terms of estimated quantities and errors,

$$\boldsymbol{r}_{j/B} = C(\boldsymbol{\delta q})C(\hat{\boldsymbol{q}})(\hat{\boldsymbol{r}}_{j/N} + \boldsymbol{\Delta r}_{j/N} - \hat{\boldsymbol{p}} - \boldsymbol{\Delta p}) \tag{3.15}$$

Introducing the approximation for $C(\boldsymbol{\delta q})$ from (3.8) and eliminating second-order error terms,

$$\begin{aligned}\boldsymbol{r}_{j/B} &= (I - [\boldsymbol{\alpha}\times])\, C(\hat{\boldsymbol{q}})(\hat{\boldsymbol{r}}_{j/N} + \boldsymbol{\Delta r}_{j/N} - \hat{\boldsymbol{p}} - \boldsymbol{\Delta p}) \\ &= (I - [\boldsymbol{\alpha}\times])\, C(\hat{\boldsymbol{q}})(\hat{\boldsymbol{r}}_{j/N} - \hat{\boldsymbol{p}}) + C(\hat{\boldsymbol{q}})(\boldsymbol{\Delta r}_{j/N} - \boldsymbol{\Delta p})\end{aligned} \tag{3.16}$$

Re-introducing $\hat{\boldsymbol{r}}_{j/B}$ using (3.14) and distributing,

$$\begin{aligned}\boldsymbol{r}_{j/B} &= \hat{\boldsymbol{r}}_{j/B} - [\boldsymbol{\alpha}\times]\, \hat{\boldsymbol{r}}_{j/B} + C(\hat{\boldsymbol{q}})\boldsymbol{\Delta r}_{j/N} - C(\hat{\boldsymbol{q}})\boldsymbol{\Delta p} \\ &= \hat{\boldsymbol{r}}_{j/B} + \left[\hat{\boldsymbol{r}}_{j/B}\times\right] \boldsymbol{\alpha} + C(\hat{\boldsymbol{q}})\boldsymbol{\Delta r}_{j/N} - C(\hat{\boldsymbol{q}})\boldsymbol{\Delta p}\end{aligned} \tag{3.17}$$

This gives a linear expression for the relative position error of the j^{th} feature

$$\boldsymbol{\Delta r}_{j/B} = \left[\hat{\boldsymbol{r}}_{j/B}\times\right] \boldsymbol{\alpha} + C(\hat{\boldsymbol{q}})\boldsymbol{\Delta r}_{j/N} - C(\hat{\boldsymbol{q}})\boldsymbol{\Delta p} \tag{3.18}$$

Therefore the covariance of the relative position error is

$$E[\boldsymbol{\Delta r}_{j/B}\boldsymbol{\Delta r}_{j/B}^T] = \left[\hat{\boldsymbol{r}}_{j/B}\times\right] P_{\boldsymbol{\alpha}} \left[\hat{\boldsymbol{r}}_{j/B}\times\right]^T + C(\hat{\boldsymbol{q}}) \left(P_{\boldsymbol{r}_{j/N}} + P_{\boldsymbol{p}}\right) C(\hat{\boldsymbol{q}})^T \tag{3.19}$$

$P_{\boldsymbol{\alpha}}$ and $P_{\boldsymbol{p}}$ are the covariances of the orientation and postion staes, respectively. They can be obtained by isolating the appropriate 3-by-3 blocks from the state covariance P. $P_{\boldsymbol{r}_{j/N}}$ is the covariance of the feature position estimate, which will be discussed in the section about obtaining the estimate. The camera measurement error can then be found from (3.12) to be

$$\begin{bmatrix} \Delta u_j \\ \Delta v_j \end{bmatrix} = \begin{bmatrix} u_j - \hat{u}_j \\ v_j - \hat{v}_j \end{bmatrix} + \begin{bmatrix} \nu_{u_j} \\ \nu_{v_j} \end{bmatrix} = \frac{\partial \boldsymbol{h}}{\partial \hat{\boldsymbol{r}}_{j/B}} \boldsymbol{\Delta r}_{j/B} + \begin{bmatrix} \nu_{u_j} \\ \nu_{v_j} \end{bmatrix} \tag{3.20}$$

where the last equality expanded (3.11) to first-order about the estimate.

Therefore, the covariance of the image-space measurement is

$$E\left[\begin{bmatrix} \Delta u_j \\ \Delta v_j \end{bmatrix} \begin{bmatrix} \Delta u_j & \Delta v_j \end{bmatrix}\right] = \frac{\partial \boldsymbol{h}}{\partial \hat{\boldsymbol{r}}_{j/B}} E[\boldsymbol{\Delta r}_{j/B} \boldsymbol{\Delta r}_{j/B}^T] \frac{\partial \boldsymbol{h}}{\partial \hat{\boldsymbol{r}}_{j/B}}^T + \begin{bmatrix} \sigma_{u_j}^2 & 0 \\ 0 & \sigma_{v_j}^2 \end{bmatrix} \tag{3.21}$$

so the covariance matrix for the full set of camera measurments is

$$S = \begin{bmatrix} \frac{\partial \boldsymbol{h}}{\partial \hat{\boldsymbol{r}}_{1/B}} E[\boldsymbol{\Delta r}_{1/B} \boldsymbol{\Delta r}_{1/B}^T] \frac{\partial \boldsymbol{h}}{\partial \hat{\boldsymbol{r}}_{1/B}}^T & & 0 \\ & \ddots & \\ 0 & & \frac{\partial \boldsymbol{h}}{\partial \hat{\boldsymbol{r}}_{m/B}} E[\boldsymbol{\Delta r}_{m/B} \boldsymbol{\Delta r}_{m/B}^T] \frac{\partial \boldsymbol{h}}{\partial \hat{\boldsymbol{r}}_{m/B}}^T \end{bmatrix} + R \tag{3.22}$$

where R is the camera noise matrix

$$R = \begin{bmatrix} \sigma_{u_1}^2 & & & & \\ & \sigma_{v_1}^2 & & 0 & \\ & & \ddots & & \\ & 0 & & \sigma_{u_m}^2 & \\ & & & & \sigma_{v_m}^2 \end{bmatrix} \tag{3.23}$$

Note that the derivative of $\boldsymbol{h}$ evaluates to the 2-by-3 matrix

$$\frac{\partial \boldsymbol{h}}{\partial \boldsymbol{r}_{j/B}} = \frac{f}{z_{j/B}^2} \begin{bmatrix} z_{j/B} & 0 & -x_{j/B} \\ 0 & z_{j/B} & -y_{j/B} \end{bmatrix} \tag{3.24}$$

and that $S \in \mathbb{R}^{2m \times 2m}$.

3.2.2 IMU Measurement Model

The gyro and accelerometer are modeled with stochastic biases $\boldsymbol{\beta_g}$ and $\boldsymbol{\beta_a}$. The measurement models are

$$\boldsymbol{\omega} = \tilde{\boldsymbol{\omega}} - \boldsymbol{\beta}_g - \boldsymbol{\eta}_{gv} \qquad \boldsymbol{a} = \tilde{\boldsymbol{a}} - \boldsymbol{\beta}_a - \boldsymbol{\eta}_{av}$$

$$\dot{\boldsymbol{\beta}}_g = \boldsymbol{\eta}_{gu} \qquad \dot{\boldsymbol{\beta}}_a = \boldsymbol{\eta}_{au} \tag{3.25}$$

where $\boldsymbol{\omega}$ and $\boldsymbol{a}$ are the angular velocity and acceleration, respectively, of the body frame relative to the terrain frame, reported in the body frame; the tilde denotes measurements; and $\boldsymbol{\eta_{gv}}$, $\boldsymbol{\eta_{gu}}$, $\boldsymbol{\eta_{av}}$, and $\boldsymbol{\eta_{au}}$ are zero-mean Gaussian random variables with variances $\sigma_{gv}^2 I$, $\sigma_{gu}^2 I$, $\sigma_{av}^2 I$, and $\sigma_{au}^2 I$, respectively.

The associated estimation models are given by

$$\hat{\boldsymbol{\omega}} = \tilde{\boldsymbol{\omega}} - \hat{\boldsymbol{\beta}}_g \qquad \hat{\boldsymbol{a}} = \tilde{\boldsymbol{a}} - \hat{\boldsymbol{\beta}}_a$$

$$\dot{\hat{\boldsymbol{\beta}}}_g = \mathbf{0} \qquad \dot{\hat{\boldsymbol{\beta}}}_a = \mathbf{0} \tag{3.26}$$

3.3 State Propagation

The IMU measurements will be used as an estimated angular velocity $\hat{\boldsymbol{\omega}}$ and acceleration $\hat{\boldsymbol{a}}$. The states in $\boldsymbol{x}$ can then be propagated according to

$$
\begin{aligned}
\dot{\hat{\boldsymbol{q}}}(t) &= \frac{1}{2}\Xi\left(\hat{\boldsymbol{q}}(t)\right)\hat{\boldsymbol{\omega}}(t) \\
\dot{\hat{\boldsymbol{p}}}(t) &= \hat{\boldsymbol{v}}(t) \\
\dot{\hat{\boldsymbol{v}}}(t) &= C^T\left(\hat{\boldsymbol{q}}(t)\right)\hat{\boldsymbol{a}}(t) \\
\dot{\hat{\boldsymbol{\beta}}}_g(t) &= \mathbf{0} \\
\dot{\hat{\boldsymbol{\beta}}}_a(t) &= \mathbf{0}
\end{aligned}
\tag{3.27}
$$

where the DCM $C(\boldsymbol{q})$ for a quaternion $\boldsymbol{q} = \begin{bmatrix}\boldsymbol{\rho}^T & q_4\end{bmatrix}^T$ is

$$
C(\boldsymbol{q}) = \Xi^T(\boldsymbol{q})\Psi(\boldsymbol{q}) \tag{3.28}
$$

for

$$
\Xi(\boldsymbol{q}) = \begin{bmatrix} q_4 I + [\boldsymbol{\rho}\times] \\ -\boldsymbol{\rho}^T \end{bmatrix} \qquad \text{and} \qquad \Psi(\boldsymbol{q}) = \begin{bmatrix} q_4 I - [\boldsymbol{\rho}\times] \\ -\boldsymbol{\rho}^T \end{bmatrix} \tag{3.29}
$$

As mentioned above, the covariance P is written in terms of the error states $\boldsymbol{\Delta x}$. Determining the covariance dynamics thus relies on the dynamics of $\boldsymbol{\Delta x}$. To that end, the quaternion error kinematics can be written

$$
\boldsymbol{\delta}\dot{\boldsymbol{q}} = \frac{1}{2}\left(\begin{bmatrix}\boldsymbol{\omega} \\ 0\end{bmatrix} \otimes \boldsymbol{\delta q} - \boldsymbol{\delta q} \otimes \begin{bmatrix}\hat{\boldsymbol{\omega}} \\ 0\end{bmatrix}\right) \tag{3.30}
$$

Written in terms of the angular velocity error $\boldsymbol{\Delta\omega}$,

$$
\boldsymbol{\delta}\dot{\boldsymbol{q}} = \frac{1}{2}\left(\begin{bmatrix}\hat{\boldsymbol{\omega}} \\ 0\end{bmatrix} \otimes \boldsymbol{\delta q} - \boldsymbol{\delta q} \otimes \begin{bmatrix}\hat{\boldsymbol{\omega}} \\ 0\end{bmatrix}\right) + \frac{1}{2}\begin{bmatrix}\boldsymbol{\Delta\omega} \\ 0\end{bmatrix} \otimes \boldsymbol{\delta q} \tag{3.31}
$$

Substituting in the identities

$$\begin{bmatrix} \hat{\boldsymbol{\omega}} \\ 0 \end{bmatrix} \otimes \boldsymbol{\delta q} = \begin{bmatrix} -[\hat{\boldsymbol{\omega}}\times] & \hat{\boldsymbol{\omega}} \\ -\hat{\boldsymbol{\omega}}^T & 0 \end{bmatrix} \boldsymbol{\delta q} \qquad \text{and} \qquad \boldsymbol{\delta q} \otimes \begin{bmatrix} \hat{\boldsymbol{\omega}} \\ 0 \end{bmatrix} = \begin{bmatrix} [\hat{\boldsymbol{\omega}}\times] & \hat{\boldsymbol{\omega}} \\ -\hat{\boldsymbol{\omega}}^T & 0 \end{bmatrix} \boldsymbol{\delta q} \tag{3.32}$$

one obtains

$$\boldsymbol{\delta \dot{q}} = -\begin{bmatrix} [\hat{\boldsymbol{\omega}}\times]\,\boldsymbol{\delta\rho} \\ 0 \end{bmatrix} + \frac{1}{2}\begin{bmatrix} \boldsymbol{\Delta\omega} \\ 0 \end{bmatrix} \otimes \boldsymbol{\delta q} \tag{3.33}$$

The second term can be linearized by assuming $\boldsymbol{\delta q}$ is close to identity, and thus

$$\boldsymbol{\delta \dot{q}} \approx -\begin{bmatrix} [\hat{\boldsymbol{\omega}}\times]\,\boldsymbol{\delta\rho} \\ 0 \end{bmatrix} + \frac{1}{2}\begin{bmatrix} \boldsymbol{\Delta\omega} \\ 0 \end{bmatrix} \tag{3.34}$$

This indicates q_4 is approximately constant and can thus be dropped. Doing so, as well as substituting $\boldsymbol{\Delta\omega} = \boldsymbol{\omega} - \hat{\boldsymbol{\omega}} = -\left(\boldsymbol{\Delta\beta}_g + \boldsymbol{\eta}_{gv}\right)$ from (3.25) and (3.26),

$$\boldsymbol{\delta\dot{\rho}} = -[\hat{\boldsymbol{\omega}}\times]\,\boldsymbol{\delta\rho} - \frac{1}{2}\left(\boldsymbol{\Delta\beta}_g + \boldsymbol{\eta}_{gv}\right) \tag{3.35}$$

Replacing $\boldsymbol{\delta\rho}$ with $\boldsymbol{\delta\alpha}/2$ as in (3.4), this can finally be written

$$\boldsymbol{\delta\dot{\alpha}} = -[\hat{\boldsymbol{\omega}}\times]\,\boldsymbol{\delta\alpha} - \left(\boldsymbol{\Delta\beta}_g + \boldsymbol{\eta}_{gv}\right) \tag{3.36}$$

The position error dynamics are simply

$$\boldsymbol{\Delta\dot{p}} = \boldsymbol{\Delta v} \tag{3.37}$$

For velocity,

$$
\begin{aligned}
\boldsymbol{\Delta}\dot{\boldsymbol{v}} &= \dot{\boldsymbol{v}} - \dot{\hat{\boldsymbol{v}}} = C^T(\boldsymbol{q})\boldsymbol{a} - C^T(\hat{\boldsymbol{q}})\hat{\boldsymbol{a}} \\
&= C^T(\boldsymbol{\delta q})C^T(\hat{\boldsymbol{q}})(\hat{\boldsymbol{a}} + \boldsymbol{\Delta a}) - C^T(\hat{\boldsymbol{q}})\hat{\boldsymbol{a}} \\
&= \left(C^T(\boldsymbol{\delta q}) - I\right) C^T(\hat{\boldsymbol{q}})\hat{\boldsymbol{a}} + C^T(\boldsymbol{\delta q})C^T(\hat{\boldsymbol{q}})\boldsymbol{\Delta a} \qquad (3.38)
\end{aligned}
$$

where $\boldsymbol{\Delta a} = \boldsymbol{a} - \hat{\boldsymbol{a}}$. To linearize dynamics, $\boldsymbol{\delta q}$ can be assumed to be identity. Doing this, as well as replacing $\boldsymbol{\Delta a}$ using (3.25) and (3.26),

$$
\boldsymbol{\Delta}\dot{\boldsymbol{v}} = -C^T(\hat{\boldsymbol{q}})\left(\boldsymbol{\Delta\beta}_a + \boldsymbol{\eta}_{av}\right) \qquad (3.39)
$$

Finally, the IMU bias dynamics follow from (3.25) and (3.26)

$$
\boldsymbol{\Delta}\dot{\boldsymbol{\beta}}_g = \boldsymbol{\eta}_{gu} \qquad \text{and} \qquad \boldsymbol{\Delta}\dot{\boldsymbol{\beta}}_a = \boldsymbol{\eta}_{au} \qquad (3.40)
$$

Defining the IMU error vector $\boldsymbol{w} = \begin{bmatrix} \boldsymbol{\eta}_{gv}^T & \boldsymbol{\eta}_{gu}^T & \boldsymbol{\eta}_{av}^T & \boldsymbol{\eta}_{au}^T \end{bmatrix}^T$, (3.36) through (3.40) can be written compactly as

$$
\boldsymbol{\Delta}\dot{\boldsymbol{x}}(t) = F(t)\boldsymbol{\Delta x}(t) + G(t)\boldsymbol{w}(t) \qquad (3.41)
$$

where

$$
F(t) = \begin{bmatrix} -\left[\hat{\boldsymbol{\omega}}(t)\times\right] & 0 & 0 & -I_3 & 0 \\ 0 & 0 & I_3 & 0 & 0 \\ 0 & 0 & 0 & 0 & -C^T\left(\hat{\boldsymbol{q}}(t)\right) \\ 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 \end{bmatrix} \qquad \text{and} \qquad G(t) = \begin{bmatrix} -I_3 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & -C^T\left(\hat{\boldsymbol{q}}(t)\right) & 0 \\ 0 & I_3 & 0 & 0 \\ 0 & 0 & 0 & I_3 \end{bmatrix} \qquad (3.42)
$$

Note that each 0 in (3.42) denotes the 3-by-3 zero matrix. These matrices are used to propagate the state error covariance P:

$$\dot{P}(t) = F(t)P(t) + P(t)F^T(t) + G(t)Q(t)G^T(t) \tag{3.43}$$

where Q is the covariance of $\boldsymbol{w}$. I.e.,

$$Q(t) = \begin{bmatrix} \sigma_{gv}^2 I_3 & 0 & 0 & 0 \\ 0 & \sigma_{gu}^2 I_3 & 0 & 0 \\ 0 & 0 & \sigma_{av}^2 I_3 & 0 \\ 0 & 0 & 0 & \sigma_{au}^2 I_3 \end{bmatrix} \tag{3.44}$$

Alternatively, the propagation can be formatted discretely. Letting a subscript of k denote estimates and measurements at time t_k and letting $k+1$ denote those at time t_{k+1}, (3.27) can be discretized using a zero-order hold to obtain the propagation

$$\begin{aligned} \hat{\boldsymbol{q}}_{k+1} &= \Phi(\hat{\boldsymbol{\omega}}_k)\hat{\boldsymbol{q}}_k \\ \hat{\boldsymbol{p}}_{k+1} &= \frac{1}{2}C\left(\hat{\boldsymbol{q}}_k\right)^T \hat{\boldsymbol{a}}_k \Delta t^2 + \hat{\boldsymbol{v}}_k \Delta t + \hat{\boldsymbol{p}}_k \\ \hat{\boldsymbol{v}}_{k+1} &= C\left(\hat{\boldsymbol{q}}_k\right)^T \hat{\boldsymbol{a}}_k \Delta t + \hat{\boldsymbol{v}}_k \\ \hat{\boldsymbol{\beta}}_{g,k+1} &= \hat{\boldsymbol{\beta}}_{g,k} \\ \hat{\boldsymbol{\beta}}_{a,k+1} &= \hat{\boldsymbol{\beta}}_{a,k} \end{aligned} \tag{3.45}$$

where $\Delta t = t_{k+1} - t_k$. The state transition matrix $\Phi(\hat{\boldsymbol{\omega}}_k)$ for the quaternion can be obtained by taking the matrix exponential of the continuous-time quaternion dynamics. It evaluates to

$$\Phi(\hat{\boldsymbol{\omega}}_k) = \begin{bmatrix} \cos\left(\frac{1}{2}||\hat{\boldsymbol{\omega}}_k||\Delta t\right) I_3 - \left[\hat{\boldsymbol{\psi}}_k \times\right] & \hat{\boldsymbol{\psi}}_k \\ -\hat{\boldsymbol{\psi}}_k^T & \cos\left(\frac{1}{2}||\hat{\boldsymbol{\omega}}_k||\Delta t\right) \end{bmatrix} \quad \text{with} \quad \hat{\boldsymbol{\psi}}_k = \frac{\sin\left(\frac{1}{2}||\hat{\boldsymbol{\omega}}_k||\Delta t\right) \hat{\boldsymbol{\omega}}_k}{||\hat{\boldsymbol{\omega}}_k||} \tag{3.46}$$

The discrete covariance propagation is

$$P_{k+1} = \exp(F\Delta t) P_k \exp(F\Delta t)^T + G Q_k G^T \tag{3.47}$$

where $\exp(F\Delta t)$ is the matrix exponential of $F\Delta t$, and the discrete IMU covariance Q_k is given by

$$Q_k = \begin{bmatrix} \left(\sigma_{gv}^2 \Delta t + \frac{1}{3}\sigma_{gu}^2 \Delta t^3\right) I_3 & \left(\frac{1}{2}\sigma_{gu}^2 \Delta t^2\right) I_3 & 0 & 0 \\ \left(\frac{1}{2}\sigma_{gu}^2 \Delta t^2\right) I_3 & \left(\sigma_{gu}^2 \Delta t\right) I_3 & 0 & 0 \\ 0 & 0 & \left(\sigma_{av}^2 \Delta t + \frac{1}{3}\sigma_{au}^2 \Delta t^3\right) I_3 & \left(\frac{1}{2}\sigma_{au}^2 \Delta t^2\right) I_3 \\ 0 & 0 & \left(\frac{1}{2}\sigma_{au}^2 \Delta t^2\right) I_3 & \left(\sigma_{au}^2 \Delta t\right) I_3 \end{bmatrix} \tag{3.48}$$

where each 0 represents the 3-by-3 zero matrix.

3.4 Least Squares Estimation of Feature Locations

The first few camera frames are used to estimate the relative locations of the features. The IMU measurements are used for this short period to propagate the state x as described previously. Since this window of time is small, the IMU will not exhibit major errors due to bias instability. These state estimates, along with image space locations for each feature, will allow the feature 3D position estimation to be expressed as a linear least squares problem.

Consider $m + 1$ camera frames, labeled $0, \ldots, m$. The zeroth image will be called the key frame, and the estimated feature positions will be reported in that reference frame. These can then be converted to the terrain-fixed frame using the estimated state x_0 at the keyframe. The propagated states at the time of the i^{th} frame will be denoted $\boldsymbol{x}_i =$

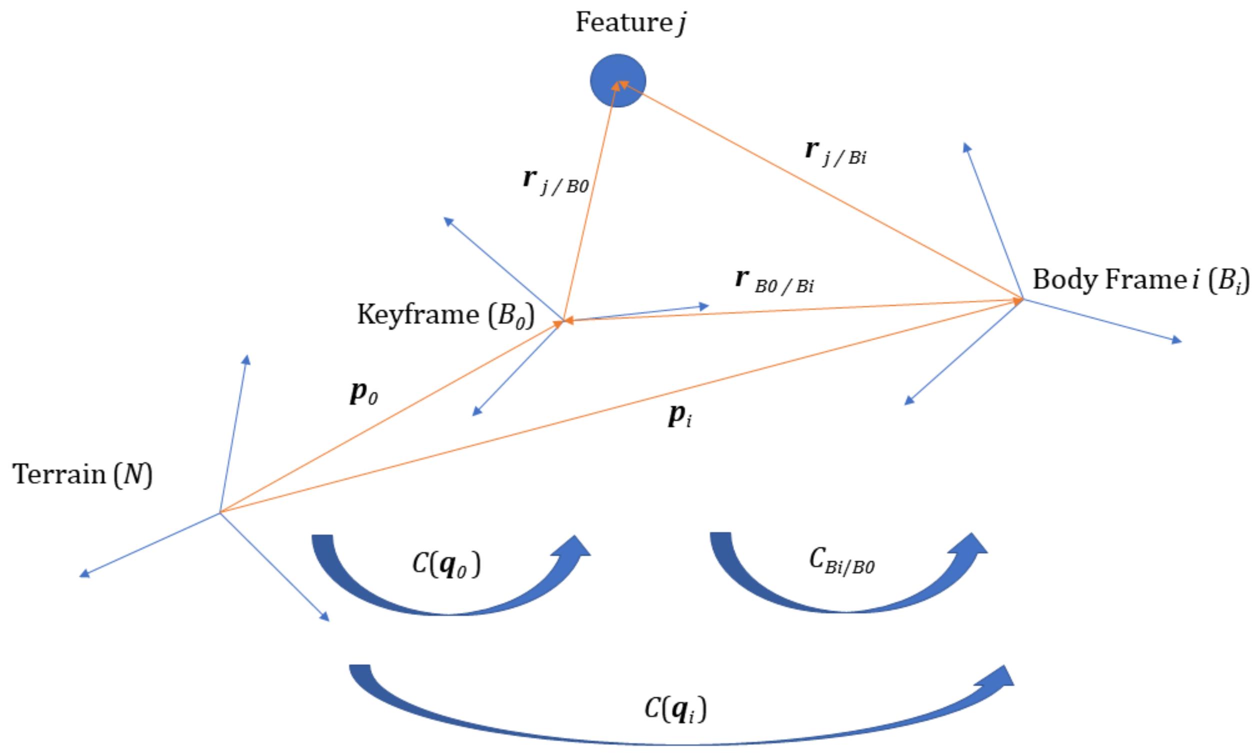


Figure 3.2: Keyframe and body frame i along with their respective lines of sight to the j^{th} feature.

$\begin{bmatrix} \boldsymbol{q}_i^T & \boldsymbol{p}_i^T & \boldsymbol{v}_i^T & \boldsymbol{\beta}_{gi}^T & \boldsymbol{\beta}_{ai}^T \end{bmatrix}^T$. The relative positions and orientations are shown in Fig. (3.2), which reveals the relationship

$$\boldsymbol{r}_{j/B_i} = C_{B_i/B_0}\boldsymbol{r}_{j/B_0} + \boldsymbol{r}_{B_0/B_i} \tag{3.49}$$

As can be seen in Fig. (3.2), the relative translation $\boldsymbol{r}_{B_0/B_i}$ and rotation C_{B_i/B_0} are known from the propagated states. Specifically, $\boldsymbol{r}_{B_0/B_i} = \boldsymbol{p}_0 - \boldsymbol{p}_i$ and $C_{B_i/B_0} = C(\boldsymbol{q}_i)C(\boldsymbol{q}_0)^T$ Expanding (3.49) into its components,

$$\begin{bmatrix} x_{j/B_i} \\ y_{j/B_i} \\ z_{j/B_i} \end{bmatrix} = \begin{bmatrix} c_{11} & c_{12} & c_{13} \\ c_{21} & c_{22} & c_{23} \\ c_{31} & c_{32} & c_{33} \end{bmatrix} \begin{bmatrix} x_{j/B_0} \\ y_{j/B_0} \\ z_{j/B_0} \end{bmatrix} + \begin{bmatrix} x_{B_0/B_i} \\ y_{B_0/B_i} \\ z_{B_0/B_i} \end{bmatrix} \tag{3.50}$$

To simplify the image-space expressions, the dimensionless, centered image-space coordinates

$$\begin{bmatrix} u' \\ v' \end{bmatrix} = \frac{1}{f}\left(\begin{bmatrix} u \\ v \end{bmatrix} - \begin{bmatrix} u_c \\ v_c \end{bmatrix}\right) \tag{3.51}$$

are introduced. Using this convention, the transformation to image-space in (2.1) can be written as

$$\begin{bmatrix} u'_{j/B_0} \\ v'_{j/B_0} \end{bmatrix} = \frac{1}{z_{j/B_0}}\begin{bmatrix} x_{j/B_0} \\ y_{j/B_0} \end{bmatrix} \quad \text{and} \quad \begin{bmatrix} u'_{j/B_i} \\ v'_{j/B_i} \end{bmatrix} = \frac{1}{z_{j/B_i}}\begin{bmatrix} x_{j/B_i} \\ y_{j/B_i} \end{bmatrix} \tag{3.52}$$

Substituting (3.50) into the expression for u'_{j/B_i} produces

$$u'_{j/B_i} = \frac{x_{j/B_0}}{z_{j/B_0}} = \frac{c_{11}x_{j/B_0} + c_{12}y_{j/B_0} + c_{13}z_{j/B_0} + x_{B_0/B_i}}{c_{31}x_{j/B_0} + c_{32}y_{j/B_0} + c_{33}z_{j/B_0} + z_{B_0/B_i}} \tag{3.53}$$

Dividing the numerator and denominator by z_{j/B_0} and using (3.52),

$$u'_{j/B_i} = \frac{c_{11}u'_{j/B_0} + c_{12}v'_{j/B_0} + c_{13} + x_{B_0/B_i}/z_{j/B_0}}{c_{31}u'_{j/B_0} + c_{32}v'_{j/B_0} + c_{33} + z_{B_0/B_i}/z_{j/B_0}} \tag{3.54}$$

Rearranging terms, this becomes

$$\left[u'_{j/B_0}\left(c_{11} - u'_{j/B_i}c_{31}\right) + v'_{j/B_0}\left(c_{12} - u'_{j/B_i}c_{32}\right) + \left(c_{13} - u'_{j/B_i}c_{33}\right)\right] z_{j/B_0} = u'_{j/B_i}z_{B_0/B_i} - x_{B_0/B_i} \tag{3.55}$$

A similar argument involving v'_{j/B_0} gives

$$\left[u'_{j/B_0}\left(c_{21} - v'_{j/B_i}c_{31}\right) + v'_{j/B_0}\left(c_{22} - v'_{j/B_i}c_{32}\right) + \left(c_{23} - v'_{j/B_i}c_{33}\right)\right] z_{j/B_0} = v'_{j/B_i}z_{B_0/B_i} - y_{B_0/B_i} \tag{3.56}$$

Combining these equations for each image $1, \ldots, m$ gives the system

$$\boldsymbol{a}_j z_{j/B_0} = \boldsymbol{b}_j \tag{3.57}$$

where

$$\boldsymbol{a}_j = \begin{bmatrix} u'_{j/B_0}\left(c_{11} - u'_{j/B_1}c_{31}\right) + v'_{j/B_0}\left(c_{12} - u'_{j/B_1}c_{32}\right) + \left(c_{13} - u'_{j/B_1}c_{33}\right) \\ u'_{j/B_0}\left(c_{21} - v'_{j/B_1}c_{31}\right) + v'_{j/B_0}\left(c_{22} - v'_{j/B_1}c_{32}\right) + \left(c_{23} - v'_{j/B_1}c_{33}\right) \\ \vdots \\ u'_{j/B_0}\left(c_{11} - u'_{j/B_m}c_{31}\right) + v'_{j/B_0}\left(c_{12} - u'_{j/B_m}c_{32}\right) + \left(c_{13} - u'_{j/B_m}c_{33}\right) \\ u'_{j/B_0}\left(c_{21} - v'_{j/B_m}c_{31}\right) + v'_{j/B_0}\left(c_{22} - v'_{j/B_m}c_{32}\right) + \left(c_{23} - v'_{j/B_m}c_{33}\right) \end{bmatrix} \tag{3.58}$$

$$\boldsymbol{b}_j = \begin{bmatrix} u'_{j/B_1}z_{B_0/B_1} - x_{B_0/B_1} \\ v'_{j/B_1}z_{B_0/B_1} - y_{B_0/B_1} \\ \vdots \\ u'_{j/B_m}z_{B_0/B_m} - x_{B_0/B_m} \\ v'_{j/B_m}z_{B_0/B_m} - y_{B_0/B_m} \end{bmatrix} \tag{3.59}$$

An estimate for z_{j/B_0} is obtained via linear least squares as

$$z_{j/B_0} = \left(\boldsymbol{a}_j^T W \boldsymbol{a}_j\right)^{-1} \boldsymbol{a}_j^T W \boldsymbol{b}_j \tag{3.60}$$

Where W is a positive-definite $2m$-by-$2m$ weighting matrix. Once z_{j/B_0} is determined, the remaining components of $\boldsymbol{r}_{j/B_0}$ are computed using (3.52) as

$$\begin{bmatrix} x_{j/B_0} \\ y_{j/B_0} \end{bmatrix} = z_{j/B_0} \begin{bmatrix} u'_{j/B_0} \\ v'_{j/B_0} \end{bmatrix} \tag{3.61}$$

The variance of the z_{j/B_0} estimate is given by

$$\sigma^2_{z_{j/B_0}} = \left(\boldsymbol{a}_j^T W \boldsymbol{a}_j\right)^{-1} \tag{3.62}$$

This can be scaled by a factor of $|u'_{j/B_0}|$ to obtain the variance of the x estimate, and by $|v'_{j/B_0}|$ for the variance of the x estimate. Thus the covariance of the $\boldsymbol{r}_{j/B_0}$ estimate is

$$P_{\boldsymbol{r}_{j/B_0}} = \sigma^2_{z_{j/B_0}} \begin{bmatrix} u'^2_{j/B_0} & 0 & 0 \\ 0 & v'^2_{j/B_0} & 0 \\ 0 & 0 & 1 \end{bmatrix} \tag{3.63}$$

The covariance of the feature position estimate can be fine-tuned by adjusting the weighting matrix design parameter. In scenarios where the body exhibits small translations between images, using a small W can help keep the filter from becoming overconfident about the position estimates.

The estimated position and covariance of the j^{th} feature can finally be transformed to the terrain frame N by performing

$$\boldsymbol{r}_{j/N} = C^T(\boldsymbol{q}_0)\boldsymbol{r}_{j/B_0} + \boldsymbol{p}_0 \tag{3.64}$$

and

$$P_{\boldsymbol{r}_{j/N}} = C^T(\boldsymbol{q}_0) P_{\boldsymbol{r}_{j/B_0}} C(\boldsymbol{q}_0) \tag{3.65}$$

The expressions in (3.64) and (3.65) are in the form needed for the filter's update step and covariance analysis.

3.5 Kalman Update

At the time of each camera image, the image space coordinates of the features will be used to take a previous state estimate $\hat{\boldsymbol{x}}^-$ and obtain an updated state error estimate $\hat{\boldsymbol{x}}^+$. This will be done by obtaining an estimated state error $\boldsymbol{\Delta x}$, and using that to correct $\hat{\boldsymbol{x}}^-$.

Fig. 3.1 shows the position vectors $\boldsymbol{r}_{j/N}$ and $\boldsymbol{r}_{j/B}$ for the j^{th} feature. As before, the feature's position in the body frame will be componentiated as $\boldsymbol{r}_{j/B} = \begin{bmatrix} x_{j/B} & y_{j/B} & z_{j/B} \end{bmatrix}^T$

Consider m features, and let the measurements be the dimensionless image-space coordinates (u_j, v_j). Then a measurement at time t_k is given by

$$\tilde{\boldsymbol{y}}_k = \left.\begin{bmatrix} u_1 \\ v_1 \\ \vdots \\ u_m \\ v_m \end{bmatrix}\right|_{t_k} = \left.\begin{bmatrix} \boldsymbol{h}\left(\boldsymbol{r}_{1/B}(\boldsymbol{x})\right) \\ \vdots \\ \boldsymbol{h}\left(\boldsymbol{r}_{m/B}(\boldsymbol{x})\right) \end{bmatrix}\right|_{t_k} + \left.\begin{bmatrix} \nu_{u_1} \\ \nu_{v_1} \\ \vdots \\ \nu_{u_m} \\ \nu_{v_m} \end{bmatrix}\right|_{t_k} \tag{3.66}$$

where the body-frame relative position of the j^{th} feature is

$$\boldsymbol{r}_{j/B}(\boldsymbol{x}) = C(\boldsymbol{q})(\boldsymbol{r}_{j/N} - \boldsymbol{p}) \tag{3.67}$$

The Jacobian matrix of $\boldsymbol{h}$ with respect to the error states can be computed as

$$\left.\frac{\partial \boldsymbol{h}}{\partial(\boldsymbol{\Delta x})}\right|_{\boldsymbol{\Delta x}=\boldsymbol{0}} = \frac{\partial \boldsymbol{h}}{\partial \boldsymbol{r}_{j/B}} \left.\frac{\partial \boldsymbol{r}_{j/B}}{\partial(\boldsymbol{\Delta x})}\right|_{\boldsymbol{\Delta x}=\boldsymbol{0}} \tag{3.68}$$

The first partial derivative was obtained in (3.24). For the remaining partial derivatives,

$$\begin{aligned} \frac{\partial \boldsymbol{r}_{j/B}}{\partial \boldsymbol{\alpha}} &= \frac{\partial}{\partial \boldsymbol{\alpha}} C(\boldsymbol{\delta q}) C(\hat{\boldsymbol{q}}^-)(\boldsymbol{r}_{j/N} - \boldsymbol{p}) \\ &= \frac{\partial}{\partial \boldsymbol{\alpha}} \left(I - [\boldsymbol{\alpha}\times]\right) C(\hat{\boldsymbol{q}}^-)(\boldsymbol{r}_{j/N} - \boldsymbol{p}) \\ &= \left[C(\hat{\boldsymbol{q}}^-)(\boldsymbol{r}_{j/N} - \hat{\boldsymbol{p}})\times\right] \end{aligned} \tag{3.69}$$

and

$$\begin{aligned}\frac{\partial \boldsymbol{r}_{j/B}}{\partial(\boldsymbol{\Delta p})} &= \frac{\partial}{\partial(\boldsymbol{\Delta p})} C(\boldsymbol{q})(\boldsymbol{r}_{j/N} - \hat{\boldsymbol{p}}^{-} - \boldsymbol{\Delta p}) \\ &= -C(\hat{\boldsymbol{q}}^{-}) \end{aligned} \tag{3.70}$$

The derivatives with respect to the other error states are 0. Thus from (3.68),

$$\left.\frac{\partial \boldsymbol{h}}{\partial(\boldsymbol{\Delta x})}\right|_{\boldsymbol{\Delta x}=\boldsymbol{0}} = \frac{\partial \boldsymbol{h}}{\partial \boldsymbol{r}_{j/B}} \begin{bmatrix} \left[\hat{\boldsymbol{r}}_{j/B}\times\right] & -C(\hat{\boldsymbol{q}}^{-}) & 0_{3,9} \end{bmatrix} \tag{3.71}$$

where

$$\hat{\boldsymbol{r}}_{j/B} = C(\hat{\boldsymbol{q}}^{-})(\boldsymbol{r}_{j/N} - \hat{\boldsymbol{p}}^{-}) \tag{3.72}$$

is the feature's relative vehicle-frame position vector evaluated at the estimated state. Therefore the measurement sensitivity matrix at time t_k is

$$H_k = \left.\begin{bmatrix} \frac{\partial \boldsymbol{h}}{\partial \boldsymbol{r}_{1/B}} \begin{bmatrix} \left[\hat{\boldsymbol{r}}_{1/V}\times\right] & -C(\hat{\boldsymbol{q}}^{-}) & 0_{3,9} \end{bmatrix} \\ \vdots \\ \frac{\partial \boldsymbol{h}}{\partial \boldsymbol{r}_{m/B}} \begin{bmatrix} \left[\hat{\boldsymbol{r}}_{m/V}\times\right] & -C(\hat{\boldsymbol{q}}^{-}) & 0_{3,9} \end{bmatrix} \end{bmatrix}\right|_{t_k} \tag{3.73}$$

The Kalman gain can then be computed by

$$K_k = P_k^{-} H_k^T \left(H_k P_k^{-} H_k^T + S_k\right)^{-1} \tag{3.74}$$

where the matrix S_k is the covariance of the camera measurement errors, including feature position estimation errors, which is shown in (3.22).

The Kalman gain is used to compute the updated state error and covariance

$$\begin{aligned}\boldsymbol{\Delta x}_k &= K_k\left[\tilde{\boldsymbol{y}}_k - \boldsymbol{h}(\hat{\boldsymbol{x}}_k^-)\right]\\ P_k^+ &= \left[I - K_kH_k\right]P_k^- \end{aligned}\tag{3.75}$$

Here, an abuse of notation allows the predicted image-space measurement of all m features to be referred to compactly as $\boldsymbol{h}(\hat{\boldsymbol{x}}_k^-)$. I.e.,

$$\boldsymbol{h}(\hat{\boldsymbol{x}}_k^-) = \begin{bmatrix}\boldsymbol{h}\left(\boldsymbol{r}_{1/B}(\hat{\boldsymbol{x}}_k^-)\right)\\ \vdots \\ \boldsymbol{h}\left(\boldsymbol{r}_{m/B}(\hat{\boldsymbol{x}}_k^-)\right)\end{bmatrix}\tag{3.76}$$

Using a matrix identity for the multiplication of quaternions, substituting the euler angle error $\boldsymbol{\alpha}_k$ into the quaternion update $\hat{\boldsymbol{q}}_k^+ = \boldsymbol{\delta q}\otimes\hat{\boldsymbol{q}}_k^-$ via (3.4) gives

$$\hat{\boldsymbol{q}}_k^+ = \begin{bmatrix}\frac{1}{2}\boldsymbol{\alpha}_k\\ 1\end{bmatrix}\otimes\hat{\boldsymbol{q}}_k^- = \begin{bmatrix}\Xi(\hat{\boldsymbol{q}}_k^-) & \hat{\boldsymbol{q}}_k^-\end{bmatrix}\begin{bmatrix}\frac{1}{2}\boldsymbol{\alpha}_k\\ 1\end{bmatrix}\tag{3.77}$$

Thus the full state update can be written

$$\begin{aligned}\hat{\boldsymbol{q}}_k^+ &= \hat{\boldsymbol{q}}_k^- + \frac{1}{2}\Xi(\hat{\boldsymbol{q}}_k^-)\boldsymbol{\alpha}_k\\ \hat{\boldsymbol{p}}_k^+ &= \hat{\boldsymbol{p}}_k^- + \boldsymbol{\Delta p}_k\\ \hat{\boldsymbol{v}}_k^+ &= \hat{\boldsymbol{v}}_k^- + \boldsymbol{\Delta v}_k\\ \hat{\boldsymbol{\beta}}_{gk}^+ &= \hat{\boldsymbol{\beta}}_{gk}^- + \boldsymbol{\Delta\beta}_{gk}\\ \hat{\boldsymbol{\beta}}_{ak}^+ &= \hat{\boldsymbol{\beta}}_{ak}^- + \boldsymbol{\Delta\beta}_{ak}\end{aligned}\tag{3.78}$$

The estimator will not preserve quaternion normalization precisely, so the quaternion can be manually re-normalized throughout the process.

The full filter implementation is tabulated in Table 3.1.

Initialize	$\hat{\boldsymbol{q}}(t_0)=\hat{\boldsymbol{q}}_0, \quad \hat{\boldsymbol{p}}(t_0)=\hat{\boldsymbol{p}}_0, \quad \hat{\boldsymbol{v}}(t_0)=\hat{\boldsymbol{v}}_0$ $\hat{\boldsymbol{\beta}}_g(t_0)=\hat{\boldsymbol{\beta}}_{g0}, \quad \hat{\boldsymbol{\beta}}_a(t_0)=\hat{\boldsymbol{\beta}}_{a0}, \quad P(t_0)=P_0$
Propagate	$\hat{\boldsymbol{\omega}}=\tilde{\boldsymbol{\omega}}-\hat{\boldsymbol{\beta}}_g, \quad \hat{\boldsymbol{a}}=\tilde{\boldsymbol{a}}-\hat{\boldsymbol{\beta}}_a$ $\dot{\hat{\boldsymbol{q}}}(t)=\frac{1}{2}\Xi\left(\hat{\boldsymbol{q}}(t)\right)\hat{\boldsymbol{\omega}}(t), \quad \dot{\hat{\boldsymbol{p}}}(t)=\hat{\boldsymbol{v}}(t), \quad \dot{\hat{\boldsymbol{v}}}(t)=C^T\left(\hat{\boldsymbol{q}}(t)\right)\hat{\boldsymbol{a}}(t)$ $\dot{P}(t)=F(t)P(t)+P(t)F^T(t)+G(t)Q(t)G^T(t)$ $F(t)=\begin{bmatrix}-[\hat{\boldsymbol{\omega}}(t)\times] & 0 & 0 & -I_3 & 0\\ 0 & 0 & I_3 & 0 & 0\\ 0 & 0 & 0 & 0 & -C^T\left(\hat{\boldsymbol{q}}(t)\right)\\ 0 & 0 & 0 & 0 & 0\\ 0 & 0 & 0 & 0 & 0\end{bmatrix}$ $G(t)=\begin{bmatrix}-I_3 & 0 & 0 & 0\\ 0 & 0 & 0 & 0\\ 0 & 0 & -C^T\left(\hat{\boldsymbol{q}}(t)\right) & 0\\ 0 & I_3 & 0 & 0\\ 0 & 0 & 0 & I_3\end{bmatrix}$
Estimate Features	$\boldsymbol{r}_{j/N}=C^T(\boldsymbol{q}_0)\boldsymbol{r}_{j/B_0}+\boldsymbol{p}_0, \quad P_{\boldsymbol{r}_{j/N}}=C^T(\boldsymbol{q}_0)P_{\boldsymbol{r}_{j/B_0}}C(\boldsymbol{q}_0)$ $\begin{bmatrix}x_{j/B_0}\\ y_{j/B_0}\end{bmatrix}=z_{j/B_0}\begin{bmatrix}u'_{j/B_0}\\ v'_{j/B_0}\end{bmatrix}, \quad P_{\boldsymbol{r}_{j/B_0}}=\sigma^2_{z_{j/B_0}}\begin{bmatrix}{u'}^{2}_{j/B_0} & 0 & 0\\ 0 & {v'}^{2}_{j/B_0} & 0\\ 0 & 0 & 1\end{bmatrix}$ $z_{j/B_0}=\left(\boldsymbol{a}_j^TW\boldsymbol{a}_j\right)^{-1}\boldsymbol{a}_j^TW\boldsymbol{b}_j, \quad \sigma^2_{z_{j/B_0}}=\left(\boldsymbol{a}_j^TW\boldsymbol{a}_j\right)^{-1}$
Gain	$K_k=P_k^-H_k^T(\hat{\boldsymbol{x}}_k^-)\left[H_k(\hat{\boldsymbol{x}}_k^-)P_k^-H_k^T(\hat{\boldsymbol{x}}_k^-)+S_k\right]^{-1}$ $H_k(\hat{\boldsymbol{x}}_k^-)=\left.\begin{bmatrix}\frac{\partial\boldsymbol{h}}{\partial\boldsymbol{r}_{1/B}}\begin{bmatrix}[\hat{\boldsymbol{r}}_{1/V}\times] & -C(\hat{\boldsymbol{q}}^-) & 0_{3,9}\end{bmatrix}\\ \vdots\\ \frac{\partial\boldsymbol{h}}{\partial\boldsymbol{r}_{m/B}}\begin{bmatrix}[\hat{\boldsymbol{r}}_{m/V}\times] & -C(\hat{\boldsymbol{q}}^-) & 0_{3,9}\end{bmatrix}\end{bmatrix}\right\rvert_{t_k}$ $S_k=\left.\begin{bmatrix}\frac{\partial\boldsymbol{h}}{\partial\hat{\boldsymbol{r}}_{1/B}}E[\boldsymbol{\Delta r}_{1/B}\boldsymbol{\Delta r}_{1/B}^T]\frac{\partial\boldsymbol{h}}{\partial\hat{\boldsymbol{r}}_{1/B}}^T & & 0\\ & \ddots & \\ 0 & & \frac{\partial\boldsymbol{h}}{\partial\hat{\boldsymbol{r}}_{m/B}}E[\boldsymbol{\Delta r}_{m/B}\boldsymbol{\Delta r}_{m/B}^T]\frac{\partial\boldsymbol{h}}{\partial\hat{\boldsymbol{r}}_{m/B}}^T\end{bmatrix}\right\rvert_{t_k}+R$
Update	$P_k^+=\left[I-K_kH_k(\hat{\boldsymbol{x}}_k^-)\right]P_k^-$ $\boldsymbol{\Delta x}_k=K_k\left[\tilde{\boldsymbol{y}}_k-\boldsymbol{h}(\hat{\boldsymbol{x}}_k^-)\right]$ $\boldsymbol{\Delta x}_k=\begin{bmatrix}\boldsymbol{\delta\alpha}_k^T & \boldsymbol{\Delta p}_k^T & \boldsymbol{\Delta v}_k^T & \boldsymbol{\Delta\beta}_{gk}^T & \boldsymbol{\Delta\beta}_{ak}^T\end{bmatrix}^T$ $\boldsymbol{h}(\hat{\boldsymbol{x}}_k^-)=\begin{bmatrix}\boldsymbol{h}\left(\boldsymbol{r}_{1/B}(\hat{\boldsymbol{x}}_k^-)\right)\\ \vdots\\ \boldsymbol{h}\left(\boldsymbol{r}_{m/B}(\hat{\boldsymbol{x}}_k^-)\right)\end{bmatrix}$ $\hat{\boldsymbol{q}}_k^+=\hat{\boldsymbol{q}}_k^-+\frac{1}{2}\Xi(\hat{\boldsymbol{q}}_k^-)\boldsymbol{\alpha}_k, \quad \hat{\boldsymbol{p}}_k^+=\hat{\boldsymbol{p}}_k^-+\boldsymbol{\Delta p}_k, \quad \hat{\boldsymbol{v}}_k^+=\hat{\boldsymbol{v}}_k^-+\boldsymbol{\Delta v}_k$ $\hat{\boldsymbol{\beta}}_{gk}^+=\hat{\boldsymbol{\beta}}_{gk}^-+\boldsymbol{\Delta\beta}_{gk}, \quad \hat{\boldsymbol{\beta}}_{ak}^+=\hat{\boldsymbol{\beta}}_{ak}^-+\boldsymbol{\Delta\beta}_{ak}$

Table 3.1: Planetary EDL MEKF implementation.

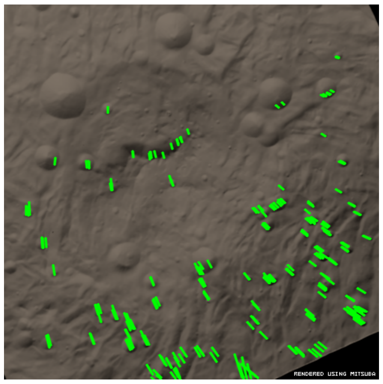


Figure 3.3: Feature tracks drawn between feature positions in two successive simulated images.

3.6 Application to Testing with Simulated Terrain

Filter performance is reported for testing using a simulated camera and IMU measurements over simulated terrain. This is used to test the accuracy of the filter, since the ground truth of the vehicle's pose is known.

The trajectory plots are shown in an inertial frame defined to be be equal to the initial body frame. The body axes are defined as discussed in section 2.2. Namely, the x-axis points to the right side of the image, the y-axis points to the bottom edge of the image, and the z-axis completes the triad by pointing along the camera's line of sight, into the image.

The simulation dataset consisted of 50 images and 500 IMU measurements. The images have a resolution of 500x500. The IMU measurements were generated from the true state of the vehicle using measurement noise values of $\sigma_{gv} = \sqrt{10} \times 10^{-4}$, $\sigma_{av} = \sqrt{10} \times 10^{-2}$, and $\sigma_{gu} = \sigma_{au} = 1 \times 10^{-5}$. The initial covariance of each portion of the state was a scalar multiple of the identity matrix, with scale factors $\sigma_{\boldsymbol{\alpha}} = \frac{\pi}{180}$, $\sigma_{\boldsymbol{p}} = \sqrt{3}$, $\sigma_{\boldsymbol{v}} = 1$, and $\sigma_{\boldsymbol{\beta}_g} = \sigma_{\boldsymbol{\beta}_a} = \frac{\pi}{180}\frac{\Delta t}{3600}$, where Δt is the time between IMU measurements. The noise in the camera measurements was assumed to have an uncertainty of $\sigma_{uj} = \sigma_{vj} = 5$ pixels.

A sample image from the feature tracking of the simulated data is shown in Fig. 3.3. Estimated 3D feature locations are overlaid atop an image in Fig. 3.4.

The estimated and true trajectories for the full flight are plotted in Fig. 3.5, along with

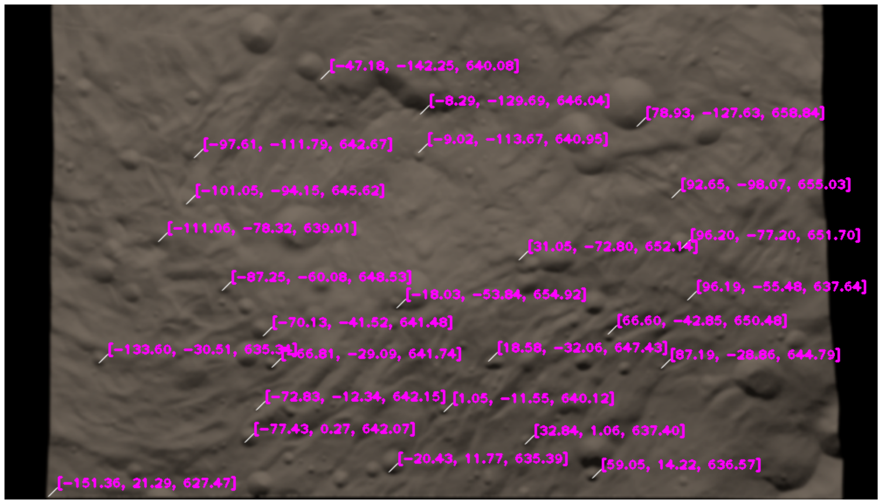


Figure 3.4: 3D feature position estimates. The true perpendicular distance from the camera to the terrain is 650, so most estimates are accurate.

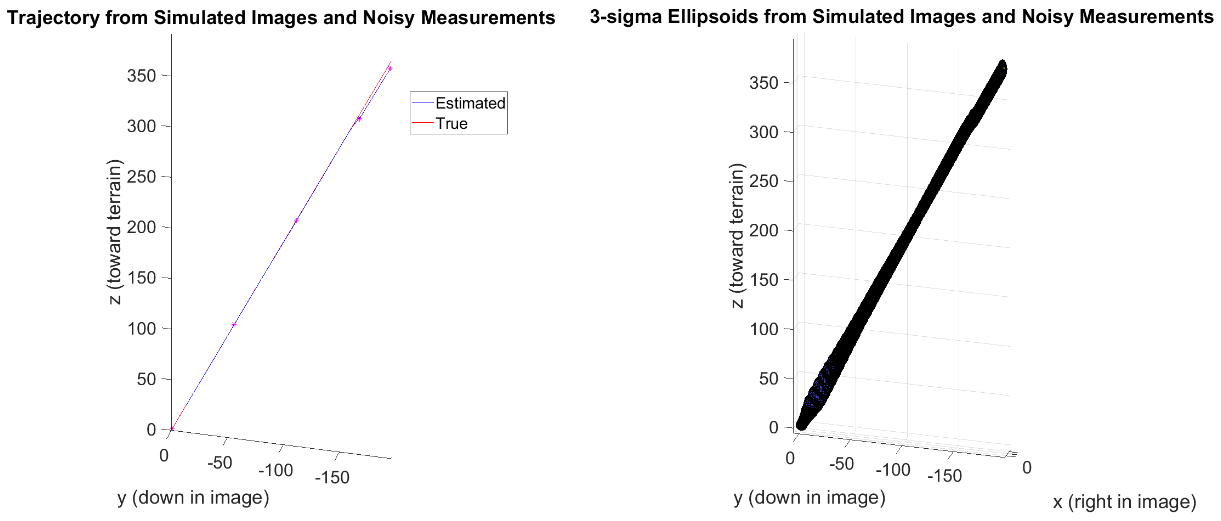


Figure 3.5: (Left): the estimated and true trajectories of the simulated body. (Right): the same trajectories overlaid by the 3σ ellipses representing estimation uncertainty.

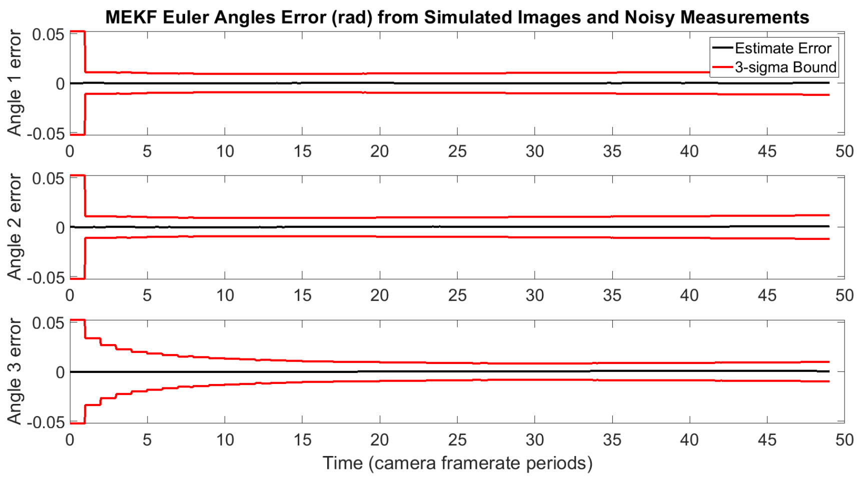


Figure 3.6: Error in Euler angle estimates from simulated dataset, along with 3σ bounds.

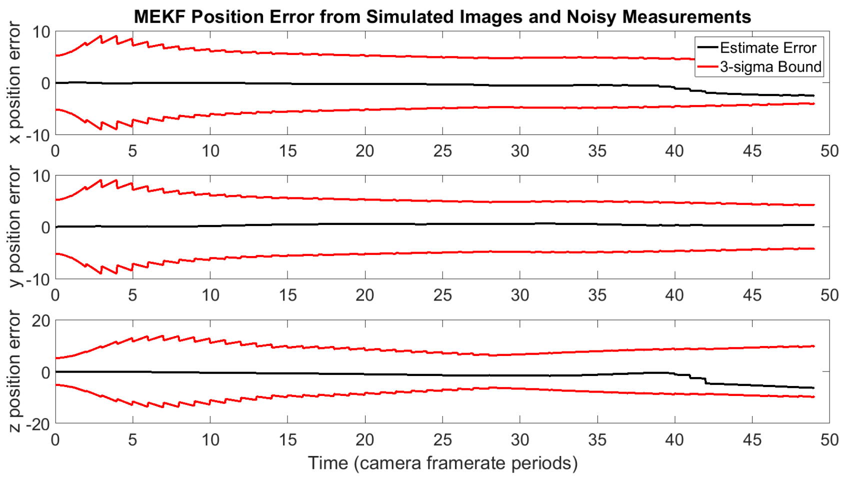


Figure 3.7: Error in position estimates from simulated dataset, along with 3σ bounds.

the 3σ bounds that show the estimator's uncertainty.

The errors in estimated Euler angles and position are shown in Figs. 3.6 and 3.7. The estimated Euler angles are obtained as follows. Denote the components of the quaternion's rotation matrix as

$$C(\hat{\boldsymbol{q}}) = \begin{bmatrix} c_{11} & c_{12} & c_{13} \\ c_{21} & c_{22} & c_{23} \\ c_{31} & c_{32} & c_{33} \end{bmatrix} = \left(q_4^2 - \boldsymbol{\rho}^T\boldsymbol{\rho}\right) I_3 + 2\boldsymbol{\rho}\boldsymbol{\rho}^T - 2q_4\left[\boldsymbol{\rho}\times\right] \tag{3.79}$$

where the $\hat{.}$ notation is suppressed for brevity. The Euler angles for a 1-2-3 rotation sequence (ϕ, θ, ψ) are

$$\begin{aligned} \phi &= \tan^{-1}\left(-\frac{c_{32}}{c_{33}}\right) \\ \theta &= \tan^{-1}\left(\frac{c_{31}}{\sqrt{1-c_{31}^2}}\right) \\ \psi &= \tan^{-1}\left(-\frac{c_{21}}{c_{11}}\right) \end{aligned} \tag{3.80}$$

Direct formulae can also be derived using the relationships shown in these equations, as shown in Schaub and Junkins [12].

3.7 Application to Testing in a Laboratory Environment

The filter performance is also demonstrated on a real dataset obtained from the NEST test bed at LASR Laboratory at Texas A&M. This is used to verify the filter is able to process real-world data and produce reasonable results.

The data from NEST consisted of images from an onboard camera and IMU, both at 30 Hz. The images have a resolution of 1280x1024 (in pixels (px.)). The uncertainty of the gyroscope is $\sigma_{gv} = \sqrt{10} \times 10^{-4}\text{rad/s}\sqrt{\text{Hz}}$ and that of the accelerometer is $\sigma_{av} = \sqrt{10} \times 10^{-2}\text{m/s}^2\sqrt{\text{Hz}}$. The camera measurement noise was assumed to have an uncertainty of $\sigma_{uj} = \sigma_{vj} = 5\text{px}$.

The NEST dataset is not accompanied by any ground-truth for the state of the body. The initial conditions are also unknown.

A sample feature track is shown in Fig. 3.8, and a feature position estimation is in Fig.

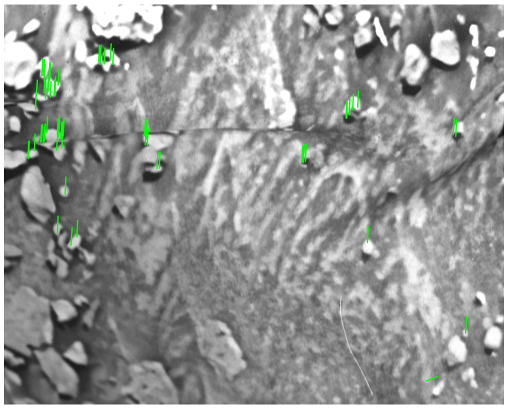


Figure 3.8: Feature tracks drawn between feature positions in two successive NEST images.

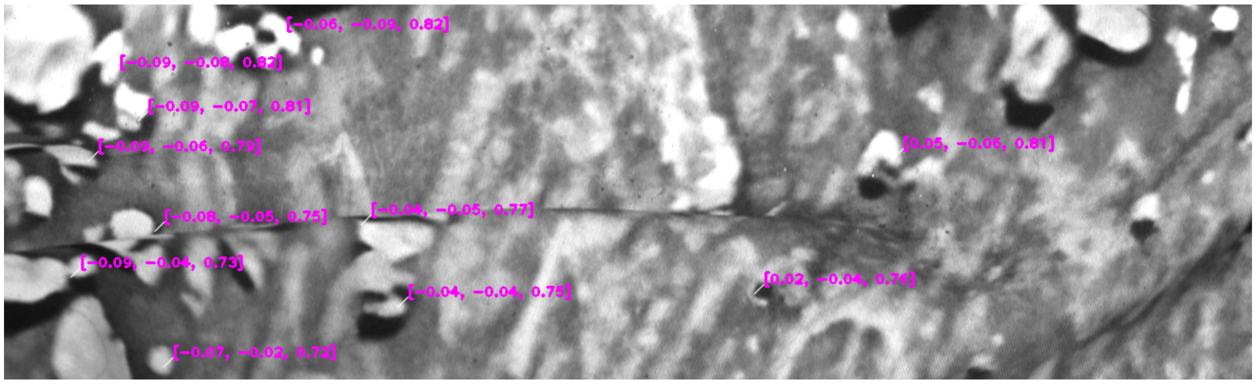


Figure 3.9: 3D feature position estimates from NEST dataset.

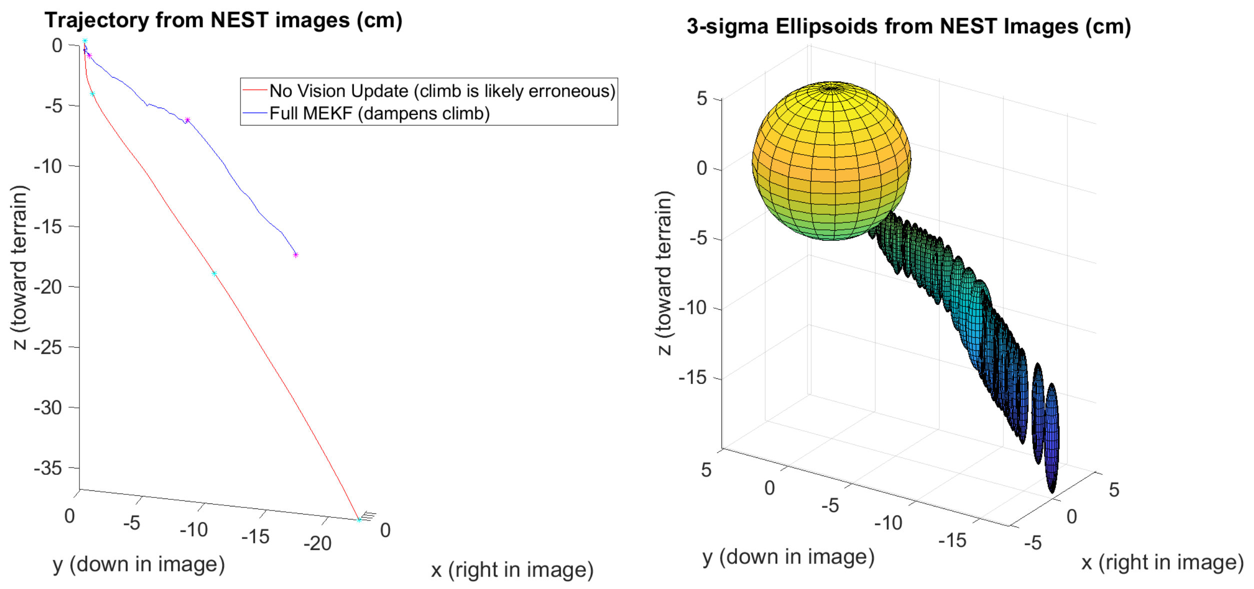


Figure 3.10: (LEFT): the estimated trajectory of the body from the NEST dataset, plotted gainst the trajectory estimated purely from IMU measurements. (Right): the estimated trajectory overlaid by the 3σ ellipses representing estimation uncertainty.

3.9. These figures reveal that the dataset for this run was quite out of focus, meaning it is not easy to track features. Thus the filter was not able to use as many features as in the simulated case, but it was still able to provide good estimates.

The estimated trajectory for this dataset is displayed in Fig. 3.10, along with the 3σ uncertainty ellipses. The effect of the lack of features can be seen from the shape of the uncertainty ellipses. They are stretched in the z-direction, which is the direction most impacted by the camera measurement updates.

The accelerometer measurements begin with a large bias in the $-z$-direction, but the filter is initialized without knowledge of this fact. The estimated trajectory therefore exhibits a constant climb. The straighter line in Fig. 3.10 shows the filter running without performing vision updates, while the shorter line features the full filter. The trajectory from the full filter exhibits a lesser climb in comparison, which shows a way that the vision updates can correct for biases in the IMU data.

4. AUTOMATED AERIAL REFUELING

Aerial refueling is a proximity operation that requires precision. A tanker, carrying fuel, and a receiver meet in the air and must make some sort of contact to initiate the refueling process. There is a lot of room for error, and so even though unmanned aircraft have existed for quite a while, it is only relatively recently that aerial refueling has been attempted with both vehicles unmanned. Developments in camera technology and research into how to use image processing in filters have made automated aerial refueling possible, since the camera allows for a more precise estimate of the relative pose of the vehicles.

This application uses a GPS and IMU on each of the two aircraft, as well as a camera on one of them. The other aircraft has a ring of LED beacons for the camera to photograph. This results in a simple vision processing stage, because the small points of light are easy to detect by placing a threshold on the image to eliminate all except the brightest spots. The beacons are arranged in a circle of known radius, so their projection in the image plane gives information about both the relative position as well as the relative orientation between the two objects. The projection of the ring of beacons forms an ellipse in the image plane. The parameters of the ellipse are determined as the solution to an optimization problem, and the ellipse parameters are then used to provide an estimate for the relative pose.

4.1 Tracked States

The state vector for the refueling problem is

$$\boldsymbol{x} = \begin{bmatrix} \boldsymbol{x}_C \\ \boldsymbol{x}_T \end{bmatrix} \tag{4.1}$$

where

$$\boldsymbol{x}_C = \begin{bmatrix} \boldsymbol{q}_C^T & \boldsymbol{p}_C^T & \boldsymbol{v}_C^T & \boldsymbol{\beta}_{Cg}^T & \boldsymbol{\beta}_{Ca}^T \end{bmatrix}^T \tag{4.2}$$

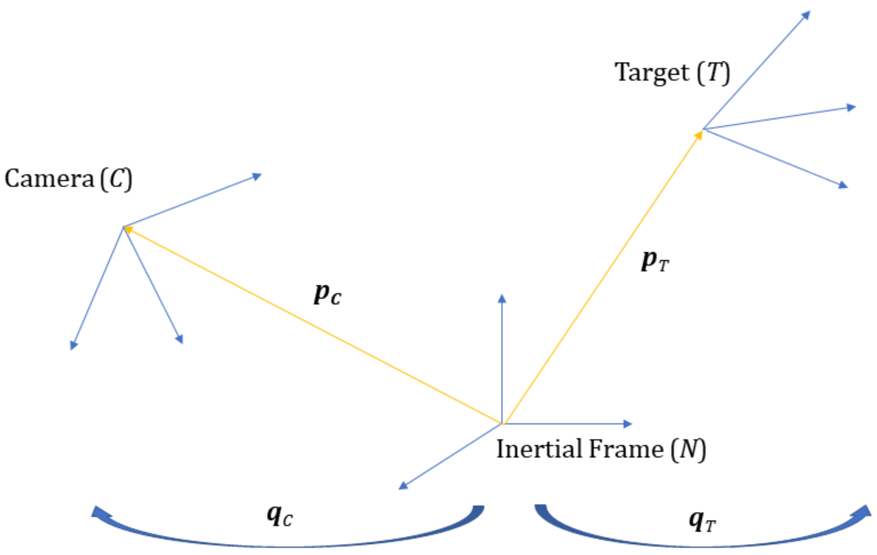


Figure 4.1: The camera (C) and target (T) body frames. There is a ring of LEDs centered at the origin of the target frame. The z-axis of C is along the line of the camera, and the z-axis for T is normal to the LEDs and pointing away from the camera.

Here, the camera's orientation $\boldsymbol{q}_C$, position $\boldsymbol{p}_C$, and velocity $\boldsymbol{v}_C$ are measured relative to the inertial reference frame N. The target's state vector $\boldsymbol{x}_T$ is defined similarly.

As in the EDL application above, an error state $\boldsymbol{\Delta x}$ will be used to compute the covariance P. This error state is defined as

$$\boldsymbol{\Delta x} = \begin{bmatrix} \boldsymbol{\Delta x}_C \\ \boldsymbol{\Delta x}_T \end{bmatrix} \tag{4.3}$$

where

$$\boldsymbol{\Delta x}_C = \begin{bmatrix} \boldsymbol{\alpha}_C^T & \boldsymbol{\Delta p}_C^T & \boldsymbol{\Delta v}_C^T & \boldsymbol{\Delta \beta}_{Cg}^T & \boldsymbol{\Delta \beta}_{Ca}^T \end{bmatrix}^T \tag{4.4}$$

and similarly for $\boldsymbol{\Delta x}_T$. The bodies and their position vectors are shown in Fig (4.1). From the way the reference frame axes are defined, the camera and target will be lined up when their z-axes coincide.

4.2 Measurement Model

The sensing devices for this application are the camera, a GPS on each body, and an IMU on each body. The IMU measurement model is identical to the one in the planetary EDL application.

4.2.1 Camera Measurement Model

In the EDL application, the image-space feature positions were used directly as the measurements for the MEKF. In this application, however, the known shape of the target can be exploited in a computer vision algorithm that outputs the relative position and orientation. The target is a ring of LEDs arranged in a circle. As will be shown in Sec. 4.2.1.2, the circle will project into the image space as an ellipse. Thus the computer vision algorithm must detect the LEDs, fit an ellipse to their image space positions, and finally estimate the relative position and orientation of the target.

The LEDs are identical, so this algorithm is not able to determine the complete relative orientation quaternion. Rather, it only determines the relative orientation of the target's normal vector. Thus the camera measurement will be composed of the measured relative position vector, $\tilde{\boldsymbol{p}}$, and a measured relative normal vector, $\tilde{\boldsymbol{n}}$. The measurement model is

$$\tilde{\boldsymbol{p}}_{T/C} = C_{C/T}\,\boldsymbol{p}_T - \boldsymbol{p}_C + \boldsymbol{\nu}_p \tag{4.5}$$

$$\tilde{\boldsymbol{n}} = C_{C/T}\,\boldsymbol{e}_3 + \boldsymbol{\nu}_n \tag{4.6}$$

where $\boldsymbol{e}_3 = \begin{bmatrix} 0 & 0 & 1 \end{bmatrix}^T$ is the target's normal vector reported in the target's reference frame, $C_{C/T} = C(\boldsymbol{q}_C)C^T(\boldsymbol{q}_T)$ is the transformation to the camera frame from the target frame, and $\boldsymbol{\nu}_p$, $\boldsymbol{\nu}_n$ are zero-mean Gaussian measurement noise with variance σ_p^2, σ_n^2, respectively. The estimation model is

$$\hat{\boldsymbol{p}}_{T/C} = \hat{C}_{C/T}\,\hat{\boldsymbol{p}}_T - \hat{\boldsymbol{p}}_C \tag{4.7}$$

$$\hat{\boldsymbol{n}} = \hat{C}_{C/T}\,\boldsymbol{e}_3 \tag{4.8}$$

where $\hat{C}_{C/T} = C(\hat{\boldsymbol{q}}_C)C^T(\hat{\boldsymbol{q}}_T)$

4.2.1.1 Fitting Ellipse to Camera Data

The ellipse fitting algorithm comes from Fitzgibbon and Fisher [13]. An ellipse is a conic section given by

$$au^2 + buv + cv^2 + du + ev + f = 0 \tag{4.9}$$

subject to the inequality $4ac - b^2 \geq 0$. Since (4.9) holds for an arbitrary scale factor, this inequality can be enforced as an equality constraint defined as

$$4ac - b^2 = 1 \tag{4.10}$$

Thus the coefficients $\boldsymbol{a} = [a, \ldots, f]^T$ of the best ellipse fit for a sequence of image-space points $(u_1, v_1), \ldots, (u_n, v_n)$ is the minimum of $\|D\boldsymbol{a}\|$ subject to the equality constraint $\boldsymbol{a}^T C\boldsymbol{a} = 1$ where

$$D = \begin{bmatrix} u_1^2 & u_1v_1 & v_1^2 & u_1 & v_1 & 1 \\ & & \vdots & & & \\ u_n^2 & u_nv_n & v_n^2 & u_n & v_n & 1 \end{bmatrix} \tag{4.11}$$

and

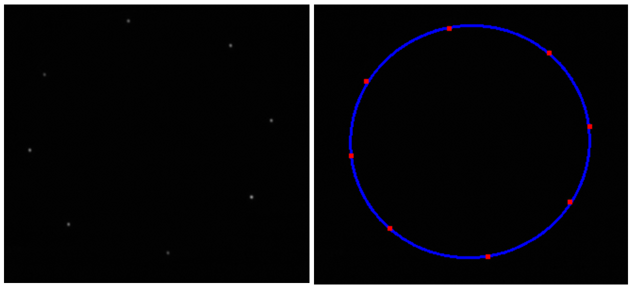


Figure 4.2: (Left): a photograph of the ring of LED beacons. (Right): the optimal ellipse fit.

$$C = \begin{bmatrix} 0 & 0 & 2 & 0 & 0 & 0 \\ 0 & -1 & 0 & 0 & 0 & 0 \\ 2 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 \end{bmatrix} \tag{4.12}$$

Utilizing the method of Lagrange multipliers, this problem reduces to the solution of the eigensystem

$$D^T D\boldsymbol{a} = \lambda C\boldsymbol{a} \tag{4.13}$$

subject to $\boldsymbol{a}^T C\boldsymbol{a} = 1$. It can be shown that this system has precisely one positive eigenvalue [13], and its eigenvector, $\boldsymbol{a}$, is the best ellipse fit for the data.

A sample of the ellipse fit is shown in Fig. (4.2). The ellipse drawn on the right image was obtained from the methods in this section.

4.2.1.2 Pose Estimation from Elliptical Projection

Fig. 4.3 shows a circular target projected onto an image plane. This section will show that its projection approximates an ellipse and derives a relationship between the ellipse parameters and the pose (translation $\boldsymbol{p}_{T/C}$ and normal vector $\boldsymbol{n}$) of the target.

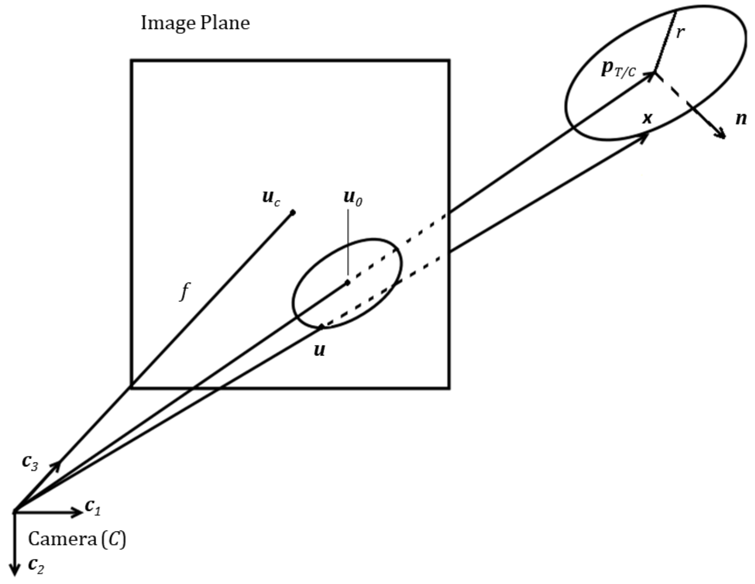


Figure 4.3: Projection of circular target with arbitrary translation $\boldsymbol{p}_{T/C}$ and orientation $\boldsymbol{n}$.

Let $\boldsymbol{x} = \begin{bmatrix} x & y & z \end{bmatrix}^T$ denote a point on the target and let $\boldsymbol{p}_{T/C} = \begin{bmatrix} x_0 & y_0 & z_0 \end{bmatrix}^T$ be its center. Furthermore, let $\boldsymbol{u} = \begin{bmatrix} u & v \end{bmatrix}^T$ and $\boldsymbol{u}_0 = \begin{bmatrix} u_0 & v_0 \end{bmatrix}^T$, respectively be the image-space projections of these two points. The camera intrinsics include the optical center $\boldsymbol{u}_c = \begin{bmatrix} u_c & v_c \end{bmatrix}^T$ and the focal length f. According to a pinhole camera model, the relationship between these quantities are

$$\boldsymbol{u} - \boldsymbol{u}_c = \frac{f}{z}\begin{bmatrix} x \\ y \end{bmatrix} \qquad \text{and} \qquad \boldsymbol{u}_0 - \boldsymbol{u}_c = \frac{f}{z_0}\begin{bmatrix} x_0 \\ y_0 \end{bmatrix} \tag{4.14}$$

or

$$\boldsymbol{u} - \boldsymbol{u}_0 = \frac{f}{z}\begin{bmatrix} x \\ y \end{bmatrix} - \frac{f}{z_0}\begin{bmatrix} x_0 \\ y_0 \end{bmatrix} \tag{4.15}$$

This can be simplified by assuming the radius of the target is small compared to the distance z_0. Under this assumption, $z \approx z_0$, which simplifies (4.15) to

$$\boldsymbol{u} - \boldsymbol{u}_0 = \frac{f}{z_0}\begin{bmatrix} x - x_0 \\ y - y_0 \end{bmatrix} \tag{4.16}$$

The target is assumed circular with known radius r, which means

$$||\boldsymbol{x} - \boldsymbol{p}_{T/C}||^2 = (x - x_0)^2 + (y - y_0)^2 + (z - z_0)^2 = r^2 \tag{4.17}$$

Further, the unit normal $\boldsymbol{n} = \begin{bmatrix} n_x & n_y & n_z \end{bmatrix}^T$ is defined perpendicular to the plane of the target, which means

$$(\boldsymbol{x} - \boldsymbol{p}_{T/C}) \cdot \boldsymbol{n} = (x - x_0)n_x + (y - y_0)n_y + (z - z_0)n_z = 0 \tag{4.18}$$

The radius constraint (4.17) and orthogonality constraint (4.18) can be combined to eliminate $z - z_0$ and obtain

$$\left(\frac{1}{rn_z}\right)^2 (\boldsymbol{x} - \boldsymbol{p}_{T/C})^T \begin{bmatrix} n_x^2 + n_z^2 & n_x n_y \\ n_x n_y & n_y^2 + n_z^2 \end{bmatrix} (\boldsymbol{x} - \boldsymbol{p}_{T/C}) = 1 \tag{4.19}$$

Transformed into image space via (4.16),

$$(\boldsymbol{u} - \boldsymbol{u}_0)^T Q (\boldsymbol{u} - \boldsymbol{u}_0) = 1 \tag{4.20}$$

where

$$Q = \left(\frac{z_0}{rfn_z}\right)^2 \begin{bmatrix} n_x^2 + n_z^2 & n_x n_y \\ n_x n_y & n_y^2 + n_z^2 \end{bmatrix} \tag{4.21}$$

Using the fact that $||\boldsymbol{n}|| = 1$, the determinant and trace of Q evaluate to

$$\det Q = \left(\frac{z_0}{rf}\right)^4 \frac{1}{n_z^2} \qquad \text{and} \qquad \operatorname{trace} Q = \left(\frac{z_0}{rf}\right)^2 \left(1 + \frac{1}{n_z^2}\right) \tag{4.22}$$

Since Q is a real symmetric 2-by-2 matrix with a positive determinant and trace, it must be positive-definite. By the spectral theorem, Q can thus be decomposed into an orthonormal eigenbasis $\boldsymbol{\varphi}_1, \boldsymbol{\varphi}_2$ with positive eigenvalues $\lambda_1 \leq \lambda_2$. The decomposition can be expressed

$$Q = \Phi \Lambda \Phi^T \tag{4.23}$$

where

$$\Phi = \left[\begin{array}{c|c} \boldsymbol{\varphi}_1 & \boldsymbol{\varphi}_2 \end{array} \right] \qquad \text{and} \qquad \Lambda = \begin{bmatrix} \lambda_1 & 0 \\ 0 & \lambda_2 \end{bmatrix} \tag{4.24}$$

This eigenbasis provides an orthogonal transformation of the image space coordinates

$$\begin{bmatrix} q_1 \\ q_2 \end{bmatrix} = \Phi^T(\boldsymbol{u} - \boldsymbol{u}_0) \tag{4.25}$$

Using the new coordinates, (4.20) can be written

$$\begin{bmatrix} q_1 & q_2 \end{bmatrix} \Lambda \begin{bmatrix} q_1 \\ q_2 \end{bmatrix} = 1 \tag{4.26}$$

or

$$\lambda_1 q_1^2 + \lambda_2 q_2^2 = 1 \tag{4.27}$$

This describes an ellipse with semimajor axis a and semiminor axis b given by

$$a = \frac{1}{\sqrt{\lambda_1}} \qquad \text{and} \qquad b = \frac{1}{\sqrt{\lambda_2}} \tag{4.28}$$

so that

$$\frac{q_1^2}{a^2} + \frac{q_2^2}{b^2} = 1 \tag{4.29}$$

Since the transformation (4.26) is orthogonal, it maintains the shape of the target's projection. This completes the proof that the projection of the circular target approximates an ellipse.

Since the eigenvalues λ_1, λ_2 of Q can be expressed using the semimajor and semiminor axes a, b using (4.28), the determinant and trace can be written in the form

$$\det Q = \lambda_1 \lambda_2 = \frac{1}{a^2 b^2} \qquad \text{and} \qquad \operatorname{trace} Q = \lambda_1 + \lambda_2 = \frac{a^2+b^2}{a^2 b^2} \tag{4.30}$$

Equating (4.30) and (4.22) and performing some algebra gives the relationships

$$r = \frac{z_0}{f}\sqrt{\frac{ab}{|n_z|}} \tag{4.31}$$

and

$$|n_z|^2 - \frac{a^2+b^2}{a^2 b^2}|n_z| + 1 = 0 \tag{4.32}$$

the solution to the quadratic equation (4.32) is $|n_z| = \frac{1}{2ab}\left[(a^2+b^2) \pm \ (a^2-b^2)\right]$. The upper sign would give $|n_z| = a/b$, which would contradict the fact that $\boldsymbol{n}$ is a unit vector. Thus the lower sign must be chosen, which means $|n_z| = b/a$. This permits either choice of sign for n_z, which correspond to the two unit vectors normal to the target, each pointing in opposite directions. The convention used in Fig. 4.3 is that $\boldsymbol{n}$ has a positive component along the $\boldsymbol{c}_3$ direction, i.e., n_z is positive. Thus the only permissible solution to (4.32) is

$$n_z = \frac{b}{a} \tag{4.33}$$

This reduces (4.31) to

$$r = \frac{z_0}{f}a \tag{4.34}$$

which indicates that the semimajor axis a is simply the radius r scaled into image space

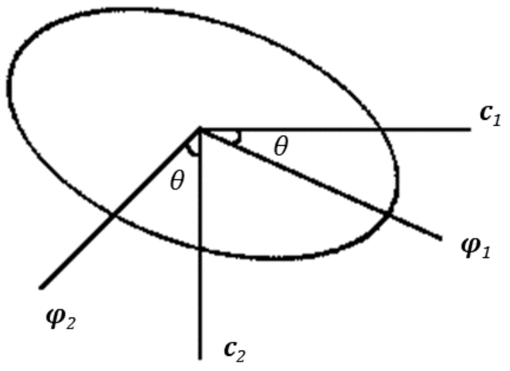


Figure 4.4: Rotation from camera axes to ellipse principal axes.

coordiantes. This is in accordance with intuition in that a video of a stationary circle undergoing rotation about one of its diameters will appear to stay the same size along that diameter, even while it appears to shrink along all other axes.

Substituting (4.33) and (4.34) into (4.21) produces

$$Q = \frac{1}{b^2}\begin{bmatrix} n_x^2 + b^2/a^2 & n_x n_y \\ n_x n_y & n_y^2 + b^2/a^2 \end{bmatrix} \tag{4.35}$$

A solution for n_x, n_y is desired in terms of the ellipse parameters. To this end, another expression for Q will be found. (4.29) shows that the othonormal eigenvectors $\boldsymbol{\varphi}_1, \boldsymbol{\varphi}_2$ must represent the principal axes of the ellipse, namely the semimajor and semiminor axes, respectively. Thus, letting θ denote the angle from the camera's $\boldsymbol{c}_1$ axis to the semimajor axis $\boldsymbol{\varphi}_1$ as in Fig. 4.4, the relationship between the bases is seen to be

$$\boldsymbol{\varphi}_1 = \cos\theta \boldsymbol{c}_1 + \sin\theta \boldsymbol{c}_2 \qquad \text{and} \qquad \boldsymbol{\varphi}_2 = -\sin\theta \boldsymbol{c}_1 + \cos\theta \boldsymbol{c}_2 \tag{4.36}$$

Substituting (4.36) into (4.23) gives

$$Q = \begin{bmatrix} \cos\theta & -\sin\theta \\ \sin\theta & \cos\theta \end{bmatrix} \begin{bmatrix} 1/a^2 & 0 \\ 0 & 1/b^2 \end{bmatrix} \begin{bmatrix} \cos\theta & \sin\theta \\ -\sin\theta & \cos\theta \end{bmatrix}$$

$$= \begin{bmatrix} \frac{\cos^2\theta}{a^2} + \frac{\sin^2\theta}{b^2} & -\sin\theta\cos\theta\left(\frac{1}{b^2} - \frac{1}{a^2}\right) \\ -\sin\theta\cos\theta\left(\frac{1}{b^2} - \frac{1}{a^2}\right) & \frac{\sin^2\theta}{a^2} + \frac{\cos^2\theta}{b^2} \end{bmatrix} \tag{4.37}$$

Equating the top-left entries of (4.35) and (4.37) gives, after some algebra,

$$n_x^2 = \left(1 - \frac{b^2}{a^2}\right)\sin^2\theta = e^2\sin^2\theta \tag{4.38}$$

where the eccentricity $e = \sqrt{1 - b^2/a^2}$ has been introduced to simplify the expression. Equating the off-diagonal entries and simplifying leads to

$$n_y = -\frac{(1 - b^2/a^2)\sin\theta\cos\theta}{n_x} = -\frac{e^2\sin\theta\cos\theta}{n_x} \tag{4.39}$$

Combining these expressions with (4.33) gives a solution for the target's unit normal vector $\boldsymbol{n}$ in terms of the ellipse parameters.

$$\boldsymbol{n} = \begin{bmatrix} \pm e\sin\theta \\ \mp e\cos\theta \\ \sqrt{1 - e^2} \end{bmatrix} \tag{4.40}$$

Thus $\boldsymbol{n}$ depends only on the eccentricity e and the tilt θ of the ellipse, and not on its size or location. Furthermore, there is an ambiguity between two distinct solutions (corresponding to a choice of the upper signs or lower signs). These reflect an inherent ambiguity in pose estimation from ellipse parameters. The illustration in Fig. 4.5 demonstrates an example of two orientations corresponding to one ellipse.

In the filter, the previous estimate for the normal vector is be compared to the two potential estimates. The dot product with each possible estimate is computed, and the one

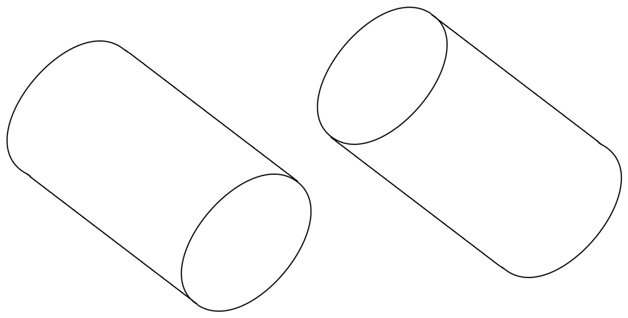


Figure 4.5: Two distinct orientations of a circular face that produce the same elliptical projection.

with the larger product is chosen. This method works well as long as the camera's frame rate is sufficiently high that the target doesn't rotate much between images.

To determine the position $\boldsymbol{p}_{T/C}$ of the target's center, (4.14) can be used to obtain

$$\boldsymbol{p}_{T/C} = \begin{bmatrix} x_0 \\ y_0 \\ z_0 \end{bmatrix} = \frac{z_0}{f} \begin{bmatrix} u_0 - u_c \\ v_0 - v_c \\ f \end{bmatrix} \tag{4.41}$$

The z_0 can be eliminated from the scale factor using (4.34). This gives

$$\boldsymbol{p}_{T/C} = \frac{r}{a} \begin{bmatrix} u_0 - u_c \\ v_0 - v_c \\ f \end{bmatrix} \tag{4.42}$$

Thus the target's position depends only on the ellipse's center (u_0, v_0), the semimajor axis a, the target's radius r, and the camera intrinsics, namely the focus f and the optical center (u_c, v_c).

4.2.2 GPS Measurement Model

The GPS provides direct estimates for the inertial positions of the two bodies. THe measurement model is

$$\tilde{\boldsymbol{p}}_C = \boldsymbol{p}_C + \boldsymbol{\nu}_C \qquad \tilde{\boldsymbol{p}}_T = \boldsymbol{p}_T + \boldsymbol{\nu}_T \tag{4.43}$$

where $\boldsymbol{\nu}_C$ and $\boldsymbol{\nu}_T$ are zero-mean Gaussian random variables with variances $\sigma_C^2 I$ and $\sigma_T^2 I$ respectively.

4.3 State Propagation

The camera and target vehicles have separate IMU devices and they are not assumed to be in sync. This section demonstrates the propagation for the camera's IMU only, because the propagation for the target's IMU is similar.

The estimated angular velocity $\hat{\boldsymbol{\omega}}_C$ and acceleration $\hat{\boldsymbol{a}}_C$ are obtained from the IMU measurments $\tilde{\boldsymbol{\omega}}_C$, $\tilde{\boldsymbol{a}}_C$ as

$$\hat{\boldsymbol{\omega}}_C = \tilde{\boldsymbol{\omega}}_C - \hat{\boldsymbol{\beta}}_{Cg}, \qquad \hat{\boldsymbol{a}}_C = \tilde{\boldsymbol{a}}_C - \hat{\boldsymbol{\beta}}_{Ca} \tag{4.44}$$

The propagation equations are the same as in the EDL section. Namely, the state propagation is:

$$\dot{\hat{\boldsymbol{q}}}_C(t) = \frac{1}{2}\Xi\left(\hat{\boldsymbol{q}}_C(t)\right)\hat{\boldsymbol{\omega}}_C(t), \qquad \dot{\hat{\boldsymbol{p}}}_C(t) = \hat{\boldsymbol{v}}_C(t), \qquad \dot{\hat{\boldsymbol{v}}}_C(t) = C^T\left(\hat{\boldsymbol{q}}_C(t)\right)\hat{\boldsymbol{a}}_C(t) \tag{4.45}$$

and the covariance propagation is

$$\dot{P}(t) = F(t)P(t) + P(t)F^T(t) + G(t)Q(t)G^T(t) \tag{4.46}$$

Note that there are 15 error states for the camera vehicle and another 15 for the target vehicle so the covariance matrix P is 30-by-30. The F and G matrices are defined

$$F(t) = \begin{bmatrix} F_C & 0_{15,15} \\ 0_{15,15} & 0_{15,15} \end{bmatrix}, \qquad G(t) = \begin{bmatrix} G_C & 0_{15,15} \\ 0_{15,15} & 0_{15,15} \end{bmatrix} \tag{4.47}$$

where

$$F_C(t) = \begin{bmatrix} -\left[\hat{\boldsymbol{\omega}}_C(t)\times\right] & 0 & 0 & -I_3 & 0 \\ 0 & 0 & I_3 & 0 & 0 \\ 0 & 0 & 0 & 0 & -C^T\left(\hat{\boldsymbol{q}}_C(t)\right) \\ 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 \end{bmatrix}, \qquad G_C(t) = \begin{bmatrix} -I_3 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & -C^T\left(\hat{\boldsymbol{q}}_C(t)\right) & 0 \\ 0 & I_3 & 0 & 0 \\ 0 & 0 & 0 & I_3 \end{bmatrix} \tag{4.48}$$

This reflects the fact that the camera IMU carries no information about the target states. For the case of the target states propagation, (4.47) is replaced by

$$F(t) = \begin{bmatrix} 0_{15,15} & 0_{15,15} \\ 0_{15,15} & F_T \end{bmatrix}, \qquad G(t) = \begin{bmatrix} 0_{15,15} & 0_{15,15} \\ 0_{15,15} & G_T \end{bmatrix} \tag{4.49}$$

where F_T and G_T are defined similarly to 4.48.

4.4 Kalman Update

Since the camera and GPS measurements come in at different times, a separate update is performed for each of the two. These update steps are described here.

4.4.1 Camera Update

The camera measurement at time t_k is

$$\tilde{\boldsymbol{y}}_k = \begin{bmatrix} \tilde{\boldsymbol{p}}_{T/C} \\ \tilde{\boldsymbol{n}} \end{bmatrix}\Bigg|_{t_k} = \begin{bmatrix} C_{C/T}\,\boldsymbol{p}_T - \boldsymbol{p}_C \\ C_{C/T}\,\boldsymbol{e}_3 \end{bmatrix}\Bigg|_{t_k} + \begin{bmatrix} \boldsymbol{\nu}_p \\ \boldsymbol{\nu}_n \end{bmatrix}\Bigg|_{t_k} \tag{4.50}$$

where $\boldsymbol{e}_3 = \begin{bmatrix} 0 & 0 & 1 \end{bmatrix}^T$ and $C_{C/T} = C(\boldsymbol{q}_C)C^T(\boldsymbol{q}_T)$. For any vector $\boldsymbol{w}$, the partial derivatives of $C_{C/T}\,\boldsymbol{w}$ with respect to the orientation errors $\boldsymbol{\alpha}_C$ and $\boldsymbol{\alpha}_T$ are

$$\begin{aligned}\frac{\partial}{\partial \boldsymbol{\alpha}_C} C_{C/T}\,\boldsymbol{w} &= \frac{\partial}{\partial \boldsymbol{\alpha}_C} C(\boldsymbol{\delta q}_C) C(\hat{\boldsymbol{q}}_C^-) C^T(\hat{\boldsymbol{q}}_T^-)\boldsymbol{w} \\ &= \frac{\partial}{\partial \boldsymbol{\alpha}_C}\left(I - [\boldsymbol{\alpha}_C \times]\right) C(\hat{\boldsymbol{q}}_C^-) C^T(\hat{\boldsymbol{q}}_T^-)\boldsymbol{w} \\ &= \left[C(\hat{\boldsymbol{q}}_C^-) C^T(\hat{\boldsymbol{q}}_T^-)\boldsymbol{w}\times\right] \\ &= \left[\hat{C}_{C/T}\,\boldsymbol{w}\times\right] \end{aligned} \tag{4.51}$$

and

$$\begin{aligned}\frac{\partial}{\partial \boldsymbol{\alpha}_T} C_{C/T}\,\boldsymbol{w} &= \frac{\partial}{\partial \boldsymbol{\alpha}_T} C(\hat{\boldsymbol{q}}_C^-) C^T(\hat{\boldsymbol{q}}_T^-) C^T(\boldsymbol{\delta q}_T)\boldsymbol{w} \\ &= \frac{\partial}{\partial \boldsymbol{\alpha}_T} C(\hat{\boldsymbol{q}}_C^-) C^T(\hat{\boldsymbol{q}}_T^-)\left(I - [\boldsymbol{\alpha}_T \times]\right)^T \boldsymbol{w} \\ &= \frac{\partial}{\partial \boldsymbol{\alpha}_T} C(\hat{\boldsymbol{q}}_C^-) C^T(\hat{\boldsymbol{q}}_T^-)\left(I + [\boldsymbol{\alpha}_T \times]\right)\boldsymbol{w} \\ &= -C(\hat{\boldsymbol{q}}_C^-) C^T(\hat{\boldsymbol{q}}_T^-)\left[\boldsymbol{w}\times\right] \\ &= -\hat{C}_{C/T}\left[\boldsymbol{w}\times\right] \end{aligned} \tag{4.52}$$

where $\hat{C}_{C/T} = C(\hat{\boldsymbol{q}}_C^-) C^T(\hat{\boldsymbol{q}}_T^-)$. Thus the measurement sensitivity matrix is

$$\begin{aligned} H_k &= \begin{bmatrix} \frac{\partial \boldsymbol{p}_{T/C}}{\boldsymbol{\Delta x}_C} & \frac{\partial \boldsymbol{p}_{T/C}}{\boldsymbol{\Delta x}_T} \\ \frac{\partial \boldsymbol{n}}{\boldsymbol{\Delta x}_C} & \frac{\partial \boldsymbol{n}}{\boldsymbol{\Delta x}_T} \end{bmatrix} \\ &= \left.\begin{bmatrix} \left[\hat{C}_{C/T}\,\hat{\boldsymbol{p}}_T^-\times\right] & -I & 0_{3,9} & -\hat{C}_{C/T}\left[\hat{\boldsymbol{p}}_T^-\times\right] & \hat{C}_{C/T} & 0_{3,9} \\ \left[\hat{C}_{C/T}\,\boldsymbol{e}_3\times\right] & 0_{3,3} & 0_{3,9} & -\hat{C}_{C/T}\left[\boldsymbol{e}_3\times\right] & 0_{3,3} & 0_{3,9} \end{bmatrix}\right|_{t_k} \end{aligned} \tag{4.53}$$

The Kalman gain is defined as

$$K_k = P_k^- H_k^T \left(H_k P_k^- H_k^T + R_k\right)^{-1} \tag{4.54}$$

where the camera measurement error matrix is

$$R_k = \begin{bmatrix} \sigma_p^2 I & 0_{3,3} \\ 0_{3,3} & \sigma_n^2 I \end{bmatrix} \tag{4.55}$$

4.4.2 GPS Update

It is assumed that the GPS devices for the two vehicles are in sync and provide estimates at the same time. The GPS measurement at time t_k is

$$\tilde{\boldsymbol{y}}_k = \left.\begin{bmatrix} \tilde{\boldsymbol{p}}_C \\ \tilde{\boldsymbol{p}}_T \end{bmatrix}\right|_{t_k} = \left.\begin{bmatrix} \boldsymbol{p}_C \\ \boldsymbol{p}_T \end{bmatrix}\right|_{t_k} + \left.\begin{bmatrix} \boldsymbol{\nu}_C \\ \boldsymbol{\nu}_T \end{bmatrix}\right|_{t_k} \tag{4.56}$$

The measurement sensitivity matrix is therefore

$$H_k = \begin{bmatrix} 0_{3,3} & I & 0_{3,9} & 0_{3,3} & 0_{3,3} & 0_{3,9} \\ 0_{3,3} & 0_{3,3} & 0_{3,9} & 0_{3,3} & I & 0_{3,9} \end{bmatrix} \tag{4.57}$$

The Kalman gain is defined as before to be

$$K_k = P_k^- H_k^T \left(H_k P_k^- H_k^T + R_k \right)^{-1} \tag{4.58}$$

where the GPS measurement error matrix is

$$R_k = \begin{bmatrix} \sigma_C^2 I & 0_{3,3} \\ 0_{3,3} & \sigma_T^2 I \end{bmatrix} \tag{4.59}$$

The full filter implementation is tabulated in Table 4.1.

Initialize	$\hat{\boldsymbol{q}}_C(t_0) = \hat{\boldsymbol{q}}_{C0}, \qquad \hat{\boldsymbol{p}}_C(t_0) = \hat{\boldsymbol{p}}_{C0}, \qquad \hat{\boldsymbol{v}}_C(t_0) = \hat{\boldsymbol{v}}_{C0}$ $\hat{\boldsymbol{\beta}}_{Cg}(t_0) = \hat{\boldsymbol{\beta}}_{Cg0}, \qquad \hat{\boldsymbol{\beta}}_{Ca}(t_0) = \hat{\boldsymbol{\beta}}_{Ca0}$ $\hat{\boldsymbol{q}}_T(t_0) = \hat{\boldsymbol{q}}_{T0}, \qquad \hat{\boldsymbol{p}}_T(t_0) = \hat{\boldsymbol{p}}_{T0}, \qquad \hat{\boldsymbol{v}}_T(t_0) = \hat{\boldsymbol{v}}_{T0}$ $\hat{\boldsymbol{\beta}}_{Tg}(t_0) = \hat{\boldsymbol{\beta}}_{Tg0}, \qquad \hat{\boldsymbol{\beta}}_{Ta}(t_0) = \hat{\boldsymbol{\beta}}_{Ta0}, \qquad P(t_0) = P_0$
Propagate	$\hat{\boldsymbol{\omega}}_C = \tilde{\boldsymbol{\omega}}_C - \hat{\boldsymbol{\beta}}_{Cg}, \qquad \hat{\boldsymbol{a}}_C = \tilde{\boldsymbol{a}}_C - \hat{\boldsymbol{\beta}}_{Ca}$ $\dot{\hat{\boldsymbol{q}}}_C(t) = \frac{1}{2}\Xi\left(\hat{\boldsymbol{q}}_C(t)\right)\hat{\boldsymbol{\omega}}_C(t), \qquad \dot{\hat{\boldsymbol{p}}}_C(t) = \hat{\boldsymbol{v}}_C(t), \qquad \dot{\hat{\boldsymbol{v}}}_C(t) = C^T\left(\hat{\boldsymbol{q}}_C(t)\right)\hat{\boldsymbol{a}}_C(t)$ $\dot{P}(t) = F(t)P(t) + P(t)F^T(t) + G(t)Q(t)G^T(t)$ $F_C(t) = \begin{bmatrix} -\left[\hat{\boldsymbol{\omega}}_C(t)\times\right] & 0 & 0 & -I_3 & 0 \\ 0 & 0 & I_3 & 0 & 0 \\ 0 & 0 & 0 & 0 & -C^T\left(\hat{\boldsymbol{q}}_C(t)\right) \\ 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 \end{bmatrix}, F(t) = \begin{bmatrix} F_C & 0_{15,15} \\ 0_{15,15} & 0_{15,15} \end{bmatrix}$ $G_C(t) = \begin{bmatrix} -I_3 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & -C^T\left(\hat{\boldsymbol{q}}_C(t)\right) & 0 \\ 0 & I_3 & 0 & 0 \\ 0 & 0 & 0 & I_3 \end{bmatrix}, G(t) = \begin{bmatrix} G_C & 0_{15,15} \\ 0_{15,15} & 0_{15,15} \end{bmatrix}$ and similarly for propagating the $\boldsymbol{x}_T$ states
Gain (Camera)	$K_k = P_k^- H_k^T(\hat{\boldsymbol{x}}_k^-)\left[H_k(\hat{\boldsymbol{x}}_k^-)P_k^- H_k^T(\hat{\boldsymbol{x}}_k^-) + R_k\right]^{-1}$ $H_k(\hat{\boldsymbol{x}}_k^-) = \left.\begin{bmatrix} \left[\hat{C}_{C/T}\,\hat{\boldsymbol{p}}_T^-\times\right] & -I & 0_{3,9} & -\hat{C}_{C/T}\left[\hat{\boldsymbol{p}}_T^-\times\right] & \hat{C}_{C/T} & 0_{3,9} \\ \left[\hat{C}_{C/T}\,\boldsymbol{e}_3\times\right] & 0_{3,3} & 0_{3,9} & -\hat{C}_{C/T}\left[\boldsymbol{e}_3\times\right] & 0_{3,3} & 0_{3,9} \end{bmatrix}\right\rvert_{t_k}$ $\hat{C}_{C/T} = C(\hat{\boldsymbol{q}}_C^-)C^T(\hat{\boldsymbol{q}}_T^-)$ $R_k = \begin{bmatrix} \sigma_p^2 I & 0_{3,3} \\ 0_{3,3} & \sigma_n^2 I \end{bmatrix}$ $\boldsymbol{h}_k(\hat{\boldsymbol{x}}_k^-) = \begin{bmatrix} \hat{C}_{C/T}\,\hat{\boldsymbol{p}}_T^- - \hat{\boldsymbol{p}}_C^- \\ \hat{C}_{C/T}\,\boldsymbol{e}_3 \end{bmatrix}$
Gain (GPS)	$K_k = P_k^- H_k^T(\hat{\boldsymbol{x}}_k^-)\left[H_k(\hat{\boldsymbol{x}}_k^-)P_k^- H_k^T(\hat{\boldsymbol{x}}_k^-) + R_k\right]^{-1}$ $H_k(\hat{\boldsymbol{x}}_k^-) = \begin{bmatrix} 0_{3,3} & I & 0_{3,9} & 0_{3,3} & 0_{3,3} & 0_{3,9} \\ 0_{3,3} & 0_{3,3} & 0_{3,9} & 0_{3,3} & I & 0_{3,9} \end{bmatrix}$ $R_k = \begin{bmatrix} \sigma_C^2 I & 0_{3,3} \\ 0_{3,3} & \sigma_T^2 I \end{bmatrix}$ $\boldsymbol{h}_k(\hat{\boldsymbol{x}}_k^-) = \begin{bmatrix} \hat{\boldsymbol{p}}_C^- \\ \hat{\boldsymbol{p}}_T^- \end{bmatrix}$
Update	$P_k^+ = \left[I - K_k H_k(\hat{\boldsymbol{x}}_k^-)\right]P_k^-$ $\boldsymbol{\Delta x}_k = K_k\left[\tilde{\boldsymbol{y}}_k - \boldsymbol{h}_k(\hat{\boldsymbol{x}}_k^-)\right]$ $\boldsymbol{\Delta x}_k = \begin{bmatrix} \boldsymbol{\delta\alpha}_k^T & \boldsymbol{\Delta p}_k^T & \boldsymbol{\Delta v}_k^T & \boldsymbol{\Delta\beta}_{gk}^T & \boldsymbol{\Delta\beta}_{ak}^T \end{bmatrix}^T$ $\hat{\boldsymbol{q}}_k^+ = \hat{\boldsymbol{q}}_k^- + \frac{1}{2}\Xi(\hat{\boldsymbol{q}}_k^-)\boldsymbol{\alpha}_k, \qquad \hat{\boldsymbol{p}}_k^+ = \hat{\boldsymbol{p}}_k^- + \boldsymbol{\Delta p}_k, \qquad \hat{\boldsymbol{v}}_k^+ = \hat{\boldsymbol{v}}_k^- + \boldsymbol{\Delta v}_k$ $\hat{\boldsymbol{\beta}}_{gk}^+ = \hat{\boldsymbol{\beta}}_{gk}^- + \boldsymbol{\Delta\beta}_{gk}, \qquad \hat{\boldsymbol{\beta}}_{ak}^+ = \hat{\boldsymbol{\beta}}_{ak}^- + \boldsymbol{\Delta\beta}_{ak}$

Table 4.1: Automated Aerial Refueling MEKF implementation.

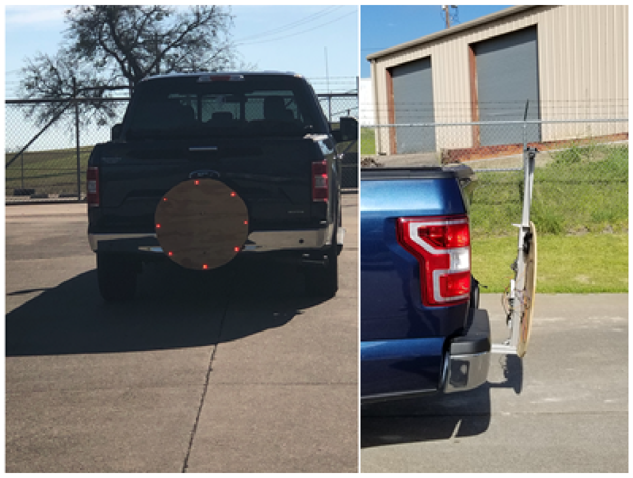


Figure 4.6: LED beacons set up for ground testing

4.5 Application to Field Testing

The MEKF was tested using two automobiles. The ring of LED beacons were affixed to a leading vehicle, and the camera was placed on the following vehicle. Both cars had and IMU and a GPS on board. The vehicles drove along a road and the MEKF provided real-time estimates of their pose.

The estimate errors for the pose of the leading vehicle are displayed in Figs. 4.7, 4.8, and 4.9. The truth model for the errors is provided by a Rauch–Tung–Striebel (RTS) smoother [9]. This utilizes posterior information about the states to update earlier state estimates and provides more accurate knowledge than a real time filter. Fig. 4.7 shows that the attitude estimates benefit particularly from the inclusion of the vision estimates.

The estimated position of the leading vehicle is overlaid on a GPS map of the area in Fig. (4.10). Since the GPS updates themselves are already very accurate and can therefore make it hard to see the effect of the vision updates, Fig. (4.11) shows a similar graphic for the position estimates with GPS updates suppressed. The blue line shows what happens when vision updates are also suppressed starting halfway through the test. The IMU builds up a bias that causes the position estimates to drift over time. The red line shows that the vision system is sufficient to correct for IMU biases even without GPS measurements.

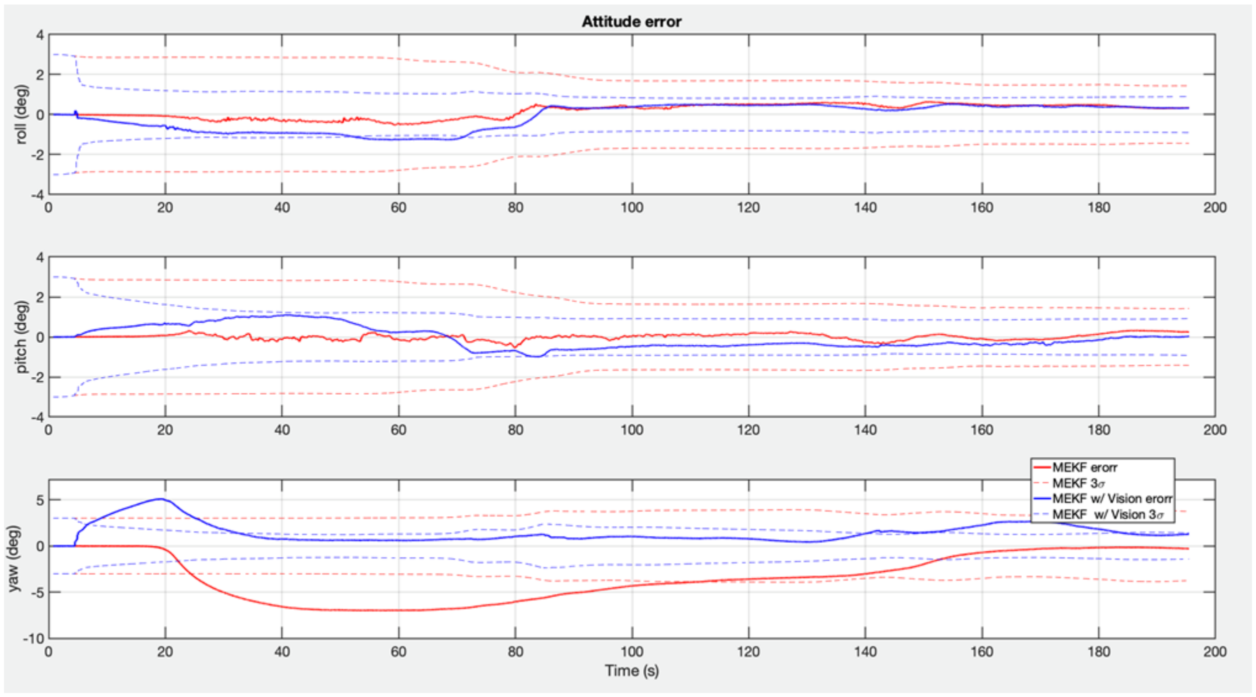


Figure 4.7: The attitude estimate errors of the leading vehicle. The red line shows the error with computer vision updates suppressed, and the blue line shows the error with vision updates included.

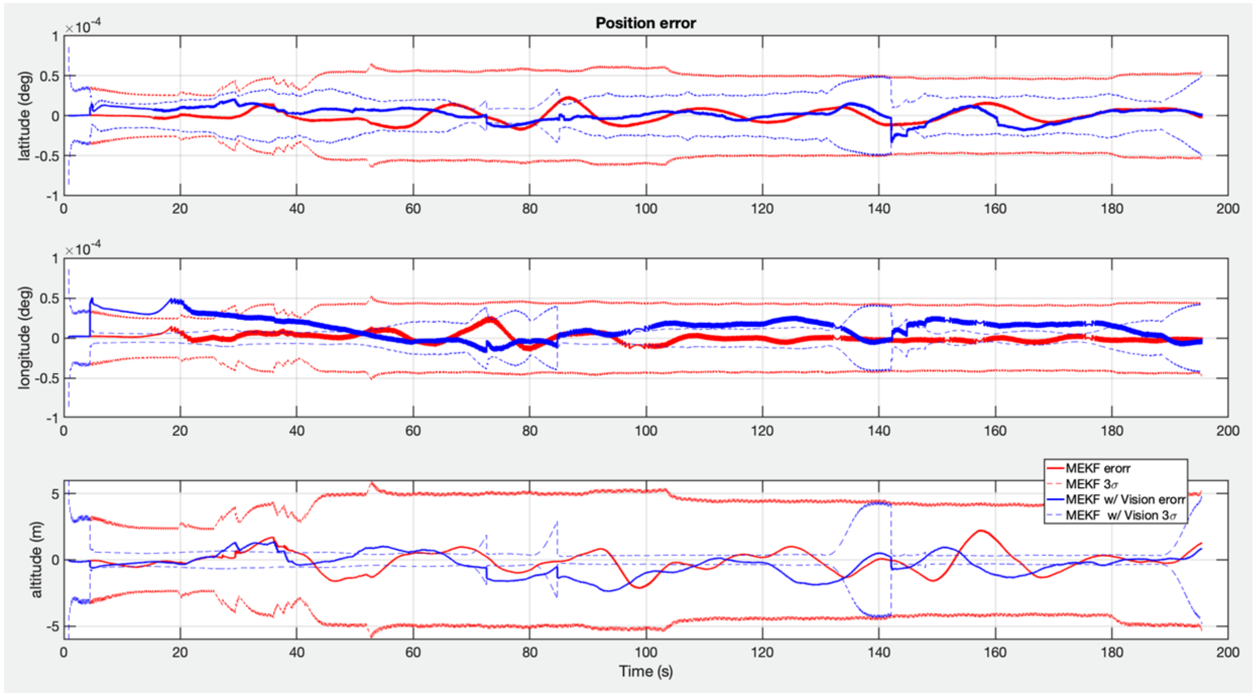


Figure 4.8: The position errors of the leading vehicle.

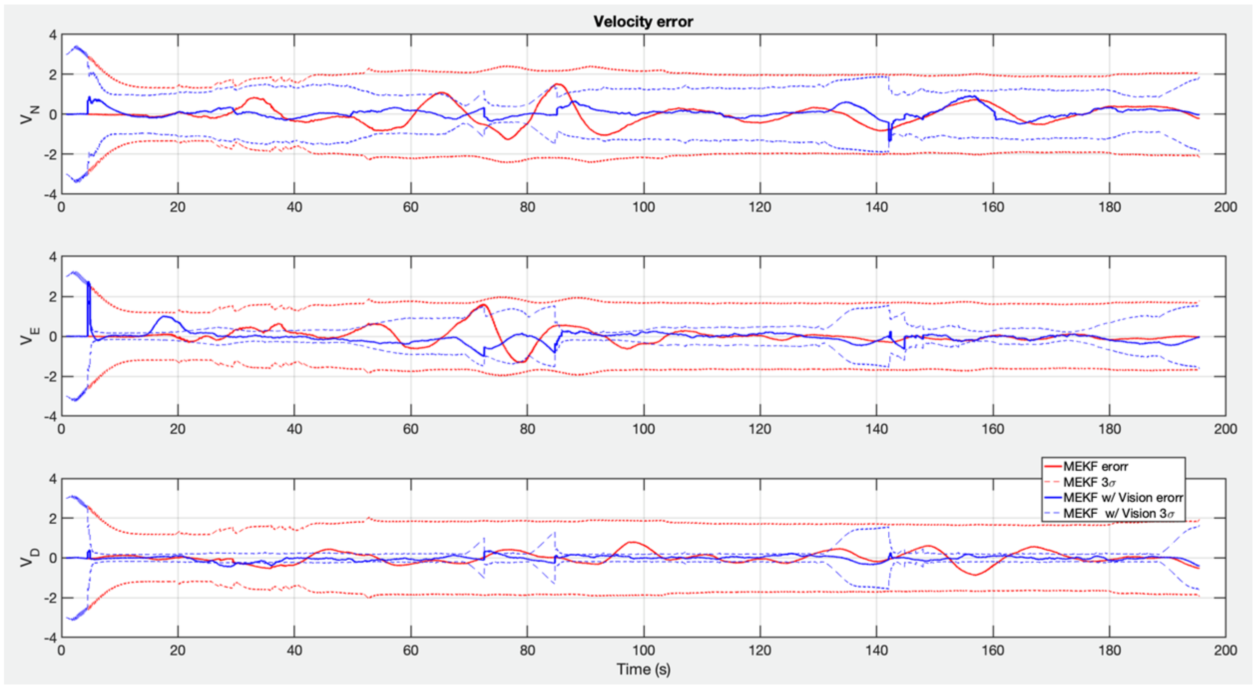


Figure 4.9: The velocity errors of the leading vehicle.

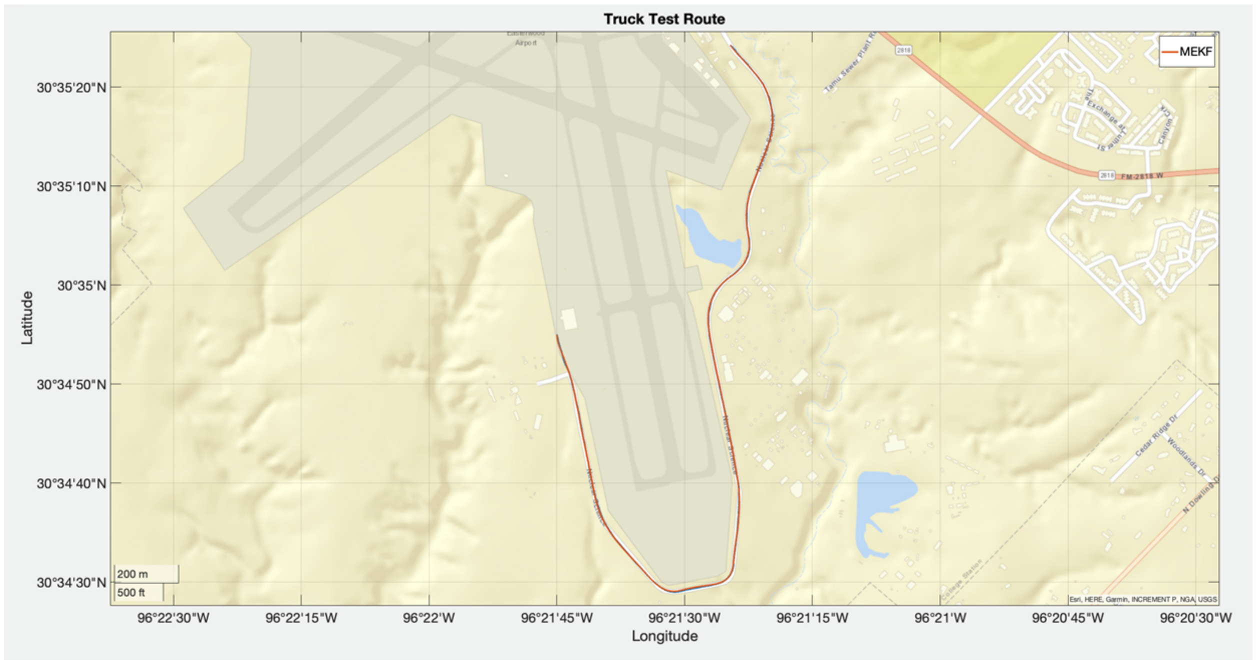


Figure 4.10: Position estimate for leading vehicle throughout ground test.

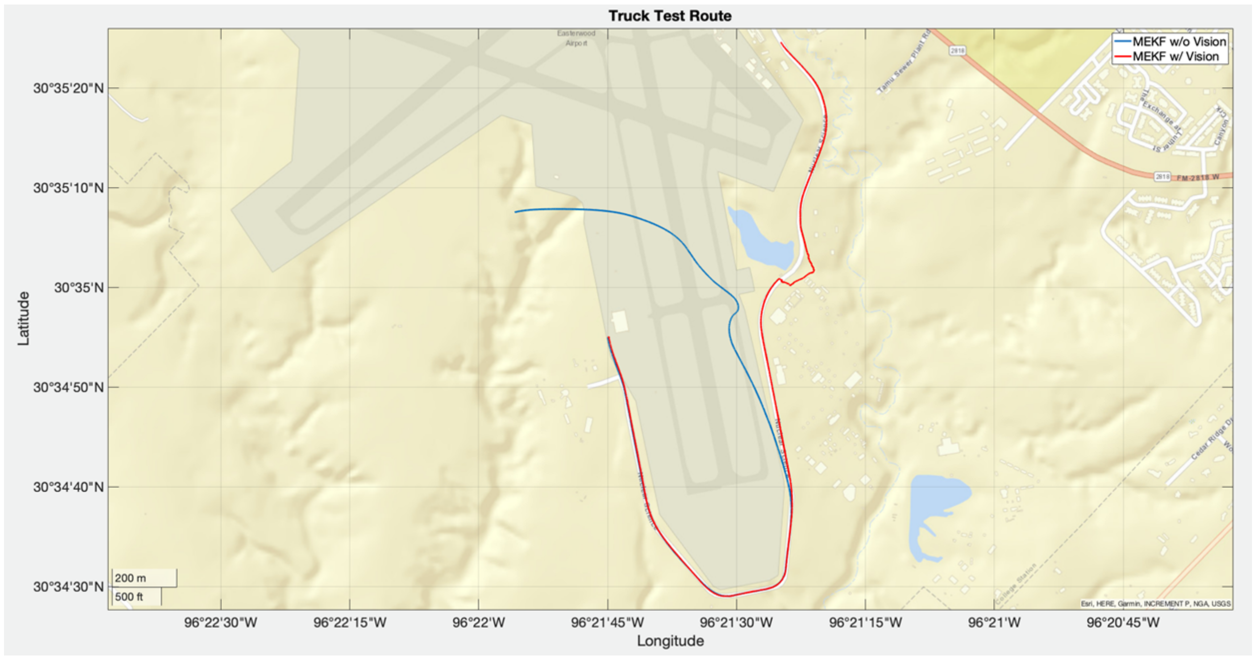


Figure 4.11: Position estimate with GPS updates suppressed. The red line includes vision updates while the red line suppresses vision updates after the halfway point of the test.

4.6 Application to Testing in a Laboratory Environment

The filter was tested at LASR lab using two of the in-house robotic platforms to represent two bodies. In this test, the camera and a probe are mounted on the Holonomic Omni-directional Motion Emulation Robot (HOMER) and Height Azimuth, and Elevation Manipulator (HAZEL), which together provide motion in six degrees of freedom. The target nozzle with its ring of LED lights are attached to a long flexible hose, which is mounted on a third platform, the Suspended Target Emulation Pendumlum (STEP), which provides motion in five degrees of freedom.

The target nozzle moves in a sinusoidal path, and the MEKF derived above collects images and provides estimates of the relative pose. This estimate is used as an input to a PID controller which seeks to insert the probe into the hose by driving their relative distance to zero. The setup of the test is shown in Fig. 4.12.

A truth model for the bodies' positions is provided by a Vicon motion capture system.

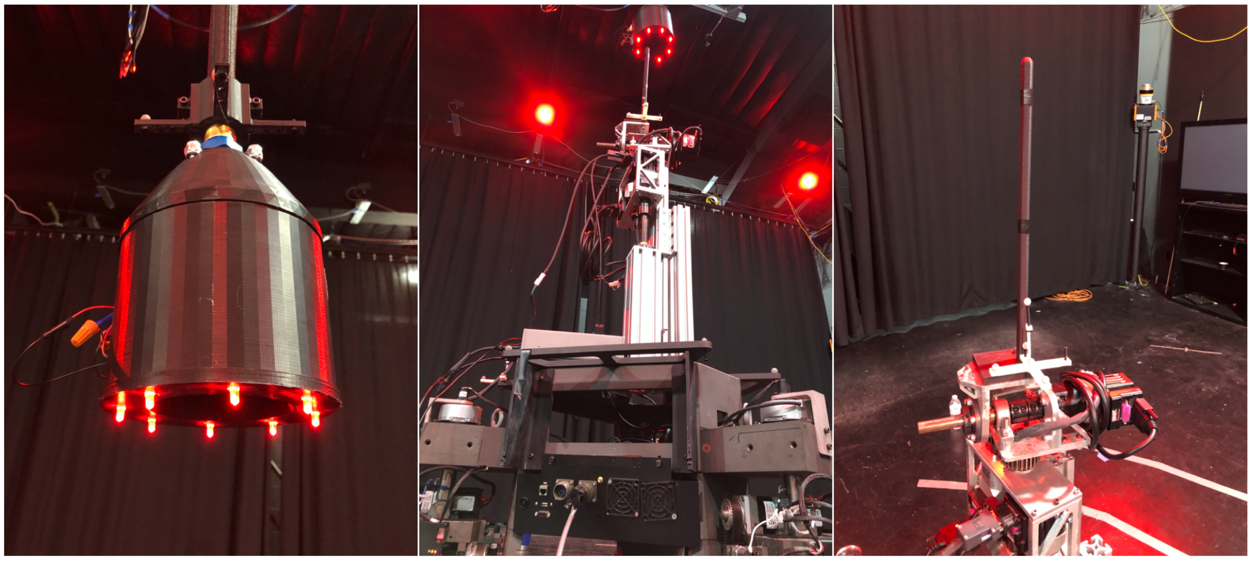


Figure 4.12: The setup for laboratory testing of the MEKF. (Left): The target nozzle and LED ring mounted on STEP. (Middle): The probe being inserted into the target nozzle by the controller. (Right): the probe mounted on HAZEL.

The results from a full run are graphed in Fig. 4.13. This demonstrates the MEKF is sufficient to provide accurate estimates that allow the controller to closely track the target's motion. There is a small lag time, but the figure shows that the controller is responsible.

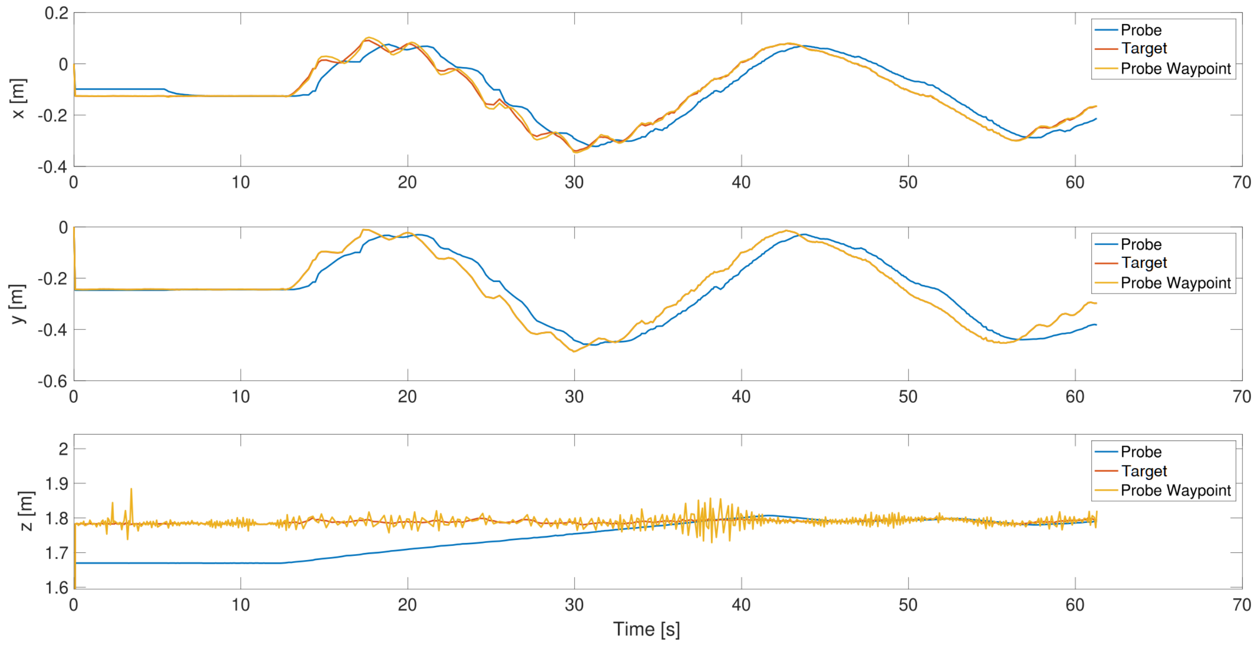


Figure 4.13: Position of probe and target for a full laboratory test. The probe waypoint line shows the estimated target position at each time, which represents the output of the MEKF estimator.

5. SPACECRAFT CAPTURE OPERATIONS*

The topic of Active Debris Removal (ADR) has gained significant attention over the past few decades. As the number of objects in Low Earth Orbit (LEO) increases from year to year, we come closer and closer to the "Kessler Syndrome" first described by Kessler and Cour-Palais [14] in 1978. In order to prevent this Kessler Syndrome from becoming a reality two things need to be done. Firstly, the number of new debris objects being put into LEO from launches needs to be reduced. However, this alone is not enough to prevent the problem; studies have indicated that at least 5 large LEO debris objects must be removed each year (along with no further release) in order to stabilize LEO for the future [15]. It is well known in the community that there are of the order of 600 US rocket bodies and about 1500 Russian spent rocket boosters in LEO that are considered to be "large". The national studies advise the removal of these objects as a priority. Analyses done by Barbee et al. [16] identify a group of high risk, high mass objects and postulate approaches to plan a sequence of missions to deorbit groups of these bodies.

Various researchers have focused on ADR technques to plan trajectories to and carry out autonomous rendezvous and proximity operations with debris objects [17, 18]. However, a detailed concept of operations on *how to* deorbit large bodies is a topic of research. For large objects in an unknown flat-spin state, the challenge of carrying out rendezvous and proximity operations becomes compounded by the highly dynamic environment of relative motion. Even to track the large bodies, it is frequently useful to tag them using radio transponder packages. In the light of these important challenges involving large spinning bodies, researchers at Land, Air and Space Robotics (LASR) laboratory are actively engaged in developing innovative tools for identification of rotation rate of a spinning rigid bodies

*Part of this chapter reprinted with permission from "Autonomous deployment of payload packages to spinning rocket bodies: approach, apparatus, and emulation using ground robotics." by C. Peck, D. Adams, J. McElreath, A. Verras, J. Hiemerl, M. Majji, M. Benedict, and J. Junkins, 2020. *2020 AAS Guidance, Navigation, and Control Conference*, Copyright 2020 by the authors of the paper.

using vision based navigation sensors, and autonomously deploy payload packages using robotic satellite technologies.

The goal of the algorithm in this chapter is for a vehicle, carrying the camera, to estimate the position and angular velocity of a tumbling rigid body, and then perform a soft capture. Specifically, the target is considered to be a rotating rocket body with a circular nozzle of known radius. It is assumed that the spacecraft has successfully approached the target such that it is within range of on-board camera and payload delivery systems. The deployed payload enters the rocket body and engages a spring-loaded catch to maintain insertion. Once deployed, the payload provides a tethered connection between the vehicle and its target, which then allows for a de-orbiting procedure to be initiated. The precise method of de-orbiting is outside the scope of this project.

Ground testing of capture is performed using the Automated Harpoon and Braking System (AHAB) designed by LASR Lab. AHAB houses the on-board computer which receives camera data, performs the motion tracking algorithm and executes the payload delivery process. The payload delivery is performed using AHAB's pneumatic cannon. The cannon accepts gas via a small cartridge and stores it in a pressure-regulated reservoir until a fire command is sent from the computer. An schematic of AHAB's cannon is shown in Fig. 5.1 below.

This payload is designed for insertion into a specified site on the tumbling rigid body. In the current instantiation, this site is assumed to be a cavity resembling a combustion chamber and a nozzle typically present in a rocket body. The AHAB deployable projectile then becomes a mechanism to complete a tethered soft capture with the spacecraft. The process of insertion was chosen over an object piercing the target to preserve the structural and/or operational integrity of both the target and the projectile. Further, it was speculated that benign operation would increase reliability of a good engagement, reduce the risk of producing debris, and eliminate the need for a hefty projectile that was meant to pierce metal.

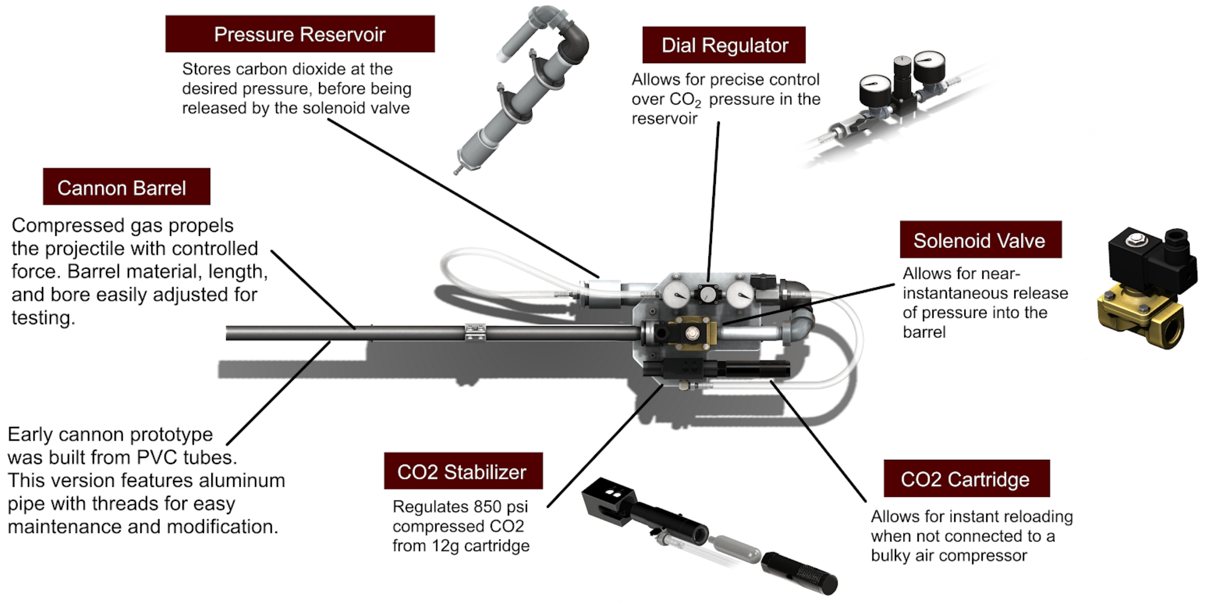

Pressure Reservoir
Stores carbon dioxide at the desired pressure, before being released by the solenoid valve
Dial Regulator
Allows for precise control over CO_2 pressure in the reservoir
Cannon Barrel
Compressed gas propels the projectile with controlled force. Barrel material, length, and bore easily adjusted for testing.
Solenoid Valve
Allows for near-instantaneous release of pressure into the barrel
Early cannon prototype was built from PVC tubes. This version features aluminum pipe with threads for easy maintenance and modification.
CO2 Stabilizer
Regulates 850 psi compressed CO2 from 12g cartridge
CO2 Cartridge
Allows for instant reloading when not connected to a bulky air compressor

Figure 5.1: AHAB air cannon schematic.

As in the aerial refueling chapter, the target has a circular shape. Thus the pose will once again be determined from the parameters of the projected ellipse. However, unlike that situation, the target is not marked by beacons. Therefore an algorithm must be used to detect an ellipse in the image. This algorithm is detailed in the next section. Once ellipses are detected in the image, some knowledge of the target's size and approximate position is used to pinpoint which ellipse represents the target nozzle. For instance, ellipses with too large or small semi-major axes can be discarded, until only the ellipse that represents the target nozzle remains.

5.1 Ellipse Detection

The algorithm used for ellipse detection comes from Libuda, et. al., (2006) [1]. It begins with an image of just edges, e.g. from a Canny edge detector. From there, it groups edge pixels into segments, segments into lines, lines into arcs, arcs into elliptic arcs, and elliptic arcs into ellipses. This step-by step grouping is pictured in Fig. 5.2.

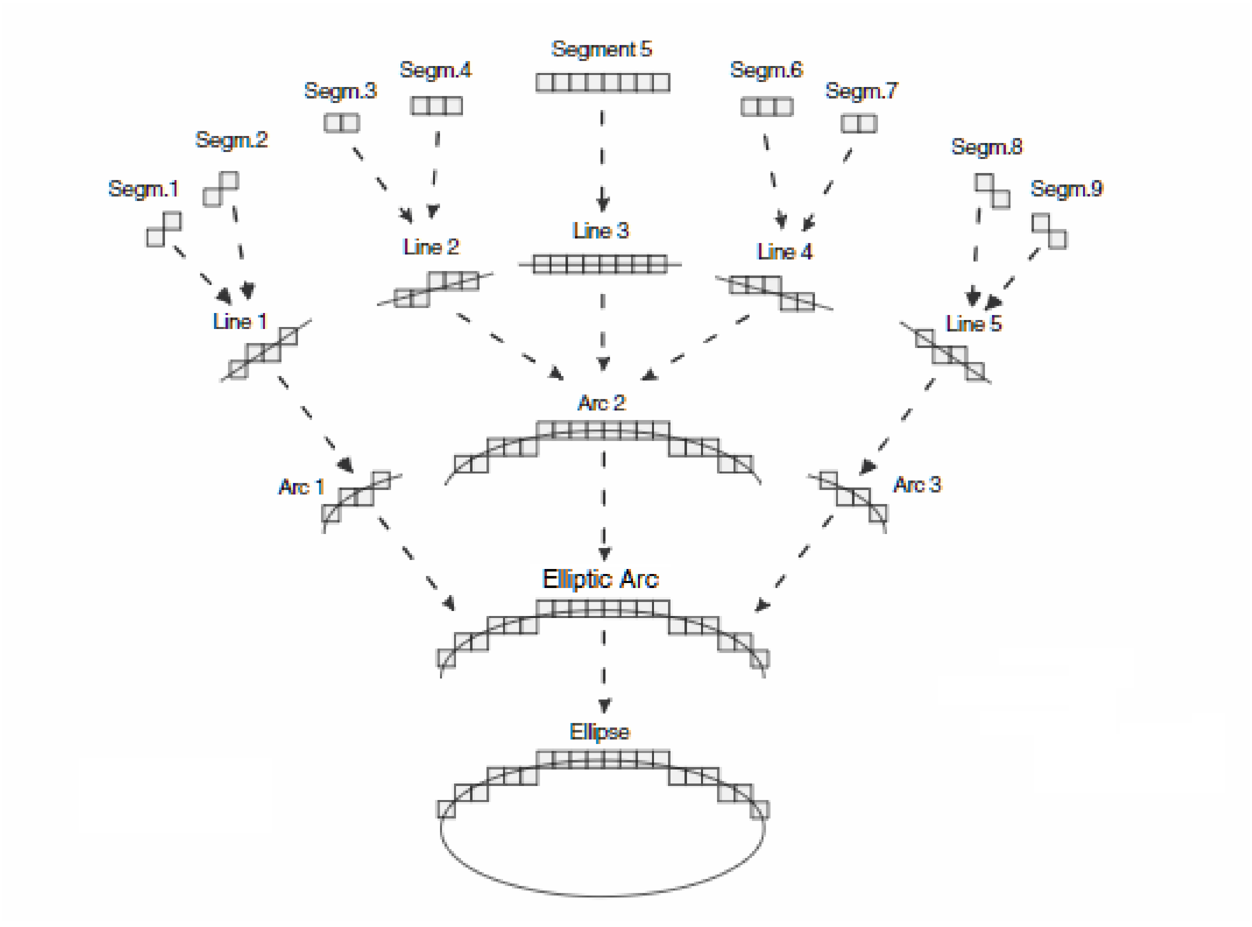


Figure 5.2: The different groupings used in the ellipse detection algorithm [1].

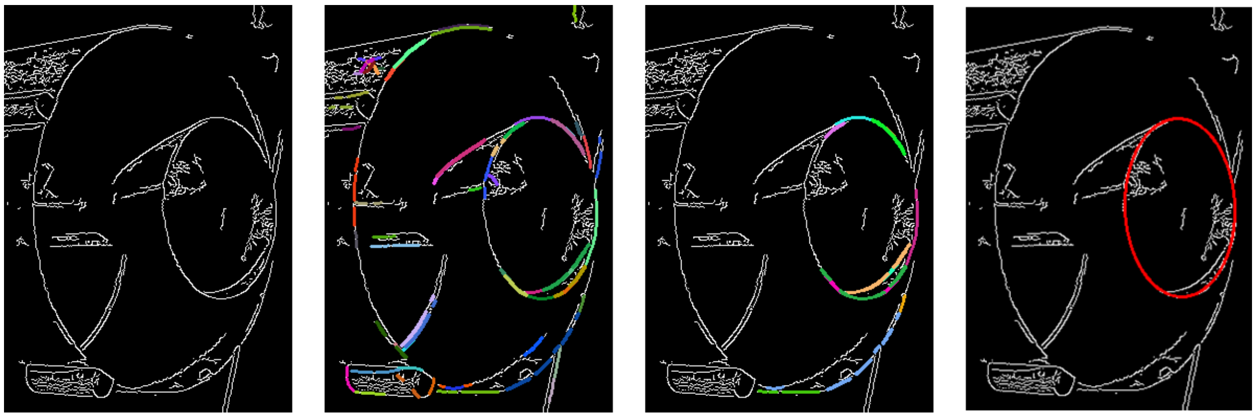


Figure 5.3: A sample run of the ellipse detection algorithm showing, from left to right: the edgemap, the detected arcs, the detected elliptic arcs, and the detected ellipse.

Each segment lies entirely vertically, horizontally, or at $\pm 45°$. Each line is made up of two or more segments that, together, have a coherent slope. The line extraction algorithm comes from Kim, et. al., (2003) [19]. Each arc is made up of two or more lines that form approximately a section of a circle. The arc extraction algorithm is from Thomas and Chan, (1989) [20]. Finally, each elliptic arc is made up of three neighboring arcs that are consistent with the same ellipse. If there are enough elliptic arcs to cover a large enough portion (25% by default) of a particular ellipse, that ellipse's parameters are determined using the algorithm from Section 4.2.1.1.

An example run of the algorithm which shows the detected segments, arcs, etc. is pictured in Fig. 5.3. The idea is that there are fewer objects detected as you proceed down the hierarchy of Fig. 5.2. This way, the more expensive tests that occur toward the bottom (e.g., testing if arcs belong to the same ellipse) are performed on fewer objects. This allows the algorithm to run quickly while still being thorough.

There are several parameters involved in running the algorithm, but four are indicated as being most important. Three of those are pictured in Figs. 5.4, 5.5, and 5.6. First, there is Θ_{err} (Fig. 5.4), which dictates how far a line in an arc can diverge from the tangent of a circle. Its default value is 18°. Second is D_{arc} (Fig. 5.5), which indicates how far apart

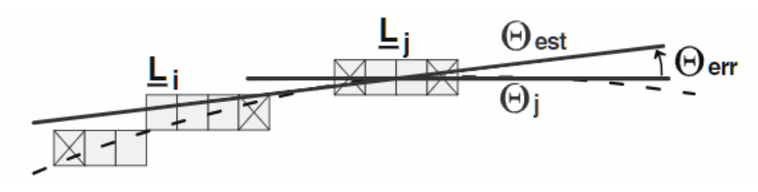


Figure 5.4: If a line L_j is to be included in an arc, its angle Θ_j must differ from the predicted tangent angle of the circle Θ_{est} by no more than Θ_{err} [1].

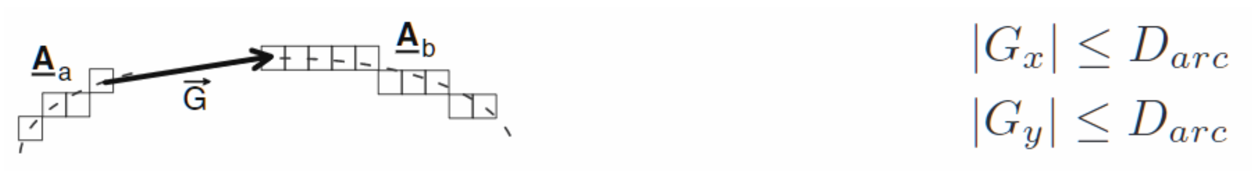
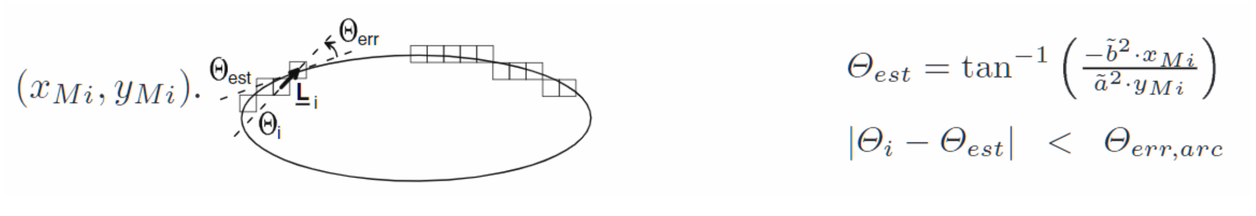


Figure 5.5: If two arcs are to be included in an elliptic arc, neither their horizontal nor their vertical distance may exceed D_{arc} [1].

arcs can be to be counted in an elliptic arc. Its default value is 37 pixels. Third is $\Theta_{err,arc}$, which restricts how far a line in an ellipse can diverge from the tangent of the ellipse. Its default value is 14°. Finally, C_{min} indicates how much of the full ellipse circumference must be included in order to say the ellipse was detected. The default is 25%, however, this doesn't mean the program will be able to find quarter-ellipses easily. It seems at its best, the algorithm will only find about 70% of the portion of ellipse that is visible.



(x_{Mi}, y_{Mi}).
Θ_{est}
Θ_{err}
$\underline{L}_i$
Θ_i
$\Theta_{est} = \tan^{-1}\left(\frac{-\tilde{b}^2 \cdot x_{Mi}}{\tilde{a}^2 \cdot y_{Mi}}\right)$
$|\Theta_i - \Theta_{est}| < \Theta_{err,arc}$

Figure 5.6: If an arc line in an ellipse has angle Θ_i and the predicted tangent of the ellipse at the midpoint of that arc is Θ_{est}, then the difference between those two angles must be below $\Theta_{err,arc}$ [1].

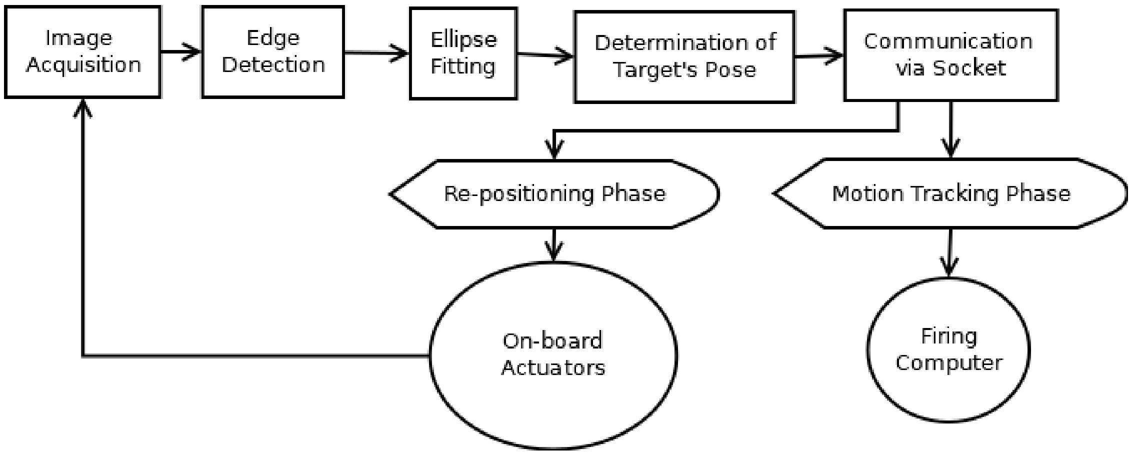


Figure 5.7: Computer vision algorithm flowchart.

5.2 Tracking and Capture

The camera is mounted near the end of the cannon barrel and is connected to a tracking computer. This computer communicates with the vehicle's on-board actuators remotely on the network via a socket program. This communication engages the actuators, which then control the cannon's azimuth and height. It also runs programs that initiate the firing sequence and the firing mechanisms based on certain confidence criteria from the algorithms.

The computer vision algorithm is in three phases. Phase 1 is the positioning step, where the vehicle lines itself up with the target. Phase 2 is the estimation of the target's angular velocity. Finally, phase 3 is the determination of a "fire time" to deploy the payload package when the target is in line with the vehicle. A flowchart depicting this algorithm is given in Fig. 5.7.

In the ground testing, the cannon barrel starts at an arbitrary azimuth and height. Thus, the purpose of the re-positioning phase is to determine the target's center of rotation. The length, L, of the rotating target nozzle is known. The ellipse fit on each image allows the software to obtain the target nozzle's position vector, $\mathbf{t}$, relative to the camera and its unit body vector, $\hat{\mathbf{b}}$. This information is sufficient to form an estimate for the center of rotation, $\mathbf{r_0}$, given by

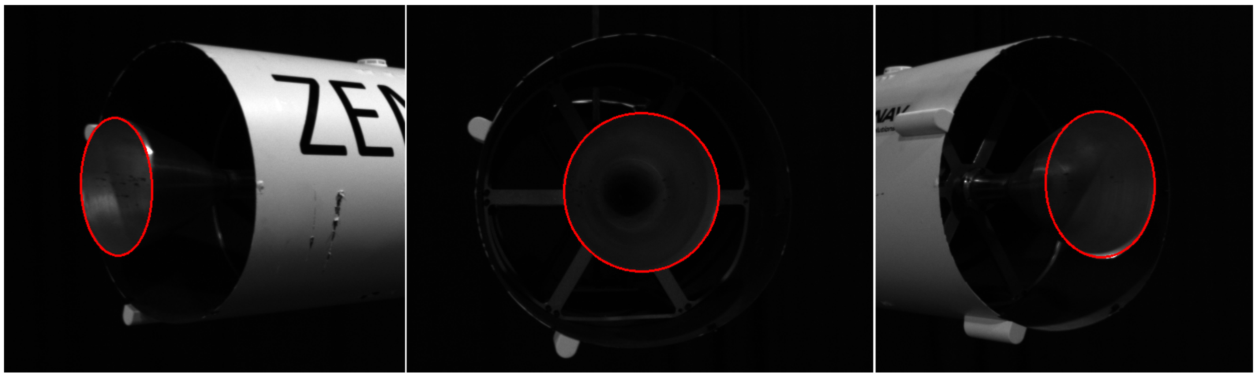


Figure 5.8: Sample ellipses fit in real-time by AHAB tracking software.

$$\mathbf{r_0} = \mathbf{t} - L\hat{\mathbf{b}} \tag{5.1}$$

Once the center of rotation is obtained, a desired azimuth and height change is sent to the on-board actuators to re-position the barrel. This procedure is repeated until the cannon is pointing at the center of rotation.

When the cannon is in position, the tracking software begins estimating the target's angular velocity. The body vector of the rotating nozzle is sufficient to compute the angle the nozzle makes with the line of the camera. The relative nozzle angle is obtained at various points in time to compute the average angular velocity during each time interval. A moving average is used to update the angular velocity estimate. Once enough data have been obtained, the software sends a firing signal to the cannon with a calibrated latency based on the harpoon's travel time.

5.3 Application to Testing in a Laboratory Environment

Some samples of the detected ellipses are shown in Fig. 5.8. These samples are from the ellipse detection algorithm running in real-time. The algorithm was shown to be able to successfully re-position the vehicle's cannon and send a firing signal at the proper time. Fig. 5.9 shows an image taken at a time the algorithm predicts the target is in line with the cannon. The red circle is overlaid where the projectile would land. Ground testing revealed

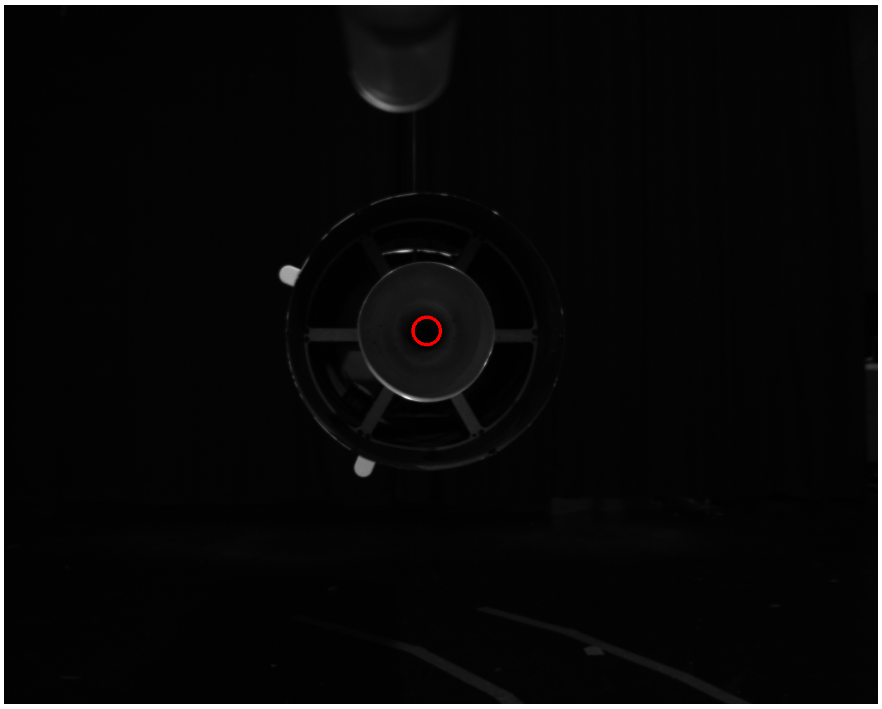


Figure 5.9: Image taken when the tracking algorithm determines target nozzle is inline with the camera. The red circle represents the location the projectile would be deployed.

that the AHAB system successfully inserted the payload in 90% of attempts.

The accuracy of the angular velocity estimator was tested by obtaining a large collection of images of the rotating target, then randomly selecting subsets of the collection to use for angular velocity estimation. This test was performed using images over the course of 3 periods of the target rotating at 0.2383 rad/s. 1000 random subsets of 9 images each were studied, and the resulting angular velocity estimates are graphed in Fig. 5.10. The resulting estimates have an average of 0.2381 rad/s and a standard deviation of 0.0006 rad/s.

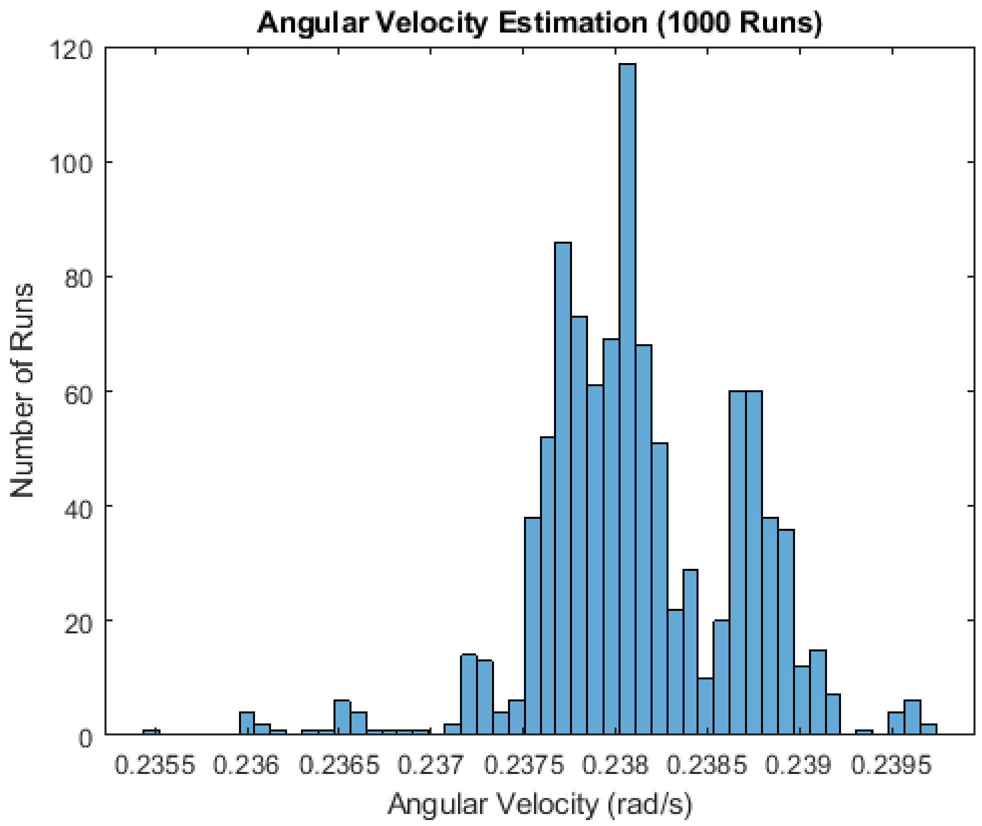


Figure 5.10: Histogram containing angular velocity estimates from 1000 subsets of images taken over a 3-period span of a 0.2383 rad/s rotation. Each subset contains 9 randomly chosen images.

6. SUMMARY

Innovative approaches to integrating monocular cameras into guidance and navigation systems are developed and tested. In two of the projects, computer vision is integrated with inertial sensor information to estimate the relative pose of a vehicle with respect to a target. In the planetary Entry, Descent, and Landing (EDL) project, the target is unknown terrain, and in the aerial refueling project, the target is a spacecraft marked by LED beacons. It is shown that by using an error quaternion, a multiplicative extended Kalman filter formulation can be implemented using a pinhole camera model to update the bias states of the inertial sensors. The measurement sensitivity matrix, and associated extended Kalman filter update associated with a minimal attitude parametrization are derived and shown to be simple extensions of the filter formulation used in attitude estimation applications. Simulation data obtained from a state-of-the-art rendering engine is utilized along with experimental data obtained from the Navigation, Estimation and Sensing Testbed (NEST) to show the efficacy of the EDL formulation derived in the paper. The aerial refuleing algorithm is verified by field testing using two automobiles, as well as by laboratory testing using a suite of robotic platforms from the Land, Air, and Space Robotics (LASR) laboratory. In the spacecraft capture operations project, a fast ellipse detection algorithm is able to detect the rocket nozzle of a rotating spacecraft and provide an accurate estimate of relative position and the target's angular velocity. This algorithm is verified using a laboratory proximity operations setup utilizing the LASR lab robotics, and the real-time capabilities are tested using a statistical analysis.

REFERENCES

[1] L. Libuda, I. Grothues, and K. F. Kraiss, "Ellipse detection in digital image data using geometric features," *Communications in Computer and Information Science*, vol. 4, pp. 229–239, 2007.

[2] R. Szeliski, *Computer Vision: Algorithms and Applications.* Springer, second ed., 2011.

[3] R. I. Hartley and A. Zisserman, *Multiple View Geometry in Computer Vision.* Cambridge University Press, second ed., 2004.

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

[5] H. Bay, A. Ess, T. Tuytelaars, and L. V. Gool, "SURF: Speeded up robust features," *Computer Vision and Image Understanding (CVIU)*, vol. 110, no. 3, pp. 346–359, 2008.

[6] S. Leutenegger, M. Chli, and R. Y. Siegwart, "BRISK: Binary robust invariant scalable keypoints," in *2011 International conference on computer vision*, IEEE, 2011.

[7] N. Trawny, A. I. Mourikis, S. I. Roumeliotis, A. E. Johnson, and J. F. Montgomery, "Vision aided inertial navigation for pin-point landing using observations of mapped landmarks," *Journal of Field Robotics*, 2006.

[8] J. M. Carson, M. M. Munk, R. R. Sostaric, J. N. Estes, F. Amzajerdian, J. B. Blair, D. K. Rutishauser, C. I. Restrepo, A. M. Dwyer-Cianciolo, G. Chen, *et al.*, "The SPLICE project: Continuing NASA development of GN&C technologies for safe and precise landing," in *AIAA Scitech 2019 Forum*, 2019.

[9] J. L. Crassidis and J. L. Junkins, *Optimal Estimation of Dynamic Systems.* Boca Raton, FL: Chapman-Hall/CRC Press, 2004.

[10] S. Soatto, A. J. Yezzi, and H. Jin, "Tales of shape and radiance in multiview stereo," in *Proceedings of IEEE International Conference on Computer Vision*, vol. ICCV-03, IEEE Computer Society, 2003.

[11] E. J. Lefferts, F. L. Markley, and M. D. Shuster, "Kalman filtering for spacecraft attitude estimation," *Journal of Guidance Control and Dynamics*, vol. 5, no. 5, pp. 417–429, 1982.

[12] H. Schaub and J. Junkins, *Analytical Mechanics of Space Systems.* AIAA Education Series, Reston, VA: AIAA, 2009.

[13] A. W. Fitzgibbon, M. Pilu, and R. B. Fisher, "Direct least-squares fitting of ellipses.," in *Proceedings of 13th International Conference on Pattern Recognition*, vol. 1, pp. 253–257, 1996.

[14] D. J. Kessler and B. G. Cour-Palais, "Collision frequency of artificial satellites: The creation of a debris belt," *Journal of Geophysical Research*, vol. 83, pp. 2637–2646, Jun 1978.

[15] N. R. Council *et al.*, *Continuing Kepler's Quest: Assessing Air Force Space Command's Astrodynamics Standards.* National Academies Press, 2012.

[16] B. W. Barbee, S. Alfano, E. Piñon, K. Gold, and D. Gaylor, "Design of spacecraft missions to remove multiple orbital debris objects," in *2011 Aerospace Conference*, pp. 1–14, 2011.

[17] J. Missel and D. Mortari, "Path optimization for Space Sweeper with Sling-Sat: A method of active space debris removal," *Advances in Space Research*, vol. 52, pp. 1339–1348, Jul 2013.

[18] V. Braun, A. Lupken, S. Flegel, J. Gelhaus, M. Mockel, C. Kebschull, C. Wiedemann, and P. Vorsmann, "Active debris removal of multiple priority targets," *Advances in Space Research*, vol. 51, no. 9, pp. 1638–1648, 2013.

[19] E. Kim, M. Haseyama, and H. Kitajima, “Fast line extraction from digital images using line segments,” *Systems and Computers in Japan*, vol. 34, no. 10, pp. 76–89, 2003.

[20] S. Thomas and Y. Chan, “A simple approach for the estimation of circular arc center and its radius,” *Computer Vision, Graphics, and Image Processing*, vol. 45, no. 3, pp. 362–370, 1989.