VISION-AIDED NAVIGATION: IMPROVED MEASUREMENTS MODELS AND A DATA DRIVEN APPROACH

A Dissertation

by

DYLAN TAYLOR CONWAY

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

DOCTOR OF PHILOSOPHY

Chair of Committee,	John L. Junkins
Committee Members,	John E. Hurtado
	Daniele Mortari
	Jinxiang Chai
Head of Department,	Rodney Bowersox

December 2016

Major Subject: Aerospace Engineering

ABSTRACT

Vision-aided navigation is the process of fusing data from visual cameras with other information sources to provide vehicle state estimation. Fusing information from multiple sources in a statistically optimal manner requires accurate stochastic models of each information source. Developing such a model for visual measurements presents a number of challenges.

Vision-aided navigation systems rely on a set of computer vision methods known as feature detection and tracking to abstract visual camera images into a data source amenable to state estimation. It is nearly universally assumed that the measurements produced by these methods have independent and identically distributed (IID) errors. This study presents evidence that directly contradicts these assumptions. Novel models for visual measurements that eliminate the IID assumption are developed. Estimators are designed around the models and tested. Results demonstrate a significant performance advantage over existing methods and also reveal new challenges and paradoxes that motivate further research.

In addition to improving vision-aided navigation models, a set of flexible and robust data-driven estimation techniques are developed and demonstrated on both canonical problems and problems in vision-aided navigation.

TABLE OF CONTENTS

LIST OF FIGURES

LIST OF TABLES

1 INTRODUCTION

Navigation is the process of determining a vehicle's state as a function of time. Such a process is necessary for both controlling a vehicle along a desired trajectory as well as for a variety of other tasks such as the processing of scientific and intelligence data. The vehicle's state may consist of the vehicle's position and orientation, and their derivatives, relative to a coordinate-frame of interest. Other quantities may also be desired depending on the application.

Modern navigation systems consist of both a suite of sensors which measure quantities related to the six degree-of-freedom vehicle motion and computer-based algorithms used to fuse the sensor output into a vehicle state estimate. In designing the computer-based algorithms, a mathematical model to relate each sensor output to the vehicle state is needed. Physical constraints and the dynamics of the system must also be modeled in a similar manner. Such models may depend on *nuisance parameters* which must be estimated in addition to the vehicle state of interest. While it is possible to use deterministic sensor models for some simple navigation tasks, all sensors are inherently stochastic. Because navigation systems typically have a sensor suite providing redundant information, accounting for the stochastic nature of each sensor can lead to more accurate and robust systems.

The process of computing a quantity as a function of stochastic measurements related to the quantity is known as *estimation*. Estimation is rooted in probability

theory and is critical in modern navigation system design. Probability theory provides a complete foundation for designing estimation routines when *exact* stochastic models of each sensor are provided. In practice, there is usually uncertainty in the stochastic model itself. This truth has implicitly degraded and limited the success of vision-aided navigation systems.

The requirement for specifying exact sensor models presents two challenges. First, the system designer must attempt to develop a sufficiently accurate model. Second, the system designer must ensure that discrepancies between the true sensor and its model do not seriously degrade the estimation routine. These two challenges are especially difficult for visual cameras imaging complicated scenes.

Visual cameras have received significant attention in the engineering literature for their potential use in navigation systems. They are low-cost and their measurements provide a rich source of information about both the operating environment and vehicle motion relative to it. While visual cameras have become well-established in certain applications involving well-controlled environments, there remains many obstacles that restrict their domain of applicability and limit the quality of achievable results. These obstacles are largely due to the explicit dependence and sensitivity of visual measurements on the operating environment which complicates the modeling task. As a result, estimators that use visual cameras, compared to other sensors, must be more robust to departures from the assumed model which are inevitable. The primary motivation of this dissertation is to improve vision-aided navigation

systems by developing more accurate models for visual camera measurements and by developing robust estimators that use the developed models.

In attempting to do accomplish this goal, several general problems in estimation are encountered. These problems relate to the design of estimators around incomplete model specifications. This motivates the development of data-driven estimators which are trained for minimum asymptotic variance using experimental data collected on the system of interest.

estimators designed using empirical system data.

1.1 Background

Visual cameras measure the wavelength-weighted intensity of light integrated over an exposure time at each of a very large number ($> 10^5$) of pixels which are arranged in a rectangular grid on an imaging plane. Each pixel corresponds to a narrow ($< 0.05^\circ$) viewing frustum centered on a particular direction. The transformation between the pixel number and the direction is computed via a projection model of the sensor. This model is parameterized by the so-called *intrinsic camera parameters* which are estimated by a calibration process. With this in mind, a calibrated visual camera can be treated as a directional light sensor.

The expected visual camera output in a given scene can be computed with the scene's *plenoptic function* which gives the intensity of light in a particular direction (2D), at a particular position (3D), at a particular wavelength (1D), at a particular

time (1D) [1]. Given the seven-dimensional plenoptic function, a camera model, and a trajectory (position and attitude as a function of time), the expected images at any discrete set of time instances is fully determined. The true images will be distributed in a neighborhood about the expected image since the light-intensity measurements have their own stochastic errors. This is one means by which to model visual measurements for their use in navigation. Unfortunately, this path is terribly complicated. Because the plenoptic function depends on the full geometry and surface properties of the scene as well as all lighting sources, it is inherently infinite dimensional. Even if it is discretized, there would still need to be a very large number of parameters to be jointly estimated along with the state quantities of primary interest. Image processing techniques offer a means to overcome this problem.

A set of image processing techniques known as *feature detection, description, and tracking* methods abstract images from a grid of intensity measurements to a set of pixel location measurements. These methods assume that the scene contains a set of point-landmarks that can be uniquely identified in multiple images. The methods both detect and measure the pixel locations that these point-landmarks project to in image space. This leads to a drastic reduction in the number of auxiliary parameters. In particular, instead of depending on the plenoptic function, the expected measurements only depend on the scene-space positions of the landmarks which must be parameterized. While this provides an enormous advantage, it does produce one

new difficulty as a side-effect. The difficulty is that a probabilistic model is much more difficult to capture. Even if a perfect probabilistic model was available for the raw pixel intensity measurements, the image processing algorithms involve complicated scene-dependent transformations of the raw data. An empirical approach seems the only way forward.

In order to develop an empirical probabilistic model for visual feature measurements, experimental data must be available. Notation must be introduced to describe this.

Let the position of a point landmark j be represented by $\boldsymbol{x}_j$ and the pose, namely position and attitude, of camera i be represented by $\boldsymbol{\theta}_i$. The measurement of landmark j by camera i is represented by $\tilde{\boldsymbol{y}}_{i,j}$. It is nearly universally assumed that $\tilde{\boldsymbol{y}}_{i,j}$ is the sum of a deterministic function of landmark position and camera pose, $\mathrm{h}(\boldsymbol{\theta}_i, \boldsymbol{x}_j)$, and an error term $\boldsymbol{\nu}_{i,j}$ The errors are assumed to be independent and identically distributed (IID):

$$\mathbb{E}\left\{\boldsymbol{\nu}_{i,j}\boldsymbol{\nu}_{k,l}^T\right\} = 0 \text{ if } i \neq k \text{ or } j \neq l \tag{1.1}$$

Lastly, the landmark position is assumed to be a constant (i.e. not changing with the image index i). For classic references using these assumptions, see Triggs [2], Hartley and Zisserman [3], or Davison [4].

The structural part of the model, $\mathrm{h}(\boldsymbol{\theta}_i, \boldsymbol{x}_j)$, is developed as follows. First, under the assumption of constant point-landmark positions, the line-of-sight (LOS) direction from the sensor to the point-landmark is a simple function of camera pose and

landmark position. Second, physics-based models of the optics imply that this direction corresponds to a particular location on the image plane. Third, it is assumed that the feature tracker can determine this image plane location. The probabilistic portion of the model accounts for the fact that the image plane location determined by the feature tracker is imperfect. If the first two steps can be accepted, on account of the fact that they are derived from physics-based principles, an empirical study can determine a model for the probabilistic component.

In particular, if a batch of measurements $\tilde{\boldsymbol{y}}_{i,j}$ for many landmarks and cameras is available along with ground truth for the landmark positions and camera poses, then samples of the error distribution can be computed directly: $\tilde{\boldsymbol{y}}_{i,j} - \mathbf{h}(\boldsymbol{\theta}_i, \boldsymbol{x}_j)$. The main difficulty with doing this in practice is getting accurate ground truth data. Special vision targets with well-surveyed corner-like landmarks (think checkerboards) are very useful for landmark position ground truth. Also, metrology systems and optical calibration equipment can be used to obtain pose ground truth. However, it appears that such a study has never been done, possibly because errors in the ground truth data may dominate the computed error samples in practice. In addition, ground truth landmark positions in general scenes are not uniquely determined.

Instead of a direct approach to studying feature localization errors, a hastier approach has been taken in the literature. Least-squares estimators are commonly used for visual measurements. In particular, these estimators find the parameters

that minimize the cost

$$\mathscr{J} = \sum_i \sum_j ||\tilde{\boldsymbol{y}}_{i,j} - \mathrm{h}(\boldsymbol{\theta}_i, \boldsymbol{x}_j)||^2 \tag{1.2}$$

This type of estimator is the Maximum Likelihood Estimator (MLE) under the assumption of IID isotropic Gaussian errors. Furthermore, and more heuristically, if the measurement equation was linear then this estimator would be the best (minimum variance) linear unbiased estimator (BLUE) by the Gauss-Markov theorem [5]. Statistical tests can then be applied to the residuals to check for consistency with the assumption of Gaussian errors.

One common theme found in the literature is that a straight-forward application of least-squares estimation leads to poorer than expected results. The culprit is often so-called *outlier* measurements: ones with much larger than expected errors. These may be due to poorly localized features or erroneously labeled correspondences. The solution often taken is to dichotomize between *good* measurements, those that are nearly Gaussian with a scale on the order of one or two pixels (or less), and *bad* measurements, those with errors larger than a few pixels.

Despite the lack of rigorous evidence to support the assumption of Gaussian IID errors and constant landmark positions, much of the literature on vision-aided navigation relies on it. At a high-level, the problem of focus in this dissertation is:

> Given feature tracking measurements extracted from visual images of a general scene and any other well-modeled sensor, jointly estimate the camera pose and landmark positions.

Historically, the field of photogrammetry has relied on *bundle adjustment* to solve this problem [2].

Bundle adjustment is an umbrella term and encompasses a number of particular methods. At its core, bundle adjustment computes an estimate as the solution on a nonlinear optimization problem. The nonlinear optimization adjusts the *bundle of rays* associated with line-of-sight directions (i.e. feature measurements) from many overlapping images to many landmarks to minimize an MLE cost function.

The MLE is an estimation framework that will be discussed in detail in Section 2. For the current discussion, it suffices to say that the MLE has a number of statistically optimal properties when the input observations have perfect probabilistic models. In addition, it is important to stress that the MLE is not a single algorithm: it is a framework. It requires the specification of likelihood equations for each information source: sensors, dynamics, constraints, etc. The likelihood equations for various information sources are then combined to form the optimization problem. Critically, the likelihood equation for one information source does not depend on what other information sources are in the estimator.

The correctness of each likelihood equation in an MLE (i.e. how well it captures the true nature of the information source) is important as it determines, in loose terms, how much confidence is placed in each information source. Therefore, the development of improved likelihood equations for an information source can be expected to improve the overall estimator performance. As stated previously, the in-

herent difficulty in modeling feature tracking methods raises doubts about the quality of the commonly used likelihood equations for such measurements. As a result, a key aim of this dissertation is to improve such models.

The use of the MLE, and the commonly associated visual models, is not restricted to bundle adjustment in photogrammetry. Over the past 15 years, there has been a large volume of literature on the so-called *Simultaneous Localization and Mapping* (SLAM) problem. The SLAM problem is the *same* problem as the bundle adjustment problem stated above. However, the term is typically used in applications involving mobile robots that need near real-time state estimates.

One of the earliest stochastic approaches to SLAM is based on the work of Smith and Cheeseman where they proposed a *stochastic map* [6]. The map contained two-dimensional landmark positions which were augmented to a three-dimensional vehicle state (two-dimensional position and bearing). The joint state was propagated and updated using an Extended Kalman Filter (EKF).

The first practical six degree-of-freedom SLAM implementation using only a monocular camera is the *MonoSLAM* system presented by Davison *et al* [4, 7]. That system implemented an EKF with a state vector consisting of camera position, attitude, their derivatives and a $3N$-dimensional sub-block containing the three-dimensional positions of N landmarks. The system assumed a zero-mean IID Gaussian model for both the process noise and measurement noise. The estimate of state and covariance was used to define search regions for each feature in the current

image. No explicit mention of outlier features is mentioned in their paper. However, as features not found cannot be used in the estimator, and due to the fact that any feature found outside the search region could amount to an outlier, outliers were implicitly rejected.

Much of the ensuing research after Davison on vision-aided SLAM involved improvements to the filter: (1) better landmark parameterizations, (2) lower computational complexities, and (3) improved outlier rejection. Civera and Davison proposed an inverse-depth parameterization for landmarks which essentially reduced the nonlinearity in the measurement equations leading to a more accurate state covariance in the EKF [8]. Montemerlo *et al.* demonstrated a way to exactly factor the problem so that, conditioned on the state estimates, each landmark can be estimated independently of all others [9]. This was implemented as a particle filter, known as *FastSLAM* and had much lower computational complexity than the EKF. An improvement in the *proposal distribution* used by the particle filter was later published under the name *FastSLAM 2.0* [10]. Both implementations assumed Gaussian IID errors.

Outlier rejection has typically relied on the strict labeling of measurements as either *inliers* or *outliers* using Random Sample and Consensus (RANSAC) methods [11]. Computer vision applications using RANSAC include *Kalmansac* and the work of Nister [12, 13].

All of the above methods, although not necessarily derived under the MLE,

make heavy use of the standard assumptions. One more recent development that improves on the above-mentioned SLAM solutions is the incorporation of batch estimation techniques as opposed to purely sequential filtering. Excellent summaries are provided by Strasdat *et al* [14, 15]. The main conclusion is that a full MLE bundle adjustment that grows in size as new data is collected is the optimal solution. In addition, various approaches are proposed to bound the size of the problem. This includes the use of a sliding window of data, creation of locally optimal sub-maps that are mathematically connected through a smaller number of six degree-of-freedom transformations, and the selection of a small subset of cameras (often called *key-frames*) to be used in the optimization.

The early work on this front was the *PTAM* system of Klein and Murray [16]. PTAM performed bundle adjustment using a small subset of cameras. The optimization was processed in the background while a front-end tracking system computed pose estimates and recorded new feature measurements for the next round of bundle adjustment. A different approach, also using bundle adjustment, was taken by Konolige and Agrawal in their *FrameSLAM* system. They used bundle adjustment to optimize the pose and landmarks of closely spaced cameras to create small sub-maps. A higher-level optimization procedure was then used to solve for the relative pose of sub-maps using features that spanned multiple sub-maps. Various other modifications to SLAM using MLE bundle adjustments methods exist.

Again, at the core of all these methods is the same common assumptions about

visual feature measurements. As Strasdat summarizes [14]:

> BA [bundle adjustment] in SFM [structure from motion], or the extended Kalman filter (EKF) and variants in SLAM all manipulate the same types of matrices representing Gaussian means and covariances. The clear reason is the special status of the Gaussian as the central distribution of probability theory which makes it the most efficient way to represent uncertainty in a wide range of practical inference. We therefore restrict our analysis to this domain.

The IID nature of the errors is also implicit in the MLE solutions developed. A primary motivation of this dissertation is to challenge these assumptions for the purpose of developing improved estimators. The nearly universal use of these assumptions make such a study especially important with potentially broad impacts.

In addition to developing improved models of visual measurements for the purposes of MLE, this dissertation addresses two broader problems. The first is the design of M-estimators, a close cousin of the MLE, given only samples from a probability distribution as opposed to the exact equation. The method for M-estimator design developed in this dissertation is referred to as *data-driven estimation* and has a direct application to vision-aided navigation.

The second problem is the incorporation of *auxiliary measurements* into M-estimators. Auxiliary measurements are defined, in this dissertation, as observations that are paired with the primary observation of interest and provide information

about the quality of the primary observation. In the context of vision-aided navigation, feature detection and matching scores may serve as auxiliary measurements. The potential use of such information extracted from empirical training data and applied to new problems is a second aspect of data-driven estimation developed in this dissertation.

A summary of the approach to address the discussed problems is given below.

1.2 Approach

1. Perform an empirical study to determine the validity of the constant-landmark and IID error assumptions for visual feature measurements in general scenes.
2. Develop improved structural models for visual feature measurements in general scenes that relax the constant-landmark and IID error assumptions.
3. Develop techniques to automate the design of estimators given samples from an unknown error distribution.
4. Develop techniques to automate the incorporation of auxiliary measurements into estimators.
5. Design estimators around improved visual feature measurement models that are inherently robust to small departures from the model and that give improved performance over existing estimator design.
6. Leverage auxiliary measurements and the data-driven estimation techniques developed in this dissertation to improve vision-aided navigation performance.

To be clear, this dissertation does *not* propose any particular SLAM or bundle adjustment method. The literature on such methods is extensive. Instead, the focus is on developing models for *visual measurements* that can be incorporated into *nearly any* SLAM or bundle adjustment method. As these methods rely heavily on the MLE, the results of this dissertation can be utilized whenever visual feature measurements are used regardless of which other sensors are used in the problem of interest. Furthermore, the contributions on data-driven estimation have potentially broad impacts outside the scope of vision-aided navigation; it is anticipated that these are important first steps to continued research.

1.3 Overview

This dissertation is organized as follows. Section 2 discusses MLE and its robust extension known as M-estimation. M-estimators systematically reduce estimator sensitivity to modeling errors which are inevitable in estimation problems: no model is perfect.

Section 3 presents some of the commonly used sensor models for vision-aided navigation. In particular, commonly used probabilistic models for visual cameras, range cameras, and inertial measurement units (IMU) are given. Section 2 and Section 3 are necessary for results developed in later sections.

Section 4 discusses two experimental methods used in this dissertation to generate measurements with accurate ground truth for sensor trajectories. The first

method is a high-fidelity visual and range image rendering tool and the second method is a hardware system consisting of real sensors. Details of the set up, calibration, and validation of these two systems are given. The measurements, simulated or real, from the experimental methods are used in a flexible batch estimator which is discussed at the end of Section 4. The estimator assumes the measurement model can be partitioned into a deterministic component that is a function of unknown parameters and a probabilistic component that is independent of the unknown parameters. The two components are assumed to act additively to generate measurements. The estimator architecture enables the user to independently specify models for the structural component and probabilistic component. This flexibility is critical for developing the contributions of this dissertation.

Improving the structural and probabilistic models used in vision-aided navigation is a primary contribution of this dissertation. Addressing both model components simultaneously is difficult as their designs are inherently coupled. Consider a case where a probabilistic model is **perfectly known** and a structural model is given which has systemic errors. The systemic errors will *look like* probabilistic errors that are larger than expected from the known probabilistic model and with different correlation structure. This can severely degrade estimator performance. On the other hand, consider the opposite case where the structural model is **perfectly known** and the probabilistic model is unknown. Again, this will degrade estimator performance as the errors are improperly modeled and hence a true Maximum Likelihood

Estimate cannot be obtained.

Systemic errors in the structural model are inevitable. Therefore, a logical path forward in designing better models for estimators is to attempt various hypothesized structural models and then attempt to determine what the required probabilistic model is that can account for both the systemic errors and the truly random errors. The design of these models requires a combination of physical insight and empirical evidence from realistic measurements with accurate ground truth. It is critical to accept the fact that in the majority of applications, no perfect structural and probabilistic model can be found with finite datasets. Furthermore, in applying the developed models to perform estimation tasks on new datasets or with slightly different systems, one cannot be certain that the model is equally applicable to the new data. This is where the robust M-estimators play a crucial role: they are inherently robust to modeling error.

As stated in the previous paragraph, hypothesizing a family of structural models is a first step in improving estimation results. Section 5 presents a set of standard assumptions used in the structural component of vision-aided navigation measurement models. The section then uses the experimental methods to build evidence to judge the extent to which these assumptions are inappropriate. This motivates the alternative models, also presented in Section 5, which attempt to remove the standard assumptions. A systematic study is performed on the structural models and the necessary probabilistic models to make them consistent with the data are

developed. This development is performed on simulated data which is much easier to obtain in abundance.

The selected probabilistic model must specify both a parametric form and the values of the parameters themselves (*e.g.* normal distribution with a particular variance). It is possible to leave some parameters free and adapt them for the given dataset. This is a very distinct concept from using test data to fully specify a model. Allowing parameters of the probabilistic model to be jointly estimated as nuisance parameters adds flexibility to the estimator and extends the applicability of the model. Methods to jointly estimate these nuisance parameters are given in detail. Results on simulated data are given in Section 5 for the various combinations of structural models and probabilistic models with and without probabilistic model nuisance parameters. The various tradeoffs between complexity, accuracy, and robustness are assessed.

Section 6 presents the motivation, development and application of data-driven estimation methods. In particular, methods are presented to design inherently robust estimators using many samples from an unknown error distribution. A subset of these methods incorporates auxiliary measurements to improve the estimator performance. Examples to evaluate performance gains are given on both generic problems and for vision-aided navigation.

Concluding remarks are given in Section 7.

2 MAXIMUM LIKELIHOOD AND ROBUST M-ESTIMATION

Maximum Likelihood Estimation (MLE) is a widely used method for parameter estimation in probabilistic systems. The core idea is to define the optimal parameter estimate as the parameter value that maximizes the probability of the observed measurements conditioned on the unknown parameters. While other metrics may be more desirable in certain applications, the MLE lends itself to very practical and computationally efficient solutions. In addition, desired properties such as minimum variance, consistency, and statistical efficiency are reached by the MLE asymptotically under fairly lax restrictions.

MLEs are designed under the assumption that the input data are distributed **exactly** according to a parametric model. On the other end of the spectrum are so-called *non-parametric* estimation techniques that attempt to fit the data with a model of appropriate complexity with minimal assumptions about the true underlying system. In many applications, the reality lies between these two extremes: the data is *approximately* distributed according to an available parametric model. Unfortunately the MLE and other classical parametric techniques are extremely sensitive to departures of the true system from the assumed model. While nonparametric techniques may offer a path around this difficulty, they do not readily incorporate approximate knowledge of the true system and therefore are not as efficient (in a

statistical sense) as needed. Robust estimation techniques offer an excellent middle ground. Although various approaches exist for robustifying classical estimators like the MLE, they all have one notion in common: designing for improved insensitivity to modeling errors.

This section first gives a brief overview of the MLE and its properties. A simple example is then given to motivate the need for more robust approaches. A robust procedure that is directly applicable to the vision-aided navigation problem (and any multivariate estimation problem) are *M-estimators*. The M-estimators are discussed both from a theoretical and a practical point of view. The properties and issues discussed will be utilized in novel approaches to vision-aided navigation presented in later sections.

2.1 MLE

This section gives a brief overview of the MLE. The MLE is first put into context among estimators in general. Then assumptions behind commonly used properties of the MLE are discussed and issues specific to vision-aided navigation are highlighted.

The MLE is just one method used to derive estimators: many others exist. Estimators are most commonly a deterministic function of a set of measurements $\boldsymbol{Y}$. The set of measurements $\boldsymbol{Y}$ is assumed to come from a probability distribution $\mathrm{p}_{\boldsymbol{\theta}}(\boldsymbol{Y})$ parameterized by $\boldsymbol{\theta}$ which is an unknown parameter vector. The estimator computes an estimate of $g(\boldsymbol{\theta})$ as a function of the measurements: $\boldsymbol{T}(\boldsymbol{Y})$ where $g()$ is an arbitrary

function (possibly identity). For a given $\boldsymbol{\theta}$, the distribution $\mathrm{p}_{\boldsymbol{\theta}}(\boldsymbol{Y})$ and the function $\boldsymbol{T}$ will determine the distribution of estimates, and hence a distribution of estimator errors $\boldsymbol{e} = \boldsymbol{T}(\boldsymbol{Y}) - g(\boldsymbol{\theta})$. In most applications, the desired estimator properties can be expressed in terms of the distribution of $\boldsymbol{e}$. This includes properties like unbiasedness, minimum variance, and potentially more expressive properties like minimizing the probability that the error exceeds some bound. While shaping the distribution of $\boldsymbol{e}$ via the choice of estimator $\boldsymbol{T}()$ at various values of $\boldsymbol{\theta}$ is conceptually elegant, it is extremely difficult to do in practice on general problems.

One case that admits an easy solution is the case of linear Gaussian measurements. In particular, $\boldsymbol{\theta}$ is to be estimated and $\mathrm{p}_{\boldsymbol{\theta}}(\boldsymbol{Y})$ is a multivariate Gaussian with mean $H\boldsymbol{\theta}$ and covariance Q where H is a known full rank matrix and Q is a known positive-definite matrix. The probability distribution of $\boldsymbol{\theta}$ conditioned on the measurements is Gaussian with mean $(H^T Q^{-1} H)^{-1} H^T Q^{-1} \boldsymbol{Y}$ and covariance $(H^T Q^{-1} H)^{-1}$. Selection of the mean as the estimate is then a uniformly unbiased minimum variance estimator. In this particular example, but not in general, the MLE leads to the same exact result.

While designing an estimator to obtain particular characteristics on the error distribution is desirable, it is not practical in most settings. The MLE on the other hand is straight-forward to apply in a broad class of problems. Furthermore, the MLE asymptotically obtains many desirable properties which are discussed in the next section. One last point should be made that is pertinent to both the MLE

and any other estimator requiring the measurement model $\mathrm{p}_{\boldsymbol{\theta}}(\boldsymbol{Y})$. The measurement model is almost always based on various assumptions and empirical data and is therefore imprecise. Many estimators do not account for this in their design and may therefore be sensitive to such discrepancies. The M-estimators discussed in this section explicitly consider measurement model impreciseness in their design.

2.1.1 Properties

In order to derive an MLE, a parametric probabilistic measurement model is needed. Let $\mathrm{p}(\boldsymbol{y} \,|\, \boldsymbol{\theta})$ be the conditional probability of a measurement $\boldsymbol{y}$ given unknown parameters $\boldsymbol{\theta}$, both being scalar or vector quantities. The MLE estimate of $\boldsymbol{\theta}$ is

$$\hat{\boldsymbol{\theta}} \equiv \operatorname{argmax}_{\boldsymbol{\theta}} \mathrm{p}(\boldsymbol{y} \,|\, \boldsymbol{\theta}) \tag{2.1}$$

The MLE is especially convenient when the vector of all measurements $\boldsymbol{y}$ can be partitioned into a set of N independent measurements $\{\boldsymbol{y}_1, \boldsymbol{y}_2, \ldots, \boldsymbol{y}_N\}$. By the definition of independence,

$$\mathrm{p}(\boldsymbol{y} \,|\, \boldsymbol{\theta}) = \prod_{i=1}^{N} \mathrm{p}(\boldsymbol{y}_i \,|\, \boldsymbol{\theta}) \tag{2.2}$$

using a slight abuse of notation (i.e. the particular form of $\mathrm{p}(\cdot \,|\, \cdot)$ is determined by the first argument to allow for non-identically distributed measurements). The definition of the MLE in Equation (2.1) is unaltered if the conditional probability function is transformed by any function that is monotonically increasing on the interval $[0, 1]$. By applying the logarithm function to the right-hand side of Equation (2.2), the

product is transformed to a summation of independent terms:

$$\log\left(\mathrm{p}(\boldsymbol{y}\,|\,\boldsymbol{\theta})\right) = \sum_{i=1}^{N} \log\left(\mathrm{p}(\boldsymbol{y}_i\,|\,\boldsymbol{\theta})\right) \tag{2.3}$$

In addition, note that minimizing the negative-log-likelihood is equivalent to maximizing the likelihood. Therefore the negative-log-likelihood will be referred to as the MLE cost. This form is more convenient for both proving various properties of the MLE and for deriving practical optimization algorithms.

Crassidis and Junkins summarize well-known MLE properties [17]:

1. Consistent: As N tends to infinity, the estimate will converge (in probability or almost surely depending on the assumptions) to the true value.
2. Asymptotic Normality: The distribution of the estimate approaches a normal distribution with mean equal to the truth and a variance that scales with N^{-1}.
3. Asymptotic Efficiency: As N tends to infinity, the MLE becomes the best consistent estimator in a mean-squared error sense.
4. Functional Invariance: Applying any transformation $\boldsymbol{\phi} = \mathrm{g}(\boldsymbol{\theta})$ does not alter the MLE in the sense that $\mathrm{g}\left(\mathbf{argmax}_{\boldsymbol{\theta}}\, \mathrm{p}(\boldsymbol{y}\,|\,\boldsymbol{\theta})\right) = \mathbf{argmax}_{\boldsymbol{\phi}}\, \mathrm{p}(\boldsymbol{y}\,|\,\boldsymbol{\phi})$ (with an abuse of notation on the second argument of the likelihood function).

The functional invariance property is important for navigation problems and vision-aided navigation (VAN) in particular because various parameterizations exist for the unknown position, attitude, and scene structure. The property implies that the optimal solution in one parameterization is equivalent to the optimal solution under a different parameterization. However, certain parameterizations will lead to

numerical algorithms that converge more quickly on the solution. As a general rule of thumb, the more linear the measurement is in the parameters, the better the convergence will be [18].

Another important issue to keep in mind is that the asymptotic properties of the MLE require certain subtle assumptions that **do not** apply in general navigation problems. The underlying cause is that the number of unknown parameters increases with the number of measurements. This issue was first addressed in a 1948 econometrics paper by Neyman and Scott in which they coined they term *the incidental parameters problem* [19]. This problem will briefly be discussed below as it is important for navigation applications.

2.1.2 The Incidental Parameters Problem

The incidental parameters problem demonstrates the breakdown of the MLE properties when the number of parameters scales with the number of the measurements. To show this, the consistency proof for the MLE will be outlined. To prove consistency, first define the score of the i-th measurement as the gradient of the log-likelihood

$$\boldsymbol{s}_i(\boldsymbol{\theta}) = \frac{\partial}{\partial \boldsymbol{\theta}} \log\left(\mathrm{p}(\boldsymbol{y}_i \,|\, \boldsymbol{\theta})\right) \tag{2.4}$$

which is a vector when the measurement $\boldsymbol{y}_i$ is scalar and $\boldsymbol{\theta}$ is a vector [17, 20]. This vector can be shown to have an expected value of zero:

$$\mathbb{E}\{\boldsymbol{s}_i(\boldsymbol{\theta})\} = \int \boldsymbol{s}_i(\boldsymbol{\theta}) \mathrm{p}(\boldsymbol{y}_i \,|\, \boldsymbol{\theta}) d\boldsymbol{y}_i \tag{2.5}$$

$$= \int \frac{\mathrm{p}'(\boldsymbol{y}_i \,|\, \boldsymbol{\theta})}{\mathrm{p}(\boldsymbol{y}_i \,|\, \boldsymbol{\theta})} \mathrm{p}(\boldsymbol{y}_i \,|\, \boldsymbol{\theta}) d\boldsymbol{y}_i \tag{2.6}$$

$$= \frac{\partial}{\partial \boldsymbol{\theta}} \int \mathrm{p}(\boldsymbol{y}_i \,|\, \boldsymbol{\theta}) d\boldsymbol{y}_i \tag{2.7}$$

$$= \frac{\partial}{\partial \boldsymbol{\theta}} 1 \tag{2.8}$$

$$= \mathbf{0} \tag{2.9}$$

The variance of the score is the Fisher Information matrix which is positive semi-definite and finite. Because each $\boldsymbol{s}_i$ has zero-mean and bounded variance, the law of large numbers (LLN) applies to its sample average:

$$\frac{1}{N} \sum_{i=1}^{N} \boldsymbol{s}_i(\boldsymbol{\theta}) \to \mathbf{0} \tag{2.10}$$

Clearly

$$\frac{1}{N} \sum_{i=1}^{N} \boldsymbol{s}_i(\hat{\boldsymbol{\theta}}) = \mathbf{0} \tag{2.11}$$

is a necessary condition for the estimate. A Taylor series expansion of this condition about the true parameter,

$$\frac{1}{N} \sum_{i=1}^{N} \boldsymbol{s}_i(\hat{\boldsymbol{\theta}}) = \frac{1}{N} \sum_{i=1}^{N} \boldsymbol{s}_i(\boldsymbol{\theta}) + \frac{1}{N} \sum_{i=1}^{N} \boldsymbol{s}'_i(\boldsymbol{\theta})(\hat{\boldsymbol{\theta}} - \boldsymbol{\theta}) \tag{2.12}$$

along with Equation (2.10) shows that the estimator is consistent if the matrix $\frac{1}{N}\sum_{i=1}^{N} \boldsymbol{s}'_i$ is nonsingular. The matrix is singular in the case that any element of $\boldsymbol{\theta}$ appears in only a finite number of measurements even as N tends to infinity. In that case the corresponding row and column (and diagonal element) of the matrix

will tend to zero which will cause the singularity.

Neyman and Scott refer to this issue as *the incidental parameters problem.* They distinguish between *structural parameters*, those appearing in an infinite number of measurements, and *incidental parameters*, those appearing in only a finite number of measurements, as N tends to infinity. They also refer to an infinite set of measurements as *partially consistent* if the set has a finite number of structural parameters and an infinite number of incidental parameters. Their primary conclusions are that for a partially consistent set of measurements:

1. The incidental parameters are not consistent which is unsurprising. More importantly, the structural parameters are not necessarily consistent.
2. Even if the structural parameters are consistent, the structural parameter estimates are not necessarily asymptotically efficient.

Although not well known in the aerospace literature, the result of Neyman and Scott is very important. For the VAN problem with a finite number of images, MLE estimates of the motion parameters are not necessarily consistent or asymptotically efficient as the number of landmarks goes to infinity. Similarly, for the case of a finite number of landmarks, the landmark parameters are not necessarily consistent or asymptotically efficient as the number of images goes to infinity. These facts are important to keep in mind when deriving estimates and their corresponding uncertainties [1]!

[1]In the linear Gaussian case, the estimates still follow a Gaussian distribution so unbiased minimum variance estimators can still be derived. Note that the results of Neyman and Scott

It should also be briefly noted that this concept is distinct from *observability*. Observability is concerned with the uniqueness of the mapping from the system input-output history to the current system state. This definition does not make recourse to probability theory which is central to the incidental parameters problem. Observability is required, but does not guarantee, a consistent estimate.

2.1.3 An Example

As a brief example of the incidental parameters problems, consider measurements of the form

$$y_{i,j} = x_i + z_j + \nu_{i,j} \; \forall i \in \{2,2,\ldots,M,(M+1)\}, \; \forall j \in \{1,2,\ldots,N\} \tag{2.13}$$

$$y_{1,j} = z_j + \nu_{i,j} \; j \in \{1,2,\ldots,N\} \tag{2.14}$$

where x_i are M unknowns and z_j are N unknowns to be estimated and the $\nu_{i,j}$ are IID Gaussian errors with variance σ^2. This is a linear Gaussian system with measurement matrix

$$H = \begin{bmatrix} 0_{N\times M} & I_{N\times N} \\ I_{M\times M} \otimes 1_{N\times 1} & 1_{M\times 1} \otimes I_{N\times N} \end{bmatrix} \tag{2.15}$$

when the parameter vector is stacked with the x_i on top and the z_j on bottom. Also, $1_{a\times b}$ is an $a \times b$ matrix with all elements equal to one, $0_{a\times b}$ is an $a \times b$ matrix with all elements equal to zero, and $I_{a\times a}$ is an $a \times a$ identity matrix. In addition, $A \otimes B$ represents the Kronecker product between matrices A and B.

As stated above for linear Gaussian systems, the parameters conditioned on the measurements are also Gaussian with known mean and covariance: the inciden-

tal parameters problem does not change this! Therefore the standard least squares estimator is efficient in the minimum variance sense (note the *not necessarily* keywords in the above results). However, we must check whether or not the estimator is consistent. To do so, obtain the parameter covariance expression P:

$$\begin{aligned}
P &= \sigma^2 (H^T H)^{-1} \\
&= \sigma^2 \left(\begin{bmatrix} 0_{M\times N} & I_{M\times M} \otimes 1_{1\times N} \\ I_{N\times N} & 1_{1\times M} \otimes I_{N\times N} \end{bmatrix} \begin{bmatrix} 0_{N\times M} & I_{N\times N} \\ I_{M\times M} \otimes 1_{N\times 1} & 1_{M\times 1} \otimes I_{N\times N} \end{bmatrix} \right)^{-1} \\
&= \sigma^2 \left(\begin{bmatrix} I_{M\times M} \otimes 1_{1\times N} 1_{N\times 1} & 1_{M\times 1} \otimes 1_{1\times N} \\ 1_{1\times M} \otimes 1_{N\times 1} & I_{N\times N} + 1_{1\times M} 1_{M\times 1} \otimes I_{N\times N} \end{bmatrix} \right)^{-1} \\
&= \sigma^2 \left(\begin{bmatrix} N I_{M\times M} & 1_{M\times N} \\ 1_{N\times M} & (M+1) I_{N\times N} \end{bmatrix} \right)^{-1}
\end{aligned} \tag{2.16}$$

where the Kronecker properties $(A \otimes B)^T = A^T \otimes B^T$ and $(A \otimes B)(C \otimes D) = (AB) \otimes (CD)$ are used. Using the block matrix inverse identity from Appendix B of Crassidis and Junkins, the marginal variance of the stacked x_i parameters and the marginal variance of the stacked z_j parameters can be obtained (i.e. the top-left and bottom

right diagonal blocks of P: P_{xx} and P_{cc} respectively) [17]:

$$P_{xx} = \left(N I_{M\times M} - \frac{1}{M+1} 1_{M\times N} 1_{N\times M} \right)^{-1} \tag{2.17}$$

$$= \left(N I_{M\times M} - \frac{N}{M+1} 1_{M\times M} \right)^{-1} \tag{2.18}$$

$$P_{zz} = \left((M+1) I_{N\times N} - \frac{1}{N} 1_{N\times M} 1_{M\times N} \right)^{-1} \tag{2.19}$$

$$= \left((M+1) I_{N\times N} - \frac{M}{N} 1_{N\times N} \right)^{-1} \tag{2.20}$$

$$\tag{2.21}$$

Note that the matrices that need to be inverted are both the sum of an invertible matrix and a rank-one matrix. Therefore the result of Miller can be used [21]:

$$(G+H)^{-1} = G^{-1} - \frac{1}{\mathrm{tr}(HG^{-1})} G^{-1} H G^{-1} \tag{2.22}$$

where G is invertible and H is rank-one. Using this result yields

$$P_{xx} = \frac{1}{N} (I_{M\times M} + 1_{M\times M}) \tag{2.23}$$

$$P_{zz} = (M+1) I_{N\times N} - \frac{M}{N} 1_{N\times N} \tag{2.24}$$

Note that the diagonal elements which are the variance of x_i and z_j are $\frac{2}{N}$ and $M+1-\frac{M}{N}$ respectively (up to the scalar σ^2). Clearly as $N \to \infty$, the variance of x_i goes to zero implying that the estimate is consistent. On the other hand, the estimate of z_j is *not* consistent: neither N, nor M, nor any combination of the two going to infinity can make the variance go to zero.

This example problem was designed to be analogous to the vision-aided navigation problem where x_i represents the camera pose and z_j represents the landmark position, and each landmark is measured once from a known pose. It was shown

that the estimates of the x_i (camera poses) were consistent as N (the number of landmarks) goes to infinity but the estimates of z_j were not consistent regardless of N or M (number of images). The reason for this makes sense intuitively. From the first image alone which has known pose in Equation (2.14), each landmark position can be estimated up to some error, namely $v_{1,j}$. Then for any other image, the measurement equation is equivalent to

$$y_{i,j} = x_i + v_{i,j} + v_{1,j} \tag{2.25}$$

Using the N measurements $\{y_{i,1}, y_{i,2}, \ldots y_{i,N}\}$ to estimate x_i is equivalent to the location estimation problem (to be discussed in detail below). The standard sample mean gives a consistent estimate. Unfortunately the results for this simple problem do not exactly carry over to the vision-aided navigation problem as multiple images from different poses are required to estimate landmark position to within a finite uncertainty.

In addition to the incidental parameters problem, there is a second issue undermining the power of the MLE. In particular, inconsistencies between the model and true system can lead to a reduction (sometimes drastic) in performance. This is the subject of the next section.

2.2 A Motivating Example: Location Estimation

The location estimation problem is a canonical example of an estimation problem for which the MLE is suitable. The problem will be discussed below and some

perhaps surprising inadequacies of the MLE in practical applications will be shown.

The location estimation problem is to obtain an estimate for the scalar parameter θ given measurements

$$y_i = \theta + \nu_i \tag{2.26}$$

where ν_i are corrupting errors that are IID. If the errors are assumed to be zero-mean Gaussian variables with variance σ^2, then the MLE cost function becomes the widely used least squares cost function:

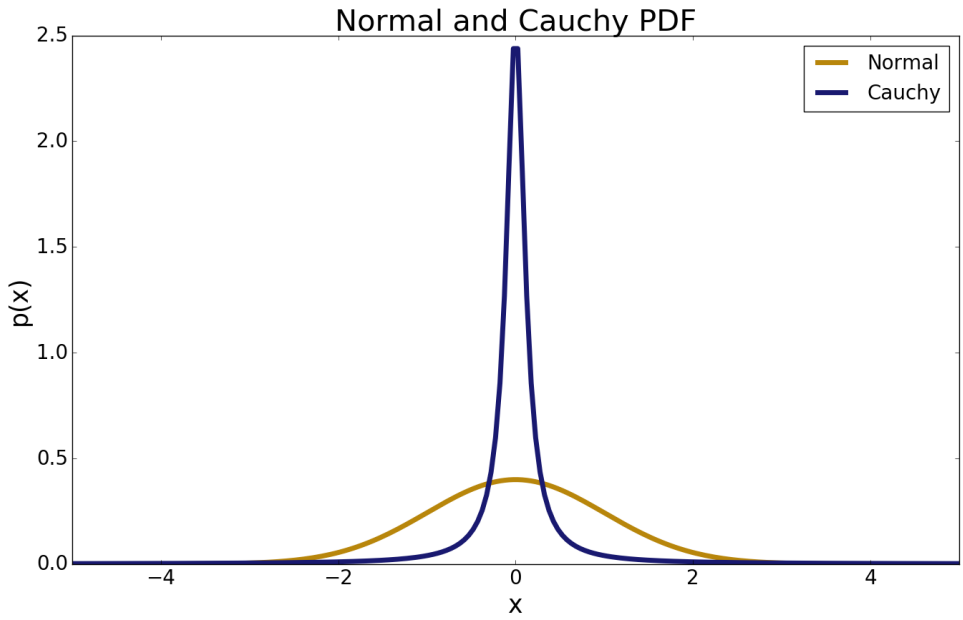
$$\hat{\theta} = \text{argmin}_\theta \sum_{i=1}^{N} (y_i - \theta)^2 \tag{2.27}$$

The resulting estimate is the sample average. This estimate is unbiased and has minimum variance, σ^2/N, among all unbiased estimators. While this may seem like a done deal, what if we consider the case where the Gaussian assumption is not precisely correct?

To study this with a simple numerical example, we let $N = 40$ and perform 10,000 trials of the location estimation problem for two cases. The first case has standard normal errors and the second case has Cauchy errors. The scale parameter of the Cauchy distribution is set such that the resulting PDF of the Cauchy and normal are equal at $x = \pm 3$. A plot of the distributions is shown in Figure (2.1). In each trial, the MLE under the assumption of Gaussian errors (i.e. the sample average) is obtained. The resulting errors are shown in the histogram in Figure (2.2).

Clearly the least squares estimator has very poor performance under Cauchy

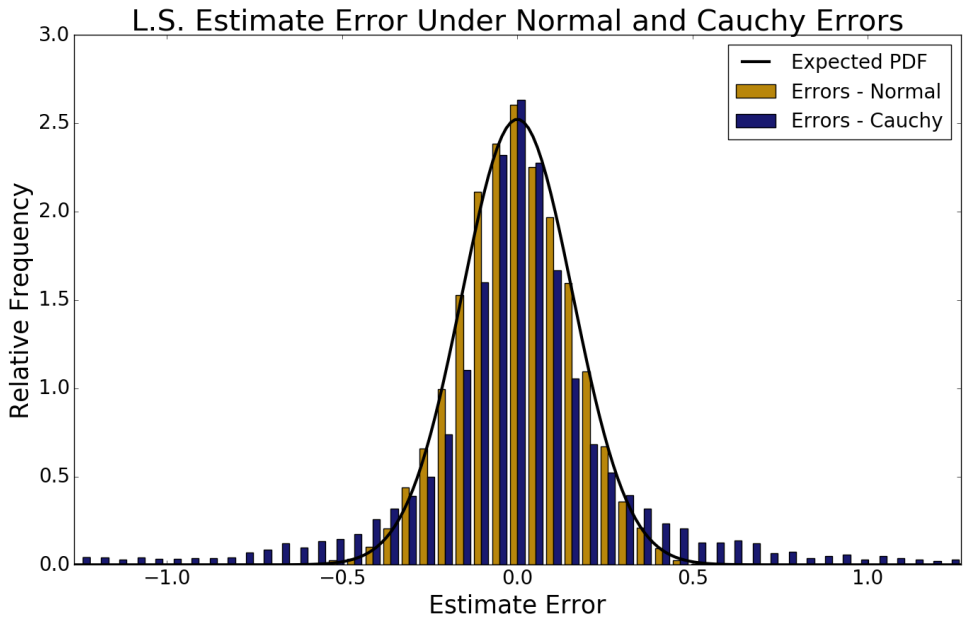
Figure 2.1: The Cauchy and normal PDF.



error. The reader may wonder how this is a criticism of the MLE. The reason is that in practice the true PDF is seldom known and such an extreme sensitivity of the estimator to the underlying error is dangerous. Even if experiments are performed to obtain samples from it, it may be difficult to distinguish one PDF from another with a finite number of samples. As Figure (2.1) suggests, the Cauchy distribution appears to have a tighter scale when looking at the middle 99.7 % of the distribution. However, the extreme tails are in fact much wider for the Cauchy which is ultimately what corrupts the resulting estimate. Those regions of the distribution are the *least* understood in experimentation (since there will be very few samples from there).

To explore what would happen if we instead used the Cauchy distribution to derive an MLE, the example above is repeated. The Cauchy distribution with scale

Figure 2.2: Location estimate errors using least squares under the normal and Cauchy distribution. Some samples of the estimate error under Cauchy measurement errors fall outside the plot limits.



γ is

$$\mathrm{p}(y\,|\,\theta) = \left[\pi\gamma\left(1+\left(\frac{y-\theta}{\gamma}\right)^2\right)\right]^{-1} \tag{2.28}$$

The corresponding cost function for the location is obtained by taking the negative-log of Equation (2.28) which yeilds

$$\hat{\theta} = \mathrm{argmin}_\theta \; -\log\left[\pi\gamma\left(1+\left(\frac{y-\theta}{\gamma}\right)^2\right)\right]^{-1} \tag{2.29}$$

$$= \mathrm{argmin}_\theta \log(\pi\gamma) + \log\left(1+\left(\frac{y-\theta}{\gamma}\right)^2\right) \tag{2.30}$$

$$= \mathrm{argmin}_\theta \log\left(1+\left(\frac{y-\theta}{\gamma}\right)^2\right) \tag{2.31}$$

The solution is computed using the SciPy implementation of Brent's method [22, 23].

RMS Error	Least Squares	Cauchy MLE
Gaussian	0.16	0.18
Cauchy	14.2	0.045

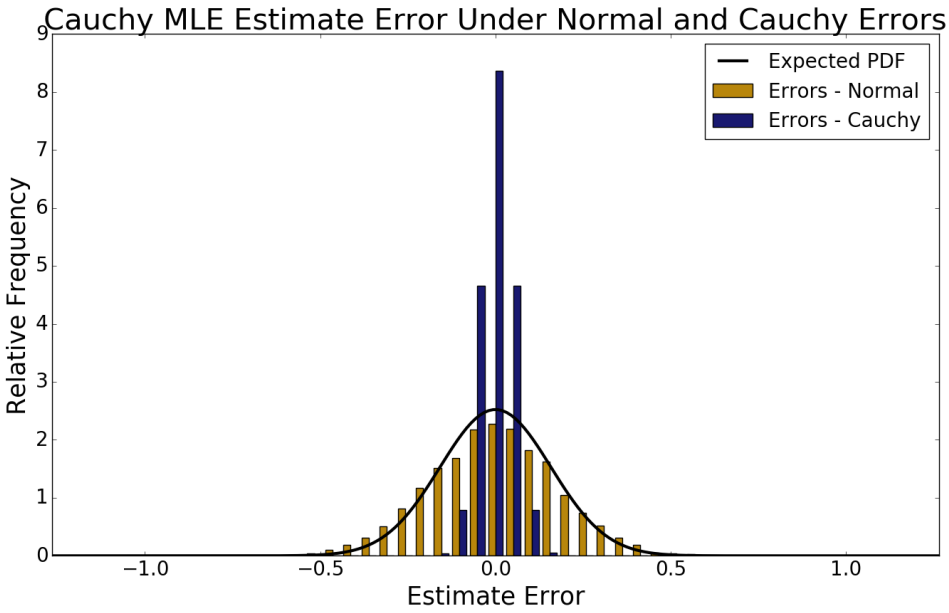
Table 2.1: RMS error of Gaussian and Cauchy MLE under Gaussian and Cauchy errors.

The results are shown in Figure (2.3). The orange histogram bars represent the estimate errors when the $N = 40$ measurements are corrupted with Cauchy errors (as in the previous plot as well). Clearly changing the cost function gives a vast improvement for the estimates on Cauchy corrupted measurements. This is displayed clearly in Table 2.1. The RMS error is reduced by a factor of 315! Interestingly, the estimates that use measurements corrupted with normal errors have only a slight degradation, a 12% increase in RMS error, when the Cauchy MLE cost is used. Comparing these estimates to their expected distribution, the solid back line, makes this obvious to see.

The degradation is inevitable because the least squares estimates are both the MLE and minimum variance unbiased estimates. The fact that the degradation is so slight is surprising. If we were unsure of whether the errors were Cauchy or normal, then using the Cauchy MLE seems more desirable: **the slight reduction in efficiency at the normal is rewarded with major improvements at the Cauchy.**

The field of *robust statistics* formalizes this qualitative tradeoff. The so-called *M-estimators* are an extremely powerful tool in robust statistics that are a key part

Figure 2.3: Location estimate errors using the Cauchy MLE under the normal and Cauchy distribution. The line labeled *expected PDF* is for the estimate error of the Gaussian MLE under Gaussian errors.



of this dissertation. They are the subject of the next section.

2.3 M-Estimators

M-estimators are defined as any estimator whose output is determined by minimizing a cost function. The MLE is a subset of these estimators whose cost function is the likelihood function (or a monotonically increasing function of the likelihood). Early work by Tukey demonstrated the deficiencies of classical parametric techniques (especially those designed around the normal distribution) and discussed the concept of contaminated distributions as a more realistic model [24]. Huber was the first to set up a rigorous theory of robustness through a minimax approach: find the

best estimator for the worst-case distribution in a neighbourhood [25]. This initial theory was followed up by a much different approach by Hampel [26, 27]. Hampel's approach is based on the influence function which, loosely speaking, gives the rate of change in an estimator with respect to the amount of contamination at a given location. Although this is an infinitesimal approach that does not need the explicit use of neighborhoods of distributions about a model, it gives the same result as Huber's minimax approach in certain canonical problems.

This section will define the M-estimator, present various measures of quantitative robustness, discuss properties, and tailor the generic M-estimator to problems of interest. Practical methods of solving for the M-estimate will be given.

Let the M-estimate $\hat{\boldsymbol{\theta}}$ be defined as

$$\mathscr{J}(\boldsymbol{\theta}) = \sum_{i=1}^{N} \rho(\boldsymbol{y}_i; \boldsymbol{\theta}) \tag{2.32}$$

$$\hat{\boldsymbol{\theta}} = \text{argmin}_{\theta}\, \mathscr{J}(\boldsymbol{\theta}) \tag{2.33}$$

A necessary condition for an optimal estimate is

$$\mathbf{0} = \frac{\partial}{\partial \boldsymbol{\theta}} \sum_{i=1}^{N} \rho(\boldsymbol{y}_i; \boldsymbol{\theta}) \tag{2.34}$$

$$= \sum_{i=1}^{N} \psi(\boldsymbol{y}_i; \hat{\boldsymbol{\theta}}) \tag{2.35}$$

where $\psi()$ is the derivative of $\rho()$ with respect to $\boldsymbol{\theta}$. The main philosophy behind M-estimators is to choose the cost function to simultaneously give good performance if the measurements happen to come from the assumed model and acceptable performance if the measurements happen to diverge from the model by some amount

(to be formalized below).

In an ideal scenario, all uncertainties in a problem can be exactly specified and the distribution of parameters conditioned on the measurements can be determined. This hypothetical distribution of parameters could then be used to choose the optimal estimate (by any optimality criterion). This is the essence of Bayesian methods. M-estimators are removed from this in two ways. First, they admit the truth; the distribution of all random quantities is not known precisely. So-called *robust Bayesian* methods do this as well by creating a *super model* of uncertainties about the assumed probabilistic model [28]. Second, robust M-estimators do not attempt to compute a distribution of parameters conditioned on the measurements. While this may limit the types of optimality criteria that can be designed around, it leads to much more computationally efficient estimators as compared to Bayesian solutions.

Compared to Bayesian (and robust Bayesian) methods, M-estimators approach model uncertainty in a very different way. Bayesian estimators attempt to parameterize the discrepancy between a nominal statistical model and a true system which leads to a range of possible outcomes depending on what the true system really is. On the other hand, robust M-estimators specify a neighborhood of models that true system must fall into. The estimate is then designed around the worst-case model in that neighborhood. Therefore, as long as the true system exists in that neighborhood, the performance will be equal to or better than the performance obtained at

the design model.

The following subsections discuss various properties of M-estimators that are relevant to later sections.

2.3.1 Influence Function

The influence function is due to Hampel and is a good starting point for a discussion of robustness [26]. A key concept in studying the influence function and many other robustness properties is to treat the estimate as a functional $T(F)$ of a distribution F. Note that the distribution F can be the empirical distribution for N samples x_i:

$$F_n(x) \equiv \frac{1}{N}\sum_{i=1}^{N} \mathbf{1}_{x_i}(x) \tag{2.36}$$

where $\mathbf{1}_{x_i}(x)$ is the unit-step function at x_i.

First consider a nominal distribution F (i.e. a cumulative distribution function (CDF) over some domain Ω) that is contaminated with a second distribution G by amount t where $t \in [0,1]$: $(1-t)F + tG$. The estimator T is Gateaux differentiable at F if there exists a function a such that for all G (see Hampel page 83 and Huber Section 2.5 [27, 29])

$$\lim_{t\to 0^+} \frac{1}{t}\left[T((1-t)F + tG) - T(F)\right] = \int_{\Omega} a(u)dG(u) \tag{2.37}$$

Now let $G(u) = \delta_x(u)$ (i.e. the unit impulse at x). Then

$$\mathrm{IF}_{F,T}(x) \equiv \lim_{t\to 0^+} \frac{1}{t}\left[T((1-t)F + t\delta_x) - T(F)\right] \tag{2.38}$$

$$\equiv \lim_{t\to 0^+} \frac{1}{t}\left[T((1-t)F + t\delta_x) - T(F)\right] \tag{2.39}$$

$$= \int_\Omega a(u)\delta_x(u) \tag{2.40}$$

$$= a(x) \tag{2.41}$$

is the *influence function* of the estimator T at F. Gateaux differentiability is required for the influence function to be defined. Note that we assume the measurements are scalar quantities and the parameter to be estimated is a vector. Therefore $\mathrm{IF}_{F,T}(x)$ is generally vector quantity (a scalar only in the special case of a scalar parameter).

The so-called influence function is a quantitative assessment of an estimators' robustness. Loosely speaking, it is the derivative of an estimate under distribution F with respect to the amount of contaminating mass at a particular location x. It is useful in assessing the asymptotic properties of an estimator subject to infinitesimal perturbations from the nominal model. While the *infinitesimal* aspect of the influence function may be suspect, the reader may find it useful to compare this to Lyapunov stability analysis.

Lyapunov stability analysis uses a scalar positive-definite function of system state to summarize the system output. For continuous-time systems a Lyapunov derivative must be negative definite to obtain asymptotic stability in a neighborhood about an equilibrium point. The estimator functional T is analogous to a Lyapunov function, the contamination amount is analogous to time, and the distribution is

analogous to system state. At a nominal model, we have a condition analogous to an equilibrium:

$$\int \mathrm{IF}_{F,T}(x)dF(x) = \mathbf{0} \tag{2.42}$$

which follows from Equation (2.37) and Equation (2.38). We then consider how small perturbations in distribution (state) by contamination amount t (time) change the estimate (Lyapunov function). Similarly to how a Lyapunov function must be negative for all possible perturbations in state (i.e. negative-definite), the influence function must be finite for all possible perturbations in distribution δ_x.

The useful properties of the influence function can be determined by a first order expansion of the estimate $T(G)$ about $T(F)$ for G in a small neighborhood about F. First, by the Glivenko-Cantelli theorem, the empirical distribution $F_{\boldsymbol{\theta}_n}$ tends to the true distribution $F_{\boldsymbol{\theta}}$ in probability and almost surely. Therefore $T(F_{\boldsymbol{\theta}_n})$ tends to $T(F_{\boldsymbol{\theta}})$. If the estimator is Fisher consistent, then

$$T(F_{\boldsymbol{\theta}}) = \boldsymbol{\theta} \; \forall \boldsymbol{\theta} \tag{2.43}$$

which, with the above, implies $T(F_{\boldsymbol{\theta}_n})$ tends to $\boldsymbol{\theta}$. In words, an estimator that is consistent at the parametric model, regardless of the parameter value, will also be consistent at the empirical distribution of samples obtained from the parametric model.

Now apply an expansion of $T(G)$ about $T(F)$

$$T(G) \approx T(F) + \int \mathrm{IF}_{T,F}(x)(dG(x) - dF(x)) + R \tag{2.44}$$

$$= T(F) + \int \mathrm{IF}_{T,F}(x)dG(x) + R \tag{2.45}$$

where R is a remainder term. Boos and Serfling give conditions under which R is on the order of the contamination amount t [30]. In particular, if the estimator is consistent and the influence function is bounded then R is on the order of t. While other conditions are possible, this condition is important in practice because bounding the influence function is a key step in robustifying an estimator. In those cases,

$$T(G) \approx T(F) + \int \mathrm{IF}_{T,F}(x) dG(x) \tag{2.46}$$

and the central limit theorem (CLT) implies

$$T(F_N) - T(F) \approx \frac{1}{N} \sum_{i=1}^{N} \mathrm{IF}_{T,F}(x_i) \tag{2.47}$$

Therefore the estimator asymptotic variance is

$$V(T,F) = \int \mathrm{IF}_{T,F}(x) \mathrm{IF}_{T,F}(x)^T dF(x) \tag{2.48}$$

Note the explicit dependence on both the estimator and distribution: the two do not have to be related as they are in expressions for the MLE.

There is one last expression to derive in this section. In particular, given the vector-valued function ψ and its associated estimator functional T, we would like to have a convenient way to determine the influence function. To do so, begin with the necessary condition for the estimator

$$\int \psi(x, T(G)) dG(x) = \mathbf{0} \tag{2.49}$$

Now let $G = (1-t)F + t\delta_c$ and substitute into the necessary condition to obtain

$$\int \psi(x, T((1-t)dF + t\delta_c)) ((1-t)dF + t\delta_c) = \mathbf{0} \tag{2.50}$$

Now take the derivative with respect to t to obtain

$$\mathbf{0} = \int \frac{\partial}{\partial T}\left(\psi(x, T(G))\right) \frac{\partial T((1-t)dF + t\delta_c)}{\partial t} dG + \int \psi(x, T(G))\left(\delta_c - dF\right) \tag{2.51}$$

Now evaluating at $t = 0$, using the definition of the influence function, and the identity in Equation (2.42) yields

$$\mathbf{0} = \int \frac{\partial}{\partial \boldsymbol{\theta}}\left(\psi(x, \boldsymbol{\theta})\right) \mathrm{IF}(c, T, F) dF + \psi(c, T(F)) \tag{2.52}$$

Solving for the influence function and making a change of variables (for notational consistency) yields

$$\mathrm{IF}(x, T, F) = \left(\int \frac{\partial}{\partial \boldsymbol{\theta}}\left(\psi(u, \boldsymbol{\theta})\right) dF(u)\right)^{-1} \psi(x, T(F)) \tag{2.53}$$

This expression is simply a specialization of the influence function to M-estimators which will be useful below.

The key conclusion of this subsection is that the influence curve quantifies how well the estimate holds up for infinitesimal perturbations, in the sense of contamination amount rather than contamination location, about the assumed model. This is in contrast to the *breakdown point* which quantifies how much contamination the estimator can handle without failing completely.

2.3.2 Gross-Error Sensitivity

The gross-error sensitivity is derived from the influence function presented in the previous section [26]. There are several variants. The unstandardized version is

defined as [27]:

$$\gamma^{\star} = \sup_{x} ||\mathrm{IF}_{T,F}(x)|| \tag{2.54}$$

To see the importance of this quantity, let F be the true distribution and G be F with ε contamination of δ_x in Equation (2.46). The estimator asymptotic bias is

$$\boldsymbol{b}(x) = \varepsilon \mathrm{IF}_{T,F}(x) \tag{2.55}$$

Therefore the gross-error sensitivity is defined as the norm of the worst-case asymptotic bias among all distributions G in the ε-contamination neighborhood of F [29].

A slightly more useful form of gross-error sensitivity is given by the information-standardized form [27]:

$$\gamma_I^{\star} \equiv \sup_{x} \left(\mathrm{IF}_{T,F}(x)^T J(T(F)) \mathrm{IF}_{T,F}(x) \right)^{1/2} \tag{2.56}$$

where $J(T(F))$ is the Fisher information matrix

$$J(T(F)) \equiv \int \boldsymbol{s}(x,\boldsymbol{\theta}) \boldsymbol{s}(x,\boldsymbol{\theta})^T dF(x) \tag{2.57}$$

and the score defined in Equation (2.4) This definition is invariant to a transformation of the parameters $\boldsymbol{\theta}$.

2.3.3 Breakdown Point

Consider the ε-contaminated distribution obtained about the assumed model F_o, defined as the one parameter family of models

$$F(x) = (1-\varepsilon)F_o(x) + \varepsilon G \tag{2.58}$$

The breakdown point is the largest ε such that the estimator error remains bounded for all G.

For example, the sample mean as an estimate of the mean of a distribution is the MLE when the nominal distribution is normal. This is asymptotically

$$\hat{\mu} = \int x dF(x) \tag{2.59}$$

Substituting the contaminated distribution

$$\hat{\mu} = \int x((1-\varepsilon)dF_o(x) + \varepsilon dG(x)) \tag{2.60}$$

which yields

$$\hat{\mu} = (1-\varepsilon)\mathbb{E}\{F_o(x)\} + \varepsilon\mathbb{E}\{G(x)\} \tag{2.61}$$

Since G is arbitrary (i.e. can have arbitrarily high mean), any $\varepsilon > 0$ can lead to unbounded estimate error for distributions in the ε-contaminated neighborhood of F.

On the opposite end of the spectrum, consider the median as an estimate of the mean of a distribution. It is defined as the solution to

$$0 = \int \text{sign}(x - \hat{\mu}) dF(x) \tag{2.62}$$

It will be shown below that this estimate under the above contamination with $G = \delta_{x^\star}$ at infinity is

$$\hat{\mu} = F_o^{-1}\left(\frac{1}{2(1-\varepsilon)}\right) \tag{2.63}$$

Since the inverse CDF $F_o^{-1}(x)$ becomes undefined for $x > 1$, the breakdown point for the median is $\varepsilon = 0.5$. In words, the median estimate error is bounded when up to 50% of the contamination is arbitrarily large. The penalty paid for this robustness is a reduction in estimator efficiency at the normal distribution.

2.3.4 Minimax Bias and Variance: Location Problem

The equations of the preceding section give the asymptotic bias and variance of an M-estimator of location for measurements corrupted by any distribution. An interesting problem is to determine the cost function that minimizes bias or variance under the worst case distribution. In most applications, we have a nominal model for the distribution that has some uncertainty associated with it. Therefore instead of considering the worst case distribution among all possible distributions, we consider only those distributions in some neighborhood of the nominal model. While many neighborhoods definitions are possible, the so called ε-contaminated neighborhoods are especially convenient. Given a nominal model, F, the ε-contaminated neighborhood is defined as the set

$$\mathscr{P}_\varepsilon(F) = \{F_\varepsilon | F_\varepsilon = (1-\varepsilon)F + \varepsilon G\} \tag{2.64}$$

where G is any distribution. The minimax bias and variance estimators will be determined below.

Assume that the nominal model is the standard normal distribution and consider the cost function

$$\rho(x) = \begin{cases} \frac{1}{2}x^2 & |x| \leq k \\ k|x| - \frac{1}{2}k^2 & |x| > k \end{cases} \tag{2.65}$$

with corresponding ψ

$$\psi(x) = \begin{cases} x & |x| \leq k \\ k\text{sign}(x) & |x| > k \end{cases} \tag{2.66}$$

and ψ'

$$\psi'(x) = \begin{cases} 1 & |x| < k \\ 0 & |x| > k \end{cases} \tag{2.67}$$

Recall that the negative-log-likelihood gives the minimum asymptotic variance for any distribution. So if we design an MLE around a distribution $Ce^{-\rho(x)}$, then that estimator is minimum variance for that distribution. It remains to investigate if such a distribution also maximizes the variance. To see this, use the expression for asymptotic variance derived above and specialize it to the given distribution and cost function (substitute Equation (2.53) into Equation (2.48)) to get the variance expression:

$$V(T,F) = \frac{\mathbb{E}\left\{\psi^2\right\}_F}{\mathbb{E}\left\{(\psi')^2\right\}_F} \tag{2.68}$$

which for the above cost function and contaminated distribution yields

$$\frac{\mathbb{E}\left\{\psi^2\right\}}{\mathbb{E}\left\{(\psi')^2\right\}} = \frac{(1-\varepsilon)\mathbb{E}\left\{\psi^2\right\}_F + \varepsilon \int_{|x|>k} k^2 g(x)dx}{(1-\varepsilon)^2 \mathbb{E}\left\{\psi'\right\}_F^2} \tag{2.69}$$

where g is the PDF associated with the CDF G. Clearly, the variance is maximized by any distribution that places all of its mass outside of the interval $[-k,k]$ (maximizes the numerator with no change to the denominator). This is precisely what the distribution constructed by $Ce^{-\rho(x)}$ does. Therefore this is the minimax variance

solution.

The minimax bias solution for any symmetric nominal model is the median which can be demonstrated in two steps. The first step is to determine the maximum bias distribution in a contamination neighborhood about any symmetric distribution for the median location estimate. The second step is to show that no estimate can have a smaller bias than the median at all distributions in the neighborhood.

In particular, the median estimate under distribution F (with PDF $f = (1-\varepsilon)f_o(x) + \varepsilon g(x)$) is obtained from

$$0 = \int \text{sign}(x - \hat{\theta})((1-\varepsilon)f_o(x) + \varepsilon g(x))dx \tag{2.70}$$

where f_o is the nominal model symmetric about zero. Note that the bias is then equal to the estimate $\hat{\theta}$ (since the true value is zero). Performing the integration yields

$$0 = F_o(\hat{\theta})(1-2\varepsilon) + \varepsilon q \tag{2.71}$$

where q is defined as

$$q = \int \text{sign}(x - \hat{\theta})g(x)dx \tag{2.72}$$

Rearranging gives a bias of

$$b \equiv \hat{\theta} = F_o^{-1}\left(\frac{1 + \varepsilon(q-1)}{2(1-\varepsilon)}\right) \tag{2.73}$$

Note that q is at least -1 if all the mass of g sits to the left of the resulting $\hat{\theta}$ and at most 1 if all mass sits to the right. Either way the magnitude of the bias is maximized by placing all mass entirely to the right or to the left (which makes sense

intuitively).

Next, it must be shown that no estimate can achieve a smaller bias. Consider any estimator that is translation invariant (i.e. adding a constant to all data will shift the estimate by that same constant). The asymptotic estimate can be expressed as a functional $T(F)$ of the underlying distribution F. By definition $T(F(x+c)) = T(F(x))+c$. Therefore if we consider two contaminated distributions, one with all mass on the left and one with all mass on the right, say $F_+(x) = F_o(x+b)$ and $F_-(x) = F_o(x-b)$, then the difference in the estimates at the two distributions is $T(F_+) - T(F_-) = 2b$. Clearly both distributions belong to the neighborhood about F_o and no estimator can achieve a bias magnitude less than b at both distributions. Since the median does achieve this bias, it is the minimax bias solution. To be clear, no distribution in the neighborhood can cause the median to give a worse estimate and no estimator can do better than the median at all possible distributions.

At this point we have shown that the Huber cost is the minimax asymptotic variance solution and the median is the minimax asymptotic bias solution. It is worthwhile to study the variance of the median and the bias of Huber for several values of ε and N (assuming N is large enough for asymptotics to apply).

The variance of the median can be computed with Equation (2.68), using the fact that $\psi'(x) = 2\delta(x)$ where $\delta(x)$ is the unit-impulse at zero. The result is

$$P_\theta = \frac{1}{N} \frac{\int \text{sign}^2(x-\theta) f(x) dx}{(\int 2\delta(x-\theta) f(x) dx)^2} \tag{2.74}$$

$$= \frac{1}{N} \frac{1}{4((1-\varepsilon) f_o(\theta) + \varepsilon g(\theta))^2} \tag{2.75}$$

Clearly this is maximized when $g(\theta) = 0$ which occurs when all mass of g sits to one side of the estimate. The result is

$$P_\theta = \frac{1}{N} \frac{1}{4(1-\varepsilon)^2 f_o^2(\theta)} \tag{2.76}$$

Figure (2.4) plots the maximum asymptotic bias and variance of the median as a function of the number of samples N for two cases of a standard normal nominal model and a standard Cauchy nominal model (i.e. scale parameter equal to one). In both cases the contamination is 1% of the total distribution (i.e. $\varepsilon = 0.01$). Note that the maximum asymptotic bias is independent of N and the maximum variance approaches the plotted curve asymptotically. The key takeaway from this plot is that the variance will dominate the square of bias for reasonable sample sizes: up to 8000 for both distributions. This suggests that the median is **not** a good estimate for minimizing total error squared (i.e. variance plus bias squared).

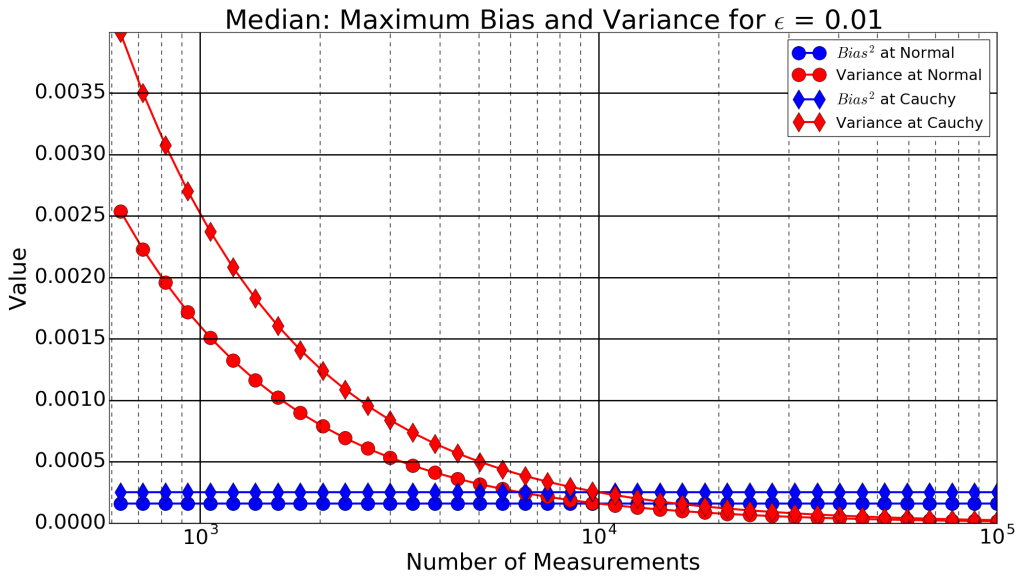
To show a similar result for the Huber cost function, we must derive the asymptotic bias of that estimate under the worst case contamination. An expression for bias can be obtained by subtracting $T(F)$ from both sides of Equation (2.46) and substituting the expression for the influence function in Equation (2.53) which yields

$$b = \frac{\mathbb{E}\{\psi\}}{\mathbb{E}\{\psi'\}} \tag{2.77}$$

where the expectation is evaluated at the true distribution. For the Huber cost under PDF f, this is

$$b = \frac{\int_{-\infty}^{-k} -kf(x)dx + \int_{-k}^{k} xf(x)dx + \int_{k}^{\infty} kf(x)dx}{\int_{-k}^{k} f(x)dx} \tag{2.78}$$

Figure 2.4: Maximum asymptotic bias and variance for the median location estimate under standard normal and standard Cauchy errors. Note the large number of samples needed for the bias to dominate the variance

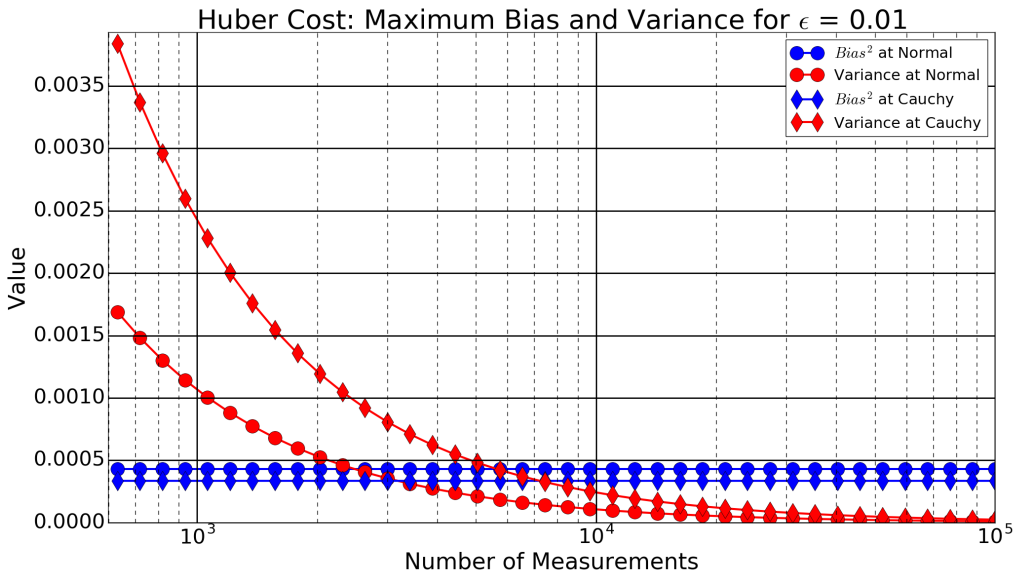


The bias is maximized when the contaminating mass falls outside of the interval $[-k,k]$ since this would both maximize the numerator and minimize the denominator in the above expression. When this is done and the fact the nominal model is symmetric is used, the above equation reduces to

$$b = \frac{k\varepsilon}{(F_o(k) - F_o(-k))(1-\varepsilon)} \tag{2.79}$$

Figure (2.5) is analogous to Figure (2.4) discussed above with the exception that the cost being studied is the Huber cost. The key takeaway from this figure is that variance tends to dominate the bias for reasonable sample sizes: 2000 for the normal and up to 7000 for the Cauchy. Comparing the two figures, we can see that the bias is in fact smaller for the median than for the Huber for both distributions.

Figure 2.5: Maximum asymptotic bias and variance for the Huber location estimate under standard normal and standard Cauchy errors. Note the large number of samples needed for the bias to dominate the variance.

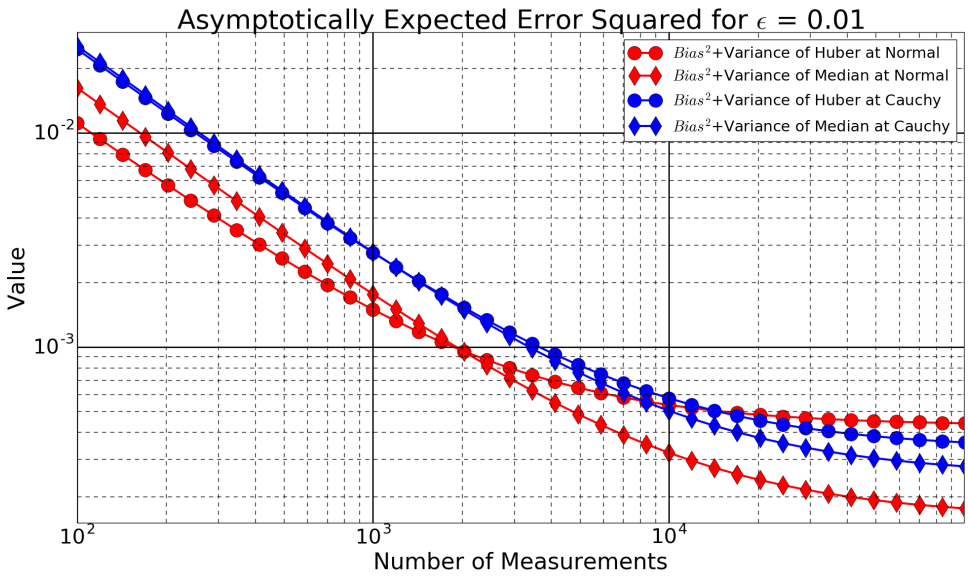


This is consistent with the theory. This suggests that the Huber cost is preferred when total squared error must be minimized and sample sizes are fairly small. Figure (2.6) addresses this in further detail

Figure (2.6) plots the maximum asymptotic sum of bias squared and variance for the Huber and median estimates at the standard and normal. This figure makes it clear that the median is only preferred for very large sample sizes. This is especially true for the normal distribution. The discrepancy is much smaller for the Cauchy.

One last topic that is instructive to study is the sample size N at which bias squared is equal to variance as a function of the contamination amount ε. Figure (2.7) plots this for both the Huber and median estimates at both the Cauchy and

Figure 2.6: Sum of maximum asymptotic bias squared and variance for the Huber and median location estimates under standard normal and standard Cauchy errors. The Huber cost is preferable for moderate sample sizes.

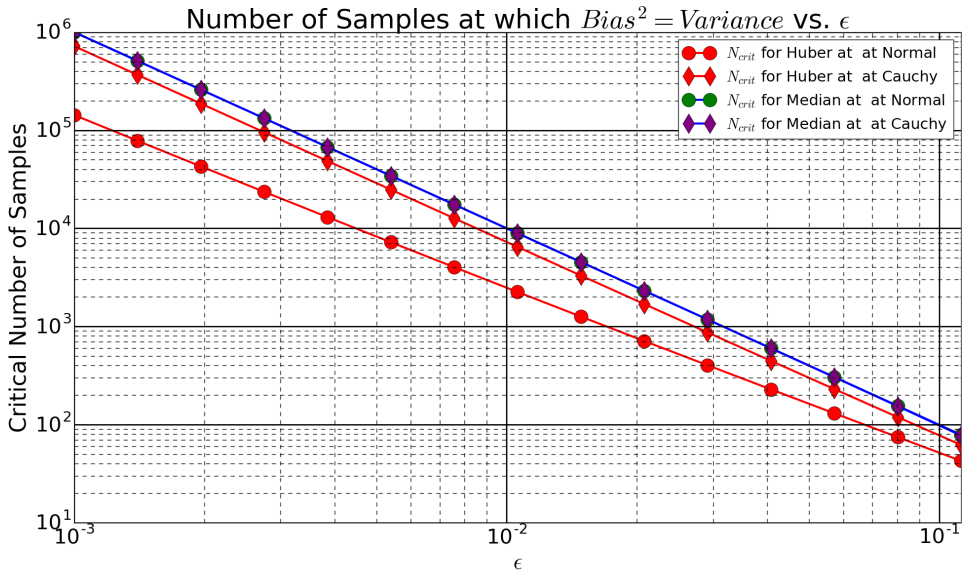


normal distribution. The key takeaway from this plot is that the sample size needed to make the median a better estimate than the Huber decreases exponentially with increasing contamination. This implies that both the sample size and contamination amount must be considered when selecting a suitable cost function.

2.3.5 Nonlinear Regression

This section discusses M-estimators for nonlinear regression problems: given measurements that are a nonlinear function of the parameters, obtain an estimate of the unknown parameters. The estimator definition and necessary conditions are given. Bias and covariance expressions are derived

Figure 2.7: Sum of maximum asymptotic bias squared and variance for the Huber and median location estimates under standard normal and standard Cauchy errors. The Huber cost is preferable for moderate sample sizes.



Consider scalar measurements of the form

$$\boldsymbol{y}_i = \mathrm{h}_i(\boldsymbol{\theta}) + \boldsymbol{\nu}_i \tag{2.80}$$

where $\mathrm{h}_i()$ is a measurement function and $\boldsymbol{\nu}_i$ is an IID unobservable random variable from a distribution F. The vector of parameters $\boldsymbol{\theta}$ must be estimated. This measurement form is especially common in aerospace applications. The results of this section will be related to similar expressions in the literature (specialized for the given measurement form) and will be used in later sections.

The M-estimate has the form $\rho(\boldsymbol{y}_i - \mathrm{h}_i(\boldsymbol{\theta}))$ and necessary condition

$$\begin{aligned}\mathbf{0}^T &= \frac{\partial \mathscr{J}(\hat{\boldsymbol{\theta}})}{\partial \boldsymbol{\theta}} \\ &= -\sum_{i=1}^{N} \psi(\boldsymbol{y}_i - \mathrm{h}_i(\hat{\boldsymbol{\theta}})) \frac{\partial \mathrm{h}_i(\hat{\boldsymbol{\theta}})}{\partial \boldsymbol{\theta}} \\ &= -\sum_{i=1}^{N} \psi_i \mathrm{h}_i' \end{aligned} \tag{2.81}$$

where ψ is a scalar function that is the derivative of ρ and h_i' is the derivative of h_i. The last line in the equation uses short-hand notation to make the following expressions more readable.

The above problem reduces to the scalar location estimation problem when $\mathrm{h}_i(\hat{\boldsymbol{\theta}}) = \theta$ which is the focus of Huber [25, 29]. The analysis below generalizes the results of Huber and others to any measurement function. After the analysis, it will be shown that the results of Huber are recovered by setting $\mathrm{h}_i(\hat{\boldsymbol{\theta}}) = \theta$ in the final expressions. Similarly, the derived expressions will be shown to be consistent with that of Hampel's which are in terms of the influence function [27].

First consider a first-order Taylor series expansion of the necessary condition in Equation (2.81) about the parameter $\boldsymbol{\theta}_o$.

$$\begin{aligned}\mathbf{0}^T &= \frac{\partial \mathscr{J}(\hat{\boldsymbol{\theta}})}{\partial \boldsymbol{\theta}} \\ &= \frac{\partial \mathscr{J}(\boldsymbol{\theta}_o)}{\partial \boldsymbol{\theta}} + \frac{\partial^2 \mathscr{J}(\boldsymbol{\theta}_o)}{\partial \boldsymbol{\theta}^2} \left(\hat{\boldsymbol{\theta}} - \boldsymbol{\theta}_o\right) \end{aligned} \tag{2.82}$$

$$\frac{\partial \mathscr{J}(\boldsymbol{\theta}_o)}{\partial \boldsymbol{\theta}} = -\sum_{i=1}^{N} \psi_i \mathrm{h}_i' \tag{2.83}$$

$$\frac{\partial^2 \mathscr{J}(\boldsymbol{\theta}_o)}{\partial \boldsymbol{\theta}^2} = -\sum_{i=1}^{N} \left[\psi_i' \mathrm{h}_i'^T \mathrm{h}_i' + \psi_i \mathrm{h}_i''\right] \tag{2.84}$$

where ψ' is the derivative of ψ.

Note that the summation in Equation (2.83) contains N terms. Let $\lambda \equiv \mathbb{E}\{\psi_i\}$. Let the variance be $\sigma_\psi^2 \equiv \text{VAR}[\psi_i]$. Then the N terms have mean and variance $\lambda \mathrm{h}_i'$ and $\sigma_\psi^2 \mathrm{h}_i' \mathrm{h}_i'^T$ respectively. Assuming these quantities are finite, the law of large numbers (LLN) can be used to determine the asymptotic expected value

$$\frac{1}{N}\frac{\partial \mathscr{J}(\boldsymbol{\theta}_o)}{\partial \boldsymbol{\theta}} \to -\frac{\lambda}{N}\sum_{i=1}^{N}\mathrm{h}_i' \tag{2.85}$$

A similar use of the LLN applied to the summation in Equation (2.84) yields

$$\frac{1}{N}\frac{\partial^2 \mathscr{J}(\boldsymbol{\theta}_o)}{\partial \boldsymbol{\theta}^2} \to A \equiv -\frac{1}{N}\mathbb{E}\left\{\psi_i'\right\}\sum_{i=1}^{N}\mathrm{h}_i'^T\mathrm{h}_i' - \frac{\lambda}{N}\sum_{i=1}^{N}\mathrm{h}_i'' \tag{2.86}$$

The next step is to apply a form of the Central Limit Theorem to the terms in Equation (2.83). The particular form is given by Kevei for the scalar case [31]: *Given IID variables x_i with mean μ and finite variance, and constants $a_{i,N}$ such that* $\max_i a_{i,N} \to 0$ *as* $N \to \infty$

$$\sum_{i=1}^{N} a_{i,N}(x_i - \mu) \to \mathscr{N}(0,1) \tag{2.87}$$

if and only if

$$\sum_{i=1}^{N} a_{i,N}^2 = \text{VAR}[x_i]^{-1} \tag{2.88}$$

A corollary can be easily obtained from this and applied for our purposes. First add $\mu\sum_{i=1}^{N} a_{i,N}$ to the sum, assuming the series is convergent, to obtain

$$\sum_{i=1}^{N} a_{i,N}x_i \to \mathscr{N}\left(\mu\sum_{i=1}^{N} a_{i,N}, 1\right) \tag{2.89}$$

Now let $a_{i,N} = \frac{c_i}{N}\text{VAR}[x]^{-1/2}\frac{N}{\sqrt{\sum_{j=1}^{N} c_j^2}}$. This choice satisfies the condition in Equation (2.88). Therefore

$$\sum_{i=1}^{N}\frac{c_i}{N}\text{VAR}[x]^{-1/2}\frac{N}{\sum_{j=1}^{N} c_j^2}x_i \to \mathscr{N}(\mu\sum_{i=1}^{N} a_{i,N}, 1) \tag{2.90}$$

Dividing the terms by $\mathrm{VAR}[x]^{-1/2} \frac{N}{\sqrt{\sum_{j=1}^{N} c_j^2}}$ yields

$$\frac{1}{N}\sum_{i=1}^{N} c_i x_i \to \mathscr{N}\left(\mu N^{-1}\sum_{i=1}^{N} c_i, \mathrm{VAR}[x] N^{-2}\sum_{i=1}^{N} c_i^2\right) \tag{2.91}$$

This last equation is precisely what is needed to obtain the asymptotic result:

$$\frac{1}{N}\frac{\partial \mathscr{J}(\boldsymbol{\theta}_o)}{\partial \boldsymbol{\theta}} \to \mathscr{N}\left(-\frac{\lambda}{N}\sum_{i=1}^{N} \mathrm{h}_i', \sigma_\psi^2 \frac{1}{N^2} \mathrm{h}_i'^T \mathrm{h}_i'\right) \tag{2.92}$$

To see this most clearly, just note that the c_i in Equation (2.91) are any particular element of the vector h_i'.

Finally, using Equation (2.82) to obtain

$$\hat{\boldsymbol{\theta}} - \boldsymbol{\theta}_o = -\left(\frac{\partial^2 \mathscr{J}(\boldsymbol{\theta}_o)}{\partial \boldsymbol{\theta}^2}\right)^{-1} \frac{\partial \mathscr{J}(\boldsymbol{\theta}_o)}{\partial \boldsymbol{\theta}} \tag{2.93}$$

and applying Slutsky's Theorem with Equation (2.92) and Equation (2.86) yields

$$\hat{\boldsymbol{\theta}} \to \mathscr{N}(\boldsymbol{\mu}_\theta, P_\theta) \tag{2.94}$$

where

$$\boldsymbol{\mu}_\theta = A^{-1}\frac{\lambda}{N}\sum_{i=1}^{N} \mathrm{h}_i' \tag{2.95}$$

and

$$P_\theta = \frac{\sigma_\psi^2}{N^2} A^{-1}\left(\sum_{i=1}^{N} \mathrm{h}_i'^T \mathrm{h}_i'\right) A^{-1} \tag{2.96}$$

The special case of the location problem leads to the following simplifications:

$$A = -\mathbb{E}\left\{\psi_i'\right\} \tag{2.97}$$

$$\boldsymbol{\mu}_\theta = -\frac{\lambda}{\mathbb{E}\left\{\psi_i'\right\}} \tag{2.98}$$

$$P_\theta = \frac{1}{N}\frac{\sigma_\psi^2}{\mathbb{E}\left\{\psi_i'\right\}^2} \tag{2.99}$$

These equations are consistent with the results of Huber.

To show consistency with that of Hampel, we need to derive the influence function. The estimator at G will asymptotically satisfy

$$\mathbf{0} = \sum_i \left[\mathrm{h}_i' \int \psi(\mathbf{y}_i - \mathrm{h}_i(T(G)) - \boldsymbol{\nu}) dG(\boldsymbol{\nu}) \right] \tag{2.100}$$

Substituting $g(x) = (1-\varepsilon)f(x) + \varepsilon\delta_c(x)$ yields

$$\mathbf{0} = \sum_i \left[\mathrm{h}_i' \int \psi(\mathbf{y}_i - \mathrm{h}_i(T(G)) - \boldsymbol{\nu}) \left((1-\varepsilon)f(\boldsymbol{\nu}) + \varepsilon\delta_c(\boldsymbol{\nu})\right) d\boldsymbol{\nu} \right] \tag{2.101}$$

$$= \sum_i \Big[\varepsilon \mathrm{h}_i' \int \psi(\mathbf{y}_i - \mathrm{h}_i(T(G)) - \boldsymbol{\nu}) \left(\delta_c(\boldsymbol{\nu}) - f(\boldsymbol{\nu})\right) d\boldsymbol{\nu} + \mathrm{h}_i' \int \psi(\mathbf{y}_i - \mathrm{h}_i(T(G)) - \boldsymbol{\nu}) f(\boldsymbol{\nu}) d\boldsymbol{\nu} \Big] \tag{2.102}$$

Taking the derivative with respect to ε yields

$$\mathbf{0} = \sum_i \Big[\mathrm{h}_i' \int \psi(\mathbf{y}_i - \mathrm{h}_i(T(G)) - \boldsymbol{\nu}) \left(\delta_c(\boldsymbol{\nu}) - f(\boldsymbol{\nu})\right) d\boldsymbol{\nu} + \mathrm{h}_i' \mathrm{h}_i'^T \int \psi'(\mathbf{y}_i - \mathrm{h}_i(T(G)) - \boldsymbol{\nu}) \frac{\partial T(G)}{\partial \varepsilon} f(\boldsymbol{\nu}) d\boldsymbol{\nu} + \mathrm{h}_i'' \int \psi(\mathbf{y}_i - \mathrm{h}_i(T(G)) - \boldsymbol{\nu}) f(\boldsymbol{\nu}) d\boldsymbol{\nu} \Big] \tag{2.103}$$

Note the derivative of $T(G)$ with respect to ε is the influence function which can be pulled out of the integral. Doing this and evaluating at $\varepsilon = 0$ yields

$$\mathrm{IF}_{F,T}(c) = \frac{\partial T(G)}{\partial \varepsilon} \tag{2.104}$$

$$= M^{-1} \sum_i \left[\mathrm{h}_i' \int \psi(\mathbf{y}_i - \mathrm{h}_i(T(F)) - \boldsymbol{\nu}) \left(\delta_c(\boldsymbol{\nu}) - f(\boldsymbol{\nu})\right) d\boldsymbol{\nu} \right] \tag{2.105}$$

$$= M^{-1} \sum_i \left[\mathrm{h}_i' \int \psi(\mathbf{y}_i - \mathrm{h}_i(\boldsymbol{\theta}) - \boldsymbol{\nu}) \left(\delta_c(\boldsymbol{\nu}) - f(\boldsymbol{\nu})\right) d\boldsymbol{\nu} \right] \tag{2.106}$$

$$= M^{-1} \sum_i \left[\mathrm{h}_i' \int \psi(-\boldsymbol{\nu}) \left(\delta_c(\boldsymbol{\nu}) - f(\boldsymbol{\nu})\right) d\boldsymbol{\nu} \right] \tag{2.107}$$

$$= M^{-1} \sum_i \left[\mathrm{h}_i' \psi(-c) \right] \tag{2.108}$$

where the assumption that the estimator is consistent is used. The matrix M is

defined by

$$M \equiv \sum_i \Big[\mathrm{h}_i' \mathrm{h}_i'^T \int \psi'(-\boldsymbol{\nu}) f(\boldsymbol{\nu}) d\boldsymbol{\nu} + \mathrm{h}_i'' \int \psi(\boldsymbol{y}_i - \mathrm{h}_i(T(G)) - \boldsymbol{\nu}) f(\boldsymbol{\nu}) d\boldsymbol{\nu} \Big] \tag{2.109}$$

$$= \mathbb{E}\{\psi'\} \sum_i \mathrm{h}_i' \mathrm{h}_i'^T + \mathbb{E}\{\psi\} \sum_i \mathrm{h}_i'' \tag{2.110}$$

$$= -NA \tag{2.111}$$

Using Equation (2.48) with Equation (2.108) gives the variance

$$V(T,F) = M^{-1} \left[\int \sum_i \Big[\mathrm{h}_i' \psi(-c) \Big] \sum_i \Big[\mathrm{h}_i' \psi(-c) \Big]^T f(c) dc \right] M^{-T} \tag{2.112}$$

$$= \sigma_\psi^2 M^{-1} \sum_i \Big[\mathrm{h}_i' \mathrm{h}_i'^T \Big] M^{-T} \tag{2.113}$$

$$= \frac{\sigma_\psi^2}{N^2} A^{-1} \sum_i \Big[\mathrm{h}_i' \mathrm{h}_i'^T \Big] A^{-T} \tag{2.114}$$

which is equivalent to the derived variance in Equation (2.96). Therefore the equations given by Hampel, although derived through a different method, are equivalent to the above results for the given measurement model [27].

Note that if the the matrix $\boldsymbol{M}$ is singular, then the influence function and variance are undefined. This amounts to an observability condition on the measurement model.

Refering to Equation (2.95), it is clear that the estimator is Fisher consistent at the model if

$$\lambda \equiv \mathbb{E}\{\psi\} \tag{2.115}$$

$$= 0 \tag{2.116}$$

where the expectation is with respect to the assumed model. The ψ function defining

the estimator should be selected to satisfy this.

Now, if the above expression is nearly true at neighboring distributions and the second derivative terms are much smaller than the first derivative terms (i.e. nearly linear measurements) then the expression for A in Equation (2.86) reduces to

$$A = \frac{-1}{N}\mathbb{E}\left\{\psi'\right\}\sum_{i}^{N}\mathrm{h}_i'\mathrm{h}_i'^{T} \tag{2.117}$$

Then the estimator covariance expression reduces to

$$P_{\boldsymbol{\theta}} = \frac{\mathbb{E}\left\{\psi^2\right\}}{\mathbb{E}\left\{\left(\psi'\right)^2\right\}}\left[\sum_{i=1}^{N}\mathrm{h}_i'\mathrm{h}_i'^{T}\right]^{-1} \tag{2.118}$$

which is a very interesting (and apparently novel) result. It implies that for a given set of measurements (i.e. the functions $\mathrm{h}_i()$), **the M-estimator which optimizes any covariance criterion can be obtained by simply optimizing over the scalar**

$$v \equiv \frac{\mathbb{E}\left\{\psi^2\right\}}{\mathbb{E}\left\{\psi'\right\}^2} \tag{2.119}$$

which is precisely the scalar quantity studied in the location estimation problem! Therefore any results obtained in that problem are applicable to the more general linear regression problem (and approximately applicable to the nonlinear regression problem).

2.3.6 M-Estimates of Scale

M-estimators are generally not scale invariant. In particular,

$$\hat{\boldsymbol{\theta}}(cy_1, cy_2, \ldots) \neq c\hat{\boldsymbol{\theta}}(y_1, y_2, \ldots) \tag{2.120}$$

For example, the Huber estimate depends on the variance of the normal distribution being designed around. This is **not** the case for the least squares estimate. A second example is the use of a hard threshold for outlier rejection in Kalman-like sequential estimators: the threshold will depend on the scale of nominal errors. If the error scale is unknown *a priori* then it must be jointly estimated with the primary parameters of interest. This is the subject of the next section. This section will discuss estimators for pure scale problems which will be leveraged in the next section.

Consider measurements of the form

$$y_i = \nu_i \tag{2.121}$$

where ν_i is modeled as a random variable with nominal distribution

$$\nu_i \sim \frac{1}{\sigma} f\left(\frac{\nu_i}{\sigma}\right) \tag{2.122}$$

The MLE for this problem is

$$\hat{\sigma} = \text{argmax}_\sigma \prod_{i=1}^{N} \frac{1}{\sigma} f\left(\frac{y_i}{\sigma}\right) \tag{2.123}$$

$$= \text{argmax}_\sigma -N \log \sigma + \sum_{i=1}^{N} \log f\left(\frac{y_i}{\sigma}\right) \tag{2.124}$$

The solution satisfies

$$\sum_{i=1}^{N} \psi_s\left(\frac{y_i}{\sigma}\right) = 0 \tag{2.125}$$

where ψ_s is

$$\psi_s\left(\frac{y_i}{\sigma}\right) \equiv 1 + \sigma \frac{\partial}{\partial \sigma} \log f\left(\frac{y_i}{\sigma}\right) \tag{2.126}$$

$$= 1 - \frac{f'\left(\frac{y_i}{\sigma}\right)}{f\left(\frac{y_i}{\sigma}\right)} \frac{y_i}{\sigma} \tag{2.127}$$

The scale estimate can be expressed as a functional $S(F)$ of the underlying

distribution F (where F is the CDF corresponding to the PDF f). Asymptotically, the estimate satisfies

$$\int \psi_s\left(\frac{y}{S(G)}\right) dG(y) = 0 \tag{2.128}$$

under a distribution G. This expression can be used to study the breakdown point of the estimator by replacing G with $(1-t)F + t\delta_\infty$ for $t \in [0,1]$. This gives

$$0 = \int \psi_s\left(\frac{y}{S(G)}\right) ((1-t)F + t\delta_\infty) \tag{2.129}$$

$$= (1-t)\int \psi_s\left(\frac{y}{S(G)}\right) dF(y) + t\psi_s(\infty) \tag{2.130}$$

The estimate breaks down when t is large enough to make $S(G) \to \infty$. The smallest t to do so is the breakdown point. To solve for this, assume $S(G) \to \infty$ so that

$$0 = (1-t)\psi_s(0) + t\psi_s(\infty) \tag{2.131}$$

$$t = \frac{-\psi_s(0)}{\psi_s(\infty) - \psi_s(0)} \tag{2.132}$$

In order for the breakdown point to be non-zero, the ψ_s function must be bounded at ∞.

Equation (2.53) can be used to obtain the influence function which is

$$\mathrm{IF}(x,T,F) = \frac{\psi_s(x/S(F))}{\int \psi_s'(y/S(F))\frac{y}{S(F)^2} dF(y)} \tag{2.133}$$

The influence function can be used to compute the asymptotic variance using Equation (2.48) which yields

$$V(S,F) = \frac{\int \psi_s^2(x/S(F)) dF(y)}{\left(\int \psi_s'(y/S(F))\frac{y}{S(F)^2} dF(y)\right)^2} \tag{2.134}$$

Huber points out that choosing to find estimators, ψ_s, that minimize this variance (even with conditions on robustness) is a poor choice. Instead, optimizing over the

standardized variance, $V(S,F)/S^2$, leads to better results [29].

One such result is from Huber that is analogous to the minimax asymptotic variance estimator for ε-neighborhoods considered in a previous section. The estimator is

$$\psi_s(x) = \begin{cases} x^2 - 1 & |x| < x_1 \\ \\ x_1^2 - 1 & |x| > x_1 \geq \sqrt{2} \end{cases} \tag{2.135}$$

where x_1 is a constant determined by ε [25, 29]. Although optimal for variance, this estimator has either fairly large bias at the normal (54.7 % for $x_1 = 1.234$) or low breakdown point (< 20 % for $x_1 = 2.37$). Andrews *et al.* empirically showed that when scale is a nuisance parameter, it is more important to have low bias and high breakdown point event at the cost of higher variance [32]. Analytical reasoning for this will be presented in the next section on the joint location and scale problem. For the rest of this section, alternative scale estimators will be presented that will be leveraged in later sections.

Hampel gives a brief study of the median absolute deviation (MAD) in [26]. This scale estimate is

$$\hat{\sigma} = \text{Median}(y - \text{Median}(y)) \tag{2.136}$$

which has the maximum breakdown point (50 %) and lowest gross-error sensitivity. The downside is very low efficiency at the Gaussian which is pointed out by Rousseeuw who proposed alternatives with much higher efficiency and only slightly worse gross-error sensitivity [26] [33].

Many authors propose a MAD normalized (MADN) so that the estimate is asymptotically unbiased for the scale parameter of a particular distribution (for example: [34, 35]) For the normal distribution, this is

$$MADN(y) = \frac{MAD(y)}{0.675} \tag{2.137}$$

In most applications, the scale is not a primary parameter of interest. It is often treated as a nuisance parameter. The next section will discuss the use of scale estimates in combination with location estimates.

2.3.7 M-Estimates of Location and Scale

A key part of this dissertation is applying robust estimation techniques to vision-aided navigation which is a nonlinear regression problem. Nonlinear regression was discussed above for the case of an assumed error model that was fully specified (e.g. normal with a specific variance). However, in some cases, we would prefer to only assume a parametric error model that does not have all parameters specified (e.g. normal with unknown variance). In particular, the error scale may be unknown *a priori*. Before discussing this in the context of nonlinear regression, the problem of jointly estimating location and scale will be considered in this section. The intuitive concepts for the location and scale problem are applicable to the nonlinear regression with unknown scale problem.

Consider the measurement model

$$y_i = \theta + v_i \tag{2.138}$$

where v_i is modeled as a random variable with nominal distribution

$$v_i \sim \frac{1}{\sigma} f_1\left(\frac{v_i}{\sigma}\right) \tag{2.139}$$

The θ and σ are treated as unknowns with θ being the primary parameter of interest and σ being a nuisance parameter. The estimator of θ can be made scale invariant by jointly estimating σ.

There are three distinct approaches to the problem. The first is to jointly optimize a single cost function over the unknown parameter θ and the scale σ. In this method, the derivatives with respect to θ and with with respect to σ define two necessary conditions for the optimal estimate. Under certain conditions, these two equations can be treated as implicit definitions of the estimate. A second method is to define the estimate as the solution to two implicit equations to define the estimate. Unlike the first method, the two equations do not have to be derived from the same cost function. The third method is to obtain a scale-invariant estimate of θ (such as least squares), use the resulting residuals to estimate σ, and then use any estimator (not necessarily scale-invariant) to determine a new estimate of θ. Then the final estimate will be scale-invariant because the scale was determined entirely by the data without any *a prior assumption*.

Before discussing the mathematical details of the methods, a high-level comment is needed. The scale is a nuisance parameter and the need for an accurate estimate of it is driven only by the need for an accurate estimate of θ. Therefore the estimator of scale should be selected to tune the properties of θ rather than the

properties of the scale estimate itself.

First consider the MLE which maximizes

$$(\hat{\theta},\hat{\sigma}) = \text{argmax}_{(\theta,\sigma)} \prod_{i=1}^{N} \frac{1}{\sigma} f_1\left(\frac{y_i-\theta}{\sigma}\right) \tag{2.140}$$

$$= \text{argmax}_{(\theta,\sigma)} -N\log\sigma + \sum_{i=1}^{N} \log f_1\left(\frac{y_i-\theta}{\sigma}\right) \tag{2.141}$$

Taking the derivative with respect to the unknowns gives the necessary conditions

$$\sum_{i=1}^{N} \psi\left(\frac{y_i-\theta}{\sigma}\right) = 0 \tag{2.142}$$

$$\sum_{i=1}^{N} \psi_s\left(\frac{y_i-\theta}{\sigma}\right) = 0 \tag{2.143}$$

where $\psi(x)$ is the derivative of $-\log f_1(x)$ and $\psi_s(x) = \psi(x)x - 1$. To be clear

$$\psi\left(\frac{y_i-\theta}{\sigma}\right) \equiv \frac{\partial}{\partial\theta}\log f_1\left(\frac{y_i-\theta}{\sigma}\right) \tag{2.144}$$

$$= \frac{f_1'\left(\frac{y_i-\theta}{\sigma}\right)}{f_1\left(\frac{y_i-\theta}{\sigma}\right)}\frac{-1}{\sigma} \tag{2.145}$$

$$0 = \frac{-N}{\sigma} + \frac{\partial}{\partial\sigma}\log f_1\left(\frac{y_i-\theta}{\sigma}\right) \tag{2.146}$$

$$\psi_s\left(\frac{y_i-\theta}{\sigma}\right) \equiv \sigma\frac{\partial}{\partial\sigma}\left(-\log\sigma + \log f_1\left(\frac{y_i-\theta}{\sigma}\right)\right) \tag{2.147}$$

$$= \frac{f_1'\left(\frac{y_i-\theta}{\sigma}\right)}{f_1\left(\frac{y_i-\theta}{\sigma}\right)}\frac{-(y_i-\theta)}{\sigma} - 1 \tag{2.148}$$

$$= \left(\frac{y_i-\theta}{\sigma}\right)\psi\left(\frac{y_i-\theta}{\sigma}\right) - 1 \tag{2.149}$$

Any estimator defined through Equation (2.142) and Equation (2.143) with ψ and ψ_s related by $\psi_s(x) = \psi(x)x - 1$ is an MLE under some distribution. As already discussed, one approach to robust estimation is to choose that distribution such that it is similar to the nominal model and such that the resulting estimator is insensitive

to departures from the assumed model.

Another approach is to choose ψ and ψ_s **not** related by $\psi_s(x) = \psi(x)x - 1$ which gives the designer more flexibility in tuning the estimator. Huber's *Proposal 2* is an example of this [25]. That estimator sets ψ based on the minimax variance estimator for location and ψ_s based on the minimax variance estimator for scale.

Yet another approach is to first obtain a scale-invariant estimate of location, use the resulting residuals to estimate scale, and then use the scale estimate in a different location estimator.

For the first two methods, the influence function for the joint estimate, $[\theta, \sigma]$ can be computed from Equation (2.53) as

$$\mathrm{IF}(x,T,F) = \begin{bmatrix} \psi(x/S(F))S(F))/\int \psi'(x/S(F))dF(x) \\ \psi_s(x/S(F))S(F))/\int \psi_s'(x/S(F))\frac{x}{S(F)}dF(x) \end{bmatrix} \tag{2.150}$$

when ψ is odd and ψ_s is even which is required for scale and translation invariant estimators [27]. Note that the estimate $S(F)$ enters into the influence function for location (i.e. the first element of the vector) while the location component does not enter into the scale part. This means that the breakdown of the scale estimator implies breakdown of the location estimator but not vice-versa.

The *Proposal 2* estimator of Huber has been studied in the literature [25] which uses ψ and ψ_s from Equation (2.66) and Equation (2.135) which are the minimax asymptotic location and scale estimates respectively. The scale ψ_s from Equation (2.135) can be generalized as

$$\psi_s(x) = \min\left(x_1^2 - \beta, x^2 - \beta\right) \tag{2.151}$$

where $\beta = 1$ in the original equation. The parameter β can be selected for asymptotic consistency at a particular model which implies that $\beta = \beta(x_1)$ In that case, and with $x_1 = k$, the breakdown point is

$$\varepsilon = \frac{\beta(k)}{\beta(k) + k^2} \tag{2.152}$$

As k increases from zero, $\beta(k)$ decreases, and the net effect is a reduction in the breakdown point but an increase in efficiency at the model: a common theme in robust estimation!

Huber tabulates the worst-case asymptotic variance of location under symmetric ε-contamination for estimators studied considered in the Princeton study (for many ε values) [29, 32]. Monte Carlo results are also tabulated in [36]. There are some clear trends. First, for moderate contamination $\varepsilon \in [0.01, 0.05]$, the Huber location cost of Equation (2.66), performs just as well when the scale determined either with Equation (2.135) or with the MAD. However, if there is a chance of large ε-contamination, then the MAD scale estimate is better than Equation (2.135) when paired with the ψ_s of Equation (2.66) (i.e. it is more robust for the same efficiency loss at the model). The redescending M-estimators of Hampel paired with MAD provide even greater robustness for large ε-contamination with reasonable efficiency loss [27]. These qualitative trends are important in designing estimators for nonlinear regression with unknown scale.

2.3.8 Nonlinear Regression with Unknown Scale

Nonlinear regression for the case of known error scale was addressed above. This section will discuss the problem when this scale is unknown and must be determined simultaneously.

The measurement model is

$$\boldsymbol{y}_i = \mathrm{h}_i(\boldsymbol{\theta}) + \boldsymbol{\nu}_i \tag{2.153}$$

where the unknown error $\boldsymbol{\nu}_i$ is assumed to come from a distribution F of the form

$$F(\boldsymbol{\nu}) = \frac{1}{\sigma} F_1(\boldsymbol{\nu}/\sigma) \tag{2.154}$$

The normal distribution with variance σ^2 is one of many possible models with this form (also the Laplace distribution and Cauchy distribution).

The MLE solution is

$$\hat{\boldsymbol{\theta}} = \mathrm{argmax}_{\boldsymbol{\theta},\sigma} \prod \frac{1}{\sigma} F_1((\boldsymbol{y}_i - \mathrm{h}_i(\boldsymbol{\theta}))/\sigma) \tag{2.155}$$

$$= \mathrm{argmax}_{\boldsymbol{\theta},\sigma} \prod \frac{1}{\sigma} F_1((\boldsymbol{y}_i - \mathrm{h}_i(\boldsymbol{\theta}))/\sigma) \tag{2.156}$$

$$= \mathrm{argmin}_{\boldsymbol{\theta},\sigma} N \log \sigma - \frac{1}{N} \sum F_1((\boldsymbol{y}_i - \mathrm{h}_i(\boldsymbol{\theta}))/\sigma) \tag{2.157}$$

The necessary conditions are found by taking the derivative of the above equation with respect to both $\boldsymbol{\theta}$ and σ. This yields

$$\mathbf{0} = \sum \frac{f_1'((\boldsymbol{y}_i - \mathrm{h}_i(\boldsymbol{\theta}))/\sigma)}{f_1((\boldsymbol{y}_i - \mathrm{h}_i(\boldsymbol{\theta}))/\sigma)} \mathrm{h}_i'(\boldsymbol{\theta}) \tag{2.158}$$

$$1 = \frac{1}{N} \sum \frac{f_1'((\boldsymbol{y}_i - \mathrm{h}_i(\boldsymbol{\theta}))/\sigma)}{f_1((\boldsymbol{y}_i - \mathrm{h}_i(\boldsymbol{\theta}))/\sigma)} ((\boldsymbol{y}_i - \mathrm{h}_i(\boldsymbol{\theta}))/\sigma) \tag{2.159}$$

To generalize these necessary conditions for an M-estimator, define a cost $\boldsymbol{\rho}()$ which

is analogous to $-\log f_1$ and then

$$\mathbf{0} = \sum \psi(r_i) \mathrm{h}_i'(\boldsymbol{\theta})$$

$$\mathbf{0} = \sum \psi_s(r_i) \tag{2.160}$$

where $\psi(r) = \rho'(r)$, $\psi_s(r) = \psi(r)r - 1$, and $r_i = (\boldsymbol{y}_i - \mathrm{h}_i(\boldsymbol{\theta}))/\sigma$.

There are no straight-forward minimax results for this problem. Nevertheless, expressions for the asymptotic bias and covariance are useful in assessing tradeoffs in the estimator design. The derivation is similar to that of the nonlinear regression with known scale case.

Begin with the necessary conditions in Equation (2.160). Perform a Taylor series expansion about some nominal parameter value $(\boldsymbol{\theta}_o, \sigma_o)$.

$$\mathbf{0} = \begin{bmatrix} \frac{1}{N}\sum \psi(r_i)\mathrm{h}_i'(\boldsymbol{\theta}) \\ \frac{1}{N}\sum \psi_s(r_i) \end{bmatrix}_{(\hat{\boldsymbol{\theta}},\hat{\sigma})} \tag{2.161}$$

$$\approx \begin{bmatrix} \frac{1}{N}\sum \psi(r_i)\mathrm{h}_i'(\boldsymbol{\theta}) \\ \frac{1}{N}\sum \psi_s(r_i) \end{bmatrix}_{(\boldsymbol{\theta}_o,\sigma_o)} + A \begin{bmatrix} \hat{\boldsymbol{\theta}} - \boldsymbol{\theta}_o \\ \hat{\sigma} - \sigma_o \end{bmatrix} \tag{2.162}$$

$$A \equiv \frac{\partial}{\partial[\boldsymbol{\theta}, \sigma]} \begin{bmatrix} \frac{1}{N}\sum \psi(r_i)\mathrm{h}_i'(\boldsymbol{\theta}) \\ \frac{1}{N}\sum \psi_s(r_i) \end{bmatrix} \tag{2.163}$$

$$= \frac{1}{N}\sum \begin{bmatrix} \left(\psi(r_i)\mathrm{h}_i'' - \psi'(r_i)\mathrm{h}_i'\mathrm{h}_i'^T/\sigma\right) & \left(\frac{1}{\sigma}\psi'(r_i)r_i\mathrm{h}_i'\right) \\ \frac{-1}{\sigma}\psi_s'(r_i)\mathrm{h}_i' & \frac{-1}{\sigma}\psi_s'(r_i)r_i \end{bmatrix}_{(\boldsymbol{\theta}_o,\sigma_o)} \tag{2.164}$$

Making use of Slutsky's Theorem as in the section on nonlinear regression yields

the asymptotic distribution of the M-estimate as

$$\begin{bmatrix} \hat{\boldsymbol{\theta}} - \boldsymbol{\theta}_o \\ \hat{\sigma} - \sigma_o \end{bmatrix} \sim \mathcal{N}(\boldsymbol{\mu}_{\bar{\boldsymbol{\theta}}}, P_{\bar{\boldsymbol{\theta}}}) \tag{2.165}$$

where

$$\boldsymbol{\mu}_{\bar{\boldsymbol{\theta}}} = \mathbb{E}\{A\}^{-1}\mathbb{E}\left\{ \begin{bmatrix} \frac{1}{N}\sum \psi(r_i)\mathrm{h}_i'(\boldsymbol{\theta}) \\ \frac{1}{N}\sum \psi_s(r_i) \end{bmatrix}_{(\boldsymbol{\theta}_o,\sigma_o)} \right\} \tag{2.166}$$

$$P_{\bar{\boldsymbol{\theta}}} = \mathbb{E}\{A\}^{-1}\mathbb{COV}\left\{ \begin{bmatrix} \frac{1}{N}\sum \psi(r_i)\mathrm{h}_i'(\boldsymbol{\theta}) \\ \frac{1}{N}\sum \psi_s(r_i) \end{bmatrix}_{(\boldsymbol{\theta}_o,\sigma_o)} \right\}\mathbb{E}\{A\}^{-T} \tag{2.167}$$

The limiting quantities in the above expression can be evaluated as

$$\mathbb{E}\{A\} = \frac{1}{N}\sum \begin{bmatrix} \mathbb{E}\{\psi(r)\}\mathrm{h}_i'' - \mathbb{E}\{\psi'(r)\}\mathrm{h}_i'\mathrm{h}_i'^T/\sigma & \frac{1}{\sigma}\mathbb{E}\{\psi'(r)r\}\mathrm{h}_i' \\ \frac{-1}{\sigma}\mathbb{E}\{\psi_s'(r)\}\mathrm{h}_i' & \frac{-1}{\sigma}\mathbb{E}\{\psi_s'(r)r\} \end{bmatrix} \tag{2.168}$$

$$\mathbb{COV}\left\{ \begin{bmatrix} \frac{1}{N}\sum \psi(r_i)\mathrm{h}_i'(\boldsymbol{\theta}) \\ \frac{1}{N}\sum \psi_s(r_i) \end{bmatrix} \right\} = \begin{bmatrix} \frac{1}{N}\mathrm{VAR}[\psi(r)]\sum \mathrm{h}_i'\mathrm{h}_i'^T & \frac{1}{N}\mathbb{E}\{\psi(r)\psi_s(r)\}\sum \mathrm{h}_i' \\ \frac{1}{N}\mathbb{E}\{\psi(r)\psi_s(r)\}\sum \mathrm{h}_i'^T & \mathrm{VAR}[\psi_s(r)] \end{bmatrix} \tag{2.169}$$

The above resulting equations are cumbersome. Some simplifications can be made for many practical problems. As an example, consider the case where the measurements are linear and distributed as

$$y_i \sim \mathcal{N}(\mathrm{h}_i'^{\mathrm{T}}\boldsymbol{\theta}, \sigma_o^2) \tag{2.170}$$

and the estimators is designed around the Gaussian assumption:

$$\psi(x) = x \tag{2.171}$$

$$\psi_s(x) = x^2 - 1 \tag{2.172}$$

This example will demonstrate how to apply the equations and will also verify that they are consistent with this classical problem. There are several expectations that must be taken under the assumption that $\hat{\boldsymbol{\theta}} = \boldsymbol{\theta}$.

$$\mathbb{E}\{\psi(r)\} = 0 \tag{2.173}$$

$$\mathbb{E}\left\{\psi'(r)\right\} = 1 \tag{2.174}$$

$$\mathbb{E}\{\psi_s(r)\} = \sigma_o^2/\sigma^2 - 1 \tag{2.175}$$

$$\mathbb{E}\left\{\psi'(r)r\right\} = 0 \tag{2.176}$$

$$\mathbb{E}\left\{\psi_s'(r)\right\} = 2\sigma_o^2/\sigma^2 \tag{2.177}$$

$$\mathbb{E}\{\psi(r)\psi_s(r)\} = 0 \tag{2.178}$$

$$\text{VAR}[\psi(r)] = \sigma_o^2/\sigma^2 \tag{2.179}$$

$$\text{VAR}[\psi_s(r)] = 2\sigma_o^4/\sigma^4 \tag{2.180}$$

Then the above matrices reduce to the following

$$\mathbb{E}\{A\} = \begin{bmatrix} \frac{-1}{N\sigma^2}\sum \mathrm{h}_i'\mathrm{h}_i'^{\mathrm{T}} & 0 \\ 0 & -2\sigma_o^2/\sigma^3 \end{bmatrix} \tag{2.181}$$

$$\mathbb{COV}\left\{\begin{bmatrix} \frac{1}{N}\sum \psi(r_i)\mathrm{h}_i'(\boldsymbol{\theta}) \\ \frac{1}{N}\sum \psi_s(r_i) \end{bmatrix}\right\} = \begin{bmatrix} \frac{\sigma_o^2}{N\sigma^4}\mathrm{h}_i'(\boldsymbol{\theta}) & 0 \\ 0 & \frac{2\sigma_o^2}{N\sigma^2} \end{bmatrix} \tag{2.182}$$

The estimator bias is asymptotically zero and the covariance is asymptotically

$$P_{\hat{\boldsymbol{\theta}}} = \begin{bmatrix} \sigma_o^2 \left(\sum \mathrm{h}_i' \mathrm{h}_i'^{\mathrm{T}}\right)^{-1} & 0 \\ 0 & \frac{\sigma_o^2}{2N} \end{bmatrix} \tag{2.183}$$

which is in agreement with results for classical linear regression under normal errors. Obviously the advantage of Equation (2.166) and Equation (2.167) is that any pair of estimator and distribution can be used.

2.3.9 Leverage Points

Consider the simplified expression for the covariance of a parameter vector $\boldsymbol{\theta}$ in the nonlinear regression case:

$$P_{\boldsymbol{\theta}} = \frac{\mathbb{E}\left\{\psi^2\right\}}{\mathbb{E}\left\{\left(\psi'\right)^2\right\}} \left[\sum_{i=1}^{N} \mathrm{h}_i' \mathrm{h}_i'^{T}\right]^{-1} \tag{2.184}$$

This expressions assumes that asymptotic results apply to all elements of $\boldsymbol{\theta}$. Even if the number of measurements goes to infinity, asymptotic results may not necessarily apply unless all parameters are *infinitely observable*: each element of $\boldsymbol{\theta}$ must show up in an infinity of measurements (loosely speaking). Remember the incidental parameters problem!

The issue becomes especially precarious for applications with a finite number of measurements where the h_i equations are not explicitly known ahead of time. The vision-aided navigation problem is a prime example of this! Visual features are identified autonomously in real-time as a camera observes the environment. The locations of the features ultimately drive the shape of the h_i equations. Those readers familiar with the problem of attitude estimation from star line-of-sight measurements

will appreciate the following example which is similar in structure.

Consider a spacecraft with two narrow field-of-view (FOV) star trackers pointed in orthogonal directions. The first star tracker sees a large number of stars (up to approximately 20) which gives good observability on two degrees of freedom (DOF) of attitude. The third DOF, namely roll about the first star tracker's boresight can be estimated without using the second star tracker, although with much less accuracy. Asymptotic theory suggests an MLE for all three DOF will tend to a Gaussian about the true attitude with decreasing covariance as the number of measurements increases (albeit a non-isotropic Gaussian). Now assume we get a single star measurement from the second, orthogonally oriented, star tracker. While we expect the measurement to be useful from geometric considerations (the corresponding $\mathbf{h}_i'\mathbf{h}_i'^T$ term will reduce the maximum eigenvalue of the covariance matrix), the unfortunate side effect is that the asymptotic theory breaks down: a single measurement almost entirely determines the estimate of one element of $\boldsymbol{\theta}$. Even if the breakdown point of the estimator's $\boldsymbol{\psi}$ function is high, a single bad measurement with undue influence can ruin the overall estimate. Such measurements are referred to as *leverage points*.

A more canonical example of a leverage point comes in the problem of estimating the slope and intercept of a line on the (x,y) plane. If all but one of the measurements are clustered near $x = 0$ and one measurement takes on a very large x value, then the latter measurement will have undue influence on the resulting estimate.

There is no clear answer to this problem in the literature. On the one hand, leverage points are *good* in that they can give great observability to certain parameters. On the other hand, they are *bad* because they can effectively reduce (significantly) the breakdown point of an estimator. Intuitively, if the information provided by a leverage point is consistent with the rest of the data, then it should be given large weight. Similarly, if the leverage point drastically changes the estimate and greatly increases the residuals of all other measurements, then it should be down weighted. Hampel gives various approaches to doing so [27]. The main idea is to adjust the estimator necessary condition from

$$\sum \psi(r_i)\mathrm{h}_i' = \mathbf{0} \tag{2.185}$$

to something of the form

$$\sum w_i \psi(r_i/\delta_i)\mathrm{h}_i' = \mathbf{0} \tag{2.186}$$

The w_i can be selected to down weight measurements while the δ_i term can be used to scale residuals outside of the *inlier region* of the ψ section [25, 27, 37]. Mallows sets $\delta_i = 1$ and relies entirely on the w_i which is determined as follows. [38, 39]

Let P be the the so-called *hat matrix* which transforms the i'th measurement into the i'th predicted measurement in a linear model (for example, see [40]). It is

called this because the matrix *puts the hat on the measurement*:

$$P_{ij} \equiv h_i'^{\mathrm{T}} \left[\sum_k h_k' h_k'^{\mathrm{T}} \right]^{-1} h_j' \tag{2.187}$$

$$\hat{y}_i \equiv \mathbb{E}\{y_i | y_1, y_2, \ldots, y_N\} \tag{2.188}$$

$$= \sum_j P_{ij} y_j \tag{2.189}$$

$$\hat{\boldsymbol{y}} = P\boldsymbol{y} \tag{2.190}$$

From classical least squares analysis, the variance of the fitted residual $\hat{y}_i - y_i$ under Gaussian errors of variance σ^2 is

$$\mathrm{VAR}[\hat{y}_i - y_i] = (1 - P_{ii})\sigma^2 \tag{2.191}$$

The Mallows estimator sets

$$w_i = \sqrt{1 - P_{ii}} \tag{2.192}$$

which is essentially adjusting the scale σ based on the quantity P_{ii} If $P_{ii} = 1$, then this means that one, and only one, measurement impacted the fit of $\hat{y}_i$ (namely y_i itself). Clearly this is not a desirable situation. On the other end of the spectrum, as the number of measurements contributing to the fit of $\hat{y}_i$ increases to infinity, P_{ii} will tend to zero and the weight will tend to unity. The downside of this estimator is that it **always** down weights points with high leverage: even if they are consistent with the data. The estimator of Schweppe seeks to overcome this.

The Schweppe estimator chooses the same weight as the Mallows estimator. In contrast to Mallows, Schweppe sets $\delta_i = w_i$ [41]. Therefore if we have a ψ function like that of Huber, then a large but accurately measured leverage point will fall into

the linear region of ψ causing the w_i and δ_i to cancel out and giving the measurement high weight. On the other hand, a poor measurement with high leverage can fall outside of the linear region of ψ and the measurement is treated as in the Mallows estimator.

The drawback of the Schweppe estimator is a lack of efficiency at the model [37]. Krasker and Welsch seek to overcome this by deriving an estimator that minimizes asymptotic variance subject to a bound on sensitivity [38]. This is consistent with the general approach of Hampel to deriving robust estimators [27]. The full details of the Krasker-Welsch estimator will not be repeated here as they are contained in the reference.

In summary, the existence of leverage points can limit the applicability of the asymptotic theory used to derive traditional M-estimators. While the literature contains a variety of methods for detecting and dealing with leverage points, there is no universal solution. Therefore studying the extent to which they exist in a given application and how to properly incorporate them into the final estimate is critical. Such a study will be presented in this dissertation for vision-aided navigation that has not been shown before.

2.3.10 Leverage Point Example: Attitude Estimation

The star tracker problem was used as an example to discuss the potential dangers introduced by leverage points. A numerical example was designed to demonstrate this in more detail. Interestingly, while robust estimators have been applied to

attitude estimation (see [42]), it does not appear that *leverage-resistant* estimators have been used for the attitude estimation problem. Therefore this section presents a novel contribution of this dissertation.

To perform this study, $n_T = 2000$ trials of the following experiment were repeated. A set of n_1 and n_2 stars (i.e. inertial frame unit vectors) were randomly generated in two 25° field-of-view sensors directed along $\boldsymbol{v}_1 = [0,0,1]$ and $\boldsymbol{v}_2 = [1,0,0]$ respectively. These inertial frame unit vectors were used to simulate image-plane measurements for two orthogonally directed star trackers using the simple projection equation:

$$\boldsymbol{y}_{i,k} = \begin{bmatrix} y_{i,k,1} \\ \\ y_{i,k,2} \end{bmatrix} = \boldsymbol{\pi}(R_i R_{s/c} \boldsymbol{u}_k) + \boldsymbol{\nu}_{i,k} \tag{2.193}$$

where R_i is the orientation of the i'th star tracker relative to the spacecraft frame, $R_{s/c}$ is the spacecraft attitude relative to the inertial frame, and $\boldsymbol{u}_k$ is a particular star. The function $\boldsymbol{\pi}()$ is the simple projection function

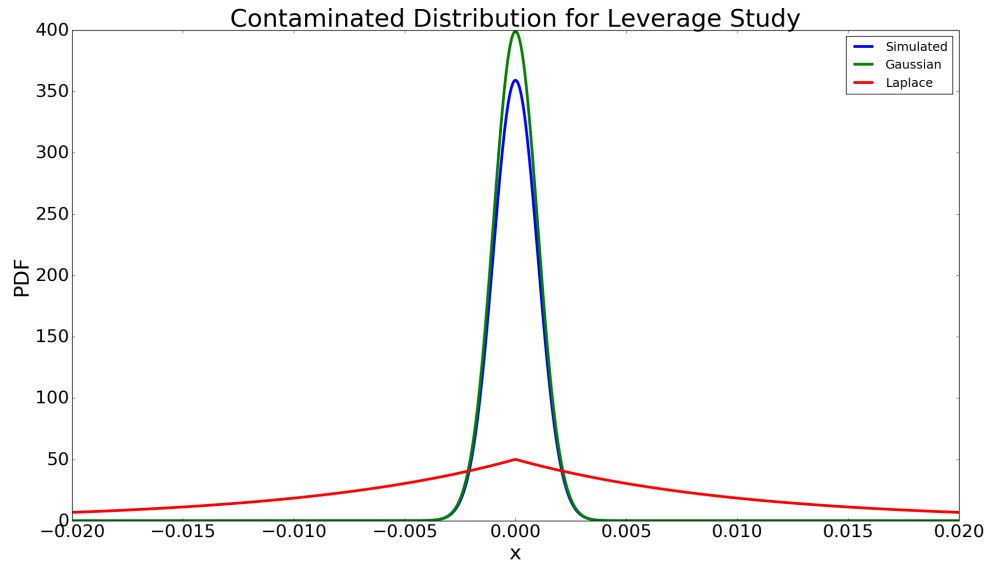
$$\begin{bmatrix} x/z \\ \\ y/z \end{bmatrix} = \boldsymbol{\pi}([x,y,z]) \tag{2.194}$$

The random noise $\boldsymbol{\nu}_{i,k}$ is sampled from the contaminated distribution

$$\mathrm{p}(\boldsymbol{\nu}) = 0.9\mathscr{N}(0,0.001) + 0.1\mathscr{L}(0,0.01) \tag{2.195}$$

which is an $\varepsilon = 0.1$ contaminated normal distribution with $\sigma = 0.001$ and with a Laplace distribution of scale $\sigma = 0.01$ as the contaminant. Note that the combined variance of this distribution is 0.0045^2. A plot of this distribution with the component Gaussian and Laplace overlaid is shown in Figure (2.8).

Figure 2.8: PDF of the contaminated distribution used for the experiments in the leverage point study. The PDF is a mixture of the Gaussian (green) and Laplace (red) PDFs.



The simulated measurements are passed to two estimators. The first estimator is the Huber M-estimator:

$$\hat{R} \equiv \text{argmin} \sum_{i=1}^{2} \sum_{k=1}^{n_i} \sum_{j=1}^{2} \rho((y_{i,k,j} - \hat{y}_{i,k,j})/\sigma) \;\; s.t. \;\; \hat{R} \in \text{SO}(3) \tag{2.196}$$

and the second estimator is a Schweppe-type estimator [43, 29, 27]:

$$\hat{R} \equiv \text{argmin} \sum_{i=1}^{2} \sum_{k=1}^{n_i} \sum_{j=1}^{2} w_{i,k,j}^2 \rho((y_{i,k,j} - \hat{y}_{i,k,j})/(\sigma w_{i,k,j})) \;\; s.t. \;\; \hat{R} \in \text{SO}(3)$$
$$w_{i,k,j} = 1 - \left[H(H^T H)^{-1} H^T\right]_{\ell\ell} \tag{2.197}$$

where H is the Jacobian of the stacked measurement vector with respect to a small change in attitude (small angle approximation [17]), ℓ is the index into the stacked measurement vector corresponding to $y_{i,k,j}$, and $\rho()$ is the Huber cost defined in Equation (2.65) with $k = 3.0$. Also, $\hat{y}_{i,k,j}$ is the predicted measurement conditioned

on $\hat{R}$. Note that the summation of i sums over the two sensors and the summation j sums over the two image-plane components of each sensor.

For this study, two cases are considered: $\{n_1 = 0, n_2 = 10\}$ and $\{n_1 = 1,\ n_2 = 9\}$. The first case represents a control case as we expect the amount of leverage for any particular measurement to be small when all measurements come from the same sensor. In this first case, the rotation about the $\boldsymbol{v}_2$ axis will be the most uncertain as all measurements are concentrated along that direction. The second case is designed to have a single measurement with very high leverage. In particular, this will be the second component of the only measurement in the first sensor: $y_{1,1,2}$. Intuitively, the estimator will adjust the roll about the $\boldsymbol{v}_2$ direction almost entirely based on this single measurement. Therefore asymptotic results do not apply and a bad measurement can seriously corrupt the resulting attitude estimate. *Note that on any given trial, the same measurements are passed to both estimators in both cases.*

The results for the first case are shown in Figure (2.9) and Figure (2.10). Both estimators give similar performance with a slight advantage for the Scweppe estimator. This is because for only 10 measurements which had a (nearly) uniform random location in the second sensor field-of-view, some may have moderate leverage and hence the moderate advantage in some cases. However, extremely high leverage points are very unlikely in this case.

On the other hand, there is one extremely high leverage point in every trial of the second case. The results of this are shown in Figure (2.11) and Figure (2.12).

Figure 2.9: Histogram of attitude errors for the $\{n_1 = 0,\ n_2 = 10\}$ case (absolute value of Euler angle).

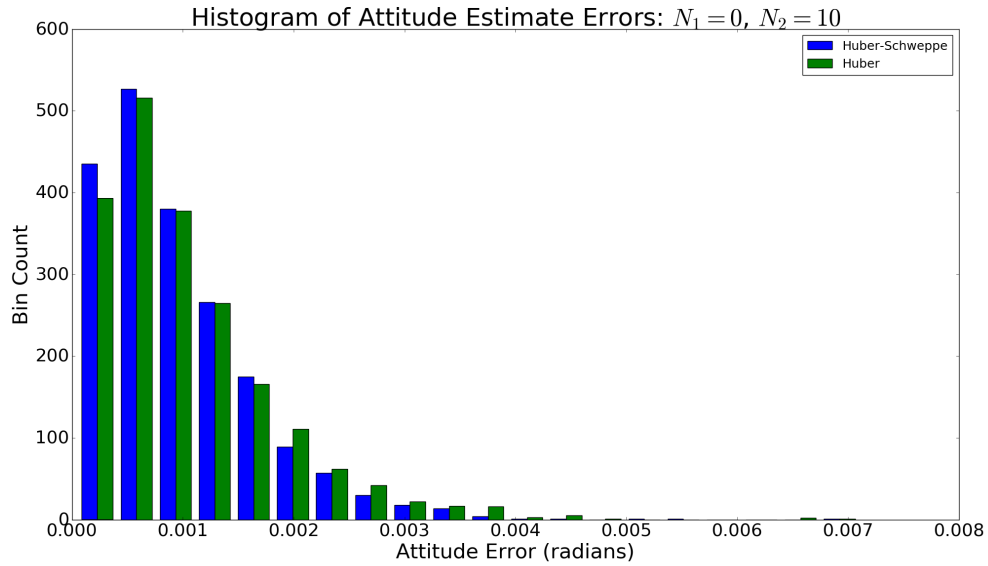


Figure 2.10: Histogram of the difference in attitude errors for the $\{n_1 = 0, n_2 = 10\}$ case.

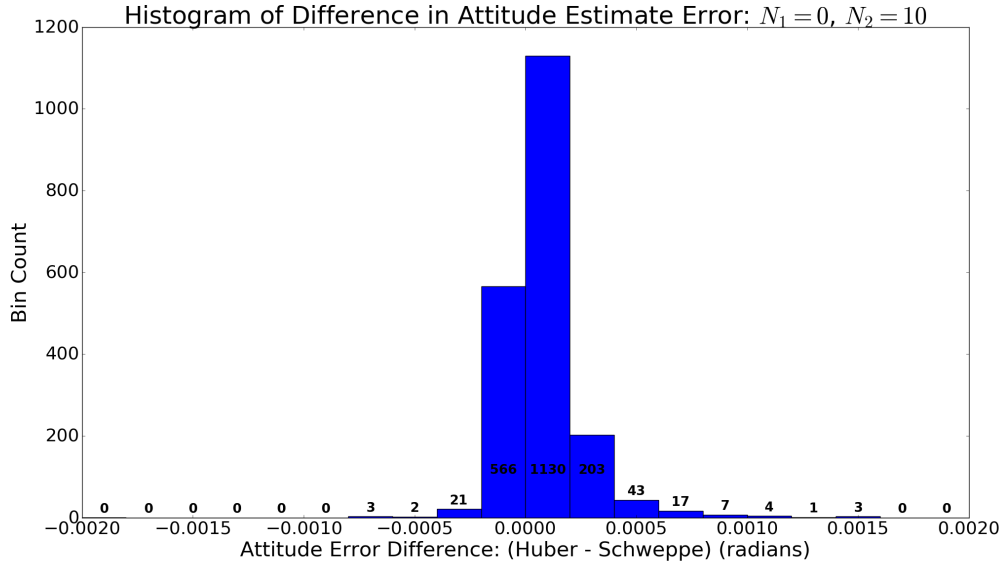
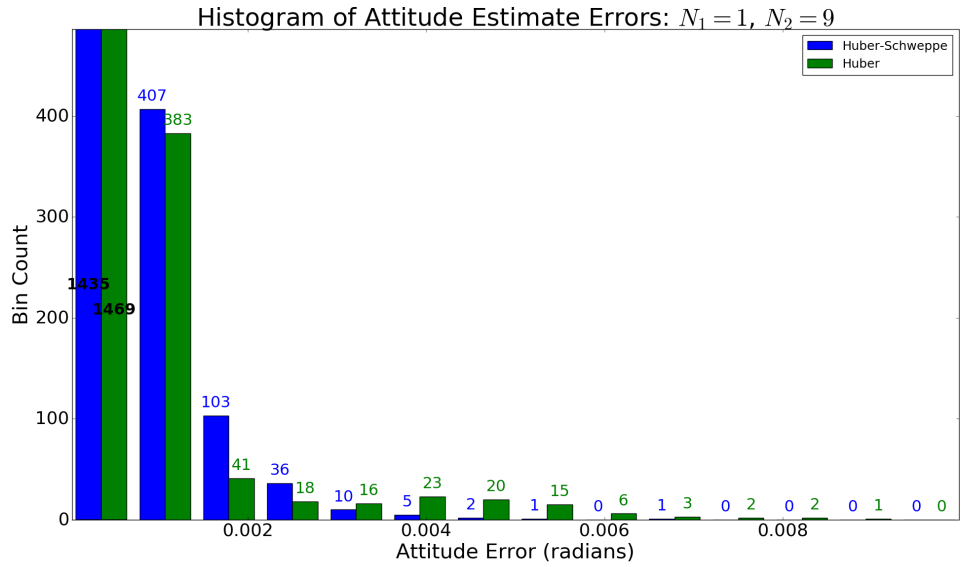


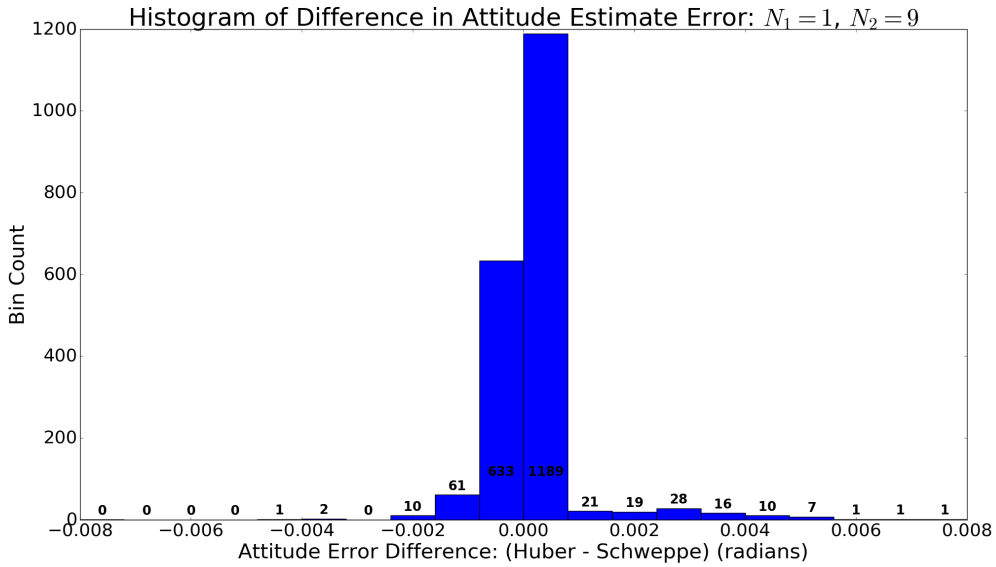
Figure 2.11: Histogram of attitude errors for the $\{n_1 = 1,\ n_2 = 9\}$ case (absolute value of Euler angle).



The results are very striking. The Scweppe-type estimator outperforms the Huber estimator in 65% of cases but the more interesting part is how the especially poor estimates are nearly eliminated by the Scweppe-type estimator. At the 99'th percentile, the Huber estimates have nearly double (1.92 times) the error as the Scweppe-type estimates. This behavior was anticipated in the above discussion.

One final point should be made that is not clear in the figures above. In the *typical case*, the addition of one measurement along the first sensor direction does improve the estimates for both estimators. The median attitude error for the Huber estimate for $n_1 = 0$ and $n_1 = 1$ was 8.42×10^{-4} and 4.92×10^{-4} radians respectively. For the Scweppe-type estimator, the median attitude error for the Huber estimate for $n_1 = 0$ and $n_1 = 1$ was 7.84×10^{-4} and 4.97×10^{-4} radians respectively. Clearly, on

Figure 2.12: Histogram of the difference in attitude errors for the $\{n_1 = 1, n_2 = 9\}$ case.



the same test data, the two estimators give similar performance for the typical case. It is the odd event of a single bad measurement at a leverage point that corrupts the Huber estimator and gives the Scweppe-type estimator the advantage.

2.4 Computing M-Estimates

The previous sections discussed the theory behind M-estimates. This section will briefly demonstrate how to actually compute M-estimates. Any nonlinear iterative least squares software can be easily modified to do so. A good implementation of the Levenberg-Marquardt algorithm or the Dog-Leg algorithm are perfect starting places for an M-estimation solver [44, 45, 46]. In their basic form, these algorithms

can solve problems of the form

$$\hat{\boldsymbol{\theta}} = \text{argmin}_{\boldsymbol{\theta}} \left(\boldsymbol{y} - \boldsymbol{h}(\boldsymbol{\theta})\right)^T W \left(\boldsymbol{y} - \boldsymbol{h}(\boldsymbol{\theta})\right) \tag{2.198}$$

where both $\boldsymbol{y}$ and the function $\boldsymbol{h}()$ are vector-valued with elements y_i and $\mathrm{h}_i()$. If W is a diagonal matrix with elements w_i then the above problem is equivalent to

$$\hat{\boldsymbol{\theta}} = \text{argmin}_{\boldsymbol{\theta}} \sum_i \left(y_i - \mathrm{h}_i(\boldsymbol{\theta})\right)^2 w_i \tag{2.199}$$

The necessary condition is

$$\mathbf{0} = \sum_i H_i \left(y_i - \mathrm{h}_i(\hat{\boldsymbol{\theta}})\right) w_i \tag{2.200}$$

where H_i is the Jacobian evaluated at the solution. The optimization algorithms iteratively compute parameter corrections and terminate once the above necessary condition is met to within some tolerance. Typically w_i is a constant held fixed across iterations.

The *iterative weighting approach* adjusts the w_i between iterations to find an M-estimate solution to

$$\hat{\boldsymbol{\theta}} = \text{argmin}_{\boldsymbol{\theta}} \sum_i \rho\left(y_i - \mathrm{h}_i(\boldsymbol{\theta})\right) \tag{2.201}$$

which has the necessary condition

$$\mathbf{0} = \text{argmin}_{\boldsymbol{\theta}} \sum_i H_i \rho'\left(y_i - \mathrm{h}_i(\boldsymbol{\theta})\right) \tag{2.202}$$

The weight is chosen as

$$w_i^{(k+1)} = \frac{\rho'\left(y_i - \mathrm{h}_i(\boldsymbol{\theta}^{(k)})\right)}{\left(y_i - \mathrm{h}_i(\boldsymbol{\theta}^{(k)})\right)} \tag{2.203}$$

From this weighting scheme and Equation (2.200), the necessary condition at con-

vergence is

$$\mathbf{0} = \sum_i H_i \left(y_i - \mathrm{h}_i(\hat{\boldsymbol{\theta}})\right) \frac{\psi\left(y_i - \mathrm{h}_i(\hat{\boldsymbol{\theta}})\right)}{\left(y_i - \mathrm{h}_i(\hat{\boldsymbol{\theta}})\right)} \tag{2.204}$$

which is the same necessary condition for the solution of the M-estimation problem in Equation (2.202).

Note that if some measurements are inherently coupled and can not be broken up into independent scalar components, a modification must be made. Consider for example an error $\boldsymbol{\nu}$ that is Gaussian with covariance R which is not diagonal. The weight is computed in a completely analogous way. The square-root of the weighted-least-squares cost $\sqrt{r^T R^{-1} r}$ is computed. The weight is then simply

$$w(r) = \frac{\psi\left(\sqrt{r^T R^{-1} r}\right)}{\sqrt{r^T R^{-1} r}} \tag{2.205}$$

2.4.1 An Important Note

The previous section presented a method to compute the robust parameter estimate by modifying an iterative least squares solver. Huber refers to this as the *location-step* and is known more generally as iterative reweighted least squares [47]. The excellent book on camera geometry and estimation by Hartley and Zisserman includes an appendix on robust estimation [3]. However, the weighting function given there is different. In particular, they give

$$w(r) = \sqrt{\frac{\rho(r)}{r^2}}$$

This seems to be motivated by the heuristic that scaling the least squares cost r^2 by this weight gives the desired cost $\rho(r)$. This has been used for example in [48].

Mathematically, it appears that these two algorithms are **not** the same. A simple example demonstrates that this is the case.

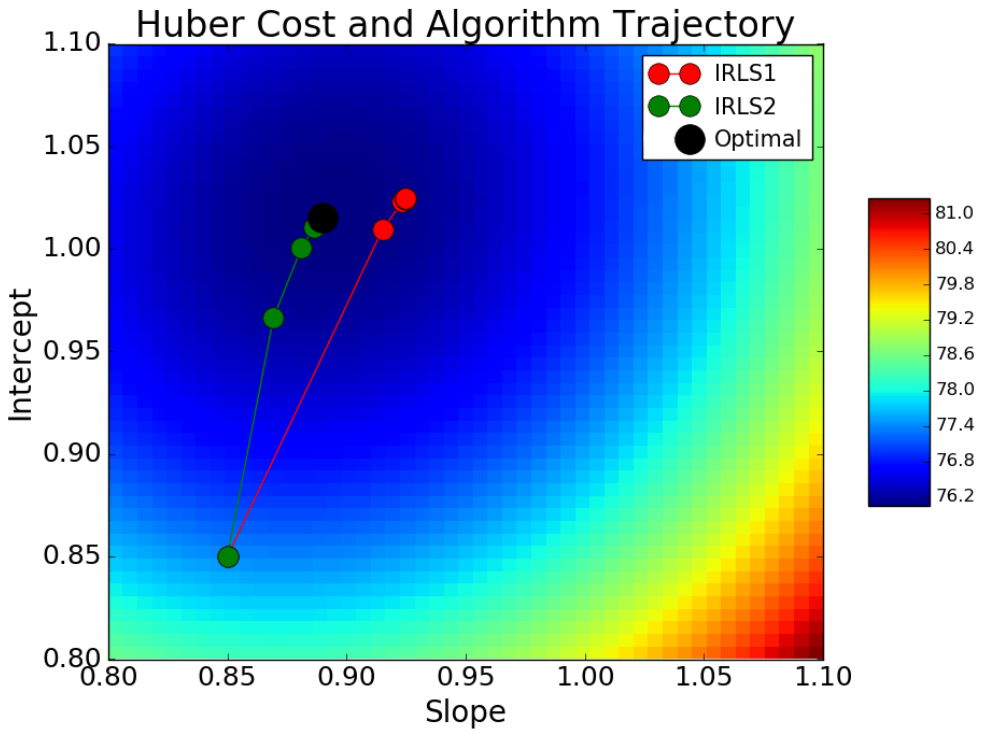
Consider the simple regression problem

$$(\hat{a},\hat{b}) = \text{argmin}_{(a,b)} \sum_i \rho(y_i - (ax_i + b)) \tag{2.206}$$

Data was simulated for this problem with 150 measurements and with $a = 1, b = 1$. The data x_i are known exactly and the measurements y_i have additive standard (unit variance) normal errors. Three different solutions were considered. The first is an iterative reweighted least squares using the weighting in Hartley and Zisserman [3]. The second is an iterative reweighted least squares using the weighting in Huber (which is similar to others and shown to give the correct necessary conditions in the previous section) [29]. The third method is a Sequential Quadratic Program (SQP) algorithm implemented in *SciPy* which will converge on the correction solution because the problem is convex.

The results are shown in Figure (2.13). The background color was generated by sweeping through the parameter space and is used to verify that the SQP solution is correct. The IRLS-2 algorithm using the weighting scheme presented in the previous section converged to the correct solution while the IRLS-1 algorithm using the weighting scheme in Hartley and Zisserman does not converge correctly.

Figure 2.13: Convergence of iterative reweighted least squares algorithm for two different weighting schemes. The IRLS-1 does not converge to the optimal solution while the IRLS-2 does.



2.5 Covariance of M-Estimates

Many applications require some type of uncertainty bound associated with an estimate. In many aerospace applications, it is common to assume that the estimate is approximately Gaussian with a given covariance. This section will briefly discuss methods to compute the expected covariance of an M-estimate.

One simple approach is to make a small modification to the classical least squares covariance. Consider the linear system

$$\boldsymbol{y} = H\boldsymbol{\theta} + \boldsymbol{v} \tag{2.207}$$

$$\boldsymbol{v} \sim \mathcal{N}(\boldsymbol{0}, R) \tag{2.208}$$

The estimate given by

$$\hat{\boldsymbol{\theta}} = \left(H^T W H\right)^{-1} H^T W \boldsymbol{y} \tag{2.209}$$

with an arbitrary weight matrix W is unbiased with covariance

$$\left(H^T W H\right)^{-1} H^T W R W H \left(H^T W H\right)^{-1} \tag{2.210}$$

The classical solution is to choose $W = R^{-1}$ to obtain the minimum variance, $\left(H^T R^{-1} H\right)^{-1}$ solution. As described in the previous section, M-estimation techniques above can be interpreted as a weighted least squares method where a diagonal W matrix is determined by the data. The weights computed during the M-estimation computation can be substituted into the diagonal elements of the covariance expression above. Furthermore, if $R = \sigma^2 I$ then the covariance expression reduces to

$$\sigma^2 \left(H^T W H\right)^{-1} H^T W^2 H \left(H^T W H\right)^{-1} \tag{2.211}$$

An alternative method that can more accurately capture the underlying (and possibly non-Gaussian) error distribution is to utilize the expressions derived above such as Equation (2.167). Clearly the expectations cannot be evaluated if the underlying distribution is unknown. To resolve this issue, the so-called *sandwich estimator* can be used which simply replaces all expectation operations with sums over the residual data [49]. While this method can give asymptotically correct covariances, it does not account for bias and can therefore underestimate the total estimation errors [50].

Huber gives several covariance expressions that utilize the *sandwich estimator*

philosophy [29, 36]. These are of the form

$$K\frac{N}{N-p}\frac{1/N\sum\psi(y_i-\mathrm{h}_i(\hat{\boldsymbol{\theta}}))^2}{\left(1/N\sum\psi(y_i-\mathrm{h}_i(\hat{\boldsymbol{\theta}}))^2\right)^2}(H^TH)^{-1} \tag{2.212}$$

where p is the number of parameters in $\boldsymbol{\theta}$. The term $N/(N-p)$ accounts for the fact the residuals under-represent the errors. Note that this term tends to unity for $N >> p$. The factor K can be used to obtain asymptotically correct results at the model when

$$K = 1 + \frac{p}{N}\frac{\mathrm{VAR}\left[\psi\right]}{\mathbb{E}\left\{\psi'\right\}^2} \tag{2.213}$$

where the expectation is evaluated at the nominal model.

2.5.1 Bootstrapping

Another technique for computing confidence bounds is bootstrapping [40]. This is a simple yet powerful concept and is an extension of the classical jackknife technique of Quenouille [51, 40].

Let $\hat{\boldsymbol{\theta}}$ be an estimate from N measurements and let $\hat{\boldsymbol{\theta}}_{(-i)}$ be an estimate with the i-th measurement removed (i.e. using $N-1$ measurements). Let $\boldsymbol{e}_i$ be the weighted discrepancy $N\hat{\boldsymbol{\theta}}+(N-1)\hat{\boldsymbol{\theta}}_{(-i)}$. Then the jackknife covariance can be computed as

$$\hat{P} = \frac{1}{N-1}\sum_{i=1}^{N}\left(\boldsymbol{e}_i - 1/N\sum_{j=1}^{N}\boldsymbol{e}_j\right)\left(\boldsymbol{e}_i - 1/N\sum_{j=1}^{N}\boldsymbol{e}_j\right)^T \tag{2.214}$$

The bootstrap technique is similar in that a covariance or confidence interval is constructed by comparing various estimates computed from different subsets of the data. Instead of systematically leaving one measurement out at a time, the bootstrap method randomly samples a subset of n measurements to compute a particular

estimate. This is repeated m times resulting in m different estimates $\hat{\boldsymbol{\theta}}_{(i,n)}$. The bootstrap covariance is then

$$\hat{P} = \frac{1}{m}\sum_{i=1}^{m}\left(\hat{\boldsymbol{\theta}}_{(i,n)} - 1/N\sum_{j=1}^{N}\hat{\boldsymbol{\theta}}_{(j,n)}\right)\left(\hat{\boldsymbol{\theta}}_{(i,n)} - 1/N\sum_{j=1}^{N}\hat{\boldsymbol{\theta}}_{(j,n)}\right)^{T} \tag{2.215}$$

Although both these methods require repeatedly obtaining M-estimates, they do give a good idea of the variability in the estimates subject to perturbations in the dataset. The computational burden for nonlinear measurements is alleviated by the fact that a given estimate can be used as a good starting guess for each M-estimate computation. Unfortunately, like other variance estimation methods, the bootstrap cannot account for systemic bias or correlated error across all measurements.

2.6 Selecting a Nominal Model

This section briefly discusses model selection for estimation problems. The model is the relationship between the measurements and the parameters of interest. This task is critical yet often overlooked. Model selection can be informed by various sources such as physical principles and experimental results. Considerations as to the number of available measurements, model complexity, and computational cost must be taken into account in choosing a model. A few common issues that are applicable to vision-aided navigation are discussed in this section.

The data available for estimation is typically partitioned into a set of individual measurements that are independent of each other when conditioned on all unknown model parameters. The model for each individual measurement of the set is typically

partitioned into an idealized part that deterministically relates parameters to *noise free* measurements and an additive probabilistic part that is independent of the parameters in the deterministic part. Nuisance parameters may appear in both the deterministic and probabilistic part of the model. These assumptions are convenient for practical M-estimation methods. However, issues can arise.

One common issue with the above assumptions is a lack of independence between the probabilistic portion of the model across measurements. Measurements that represent a time series can have positive serial correlations that are difficult to deal with. Solutions to this problem in the literature are very limited and require lots of insight into the application of interest. Typical approaches include using a weight matrix that captures the correlation (like in classical least squares), adding parameters for a bias that evolves in time, and statistical tests for the amount of correlation such as the Durbin-Watson test [52, 53, 54].

Another issue is the assumption of independence between the deterministic and probabilistic portion of the model. Systemic errors in the deterministic portion of the model may invalidate this assumption. Experimental testing can determine whether or not this is the case. If the assumption is invalid, attempts can be made to capture it with added nuisance parameters. This dissertation will investigate this problem in the context of vision-aided navigation.

Lastly, the issue of selecting the probabilistic model must be addressed. A core theme of this section is that the MLE is excellent if it is designed around

the true probabilistic model but can fail if the design model differs slightly from the truth. Robust estimation methods can be designed for minimax optimality in a neighborhood near a nominal model. An open question is how to choose this nominal model.

A clean theoretical approach is as follows. Generate a large amount of experimental measurements with perfect ground truth for the deterministic model parameters. For the measurement model $y_i = \mathrm{h}_i(\boldsymbol{\theta}) + \boldsymbol{\nu}_i$, this would involve ground truth for $\boldsymbol{\theta}$. This allows for samples of the noise $\boldsymbol{\nu}_i$ to be obtained. As the number of samples goes to infinity, the distribution of $\boldsymbol{\nu}_i$ can be exactly determined by the Glivenko-Cantelli theorem [40]. Then the MLE can be designed around the distribution.

In practice, this is not possible for several reasons. First, only a finite number of samples can be taken. If a nonparametric technique like a kernel density estimate is used to fit a distribution to the samples, there will inevitably appear to be zero probability in the regions of the sample space where no samples occurred even if there is a small chance of such a sample occurring [55, 56, 40]. The resulting MLE would incorrectly prohibit a residual from occupying this region which can seriously degrade the estimate. Second, experimental tests are unlikely to replicate the true operating conditions. This may lead to an imperfect density estimate. Third, such an approach cannot take into account all possible self-correlations in the data (across all different time scales). For these reasons and more, it appears that the better approach is to find a model that is *good enough* and then design the estimator to

be insensitive to modeling errors. One way to determine a neighborhood about a nominal model given experimental data is the Kolmogrov-Smirnov test.

2.6.1 Kolmogrov-Smirnov Test

The one-sided Kolmogrov-Smirnov test computes a p-value for the null hypothesis that a given empirical distribution was obtained from a particular distribution [57]. Given N samples ν_i and a nominal model F, the test statistic is

$$\tau = \max_\nu |F(\nu) - \sum_i 1_{\nu > \nu_i}(\nu)| \tag{2.216}$$

Given N and τ, the p-value for the null hypothesis can be found in [57]. Note that the ε-Kolomogrov neighborhood about F is the set of distributions G such that

$$\varepsilon > \max_\nu |F(\nu) - G(\nu)| \tag{2.217}$$

Therefore the p-value can be used to rigorously determine the size of the Kolmogrov neighborhood about the model F at a given confidence level.

3 PROBABILISTIC SENSOR MODELS

Estimation methods require probabalistic models that relate the unknown parameters to the inputs. This section discusses probabalistic models for sensors commonly used in navigation.

3.1 Feature Detection and Tracking

Feature tracking is an essential task in vision-aided navigation systems. Visual and geometric feature tracking enables certain estimation tasks by abstracting the raw measurements from visual cameras and range sensors into a more convenient form. Feature tracking has three primary subtasks. First, images of a scene must be processed to measure the pixel locations of imaged physical points through a process called *feature detection*. These physical points will be referred to as *landmarks*. Second, a *feature descriptor* must be computed and associated with each landmark. The purpose of the feature descriptor is to create a unique *signature* for each landmark. Third, *feature matching* is used to solve the correspondence problem between one set of feature descriptors (from an image or map) and a second set of feature descriptors (from an image). The solution to the correspondence problem is enabled by a *similarity measure* or *matching score* that can be computed between any two visual descriptors. The space of feature descriptors along with their matching score represents a metric space. In addition to the matching score, feature tracking can be

aided by *a priori* knowledge of sensor motion and scene structure.

At a high level, the process of tracking features works as follows. On the first image in a sequence, a *feature detection algorithm* is performed to identify the image-space locations of new features. The feature detector gives a *detection score* to each pixel. Pixels that are local extrema of this score and exceed a predefined threshold are labeled as features. Then a *feature description algorithm* is used to obtain descriptors for all the detected features. The descriptor is typically computed using all pixels in a neighborhood about the detected feature location. Subsequent images can be handled in one of two ways. The first way involves reperforming the detection and description steps and then using the descriptors to find the correspondence between the first image and current image. The second way, which is only valid for small motion between images, involves a search about the expected location of each feature. For each feature, the search is carried out by comparing the descriptor in the original image to all pixels within a pre-defined search window in the current image. In both approaches, candidate matches whose score exceeds a threshold are labeled as matches. The system designer can chose when to identify new landmarks. In the first approach above, any detected feature that has no valid correspondence can be labeled as a new landmark. The more common approach is to only instantiate new landmarks when the number of actively tracked landmarks drops below a predefined threshold.

A basic understanding of the underlying visual detection, description, and

matching algorithms is needed for the presentation of methods in later sections that address the following questions:

- How is a feature detection score related to localization error?
- Can a feature descriptor be used to predict the quality of a feature?
- Can changes in a feature descriptor be used to monitor changes in feature quality?
- Can the matching score be used to monitor feature quality?

The above questions have received some qualitative attention in the literature. One should certainly be suspicious of features with low matching and detection scores. However, these effects have not been rigorously quantified. Furthermore, knowledge of these effects has not been incorporated into vision-aided navigation systems beyond the basic dichotomizing of good and bad features based on a threshold score.

This section will discuss why feature tracking is necessary and how it works. An overview of specific detection, description, and matching algorithms is given. This discussion will motivate novel methods developed in later sections. The primary conclusions of this section are:

- Feature tracking produces a measurement abstraction that drastically reduces the dimensionality of the navigation problem.
- A visual and/or geometric feature is assumed to correspond to a fixed location in a static scene.
- Feature detection is the process of identifying the image space location of phys-

ical landmarks in the scene without *a priori* knowledge.

- Feature tracking is the process of computing the image space trajectory of a landmark across images separated by short time scales.
- Feature description is the process of associating a landmark with a compact representation of visual appearance.
- Feature matching is the process of computing correspondences between visual descriptors across multiple frames.
- Feature description and matching can perform the same task as feature tracking.
- The internal state of both feature trackers and descriptor-matcher pairs give important information about the quality of the output feature trajectory measurement.

3.1.1 Why Use Features?

The problem of parameter estimation using measurements requires a model of how the unknown parameters relate to the observed measurement. A depth image from a LIDAR (light detection and ranging) or TOF (time of flight) sensor depends on the sensor parameters, the pose of the sensor relative to the scene, and the **full geometry of the scene**. The scene geometry cannot be parametrized by a finite number of parameters in general. A visual image is significantly more complex. In addition to the scene geometry and pose, the image depends on the properties of the surface and the properties of all lighting sources. Attempting to estimate even

a discretized version of this parameter space is enormously expensive. Features are used to overcome these problems: they abstract the raw sensor measurements into a pixel location of a physical landmark.

The assumption of a set of M landmarks in a scene that can be located in images reduces the scene parameters from some infinite dimensional space into a finite dimensional vector. In most approaches to the SLAM problem, this vector contains the Euclidean coordinates of each landmark leading to a vector of dimension $3M$. Although other parameterizations exist, inverse-depth and parallax angle for example, the assumption of a fixed landmark position is ubiquitous. Later sections will discuss that assumption in much more detail.

For the purposes of estimation, the feature tracker can be viewed as a black-box whose outputs are the inputs to the estimator. The desired qualities of a good feature tracker are

- Detects a large number of landmarks (more measurements give better estimates)
- Feature tracks for a landmark have small errors that are independent between frames (lower measurement error gives better estimates)
- Feature tracks are persistent over many images (stronger observability gives better estimates)

In later sections of this dissertation, the internal representation of a feature is used to predict how well-suited a particular landmark is to the estimation process. The main

idea is that if landmark quality can be ascertained by the visual description alone, then intelligent decisions can be made as to which landmarks are incorporated into the estimator and which are discarded. Furthermore, the landmark quality information can be used to select estimator parameters (e.g. weights). The following sections contain a discussion of feature tracking methods. This background information is essential to the presentation of methods in later sections.

3.1.2 State of the Practice

As the literature on feature tracking algorithms is enormous, it is wise to focus attention on those that have been successfully applied to navigation problems. Surprisingly, despite the very large number of algorithms, a fairly small number have been used for navigation. The multi-scale Shi-Tomasi detector has been used in many successful SLAM systems [58, 59, 4]. Klein and Murray first used a multi-scale FAST detector to efficiently get feature candidates. They then discard candidates whose Harris score (very similar to the Shi-Tomasi score) falls below a predefined threshold. Leutenegger also took this approach [60]. Another, much newer, detector called CenSurE has been used in a number of SLAM system to detect new features [61, 62, 63].

For feature description, there has been an overwhelming trend to use the keypoint image patch itself as the descriptor. All of the above mentioned SLAM systems do this. However, the methods of matching the descriptor (i.e. image patch) to the current image differentiates the above algorithms. The image patch matching meth-

ods can be divided into two categories. The first category warps the representative patch into the current frame using the landmark position (associated with the patch) and a prior estimate of pose. The second category does not use any pose information to initialize the feature matching. While the first category has the potential for higher matching success, the second category will be more robust to temporary increases in pose error.

3.1.3 A Unified Approach

For pose estimation applications, feature tracking algorithms should be able to return metrics that can be related to tracking error. A large class of tracking algorithms are designed around the idea that pixel patches undergo a deformation between images. Therefore tracking error can be partitioned into two sources. The first source is a modeling error: the discrepancy between the true deformation and the assumed deformation. The second source is errors in the estimated deformation parameters due to per-pixel noise. The following analysis generalizes this class of tracking algorithms. Algorithms that fall outside of this class are often built around heuristics and are less ammenable to rigorous error analysis. Therefore they are not discussed here.

Consider an image patch at time t_1, $\mathrm{I}(\boldsymbol{x}, t_1)\ \forall \boldsymbol{x} \in \mathscr{W}$. Assume that under ideal circumstances, this image patch undergoes a deformation such that

$$\mathrm{I}(\mathrm{f}(\boldsymbol{x}, \boldsymbol{\theta}), t_2) = \mathrm{I}(\boldsymbol{x}, t_1) + \boldsymbol{\nu}(\boldsymbol{x}) \tag{3.1}$$

where $\boldsymbol{\nu}(\boldsymbol{x})$ represents noise in the two images which is not correlated in time or

space. In this equation, $\boldsymbol{\theta}$ parameterizes the deformation function f. Given some distribution for the noise, a cost function $\boldsymbol{\rho}(\cdot)$ can be selected such that optimizing

$$\boldsymbol{\varepsilon} = \sum_{\boldsymbol{x} \in \mathscr{W}} \left[\boldsymbol{\rho} \left(\mathrm{I}(\mathrm{f}(\boldsymbol{x}, \boldsymbol{\theta}), t_2) - \mathrm{I}(\boldsymbol{x}, t_1) \right) \right] \tag{3.2}$$

over the unknown parameters gives a Maximum Likelihood Estimate (MLE) of $\boldsymbol{\theta}$. The resulting stationarity condition is

$$\mathbf{0} = \sum_{\boldsymbol{x} \in \mathscr{W}} \left[S \left(\mathrm{I}(\mathrm{f}(\boldsymbol{x}, \boldsymbol{\theta}), t_2) - \mathrm{I}(\boldsymbol{x}, t_1) \right) B(\boldsymbol{x}) \boldsymbol{g}_2(\boldsymbol{x}) \right] \tag{3.3}$$

where $S(\cdot) = \frac{d\boldsymbol{\rho}(\cdot)}{d(\cdot)}$ is the score function, $B = \frac{\partial \mathrm{f}(\boldsymbol{x}, \boldsymbol{\theta})}{\partial \boldsymbol{\theta}}$, and $\boldsymbol{g}_2(\boldsymbol{x})$ is the gradient of the second image patch.

Such a framework enables the joint design of the feature detection, description, and matching process. The detector can locate patches, $\mathscr{W}$, such that the solution for the unknown parameters will be well-conditioned. The descriptor can be the image patch $\mathscr{W}$ itself. The matcher can give a metric on how well the resulting residual $\boldsymbol{\varepsilon}$ is consistent with the assumed noise distribution conditioned on the correct deformation model.

The most common feature detection and tracking algorithms can be derived by selecting a quadratic cost $\boldsymbol{\rho}(\cdot) = (\cdot)^2$ which is optimal for Gaussian noise. The two most commonly used deformation models are the affine model, $\mathrm{f}(\boldsymbol{x}) = D\boldsymbol{x} + \boldsymbol{d}$, and the translation model $\mathrm{f}(\boldsymbol{x}) = \boldsymbol{x} + \boldsymbol{d}$. First consider the translation model which leads to a score function

$$\mathbf{0} = \sum_{\boldsymbol{x} \in \mathscr{W}} \left[\left(\mathrm{I}(\mathrm{f}(\boldsymbol{x}, \boldsymbol{\theta}), t_2) - \mathrm{I}(\boldsymbol{x}, t_1) \right) \boldsymbol{g}_2(\boldsymbol{x}) \right] \tag{3.4}$$

Furthermore, let the deformed second image be approximated by a first-order expansion about an undeformed version: $\mathrm{I}(\mathrm{f}(\boldsymbol{x}, \boldsymbol{\theta}), t_2) \approx \mathrm{I}(\boldsymbol{x}, t_2) + \boldsymbol{g}_2(\boldsymbol{x})\boldsymbol{d}$. Substituting this into the the score function and solving for $\boldsymbol{d}$ yields

$$\boldsymbol{d} = A^{-1}\boldsymbol{e} \tag{3.5}$$

where

$$\boldsymbol{e} = \sum_{\mathscr{W}} \left[w(\boldsymbol{x}) \left(\mathrm{I}(\boldsymbol{x}, t_2) - \mathrm{I}(\boldsymbol{x}, t_1) \right) \boldsymbol{g}_2(\boldsymbol{x}) \right] \tag{3.6}$$

$$A = \sum_{\mathscr{W}} \left[w(\boldsymbol{x}) \boldsymbol{g}_2(\boldsymbol{x}) \boldsymbol{g}_2^T(\boldsymbol{x}) \right] \tag{3.7}$$

where $w(\boldsymbol{x})$ is an optional weighting function that can be set to unity over the window to obtain the true MLE. In practice, many methods use a kernel function such as a Gaussian to emphasize the center of the window. These equations can be used to justify a large number of feature detection and tracking algorithms.

3.1.4 Feature Detection

Feature detection is the first step in feature tracking. The input to this step is a visual image or range image or both and the output is a set of pixel locations of landmarks. The nearly universal approach in feature detection is to apply an *interest operator* to all pixel locations in the image and identify local extrema. The interest operator is a functional of the raw image, $f : \mathrm{I}(\boldsymbol{x}) \mapsto \mathrm{f}(x)$, which gives some measure of how well a feature can be localized. A feature is well localized if an estimate of location is insensitive to small perturbations (i.e. the addition of per-pixel noise). For example, an edge can only be localized well in the direction orthogonal to the edge

gradient: the location estimate along the edge is highly uncertain (an issue known as the *aperture problem*. On the other hand, a right-angle-corner is well localized in both directions.

These intuitive generalizations motivate all corner detection algorithms in the literature. The basic idea is that small translational perturbations to an image patch should give a very low similarity measure to the original patch. Early work by Beaudet identified the need for strong image gradients in all directions. Later work by authors like Harris, Stephens, Shi, Tomasi, and Kanade all followed this direction. Their algorithms can be unified in light of Equation (3.5). The matrix A should be positive definite with eigenvalues far from zero in order for the displacement to be well-defined. The Tomasi-Kanade detector look for maxima in $\mathsf{min}(\lambda_1, \lambda_2)$ where λ_i is the i'th eigenvalue of A [64]. The Harris corner detector computes $\mathsf{det}(A)^2 - \kappa\mathsf{trace}(A) = \lambda_1\lambda_2 - \kappa(\lambda_1 + \lambda_2)^2$ which is computationally more efficient that the Tomasi-Kanade detector [65].

Other approaches also attempt to find regions of the image with strong gradients and curvature. The FAST corner detector does this in a very coarse, yet highly-efficient manner by testing each pixel for a large enough contiguous set of pixels on a Bresenham circle that are sufficiently different in intensity than the center pixel [66].

Many other feature detectors exist in the literature. Surveys by Tuytelaars and Guo studies many in detail [67, 68]. Further discussion of particular algorithms

is beyond the scope of this work because many algorithms are just variants on the above concepts. The primary conclusion is that all detectors find local extrema of some metric over the image. Features are extrema that exceed some user-defined threshold. It is expected that the stronger the detection score the better the feature will be.

The one other issue that should be mentioned is the issue of scale. The above methods require the specification of a window size. Consider a certain landmark in the scene that appears as a sharp corner over a small pixel window, say 5×5, when viewed at a distance. When viewing this same landmark from a much closer viewpoint, a 5×5 pixel window may not be large enough to capture the corner structure. The landmark has appeared to change *scale* in this case. Certain feature detectors are specifically designed to be approximately scale invariant (usually under some assumptions of pixel deformation). Other feature detectors can be made scale invariant by constructing so-called *image pyramids*. An image pyramids is an ordered set of images such that each successive image is obtained from the previous image by blurring and subsampling it, starting from the original observed image. This has the effect of emulating views of the same region of the scene from various distance. Feature detectors can be applied to each level of the pyramid to detect features at the various scales. This will be relevant to methods discussed in later sections. Next, algorithms to track detected features are discussed.

3.1.5 Feature Tracking

Feature tracking is the process of computing a feature location trajectory across a set of images given the position in the first image. As mentioned above, there are two approaches to this: computing descriptors and matching across images or searching for feature matches in small prediction windows. The latter approach is more suited to vision-aided navigation tasks and will be discussed in detail here. The most common descriptor with this approach is the image patch itself.

Given a window in an image, centered at some location, the feature tracker must find a window in the subsequent image similar in appearance to the window in the original image. The displacements of this visual patch between images gives the featue trajectory. Visual tracking methods differ in their similarity metric, the allowable deformation between frames, and the method of solution for the assumed deformation. There are also variations in preprocessing steps like image processing filters and in how the methods chose the size of the window patch.

Early methods assume translational changes only between images and perform displacement estimation using frequency-domain techniques [69]. They take advantage of the convolution theorem which states that the convolution of two functions is equal to the inverse Fourier transform of their Fourier domain product. Therefore, by applying a Fourier transform to two images patches, taking the product, then taking the inverse Fourier transform, the resulting peak gives the displacement. The magnitude and sharpness of the peak is used to determine the quality of the resulting

correspondence. This has been recently applied to terrain relative navigation (TRN) for extra-terrestrial landing applications [70]. This has also been applied to solving the dense correspondence problem for stereo cameras [71].

A second set of methods operate in the spatial-domain. Within this set, some methods simply compute a cost for all displacements between the original and new image patch. Costs such as the sum of absolute differences (SAD) and the nomralizes cross-correlation have been proposed [72, 73]. Another set of methods that work in the spatial domain perform iterative least squares to find the optimal displacement. In fact, the well known Lucas-Kanade tracker uses the solution to Equation (3.2) in Equation (3.5) to find the displacement of an image patch from one image to the next [74]. Other methods have more complicated allowable transformations. For example, the method by Fuh involves the following cost function [75].

$$\varepsilon(M, \boldsymbol{d}, r, c) = \sum_{W} \left[((r\mathbf{J}(M\boldsymbol{x} + \boldsymbol{d}) + c - \mathbf{I}(\boldsymbol{x}))^2 \right] \tag{3.8}$$

where r and c represent a scaling and offset in pixel intensities respectively. The introduction of the general matrix M, along with the displacement d, allows for general affine deformations. Fuh presents a method to eliminate r and c from the stationarity conditions and in terms of M and $\boldsymbol{d}$. The remaining six-dimensional parameter space is then searched for an optimum.

One important problem is that that image patch can be appear to change in size between frames due to motion towards or away from an object. In this case, if using a window of fixed dimension in pixel space in two frames, the windows will

not cover the same portion of the scene. Manmatha gives an important early use of scale space techniques to deal with this issue [76]. The main idea is motivated by the fact that the convolution of an image with a Gaussian kernel followed by subsampling emulates gives an image patch that looks like it has been viewed from a greater distance. Iteratively performing the convolution and subsampling creates a pyramid of images that emulate views from many distances. Feature tracking can be performed at each level of the pyramid.

3.2 Camera Models

The key output of the feature tracker is the pixel location of a point landmark. A camera model is needed to relate this pixel location with the landmark location and camera pose. Let $\boldsymbol{r}_{f/n}$ be a vector from the origin of a scene-fixed reference frame, n^+, to a landmark. Let $\boldsymbol{r}_{c/n}$ be a vector from the origin of the same reference frame to a camera-fixed reference frame, c^+. Finally let $\boldsymbol{R}_{c/n}$ be the rotation matrix that transforms coordinates from the n^+ frame to the c^+ frame. *The camera frame* merits further discussion.

A common practice in the literature is define the camera frame such that the origin is at the optical center, the x-axis is along the pixel rows, the y-axis is along the pixel columns, and the z-axis completes the coordinate system (i.e. outward along the optical axis). While this may be suitable for simulation and analysis, it is poorly defined for practical applications. In particular, the user typically only has

knowledge of a reference frame fixed to the external camera housing. This reference frame is used to identify all mounting holes. However, even if the camera had an ideal unique optical sensor and orthogonal coordinate system tied to perfect pixel rows and columns, the user cannot immediately relate that to the frame attached to the external housing. This fact is further complicated by the fact that any real lens introduces distortion that invalidates the ideal pinhole model. The solution to both of these problems is *camera calibration*.

Camera calibration is the process of determining a function that maps from three dimensional points in the camera's housing frame (or any well-established frame rigidly attached to the camera, say b^+) to two dimensional pixel locations. Let $\boldsymbol{r}_{f/b}$ be the position of a landmark relative to b^+. The camera projection function $\mathrm{h}()$ is parameterized by $\boldsymbol{\theta}$ and determines the pixel location $\boldsymbol{u}$:

$$\boldsymbol{u} = \mathrm{h}(\left[\boldsymbol{r}_{f/b}\right]_b, \boldsymbol{\theta}) \tag{3.9}$$

The main idea in calibration is to image a known calibration target: $\left[\boldsymbol{r}_{f/n}\right]_n$ for many landmarks. Then an algorithm must be used to jointly estimate $\boldsymbol{\theta}$ and the unknown b^+-to-n^+ pose at each image (so that $\left[\boldsymbol{r}_{f/b}\right]_b$ can be computed from $\left[\boldsymbol{r}_{f/n}\right]_n$). If an additional laboratory metrology system can determine the camera housing frame to the calibration target frame at the time of each image, then the unknown body frame, b^+, to camera image frame, c^+, can be estimated as well. More explicitly, the function is

$$\boldsymbol{u} = \mathrm{h}\left(R_{c/b}\left(R_{b/n}\left(\left[\boldsymbol{r}_{f/n}\right]_n - \left[\boldsymbol{r}_{b/n}\right]_n\right) + \left[\boldsymbol{r}_{c/b}\right]_b\right), \boldsymbol{\theta}\right) \tag{3.10}$$

The parameters for $\boldsymbol{\theta}$, $R_{c/b}$, and $\left[\boldsymbol{r}_{c/b}\right]_b$ are treated as the constant *intrinsic parameters* and the parameters for $R_{b/n}$ and $\left[\boldsymbol{r}_{b/n}\right]_n$ (i.e. pose) are treated as time varying *extrinsic parameters*.

There are various models for the function $\mathbf{h}()$. Nearly all models rely on a combination of the ideal pinhole model plus a generic function approximation technique to account for discrepancies from the pinhole model. The particular choice depends on the given application. The main drivers of the decision are:

1. How much calibration data and of what quality? A proper balance between bias and variance will favor less complex models (fewer parameters) when less calibration data is available. Furthermore, the data must be of high quality: a large volume of data cannot overcome systemic errors in the data. For example, calibration targets with surveyed corners must have the target-frame locations of the corners measured to high accuracy.
2. Desired accuracy? If very high accuracy is needed (say $< 0.1\,pixels$) then this will necessitate a more complex model which will need more calibration data.
3. Computational complexity? For applications with time constraints (i.e. real-time navigation), the computational cost of performing the forward and inverse mapping (i.e. unit-vector in Euclidean coordinates to pixel space and vice-versa) must be considered.

An excellent and commonly used approach is given in [77]. Their main idea is to assume the existence of an ideal camera frame. First define the projection function

which maps from 3D Euclidean coordinates to *normalized image space*

$$\boldsymbol{u} \equiv \boldsymbol{\pi}([x\, y\, z]^T) \tag{3.11}$$

$$= [u\, v]^T \tag{3.12}$$

$$= \begin{bmatrix} x/z \\ y/z \end{bmatrix} \tag{3.13}$$

The distortion free pinhole projection equations are an affine transformation of the projection

$$\boldsymbol{u}_o = \mathrm{h}_o([x\, y\, z]^T) \tag{3.14}$$

$$= [u_o\, v_o]^T \tag{3.15}$$

$$= \begin{bmatrix} f_u u + u_c \\ f_v v + v_c \end{bmatrix} \tag{3.16}$$

$$\left[\boldsymbol{u}^T\ 1\right]^T = K\left[\boldsymbol{\pi}([x\, y\, z]^T);\ 1\right]^T \tag{3.17}$$

$$K = \begin{bmatrix} f_u & 0 & c_u \\ 0 & f_v & c_v \\ 0 & 0 & 1 \end{bmatrix} \tag{3.18}$$

where f_u, f_v, c_u, and c_v are parameters (i.e. focal length and pixel center). The authors then assume the addition of radial distortion, decentering distortion, and thin prism distortion which causes perturbations of $\boldsymbol{\delta}_r$, $\boldsymbol{\delta}_d$, and $\boldsymbol{\delta}_p$ respectively. These terms are motivated by physical considerations of real lenses and can be expressed

as

$$\begin{aligned}\boldsymbol{\delta}_r &= [\delta_{ru}, \delta_{rv}]^T \\ &= \begin{bmatrix} k_1 u(u^2+v^2) \\ k_2 v(u^2+v^2) \end{bmatrix}\end{aligned} \tag{3.19}$$

$$\begin{aligned}\boldsymbol{\delta}_d &= [\delta_{du}, \delta_{dv}]^T \\ &= \begin{bmatrix} p_1(3u^2+v^2)+2p_2uv \\ p_2(3v^2+u^2)+2p_1uv \end{bmatrix}\end{aligned} \tag{3.20}$$

$$\begin{aligned}\boldsymbol{\delta}_p &= [\delta_{pu}, \delta_{pv}]^T \\ &= \begin{bmatrix} s_1(u^2+v^2) \\ s_2(u^2+v^2) \end{bmatrix}\end{aligned} \tag{3.21}$$

The total distortion $\boldsymbol{\delta} = \boldsymbol{\delta}_r + \boldsymbol{\delta}_d + \boldsymbol{\delta}_p$ can be expressed as

$$\delta_u = (a_1+a_3)u^2 + a_4uv + a_1v^2 + a_5u(u^2+v^2) \tag{3.22}$$

$$\delta_v = (a_2+a_4)v^2 + a_3uv + a_2v^2 + a_5v(u^2+v^2) \tag{3.23}$$

Therefore there are 9 unique parameters $\boldsymbol{\theta} = [f_u, f_v, c_u, c_v, a_1, a_2, a_3, a_4, a_5]$. The final pixel location (i.e. row and column in the image) is then

$$\boldsymbol{u}' = \mathrm{h}(\pi([x\, y\, z]^T)) \tag{3.24}$$

$$= \begin{bmatrix} f_u(u+\delta_u(u,v))+u_c \\ f_v(v+\delta_v(u,v))+v_c \end{bmatrix} \tag{3.25}$$

This nested model will simplify the design of estimators which require Jacobians of measurements with respect to parameters.

Given a measured $\boldsymbol{u}'$, the pinhole equations can be used to obtain an approxi-

mation to the normalized pixel value $\boldsymbol{u}$

$$u \approx \hat{u} \equiv (u' - c_u)/f_u \tag{3.26}$$

$$v \approx \hat{v} \equiv (v' - c_v)/f_v \tag{3.27}$$

This $\hat{u}$ and $\hat{v}$ can be substituted into the distortion equations to get an approximation to the distortion $\hat{\boldsymbol{\delta}} \equiv \boldsymbol{\delta}(\hat{u}, \hat{v}$. This approximation is sufficient for most applications because the distortion is typically on the order of 1% of the image size and varies slowly across the image. If more accuracy is needed, this estimate can be used as a starting guess in a nonlinear solver to more accurately determine the normalized pixel location. Other forms of $\boldsymbol{\theta}$ are possible.

The key conclusion of this section is that cameras must be calibrated in order to relate pixel measurements to a Euclidean reference frame. This calibration uses noisy measurement data and therefore leads to imperfect calibrations. Systemic errors in camera models add additional error to the mapping between image space and reference frames of interest. These errors should be accounted for in estimators that use pixel locations as a measurement input type.

It should also be noted that other parameterizations of pose can be used. The form presented above is convenient for navigation applications and is commonly used. Other parameterizations can be easily converted into the presented one and vice versa.

3.2.1 Error Propagation

It is important to study how error propagates through these equations. In particular, we want to obtain first-order equations of the form *Measurement = Function of Estimated Parameters + Linear Function of Parameter Errors + Random Noise.* This will be done for all measurements that are inputs to the filter (including probabilistic system dynamics which can be treated as a measurement). The resulting equations are used directly in the estimation

The equations are easily derived using a first-order Taylor series expansion about the parameters. This dissertation will follow the general notation of Crassidis and Junkins in that the true value will be represented by an unadorned symbol, say x, its estimated value will have a *hat*, $\hat{x}$, and the difference between the two will be $deltax = \hat{x} - x$ [17]. Measured values will be adorned with a *tilde*.

Before deriving the equations another note from Crassidis and Junkins is useful [17]. A rotation matrix $R_{b/a}$ has three degrees-of-freedom and can take on many parameterizations. The discrepancy between the true and estimated rotation matrix can be expressed as $R_{b/a} = \delta R_{b/a} \hat{R}_{b/a}$ where $\delta R_{b/a}$ is a small rotation. The parameterization of $\hat{R}_{b/a}$ can be different from the parameterization of $\delta R_{b/a}$. The parameterization of the latter should be chosen so that the change in the matrix elements with respect to the parameters in a neighbourhood about $\delta R_{b/a} = I$ is as smooth as possible. A small Euler angle rotation $\delta \boldsymbol{\theta}_{b/a}$ or a three element differential quaternion both meet this criteria and are equivalent up to a scale factor of 2. In

particular $\delta R_{b/a} \approx I - [(\delta\boldsymbol{\theta}_{b/a})^{\times}]$ up to first order where $[\boldsymbol{a}^{\times}]$ is the standard cross product matrix formed from a three element vector. The equations below will use this parameterization but it should be noted that the differential quaternion can be easily substituted.

The Taylor series expansion of Equation (3.25) using Equation (3.10), and Equation (3.13) yields

$$\begin{aligned}\tilde{\boldsymbol{u}}' = \hat{\boldsymbol{u}}' &+ J_{\boldsymbol{u}',\delta\boldsymbol{\theta}_{c/b}}\delta\boldsymbol{\theta}_{c/b} + J_{\boldsymbol{u}',\delta\boldsymbol{\theta}_{b/n}}\delta\boldsymbol{\theta}_{b/n} \\ &+ J_{\boldsymbol{u}',\delta\boldsymbol{r}_{c/b}}\delta\boldsymbol{r}_{c/b} + J_{\boldsymbol{u}',\delta\boldsymbol{r}_{b/n}}\delta\boldsymbol{r}_{b/n} \\ &+ J_{\boldsymbol{u}',\delta\boldsymbol{r}_{f/n}}\delta\boldsymbol{r}_{f/n} \\ &+ J_{\boldsymbol{u}',\delta\boldsymbol{\theta}}\delta\boldsymbol{\theta} + \boldsymbol{\nu}\end{aligned} \tag{3.28}$$

where $\boldsymbol{\nu}$ is random noise. The notation $J_{a,b}$ represents the Jacobian of a with respect to b. The nested model derived above will simplify the computation of these Jacobians. In particular, for $x = \delta\boldsymbol{\theta}_{c/b}$, or $x = \delta\boldsymbol{\theta}_{c/b}$, or $x = \delta\boldsymbol{\theta}_{c/b}$, or $x = \delta\boldsymbol{\theta}_{c/b}$, or $x = \delta\boldsymbol{\theta}_{c/b}$:

$$J_{\boldsymbol{u}',\boldsymbol{x}} = J_{\mathrm{h}(\pi),\pi} J_{\pi([\boldsymbol{r}_{f/n}]_n),[\boldsymbol{r}_{f/n}]_n} J_{[\boldsymbol{r}_{f/n}]_n,x} \tag{3.29}$$

The matrix $J_{h(\pi),\pi}$ depends on the camera distortion model used and will not be given in this section. The matrix $J_{\pi([\boldsymbol{r}_{f/n}]_n),[\boldsymbol{r}_{f/n}]_n}$ is common to all camera models and is

$$J_{\pi([\boldsymbol{r}_{f/n}]_n),[\boldsymbol{r}_{f/n}]_n} = \begin{bmatrix} 1/z & 0 & -x/z^2 \\ 0 & 1/z & -y/z^2 \end{bmatrix} \tag{3.30}$$

where $\left[\boldsymbol{r}_{f/n}\right]_n = [x\, y\, z]^T$. Using Equation (3.10), the other matrices are

$$J_{\left[\boldsymbol{r}_{f/n}\right]_n, \delta\boldsymbol{\theta}_{c/b}} = \left[\left(R_{c/b}\left(R_{b/n}\left(\left[\boldsymbol{r}_{f/n}\right]_n - \left[\boldsymbol{r}_{b/n}\right]_n\right) + \left[\boldsymbol{r}_{c/b}\right]_b\right)\right)^X\right] \tag{3.31}$$

$$J_{\left[\boldsymbol{r}_{f/n}\right]_n, \delta\boldsymbol{\theta}_{b/n}} = R_{c/b}\left[\left(R_{b/n}\left(\left[\boldsymbol{r}_{f/n}\right]_n - \left[\boldsymbol{r}_{b/n}\right]_n\right)\right)^X\right] \tag{3.32}$$

$$J_{\left[\boldsymbol{r}_{f/n}\right]_n, \delta\boldsymbol{r}_{f/n}} = R_{c/b}R_{b/n} \tag{3.33}$$

$$J_{\left[\boldsymbol{r}_{f/n}\right]_n, \delta\boldsymbol{r}_{b/n}} = -R_{c/b}R_{b/n} \tag{3.34}$$

$$J_{\left[\boldsymbol{r}_{f/n}\right]_n, \delta\boldsymbol{r}_{c/b}} = R_{c/b} \tag{3.35}$$

In addition, the Jacobian of the measurement with respect to the calibration parameters $\boldsymbol{\theta}$ is specific to the distortion model and will not be given here.

3.3 Range Cameras

The term *range camera* is used for any device that can measure the range from the device to a physical surface at *many* different bearing angles in a *short* time span. The extent of *many* and *short* in this definition are admittedly imprecise. Devices like a laser range finder that only measure range along a single axis certainly do not meet the definition while a device like Microsoft Kinect sensor which outputs range on a 480 $\times$ 640 grid certainly do. The variety of such devices prevents clear boundaries to be drawn.

For the purposes of analysis, much of the same equations that apply to visual cameras in the previous section apply to range cameras with some minor modification. In particular, there is a parameter vector $\boldsymbol{\theta}$ and an unknown body-to-sensor pose that is used to map from a direction in sensor coordinates to a direction and

offset in body coordinates, and vice verse. The direction in sensor coordinates can be scaled by the measured range to obtain a 3D vector in sensor coordinates.

The parameter vector $\boldsymbol{\theta}$ may also contain parameters for computing the range from lower level measurements. For example, time-of-flight range cameras will typically need to calibrate clock biases and signal delays.

The nested structure for the visual camera model can be applied here as well with the exception that no projection function is needed. Therefore all the Jacobians derived in the previous section apply if the 2×3 projection Jacobian, $\boldsymbol{J}_{\boldsymbol{\pi}(\left[\boldsymbol{r}_{f/n}\right]_n),\left[\boldsymbol{r}_{f/n}\right]_n}$ of Equation (3.30) is simply replaced with the 3×3 identity matrix.

3.3.1 Dense Versus Sparse

Before concluding the discussion on range sensors, it should be noted that there are several distinct ways to incorporate range data into a navigation filter. These are

1. Auxiliary to Visual Camera: Visual features are detected and tracked in the visual camera. The range camera is used only to obtain the range to each feature.
2. Feature Tracking Independent of Visual Tracking: Point cloud features analogous to visual features are detected and tracked in the range image independently of the visual camera. There is an extensive literature on this: [78, 79, 80, 81]. This adds additional landmark parameters to the problem but can be a more robust solution from a systems engineering prospective since the

the failure modes can be separated.

3. Surface Alignment Odometry: This approach aligns the point cloud from a range image with a previous range image to determine changes in pose over time. Iterative closest point (ICP) and its variants are at the core of this approach [82, 83, 84].
4. Absolute Surface Alignment: This approach attempts to build a dense 3D map of the scene. Range images are aligned to the map surface and used to update the map estimate. Kinect Fusion and its extension by Conway are examples of this approach [85, 86, 87, 88].

The best approach varies depending on the application. The dense mapping approach is excellent for small workspaces and systems with sufficient computational power. The dense odometry approach is easy to implement and computationally efficient but cannot be used to accurately propagate pose over extended periods of time. Tracking features in range images independently is only feasible if the range image is sufficiently dense For range cameras with a low density of range measurements, the first approach is suitable. The first approach has the additional advantage that the scale and observability issues associated with visual-only features do not arise. This will be the approach used in experiments with range cameras presented later in the dissertation.

3.4 IMUs

An inertial measurement unit (IMU) is a device that measures specific acceleration and angular velocity which is used to propagate vehicle motion forward in time. Specific acceleration is simply the acceleration relative to an inertial reference frame minus gravity. Although these quantities can be estimated from a model of the forces and torques on a body, an IMU can typically measure the quantities more accurately and reliably. Therefore the added cost (financial, mass, volume, power) of an IMU is justified by the increased flexibility and reduced computational complexity of force and torque modeling.

The IMU measurements are

$$\tilde{\boldsymbol{a}}_i = R_i\left(\boldsymbol{a}_i - \boldsymbol{g}\right) + \boldsymbol{\alpha}_i + \boldsymbol{s}_i \tag{3.36}$$

$$\tilde{\boldsymbol{\omega}}_i = \boldsymbol{\omega}_i + \boldsymbol{\beta}_i + \boldsymbol{\nu}_i \tag{3.37}$$

where $\boldsymbol{s}_i$ and $\boldsymbol{\nu}_i$ are white-noise processes with covariance $\sigma_s^2 I$ and $\sigma_\nu^2 I$ respectively. The biases $\boldsymbol{\alpha}_i$ and $\boldsymbol{\beta}_i$ are random walk processes driven by white-noise with covariance $\sigma_z^2 I$ and $\sigma_w^2 I$ respectively:

$$\boldsymbol{\alpha}_i = \boldsymbol{\alpha}_{i-1} + \boldsymbol{z}_i \tag{3.38}$$

$$\boldsymbol{\beta}_i = \boldsymbol{\beta}_{i-1} + \boldsymbol{w}_i \tag{3.39}$$

The discrete-form of the zero-jerk kinematic model for position and velocity is

$$\boldsymbol{r}_i = \boldsymbol{r}_{i-1} + \delta t \boldsymbol{v}_{i-1} + \frac{1}{2}\delta t^2 \left(R_i^T\left(\tilde{\boldsymbol{a}}_i - \boldsymbol{\alpha}_i - \boldsymbol{s}_i\right) + \boldsymbol{g}\right) \tag{3.40}$$

$$\boldsymbol{v}_i = \boldsymbol{v}_{i-1} + \delta t \left(R_i^T\left(\tilde{\boldsymbol{a}}_i - \boldsymbol{\alpha}_i - \boldsymbol{s}_i\right) + \boldsymbol{g}\right) \tag{3.41}$$

where the subscripts represent time index. The vector $\boldsymbol{r} = \left[\boldsymbol{r}_{m/n}\right]_n$ is inertial position and $\boldsymbol{v} = \left[\boldsymbol{r}_{m/n}\right]_n$ is inertial velocity (time derivative of inertial position). The rotational dynamics are more difficult to deal with as the rotational parameter propagation is inherently nonlinear. For the purposes of propagating the estimated quaternion using the gyro data, the discrete-form of the quaternion update equations can be used (see [17]).

$$\boldsymbol{q}_{i+1} = \Omega(\tilde{\boldsymbol{\omega}}_i - \boldsymbol{\beta}_i - \boldsymbol{\nu}_i)\boldsymbol{q}_i \tag{3.42}$$

Note that the translational model is second-order (acceleration-level coordinates are measured) while the rotational model is first-order (velocity-level coordinates are measured). In practice this is often sufficient because commonly used IMUs output data at a high rate $\geq 200\,\text{Hz}$ which is much larger than the rate of visual information which is typically between $10\,\text{Hz}$ and $60\,\text{Hz}$.

IMU Parameters

Before moving on to the error model, one more issue with the IMU model should be clarified. Accelerometer and gyroscope manufacturers typicaly specify the so called *bias stability* and *noise density* of sensors. Sometimes the noise density may be replaced by the *angle random walk* for gyroscopes. These quantities are not equivalent to the white-noise and random walk variances in the above model. In this section, the parameters of the above model are related to the quantities specified by the manufacturer. Only gyroscopes are discussed in this section but the results for accelerometers are directly analagous.

Consider a single-axis gyroscope at rest. The model is

$$\beta_i = \beta_{i-1} + w_i \tag{3.43}$$

$$= \beta_0 + \sum_{j=1}^{i} w_j \tag{3.44}$$

$$\tilde{\omega}_i = \beta_i + \nu_i \tag{3.45}$$

$$= \beta_0 + \sum_{j=1}^{i} w_j + \nu_i \tag{3.46}$$

The bias stability is defined as the minimum of the Allan-deviation, $\mathrm{AD}(N)$. The Allan-deviation is computed as follows. First, a large data set of $\bar{N}$ measurements is needed with the sensor at rest. For a given bin size N, the Allan-deviation is

$$\mu_i^{(N)} \equiv \frac{1}{N} \sum_{j=N(i-1)+1}^{Ni} \tilde{\omega}_j \tag{3.47}$$

$$\mathrm{AVAR}(N) \equiv \frac{1}{2(M-1)} \sum_{i=1}^{M-1} \left[\left(\mu_{i+1}^{(N)} - \mu_i^{(N)} \right)^2 \right] \tag{3.48}$$

$$\mathrm{AD}(N) \equiv \sqrt{\mathrm{AVAR}(N)} \tag{3.49}$$

where $M = \mathrm{floor}(\bar{N}/N)$ is the number of bins. To obtain the expected value for $\mathrm{AVAR}(N)$ in terms of the IMU parameters, the model equations are substituted into

Equation (3.49) and the expectation is taken as follows.

$$\begin{aligned}
\mu_i^{(N)} &= \frac{1}{N}\sum_{j=N(i-1)+1}^{Ni}\left[\beta_0 + \sum_{k=1}^{i} w_k + \nu_j\right] \\
&= \beta_0 + \frac{1}{N}\sum_{j=N(i-1)+1}^{Ni}\left[c_{1,N,j}w_j + \nu_j\right] \\
\mu_{i+1}^{(N)} &= \beta_0 + \frac{1}{N}\sum_{j=N(i-1)+1}^{Ni} w_j + \frac{1}{N}\sum_{j=Ni}^{N(i+1)}\left[c_{1,N,j}w_j + \nu_j\right] \\
\mu_{i+1}^{(N)} - \mu_i^{(N)} &= \frac{1}{N}\sum_{j=Ni}^{N(i+1)}\left(\nu_j - \nu_{j-N}\right) + \frac{1}{N}\sum_{j=Ni}^{N(i+1)}\left(w_j - (c_{1,N,j} - N)w_{j-N}\right) \\
\mathbb{E}\{\mathrm{AVAR}(N)\} &= \frac{1}{2(M-1)}\sum_{i=1}^{N-1}\mathbb{E}\left\{\left(\mu_{i+1}^{(N)} - \mu_i^{(N)}\right)^2\right\} \\
&= \frac{1}{2N^2}\left[\sigma_w^2\sum_{i=1}^{N}\left(c_{1,N,i}^2 + (N - c_{1,N,i})^2\right) + 2N\sigma_\nu^2\right] \\
&= \frac{1}{N}\sigma_\nu^2 + \left(\frac{2N}{3} + \frac{1}{3N}\right)\sigma_w^2
\end{aligned} \tag{3.50}$$

The location for the minimum can be found by taking the derivative of Equation (3.50) with respect to N, setting the expression to zero, and solving for N.

$$N_{\min} = \sqrt{\frac{3\sigma_\nu^2}{2\sigma_w^2} + \frac{1}{2}} \tag{3.51}$$

Substituting Equation (3.51) into Equation (3.50) gives the value of the bias stability in terms of the model parameters (an ugly but easy to compute expression). In practice, $\sigma_\nu^2 >> \sigma_w^2$. Therefore a reasonable approximation is

$$\mathrm{AVAR}(N) \approx \frac{1}{N}\sigma_\nu^2 + \frac{2N}{3}\sigma_w^2 \tag{3.52}$$

In this case, the minimum Allan variance is located at $N_{\min} = \frac{3\sigma_\nu^2}{2\sigma_w^2}$. At this location, the Allan variance is

$$\mathrm{minAVAR} \approx \frac{5}{2}\sigma_w^2 \tag{3.53}$$

One last point should be made about the about derivations. It was assumed that the Allan variance was obtained from IMU samples at the same sampling interval as the IMU model discretization time interval. If this is not true, then the following correction can be applied where Δt_{model} and Δt_{AV} are the sampling times for the model and Allan variance data respectively.

$$\sigma_w^2 = \frac{2\Delta t_{model}}{5\Delta t_{AV}} \,(\text{minAVAR}) \tag{3.54}$$

The angle random walk (ARW) is a common parameter found on gyroscope datasheets (in place of σ_v^2) and can be defined as follows. If the bias-compensated gyro signal is integrated over a short time interval, then ARW is the square root of the expected angle error variance. Let $\Delta\theta$ be the angle error accumulated over a period T. The interval should be short enough so that the bias is nearly constant over the interval. In this case, the error over an interval of length T with sampling time Δt nd N measurements (so that $T = N\delta t$) will be

$$\Delta\theta \approx \sum_i^N \nu_i \delta t \tag{3.55}$$

$$\text{ARW} \equiv \sqrt{\frac{\mathbb{E}\left\{\Delta\theta^2\right\}}{T}} \tag{3.56}$$

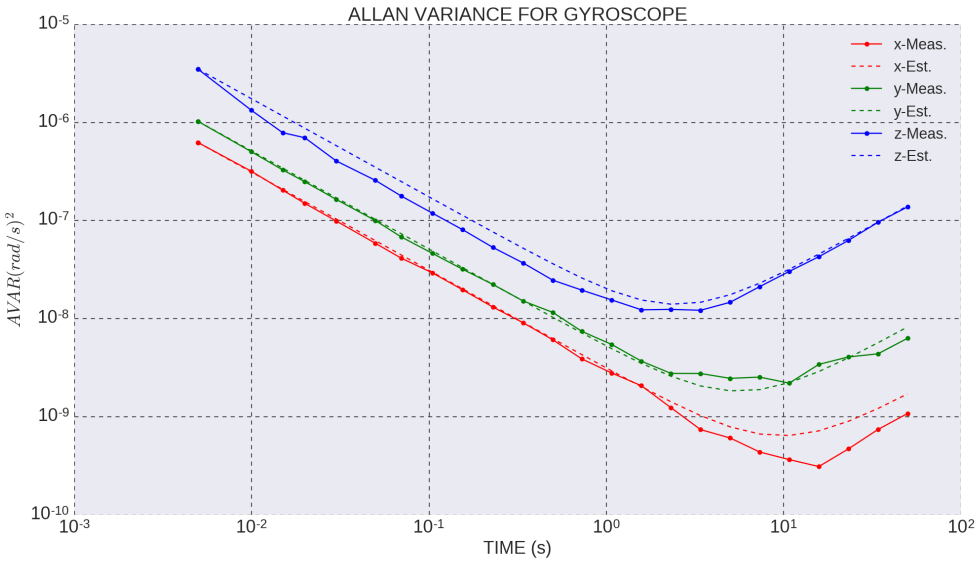
$$= \sqrt{\frac{\Delta t^2 \mathbb{E}\left\{\sum_{i=1}^N \nu_i \sum_{j=1}^N \nu_j\right\}}{T}} \tag{3.57}$$

$$= \sqrt{\frac{N\sigma_v^2 \Delta t^2}{T}} \tag{3.58}$$

$$= \sqrt{\frac{T\Delta t\sigma_v^2}{T}} \tag{3.59}$$

$$= \sigma_v \sqrt{\Delta t} \tag{3.60}$$

Figure 3.1: Allan Variance for the VN100 gyros

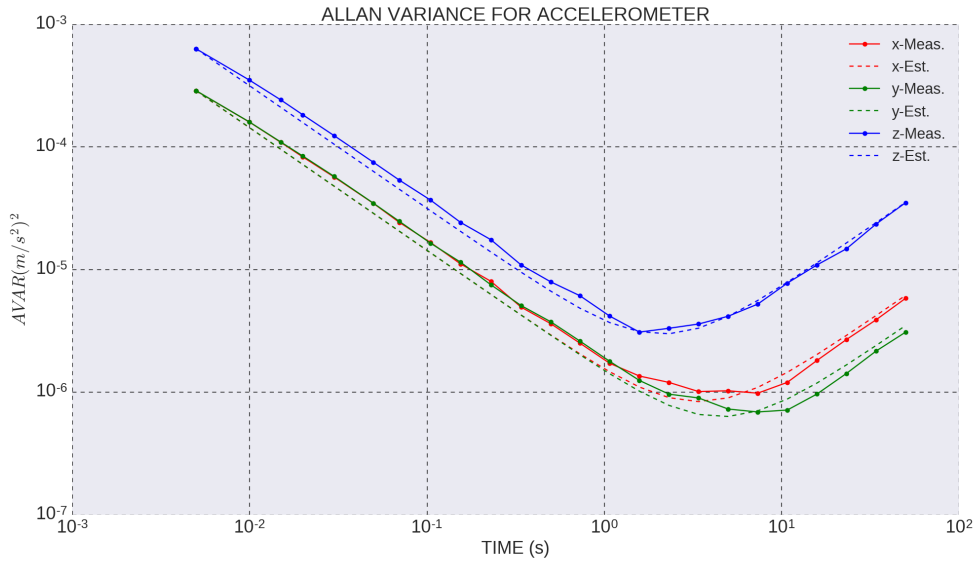


Therefore the σ_v used in the estimator and simulation models can be easily obtained from the ARW and sampling time reported on the datasheet.

Consider an Allan variance plot with log-log scales (base ten) with $\log(N)$ on the x-axis. When N is small such that $\frac{1}{N}\sigma_v^2 >> \frac{2N}{3}\sigma_w^2$, the y-axis will be approximately $\log(\sigma_v^2) - \log(N)$. Therefore the ARW can be determined by reading off the value at $N = 1$ on this plot.

An example for the *VectorNav VN100* 6-axis IMU is given below. The results for the gyros are given in Figure (3.1) and for the accelerometers in Figure (3.2). The results for the parameters are aggregated into Table (3.1) and Table (3.2) .

Figure 3.2: Allan Variance for the VN100 accelerometers



3.4.1 IMU Propagation Errors

In order to use IMU data in the weighted least-squares MLE solution, it is necessary to have an equation for the error in propagating the state between the time of two images. Let $\hat{\boldsymbol{x}}_0$ represent the state estimate at the time of one image and $\hat{\boldsymbol{x}}_N$ represent the state estimate at the time of a second image after N IMU readings, each separated in time by δt. Then $\boldsymbol{e} = \mathrm{f}(\hat{\boldsymbol{x}}_0) - \hat{\boldsymbol{x}}_N$ gives the error when f() is the function that uses IMU data to propagate the system. The expected value of $\boldsymbol{e}$ is zero. To derive the MLE solution, two equations are needed. The first is the covariance of $\boldsymbol{e}$ conditioned on zero initial-condition errors (i.e. $\hat{\boldsymbol{x}}_0 = \boldsymbol{x}_0$). The second is the Jacobian of $\boldsymbol{e}$ with respect to $\hat{\boldsymbol{x}}_0$. Note that when using this form of the error, the Jacobian with respect to $\hat{\boldsymbol{x}}_N$ is trivially the negative of the identity matrix.

Axis	BS (rad/s)	ARW ($rad/s^{3/2}$)	σ_v^2 ($(rad/s)^2$)	σ_w^2 (rad^2/s^3)
x	1.75e-05	0.011	1.23e-10	6.20e-07
y	4.68e-05	0.014	8.76e-10	1.02e-06
z	1.10e-4	0.0263	4.84e-09	3.47e-06

Table 3.1: VN100 gyroscope parameters from test data at 200 Hz.

Axis	BS (m/s)	ARW ($m/s^{3/2}$)	σ_v^2 ($(m/s)^2$)	σ_w^2 (m^2/s^3)
x	0.000989	0.239	3.91e-07	0.000286
y	0.000829	0.239	2.75e-07	0.000286
z	0.001755	0.354	1.23e-06	0.000629

Table 3.2: VN100 accelerometer parameters from test data at 200 Hz.

The derivation of these two equations are given below. The derivation proceeds by subtracting the estimated model (i.e. noise-free with estimated initial quantities) from the true model. This is very straight-forward for the biases:

$$\delta\boldsymbol{\alpha}_i \equiv \hat{\boldsymbol{\alpha}}_i - \boldsymbol{\alpha}_i = \delta\boldsymbol{\alpha}_{i-1} - \boldsymbol{z}_i \tag{3.61}$$

$$\delta\boldsymbol{\beta}_i \equiv \hat{\boldsymbol{\beta}}_i - \boldsymbol{\beta}_i = \delta\boldsymbol{\beta}_{i-1} - \boldsymbol{w}_i \tag{3.62}$$

For the purposes of error analysis, the three-element differential Euler angles $\boldsymbol{\theta}$ is used to represent the rotation between times $i-1$ and i.

$$R_i = \left(I - \left[(\boldsymbol{\theta}_i)^{\times}\right]\right) R_{i-1} \tag{3.63}$$

The true value of these angles are related to the angular rate measurement through

$$\boldsymbol{\theta}_i = \delta t(\tilde{\boldsymbol{\omega}}_i - \boldsymbol{\beta}_i - \boldsymbol{\nu}_i) \tag{3.64}$$

Of course, the angular rate used to propagate the estimates does not know the true bias or noise:

$$\hat{\boldsymbol{\theta}}_i = \delta t(\tilde{\boldsymbol{\omega}}_i - \hat{\boldsymbol{\beta}}_i) \tag{3.65}$$

which also assumes that the angular rate is constant. The rotational error is therefore

$$\hat{R}_i \equiv \left(I - \left[(\delta\boldsymbol{\theta}_i)^\times\right]\right) R_i \tag{3.66}$$

$$\delta\boldsymbol{\theta}_i \equiv \hat{\boldsymbol{\theta}}_i - \boldsymbol{\theta}_i \tag{3.67}$$

$$= \delta t\left(\delta\boldsymbol{\beta}_i + \boldsymbol{\nu}_i\right) \tag{3.68}$$

$$\delta\boldsymbol{q}_{i+1} = \delta\boldsymbol{q}_i + \frac{1}{2}\delta\boldsymbol{\theta}_i \tag{3.69}$$

$$= \delta\boldsymbol{q}_i + \frac{1}{2}\delta t\left(\delta\boldsymbol{\beta}_i + \boldsymbol{\nu}_i\right) \tag{3.70}$$

The last line in the above equation is due to the fact that the three-element differential $\delta\boldsymbol{q}$ quaternion is related to $\delta\boldsymbol{\theta}$ through $\delta\boldsymbol{q} = \frac{1}{2}\delta\boldsymbol{\theta}$. Also note that the first line is not used to propagate the rotation matrix: it is only used for error analysis. Before giving the position and velocity errors, the acceleration error must be analyzed.

$$\delta\boldsymbol{a}_i \equiv \hat{\boldsymbol{a}}_i - \boldsymbol{a}_i \tag{3.71}$$

$$= \hat{R}_i^T\left(\tilde{\boldsymbol{a}}_i - \hat{\boldsymbol{\alpha}}_i\right) - R_i^T\left(\tilde{\boldsymbol{a}}_i - \boldsymbol{\alpha}_i - \boldsymbol{s}_i\right) \tag{3.72}$$

$$= \hat{R}_i^T\left(\tilde{\boldsymbol{a}}_i - \hat{\boldsymbol{\alpha}}_i\right) - \hat{R}_i^T\left(I - \left[(\delta\boldsymbol{\theta}_i)^\times\right]\right)\left(\tilde{\boldsymbol{a}}_i - \boldsymbol{\alpha}_i - \boldsymbol{s}_i\right) \tag{3.73}$$

$$\approx \hat{R}_i^T\left(\boldsymbol{s}_i - \delta\boldsymbol{\alpha}_i - \left[(\tilde{\boldsymbol{a}}_i)^\times\right]\delta\boldsymbol{\theta}_i\right) \tag{3.74}$$

$$= \hat{R}_i^T\left(\boldsymbol{s}_i - \delta\boldsymbol{\alpha}_i - 2\left[(\tilde{\boldsymbol{a}}_i)^\times\right]\delta\boldsymbol{q}_i\right) \tag{3.75}$$

The position and velocity errors are then easily expressed in terms of the acceleration error assuming a zero-jerk model.

$$\delta \boldsymbol{r}_i = \delta \boldsymbol{r}_{i-1} + \delta t \delta \boldsymbol{v}_{i-1} + \frac{1}{2}\, \delta t^2 \delta \boldsymbol{a}_i \tag{3.76}$$

$$\delta \boldsymbol{v}_i = \delta \boldsymbol{v}_{i-1} + \delta t \delta \boldsymbol{a}_i \tag{3.77}$$

Because IMUs normally have a much higher data-rate than visual sensors like cameras, it is necessary to get expressions for the state errors due to many individual IMU readings, say N of them. This can be done as follows using $i = 0$ for the initial time index.

$$\delta \boldsymbol{\alpha}_N = \delta \boldsymbol{\alpha}_0 - \sum_{i=1}^{N} \boldsymbol{z}_i \tag{3.78}$$

$$\delta \boldsymbol{\beta}_N = \delta \boldsymbol{\beta}_0 - \sum_{i=1}^{N} \boldsymbol{w}_i \tag{3.79}$$

$$\delta \boldsymbol{q}_N = \delta \boldsymbol{q}_0 + \frac{1}{2} \sum_{i=1}^{N} \delta \boldsymbol{\theta}_i \tag{3.80}$$

$$= \delta \boldsymbol{q}_0 + \frac{1}{2} \delta t \sum_{i=1}^{N} \left[\boldsymbol{v}_i + \delta \boldsymbol{\beta}_0 - \sum_{j=1}^{i} \boldsymbol{w}_j \right] \tag{3.81}$$

$$= \delta \boldsymbol{q}_0 + \frac{1}{2} N \delta t \delta \boldsymbol{\beta}_0 + \frac{1}{2} \delta t \sum_{i=1}^{N} \left[\boldsymbol{v}_i - (N - i + 1)\, \boldsymbol{w}_i \right] \tag{3.82}$$

$$= \delta \boldsymbol{q}_0 + \frac{1}{2} N \delta t \delta \boldsymbol{\beta}_0 + \frac{1}{2} \delta t \sum_{i=1}^{N} \left[\boldsymbol{v}_i - c_{1,N,i} \boldsymbol{w}_i \right] \tag{3.83}$$

$$\delta \boldsymbol{v}_N = \delta \boldsymbol{v}_0 + \delta t \sum_{i=1}^{N} \delta \boldsymbol{a}_i \tag{3.84}$$

$$= \delta \boldsymbol{v}_0 + \delta t \sum_{i=1}^{N} \left[\hat{R}_i^T \left(\boldsymbol{s}_i - \delta \boldsymbol{\alpha}_i - 2 \left[(\tilde{\boldsymbol{a}}_i)^{\times} \right] \delta \boldsymbol{q}_i \right) \right] \tag{3.85}$$

$$= \delta \boldsymbol{v}_0 + \delta t \sum_{i=1}^{N} \left[\hat{R}_i^T \left(\boldsymbol{s}_i - \delta \boldsymbol{\alpha}_0 - 2 \left[(\tilde{\boldsymbol{a}}_i)^{\times} \right] \left(\delta \boldsymbol{q}_0 + \frac{1}{2} N \delta t \delta \boldsymbol{\beta}_0 \right) \right) \right]$$

$$+\delta t\sum_{i=1}^{N}\left[\hat{R}_i^T c_{1,N,i}\left(\boldsymbol{z}_i-\delta t\left[(\tilde{\boldsymbol{a}}_i)^\times\right]\boldsymbol{v}_i\right)\right]$$
$$+\delta t^2\sum_{i=1}^{N}\left[\hat{R}_i^T\left[(\tilde{\boldsymbol{a}}_i)^\times\right]c_{2,N,i}\boldsymbol{w}_i\right] \tag{3.86}$$

$$\delta\boldsymbol{r}_N=\delta\boldsymbol{r}_0+\delta t\sum_{i=1}^{N}\delta\boldsymbol{v}_i \tag{3.87}$$
$$=\delta\boldsymbol{r}_0+N\delta t\,\delta\boldsymbol{v}_0$$
$$+\delta t^2\sum_{i=1}^{N}\left[\hat{R}_i^T c_{1,N,i}\left(\boldsymbol{s}_i-\delta\boldsymbol{\alpha}_0-2\left[(\tilde{\boldsymbol{a}}_i)^\times\right]\left(\delta\boldsymbol{q}_0+\frac{1}{2}N\delta t\,\delta\boldsymbol{\beta}_0\right)\right)\right]$$
$$+\delta t^2\sum_{i=1}^{N}\left[\hat{R}_i^T c_{2,N,i}\left(\boldsymbol{z}_i-\delta t\left[(\tilde{\boldsymbol{a}}_i)^\times\right]\boldsymbol{v}_i\right)\right]$$
$$+\delta t^3\sum_{i=1}^{N}\left[\hat{R}_i^T\left[(\tilde{\boldsymbol{a}}_i)^\times\right]c_{3,N,i}\boldsymbol{w}_i\right] \tag{3.88}$$

where the coefficients are $c_{1,N,i}=N-i+1$, $c_{2,N,i}=\frac{1}{2}(N-i+1)(N-i+2)$, and $c_{3,N,i}=\frac{1}{6}(N-i+1)(N-i+2)(N-i+3)$.

The Jacobian of the IMU propagation error $\boldsymbol{e}$ with respect to the state $\boldsymbol{x}_0$ is easily obtained from inspection of the above equations.

$$J_{IMU}=\begin{bmatrix} I & 0 & 0 & 0 & \frac{1}{2}N\delta t I \\ J_{IMU}^{rq} & I & N\delta t I & J_{IMU}^{r\alpha} & J_{IMU}^{r\beta} \\ J_{IMU}^{vq} & 0 & I & J_{IMU}^{v\alpha} & J_{IMU}^{v\beta} \\ 0 & 0 & 0 & I & 0 \\ 0 & 0 & 0 & 0 & I \end{bmatrix} \tag{3.89}$$

$$J_{IMU}^{rq}\equiv-2\delta t\sum_{i=1}^{N}\hat{R}_i^T\left[(\tilde{\boldsymbol{a}}_i)^\times\right]c_{1,N,i} \tag{3.90}$$

$$J_{IMU}^{r\alpha}\equiv-\delta t\sum_{i=1}^{N}\hat{R}_i^T c_{1,N,i} \tag{3.91}$$

$$J_{IMU}^{r\beta}\equiv-N\delta t^2\sum_{i=1}^{N}\hat{R}_i^T\left[(\tilde{\boldsymbol{a}}_i)^\times\right]c_{1,N,i} \tag{3.92}$$

$$J_{IMU}^{vq} \equiv -2\delta t \sum_{i=1}^{N} \hat{R}_i^T \left[(\tilde{\boldsymbol{a}}_i)^\times\right] \tag{3.93}$$

$$J_{IMU}^{v\alpha} \equiv -\delta t \sum_{i=1}^{N} \hat{R}_i^T \tag{3.94}$$

$$J_{IMU}^{v\beta} \equiv -N\delta t^2 \sum_{i=1}^{N} \hat{R}_i^T \left[(\tilde{\boldsymbol{a}}_i)^\times\right] \tag{3.95}$$

3.4.2 IMU Error Information Matrix

In performing the MLE solution, the IMU data linking the states at the time of two consecutive images is treated as a single 15-dimensional measurement. This matrix, denoted P_N, can be written in block form as

$$P_N = \begin{bmatrix} P_N^{qq} & P_N^{qr} & P_N^{qv} & P_N^{q\alpha} & P_N^{q\beta} \\ P_N^{rq} & P_N^{rr} & P_N^{rv} & P_N^{r\alpha} & P_N^{r\beta} \\ P_N^{vq} & P_N^{vr} & P_N^{vv} & P_N^{v\alpha} & P_N^{v\beta} \\ P_N^{\alpha q} & P_N^{\alpha r} & P_N^{\alpha v} & P_N^{\alpha\alpha} & P_N^{\alpha\beta} \\ P_N^{\beta q} & P_N^{\beta r} & P_N^{\beta v} & P_N^{\beta\alpha} & P_N^{\beta\beta} \end{bmatrix} \tag{3.96}$$

There are 15 unique blocks in the above symmetric matrix (e.g. $P_N^{rv} = \left(P_N^{vr}\right)^T$). These blocks are derived below using the results from above. For example, $P_N^{vq} = \mathbb{E}\left\{\delta\boldsymbol{v}_N \delta\boldsymbol{q}_N^T\right\}$ is found by simply substituting the above expressions for $\delta\boldsymbol{v}_N$ and $\delta\boldsymbol{q}_N$. To reduce the computational complexity with little-to-no loss in accuracy, the rotation and measured acceleration are assumed constant over the interval in the covariance computation. Note that this assumption can lead to unmodeled correlations in the state propagation errors, especially in the case where translation jerk or angular acceleration are large. In performing the expectations, sums of coefficients of

the following form are used to simplify the equations (i.e. eliminate the summation symbol).

$$\gamma_{0,1} \equiv \sum_{i=1}^{N} c_{1,N,i}$$
$$= \frac{1}{2}N^2 + \frac{1}{2}N \tag{3.97}$$
$$\gamma_{0,2} \equiv \sum_{i=1}^{N} c_{2,N,i}$$
$$= \frac{1}{2}\left(\frac{1}{3}N^3 + N^2 + \frac{2}{3}N\right) \tag{3.98}$$
$$\gamma_{0,3} \equiv \sum_{i=1}^{N} c_{3,N,i}$$
$$= \frac{1}{6}\left(\frac{1}{4}N^4 + \frac{3}{2}N^3 + \frac{11}{4}N^2 + \frac{3}{2}N\right) \tag{3.99}$$
$$\gamma_{1,1} \equiv \sum_{i=1}^{N} c_{1,N,i}c_{1,N,i}$$
$$= \frac{1}{3}N^3 + \frac{1}{2}N^2 + \frac{1}{6}N \tag{3.100}$$
$$\gamma_{1,2} \equiv \sum_{i=1}^{N} c_{1,N,i}c_{2,N,i}$$
$$= \frac{1}{2}\left(\frac{1}{4}N^4 + \frac{5}{6}N^3 + \frac{3}{4}N^2 + \frac{1}{6}N\right) \tag{3.101}$$
$$\gamma_{1,3} \equiv \sum_{i=1}^{N} c_{1,N,i}c_{3,N,i}$$
$$= \frac{1}{6}\left(\frac{1}{5}N^5 + \frac{5}{4}N^4 + \frac{5}{2}N^3 + \frac{7}{4}N^2 + \frac{3}{10}N\right) \tag{3.102}$$
$$\gamma_{2,2} \equiv \sum_{i=1}^{N} c_{2,N,i}c_{2,N,i}$$
$$= \frac{1}{4}\left(\frac{1}{5}N^5 + N^4 + \frac{5}{3}N^3 + N^2 + \frac{2}{15}N\right) \tag{3.103}$$
$$\gamma_{2,3} \equiv \sum_{i=1}^{N} c_{2,N,i}c_{3,N,i}$$
$$= \frac{1}{12}\left(\frac{1}{6}N^6 + \frac{13}{10}N^5 + \frac{11}{3}N^4 + \frac{9}{2}N^3 + \frac{13}{6}N^2 + \frac{1}{5}N\right) \tag{3.104}$$

$$\gamma_{3,3} \equiv \sum_{i=1}^{N} c_{3,N,i} c_{3,N,i}$$
$$= \frac{1}{36}\left(\frac{1}{7}N^7 + \frac{3}{2}N^6 + \frac{61}{10}N^5 + 12N^4 + \frac{23}{2}N^3 + \frac{9}{2}N^2 + \frac{9}{35}N\right) \tag{3.105}$$

$$P_N^{qq} = \frac{1}{4}\delta t^2 \left(N\sigma_v^2 + \sigma_w^2 \gamma_{1,1}\right) I \tag{3.106}$$

$$P_N^{qr} = \frac{1}{2}\delta t^3 \left(\gamma_{2,0}\sigma_v^2 + \gamma_{3,1}\sigma_w^2\right) \hat{R}^T \left[(\tilde{\boldsymbol{a}}_i)^\times\right] \tag{3.107}$$

$$P_N^{qv} = \frac{1}{2}\delta t^3 \left(\gamma_{1,0}\sigma_v^2 + \gamma_{2,1}\sigma_w^2\right) \hat{R}^T \left[(\tilde{\boldsymbol{a}}_i)^\times\right] \tag{3.108}$$

$$P_N^{q\alpha} = 0 \tag{3.109}$$

$$P_N^{q\beta} = -\frac{1}{2}\delta t \gamma_{1,0}\sigma_w^2 I \tag{3.110}$$

$$P_N^{rr} = \delta t^2 \left(\gamma_{1,1}\sigma_s^2 + \gamma_{2,2}\sigma_z^2\right) I$$
$$- \delta t^4 \left(\gamma_{2,2}\sigma_v^2 + \gamma_{3,3}\sigma_w^2\right) R_i^T \left[(\tilde{\boldsymbol{a}}_i)^\times\right]^2 R_i \tag{3.111}$$

$$P_N^{rv} = \delta t^2 \left(\gamma_{1,0}\sigma_s^2 + \gamma_{2,1}\sigma_z^2\right) I$$
$$- \delta t^4 \left(\gamma_{2,1}\sigma_v^2 + \gamma_{3,2}\sigma_w^2\right) R_i^T \left[(\tilde{\boldsymbol{a}}_i)^\times\right]^2 R_i \tag{3.112}$$

$$P_N^{r\alpha} = -\delta t \gamma_{2,0}\sigma_z^2 R_i^T \tag{3.113}$$

$$P_N^{r\beta} = -\delta t^2 \gamma_{3,0}\sigma_w^2 R_i^T \left[(\tilde{\boldsymbol{a}}_i)^\times\right] \tag{3.114}$$

$$P_N^{vv} = \delta t^2 \left(N\sigma_s^2 + \gamma_{1,1}\sigma_z^2\right) I$$
$$- \delta t^4 \left(\gamma_{1,1}\sigma_v^2 + \gamma_{2,2}\sigma_w^2\right) R_i^T \left[(\tilde{\boldsymbol{a}}_i)^\times\right]^2 R_i \tag{3.115}$$

$$P_N^{v\alpha} = -\delta t \gamma_{1,0}\sigma_z^2 R_i^T \tag{3.116}$$

$$P_N^{v\beta} = -\delta t^2 \gamma_{2,0}\sigma_w^2 R_i^T \left[(\tilde{\boldsymbol{a}}_i)^\times\right] \tag{3.117}$$

$$P_N^{\alpha\alpha} = N\sigma_z^2 I \tag{3.118}$$

$$P_N^{\alpha\beta} = 0 \tag{3.119}$$

$$P_N^{\beta\beta} = N\sigma_w^2 I \tag{3.120}$$

3.5 Observability

This section will conclude with a brief section on observability.

The problem of determining pose given visual measurements of *surveyed landmarks* has been studied extensively in the literature. Given three or more such correspondences in general positions (i.e. landmarks not on a line), there are up to four distinct pose solutions [89, 90]. This ambiguity can be resolved with measurement of a fourth surveyed landmark [91].

The situation is much different if the visual measurements are made of *unsurveyed landmarks* in general position. The only information obtained from a single image of the landmarks is the identification of a line in camera coordinates for each landmark to lie on. Given two such images, the situation is much better. Hartley and Zisserman give an excellent overview of two view reconstruction in Chapter 10 of [3]. In particular, if the fundamental matrix can be estimated and the camera calibration parameters are known, then the scene and pose can be reconstructed up to an ambiguity if pose and scale. The pose ambiguity is due to the fact that there is no information about a global reference frame to map the landmarks in and estimate pose relative to. The scale ambiguity manifests itself as an unknown distance between the two cameras, and an unknown distance from the cameras to the landmarks. The same ambiguity in the two view case is present in the N view case

[3].

The addition of IMU data to the visual measurements adds significant observability but also adds new parameters, namely scene scale, IMU biases and possibly an unknown gravity vector. Furthermore, the inertial motion of the target scene must be known: if it is not, these parameters must be estimated as well in order for the IMU information to be used. A paper by Martinelli, a paper by Kelly and Sukhatme, and a paper by Jones and Soatto, give excellent theoretical results on this problem [92, 93, 59]. The observability is governed by the particular trajectory followed. The key result of Martinelli is that for six or more images of two or more unsurveyed landmarks, the biases, gravity, scale, and trajectory can be reconstructed up to an ambiguity in pose. This is a well known gauge freedom which is typically resolved by treating the pose of the first camera as the global reference frame [2]. Kelly takes a different approach and shows that, in addition to scale, IMU biases, gravity, pose, and velocity, the camera-to-IMU pose is also observable given a sufficient number of images of four or more landmarks. The particular analysis is done in a continuous time formulation and shows that the parameters are observable in an arbitrarily small amount of time. Jones and Soatto give more specific cases of the types of excitation needed to observe the same parameters that Kelly demonstrated to be observable. They also present a filter to estimate the parameters sequentially [59].

The observability studies for visual and range camera systems is limited (with

or without an IMU). However, this case is much simpler and the results of Kelly in particular can be easily extended [93]. In particular, consider a co-registered visual and range camera without an IMU. Using the first image in a sequence to define a global frame, all features detected in the first image can be immediately mapped without scale ambiguity. If there are three or more such mapped landmarks in subsequent images, the pose can be determined at those times which allows velocities and accelerations to be estimated as well.

If the cameras are not co-registered (i.e. there relative pose is unknown) but the correspondence problem can be solved between the visual images and range images, then all the above parameters are still observable in addition to the visual to range camera pose. This follows from the following logic. All features detected in the first range image can be mapped in that reference frame. From the observability of visual camera pose from measurements of mapped landmarks, the visual camera pose relative to the range camera can be immediately computed. Then this reduces to the above case where the relative pose was assumed known *a priori*.

4 EXPERIMENTAL METHODS

This section will discuss two distinct approaches to experimental testing used in this dissertation.

The first method utilizes a tool chain developed for this dissertation which renders photo-realistic images. The tool allows for perfect control of the camera projection model, lighting model, camera trajectory, and the scene. This control has two distinct advantages. First, it enables a better assessment of the accuracy of a vision-aided navigation system because any experiments will real images will inevitably have imperfect *'ground truth'* data and imperfect camera models. Second, it has an enormous advantage in model validation. In particular, the ubiquitous *constant landmark and IID error* assumption can be more easily assessed because parameters for each landmark can be estimated independently of all other landmarks. This is because the coupling introduced by jointly estimating pose is removed through the use of perfect ground truth data. Both of these facts will be leveraged in later sections.

The second method is to use data from real sensors. A visual camera, range camera, and IMU on a single platform was developed in the LASR lab [94]. The system was equipped with Vicon tracking beacons to provide ground truth data and an extensive calibration procedure was performed as part of this dissertation. The platform is used to obtain real sensor data to test the algorithms developed in later

sections. Although the advantage of perfect ground truth in the first method is lost, it is necessary to demonstrate gains in accuracy and robustness in real systems. This method accomplishes that task.

This section will discuss these methods in detail. Calibration procedures and validation of the tools are presented.

4.1 Rendering Tool for Analysis

Developing models to validate models for vision-aided navigation requires an ability to accurately compute *ideal measurements* under the model assumptions. Ideal measurements are defined as the *noise-free* measurements under a given model. In order to do this, accurate ground truth data for sensor parameters, both intrinsic and extrinsic, as well as landmark position are required. This presents two challenges when using real data. First, attitude errors on the order of 0.01° will cause an error on the order of 1 pixel for common cameras and optics. Therefore, the ability to estimate ideal measurements to an accuracy of 0.1 pixels with ground truth data is unrealistic for most ground truth systems. Second, the notion of a true fixed landmark position in images of real scenes is dubious in an of itself: this is an assumption that this dissertation challenges. These issues suggest that synthetic data with perfect ground truth may be a better option for developing models (which can then be verified with real data). A rendering tool has been developed for this purpose.

The rendering tool can generate photo-realistic images for a given camera model

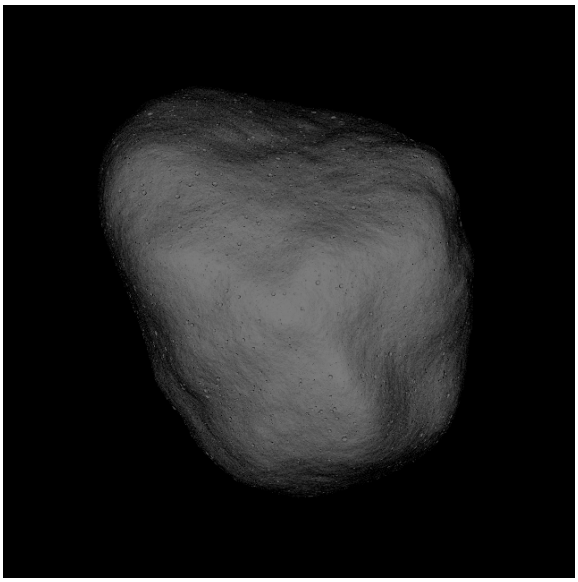


Figure 4.1: Sample image of Temple 1 comet using rendering tool.

and trajectory. A high fiedlity geometric model of the Temple-1 comet was obtained from NASA JPL [95]. An example image using the Oren-Nayar lighting model is given in Figure (4.1) [96]. The Oren-Nayar lighting model computes the intensity of light reflected from a rough surface.

Consider a triangle facet model of a scene to be imaged. The important angles at a particular facet are: the normal-to-sun angle θ_s and the normal-to-camera angle θ_c. The computed reflection is

$$L = c_1 c_2 \cos(\theta_i)(A + B\max(0, \cos(\theta_i - \theta_r)) \sin(\alpha) \tan\beta \tag{4.1}$$

where c_1 is the albedo and c_2 is proportional to incident light intensity. The α and β are simply the maximum and minimum of θ_s and θ_r respectively. The values of A and B are functions of the roughness parameter σ:

$$A = 1 - \frac{0.5\sigma^2}{\sigma^2 + 0.57} \tag{4.2}$$

$$B = \frac{0.45\sigma^2}{\sigma^2 + 0.09} \tag{4.3}$$

which is equal to the standard deviation of *roughness slopes* (i.e. surface slopes at the scale of an individual facet). This was set as $\sigma = 0.01$ for the images rendered of Temple-1 in this dissertation.

Prior to using the rendering tool for model analysis and algorithm testing, it is important to verify that the results it gives are consistent with the assumed pinhole camera model which is done in the next section.

4.1.1 Validation

In this section, a camera calibration is performed on the rendered images. The rendering tool accepts a camera parameter file as input. The model used is that given in Equation (3.16) without any additional distortion. The input parameter file contains a single focal length, f (equal to f_u and f_v of Equation (3.16)), the principal point, (c_u, c_v) and image size for the camera. In comparison with the models given in The focal length and principal point are used to compute the standard camera calibration matrix

$$K = \begin{bmatrix} f & 0 & c_u \\ 0 & f & c_v \\ 0 & 0 & 1 \end{bmatrix} \tag{4.4}$$

This completely describes the camera model of the rendering tool: no distortion is added. To verify that the computed images are consistent with this model, the Caltech camera calibration toolbox is used to perform a calibration [97].

No distortion; Aspect Ratio = 1.				
Parameter	Model	Estimate	Error	1-σ
f	1000.00	999.42	0.58	4.12
c_u	322.00	321.67	0.33	1.73
c_v	235.00	234.70	0.30	1.74

No Distortion.				
Parameter	Model	Estimate	Error	1-σ
f_u	1000.00	999.35	0.65	4.13
f_v	1000.00	999.35	0.53	4.13
c_u	322.00	321.59	0.41	1.74
c_v	235.00	234.74	0.26	1.74

Full calibration.				
Parameter	Model	Estimate	Error	1-σ
f_u	1000.00	999.32	0.68	4.19
f_v	1000.00	999.32	0.55	4.18
c_u	322.00	322.07	-0.07	4.37
c_v	235.00	237.23	-2.23	4.55
d_1	0.0000	0.0005	-0.0005	0.0495
d_2	0.0000	-0.0402	0.0402	1.7008
d_3	0.0000	0.0010	-0.0010	0.0017
d_4	0.0000	0.0002	-0.0002	0.0016
d_5	0.0000	0.0000	-0.0000	0.0000

Table 4.1: Results of three calibrations on a set of forty images.

Forty images of a checkerboard model are rendered from various poses. The corners of the checkerboard pattern are extracted and passed to a calibration routine. Three separate calibrations are performed. The first assumes no distortion and assumes unit aspect ratio (i.e. square pixels: $f_u = f_v$). The second assumes no distortion but removes the unit aspect ratio assumption. The third does not assume zero distortion or unit aspect ratio. The results are tabulated in Table (4.1) .

An important implication see in Table (4.1) is that as the number of parameters in the calibration model increases, the certainty in the estimates degrades.

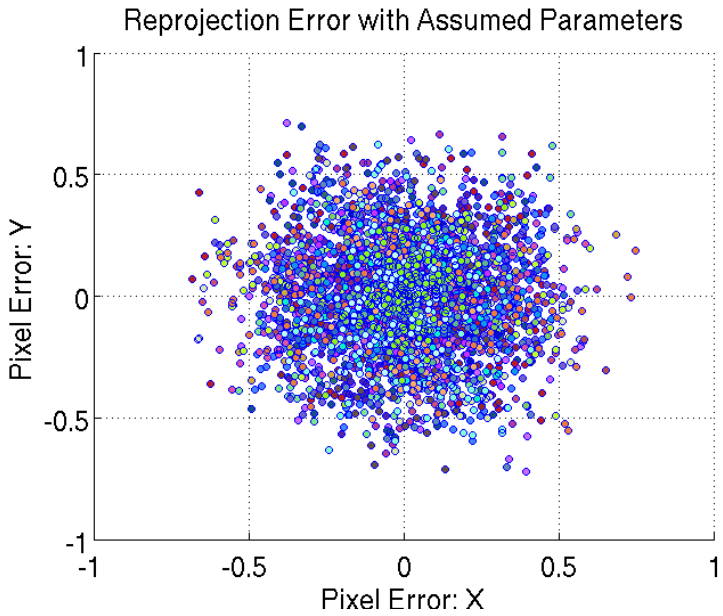


Figure 4.2: Difference between extracted corners and estimated corner locations on checkerboard pattern assuming the model parameters are correct. Each color represents a different image.

This is commonly referred to as the *bias-variance tradeoff*: added model complexity increases the variability in the results but reduces systemic bias. For the purposes here, namely model validation, the results are sufficient to conclude that the assumed model is consistent with the data in all three calibrations.

Using the assumed camera parameters and pose, the estimated measurements are compared to the extracted corners. The resulting residual errors are plotted in Figure 4.2. These residuals appear to be zero mean with isotropic covariance on the order of $0.2^2\,\text{pixels}^2$. This suggests that the model parameters used by the rendering tool are consistent with the output images. [1]

[1]Note that the pixel convention used here has the origin at the very top left of the image (not in the top-left pixel center).

4.2 LENS: LASR Embedded Navigation Stack

This section describes the LASR Embedded Navigation Stack (LENS) developed by Whitten [94]. The LENS is a flexible platform for prototyping and testing navigation systems. The key components of the platform relevant to this dissertation are

1. *Point Grey Blackfly* 1.3 MP monochrome *GigE* camera with 12 mm *Mega Pixel* fixed focal length lens. The camera has a global shutter and a Sony ICX445 CCD 1/3" 1288 $\times$ 964 pixel sensor (3.75 micrometer pixel size).
2. *MESA Swiss Ranger 4500* time-of-flight camera. The camera is equipped with LEDs which transmit a waveform towards the scene. A filter on the camera lens allows light matching the transmitted wavelength to pass returned light through to a 176 $\times$ 144 grid of pixels which sample the returned intensity. Correlation with the transmitted pulse is used to measure time-of-flight for each pixel individually with an accuracy on the order of 1 centimeter.
3. *VectorNav VN100* IMU is a sensor suite containing a 3-axis gyroscope, 3-axis accelerometer, and 3-axis magnetometer. Testing and characterization of the gyroscope and accelerometer is summarized in Table (3.2) and Table (3.1) . The magnetometer is not used in the experiments presented in this dissertation.
4. A set of 10 *Vicon* markers are attached to the LENS to capture ground truth pose of the platform. Vicon is accurate to 1 millimeter. The baseline of the markers is on the order of 25 centimeters which should allow for an attitude

accuracy on the order of 1 milli-radians (i.e. $\frac{0.1cm}{25cm}(10\,\text{markers})^{\frac{-1}{2}}$).

4.2.1 Calibration

The first major challenge in dealing with this system is that the relative pose between all sensors and the intrinsic parameters of each sensor are unknown *a priori*. Consistent with the model developed in Section 3, all sensor poses relative to the LENS body frame are determined. The LENS frame is defined as the frame that the Vicon beacon positions are described relative to.

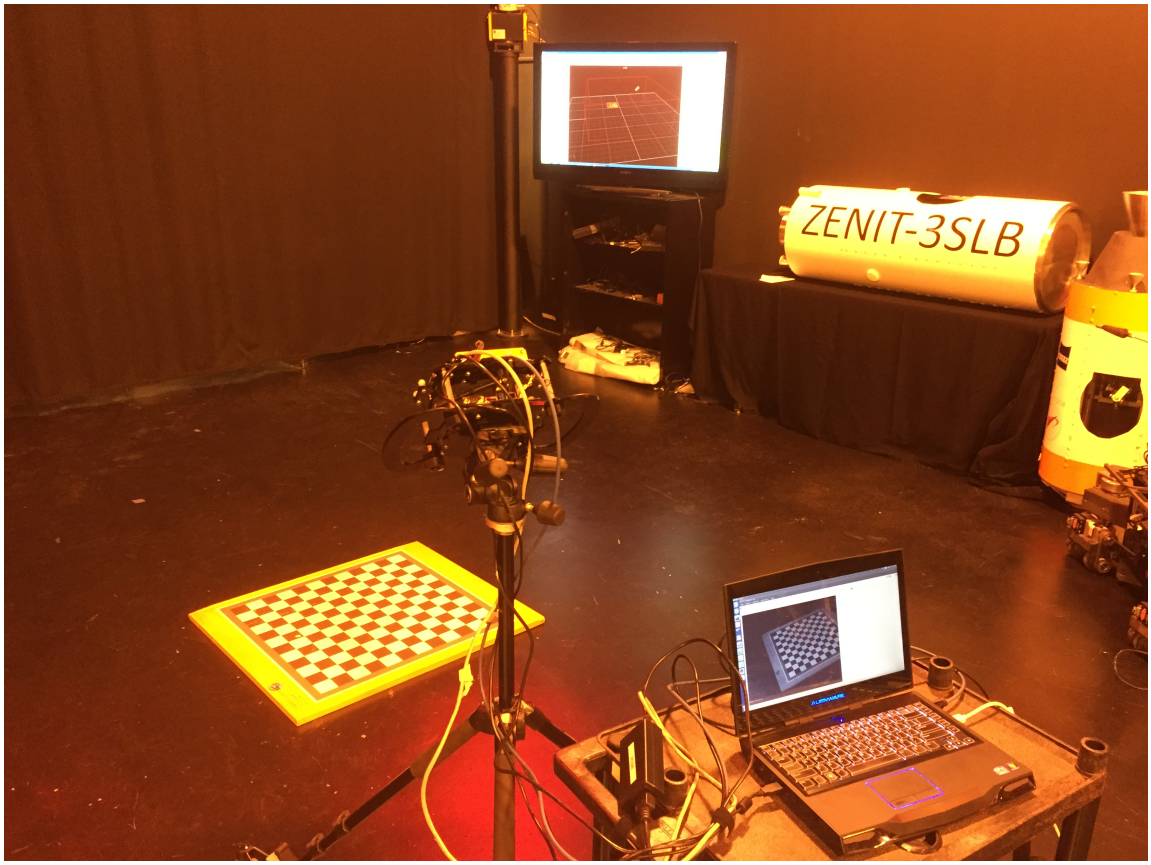
The process to define these positions and rotations is as follows. The LENS is built as a set of 3/16" rigid acrylic sheets separated by standoffs. Seven of the beacons are on one sheet and three are on a second. The LENS z-axis is defined to be normal to these sheets and the x-y plane is defined to pass through the three beacons on the second plane. Therefore there is only one unique unknown z-value which must be determined (i.e. it is common to all of the other seven beacons). In addition, one of the three beacons on the second plane is defined as the origin. The other two beacons are used to define the y-axis by requiring that they share the same y-value. Therefore the total number of parameters that describe the beacon locations is 18 (one z-value for seven beacons, an x-value and y-value for each of seven beacons, a y-value shared between two beacons, and an x-value for each of those two beacons). These constraints are incorporated into a Vicon template file and a batch of 1000 Vicon frames is used to determine the positions which are output into a skeleton file. This completes the LENS body frame specification.

There are two alternatives to estimating the body-to-sensor transformation for each sensor. The first is to compute them based on the CAD model locations of the mounting holes of the Vicon beacons and sensors. This method is very easy to do but is not very accurate for the cameras due to the fact that the exact origin of the camera frames (visual and range) is difficult to measure precisely. On the other hand, the method will be fairly accurate for the IMU as the IMU frame is well defined by the manufacturer relative to mounting holes. The second method has two steps. The first step is to perform an inter-sensor calibration to get the relative pose between each sensor. The second step is to take images of a calibration target that can be tracked in the Vicon lab frame. Since the LENS-to-lab pose and target-to-lab pose is given by Vicon and the camera-to-target pose can be determined from the images, the LENS-to-camera pose can be computed. The second method was used to obtain the calibration and the first method was used as a sanity check on the resulting solution. An image of the experimental method for the second step of the second method (obtaining LENS-to-camera pose) is shown in Figure (4.3).

The inter-sensor calibration was performed using a set of tools developed by Furgale *et al* [98, 99]. The tool can perform

1. Multiple camera calibration: Compute the intrinsic parameters of multiple cameras and the inter-camera pose between each pair of cameras given images (time-synced from all cameras) of a known target.
2. Camera-IMU Calibration: Compute the intrinsic parameters of the camera and

Figure 4.3: Experimental setup to calibrate the LENS-to-camera pose. The LENS frame is defined by the Vicon beacons. The checkerboard also has a frame defined by small beacons located at a subset of the checkerboard corners.



the relative pose between a camera and IMU given images and simultaneous IMU data.

3. Multiple Camera-IMU Calibration: Computes the intrinsic parameters of each camera, and the pose of each camera relative to the IMU given time-synced images and IMU data.

The first tool performs a simple batch optimization over the intrinsinc parameters of both cameras, the extrinsic parameters for each image of one camera, and the

inter-camera pose. This is very similar to standard camera calibration.

The second two tools which compute the IMU-to-camera transformations are much different. They model the IMU-to-target pose as a continuous time function that is parametrized by spline coefficients. This leads to a lower dimensional parameter space as compared to estimating the IMU pose at all IMU output times (because the IMU is a high rate sensor). This requires the camera to have a frame rate higher than the dominant frequencies of the motion during calibration data collection. Therefore is important to give the camera smooth motion (minimize jerking as much as possible).

Note that these tools were designed for visual cameras: not the time-of-flight type cameras. Fortunately, an easy work around was found. The *Swiss Ranger* can also output the intensity of returned light. A checker board calibration target was found that had a significantly different albedo at the wavelength used by the *Swiss ranger*. Therefore the returned intensity image could be passed to the calibration tool as if it was a grayscale image. For this particular calibration tool, the *Swiss Ranger* images were too small for the target corner extraction program. The work-around for this was to simply scale the image by a factor of four and then account for this in the resulting intrinsic parameters.

The first attempt of this calibration utilized the Multiple Camera-IMU Calibration tool and was **unsuccessful**. Software was developed to interface with the cameras and send a software trigger to each camera at the same time. It appears

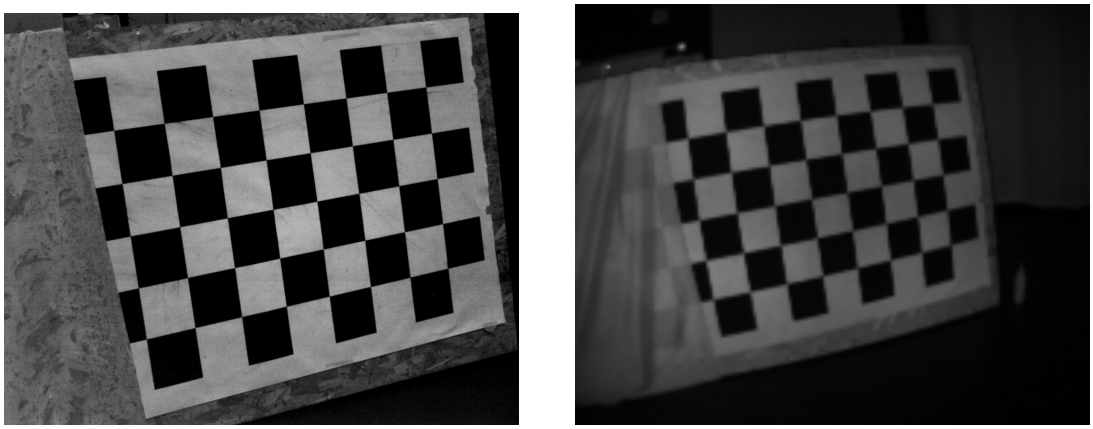


Figure 4.4: Calibration images from the *Point Grey Blackfly* (left) and *Swiss Range* intensity image scaled by four in size (right).

that the latency between the software trigger time and actual image capture was significantly different for each camera. Therefore, the calibration tool could not fit a single fixed camera-to-camera transformation that was consistent with the data.

To work around this, the inter-sensor calibration was broken into two stages. In the first stage, the Multiple Camera Calibration tool was used. The LENS was held stationary while the images were triggered to guarantee no motion during the trigger-to-capture latency of both cameras. This also had the added benefit of eliminating motion blur. This was especially important because the exposure time was increased to compensate for a small aperture and low digital gain which were used to reduce focusing issues and reduce image noise respectively. A side-by-side comparison of the images from the two cameras is shown in Figure (4.4).

4.3 Batch Estimation

A flexible tool for batch estimation was developed for this dissertation. The tool is an extension of the excellent work of Kuemmerle *et al.* in developing the General Graph Optimization tool known as *g2o* [48]. The open source *g2o* software solves graph optimization problems. A graph optimization problem is defined as follows. The parameters to be optimized over are partitioned into a set of graph vertices $\hat{\boldsymbol{\theta}}_i$. The vertices are connected by edges which represent measurements. The k'th measurement can constrain n_k vertices $\hat{\boldsymbol{\theta}}_{k_1}, \ldots \hat{\boldsymbol{\theta}}_{k_{n_k}}$ by introducing an error term $\boldsymbol{e}_k$ into the cost function where $\boldsymbol{e}_k$ is a function of the measurement and associated vertices. The error term $\boldsymbol{e}_k$ has a weighted squared cost weight matrix Ω_k. This leads to a solution

$$\hat{\boldsymbol{\theta}} = \text{argmin}_{\boldsymbol{\theta}} \sum_{k \in \mathscr{C}} \boldsymbol{e}_k(\hat{\boldsymbol{\theta}}_{k_1}, \ldots \hat{\boldsymbol{\theta}}_{k_{n_k}}) \Omega_k \boldsymbol{e}_k(\hat{\boldsymbol{\theta}}_{k_1}, \ldots \hat{\boldsymbol{\theta}}_{k_{n_k}}) \tag{4.5}$$

that is graph optimal.

The *g2o* software also allows the user to specify an alternative cost function $\boldsymbol{\rho}()$ for each measurement k individually. The software adjusts the weight matrix on each iteration in the manner presented in Section 2.4.

In order to use *g2o* in this dissertation, software defining all vertices, edges, and cost functions had to be developed. The details of these will be given in Section 5.

The optimizer iteratively computes and applies corrections to the parameters. On each iteration, a sparse linear system is solved. The solver can be configured by

the user.

Preliminary testing was performed with both the Gauss-Newton, Levenberg-Marquardt solver and the Dog Leg algorithm discussed in Section 2.4. The latter two algorithms gave similar performance which was superior to the first. Ultimately the Levenberg-Marquardt solver was selected for all results shown in this dissertation.

4.3.1 Attitude Parametrization

One important issue in the vision-aided navigation problem is the parametrization of attitude. Attitude has three degrees-of-freedom which necessitates at least three parameters. However, there are no known singularity-free minimal (three element) parametrizations. Higher dimensional parameterizations require constraints to be enforced between the parameters. Enforcement of these constraints complicates the solution.

The *g2o* software has a convient hook to handle problems of this type. In particular, the parametrization of the correction of $\boldsymbol{\theta}$ can be different than the the parametrization of $\boldsymbol{\theta}$ itself. In the results in this dissertation, the attitude components of $\boldsymbol{\theta}$ are represented by quaternions. The three-element differential quaternion $\boldsymbol{\delta q}$ is used to represent the correction to the four-element quaternion $\boldsymbol{q}$. The corrected quaternion is

$$\boldsymbol{q} \leftarrow \begin{bmatrix} \boldsymbol{\delta q} \\ 1 \end{bmatrix} \otimes \boldsymbol{q} \tag{4.6}$$

where the quaternion multiplication operator is defined in Crassidis and Junkins [17].

4.3.2 Solving for Scale

A key extension to *g2o* developed in this dissertation is a scale estimation module. The module jointly estimates the scale of the errors and the unknown parameters $\boldsymbol{\theta}$. The theory behind this problem is presented in Section 2.3.8. The current section gives the algorithm implemented in the batch estimator to solve the necessary conditions given in Equation (2.160) of Section 2.3.8 which are

$$\begin{aligned} \mathbf{0} &= \sum \psi(r_i) \mathrm{h}_i'(\boldsymbol{\theta}) \\ \mathbf{0} &= \sum \psi_s(r_i) \end{aligned} \tag{4.7}$$

where $r_i = (y_i - \mathrm{h}_i(\boldsymbol{\theta}))/\sigma$ is the scaled residual. This algorithm is given by Huber [29].

Let the residual be computed at the estimate at a given iteration $\boldsymbol{\theta}^{(k)}$ and $\sigma^{(k)}$. The scale can be updated as

$$\sigma^{(k+1)} = \sigma^{(k)} \sqrt{\frac{1}{ma} \sum_i \psi_s(r_i)} \tag{4.8}$$

where m is the number of measurements and a is a constant used to obtain an unbiased correction. In the location-scale problem with Gaussian errors, this reduces to

$$\sigma^{(k+1)} = \sigma^{(k)} \sqrt{\frac{1}{m-n} \sum_i r_i^2} \tag{4.9}$$

$$= \sigma^{(k)} \sqrt{\frac{1}{m-n} \sum_i (y_i - \mathrm{h}_i(\boldsymbol{\theta}^{(k)}))^2 / (\sigma^{(k)})^2} \tag{4.10}$$

$$= \sqrt{\frac{1}{m-n} \sum_i (y_i - \mathrm{h}_i(\boldsymbol{\theta}^{(k)}))^2} \tag{4.11}$$

where n is the number of parameters.

The updated parameters $\boldsymbol{\theta}$ can be computed as

$$\boldsymbol{\theta}^{(k+1)} = \boldsymbol{\theta}^{(k)} + \left(\sum_i w_i \mathrm{h}_i'(\boldsymbol{\theta}^{(k)}) \mathrm{h}_i'^{\mathrm{T}}(\boldsymbol{\theta}^{(k)}) \right)^{-1} \sum_i w_i r_i \mathrm{h}_i'(\boldsymbol{\theta}^{(k)}) \tag{4.12}$$

where the weight w_i is

$$w_i = \psi(r_i)/r_i \tag{4.13}$$

which is analagous to Equation (2.203) but with a jointly estimated scale parameter.

The approach by Huber is to recompute both the scale and $\boldsymbol{\theta}$ parameters on each iteration [29]. There are other alternatives in the literature. Maronna suggests using a scale-free estimate such as the $\mathscr{L}_1$ norm to get a preliminary estimate of parameters and then use the corresponding residuals to get a scale estimate [35]. The scale estimate is then used to get a corrected parameter estimate. In this sense, the algorithm is a *one-step scale estimator*. Note that Maronna warns against using the $\mathscr{L}_1$ norm in the *random-carriers linear regression problem* which is not applicable in this work. The least squares algorithm is also scale indepedent but is known to be very sensitive to modeling errors. The least-median-of-squares estimator developed by Rousseeuw is a more robust option for one-step scale estimators [100].

The software developed in this dissertation allows the flexibility of choosing how often to update the scale (*e.g.* once or on every iteration) and the ability to change the cost function after scale estimation is deemed complete. Results for different schemes will be given in Section 5.

4.3.3 Fixing Gauge Freedom

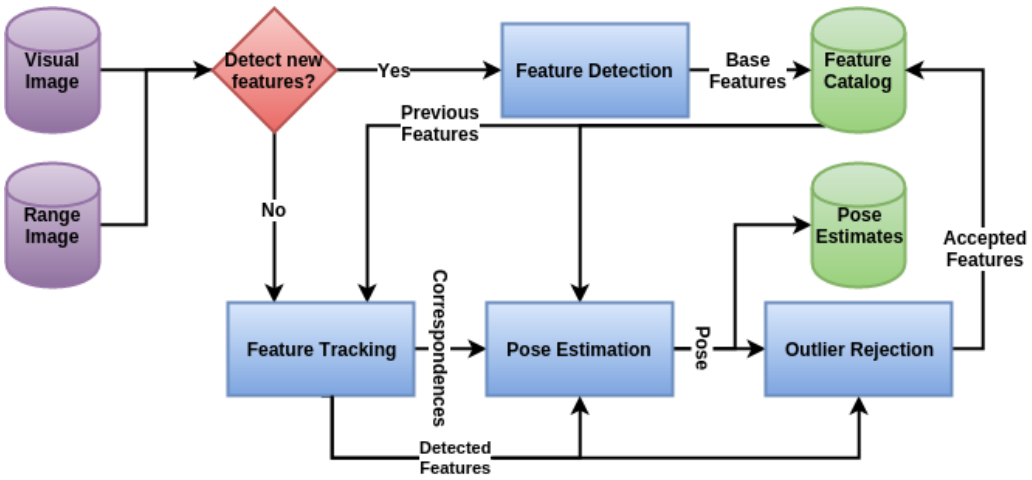
Visual measurements are typically used with other measurement types like IMU and ranging data to estimate sensor trajectories. However, under the motivation of improving visual measurement models for the purposes of estimation, many tests will be performed using only visual measurements.

Section 3.5 discusses observability issues for visual only measurements. When all landmarks are unsurveyed (i.e. their locations are unknown) there is a gauge freedom of scale and a single rigid-body transformation. To remove the scale ambiguity, the batch estimator is given a *pseudo-measurement* between the first and last camera of the sequence. As long as the cost function associated with this meaasurement is monotonically increasing with the magnitude of the error, the ambiguity will be removed. The particular form of the cost is irrelevant: it does not matter if the constraint is exactly met or not. Therefore a simple least squares cost is used. The rigid-body transformation ambiguity is removed by setting the scene refrence frame to be exactly equal to the camera frame at the time of the first image. This effectively removes the pose parameters of the first camera from the optimizer.

4.4 Computer Vision Pipeline

A computer vision pipeline was developed which serves two purposes. First, the pipeline accepts visual images or visual-and-range images and outputs the visual feature trajectories needed by the estimator. Second, the pipeline computes initial

Figure 4.5: Block diagram of the developed computer vision pipeline.



estimates of pose and landmark parameters.

A block diagram of the pipeline is shown in Figure (4.5). The first image is always used to detect features. If a sufficient number of features were tracked (or detected) in the previous image, then feature tracking is performed on the current image. Feature tracking simply solves the correspondence problem between features in two images. The feature locations in the current image are used along with the feature locations in a previous image (configurable to be the most recent image or the most recent *base image*) to perform pose estimation. If range images are available, this pose estimation uses a RANSAC implementation of the OLTAE algorithm developed for this dissertation [11, 101]. Otherwise, the fundamental matrix is computed in place of the true 6 DOF pose [3]. The pose estimate is then used for outlier rejection. The resulting features and pose estimates are outputs of the pipeline.

In the spirit of this dissertation, the outlier rejection threshold are extremely loose. The majority of features that are rejected are due to failure of the visual

tracking algorithm to return any correspondence.

The developed pipeline uses *OpenCV*, an open source computer vision library [102]. The pipeline was designed to be flexible enough so that any feature detection or tracking algorithm can be used in a *plug-and-play* fashion. Unless otherwise noted, all results use the ORB feature detector and a multi-scale implementation of the Lukas-Kanade tracker [103, 104, 74, 64].

Before concluding the section, a brief discussion of the RANSAC OLTAE solution is needed. This implementation is unique to this dissertation and was found to give excellent performance.

4.4.1 RANSAC OLTAE

The RANSAC OLTAE developed for this dissertation works as follows. Random hypothesized consensus sets of three measurements are iteratively generated. For each hypothesized set:

1. OLTAE is performed to compute pose
2. Measurements are labeled as inliers or outliers with a threshold that depends on the range to each measurement (so that errors in pixel-space are properly scaled). This threshold is intentionally set to be very loose.
3. The largest inlier set is used in a robust OLTAE routine.

   The robust OLTAE algorithm is an iterative M-estimator using a truncated

Huber cost function:

$$\rho(e) = \begin{cases} \frac{1}{2}e^2 & |e| < \delta_{near} \\ \delta_{near}|e| - \frac{1}{2}\delta_{near}^2 & \delta_{far} < |e| \leq \delta_{near} \\ \delta_{near}\delta_{far} - \frac{1}{2}\delta_{near}^2 & |e| \geq \delta_{near} \end{cases} \tag{4.14}$$

This estimation routine, although not a primary contribution of this dissertation, provided more reliable pose estimates. The advantage comes from the combination of a loose threshold on inliers and the robust nature of the batch solution on inliers. The loose threshold leads to an acceptable solution in fewer RANSAC iterations and reduces the false-positive rate for outlier labeling. The robust OLTAE can work with lower quality preliminary solutions and provides a soft down-weighting of poor measurements.

5 VISUAL MEASUREMENT MODELING

This section discusses models for visual measurements which are needed for vision-aided navigation. This section proceeds as follows. First the standard structural model is presented. This model assumes constant landmark positions and IID feature measurement errors. An analysis of the consistency of this model with experimental data is presented. Evidence of an inconsistency suggests other models may be necessary. Four novel structural models are proposed which are important contributions of this dissertation.

For each of the five models, one standard and four novel, the equations are given and methods of estimating the parameters under a class of probability distributions for the random errors are discussed. The solution to the estimation problem for the case of unknown error scale is also given for each model.

Next, a large dataset used to test these models is described. The dataset involves rendered images over a range of different trajectory types. The dataset contains ground truth for the trajectory but not ground truth for the landmarks which are automatically determined by the feature detection algorithm.

The dataset is partitioned into two groups. The first group is used to develop probabilistic models for the random errors under the assumption of a specific structural model. The second group is used to test the performance of various estimators under the assumption of a particular pair of probabilistic and structural models.

These two datasets will be referred to as the *learning* dataset and the *evaluation* dataset respectively.

In developing the probabilistic model, there are two options. The first option involves completely specifying the model based on the learning dataset. The second option involves leaving some parameters of the model free to be jointly estimated based on the evaluation dataset. In particular, the *scale* of the errors can be left free. Both options are considered in this section.

In order to implement the second option, algorithms to jointly estimate the free parameters must be developed. Such an algorithm is presented in this section. Modifications of this algorithm to the particular model are highlighted.

The estimators derived from each structural model and associated probabilistic model are applied to the test dataset. Performance is analyzed both for each trajectory type individually and aggregated over all trajectory types. This is done for both the case of free and fixed error scale. Key trends of interest between estimator performance, tuning parameters, and the trajectory class are enumerated. For each noted trend, a thorough discussion is given with analysis to support explanations for the trends. Finally, tradeoffs in the designed estimators are summarized.

5.1 Test Datasets

Before getting into the details of the newly developed visual models, test datasets used to study the models are described. The datasets serve two purposes.

The first purpose is in developing the models themselves. The second purpose is in evaluating estimators designed around the developed models. It is important to partition the dataset into two separate groups: one for the first purpose and one for the second. Using the same dataset for both purposes may lead to an overfit of the model to the particular data. The two sets will be referred to as the *learning* set and *evaluation* set respectively.

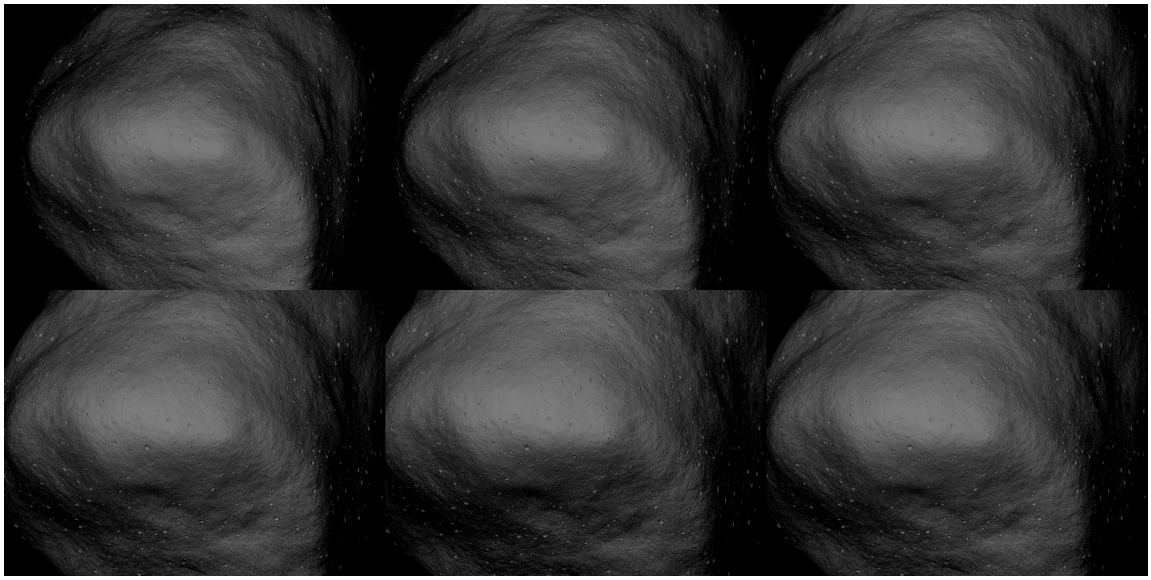
Each group has three different classes of camera trajectories. For each trajectory in each group, there are 100 sets of 20 images rendered with the rendering tool discussed in Section 4.1. Each set has a randomly generated trajectory belonging to one of three trajectory classes. The trajectory classes are

1. *Descent*: The camera descends towards the Temple-1 model as the model rotates by 45 degrees relative to the sun. The camera maintains constant comet-relative attitude and constant position in the plane orthogonal to vertical (i.e. only altitude changes from 10 to 8 kilometers). The initial comet-relative position is random (initial altitude is fixed) and the attitude is set such that the camera points towards the center of the comet and so that the sun is initially behind the camera. An example is shown in Figure (5.1) where a subset of six of the twenty images are displayed.
2. *Short Orbit*: The camera points towards the comet center as it moves through 15° of a circular orbit.
3. *Long Orbit*: The camera points towards the comet center as it moves through

30° of a circular orbit. The altitude is constant at 10 kilometers and the attitude is set such that the camera points to the comet center with the sun nominally behind the camera.

A Lucas-Kanade feature tracker was used to generate the feature tracks for each image set. The feature tracks can either be studied for model development (learning set) or can be passed to the estimator described in Section 4.3 (evaluation set). This estimator is configurable: the model type, cost function type, and cost function parameters can be set via command line arguments. Therefore the same feature tracks can be run through the various estimators discussed in this section.

Figure 5.1: A subset of six images out of twenty along a descent trajectory. The lighting angle changes during the sequence. The first image is in the top-left and the others are ordered clockwise.



5.2 Standard Model

The standard model used in vision-aided navigation systems with feature tracking assumes that all tracked features correspond to landmarks with fixed location and that each feature measurement error is independent of all others. In particular, consider landmark j which is located at a fixed location $\boldsymbol{x}_j$ with respect to a scene-fixed reference frame. The measurement of this landmark at time i is

$$\tilde{\boldsymbol{y}}_{i,j} = \pi\left(KR_{c/m}\left(R_i\left(\boldsymbol{x}_j + \boldsymbol{r}_i\right) + \left[\boldsymbol{r}_{c/m}\right]_m\right)\right) + \boldsymbol{v}_{i,j} \tag{5.1}$$

where $R_{c/m}$ is the rotation matrix that transform body-fixed-navigation-frame (say the IMU frame) coordinates to camera coordinates and $\left[\boldsymbol{r}_{c/m}\right]_m$ is the IMU-to-camera vector in IMU coordinates. The unknown pose of the IMU relative to global coordinates is represented by the rotation matrix R_i and position vector $\boldsymbol{r}_i$. The projection function $\pi()$ can take on various forms depending on the camera used. More details can be found in Section 3.2 which presents the same structural model.

Note that, for n_ℓ landmarks and n_c images, there are $3n_\ell + 6n_c$ unknown parameters: three for each landmark and six for each camera. Let $\boldsymbol{\theta}_\ell$ and $\boldsymbol{\theta}_c$ represent the unknown landmark and camera parameters respectively. Also let

$$\boldsymbol{\theta} = \begin{bmatrix} \boldsymbol{\theta}_c \\ \boldsymbol{\theta}_\ell \end{bmatrix} \tag{5.2}$$

be the joint parameter vector.

For the purposes of the estimator, the random variable $\boldsymbol{v}_{i,j}$ is assumed to be a zero-mean isotropic random variable with each component having a distribution of

the form

$$\nu \sim \frac{1}{\sigma}\mathrm{f}\left(\frac{\nu}{\sigma}\right) \tag{5.3}$$

where σ is the scale of the errors. Therefore, an MLE designed around this distribution has a solution

$$\hat{\boldsymbol{\theta}} = \mathrm{argmax}_{\boldsymbol{\theta}} \sum_{i,j} \log \frac{1}{\sigma}\mathrm{f}\left(\frac{1}{\sigma}\left(\tilde{\mathbf{y}}_{i,j} - \pi\left(KR_{c/m}\left(R_i\left(\boldsymbol{x}_j + \boldsymbol{r}_i\right) + \left[\boldsymbol{r}_{c/m}\right]_m\right)\right)\right)\right) \tag{5.4}$$

This problem has an important sparsity that should be taken advantage of in order to reduce the computational complexity. To see that this is true, consider the result from Hartley and Zisserman [3]. Define a joint measurement vector

$$\tilde{\boldsymbol{Y}} \equiv \left[\tilde{\mathbf{y}}_{1,1}^{\mathrm{T}}, \tilde{\mathbf{y}}_{1,2}^{\mathrm{T}}, \ldots, \tilde{\mathbf{y}}_{1,n_\ell}^{\mathrm{T}}, \tilde{\mathbf{y}}_{2,1}^{\mathrm{T}}, \ldots, \tilde{\mathbf{y}}_{n_c,n_\ell}^{\mathrm{T}}\right]^T \tag{5.5}$$

The M-estimate (or MLE) is then equivalent to

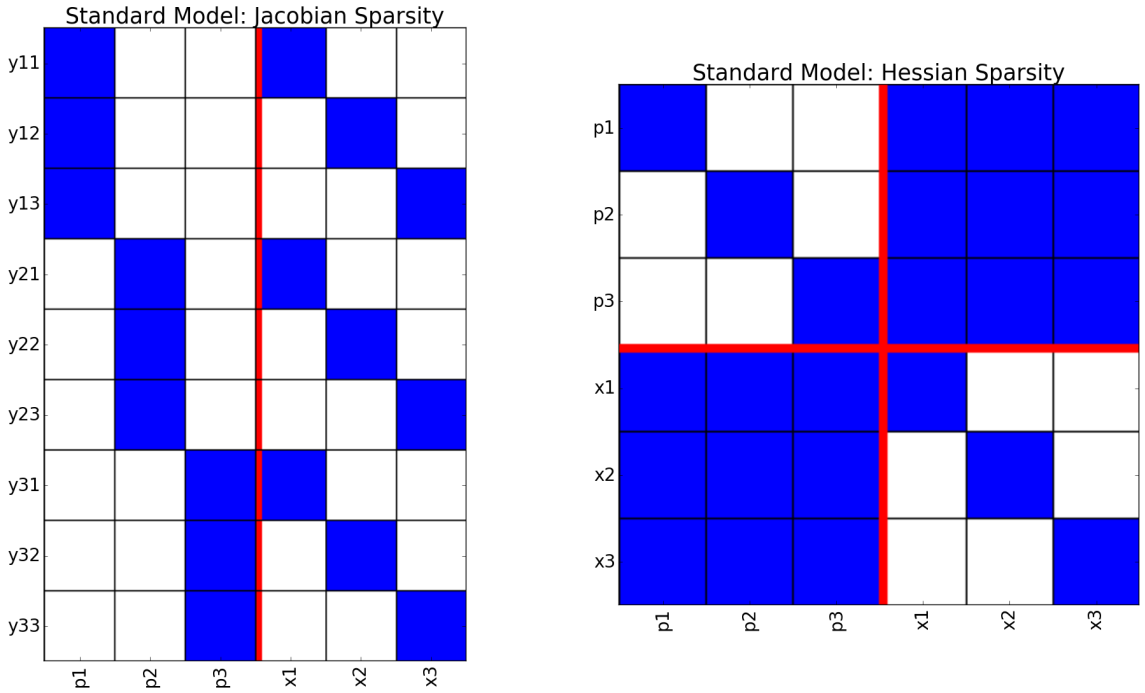
$$\hat{\boldsymbol{\theta}} = \mathrm{argmax}_{\boldsymbol{\theta}} \sum_k \rho(\tilde{\boldsymbol{Y}}_k - \hat{\boldsymbol{Y}}_k) \tag{5.6}$$

where $\rho(x) = \log \frac{1}{\sigma}\mathrm{f}(x/\sigma)$ and the summation is over the individual elements of the residual vector. Each iteration requires solving an equation of the form

$$\delta\boldsymbol{\theta}^{(k+1)} = \left(H^T W H\right) H^T W \left(\tilde{\boldsymbol{Y}} - \hat{\boldsymbol{Y}}^{(k)}\right) \tag{5.7}$$

where W is the diagonal weight matrix, $\hat{\boldsymbol{Y}}^{(k)}$ is the expected measurement conditioned on the current estimate $\boldsymbol{\theta}^{(k)}$, and H is the full Jacobian of the predicted measurement $\hat{\boldsymbol{Y}}^{(k)}$ with respect to the parameter correction $\delta\boldsymbol{\theta}$. The equations for these Jacobians are given in Section 3.2. Note that the H matrix is not fully populated: $\hat{\mathbf{y}}_{i,j}$ only depends on two sub-blocks of $\boldsymbol{\theta}$. The resulting Jacobian H and Hessian $H^T H$ has the form shown in Figure (5.2).

Figure 5.2: Block structure of the linearized system for constant landmarks and IID errors (two cameras, three landmarks): Jacobian (left) and Hessian (right). The blue and white blocks represent non-zero and zero terms respectively. The red lines partition the pose and landmark parameters.



If the Hessian matrix is fully populated, then the complexity of the solution is $\mathcal{O}((3n_\ell + 6n_c)^3)$ which is a well-known fact for dense linear systems. The sparse form of the Hessian shown above allows the elimination of the landmark (or camera) parameters so that the camera (or landmark) parameter corrections can be solved for independently. This step has complexity $\mathcal{O}((6n_c)^3)$ (or $\mathcal{O}((3n_\ell)^3)$). Then the correction to each individual landmark (or camera) parameter block can be computed independently. This step has complexity $\mathcal{O}(n_\ell)$ (or $\mathcal{O}(n_c)$). The choice of whether to first eliminate landmark parameters or camera parameters should be based on the number of such blocks. If $n_c > n_\ell$, then the camera parameters should be eliminated

first and vice versa. This is automatically handled by the optimization framework.

5.2.1 Serial Correlations

The standard model has a single three-dimensional position parameter to represent the landmark at the time of all images. Given a set of feature trajectories (in image space), landmark positions can be estimated and then used to compute residuals. These residuals could be used as proxies for the true errors to study the error distribution. This presents three issues. First, statistics on the residuals will appear smaller than the same statistics on the true errors because the landmark estimate is biased for the given measurements. Second, all the pseudo-error samples for a given feature trajectory will be coupled. Third, there is a certain circular logic in estimating the landmark position, which would implicitly assume a preliminary nominal model, and then using those residuals to obtain a new nominal model. Clearly the new nominal model will depend on the preliminary one.

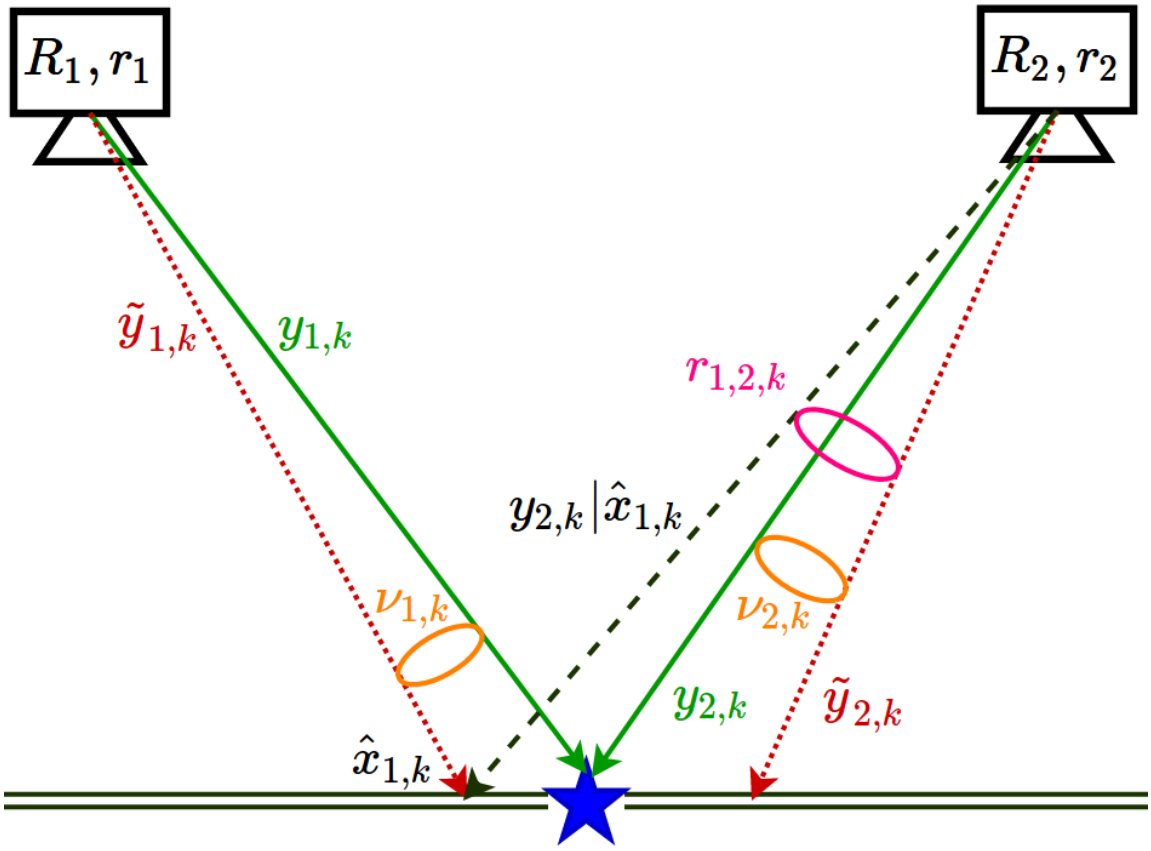
A way around this that minimizes the coupling is as follows. Given a single measurement $\tilde{\boldsymbol{y}}_{j,k}$, its known depth $\lambda_{j,k}$ from the rendered range image, and the known camera parameters $\boldsymbol{\theta}_{c_j}$, a scene-frame landmark position can be computed

$$\hat{\boldsymbol{x}}_{j,k} = R_j^T \left(R_{c/m}^T \pi^{-1} \left(\tilde{\boldsymbol{y}}_{j,k} \right) \lambda_{j,k} - \left[\boldsymbol{r}_{c/m} \right]_m \right) - \boldsymbol{r}_j \tag{5.8}$$

This landmark can then be projected into any other image and compared to the measurement to get a residual

$$\boldsymbol{r}_{i,j,k} \equiv \tilde{\boldsymbol{y}}_{i,k} - \pi \left(R_{c/m} \left(R_i \left(\boldsymbol{x}_{j,k} + \boldsymbol{r}_i \right) + \left[\boldsymbol{r}_{c/m} \right]_m \right) \right) \tag{5.9}$$

Figure 5.3: Illustration of pseudo-residual generation process for studying visual feature localization errors.



An illustration of this process with $i = 1$ and $j = 2$ is shown in Figure (5.3).

The error $\boldsymbol{r}_{i,j,k}$ clearly depends on both measurement errors, $\boldsymbol{\nu}_{i,k}$ and $\boldsymbol{\nu}_{j,k}$. This dependence can be analyzed as follows. First, let $\tilde{\mathbf{y}}_{j,k} = [\tilde{u}_j, \tilde{v}_j]^T$. Also define the forward and the inverse projection functions as

$$\pi([x,y,z]) = [fx/z + c_x,\ fy/z + c_y]^T \tag{5.10}$$

$$\pi^{-1}([u,v]) = [(u-c_x)/f,\ (v-c_y)/f,\ 1]^T \tag{5.11}$$

$$\tag{5.12}$$

Note that this is the model used in the rendering tool presented in Section 4.1. For a clearer analysis and without loss in generality, let $R_{c/m} = I$ and $\left[\boldsymbol{r}_{c/m}\right]_m = \mathbf{0}$. Then the estimated $\hat{\boldsymbol{x}}_{j,k}$ is

$$\hat{\boldsymbol{x}}_{j,k} = R_j^T \left(\pi^{-1}\left(\tilde{\mathbf{y}}_{j,k}\right)\lambda_{j,k}\right) - \boldsymbol{r}_j \tag{5.13}$$

$$= R_j^T \left(\pi^{-1}\left(\pi\left(R_j\boldsymbol{x}_k + \boldsymbol{r}_j\right) + \boldsymbol{\nu}_{j,k}\right)\lambda_{j,k}\right) - \boldsymbol{r}_j \tag{5.14}$$

$$= R_j^T \left(\pi^{-1}\left([\bar{u}_j + \nu_{u_j},\ \bar{v}_j + \nu_{v_j}]\right)\lambda_{j,k}\right) - \boldsymbol{r}_j \tag{5.15}$$

$$= R_j^T \left(\frac{1}{f}[\bar{u}_j + \nu_{u_j} - c_x,\ \bar{v}_j + \nu_{v_j} - c_y,\ f]\lambda_{j,k}\right) - \boldsymbol{r}_j \tag{5.16}$$

$$= \boldsymbol{x}_k + R_j^T \left(\frac{1}{f}[\nu_{u_j},\ \nu_{v_j},\ 0]\lambda_{j,k}\right) \tag{5.17}$$

where the first and second term of the last equation is the *true landmark position* and the error in the estimated landmark position respectively, under the standard model assumption.

Now the error term $\boldsymbol{r}_{i,j,k}$ is

$$\boldsymbol{r}_{i,j,k} = \tilde{\mathbf{y}}_{i,k} - \pi\left(\left(R_i\left(\hat{\boldsymbol{x}}_{j,k} + \boldsymbol{r}_i\right)\right)\right) \tag{5.18}$$

$$= \tilde{\mathbf{y}}_{i,k} - \pi\left(\left(R_i\left(\left[\boldsymbol{x}_k + R_j^T\left(\frac{1}{f}[\nu_{u_j},\ \nu_{v_j},\ 0]\lambda_{j,k}\right)\right] + \boldsymbol{r}_i\right)\right)\right) \tag{5.19}$$

There are two ways to proceed. The first is to linearize about $\boldsymbol{\nu}_{j,k} = \mathbf{0}$. In that case,

$$\boldsymbol{r}_{i,j,k} \approx \mathbf{y}_{i,k} - \frac{\partial \mathbf{y}_{i,k}}{\partial \boldsymbol{\nu}_{j,k}}\boldsymbol{\nu}_{j,k} \tag{5.20}$$

where the second term is equal to

$$\frac{\partial \mathbf{y}_{i,k}}{\partial \boldsymbol{\nu}_{j,k}}\boldsymbol{\nu}_{j,k} = \lambda_{j,k}\begin{bmatrix} 1/z & 0 & -x/z^2 \\ 0 & 1/z & -y/z^2 \end{bmatrix} R_i R_j^T \begin{bmatrix} \nu_{u_j} \\ \nu_{v_j} \\ 0 \end{bmatrix} \tag{5.21}$$

and $R_i \boldsymbol{x}_k + \boldsymbol{r}_i \equiv [x, y, z]^T$. If the rotation between images is negligible, then $R_i R_j^T \approx I$. If in addition, the translation is small between images, then $\lambda_{j,k} \approx z$. Then

$$\boldsymbol{r}_{i,j,k} = \boldsymbol{\nu}_{i,k} - [\nu_{u_j}, \nu_{v_j}]^T \tag{5.22}$$

On the other hand, we can make the simplifying assumptions ($R_i R_j^T \approx I$ and $\lambda_{j,k} \approx z$) without linearizing which yields

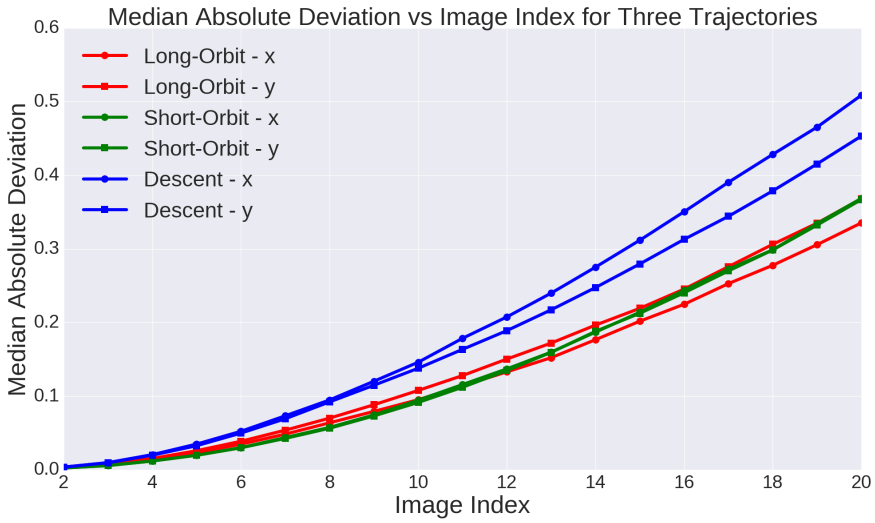
$$\boldsymbol{r}_{i,j,k} = \boldsymbol{\nu}_{i,k} - \boldsymbol{\nu}_{j,k} \tag{5.23}$$

These two results are clearly in agreement.

The pseudo-residual $\boldsymbol{r}_{i,j,k}$ can be computed between any pair of views, i and j, and for each landmark k. If the errors are truly IID with a constant landmark, then $\mathbb{E}\left\{\boldsymbol{r}_{i,j,k}\boldsymbol{r}_{i,j,k}^T\right\}$ should be constant (i.e. not depending on i, j, or k for $i \neq j$). The learning dataset described in Section 5.1 is used to see if this is the case. For all landmarks in all sets of images, the error $\boldsymbol{r}_{i,1,k}$ is computed for all i. By setting the middle index equal to one, we are always using the first image to generate a landmark position estimate. Instead of estimating the $\mathbb{E}\left\{\boldsymbol{r}_{i,j,k}\boldsymbol{r}_{i,j,k}^T\right\}$ as the sample covariance, the median-absolute-deviation (MAD) is computed for each of the diagonal components. As discussed in Section 2.3.6, the MAD is a robust estimate of scale. This is done for each image index i separately, using data aggregated over all landmarks in all sets for a given trajectory class. The results are given in Figure (5.4). Clearly, the MAD increases when plotted against image index. This makes sense intuitively but is in direct contradiction with the assumed model.

One possible model for the errors is a first-order auto-regressive model of the

Figure 5.4: Median-absolute-deviation for each component of $\boldsymbol{r}_{i,1,k}$ versus image index i for three different trajectory classes. Each class has 100 sets of 20 images with 200 landmarks per set.



form

$$\boldsymbol{v}_{1,j} = \boldsymbol{e}_{1,j} \tag{5.24}$$

$$\boldsymbol{v}_{i,j} = \alpha \boldsymbol{v}_{i-1,j} + \boldsymbol{e}_{i,j} \tag{5.25}$$

for $i \in \{2,3,\ldots,N_j\}$ for N_j images of landmark j. Furthermore, let the errors $\boldsymbol{e}_{i,j}$ be distributed as

$$\boldsymbol{e}_{i,j} \sim \frac{1}{\sigma_i} \mathrm{p}(\boldsymbol{e}/\sigma_i) \tag{5.26}$$

This implies that the stochastic process is non-stationary (i.e. the error scale can vary with the image index i). Note that the Cauchy distribution, Huber distribution, Gaussian distribution, and Laplace distribution all have this form.

We would like to estimate the parameters of the process given the pseudo-residuals that we can compute $\boldsymbol{r}_{i,1,k} \approx \boldsymbol{v}_{i,j} - \boldsymbol{v}_{1,j}$. For the sake of simplicity, each of

the two components (each direction in the image plane) of the residuals are treated separately. Define a parameter set consisting of

$$\{\alpha, \sigma_1, \ldots, \sigma_N, \nu_{1,1}, \ldots, \nu_{1,M}\}$$

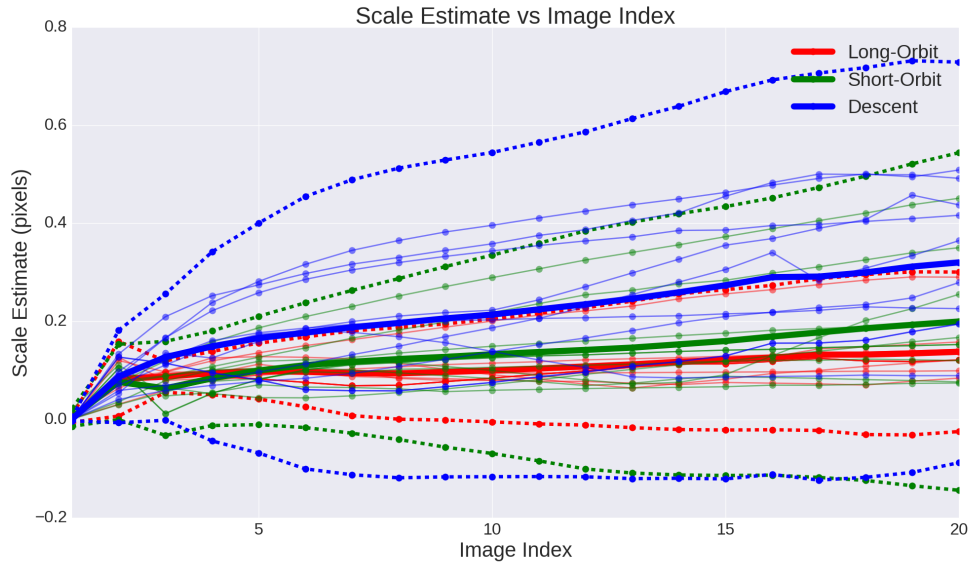
where $N = 20$ is the number of images per set and M is the total number of landmarks across all sets. The MLE of these parameters is defined as the arguments that minimize the cost

$$\begin{aligned} J = \sum_{k=1}^{M} \Bigg[& \log \sigma_1 - \log \left(\mathrm{p} \left(\frac{\nu_{1,k}}{\sigma_1} \right) \right) \\ & + \log \sigma_2 - \log \left(\mathrm{p} \left(\frac{(r_{2,1,k} - \nu_{1,k} - \alpha \nu_{1,k}}{\sigma_2} \right) \right) \\ & + \sum_{i=2}^{T-1} \log \sigma_{i+1} - \log \left(\mathrm{p} \left(\frac{r_{i+1,1,k} - \nu_{1,k} - \alpha (r_{i,1,k} - \nu_{1,k})}{\sigma_{i+1}} \right) \right) \Bigg] \end{aligned} \tag{5.27}$$

The $\nu_{1,k}$ parameters act as *fixed-effects* and are essentially nuisance parameters.

In order to perform the optimization, the Huber distribution is used which has the assumed form $\frac{1}{\sigma}\mathrm{p}(x/\sigma)$. The *SciPy* implementation of limited-memory Broyden-Fletcher-Goldfarb-Shanno algorithm (with bounds) is used to solve for the parameters [22]. The value of α is restricted to the interval $[-1, 1]$ and the error scales are given a lower bound of 0.001. Each trajectory class is treated separately and ten trials are performed per trajectory class with two sets of 20 images per trial. For the long-orbit trajectory, $\alpha = 1$ in nine out of ten cases ($\alpha = 0.93$ in one case). For the short-orbit trajectory, $\alpha = 1$ in nine out of ten cases ($\alpha = 0.73$ in one case). For the descent trajectory, $\alpha = 1$ in nine out of ten cases ($\alpha = 0.96$ in one case). In addition, the parameter identification routine failed to return a valid answer in two, one, and

Figure 5.5: Estimate of error scale versus index for the first-order autoregressive model of visual errors. The semi-transparent lines are the results for individual trials, the thick solid lines are the bootstrap means, and the dashed lines are the bootstrap means plus or minus three times the bootstrap standard deviations.



one cases out of ten for each of the long-orbit, short-orbit, and descent trajectory respectively. The resulting scale estimates are shown in Figure (5.5).

Note that it is important to assess the accuracy of these parameter estimates. Although this is an MLE for the assumed model, both the potential for systemic errors and the *Incidental Parameters Problem* prevent the standard variance estimate from applying. Instead, a bootstrap variance estimate can be obtained [40]. Given a number of estimates of a parameter $\hat{\theta}_i$, the bootstrap mean, μ_{bs}, and bootstrap variance, v_{bs}, are

$$\mu_{\text{bs}} = \frac{1}{N}\sum_{i=1}^{N}\hat{\theta}_i \tag{5.28}$$

$$v_{\text{bs}} = \frac{1}{N-1}\sum_{i=1}^{N}\left(\mu_{\text{bs}} - \hat{\theta}_i\right)^2 \tag{5.29}$$

The bootstrap mean and the square root of the bootstrap variance are used to obtain the heavy-weight and dashed lines of Figure (5.5) respectively.

An important conclusion can be drawn from this figure. Although Figure (5.4) seems to suggest that the error scales increase rapidly with image index, Figure (5.5) shows that the error scale is nearly constant. This implies that the apparent increase in scale is mostly explained by strong-positive error correlations (i.e. $\alpha \approx 1$).

One additional note should be made concerning Figure (5.5). The different trajectory classes have errors with different behaviors. For example, the error scale for the descent class appears to be larger than that of the other two classes. This implies that the probabilistic model will inherently depend on the nature of the motion which is unsurprising. Therefore error scale estimation may be necessary if the nature of the motion is unknown *a priori*.

5.3 Autoregressive Model

The autoregressive model is motivated by the empirical results of the previous section. Similar to the standard model, the expected measurement is

$$\hat{\boldsymbol{y}}_{i,j} = \boldsymbol{\pi}\left(KR_{c/m}\left(\hat{R}_i\left(\hat{\boldsymbol{x}}_j + \hat{\boldsymbol{r}}_i\right) + \left[\boldsymbol{r}_{c/m}\right]_m\right)\right) \tag{5.30}$$

Two types of edges are added to the optimization graph. The first is of the form

$$\boldsymbol{e}_{1,j} = \tilde{\boldsymbol{y}}_{1,j} - \hat{\boldsymbol{y}}_{1,j} \tag{5.31}$$

which relates the first camera pose to a landmark and is the same edge type used in the standard model. The second edge type is of the form

$$\boldsymbol{e}_{i,j} = (\tilde{\boldsymbol{y}}_{i,j} - \hat{\boldsymbol{y}}_{i,j}) - \alpha(\tilde{\boldsymbol{y}}_{i-1,j} - \hat{\boldsymbol{y}}_{i-1,j}) \tag{5.32}$$

for $i \in \{2,3,\ldots\}$. Note that if $\alpha = 0$, which implies zero error correlation, then the second edge type is the same as the first. Furthermore, the autoregressive model reduces to the standard model in that case.

The autoregressive model introduces additional coupling as compared to the standard model. In particular, the upper-left matrix is now block tri-diagonal. Nevertheless, the structure of the problem is still similar which leads to similar computation times. The structure of the Jacobian and Hessian for the autoregressive model are shown in Figure (5.6). The added coupling can be seen in the upper-left of the Jacobian matrix: each measurement (except the first on each feature track) involves two cameras instead of one which leads to the block tri-diagonal Hessian.

5.4 Landmark-Walk Model

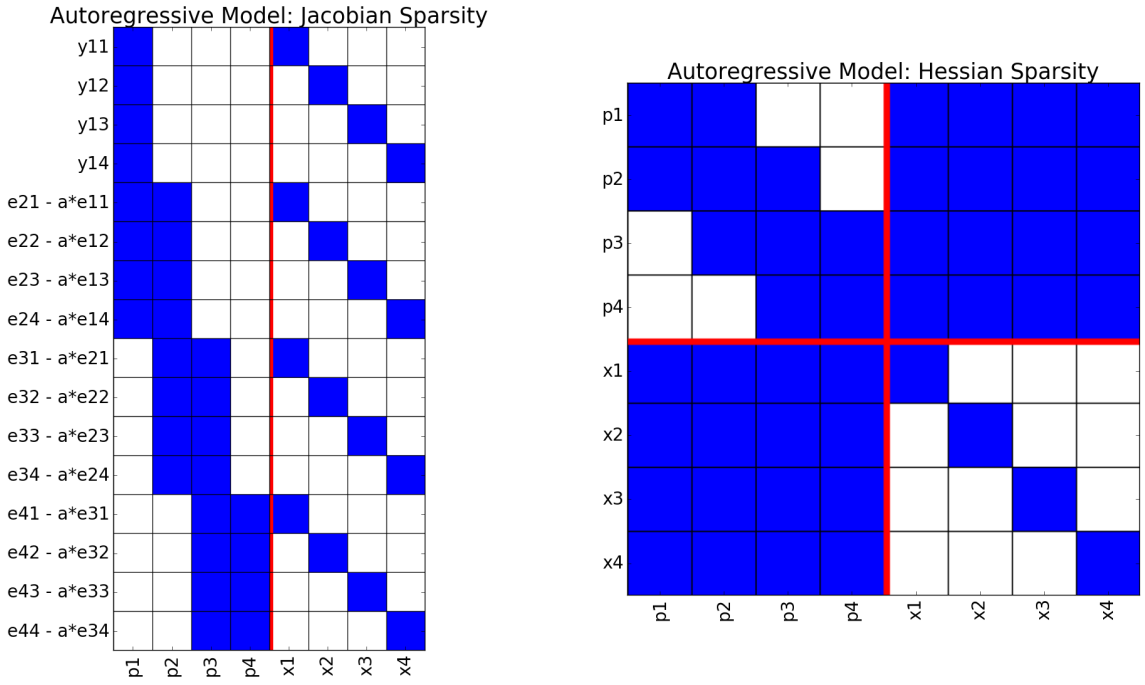
Following similar notation to the previous section, landmark j is located at location $\boldsymbol{x}_{i,j}$ with respect to an inertial reference frame at the time of image i. The measurement of this landmark at time i is

$$\tilde{\boldsymbol{y}}_{i,j} = \boldsymbol{\pi}\left(KR_{c/m}\left(R_i\left(\boldsymbol{x}_{ij} + \boldsymbol{r}_i\right) + \left[\boldsymbol{r}_{c/m}\right]_m\right)\right) + \boldsymbol{v}_{i,j} \tag{5.33}$$

The landmark location satisfies

$$\boldsymbol{x}_{i+1,j} = \boldsymbol{x}_{i,j} + \boldsymbol{w}_{i,j} \tag{5.34}$$

Figure 5.6: Block structure of the linearized system for the autoregressive model (four cameras, four landmarks): Jacobian (left) and Hessian (right). The blue and white blocks represent non-zero and zero terms respectively. The red lines partition the pose and landmark parameters.



where $\boldsymbol{w}_{i,j}$ is an IID zero-mean random variable.

Unlike the standard model, this model allows the landmark to move in 3D space. Let n_c be the number of images and n_ℓ be the number of landmarks. If all landmarks are seen in all images then the total number of parameters in this problem is $6n_c + 3n_c n_\ell$ as compared to $6n_c + 3n_\ell$ for the standard model. To look at the asymptotic behavior as n_c and n_ℓ go to infinity at a fixed ratio $\alpha = n_c/n_\ell$, then the number of parameters tends to $3\alpha n_c^2$ as compared to $(6+3\alpha)n_c$ for the standard model. With a naive implementation of a least-squares-type of optimizer, the random walk model has a complexity of $\mathcal{O}(n_c^6)$ compared to a complexity of $\mathcal{O}(n_c^3)$ for the

standard model. This uses the fact that the complexity of solving a linear system is cubic in the number of parameters.

Furthermore, the model as presented above requires the specification of *pseudo-measurement* constraints of the form of the form

$$\mathbb{E}\left\{\boldsymbol{x}_{i+1,j} - \boldsymbol{x}_{i,j}\right\} = \mathbf{0} \tag{5.35}$$

in addition to the visual feature measurements. This further complicates the graph structure.

A simple improvement can be made to avoid this and improve the model. To do so, one can **define** the landmark j at time i to be

$$\boldsymbol{x}_{i,j} = R_i^T \left(R_{c/m}^T \boldsymbol{\pi}^{-1}\left(\tilde{\boldsymbol{y}}_{i,j}\right) \lambda_{i,j} - \left[\boldsymbol{r}_{c/m}\right]_m \right) - \boldsymbol{r}_i \tag{5.36}$$

where the inverse projection function $\boldsymbol{\pi}^{-1}()$ converts the pixel location to a unit-vector of camera coordinates and $\lambda_{i,j}$ is an unknown camera-to-surface depth along the measured direction. This reduces the number of parameters of each landmark from three to one which reduces the total number of parameters by an amount $2n_c n_\ell$! It also eliminates the need to impose the constraints from visual features into the graph structure. Instead, all graph edges are of the form

$$\mathbb{E}\left\{\boldsymbol{x}_{i+1,j}(\boldsymbol{\theta}_{c_{i+1}}, \lambda_{i+1,j}) - \boldsymbol{x}_{i,j}(\boldsymbol{\theta}_{c_i}, \lambda_{i,j})\right\} = \mathbf{0} \tag{5.37}$$

Clearly the function $\boldsymbol{x}_{i,j}$ contains the visual measurement $\tilde{\boldsymbol{y}}_{i,j}$.

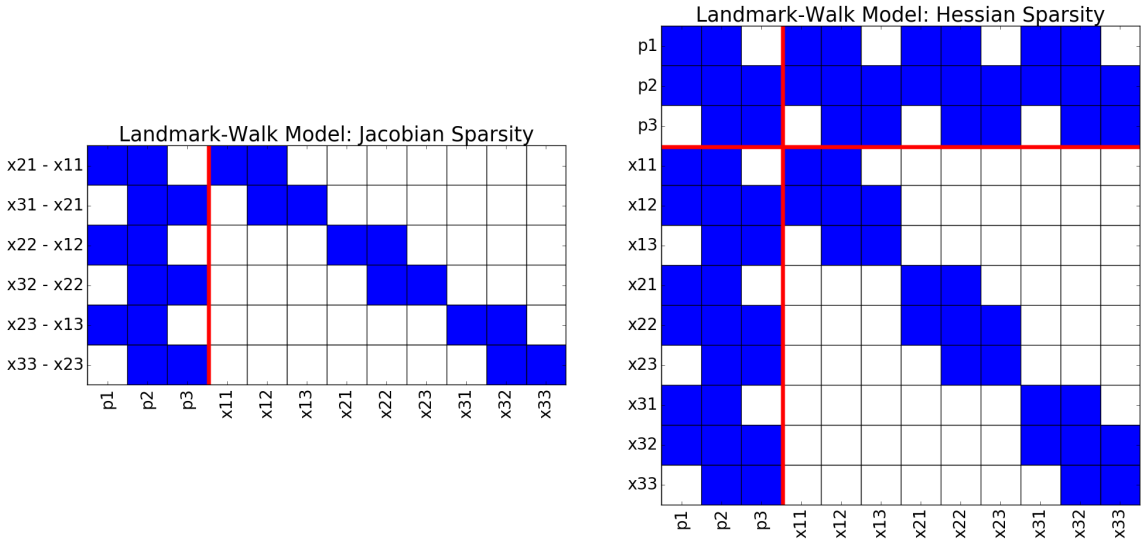
Similarly to the standard model, we can look at the structure of the Jacobian and Hessian which are shown in Figure (5.7). Note that the upper-left block of

the Hessian corresponding to camera parameters is block-tridiagonal. The lower-right block corresponding to landmark parameters is block-diagonal and of higher dimension: the sub-blocks contain a tri-diagonal structure. The primary off-diagonal blocks are densely populated. Therefore, it is always best to eliminate the landmark parameters first in this model. The solution to the camera parameters is order $\mathcal{O}((6n_c)^3)$ which is the same as the standard model. Then the landmark parameters can be solved for. The solution of each landmark parameter block $[\lambda_{1,j}, \lambda_{2,j}, \ldots, \lambda_{n_c,j}]$ is independent (over the index j). The n_ℓ landmark blocks each have dimension n_c and are tridiagonal. This means that the complexity of this step is $\mathcal{O}(n_\ell n_c)$ [105]. This is potentially much larger than in the standard model which has a complexity of $\mathcal{O}(n_\ell)$. The difference in complexity depends on the application (i.e. the relative number of landmarks and cameras).

One critical aspect of this model, which turns out to be a major drawback, is that each feature track (i.e. one landmark seen in all images) is treated as an individual measurement. This is in contrast to the previous two models that treat a single feature measurement (one landmark in one image) as an individual measurement. Therefore in this model, the robust cost function acts on an entire feature track instead of individual feature measurements. This means that a single bad feature measurement along a track can down weight in entire track leading to an unnecessary loss in efficiency.

This model is admittedly motivated by a heuristic concept. The motivation is

Figure 5.7: Block structure of the linearized system for the random walk model (three cameras, three landmarks): Jacobian (left) and Hessian (right). The blue and white blocks represent non-zero and zero terms respectively. The red lines partition the pose and landmark parameters.



that such a model can capture the serial correlations that appear in the data. This must be validated with testing.

5.5 Correlated Error Model

Consider the landmark measurement equation

$$\tilde{\boldsymbol{y}}_{i,j} = \boldsymbol{\pi}\left(KR_{c/m}\left(R_i\left(\boldsymbol{x}_j + \boldsymbol{r}_i\right) + \left[\boldsymbol{r}_{c/m}\right]_m\right)\right) + \boldsymbol{v}_{i,j} \tag{5.38}$$

One way to capture serial correlation is to let

$$\boldsymbol{v}_{i+1,j} = \begin{cases} \boldsymbol{v}_{i,j} + \boldsymbol{w}_{i,j} & i > 1 \\ \boldsymbol{w}_{1,j} & i = 1 \end{cases} \tag{5.39}$$

where the $\boldsymbol{w}_{i,j}$ are assumed to be independent. This model amounts to a random walk model in image space as opposed to a random walk in scene space presented in the previous section. Consider stacking the errors for a single landmark as $\bar{\boldsymbol{\nu}}_j \equiv [\boldsymbol{\nu}_{1,j}^T \boldsymbol{\nu}_{2,j}^T \dots \boldsymbol{\nu}_{n_c,j}^T]^T$. If the covariance of $\boldsymbol{w}_{i,j}$ is $\sigma_w^2 I_{2\times 2}$ and $\mathbb{E}\left\{\boldsymbol{w}_{i,j}\boldsymbol{w}_{k,l}\right\} = 0$ when $i \neq k$ or $j \neq l$, then the covariance of $\bar{\boldsymbol{\nu}}_j$ is

$$R_j = \sigma_w^2 \begin{bmatrix} 1 & 1 & 1 & \dots & 1 \\ 1 & 2 & 2 & \dots & 2 \\ 1 & 2 & 3 & \dots & 3 \\ 1 & 2 & 3 & \dots & 4 \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ 1 & 2 & 3 & \dots & n_c \end{bmatrix} \otimes I_{2\times 2} \tag{5.40}$$

where $\otimes$ is the Kronecker product. The inverse of this covariance matrix is needed for use in the estimator. Numerical examples were performed in an attempt to find a pattern in the inverse of matrices of the above form for different n_c. A clear pattern emerged which can be shown to be the correct inverse.

$$R_j^{-1} = \sigma_w^{-2} \begin{bmatrix} 2 & -1 & 0 & 0 & \dots & 0 \\ -1 & 2 & -1 & 0 & \dots & 0 \\ 0 & -1 & 2 & -1 & \dots & 0 \\ \vdots & \vdots & \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \dots & -1 & 2 & -1 \\ 0 & 0 & \dots & 0 & -1 & 1 \end{bmatrix} \otimes I_{2\times 2} \tag{5.41}$$

To prove that this is the inverse, simply multiply them together:

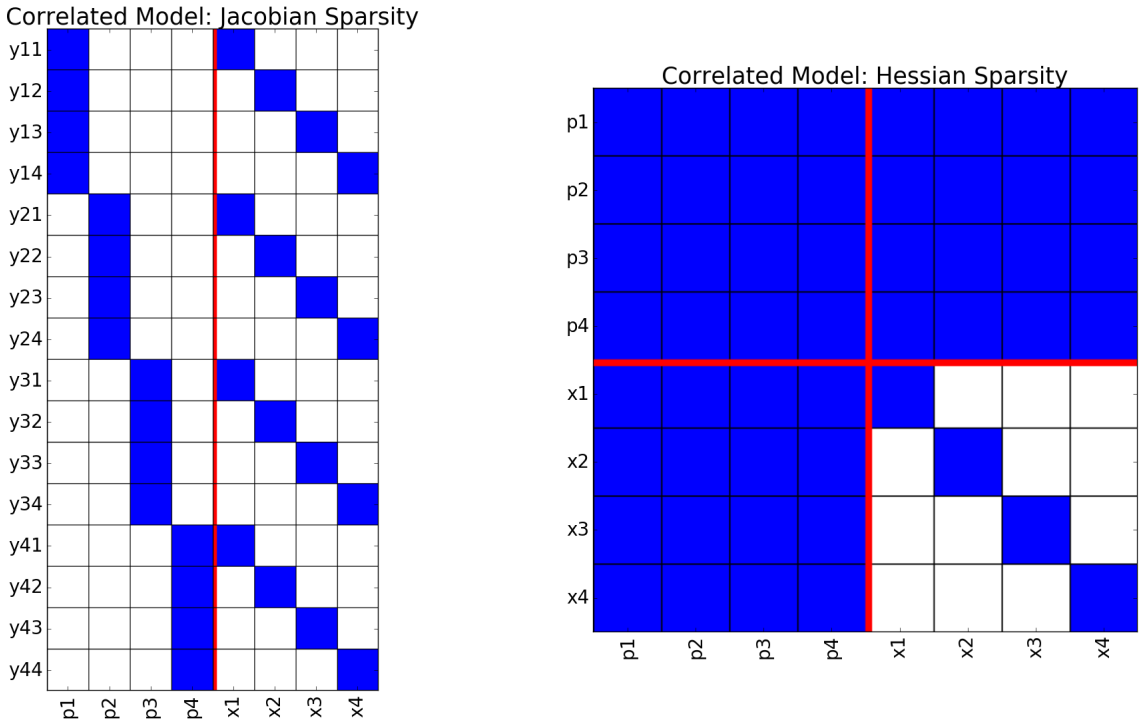
$$\begin{bmatrix} I & I & I & \dots & I \\ I & 2I & 2I & \dots & 2I \\ I & 2I & 3I & \dots & 3I \\ I & 2I & 3I & \dots & 4I \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ I & 2I & 3I & \dots & n_c I \end{bmatrix} \begin{bmatrix} 2I & -I & 0 & 0 & \dots & 0 & 0 & 0 \\ -I & 2I & -I & 0 & \dots & 0 & 0 & 0 \\ 0 & -I & 2I & -I & \dots & 0 & 0 & 0 \\ \vdots & \vdots & \vdots & \vdots & \ddots & \vdots & \vdots & \vdots \\ 0 & 0 & 0 & 0 & \dots & -I & 2I & -I \\ 0 & 0 & 0 & 0 & \dots & 0 & -I & I \end{bmatrix}$$

$$= \begin{bmatrix} 2I-I & -I+2I-I & -I+2I-I & \dots & -I+2I-I \\ 2I-2I & -I+4I-2I & 2I-4I+2I & \dots & 2I-4I+2I \\ 2I-2I & -I+4I-3I & -2I+6I-3I & \dots & 3I-6I+3I \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ 2I-2I & -I+4I-3I & -2I+6I-4I & \dots & -(n_c-1)I+(n_c-1)I \\ 2I-2I & -I+4I-3I & -2I+6I-4I & \dots & -(n_c-1)I+n_c I \end{bmatrix}$$

which is clearly a large identity matrix.

Similarly to the previous two sections, we can look at the structure of the Jacobian and Hessian which are shown in Figure (5.8). Note that the Jacobian is the same as in the standard model but the non-diagonal weight matrix induces a different structure on the Hessian. In particular, the upper-left becomes densely populated and the lower-right maintains the block diagonal structure.

Figure 5.8: Block structure of the linearized system for the correlated error model (four cameras, four landmarks): Jacobian (left) and Hessian (right). The blue and white blocks represent non-zero and zero terms respectively.



5.6 Alpha-Correlated Error Model

This model is a generalization of the one presented in the previous section. Consider additive pixel space errors of the form

$$\boldsymbol{v}_{i+1,j} = \begin{cases} \alpha \boldsymbol{v}_{i,j} + \boldsymbol{w}_{i,j} & i > 1 \\ \boldsymbol{w}_{1,j} & i = 1 \end{cases} \tag{5.42}$$

where the $\boldsymbol{w}_{i,j}$ are assumed to be independent and α is a scalar constant. When $\alpha = 1$, this is the original correlated error model. When $\alpha = 0$, this is the standard model. By tuning α in the range $[0, 1]$, one can attempt to capture the correlation

structure.

Note that the error can also be expressed as

$$\boldsymbol{v}_{i,j} = \sum_{k=1}^{i} \alpha^{i-k} \boldsymbol{w}_{k,j} \tag{5.43}$$

Then the (i,j) element of the 2×2 block covariance matrix is

$$\mathbb{E}\left\{ \boldsymbol{v}_{i,c} \boldsymbol{v}_{j,c}^T \right\} = \sigma^2 I_{2\times 2} \sum_{k=1}^{\min(i,j)} \alpha^{i+j-2k} \tag{5.44}$$

The structure of the matrix looks like

$$R_j = \sigma_w^2 \begin{bmatrix} 1 & \alpha & \dots & \alpha^{n_c-1} \\ \alpha & 1+\alpha^2 & \dots & \alpha^{n_c-2}+\alpha^{n_c} \\ \alpha^2 & \alpha+\alpha^3 & \dots & \alpha^{n_c-1}+\alpha^{n_c-3}+\alpha^{n_c-5} \\ \vdots & \vdots & \ddots & \vdots \\ \alpha^{n_c-1} & \alpha^{n_c-2}+\alpha^{n_c} & \dots & \sum_{k=1}^{n_c} \alpha^{2n_c-2k} \end{bmatrix} \otimes I_{2\times 2} \tag{5.45}$$

where $\otimes$ is the Kronecker product. The inverse does not have an obvious analytical solution in this case. This is not a major issue as the inverse only needs to be evaluated once because the covariance is independent of the unknown parameters. In addition, the inverse does not depend on the measurements and can therefore be computed offline *a priori* for each possible value of n_c (determined by the feature tracker and application).

Note that the structure and sparsity of the problem is exactly the same as in the case of the original correlated error model.

5.7 Alpha-Correlated Pixel-Walk Model

The model presented in this section is the most conceptually appealing of the above models. The downside to the α-correlated error model of the previous section is that each feature track (i.e. one landmark in many images) is treated as a single measurement with a single scalar weight. Therefore, if there is one very poor measurement in a feature track that is otherwise okay, then the entire track will be down weighted. This is precisely the issue with the correlated error model. It would be more appropriate to down weight the single poor measurement. However, the means to do this in a model with truly correlated errors is not clear.

A method to selectively down weight individual measurements with correlated errors was developed. In particular, consider the measurement model

$$\tilde{\boldsymbol{Y}} = \boldsymbol{h}(\boldsymbol{\theta}) + A\boldsymbol{w} \tag{5.46}$$

where $\tilde{\boldsymbol{Y}}$ is an m-dimensional measurement vector, $\boldsymbol{h}$ is a generic measurement function with unknown parameters of interest, $\boldsymbol{\theta}$, A is a constant matrix, and $\boldsymbol{w}$ is an m-dimensional vector of zero-mean IID random variables. For example, if $\boldsymbol{w}$ is an isotropic Gaussian with covariance $\sigma^2 I_{m\times m}$ and $\boldsymbol{h}(\boldsymbol{\theta}) = H\boldsymbol{\theta}$ then the MLE is given by

$$\hat{\boldsymbol{\theta}} = \text{argmin}_{\boldsymbol{\theta}} \left(\tilde{\boldsymbol{Y}} - \boldsymbol{h}(\boldsymbol{\theta})\right)^T (AA^T)^{-1} \left(\tilde{\boldsymbol{Y}} - \boldsymbol{h}(\boldsymbol{\theta})\right) \tag{5.47}$$

This optimization problem has the difficulty that each element of $\tilde{\boldsymbol{Y}}$ cannot be treated independently due to the correlated error: a single outlier element of $\boldsymbol{w}$ can corrupt

the entire solution.

Instead of the above solution, consider jointly estimating $\boldsymbol{\theta}$ and $\boldsymbol{w}$ by minimizing

$$\sum_i \rho(w_i) \tag{5.48}$$

subject to the condition

$$\tilde{\boldsymbol{Y}} = \boldsymbol{h}(\boldsymbol{\theta}) + A\boldsymbol{w} \tag{5.49}$$

For the particular example above, $\rho(w) = w^2$, but **any** $\rho()$ function can be used. It can be easily shown that this will give the same result for the above example. In particular, define $\boldsymbol{w}$ as

$$\boldsymbol{w} \equiv A^{-1}\left(\tilde{\boldsymbol{Y}} - \boldsymbol{h}(\boldsymbol{\theta})\right) \tag{5.50}$$

then substitute this into the cost function to obtain

$$\hat{\boldsymbol{\theta}} = \text{argmin}_{\boldsymbol{\theta}}\, \boldsymbol{w}^T\boldsymbol{w} \tag{5.51}$$

$$= \text{argmin}_{\boldsymbol{\theta}} \sum_i w_i^2 \tag{5.52}$$

$$= \text{argmin}_{\boldsymbol{\theta}} \left(\tilde{\boldsymbol{Y}} - \boldsymbol{h}(\boldsymbol{\theta})\right)^T (AA^T)^{-1} \left(\tilde{\boldsymbol{Y}} - \boldsymbol{h}(\boldsymbol{\theta})\right) \tag{5.53}$$

Clearly this is equivalent to the original cost function.

Implementing this is computationally expensive but a usually very good approximation can be used. In particular, the elements of $\boldsymbol{w}$ are added as vertices to the optimization graph. In addition, two types of edges are added. The first type models the cost $\rho(w_i)$. The second type approximates the constraint $\tilde{\boldsymbol{Y}}_i - \boldsymbol{h}(\boldsymbol{\theta}) - (A\boldsymbol{w})_i$ by adding a least-squares error with very high weight to the cost. The weight can be increased if the constraint is violated beyond a certain threshold. In practice, this

method has been successful.

The particular form of A has not yet been specified with the exception that it should be full rank. A natural choice is as follows. Consider images of a single landmark at many measurement times $\tilde{\boldsymbol{y}}_i$ where the index i denotes measurement time. Let the errors be

$$\tilde{\boldsymbol{y}}_1 = \mathrm{h}_1(\boldsymbol{\theta}) + \boldsymbol{w}_1 \tag{5.54}$$

$$\tilde{\boldsymbol{y}}_2 = \mathrm{h}_2(\boldsymbol{\theta}) + \alpha\boldsymbol{w}_1 + \boldsymbol{w}_2 \tag{5.55}$$

$$\tilde{\boldsymbol{y}}_3 = \mathrm{h}_3(\boldsymbol{\theta}) + \alpha^2\boldsymbol{w}_1 + \alpha\boldsymbol{w}_2 + \boldsymbol{w}_3 \tag{5.56}$$

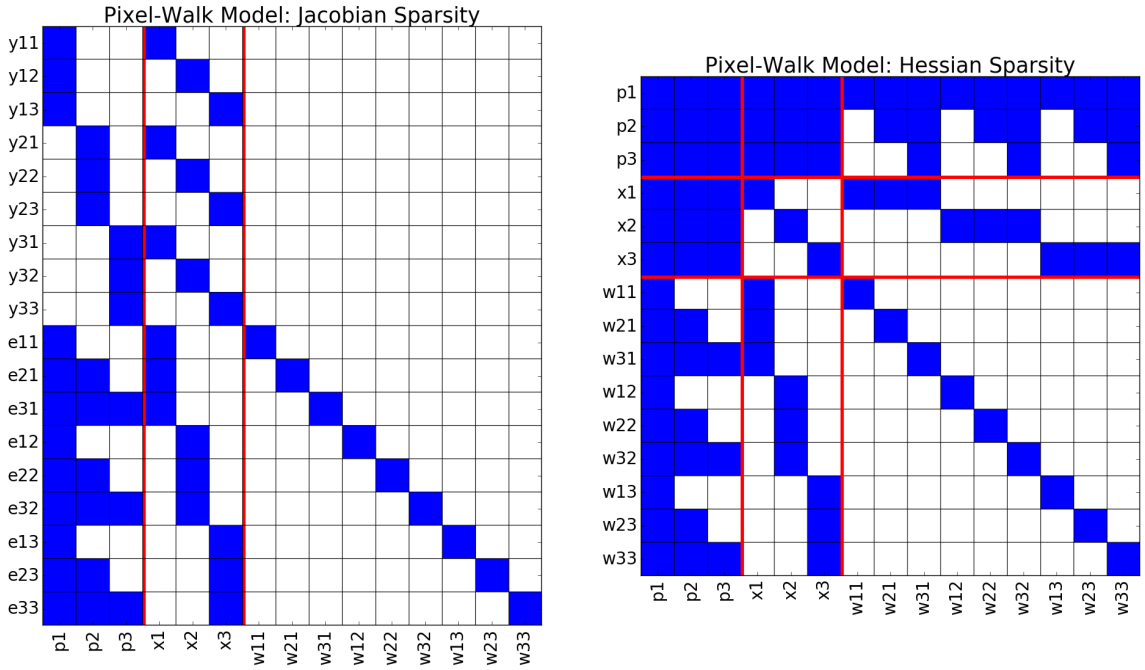
$$\vdots \tag{5.57}$$

Then the matrix A is

$$A = \begin{bmatrix} 1 & 0 & 0 & \dots 0 \\ \alpha & 1 & 0 & \dots \\ \alpha^2 & \alpha & 1 & \dots \\ \vdots & \vdots & \vdots & \ddots \quad \vdots \\ \alpha^{m-1} & \alpha^{m-2} & \alpha^{m-3} & \dots \quad 1 \end{bmatrix} \otimes I_{2\times 2} \tag{5.58}$$

Note that this reduces to the uncorrelated error model when $\alpha = 0$ and the amount of correlation increases as α increases. This particular form for $\alpha \in (0,1]$ is intuitively consistent with the way that the Lucas-Kanade feature tracker operates. If a large error is present in one frame due to some visual artifact, it will likely cause a very similar error in the next frame. The validity of this model and the performance of estimators designed around it are studied below.

Figure 5.9: Block structure of the linearized system for the correlated error model (three cameras, three landmarks): Jacobian (left) and Hessian (right). The blue and white blocks represent non-zero and zero terms respectively. The red lines partition the pose parameters, landmark parameters, and nuisance parameters.



Similar to the other models, we can look at the structure of the Jacobian and Hessian which are shown in Figure (5.9). This model is inherently more computationally expensive because of the much larger number of parameters. However, the Hessian block for the added nuisance parameters (lower-right) has a block diagonal structure which prevents the model from being prohibitively expensive.

5.8 Summary of Models

The previous sections presented numerous models for visual measurement errors. These models are summarized here. The standard model that is ubiquitous

in the literature assumes constant landmark positions and additive IID errors. Evidence invalidating either the independence assumption or the identically distributed assumption was found and motivated four novel models.

Each novel model attempts to capture error correlations in time. The landmark model does this by allowing the landmark to move and hence removes the constant landmark position assumption at the cost of adding a very large number of parameters to the model. The other three models remove the independence assumption.

The correlated error model removes the independence assumption in the most explicit way possible. A feature track is treated as an individual measurement as opposed to treating the feature location at each image time as individual measurements. The covariance of this stacked measurement captures the correlation by having non-zero off-diagonal blocks.

The pixel-walk model captures the error coupling by treating the image-space error as an α-correlated random-walk process. Note that this is the same idea behind the correlated error model. However, the parameterization is different which enables each feature to be treated as an individual measurement. This increases the flexibility of the robustification methods. In addition to the standard model parameters, the image-space walk-vector between each image time is treated as a parameter. The walk-vectors are assumed to be IID. Then the linear combinations of walk-vectors that are treated as the image-space additive errors on the feature locations have the desired coupling. This model requires hard constraints to be met which are

approximated through a penalty method.

The autoregressive model is very similar in nature to the pixel-walk model: it captures the same correlation and treats each feature as an individual measurement. However, this model does not need a penalty method to enforce constraints. In addition, it has the same relatively low-dimensional parameter space as the standard model. It does so by a neat parameterization of the individual edges of the optimization graph (i.e. the individual measurements).

5.9 Scale Estimation

If the scale is unknown *a priori* then it must be jointly estimated with the other parameters. A detailed analysis of the general nonlinear regression problem with unknown scale is given in Section 2.3.8. See Equation (2.169) for the associated joint covariance. The current section tailors the results of that section for the scale estimation problem for each of the models presented above.

Consider the MLE cost function

$$\mathrm{J}(\boldsymbol{\theta}, \sigma) = \sum_i -\log\left(\mathrm{p}(\tilde{\mathbf{y}}_i - \mathrm{h}_i(\boldsymbol{\theta}), \sigma\right) \tag{5.59}$$

where σ is a scale parameter (not necessarily standard-deviation). The method used to solve this problem is as follows. An initial guess of scale, $\sigma^{(0)}$, is used to obtain an initial estimate of $\boldsymbol{\theta}^{(0)}$. Then, the algorithms performs iterations in which the scale parameter is recomputed and used to perform one-step of the nonlinear parameter correction. This continues until convergence.

The nonlinear parameter correction is presented in Section 2.3.8. The scale parameter correction uses gradient descent. In particular,

$$\sigma^{(k+1)} = \sigma^{(k)} + \gamma \frac{\partial J}{\partial \sigma} \tag{5.60}$$

The derivatives depend on the designed cost and distribution. The two key distributions used in this section will be the Huber and Cauchy distribution.

The Huber cost,

$$\rho(x,\sigma) = \begin{cases} \frac{1}{2}x^2/\sigma^2 & |x|/\sigma \le k \\ k|x|/\sigma - \frac{1}{2}k^2 & |x|/\sigma > k \end{cases} \tag{5.61}$$

has the corresponding distribution

$$\mathrm{p}(x,\sigma) = \frac{1}{\sqrt{2\pi\sigma^2}} e^{-\rho(x,\sigma)} \tag{5.62}$$

Then the derivative needed for gradient descent is

$$\frac{\partial J}{\partial \sigma} = \sum_i \frac{1}{\sigma} - \begin{cases} x_i^2/\sigma^3 & |x_i|/\sigma \le k \\ k|x_i|/\sigma^2 & |x_i|/\sigma > k \end{cases} \tag{5.63}$$

All tests in this section use $k = 3.0$.

The Cauchy distribution is

$$\mathrm{p}(x,\sigma) = \frac{1}{\sqrt{\pi\sigma}\,(1 + x^2/\sigma^2)} \tag{5.64}$$

Note that σ^2 is **not** the variance of this distribution (the integral defining variance is undefined for the Cauchy distribution). The derivative for gradient descent is

$$\frac{\partial J}{\partial \sigma} = \sum_i \frac{1}{\sigma} - \frac{2x_i^2/\sigma^3}{1 + x_i^2/\sigma^2} \tag{5.65}$$

For the standard model, the scale estimate is applied to each component of each

feature measurement residual. For the landmark-walk model, the scale estimate is applied to each component of the landmark-walk parameter estimate. For the pixel-walk model, the scale estimate is applied to each component of the pixel-walk parameter estimate. The correlated error model is slightly more difficult to work with because the measurements are inherently vector-valued. Nevertheless, we can handle it in the same manner. The vector-valued Huber and Cauchy distributions are obtained from their scale counterparts by replacing the norms with

$$x_i^2/\sigma^2 \rightarrow \boldsymbol{x}_i^T R^{-1} \boldsymbol{x}_i \tag{5.66}$$

where $\boldsymbol{R}$ is the covariance matrix as defined in Equation (5.45). The scalar σ_w parameter in Equation (5.45) is the parameter being estimated in the scale step.

5.10 Sampling the Error Distribution

A key part of this dissertation is improving models used for estimation with visual measurements. To better inform the cost functions, it is desirable to obtain samples from the error distribution of the measurements. Ideally, a dataset containing both measurements and perfect truth for all parameters can be used to do this. While the rendering tool contains truth for the unknown pose trajectory and camera intrinsic parameters, the landmark parameters are not known. The landmark parameters are inherently difficult to model because they involve a complex interaction between the unknown scene, the imaging sensor, and the image processing algorithms (i.e. feature detection and tracking). The same reasons make it difficult

to obtain error samples from the models presented above. Methods to get approximate samples from the unknown distributions for each of the models is presented below.

5.10.1 Landmark-Walk Model

The landmark-walk model is the easiest to generate error samples from. The measurement, known depth, and known camera pose are used to compute a landmark position for each feature measurement as in Equation (5.8). The difference for this model is that this model **defines** the landmark position. The differences between the sequential landmark positions for a feature trajectory are direct samples of the error

$$\boldsymbol{w}_{i,j} = \boldsymbol{x}_{i+1,j} - \boldsymbol{x}_{i,j} \tag{5.67}$$

The one drawback of this model is that the errors should obviously depend on range to the surface. Therefore they will be scaled by the corresponding range measurement (or estimate) to get a *range-normalized error-scale*:

$$\boldsymbol{w}_{i,j} = (\boldsymbol{x}_{i+1,j} - \boldsymbol{x}_{i,j})/\lambda_{i,j} \tag{5.68}$$

Note that the range is known for the synthetic images which enables the samples of $\boldsymbol{w}_{i,j}$ to be obtained.

5.10.2 Other Error Models

For all the other error models, the only viable solution is to rely on the pseudo-residuals $\boldsymbol{r}_{i,j,k}$ defined in Equation (5.19) which are approximately:

$$\boldsymbol{r}_{i,j,k} = \boldsymbol{\nu}_{i,k} - \boldsymbol{\nu}_{j,k} \tag{5.69}$$

For the standard model, $\boldsymbol{r}_{i,1,k}$ for all i and k. Clearly the sum of two error samples is not a good representation of individual error samples. Under the standard model assumption, the errors are IID. The residual should be scaled as $\boldsymbol{r}_{i,1,k}/\sqrt{2}$ so that the statistics of the scaled pseudo-residuals will be equal to the statistics of the error samples **at the normal distribution**.

The α-parameterized models at $\alpha = 1$ have *underlying* errors equal to $\boldsymbol{r}_{i+1,i,k}$ which is very convenient. The errors are *underlying* in the sense that they are not the additive feature localization errors themselves but the errors that define the edges of the optimization landmark and are therefore the ones that must be well-modeled (since this is what the cost function must be designed around).

5.10.3 Experimental Results

An experiment was designed to study visuals errors under the models described above. The Temple 1 comet model and rendering tool as described in Section 4.1 were used to obtain 100 sets of 20 images for each of three trajectory classes, long-orbit, short-orbit, and descent trajectories, all of which are described in Section 5.1.

For each set of images, feature detection with the ORB detector is performed

on the first image. Features are tracked from frame-to-frame with the Lucas-Kanade tracker as described in Section 4.4. The resulting feature trajectories are then processed individually to obtain approximate error samples using the equations presented in the previous sections of this section. There are approximately 200,000 samples generated (20 images, 100 sets, 100 features per image). For each model, an attempt is made to fit several distributions. A Gaussian, Cauchy, Laplace, and generic Kernel Density Estimate (KDE) are fit to all samples. Additionally, a Gaussian is fit to the middle 99 % and middle 95 % of the data. The results for each model are discussed below. Histogram plots and CDF plots are shown for the descent trajectory class only. The results are tabulated for all trajectory classes.

The standard model error samples for the descent trajectory are shown in Figure (5.10) and Figure (5.11). Qualitatively, the Gaussian fit to all samples is very poor. This is because the true distribution has high kurtosis which inflates the fitted Gaussian variance. The result is an underestimate of the probability over the majority of the data. The Gaussian fit improves as the data is trimmed from 99 % to 95 %. The Cauchy and Laplace appear to give good fits. The Laplace fits the middle 90 % of the data very well but underestimates the tails. The Cauchy is a very heavy-tailed distribution and therefore better captures the behavior there. Note that the majority of the errors fall within one pixel of zero.

The landmark-walk model error samples for the descent trajectory are shown in Figure (5.13) and Figure (5.14). Note that these errors are normalized by the

Figure 5.10: Approximate samples from error distribution under the standard model. Samples genereated with 100 sets of 20 images along random descent trajectories.

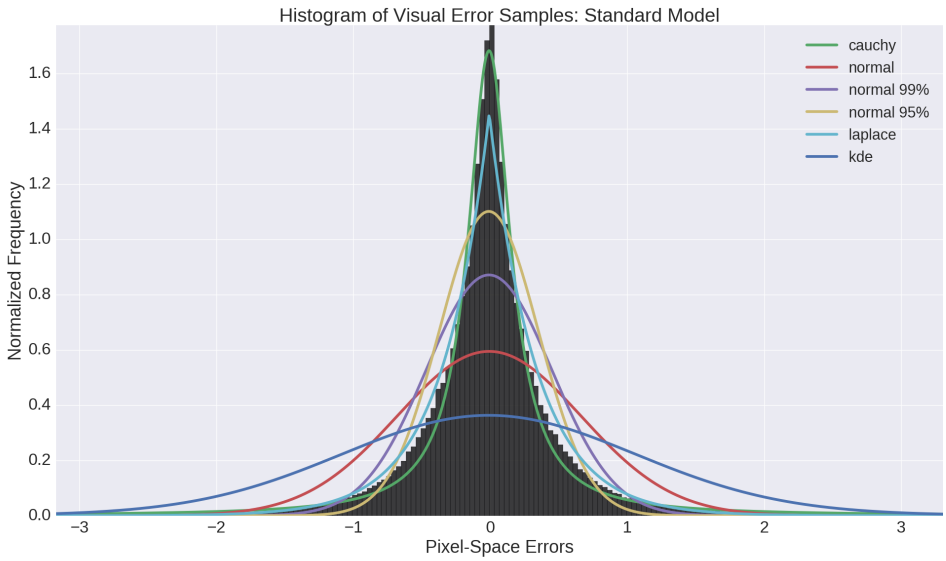


Figure 5.11: View of the tails of the samples in Figure (5.10)

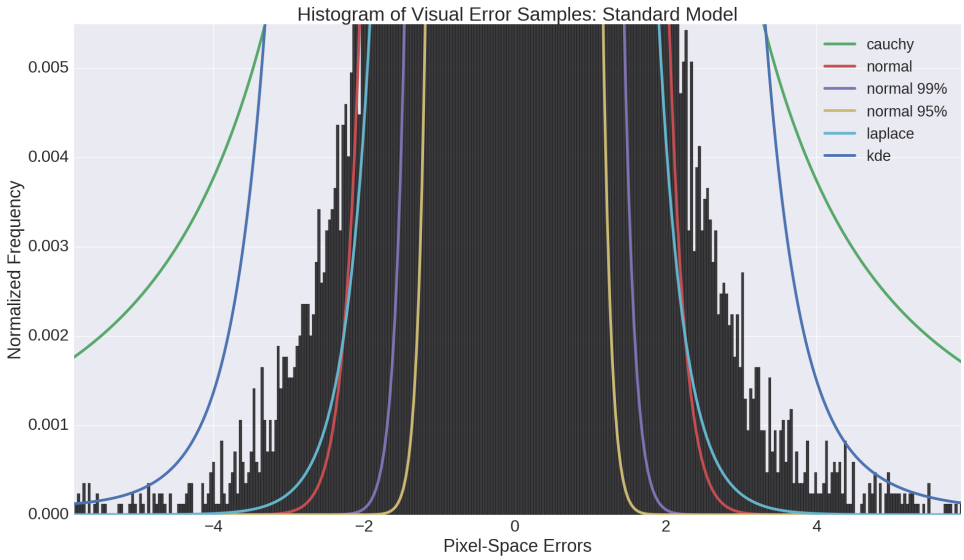
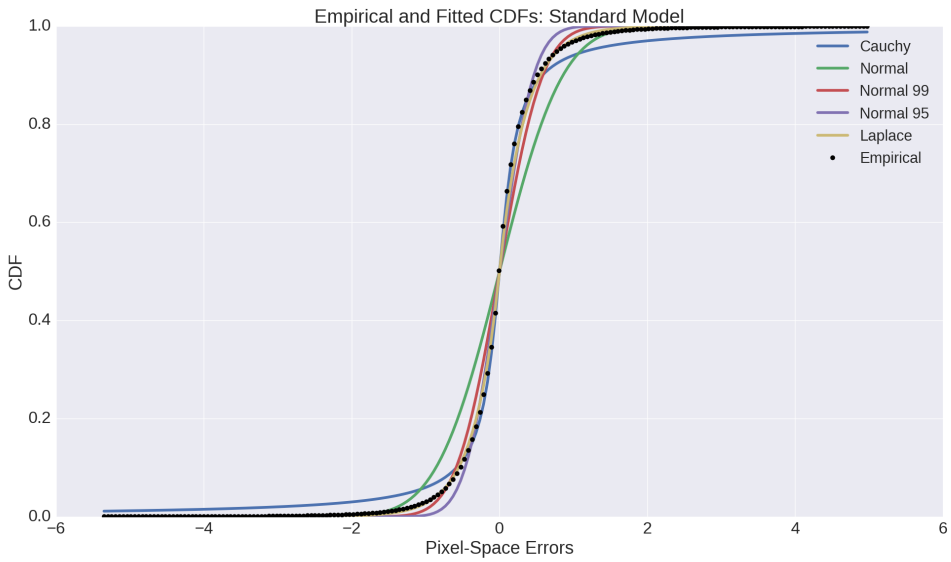


Figure 5.12: CDF of samples and distributions in Figure (5.10)



range to the surface along the given pixel ray. This model seems to have even higher kurtosis than the standard model. The extreme tails stretch the Gaussian fit. Again, the Cauchy distribution appears to give the best fit to the samples.

The correlated error model for the descent trajectory has samples with much lower kurtosis than the previous two models. The Gaussian fit to the middle 99% and 95% of the data set and the Laplace both appear to fit the samples well. Nevertheless, there are still a small number ($< 0.1\%$) of samples that appear as outliers under the Gaussian and Laplace. The Cauchy captures these but has a poorer fit to the middle 99.9 % of the data. One interesting point to notice is the much smaller scale of the errors in the correlated model as compared to the standard model: 0.15 pixels compared 1.0 pixels for the middle 95% of the data. This is because these errors

Figure 5.13: Approximate samples from error distribution under the landmark-walk model. Samples generated with 100 sets of 20 images along random descent trajectories.

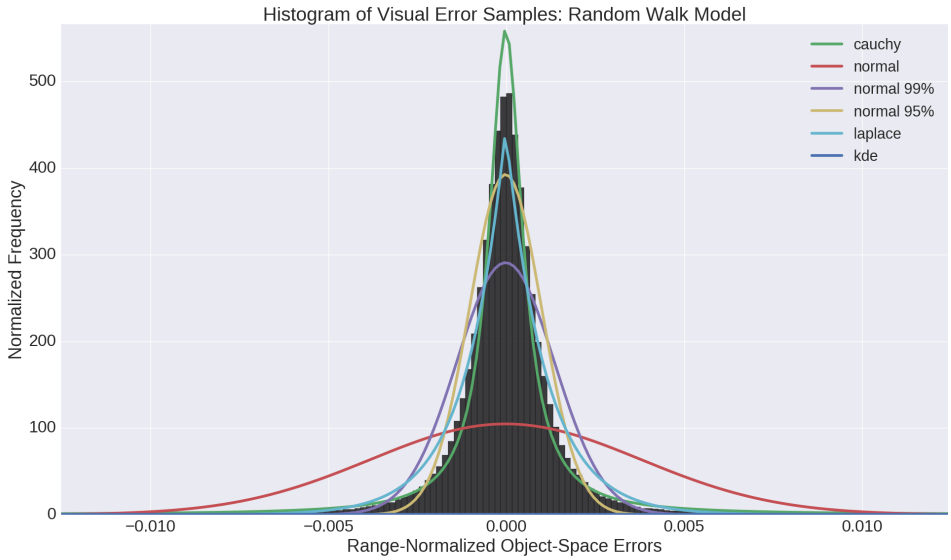


Figure 5.14: View of the tails of the samples in Figure (5.13)

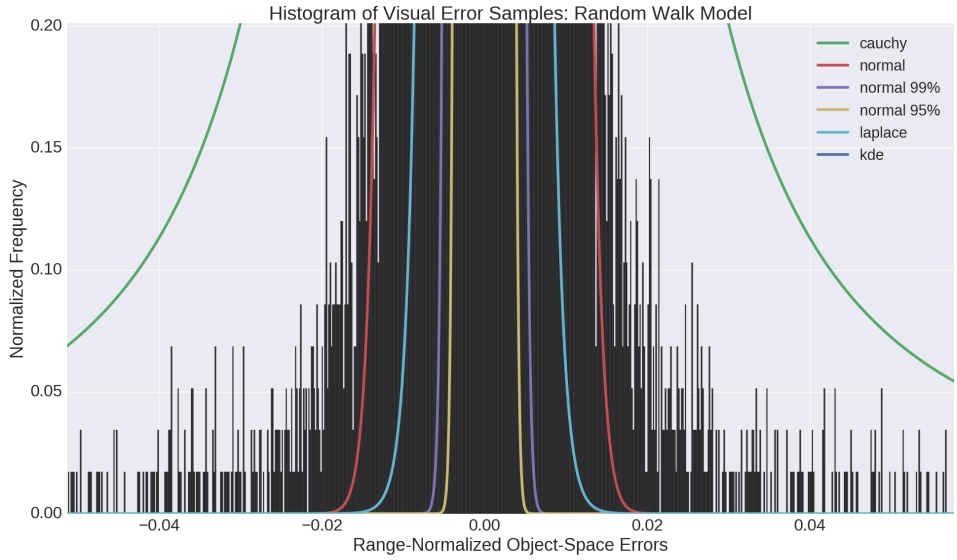
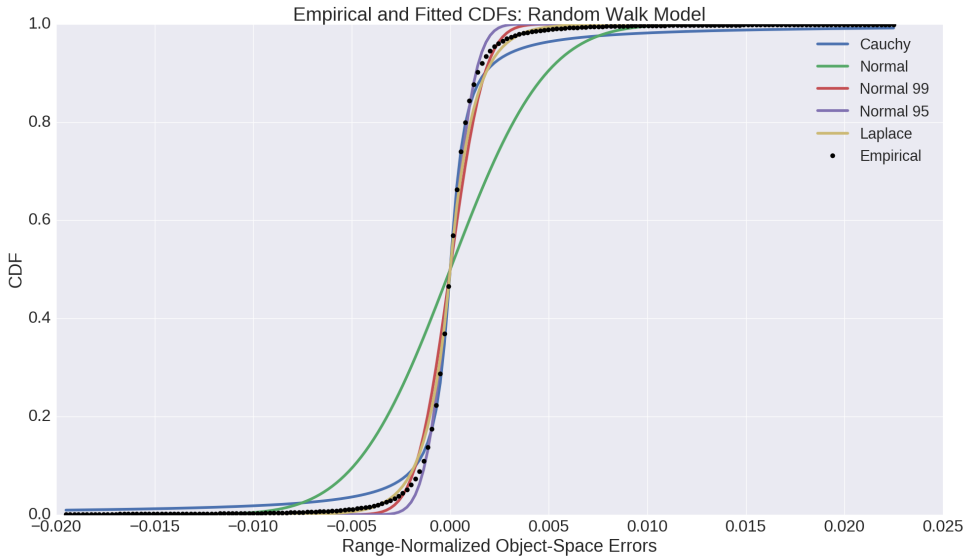


Figure 5.15: CDF of samples and distributions in Figure (5.13)



involve only a frame-to-frame error instead of frame-to-initial error as is seen in the standard model.

5.10.4 Tables of Distribution Fits

The results of the previous three subsections are quantitatively summarized in the tables below. For each distribution and model pair, the scale estimate and three goodness-of-fit statistics are reported for each trajectory class. The goodness-of-fit statistics are

1. Total Log-Likelihood: $\frac{1}{N}\sum_i^N \log \hat{f}(x_i)$
2. Kolmogrov Distance: $\max_x |\hat{F}(x) - F_n(x)|$
3. Cramer von-Mises Criterion: $\int_{-\infty}^{\infty} \left(\hat{F}(x) - F_n(x)\right)^2 dx$

where $\hat{F}(x)$ and $\hat{f}(x)$ is the estimated distribution and density respectively. A small

Figure 5.16: Approximate samples from error distribution under the correlated error model. Samples generated with 100 sets of 20 images along random descent trajectories.

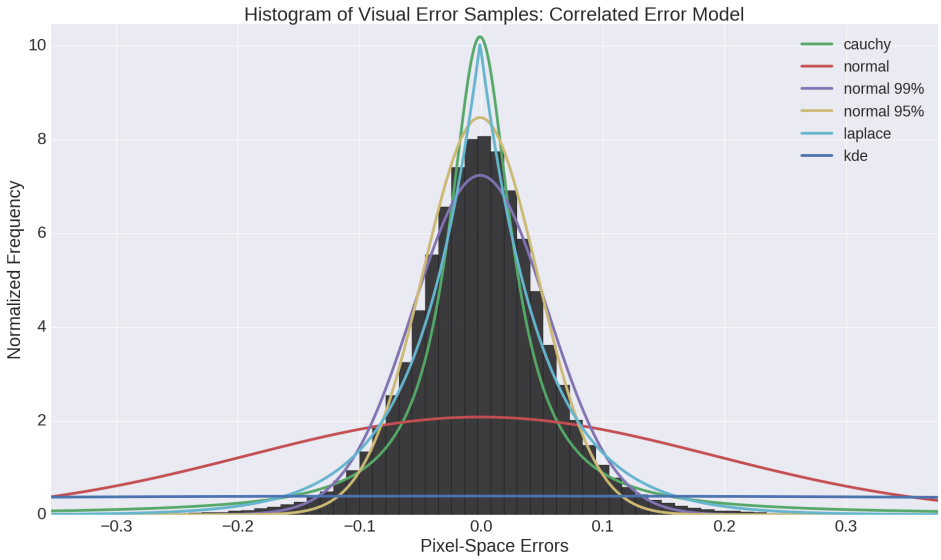


Figure 5.17: View of the tails of the samples in Figure (5.16)

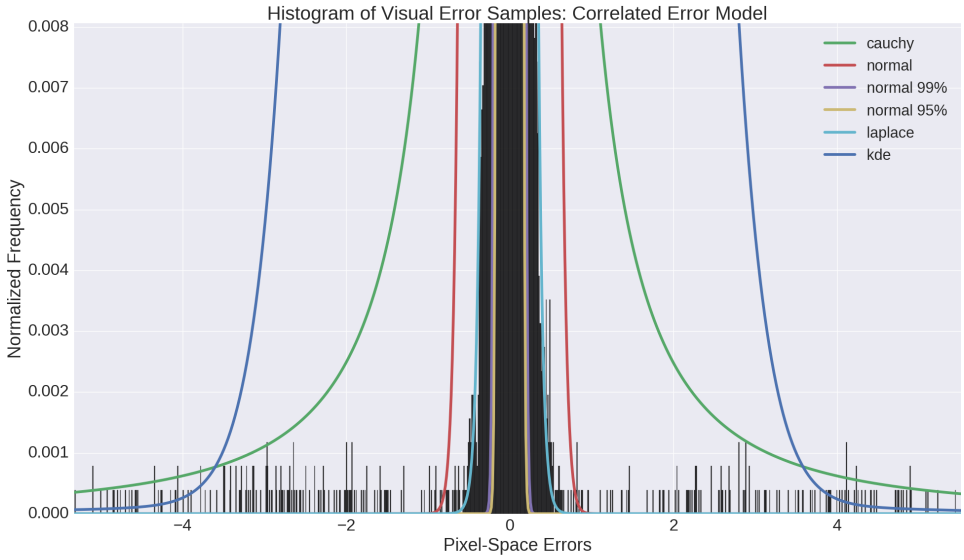
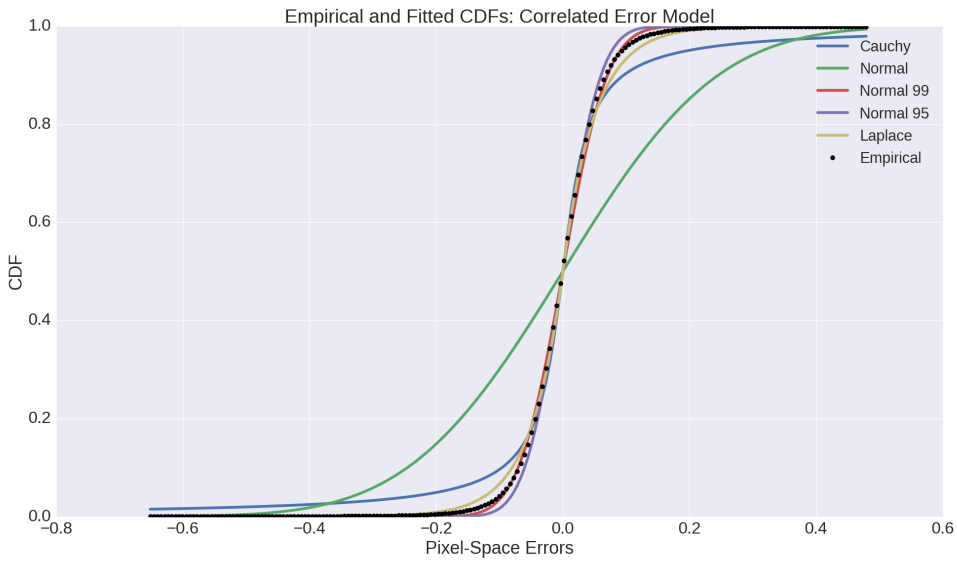


Figure 5.18: CDF of samples and distributions in Figure (5.16)



(large negative) total log-likelihood implies that the distribution is a bad fit. This statistic is especially sensitive to outlier-like samples. It is very important for the total log-likelihood to be high for the designed distribution. The Kolmogrov distance is useful for characterizing how much the samples disagree with a particular distribution at the worst location in the sample space. The smaller the statistic, the better the fit. The Cramer von-Mises criterion gives a global characterization of the fit (over the sample space) that is less sensitive to outliers than the total log-likelihood.

Table (5.1) summarizes the scale estimates and goodness-of-fits for each model-distribution pair for the descent trajectory class. Several conclusions can be drawn from this table. First, consider the normal distributions with scale estimated from the full and trimmed samples. The total log-likelihood is an extremely large negative

Distributions fitted to standard model errors.				
Distribution	Scale Est.	Likelihood	Kolmogrov	Cramer von-Mises
Normal	0.671	$-\infty$	0.150	0.0219
Normal 99 %	0.458	$-\infty$	0.091	0.00453
Normal 95 %	0.362	$-\infty$	0.058	0.00247
Cauchy	0.189	-1.268	0.030	0.00402
Laplace	0.344	-1.250	0.042	6.93e-4

Distributions fitted to random-walk model errors.				
Distribution	Scale Est.	Likelihood	Kolmogrov	Cramer von-Mises
Normal	0.0038	$-\infty$	0.259	3.60e-4
Normal 99 %	0.0014	$-\infty$	0.089	1.54e-5
Normal 95 %	0.0010	$-\infty$	0.036	5.72e-6
Cauchy	0.00057	5.20	0.033	1.29e-5
Laplace	0.0011	5.10	0.054	6.68e-6

Distributions fitted to correlated-error model errors.				
Distribution	Scale Est.	Likelihood	Kolmogrov	Cramer von-Mises
Normal	0.191	$-\infty$	0.268	0.0196
Normal 99 %	0.055	$-\infty$	0.026	5.64e-5
Normal 95 %	0.047	$-\infty$	0.030	1.30e-4
Cauchy	0.031	1.314	0.056	0.00129
Laplace	0.050	1.309	0.030	1.19e-4

Table 5.1: **Descent Trajectory Results**: Scale estimates for various distributions under the assumption of the three models. Different goodness of fit statistics are reported for each.

number in all cases. It is reported as $-\infty$ because it caused an overflow error with double-precision arithmetic. This large negative total log-likelihood is due to the fact that the samples simply cannot be described by a normal distribution. For any scale (i.e. variance) that captures the majority of the data, there are samples that appear nearly impossible. The zoomed in figures above illustrate this (i.e. Figure (5.11), Figure (5.14), and Figure (5.17)).

The error samples for the standard model seemed to be described well with

both the Cauchy and Laplace distributions. The lower Cramer von-Mises criterion for the Laplace implies that the Laplace gives a better global fit than the Cauchy. On the other hand, the lower Kolmogrov distance for the Cauchy implies that the Cauchy and true distribution lie in a smaller Kolmogrov neighborhood. Interestingly, the normal distribution fitted to the 5% trimmed samples has a lower Cramer von-Mises criterion than the Cauchy. From Figure (5.10), it appears this is due to the better fit for the middle of the data despite the poorer fit in the tails where the true distribution is already small.

The error samples for the landmark-walk model appear to be more amenable to a properly scaled normal distribution because of the low Cramer von-Mises criterion at both the 1% and 5% trimmed normal scale estimates. In fact, the normal with 5% trimmed scale estimate gives the lowest Cramer von-Mises criterion out of all distributions. This distribution has nearly the lowest Kolmogrov distance as well: the Cauchy is only 10 % better. Compared to the standard model, the distributions for error samples seem to fit the landmark-walk model better: the Cramer von-Mises and total log-likelihoods are more favorable and the Kolomogrov distances are similar.

The error samples for the correlated error model are qualitatively similar to those of the random-walk model which is unsurprising because the models are conceptually similar. Again, the normal with scale estimates from trimmed samples and the Laplace have favorable Cramer von-Mises criteria and Kolmogrov distances.

Distributions fitted to standard model errors.				
Distribution	Scale Est.	Likelihood	Kolmogrov	Cramer von-Mises
Normal	0.859	$-\infty$	0.179	0.0413
Normal 99 %	0.572	$-\infty$	0.118	0.0116
Normal 95 %	0.408	$-\infty$	0.067	0.00443
Cauchy	0.190	-0.713	0.019	0.00235
Laplace	0.406	-0.791	0.058	0.00274

Distributions fitted to random-walk model errors.				
Distribution	Scale Est.	Likelihood	Kolmogrov	Cramer von-Mises
Normal	0.0059	$-\infty$	0.295	6.89e-4
Normal 99 %	0.0019	$-\infty$	0.113	3.90e-5
Normal 95 %	0.0012	$-\infty$	0.035	8.26e-6
Cauchy	0.00067	5.02	0.034	1.33e-5
Laplace	0.0015	4.80	0.080	2.38e-5

Distributions fitted to correlated-error model errors.				
Distribution	Scale Est.	Likelihood	Kolmogrov	Cramer von-Mises
Normal	0.151	$-\infty$	0.189	0.00826
Normal 99 %	0.081	$-\infty$	0.064	5.90e-4
Normal 95 %	0.060	$-\infty$	0.030	2.42e-4
Cauchy	0.038	1.069	0.044	0.00112
Laplace	0.064	1.057	0.033	1.80e-4

Table 5.2: **Long-Orbit Trajectory Results**: Scale estimates for various distributions under the assumption of the three models. Different goodness of fit statistics are reported for each.

Interestingly, the Cauchy has a relatively high Kolmogrov distance as compared to other models.

The fitted distributions for the long-orbit and short-orbit trajectory classes exhibit similar behavior. For the standard model, the normal distribution appears to be a poor fit regardless of the scale estimate. Although the tighter scale estimates associated with the trimmed normal do fit better than the untrimmed scale estimate, the Kolmogrov and Cramer von-Mises scores are higher than they are for both the

Distributions fitted to standard model errors.				
Distribution	Scale Est.	Likelihood	Kolmogrov	Cramer von-Mises
Normal	0.661	$-\infty$	0.145	0.0201
Normal 99 %	0.513	$-\infty$	0.108	0.00811
Normal 95 %	0.387	$-\infty$	0.068	0.00376
Cauchy	0.183	-0.663	0.022	0.00255
Laplace	0.365	-0.685	0.050	0.00148

Distributions fitted to random-walk model errors.				
Distribution	Scale Est.	Likelihood	Kolmogrov	Cramer von-Mises
Normal	0.0036	$-\infty$	0.256	3.44e-4
Normal 99 %	0.0013	$-\infty$	0.076	1.23e-5
Normal 95 %	0.00096	$-\infty$	0.033	4.87e-6
Cauchy	0.00055	5.24	0.035	1.33e-5
Laplace	0.0011	5.16	0.043	5.23e-6

Distributions fitted to correlated-error model errors.				
Distribution	Scale Est.	Likelihood	Kolmogrov	Cramer von-Mises
Normal	0.082	1.087	0.099	0.00132
Normal 99 %	0.062	$-\infty$	0.044	1.85-4
Normal 95 %	0.050	$-\infty$	0.030	1.63e-4
Cauchy	0.032	1.261	0.048	0.00102
Laplace	0.049	1.318	0.016	3.17e-5

Table 5.3: **Short-Orbit Trajectory Results**: Scale estimates for various distributions under the assumption of the three models. Different goodness of fit statistics are reported for each.

Cauchy and Laplace distribution. Although the Laplace fits have higher Kolmogrov score than the Cauchy (0.058 vs. 0.019 and 0.050 vs. 0.022), the Cramer von-Mises score is similar or better than that of the Cauchy fit (0.00274 vs. 0.00235 and 0.00148 vs. 0.00255). This suggests that an $\mathscr{L}_1$ type estimator may be well-suited to these trajectories.

For the landmark-walk model, the normal distribution fit at the tighter error scale estimates of the trimmed normal appear to give a good fit. The 95% trimmed

normal estimate has the lowest Cramer von-Mises score and nearly the lowest Kolmogrov score of all distribution fits. This suggests a Huber cost function will be well-suited to this model and trajectory class.

For the $\alpha = 1$ correlated errors, both the 95% trimmed normal and the Laplace fit give low Kolmogrov and Cramer von-Mises scores. The Kolmogrov score is especially low for the Laplace distribution in the short-orbit trajectory class: the empirical CDF is always within 0.016 of the fitted Laplace CDF. This suggests an $\mathscr{L}_1$ type estimator will have high efficiency in this case.

The above discussion summarizes how well different distributions fit the error samples under different models. Intuitively, we would expect an MLE designed under a particular distribution will do *better* the more *accurately* the distribution fits the actual errors. The definition of a *better* estimate is up to the system designer and the definition of an *accurate* fit is not given by theory. The above results give a good starting place to explore different estimators. This is presented in the following sections.

5.11 Performance Analysis

The aim of this section is to study the performance of estimators designed around the models presented in the previous sections. The introduction to this section, given below, describes the estimators to be tested, the input to the estimators, and the performance metric computed on the output of the estimators.

The five structural models presented above were studied. To fully specify any model, a probabilistic component must be added. Based on the results of Section 5.10, the Laplace and Cauchy distributions were reasonable fits to the data. In addition, the normal distribution fits a large percentage, but not all, of the samples. This motivates two different probabilistic models and hence two different cost functions.

One cost function is the negative-log-likelihood of the multivariate Cauchy distribution:

$$\rho(x) = \log(1 + \frac{1}{\sigma^2} \boldsymbol{r}^T \Omega \boldsymbol{r}) \tag{5.70}$$

where $\boldsymbol{r}$ is the difference between the measured and expected value conditioned on the parameters. Also, Ω is the unit scale version of the inverse of the measurement covariance. This is simply identity for the all models except the correlated error model. For the correlated error model, it is defined in Equation (5.41) with $\sigma = 1$.

The second cost function is the Huber cost function:

$$\rho(x) = \begin{cases} \frac{1}{2}x^2 & |x| \leq k \\ k|x| - \frac{1}{2}k^2 & |x| > k \end{cases} \tag{5.71}$$

where $x = \sqrt{\boldsymbol{r}\Omega^T \boldsymbol{r}/\gamma^2}$. As $\gamma \to 0$ for fixed k, the Huber cost function approaches

the $\mathscr{L}_1$ norm which is the negative-log-likelihood of the Laplace distribution studied above. As $\gamma \to \infty$ for fixed k, the Huber cost function approaches the least squares cost which is the negative-log-likelihood of the Gaussian distribution. Because all measurements are treated with the same scale, the $\mathscr{L}_1$ cost and least squares cost are both scale invariant estimates. For finite non-zero γ values, the Huber cost depends on scale. For a fixed value of k, say $k = 1$, decreasing γ will increase the number of measurements in the *outlier-region* (i.e. linear region) and hence increase the number of measurements being down-weighted relative to a least squares cost. If the errors were truly Gaussian with covariance $\sigma^2\Omega^{-1}$, then $|x|^2\gamma^2/\sigma^2$ is a χ^2-variable with degrees-of-freedom equal to the dimension of $\boldsymbol{r}$. If one would like to let a proportion p of the measurements be down weighted, then one must use the χ^2 CDF to determine what the ratio γ^2/σ^2 should be. For example, for 2D measurements with $p = 0.05$ and $k = 1$, the inverse CDF at $1 - 0.025$ is 5.99 which is the critical value for $|x|^2\gamma^2/\sigma^2$. Therefore when $k = 1$, one should set $\gamma^2/\sigma^2 = 5.99$ or $\gamma^2 = 5.99\sigma^2$ for a given σ^2. In this way, we can use the results of the previous section for the normal distributions fitted to the trimmed dataset.

The measurement input to the estimator was generated from the computer vision pipeline developed for this dissertation and discussed in Section 4.4. The outputs of the pipeline are a data structure containing the feature measurements, an initial guess of landmark positions, and an initial guess of camera pose relative to the first image. This is all the necessary inputs for the estimator. However, there is

one issue. The combination of the measurement function and cost function lead to non-convex optimization problems in all cases. Therefore the estimator output may depend on the input initial guess. This will be dealt with explicitly below.

Finally, it is desirable to have performance metrics that can be computed on the estimator output, namely the camera trajectory. Several performance metrics are possible and the best choice depends on the given application. The performance metric used here is the root-mean-square (RMS) translational and angular error computed for each individual trajectory. This was selected simply because it is widely used and is easily interpreted.

Before concluding the introduction to this section, a very important note is in order. Because the focus of this section is to study how well different estimators perform on visual measurements, *no other (non-visual) measurement type is included* as this would conflate the visual measurement cost function with the other measurement cost function. The downside to this is that the metric scale of the trajectory and scene is unobservable as described in Section 3.5. This can cause numerical issues in the optimization routine. In addition, the metric scale is needed in order to compare to the ground truth trajectory (which is known exactly for this dataset). To overcome this, a pseudo-measurement was added to fix the distance between the first and last camera. The value of this distance is based on the *a priori* estimate of camera pose generated during feature tracking as described in Section 4.4. This resolves the numerical issue. To then make a fair comparison to the ground truth, two

options are possible. The first is to use the ratio between the true and *a priori* first-to-last camera distance to scale the trajectory. The second is to numerically solve for the optimal trajectory scaling that would minimize an error metric computed between the true and estimated trajectory. The latter gives a fair assessment because any metric scaling of the trajectory (and landmarks) does not change the likelihood function (or robust cost). It turns out that both options give nearly identical results. The latter option was used in all results below.

The rest of this section is organized as follows:

1. A detailed explanation of the experimental design is given. The design specifies a set of three high-level tests to be performed.
2. The first high-level test results are presented. This test studies the performance of a large number of estimators on a relatively small number of datasets. In these tests, the initial guess of pose is taken as the output of the feature tracking pipeline.
3. The second high-level test results are presented. This test studies the sensitivity of the estimators' output to the initial guess. This is performed for a large number of estimators on a small number of samples.
4. The results of the first two high-level tests are used to select a subset of estimators for further testing. A discussion is given to justify this choice.
5. The third high-level test results are presented. A smaller number of estimators are used in this test but on a larger number of samples.

6. A discussion of all results is given.

5.11.1 Experimental Design

This study can be broken up into three tests. The aim of the first test is to gain a basic understanding of the performance of the estimators derived from the various probabilistic models and how their internal tuning parameters impact their performance. The aim of the second test is to study the sensitivity of the output of each estimator to the initial guess. The first two tests are used to select a subset of estimators for the third study which is a more rigorous statistical analysis of performance. Difficulties in designing the tests are discussed below and then details of the tests are given.

The first difficulty is that there is a large number of estimators to be studied. In particular, there are five different models. Two of the models have one parameter, namely scale. Three of the models have two parameters, namely scale and correlation. Furthermore, two different cost functions, to be discussed below, need to be tested at each parameter setting. Even if scale and correlation are only tested at five different values, this leads to ten different estimators for each of the first two models and fifty estimators for each of the other three models. In addition, the simultaneous scale estimator for each model adds another two estimators for each of the first two models and ten estimators for each of the other three models: a total of 204 estimators!

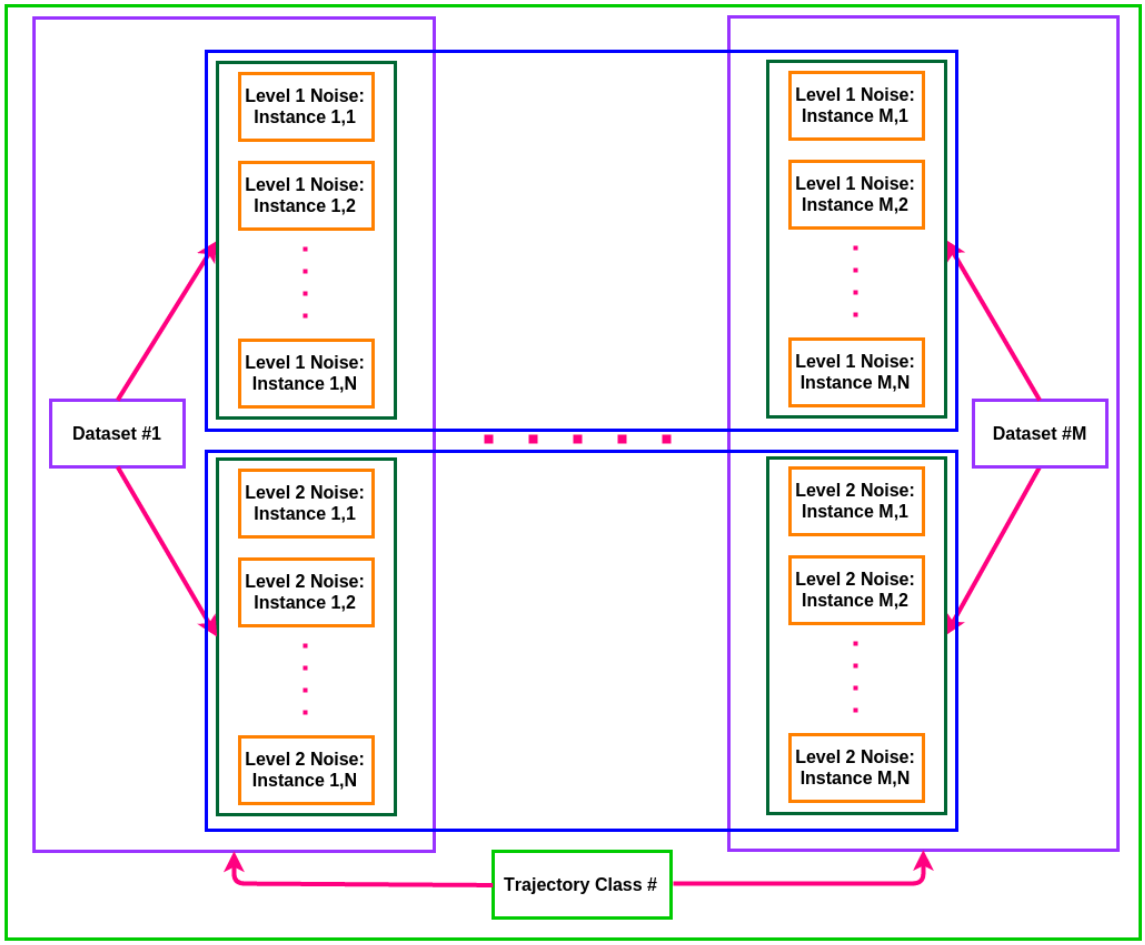
The second difficulty is that each estimator must be applied to many datasets in order to assess the performance. This is because variability in the estimator output

(i.e. the performance metric) is due to both the type of estimator and the particular dataset: the first cause is of primary interest and the second cause is a confounding factor. Furthermore, there are three aspects of a dataset that cause variability in the estimator output. The first is the set of visual feature measurements and the second is the pose estimate that the feature tracker generates which is used as the initial guess for the estimator. The third difficulty is that the performance of each estimator is expected to also depend on which trajectory class is used. In summary, there are three confounding factors: the particular dataset, the initial guess associated with the dataset, and the trajectory of the dataset. The high-variation in performance due to confounding factors implies a large number of datasets need to be tested to asses the performance of an estimator. Coupled with the fact that so many estimators need to be tested, this represents a large computational burden.

The difficulties caused by confounding factors can be handled by an intelligent experimental design. First we need to specify the *experimental units*.

An experimental unit is defined in this study as a particular dataset (i.e. a set of images with associated feature measurements) and the corresponding initial guess. The *treatments* are the individual estimators. The response of each experimental unit to each treatment is the RMS translational and angular error of the estimator output on the dataset. In order to increase the power of statistical tests, the variability in experimental unit response to different treatments can be effectively reduced by *statistical blocking*, a commonly used tool in experimental design that groups similar

Figure 5.19: Statistical blocking implemented in this study.



experimental units together. For the purposes of this study, a three-tiered approach to blocking has been taken. This approach is illustrated in Figure (5.19).

Figure (5.19) shows the three layers of statistical blocking used. The individual experimental units are shown in the orange boxes. At the highest level, experimental units are blocked based on which trajectory class they belong to (shown in light-green). At the intermediate level, the experimental units are blocked based on the error level of the initial guess (shown in blue). At the lowest level, the experimental

units are blocked based on the dataset they are derived from (shown in dark-green). Estimators are applied to each experimental unit (shown in orange). To compare two particular estimators at a given noise level and trajectory class, the differences in performance metric for each experimental unit (associated with that block) are computed and then used in a Wilson Signed Rank Test (WSRT).

The WSRT is a paired difference test that compares the difference in response among two treatments for each experimental unit. The WSRT is nonparametric in the sense that it makes no assumption about the distribution of the individual samples. The nonparametric property makes it well-suited to the application at hand.

With the above framework in place, the three primary tests can be discussed. The first test seeks to understand the performance of all structural models at a variety of parameters. For this test, ten datasets are used from each of the three trajectory classes. For each dataset, the initial guess is chosen as the vision pipeline output pose estimate. The estimators tested are the following. The standard model and landmark-walk model are used with scale $\sigma \in \{0.01, 0.05, 0.1, 0.5, 1.0\}$ for both the Huber and Cauchy cost. The autoregressive, pixel-walk, and correlated error model, are used with scale $\sigma \in \{0.01, 0.05, 0.1, 0.5, 1.0\}$ for both the Huber and Cauchy cost. It was decided to use $\alpha = 1.0$ in all tests. When $\alpha = 0.0$, the α-parameterized models are mathematically equivalent to the standard model. The choice of $\beta = 1.0$ is motivated by the mechanics of the Lucas-Kanade (and similar frame-to-frame

trackers) which is used in the pipeline and by the testing results of Section 5.2.1. The results of this first test are given in Section 5.11.2.

One issue with this test has to do with the quality of the initial guess. Note that the pipeline uses depth measurements in addition to feature tracks. As a result, the accuracy of the resulting initial guess will be higher than if feature measurements alone had been used. *Therefore, it is possible that for some estimators, the initial guess is better than the global-optimum for the estimator.* This partially motivates the second and third tests.

The second test uses the same set of estimators as the first test. In the second test, well-controlled random error is added to the ground truth to be used as the initial guess for the estimators. Two levels of noise are added. The first is zero-noise (i.e. the initial guess is the truth). The second is zero-mean Gaussian error with $\sigma = 0.05$ distance units and $\sigma = 0.01$ radians added to each component of position and attitude respectively. Five instances of the second noise level are added per dataset. The output of the five instances of the second noise level are compared to the noise free case. Note that this test treats departs from the blocking scheme used in the other tests. In particular, the purple-colored block in Figure (5.19) (i.e. a dataset with many initial guesses) is treated as an experimental unit. The results are given in Section 5.11.3.

The results of the first and second test are used to select a subset of estimators for the third test. The full blocking scheme of Figure (5.19) is used in the third

test. A large number of samples is then used which enables the use of the WSRT to make statistically significant statements about performance. The results are given in Section 5.11.5.

5.11.2 Basic Performance Study

This section presents results from the first of three tests. The purpose of this study is to obtain a basic understanding of the performance of the various estimators and identify viable candidates for further testing. The estimators discussed in the previous section, designed around the five structural models, are used to estimate the pose and landmark parameters for ten datasets from each of the three trajectory-classes. In all cases, the pose estimate output from the vision pipeline is used as the initial guess. Two sets of plots are generated for the results. The first gives the median RMS position and angle error over the ten datasets (i.e. one position value and one angle value per estimator per trajectory). The median is plotted against the scale of the estimator. The results for each trajectory class are given on separate plots and are collected in Figure (5.20). The RMS error of each of the ten individual datasets are given in Figure (5.21). These results are discussed below. Inter-model trends are discussed first followed by intra-model trends.

The standard model seems to work best with the $\sigma = 0.01$ Cauchy cost in the long-orbit and short-orbit cases and with the $\sigma = 0.1$ Cauchy cost in the descent case. This is surprising as the analysis in Section 5.10 suggested that either a Huber scale of $\sigma = 0.4$ or a Cauchy scale of $\sigma = 0.2$ was a good fit for each of the motion

Figure 5.20: Median estimator performance over ten datasets from each trajectory class versus estimator scale parameter. Initial pose estimate from vision pipeline.

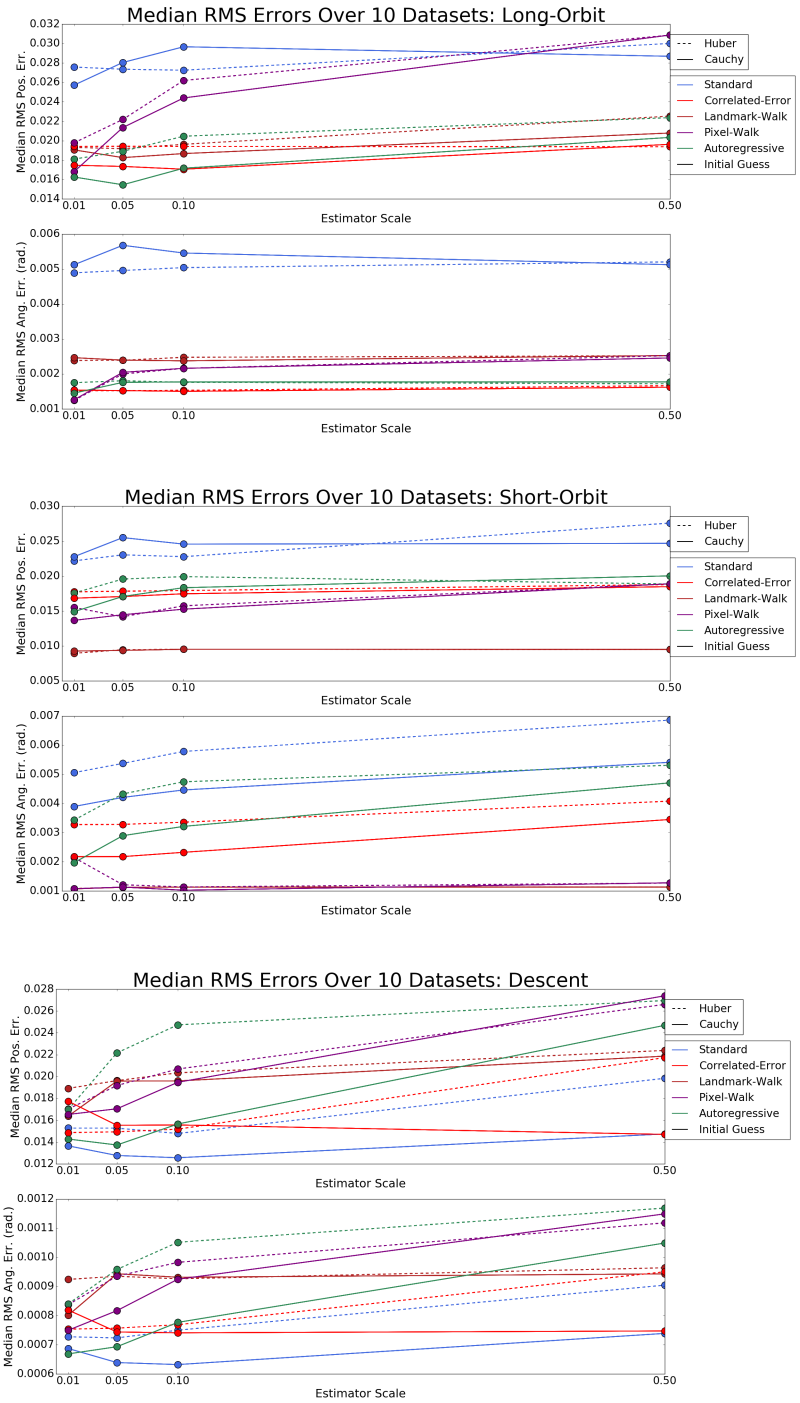
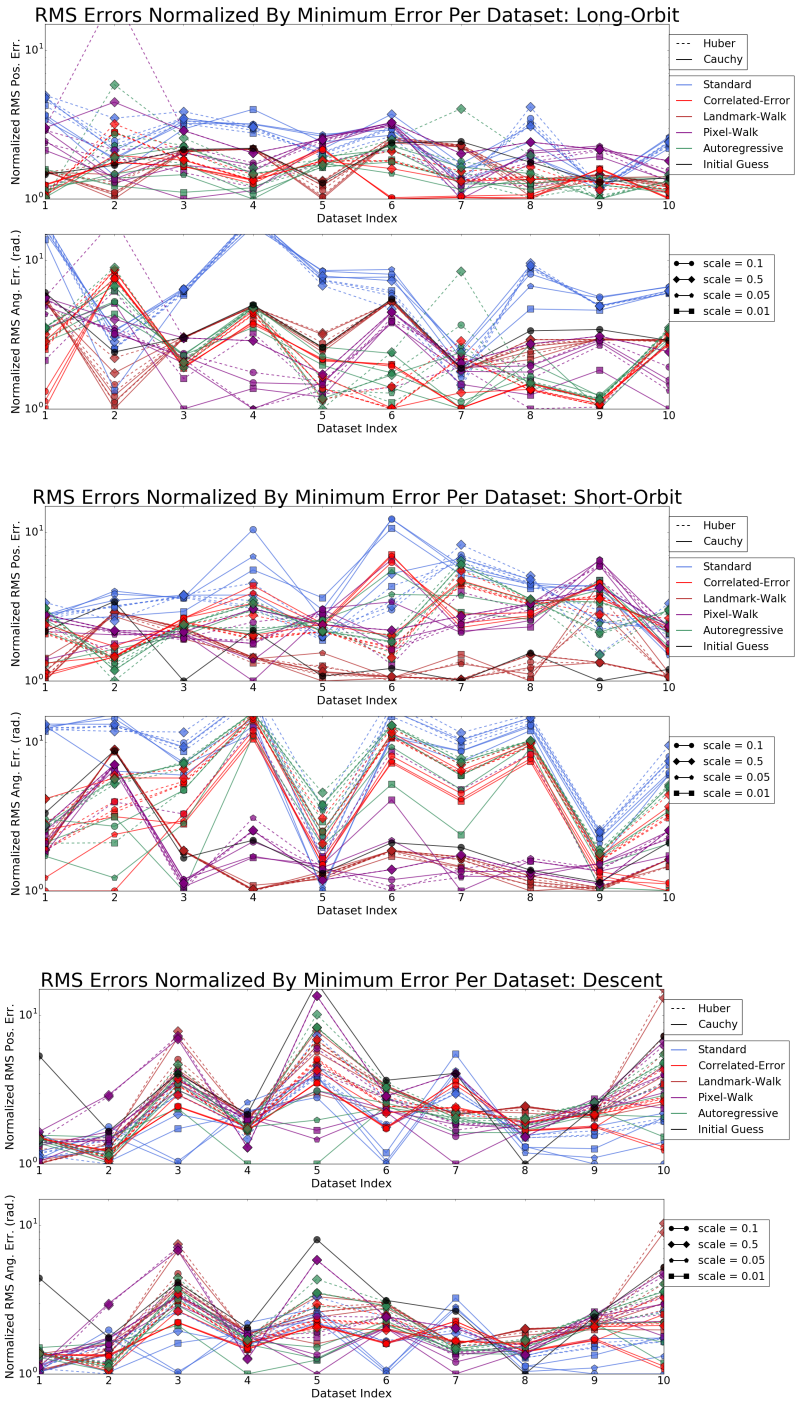


Figure 5.21: RMS error for various estimators on each of ten datasets from each trajectory class. Initial pose estimate from vision pipeline.



cases (see Table (5.1) for example). This may indicate a more fundamental issue with the standard model. Also, the fact that performance improves with decreasing scale for the Huber distribution in particular, suggests an $\mathscr{L}_1$ cost may be best as the very small scale essentially makes the Huber cost look like an $\mathscr{L}_1$ cost.

In some cases, the tight Cauchy scale did give large errors. For example, see the fourth dataset for the short-orbit in Figure (5.21), where the position and angle errors are especially high for the standard model with Cauchy cost. Part of this may be a convergence issue. As will be shown in the second test, convergence issues occur with the Cauchy cost at a small scale.

Another interesting behavior for the standard model is that in some cases, the attitude error is small relative to other estimators while the position error is large. See for example, the fifth dataset in the short-orbit case where the standard model with Cauchy cost and scale 0.01 has the highest position error but nearly the lowest attitude error. An explanation for that behavior is as follows.

Although the behavior may seem surprising, a simple thought experiment can provide needed insight. First consider a scenario of a planar scene of mapped landmarks with a camera pointed normal to the plane and moving along the plane (similar to what occurs in the short-orbit case). If all features have the same localization error at one instant in time, then this will cause a large position error but little-to-no angle error. For a nearly planar scene, large inter-feature error correlations can cause large position error growth without large angle error growth. This type of correla-

tion has not been modeled and its effect on estimator performance in general scenes is difficult to assess. Nevertheless, it seems to be a reasonable explanation for the observed behavior. In the cases considered here, where landmarks are not mapped *a priori*, the issue can be exacerbated by a lack of observability. This would explain why the effect is most evident in the short-orbit case where there is less motion than in the long-orbit case and where all features undergo a similar displacement in image space.

The results in Figure (5.20) seem to suggest that the landmark-walk estimator is insensitive to scale in most cases (note that it closely overlaps with the pixel-walk estimator). This is consistent with Figure (5.21) where all the trend-lines associated with the landmark-walk model (maroon colored) are closely bunched together. In addition to being closely bunched together, they also seem to follow the trend-line for the initial guess itself (black colored). This suggests that the landmark-walk estimator is frequently converging to a local minima very close to the initial guess. Convergence issues for this estimator will be confirmed in the second test.

The correlated error model also seems to be insensitive to scale and choice of cost (Huber or Cauchy). Looking at Figure (5.21), the correlated error model results do *not* appear to track the initial guess, unlike in the landmark-walk case. This suggests that the correlated error model does not suffer the convergence issues that the landmark-walk model does. This is confirmed in the second test.

The autoregressive model dues have significant variation due to scale and dis-

tribution in the short-orbit and descent trajectory classes but not in the long-orbit class. There is a clear reduction in error for the Cauchy cost over the Huber cost at all scales. In addition, the tighter scales appear to give better performance.

The pixel-walk estimator has very similar trends as the autoregressive estimator. This is unsurprising as these two models are very similar. Again, there seems to be an advantage for tighter scales and an advantage for the Cauchy cost over the Huber cost, especially when looking at Figure (5.20). Looking at Figure (5.21), it does appear that the Huber is competitive with the Cauchy cost in many cases.

This concludes the section on intra-model trends. The next section will compare the behavior across the different models.

Up to this point, the discussion has focused on the performance on each model. In the following discussion, the models are compared. First, the results for the long-orbit trajectory are discussed, followed by the short-orbit cases, and finishing with the descent cases.

From Figure (5.20), the standard model cost that gives the best typical-case performance is the $\sigma = 0.01$ Cauchy which has an attitude error of 0.0051 radians and a position error of 0.026 distance units. One key result is that the best-tuned estimators under each of the newly proposed models does better than the best-tuned estimator under the standard model! The performance of each novel estimator on the long-orbit trajectory are briefly summarized below. The landmark-walk model with a $\sigma = 0.1$ Cauchy cost gives 0.0024 and 0.018 angle and position error respectively:

a reduction of 51 % and 31% over the standard model. The pixel-walk model with a $\sigma = 0.01$ Cauchy cost and $\alpha = 1.0$ gives 0.0013 and 0.017 angle and position error respectively: a reduction of 73 % and 35%. The correlated error model with a $\sigma = 0.1$ Cauchy cost and $\alpha = 1.0$ gives 0.0015 and 0.017 angle and position error respectively: a reduction of 69 % and 35%. The autoregressive model with a $\sigma = 0.01$ Cauchy cost and $\alpha = 1.0$ gives 0.0017 and 0.017 angle and position error respectively: a reduction of 65 % and 35%. The fact that each of these estimators can outperform the standard model by such a large margin is a key result of this dissertation! Several other trends should be noted.

For the short-orbit case, the standard estimator performs best in the typical case with a $\sigma = 0.01$ Cauchy cost where the angle and position error is 0.0039 and 0.023 respectively. The landmark-walk model with a $\sigma = 0.01$ Huber cost gives 0.0012 and 0.00896 angle and position error respectively: a reduction of 69 % and 61%. Similar to the long-orbit case, the landmark-walk model estimator appears very insensitive to changes in error scale. The pixel-walk model with a $\sigma = 0.01$ Cauchy cost and $\alpha = 1.0$ gives 0.0013 and 0.014 angle and position error respectively: a reduction of 67 % and 43%. The correlated error model with a $\sigma = 0.01$ Cauchy cost and $\alpha = 1.0$ gives 0.0022 and 0.017 angle and position error respectively: a reduction of 26 % and 35%. The autoregressive model with a $\sigma = 0.01$ Cauchy cost and $\alpha = 1.0$ gives 0.002 and 0.015 angle and position error respectively: a reduction of 49 % and 35%. Again, the newly proposed models all outperform the standard estimator.

The behavior for the descent trajectory class is different then the previous two trajectory classes. For the standard estimator, the best version is the Cauchy with $\sigma = 0.1$ which is a much larger scale then what worked best in the previous cases. The angle and position error are 0.00064 and 0.00125 respectively at that estimator. The landmark-walk model performs **worse** than this: 0.00080 and 0.0165 at the $\sigma = 0.01$ Cauchy cost which is the best tuned landmark-walk estimator. This is an **increase** of 25% and 32% respectively. The other three novel estimators give very slightly better performance than the standard model. In particular, the best-tuned pixel-walk estimator is the $\sigma = 0.1$ Cauchy estimator which yields an angle and position error of 0.00075 and 0.0166: an increase of 14.7% and 24.7%. The best-tuned correlated error estimator is the $\sigma = 0.5$ Cauchy estimator which yields an angle and position error of 0.00075 and 0.0147: an increase of 14.7% and 15.0%. The best-tuned autoregressive estimator is the $\sigma = 0.05$ Cauchy estimator which yields an angle and position error of 0.00069 and 0.0138: an increase of 7.2% and 9.4%. Based on these results the α-parameterized estimators have a slight reduction in performance over the standard estimator for the descent trajectory class.

To summarize the key issues in the above discussion, in the long-orbit and short-orbit cases, there is clear evidence to suggest that the α-parameterized models with $\alpha = 1$ outperform the standard estimator. This trend is not nearly as clear in the descent motion cases where the α-parameterized estimators give slightly worse performance. The reason for the different behaviors between the motion cases is

discussed later in Section 5.11.6. These results are primarily used to develop a basic understanding of the estimator performance. Along with the results from the second test which is presented in the next section, the results of this test are used to select a subset of estimators for a statistically rigorous performance evaluation.

5.11.3 Sensitivity to Initial Guess

Much of the discussion in this section up to this point has been on developing accurate models of visual measurement errors for the purpose of designing estimators. Such estimators are an important result in their own right. However, in practice, optimization algorithms are needed that can obtain the mathematically correct estimate. The nonlinearity of the camera projection equations leads to a non-convex optimization problem even when the cost function is convex (as it is for least-squares and the Huber cost but not for the Cauchy cost). Therefore there is no guarantee of convergence to a global minimum. In presenting results on the performance of these estimators, it is important to have a sense of how well each estimator converges.

A sensitivity study was setup to understand how the estimator output depends on the input initial guess. To do so, a total of $1+n_{trials}$ estimates were obtained for each of $n_{sets} = 10$ (5 of the long-orbit and 5 of the descent type) datasets for several representative estimators. One of the $1+n_{trials}$ estimates uses the ground truth to initialize the estimator. The other $n_{trials} = 5$ estimates use random perturbations about the ground truth. The random perturbations are generated by adding random Gaussian errors to the translation vector and rotating the true rotation matrix by a

small rotation matrix constructed from three random Gaussian Euler angles (1-2-3 sequence). The scale of these errors is $\sigma = 0.02$ distance units for translation and $\sigma = 0.005\text{rad}$ for attitude. These values were selected as they have a similar order of magnitude as the standard estimator output.

For each of the n_{trials} trials of each dataset with each estimator, the RMS of the difference between the resulting estimate and the estimate initialized at the truth is obtained. Of the n_{trials} RMS (angle and position) values for each estimator-dataset pair, both the median and maximum are saved for plotting to see both a typical and worst-case value. The low number of trials, $n_{trials} = 5$, was selected out of practical constraints. A total of 20 estimators were considered in this analysis and had to be run on each of $n_{sets}(n_{trials} + 1)$ datasets. This gave a total of $20 \times 10 \times 6 = 1200$ estimate computations for this analysis alone. While some of the estimators have fairly short run times, < 10 seconds for the standard model, the pixel-walk estimator can take up to 100 seconds. If more resources and time were available, a larger number of datasets and trials would have been used to obtain a more comprehensive understanding of convergence.

Note that if this were a convex problem, the RMS difference would be zero for all n_{trials} trials, as each trajectory would converge to the same global minimum. They key question that this section seeks to answer is if typical variations in the initial guess are large enough to cause the estimator to converge on different local minima. To be clear, this is not used to determine estimator performance since we

are comparing estimates to other estimates (and not to ground truth). However, this will be used to help explain variation in performance across the datasets.

The results are aggregated in Table (5.4) . Note that in the left-hand column, the standard model is represented by *ST*, the landmark-walk model by *LW*, the correlated error model by *CE*, the autoregressive model by *AR*, and the pixel-walk model by *PW*. In addition the Huber and Cauchy costs are represented by *H* and *C* respectively. The last identified in the left-hand column is the scale of the estimator. For example, the estimator in the first row is *ST-H-0.01* which is the standard model with Huber cost at a scale of 0.01. The other columns give the maximum and median RMS difference, as described above.

Consider the results of the standard estimator in the top eight rows of Table (5.4) . Some expected trends are as expected. First, the Huber estimator appears to converge on the same solution in all cases: the RMS difference is on the order of 10^{-6} and 10^{-7} for position and angle respectively. The Cauchy with $\sigma = 0.5$ gives a similar result. As the scale of the Cauchy is reduced, the convex region of the Cauchy cost function shrinks which increases the difficulty in finding the optimal solution. To see this, note that the Cauchy cost and its first two derivatives are

$$\rho(x) = \log(1 + x^2/\sigma^2) \tag{5.72}$$

$$\rho'(x) = \frac{2x/\sigma^2}{1 + x^2/\sigma^2} \tag{5.73}$$

$$\rho''(x) = \frac{2/\sigma^2}{1 + x^2/\sigma^2} - \frac{4x^2/\sigma^4}{(1 + x^2/\sigma^2)^2} \tag{5.74}$$

The second derivative is positive on the interval $[-\sigma, \sigma]$ and negative outside of that.

	Long-Orbit				Descent			
	Angle		Trans.		Angle		Trans.	
Est.	Med.	Max	Med.	Max	Med.	Max	Med.	Max
ST-H-0.01	1.3e-6	6.4e-6	1.3e-5	1.0e-4	2.1e-6	1.6e-5	4.7e-5	3.6e-4
ST-H-0.1	1.5e-6	1.1e-5	2.1e-5	9.9e-5	1.2e-6	1.0e-5	2.7e-5	2.4e-4
ST-H-0.5	5.8e-7	3.5e-6	7.5e-6	3.3e-5	7.3e-7	7.3e-6	1.7e-5	1.7e-4
ST-C-0.01	7.6e-4	2.3e-3	1.4e-2	2.9e-2	4.1e-4	1.4e-3	9.7e-3	3.2e-2
ST-C-0.1	2.9e-5	2.0e-4	5.1e-4	3.5e-3	3.2e-5	1.3e-4	7.1e-4	3.0e-3
ST-C-0.5	8.3e-7	3.5e-5	1.1e-5	4.9e-4	2.5e-7	1.7e-6	6.8e-6	4.0e-5
LW-H-0.01	4.2e-3	8.5e-3	7.7e-2	1.4e-1	8.5e-6	4.1e-5	1.9e-4	8.8e-4
LW-H-0.1	4.1e-3	8.6e-3	7.6e-2	1.3e-1	2.5e-5	1.4e-4	5.4e-4	3.0e-3
LW-C-0.01	4.1e-3	8.9e-3	7.2e-2	1.8e-1	6.5e-6	5.7e-5	1.4e-4	1.3e-3
LW-C-0.1	4.0e-3	9.2e-3	7.2e-2	1.3e-1	1.5e-5	1.0e-4	3.1e-4	2.1e-3
CE-H-0.01	1.1e-5	9.0e-5	7.3e-5	6.4e-4	4.5e-6	2.0e-5	1.0e-4	4.5e-4
CE-H-0.1	6.3e-6	6.2e-5	4.4e-5	4.3e-4	5.3e-6	1.8e-5	1.2e-4	4.2e-4
CE-C-0.01	3.6e-6	5.6e-5	3.9e-5	2.9e-4	8.4e-6	2.7e-5	1.8e-4	6.1e-4
CE-C-0.1	8.7e-6	3.0e-5	5.1e-5	1.4e-4	8.1e-6	3.0e-5	1.8e-4	7.0e-4
PW-H-0.01	1.8e-4	5.1e-4	1.2e-3	3.2e-3	8.9e-6	1.6e-5	2.0e-4	3.8e-4
PW-H-0.1	5.0e-4	1.3e-3	2.8e-3	8.5e-3	9.0e-5	3.9e-4	2.0e-3	9.2e-3
PW-C-0.01	3.2e-4	9.2e-4	2.1e-3	7.2e-3	8.4e-5	5.5e-4	1.8e-3	1.3e-2
PW-C-0.1	5.4e-4	1.5e-3	3.1e-3	1.8e-2	1.1e-4	6.6e-4	2.5e-3	1.6e-2
AR-H-0.01	2.2e-6	1.7e-5	1.7e-5	1.0e-4	6.5e-6	3.0e-5	1.5e-4	6.9e-4
AR-H-0.1	1.5e-6	4.5e-5	1.5e-5	3.3e-4	5.9e-6	2.6e-5	1.4e-4	5.8e-4
AR-C-0.01	2.1e-4	5.5e-4	3.7e-3	1.1e-2	2.7e-4	8.0e-4	6.3e-3	1.9e-2
AR-C-0.1	3.5e-6	7.5e-5	4.8e-5	2.9e-4	1.0e-5	7.5e-5	2.3e-4	1.7e-3

Table 5.4: Results of sensitivity study. Median and maximum difference in RMS error between solutions using nominal and perturbed initial guesses.

Clearly the interval shrinks as σ decreases which may cause either an increase in the number of local minima or a decrease in the region-of-attraction to the correct local minimum. On the other hand, the second derivative of the Huber cost

$$\rho(x,\sigma) = \begin{cases} \frac{1}{2}x^2/\sigma^2 & |x|/\sigma \leq k \\ k|x|/\sigma - \frac{1}{2}k^2 & |x|/\sigma > k \end{cases} \tag{5.75}$$

is positive on the interval $[-k\sigma, k\sigma]$, zero outside of the interval, and negative

nowhere. Although the Huber cost function is a convex cost function, it is not convex in the pose and landmark parameters which make up the actual estimator cost function. Nevertheless, this likely explains the reduced sensitivity to the estimators using a Huber cost. The sensitivity to scale for the Cauchy cost is clearly apparent in the figures: as the scale goes from $\sigma = 0.5$ to $\sigma = 0.1$ to $\sigma = 0.01$, the RMS angle differences go from 10^{-7} to 10^{-5} to 10^{-4} with similar changes for position.

Before proceeding, it is worth determining if the small differences for the Huber cost and the $\sigma = 0.5$ Cauchy cost are due to the stopping condition of the optimization routine or due to separate but nearly-collocated local minima. The optimization is terminated when the normalized reduction in cost, $(\mathscr{J}^{(k)} - \mathscr{J}^{(k+1)})/\mathscr{J}^{(k)}$ between two iterations is below 10^{-6}. Performing a first-order Taylor series expansion of this expression about a parameter correction $\delta\boldsymbol{\theta}$ of zero yields

$$(\mathscr{J}^{(k)} - \mathscr{J}^{(k+1)})/\mathscr{J}^{(k)} \approx \frac{-1}{\mathscr{J}^{(k)}} \frac{\partial \mathscr{J}^{(i+k)}}{\partial \delta\boldsymbol{\theta}} \tag{5.76}$$

If a least squares cost function is used for $M = 20$ measurements of each of $N = 200$ landmarks, then a typical value for the cost will be $\approx 2 \times MN = 8000$ (based on the mean of a χ^2-distribution of residuals). The Jacobian of the cost with respect to the parameter correction is

$$\frac{\mathscr{J}^{(k+1)}}{\partial \delta\boldsymbol{\theta}} = \sum_{i=1}^{M} \sum_{j=1}^{N} \frac{-2}{\sigma^2} (\tilde{\mathbf{y}}_{i,j} - \hat{\mathbf{y}}_{i,j}) \frac{\partial \hat{\mathbf{y}}_{i,j}}{\partial \delta\boldsymbol{\theta}} \tag{5.77}$$

Therefore a small parameter correction reduces the cost approximately by a factor

of

$$\frac{-1}{\mathscr{J}^{(k)}}\frac{\partial \mathscr{J}^{(k+1)}}{\partial \delta\boldsymbol{\theta}} \approx \frac{1}{\mathscr{J}^{(k)}}\sum_{i=1}^{M}\sum_{j=1}^{N}\frac{2}{\sigma^2}(\tilde{\mathbf{y}}_{i,j}-\hat{\mathbf{y}}_{i,j})\frac{\partial \hat{\mathbf{y}}_{i,j}}{\partial \delta\boldsymbol{\theta}} \tag{5.78}$$

Now consider the predicted measurement at time i of landmark j:

$$\hat{\mathbf{y}}_{i,j} = \boldsymbol{\pi}\left(\hat{R}_i\hat{\mathbf{x}}_j + \hat{\mathbf{t}}_i\right) \tag{5.79}$$

where

$$\boldsymbol{\pi}([x,y,z]) = [fx/z+c_x,\, fy/z+c_y] \tag{5.80}$$

To perform an order of magnitude analysis, the Jacobian of the predicted measurement with respect to the parameters when $f=1000$, $\hat{R}_i = I$, $\hat{\mathbf{t}}_i = \mathbf{0}$, and $\hat{\mathbf{x}}_j = [0,0,8]$ (which is representative for this dataset) is

$$\frac{\partial \hat{\mathbf{y}}_{i,j}}{\partial \hat{\mathbf{t}}_i} = \begin{bmatrix} 1000/8 & 0 & 0 \\ 0 & 1000/8 & 0 \end{bmatrix} \tag{5.81}$$

$$\frac{\partial \hat{\mathbf{y}}_{i,j}}{\partial \delta\boldsymbol{\theta}_i} = \begin{bmatrix} 1000/8 & 0 & 0 \\ 0 & 1000/8 & 0 \end{bmatrix} \left[([0,0,8])^{\times}\right] \tag{5.82}$$

$$= \begin{bmatrix} 0 & -1000 & 0 \\ 1000 & 0 & 0 \end{bmatrix} \tag{5.83}$$

A correction of Δ to either the first two elements of $\hat{\mathbf{t}}_i$ or $\delta\boldsymbol{\theta}_i$ changes the predicted measurement by 125Δ or 1000Δ respectively. For the translational change, a typical change in the least squares cost is a factor of

$$\sqrt{\mathrm{VAR}\left[(\mathscr{J}^{(k)}-\mathscr{J}^{(k+1)})/\mathscr{J}^{(k)}\right]} \approx \sqrt{\mathrm{VAR}\left[\frac{1}{\mathscr{J}^{(k)}}\sum_{i=1}^{M}\sum_{j=1}^{N}\frac{2}{\sigma^2}(\tilde{\mathbf{y}}_{i,j}-\hat{\mathbf{y}}_{i,j})\frac{\partial \hat{\mathbf{y}}_{i,j}}{\partial \delta\boldsymbol{\theta}}\right]}$$

$$= \frac{1}{\mathscr{J}^{(k)}}\sqrt{\sum_{j=1}^{N}\frac{4\sigma^2}{\sigma^4}125^2\Delta^2}$$

$$= \frac{1}{2MN}\sqrt{N\frac{4\sigma^2}{\sigma^4}125^2\Delta^2} \tag{5.84}$$

$$= \frac{2\times 125\Delta}{40\sqrt{200}\sigma^2} \tag{5.85}$$

$$= 125\times 0.0035\Delta \tag{5.86}$$

$$= 0.44\Delta \tag{5.87}$$

and 3.5Δ for an angular correction (when $\sigma = 1.0$). Note that the dependence on σ is via the error scale and *not* due to weighting in the cost function. The critical value of 10^{-6} for the stopping criteria corresponds to a Δ of 2.3×10^{-6} and 2.8×10^{-7} for translational and angular corrections respectively. Note that the critical values for Δ will be slightly larger for the Huber cost than for the least squares cost (in the above analysis) because the Jacobian of the Huber cost with respect to the parameters will be equal or less than that of the least squares cost. This is especially true for measurements with large residuals which further amplifies the effect. The reduction in the Jacobian is exaggerated for the Cauchy cost whose slopes tend to zero as the residual magnitude increases.

The above analysis, although very coarse, is in agreement with the spread of the trials for the Huber cost and $\sigma = 0.5$ Cauchy cost in Table (5.4) . This suggests that these small variations are simply due to the termination criteria of the optimization routine and not due to convergence to different local minima which is an important point.

Next, consider the landmark-walk model. The convergence results for this

model showed much different behavior between the long-orbit and descent trajectories. From Table (5.4) , the long-orbit case can have variations up to 0.1 distance units and 0.01 radians just due to convergence issues. These values are about ten times higher than they are in the descent case.

The pixel-walk model had issues with convergence for the Cauchy costs in both the long-orbit and descent cases. Interestingly, the issue was more pronounced at the larger Cauchy scale in the descent case but not the long-orbit case. This behavior was also seen in the autoregressive model. Possible explanations for this are given later in Section 5.11.6.

The correlated error model had the most consistent convergence out of the α-parameterized models. The convergence seemed to be insensitive to the scale parameter and seemed to be similar between the Cauchy and Huber costs.

This concludes the results for the second study of this test. The results of this test and the first are used to select a subset of estimators for a more rigorous analysis. The selection is discussed in the next section followed by the results of the third test.

5.11.4 Selecting Estimators

This section discusses the selection of a subset of estimators for further testing. It was desired to include at least one estimator from each of the five models. In order to demonstrate that an advantage of designing estimators around the four novel models instead of the standard model, it is important that the standard model

is given a *fair chance*. For this purpose, a relatively large number of estimators were selected for the standard model to ensure that at least one estimator designed around the standard model was well-tuned. The selected standard model estimators are those with both the Cauchy and Huber costs at scales 0.01, 0.05, 0.1, and 0.5. These are *all* the standard model estimators tested and documented in Section 5.11.2. Standard model estimators with scale greater than 0.5 were found to have inferior performance and eliminated from further consideration.

For the autoregressive model, the Cauchy costs were found to have better performance than the Huber costs in Section 5.11.2. In addition, the results of Section 5.11.3 did not reveal a large difference in initial guess sensitivity between the two costs. Therefore, the Cauchy cost with scale 0.01, 0.05, and 0.1 were selected.

For the pixel-walk model, the Cauchy costs had an advantage is most cases over the Huber cost. However, there were occasional convergence issues with the Cauchy cost that were not as pronounced in the Huber cost. In addition the optimal scale appeared to be around 0.01 to 0.1. Therefore the selected pixel-walk estimators were one with Cauchy cost at scale 0.05 and one with Huber cost at scale 0.05.

For the correlated error model, the Cauchy cost had superior performance and there was no significant difference in convergence between the Cauchy and Huber costs. In addition, the optimal scale appeared to be between 0.1 and 0.5. Therefore the Cauchy scale with scale 0.1 and 0.5 were selected.

In addition, an estimator to jointly estimate scale was selected for the Cauchy

and Huber cost for both the standard model and the autoregressive model. The 20 selected estimators, ten standard and ten novel, underwent further testing which is presented in the next section.

5.11.5 Blocked Paired-Different Tests

This section presents results for the third test of this study. In this test, fifty datasets were used from both the long-orbit and descent trajectory class. The short-orbit was not included as it had qualitatively similar behavior to the long-orbit case. The initial guess for each dataset was corrupted with a high-level of zero-mean Gaussian noise: $\sigma = 0.25$ distance units for translation and $\sigma = 0.03$. The RMS error for translation and attitude were computed by comparing the results from each estimator on each dataset to the corresponding ground truth for each dataset. The raw results of the test are given in Figure (5.22). Box-and-whisker plots in Figure (5.23) and Figure (5.24) give a clearer picture of the relative performance. The labels in the box-and-whisker plots use the same notation as Table (5.4) with the additional convention of an X to indicate automatic scale estimation (as in ST-C-X for example). A discussion is needed to make sense of the raw data.

First, it is clear that of the novel models, the autoregressive model and correlated error model are superior for both the long-orbit and descent trajectory class. The four estimators derived from the autoregressive model gave similar performance to each other. The autoregressive estimator with automatic scale estimation gave very similar performance to the other autoregressive estimators. This is unsurpris-

Figure 5.22: RMS error for multiple estimators on the long-orbit (top) and descent (bottom) trajectory class.

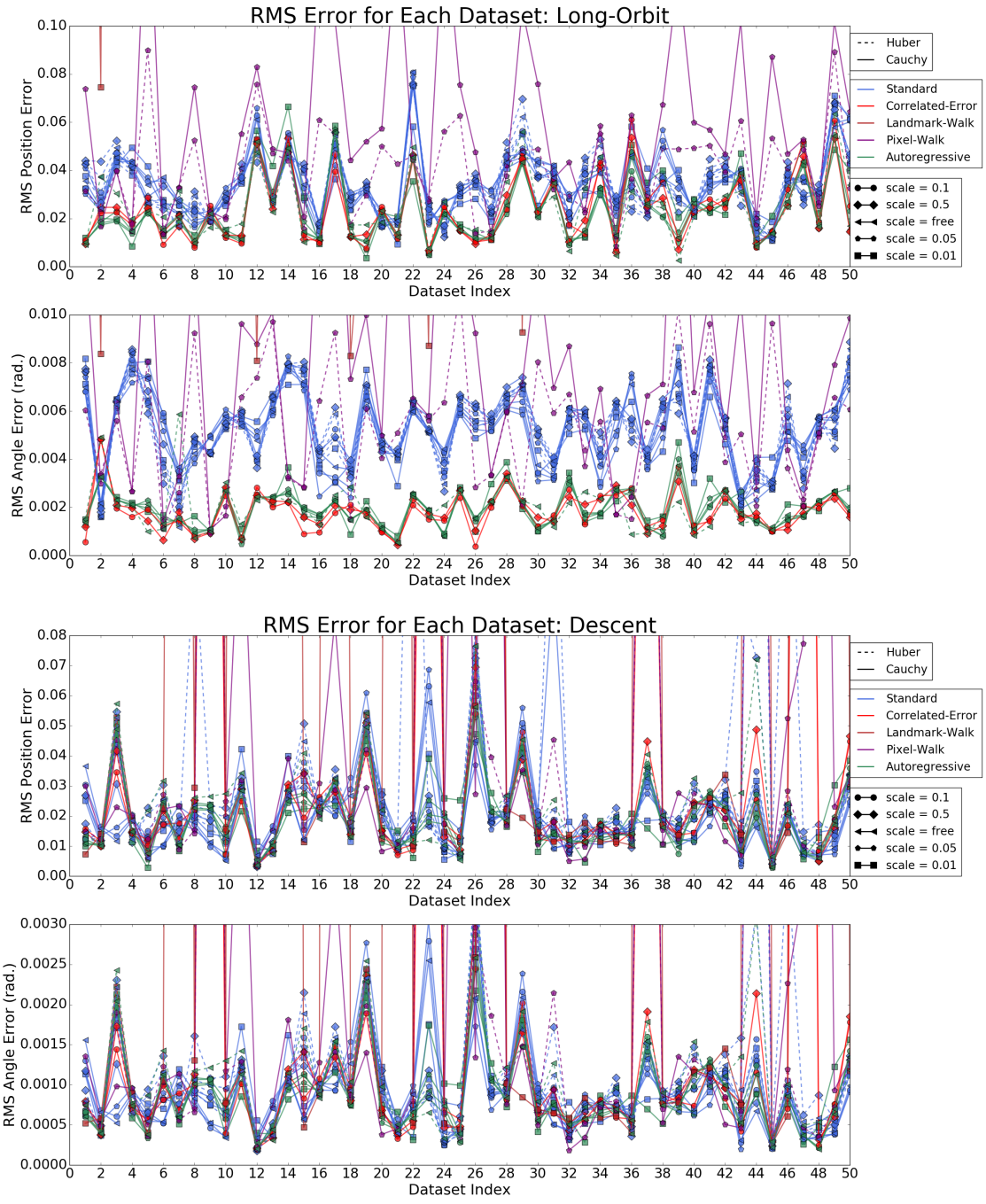


Figure 5.23: Long-orbit trajectory class: Box-and-whisker plot for RMS position (top) and angle (bottom) errors. Bottom: Box-and-whisker plot for RMS angle errors.

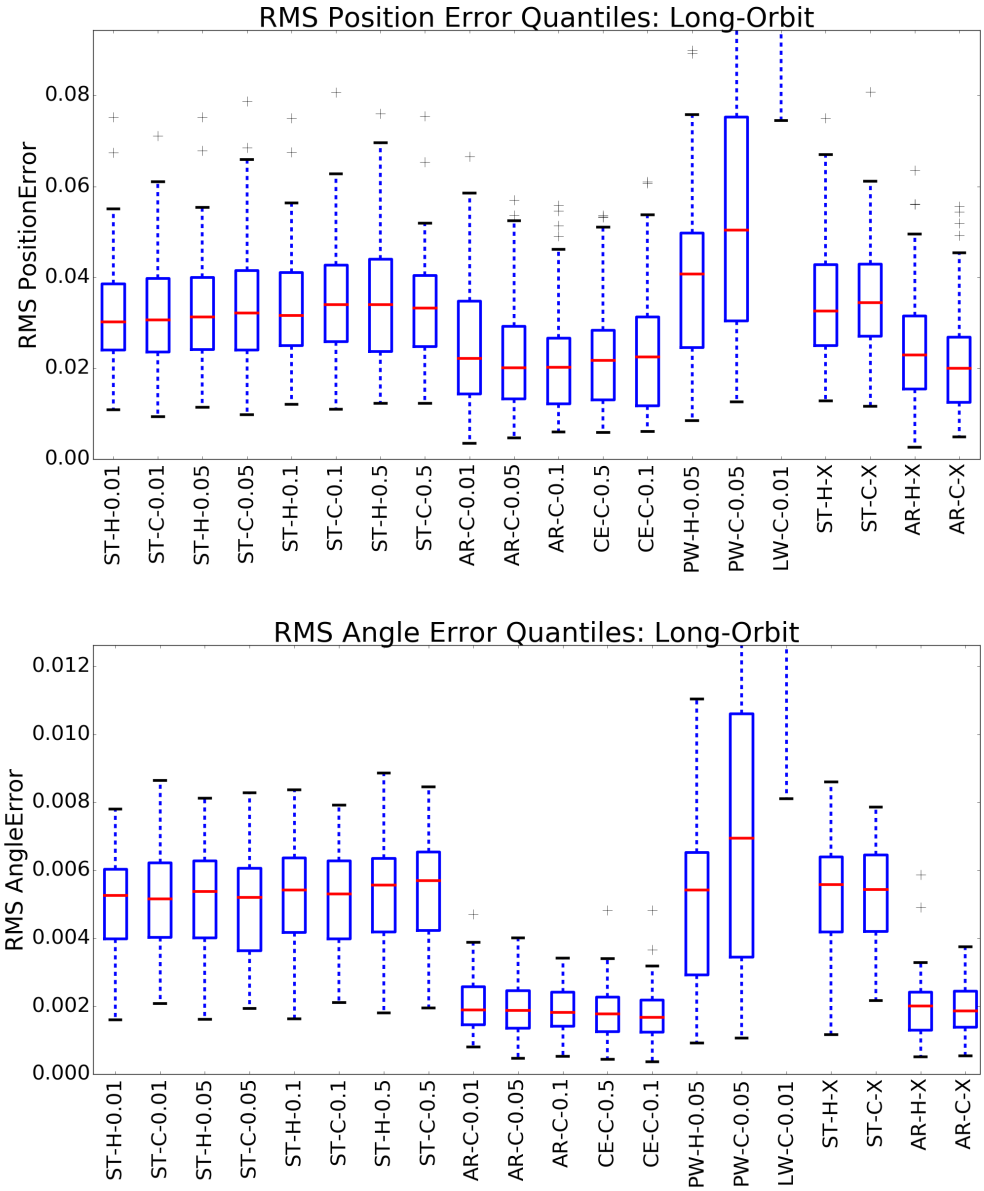
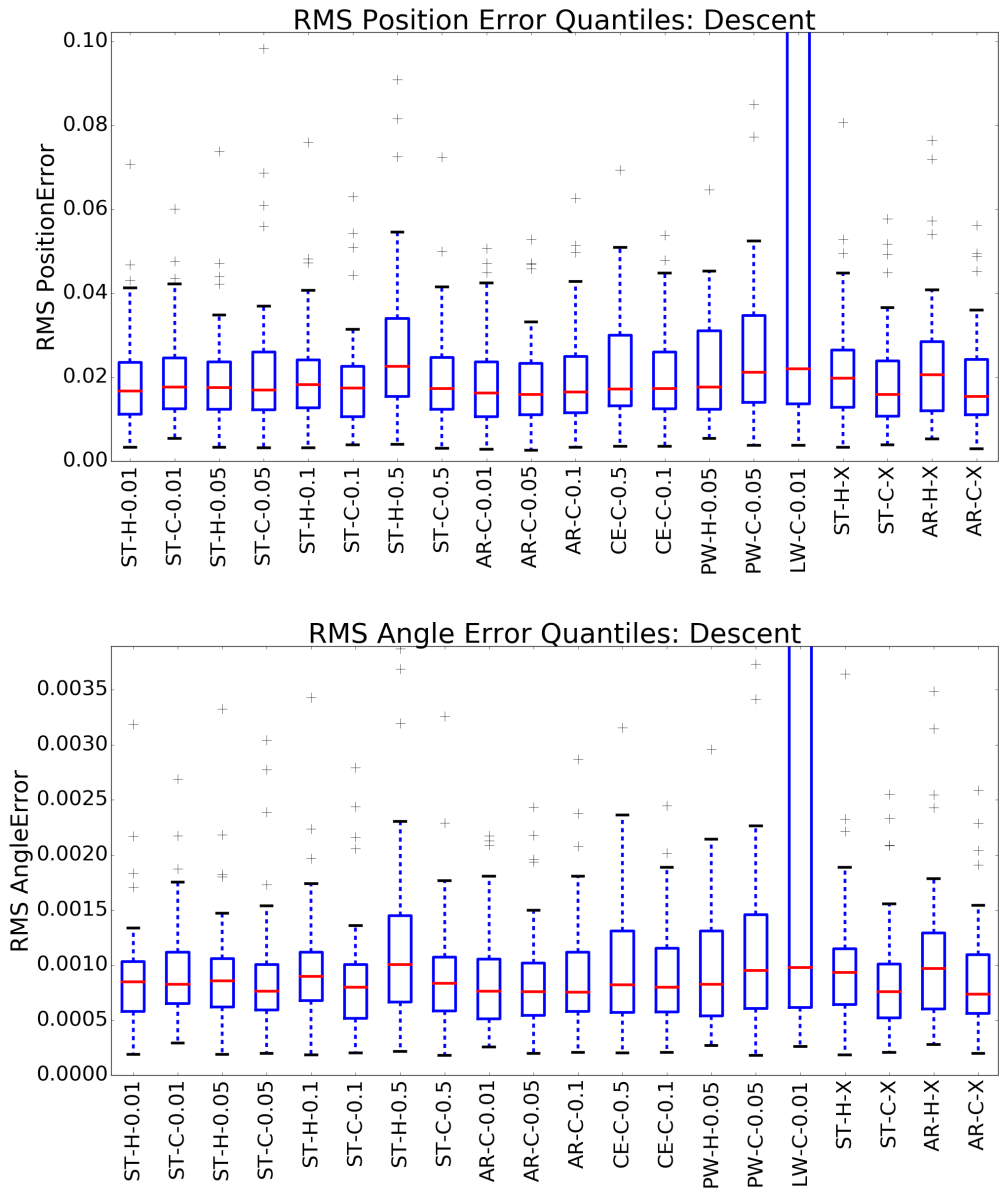


Figure 5.24: Descent trajectory class: Box-and-whisker plot for RMS position (top) and angle (bottom) errors. Bottom: Box-and-whisker plot for RMS angle errors.



ing as the automatic scale estimation returned a scale between 0.05 and 0.1 in all cases. Table (5.5) compares all standard estimators to the autoregressive estimator with automatically estimated scale. For the long-orbit case, the table shows that the correlated error estimator reduces the angle and position error by more that 65% and 25% respectively for *all* instances of the standard model estimator! Additionally for the long-orbit case, the WSRT p-value is reported at essentially zero which provides strong evidence that the autoregressive estimator dominates the standard estimator. The advantage of the autoregressive model is significantly reduced in the descent trajectory case. The reduction in error is on the order of 1% to 10% for both angle and position. The standard model estimator with $\sigma = 0.1$ Cauchy cost is most competitive with the autoregressive estimator as indicated by the WSRT p-value.

Of the two estimators derived from the correlated model, the $\sigma = 0.1$ Cauchy cost gave the best performance which is clear from Figure (5.23). Table (5.6) compares all standard estimators to the correlated error estimator with $\sigma = 0.1$ Cauchy cost. The results are similar to that of the autoregressive model. In the long-orbit case, the WSRT p-value provides very strong evidence that the correlated error model dominates the standard model. In addition, angle and position errors are reduced by more than 60% and 30% relative to all instances of the standard model estimator. Like in the autoregressive model, the performance advantage of the correlated error model is greatly reduced (or eliminated) for the descent trajectory case.

In summary, the autoregressive and correlated error models demonstrated bet-

Table 5.5: Table of WSRT p-values and percent reduction in errors for the autoregressive estimator with automatic scale estimation compared to all instances of the standard estimator.

	Angle		**Position**	
Est.	p	$\frac{e_{ST}-e_{AR}}{e_{ST}} \times 100\%$	p	$\frac{e_{ST}-e_{AR}}{e_{ST}} \times 100\%$
Long-Orbit				
ST-0.01-H	0.000	66.340 %	0.000	34.932 %
ST-0.01-C	0.000	66.037 %	0.000	28.174 %
ST-0.05-H	0.000	67.673 %	0.000	37.159 %
ST-0.05-C	0.000	65.373 %	0.000	33.058 %
ST-0.1-H	0.000	67.822 %	0.000	38.170 %
ST-0.1-C	0.000	66.185 %	0.000	35.692 %
ST-0.5-H	0.000	68.517 %	0.000	41.622 %
ST-0.5-C	0.000	69.436 %	0.000	39.093 %
Descent				
ST-0.01-H	0.260	6.004 %	0.316	6.487 %
ST-0.01-C	0.118	5.247 %	0.164	5.135 %
ST-0.05-H	0.072	7.176 %	0.107	8.462 %
ST-0.05-C	0.306	9.915 %	0.254	12.375 %
ST-0.1-H	0.012	9.684 %	0.023	8.552 %
ST-0.1-C	0.513	11.462 %	0.425	9.226 %
ST-0.5-H	0.000	16.157 %	0.000	23.380 %
ST-0.5-C	0.202	6.746 %	0.227	0.861 %

ter performance than the landmark-walk model and pixel-walk model. Furthermore, the autoregressive model and correlated error model gave a major reduction in error over the standard model for the long-orbit cases. The performance in the descent cases shows similar performance between all three models. The tables and figures of this section validate the use of measurement models to capture $\alpha = 1.0$ correlated errors. The development of estimators using models that capture this correlation and the demonstrated performance gains are a key contribution of this dissertation.

Table 5.6: Table of WSRT p-values and percent reduction in errors for the correlated error estimator with $\sigma = 0.1$ Cauchy cost compared to all instances of the standard estimator.

	Angle		**Position**	
Est.	p	$\frac{e_{ST}-e_{CE}}{e_{ST}} \times 100\%$	p	$\frac{e_{ST}-e_{CE}}{e_{ST}} \times 100\%$
Long-Orbit				
ST-0.01-H	0.000	61.590 %	0.000	38.962 %
ST-0.01-C	0.000	64.135 %	0.000	36.029 %
ST-0.05-H	0.000	62.179 %	0.000	39.811 %
ST-0.05-C	0.000	63.199 %	0.000	33.903 %
ST-0.1-H	0.000	62.869 %	0.000	40.648 %
ST-0.1-C	0.000	62.890 %	0.000	40.029 %
ST-0.5-H	0.000	63.816 %	0.000	41.234 %
ST-0.5-C	0.000	63.834 %	0.000	40.642 %
Descent				
ST-0.01-H	0.552	1.304 %	0.687	0.243 %
ST-0.01-C	0.280	5.486 %	0.289	11.402 %
ST-0.05-H	0.384	3.789 %	0.490	-2.404 %
ST-0.05-C	0.567	9.008 %	0.663	0.845 %
ST-0.1-H	0.289	-0.134 %	0.362	-0.869 %
ST-0.1-C	0.761	-5.329 %	0.811	-12.517 %
ST-0.5-H	0.050	13.761 %	0.059	11.425 %
ST-0.5-C	0.645	0.970 %	0.733	-2.289 %

5.11.6 Discrepancy Between Trajectory Types: Linear Covariance Analysis

One important trend that was found is that the α-parameterized estimators performed much better than the standard estimator on the long-orbit and short-orbit trajectories but not on the descent trajectories. This was a surprising result as the analysis and empirical evidence of Section 5.2.1 suggested that the descent trajectory had errors similar to the other trajectory classes, namely an autoregressive error process with $\alpha = 1.0$. It is important to find a plausible explanation for this as

it is a very significant trend.

To determine whether or not the geometry of the different trajectories is the cause of the discrepancy, a linear covariance analysis can be performed. Consider the stacked measurement vector $\tilde{\boldsymbol{Y}}$, the stacked landmark parameters $\boldsymbol{\theta}_\ell$, the stacked pose parameters $\boldsymbol{\theta}_p$, the stacked noise $\bar{\boldsymbol{\nu}}$, and the stacked measurement function

$$\tilde{\boldsymbol{Y}} = \boldsymbol{h}(\boldsymbol{\theta}_p, \boldsymbol{\theta}_\ell) + \bar{\boldsymbol{\nu}} \tag{5.88}$$

A weighted least squares algorithm (including the robust versions), converges when

$$(H^T W H)^{-1} H^T W (\tilde{\boldsymbol{Y}} - \hat{\boldsymbol{Y}}(\hat{\boldsymbol{\theta}}_p, \hat{\boldsymbol{\theta}}_\ell)) = \boldsymbol{0} \tag{5.89}$$

where H is the Jacobian with respect to the parameters and W is the weight matrix at the final iteration. The error in the parameters can be approximated by linearizing the termination condition about the estimate:

$$(H^T W H)^{-1} H^T W (\bar{\boldsymbol{\nu}} - H \delta\boldsymbol{\theta}) = \boldsymbol{0} \tag{5.90}$$

Solving for the error in the joint parameter error $\delta\boldsymbol{\theta}$ yields

$$\delta\boldsymbol{\theta} = (H^T W H)^{-1} H^T W \bar{\boldsymbol{\nu}} \tag{5.91}$$

If the estimator was designed around the IID error assumption, then the W matrix should be isotropic diagonal. If on the other hand, correlated or autoregressive error of the form

$$\bar{\boldsymbol{\nu}} = A\boldsymbol{w} \tag{5.92}$$

for IID $\boldsymbol{w}$ was assumed, then $W = (AA^T)^{-1}$, where A is defined in Equation (5.58). We can consider the parameter error covariance for each estimator when the error is

truly either IID or correlated. The four cases are

1. IID estimator with IID errors:

$$P_{I,I} = \sigma^2 (H^T H)^{-1} \tag{5.93}$$

2. IID estimator with autoregressive errors:

$$P_{I,A} = \sigma^2 (H^T H)^{-1} (H^T (AA^T)^{-1} H)^{-1} (H^T H)^{-1} \tag{5.94}$$

3. Autoregressive estimator with IID errors:

$$P_{A,I} = \sigma^2 (H^T (AA^T)^{-1} H)^{-1} H^T H (H^T (AA^T)^{-1} H)^{-1} \tag{5.95}$$

4. Autoregressive estimator with autoregressive errors:

$$P_{A,A} = \sigma^2 (H^T (AA^T)^{-1} H)^{-1} \tag{5.96}$$

Up to the nonlinearity of the measurement equations, these covariance expressions are exact when the errors are exactly Gaussian (either IID or correlated as stated above). Furthermore, the estimators associated with $P_{A,A}$ and $P_{I,I}$ are unbiased minimum variance estimators if the assumptions are met. However, $P_{A,A}$ can be compared to $P_{A,I}$ for various instances of H (i.e. instances of landmark configuration and trajectory) to see what is gained by accounting for the correlated errors. This can be done as follows. The pose trajectories and landmark configurations in the evaluation datasets for the long-orbit and descent trajectory classes are used. For each dataset, the true parameters and feature tracking results (to determine landmark visibility only) are used to evaluate the Jacobian H. Note that the first camera pose is treated as a constant and the distance between the first camera and last camera is

treated as a very accurate measurement. This resolves any singularity issues (caused by a lack of observability). The Jacobian is then used to compute the four covariance expressions above with $\sigma = 1$. The diagonal elements of the covariance matrices corresponding to camera position and angle are summed up separately which gives an expected value for the sum of squared position and angle errors respectively.

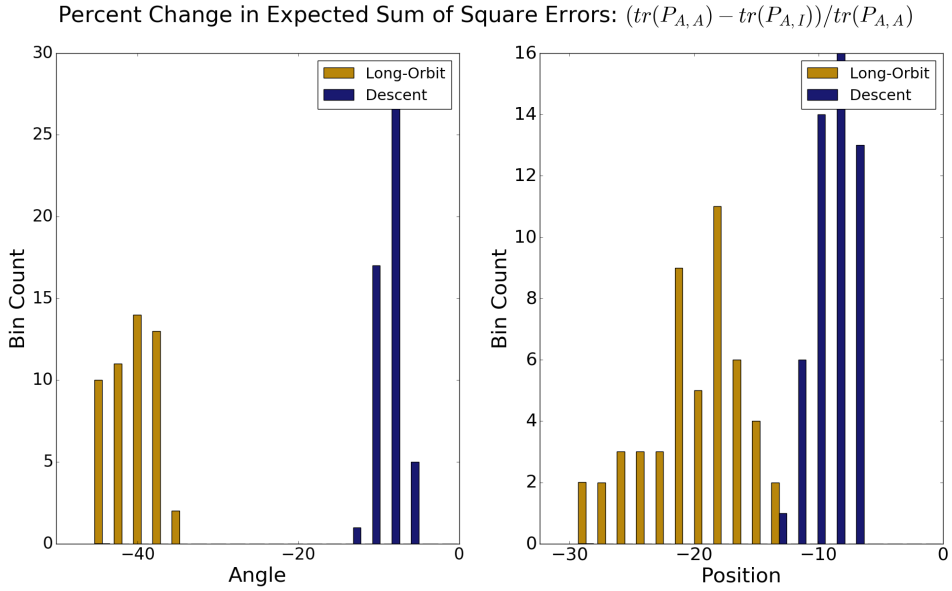
The results make sense to treat as paired samples: for each dataset, independently compare $P_{A,A}$ to $P_{A,I}$ to reduce confounding factors particular to the dataset in question. To further make the difference clear, the percent change is reported:

$$\frac{\mathrm{tr}_\star(P_{A,A}) - \mathrm{tr}_\star(P_{A,I})}{\mathrm{tr}_\star(P_{A,A})} \tag{5.97}$$

where $\mathrm{tr}_\star()$ indicates the trace of the matrix of a particular subset of elements (either camera position or angle parameters). The results are presented in Figure (5.25). Note that for the results in this figure, $\sigma^2 = 1$ was used to evaluate the covariance expressions.

Figure (5.25) demonstrates a very important result. Given that the errors are truly autoregressive rather than IID, the advantage of using an autoregressive estimator over an IID (standard) estimator is greatly diminished. For position errors in the long-orbit case, a reduction of 15 % to 30% on the mean-squared error is expected compared to only 5 % to 10 % in the descent case. Similarly for angle errors, a reduction of 40 % on the meas-squared error is expected compared to only 10 % in the descent case. *This is consistent with the results of Section 5.11.5: see Table (5.5) and Table (5.6)* . This linear covariance analysis applied to the particular

Figure 5.25: Percent change in expected sum of squared angle and position errors between the autogressive and standard estimator when the errors are autoregressive ($\alpha = 1$).

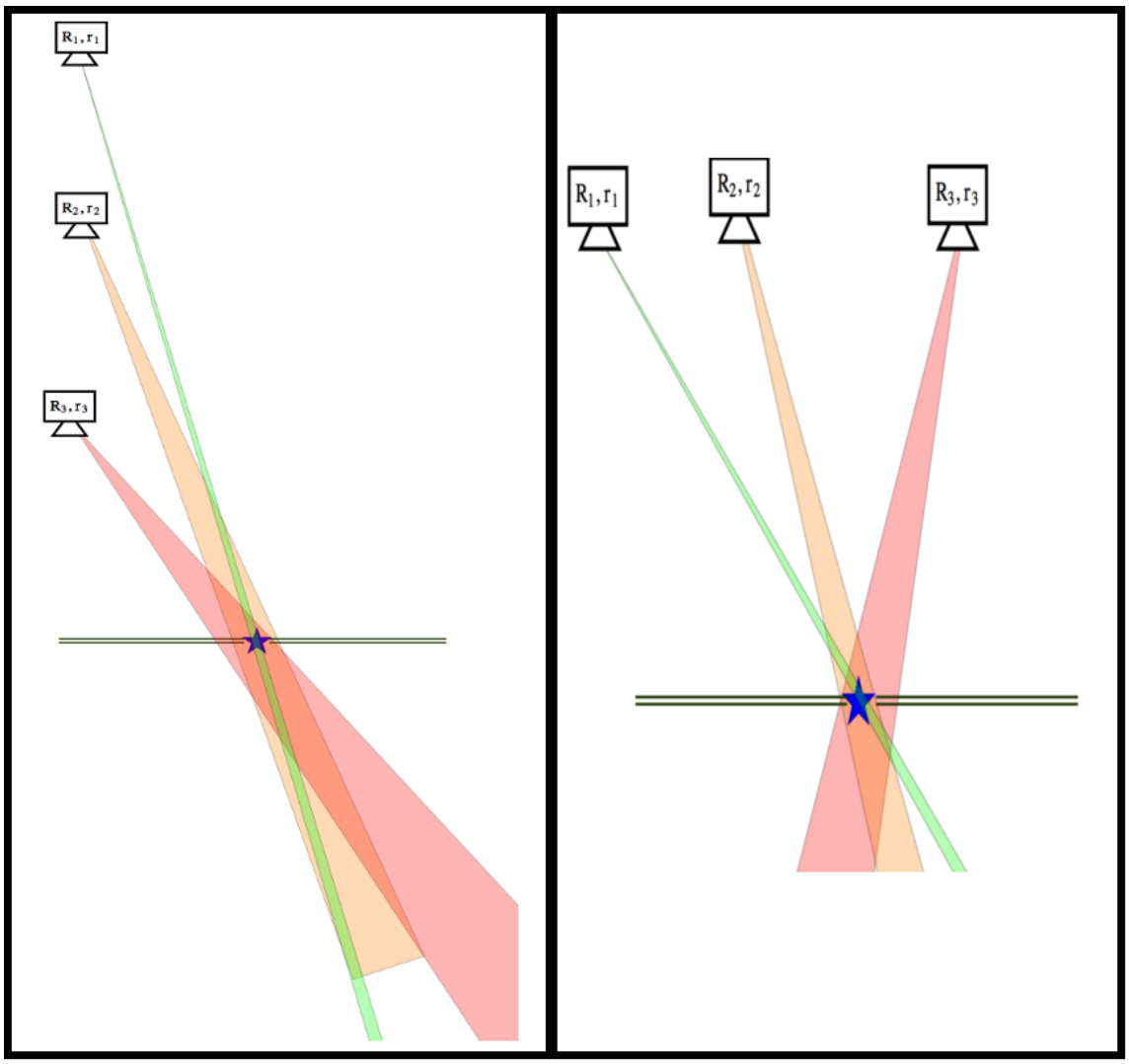


trajectories of interest provides rigorous evidence as to why little-to-advantage was seen for the autoregressive estimator over the standard estimator in the descent trajectories even though a large advantage was seen in the long-orbit motion. An intuitive reason for this can be seen as follows.

The accuracy of the pose estimates is inherently linked to the accuracy of the landmark estimates. The larger the baseline between multiple views of a landmark, the better the estimate will be. In the long-orbit case, the motion is mostly orthogonal to the landmark viewing directions. For IID errors in the long-orbit case, each subsequent view provides more and more information to better localize the landmark. However, if the errors are autoregressive then a tradeoff is implicitly made

between a larger baseline and essentially increasing feature localization errors. On the other hand, in the descent motion case, neglecting the fact that the errors are autoregressive is not as detrimental because the extra baseline in later images is more essential to providing observability. Figure (5.26) illustrates this graphically. The frustums originating from each camera represent a region of uncertainty for the feature measurement (i.e. the blue star). As the motion proceeds, these frustums become wider indicating an increase in error. The left side of the figure shows motion representative of the descent trajectory. The region of overlap for the first two frustums (green and orange) is very large implying that the landmark position has weak observability. Even though the red frustum is much wider, because of the large baseline, it gives significantly more information about the landmark position. This is in contrast to the right-hand side of the figure which represents the long-orbit motion. Even though the last frustum (red) has a large baseline to the first, the large error (frustum width) prevents it from reducing the total overlap region. To summarize, if we had neglected to take into account the autoregressive and hence increasing nature of errors in the descent case, it would not have mattered as much because all measurements would still need high weight. On the other hand, in the long-orbit case, by accounting for the more accurate error model, more weight can be placed on measurements with a low baseline. Again, this explains why the α-parameterized models performed well in the long-orbit case with $\alpha = 1$ but only similarly to the standard model in the descent case.

Figure 5.26: Geometry of different motion cases for autoregressive errors.



The above analysis explains why the $\alpha = 1$ estimators had a reduced advantage. A second factor may be due to the underlying mechanism for the error correlations. Recall that the Lucas-Kanade tracker assumes that the image deformation between image I_2 and I_1 is planar plus zero-mean additive noise (of pixel intensities):

$$\mathrm{I}_2(\boldsymbol{x}) = \mathrm{I}_1(\boldsymbol{x} + \boldsymbol{d}) + \boldsymbol{v}(\boldsymbol{x}) \tag{5.98}$$

The tracker solves for the displacement $\boldsymbol{d}$ that minimizes the least squares cost function

$$\hat{\boldsymbol{d}} = \text{argmin}_{\boldsymbol{d}} \sum_{\boldsymbol{x} \in W} \left(\text{I}_2(\boldsymbol{x}) - \text{I}_1(\boldsymbol{x}+\boldsymbol{d})\right)^2 \tag{5.99}$$

which yields the necessary condition

$$\mathbf{0} = \sum_{\boldsymbol{x} \in W} \left(\text{I}_2(\boldsymbol{x}) - \text{I}_1(\boldsymbol{x}+\hat{\boldsymbol{d}})\right) \nabla \text{I}_1(\boldsymbol{x}+\hat{\boldsymbol{d}}) \tag{5.100}$$

If instead of the assumed deformation model, the true deformation model is

$$\text{I}_2(\boldsymbol{x}) = \text{I}_1(\boldsymbol{x}+\boldsymbol{d}+\text{f}(\boldsymbol{x})) + \boldsymbol{\nu}(\boldsymbol{x}) \tag{5.101}$$

for an arbitrary function $\text{f}(\boldsymbol{x})$, then this will introduce a systemic error. Let $\boldsymbol{d} = \hat{\boldsymbol{d}} - \boldsymbol{\delta}$ be the true displacement where $\boldsymbol{\delta}$ is the localization error. If both the added deformation and localization error are small then,

$$\text{I}_2(\boldsymbol{x}) \approx \text{I}_1(\boldsymbol{x}+\hat{\boldsymbol{d}}) + \nabla \text{I}_1(\boldsymbol{x}+\boldsymbol{d}) \left(\text{f}(\boldsymbol{x}) - \boldsymbol{\delta}\right) \tag{5.102}$$

Substituting the preceding two expressions into the necessary condition yields

$$\begin{aligned} \boldsymbol{\delta} = \sum_{\boldsymbol{x} \in W} \left(\nabla \text{I}_1(\boldsymbol{x}+\boldsymbol{d}) \nabla \text{I}_1(\boldsymbol{x}+\boldsymbol{d})^T\right)^{-1} \sum_{\boldsymbol{x} \in W} \Big(& \nabla \text{I}_1(\boldsymbol{x}+\boldsymbol{d}) \nabla \text{I}_1(\boldsymbol{x}+\boldsymbol{d})^T \text{f}(\boldsymbol{x}) \\ & + \nabla \text{I}_1(\boldsymbol{x}+\boldsymbol{d}) \boldsymbol{\nu}(\boldsymbol{x}) \Big) \end{aligned} \tag{5.103}$$

If $\text{f}(\boldsymbol{x}) = \mathbf{0}$ and the pixel intensity error is zero-mean, then the localization bias (found by taking the expectation of the above expression) is zero. A non-zero deformation function will introduce a non-zero bias. The deformation will be driven by the scene-relative motion which will vary between the different features. During a descent like motion, it is possible that the deformations in one part of the field-of-view will be equal and opposite to the deformations in the opposite part of the field-of-view (by

symmetry). Therefore the biases in localization error during descent may cancel each other out. This is less likely to occur during the orbit-like motion where all features move in the same direction which could also explain the difference in behavior. The next section addresses this.

5.11.7 Discrepancy Between Trajectory Types: Mixed-Simulation Analysis

A clear trend emerged in Section 5.11.5. The proposed models that capture error correlation in time had a clear advantage in accuracy over the standard model for the long-orbit trajectory class but not the descent trajectory class. A logical hypothesis is the standard model assumptions more accurately describe the measurements for the descent like motion than the autoregressive or correlated error model. This hypothesis is contradicted by the system identification performed in Section 5.2.1 (see Figure (5.4) in particular). In addition, the previous section demonstrated via a linear covariance analysis that the geometry of the motion can reduce the advantage of the $\alpha = 1$ models. This section presents further evidence that the geometry is a key factor.

As discussed above, it may be that the relative geometry of the motion and landmarks reduces the advantage of the more complicated autoregressive and correlated error models. However, it was concluded that more analysis was needed to confirm or reject this conjecture. Such an analysis was performed as follows using the same trajectories and feature tracking results used in the previous sections (i.e.

the evaluation dataset). The steps are as follows for each dataset:

1. Use the detected feature locations for the first image and the corresponding range measurements to define a set of landmarks (which are treated as scene-fixed).
2. Simulate an autoregressive noise process with $\alpha = 1$ and an IID noise process for all landmarks and images. Both are driven by Gaussian noise: $\sigma = 0.1$ for the autoregressive and $\sigma = 0.5$ for the IID.
3. Simulate perfect measurements of the landmarks using the same trajectory and camera parameters.
4. Create two simulated measurement sets: one by adding the autoregressive errors and one by adding the IID errors.
5. Use the feature tracking results associated with each landmark to determine visibility. Write all visible measurements (from each of the two sets) to their own file.

This was done for 100 datasets of both the long-orbit and descent trajectory. This represents a very controlled simulated dataset which mixes two different simulation methods: high-quality rendered images that give accurate landmark distributions and occlusion results, with a perfect error model (autoregressive and IID). Therefore it exactly captures the relative geometry of motion and landmarks and has a precise probabilistic model.

For each of the four groups of simulated datasets (each pair of one of two

trajectories and one of two simulated models), both the autoregressive estimator (Huber $\sigma = 0.1$, $\alpha = 1$) and the standard estimator (Huber $\sigma = 0.5$) were applied. If the only factor governing estimator accuracy were the underlying noise model, then one would expect the autoregressive estimator to always outperform the standard estimator when the errors are in fact autoregressive and vice versa. As it turns out, this was *not the case*!

To compare the estimation results, the RMS angle and position errors for each dataset within each of the four groups was computed. Comparisons were performed on each of the four groups using a WSRT. With each group, there are 100 batches of measurements. The same batch of measurements is passed to two different estimators, the output of which is treated as a paired sample.

Note that even within a particular class of trajectories, the exact motion, configuration of detected landmarks, and simulated random noise makes each dataset inherently different. By using the WSRT which is a paired-difference test, these confounding factors can be largely mitigated. The different groups are kept separate as a form of statistical blocking because we expect each group to behave differently (which is precisely why this analysis was performed).

The results are summarized in Table (5.7) . The large number of paired samples for each group (one hundred), led to a high level of confidence in certain results. In particular, the standard estimator outperforms the autoregressive estimator when the errors are IID at an essentially 100% confidence level for both motion types.

Dataset	Position	Angle
Long-Orbit IID	9.8×10^{-17}	9.8×10^{-17}
Long-Orbit AR	$1 - 3.6 \times 10^{-4}$	$1 - 5.3 \times 10^{-17}$
Descent IID	3.9×10^{-18}	3.9×10^{-18}
Descent AR	$1 - 0.28$	$1 - 0.31$

Table 5.7: Test results of a two-sided WSRT: A number near zero suggests that the standard estimator performed better than the autoregressive estimator and a number near one suggests the autoregressive estimator performed better than the standard estimator on the given block of datasets.

For autoregressive errors in the long-orbit case, the autoregressive estimator outperformed the standard estimator at a confidence level above 99.95% for both position and attitude errors. The big surprise comes from the descent trajectory with autoregressive errors where the test was inconclusive! While this was a surprising result, it is consistent with both the system modeling results of Section 5.2.1, which suggested strongly correlated errors, and the estimation results of Section 5.11.5 which showed similar performance for both estimators on the descent trajectory. Furthermore, testing in Section 5.10 showed that the fitted distributions were zero-mean which suggests there is no inter-landmark error correlation. This analysis compliments the linear covariance analysis in Figure (5.25) and gives strong evidence that even though the errors may be autoregressive for the descent trajectory, the addition of an autoregressive estimator does not provide a statistically significant advantage. It seems the only remaining possibility for the discrepancy is the geometry. An intuitive argument for this was given in Section 5.11.6 (see Figure (5.26)).

5.12 Summary

This section first presented evidence contradicting the IID assumption that is nearly universally assumed in vision-aided navigation systems. System identification results showed strong positive serial correlations in feature measurement errors. Four structural models were proposed to capture this. One of the models did so by allowing the landmarks to evolve according to a random walk process. The other three models did so by explicitly modeling the correlation. Under the assumption of each of the three model types, namely standard IID, landmark-walk, and correlated error, a technique was proposed to obtain samples from the error distribution using a high-quality rendering tool. The application of these techniques to several trajectory classes was used to obtain probabilistic models for the unknown error distribution. The resulting models were then tested.

Testing results for the long-orbit motion case demonstrated a significant performance advantage for the autoregressive model and correlated error model over the standard model: approximately 60 % reduction in angle errors and 30 % reduction in position errors. The improvement for the descent motion case was much smaller: approximately 10% for angle and position errors. A linear covariance analysis and a mixed-simulation demonstrated that the discrepancy was *not* because the autoregressive model or correlated error are less applicable to the descent motion case. Instead, the analysis and an intuitive argument suggested the geometry of the motion relative to the landmarks simply reduced the advantage of the more accurate

error models.

The developed models and their demonstrated advantages are a key contribution of this dissertation. Note that in developing a vision-aided navigation system, these models can be used regardless of what other sensors and information sources are available. This is an important advantage of the MLE and M-estimation framework which enable the models developed here to have broad impact.

6 DATA-DRIVEN ESTIMATION

This section discusses techniques for data-driven estimation. Estimators with a probabilistic optimality criterion require a probabilistic model relating the observations to the parameters of interest. For M-estimation, the model is used to design the cost function. As discussed previously in this dissertation, there is an inevitable lack of precision in models for real systems. This motivated the discussion of M-estimation as an extension of the MLE. The theory of M-estimation gives a foundation on which to design a cost function given a nominal model and a neighborhood around the model in which the true system is known to exist in. Unfortunately, the theory does not give a rigorous method to select either the nominal model or the neighborhood. This section presents one solution to that problem.

The class of measurement models considered in this section is the same class analyzed in Section 2. In particular, each observation is assumed to be the sum of a known deterministic function of the parameters and an additive zero-mean error. Furthermore, it is assumed that the distribution of the additive error is unknown *a priori*. This section seeks to design M-estimator cost functions for a given measurement model when a large number of samples from the unknown distribution are available. Such samples can be obtained through either experimentation or high-fidelity simulation of the system of interest. A method to accomplish such a goal is the subject of the first section of this section which includes sample problems as a

proof-of-concept.

An extension of the first section is to design estimators with the aid of *auxiliary measurements*. The term *auxiliary measurements* is used here to indicate any data that is paired with the primary observation (i.e. the one typically used in the estimator) that can provide information about the underlying distribution that the observation came from. Auxiliary measurements occur in a variety of applications including computer vision. In particular, in addition to measuring feature locations, feature tracking methods also output various values related to matching metrics and detection scores. It is hypothesized that the underlying distribution of errors depends of such metrics (ex: large matching error implies larger localization error). If this is in fact the case, that then such metrics and scores would meet the definition of an auxiliary measurement. Evidence to support this hypothesis is given in the second section.

The third section of this section then presents a method to design estimator cost functions that depend on the auxiliary measurements. In the context of MLE, this can be viewed as a Maximum Conditional Likelihood Estimate (MCLE) where the likelihood is further conditioned on the auxiliary measurement (in addition to the usual parameters of interest). The term MCLE will also be used to describe estimators related to the true MCLE in the same way that M-estimators are related to the true MLE. Examples problems demonstrate the feasibility of such methods.

The section concludes by applying the methods developed in this section to

the vision-aided navigation problem. The auxiliary measurements identified in the second section, are used to train an estimator which is then applied to the vision-aided navigation problem.

6.1 Minimum Variance Estimators

At first, developing a single method to design estimators given only the error samples seems intractable. However, there is a critical insight that greatly simplifies the problem. In particular, Equation (2.118) shows that the covariance of parameter errors in the nonlinear regression problem is asymptotically proportional to the location problem variance:

$$v \equiv \frac{\mathbb{E}\left\{\psi^2\right\}}{\mathbb{E}\left\{\psi'\right\}^2} \tag{6.1}$$

Therefore for a given set of measurement samples, an estimator can be designed to be minimum variance on the location problem. The same estimator will then be minimum variance for the general nonlinear regression problems.

The analytical results for M-estimate covariance in nonlinear regression suggests the following method. First parameterize an arbitrary cost function: $\rho(x;\boldsymbol{\beta})$ where $\boldsymbol{\beta}$ parameterize the function. Second, optimize the cost function parameters so that

$$\hat{\boldsymbol{\beta}} = \operatorname{argmin} \frac{\mathbb{E}\left\{(\rho'(x;\boldsymbol{\beta}))^2\right\}}{\mathbb{E}\left\{\rho''(x;\boldsymbol{\beta})\right\}^2} \tag{6.2}$$

where the expectation is approximated using a set of error samples $\{x_i\}_{i=1}^N$ for N samples. Third and finally, evaluate the designed cost function $\rho(x;\hat{\boldsymbol{\beta}})$ on the problem

of interest.

The first step requires picking a parametric equation to serve as a cost function. Such a function should be

1. Flexible enough to provide good performance on a variety of distributions.
2. Small enough number of parameters to be well-defined for the amount of data available.
3. Be well-suited for the numerical solution of estimates (ex: convex).

The first two items are competing interests. The theoretical results of Huber and Hampel can provide guidance in selecting the function. In particular the Huber function with quadratic *inlier-region* and $\mathscr{L}_1$ *outlier-region* clearly has outstanding theoretical properties. However, redescending cost functions (those with a non-monotonically increasing derivative) can provide added insensitivity to gross outliers. This motivates the following cost function:

$$\alpha_1 \equiv \beta_1 \tag{6.3}$$

$$\alpha_2 \equiv \beta_1 + \beta_2 \tag{6.4}$$

$$\alpha_3 \equiv \beta_1 + \beta_2 + \beta_3 \tag{6.5}$$

$$\rho(x;\boldsymbol{\beta}) = \begin{cases} \frac{1}{2}x^2/\alpha_0 & |x| \le \alpha_1 \\ |x|\alpha_1/\alpha_0 - \frac{1}{2}\alpha_1^2/\alpha_0 & \alpha_1 < |x| \le \alpha_2 \\ a(x-\alpha_2)^2 + b(x-\alpha_2) + c & \alpha_2 < |x| \le \alpha_3 \\ a(\alpha_3-\alpha_2)^2 + b(\alpha_3-\alpha_2) + c & |x| > \alpha_3 \end{cases} \tag{6.6}$$

where the constants a, b, and c are functions of $\boldsymbol{\beta}$, chosen such that the cost is continuous, the derivative of the cost is continuous, and the cost obtains a zero slope at $x = \beta_3$. The definition of α_1, α_2, and α_3 in terms of $\boldsymbol{\beta}$ ensures that the transition regions are correctly ordered, $\alpha_1 < \alpha_2 < \alpha_3$, whenever $\boldsymbol{\beta} > \mathbf{0}$ which is easier to enforce in the optimization routine as a lower bound on the parameters instead of an inequality constraint between parameters.

The selected cost is similar to the Huber cost except that it gradually *rolls off* spatially to a constant cost. Furthermore, the parameters were selected such that boundaries of the different cost regions are independent of the scale β_0. If the same cost function is used for all observations in an estimator, then the estimate of independent of β_0 which will have important implications in later sections. In addition, the β_1 term can be set arbitrarily close to zero with β_2 and β_3 arbitrarily large which emulates an L_1 cost without causing any singularities. Although this cost is non-convex, the size of the convex region can be made as large as desired.

The second high-level step of the method is to optimize the cost function parameters to minimize the asymptotic variance. Given N samples, $\{x_i\}_{i=1}^N$, this requires minimizing

$$\frac{\sum_{i=1}^N \rho'(x_i;\boldsymbol{\beta})^2}{\left(\sum_{i=1}^N \rho''(x_i;\boldsymbol{\beta})\right)^2} \tag{6.7}$$

The finite number of samples can lead to a number of difficulties with this optimization criteria.

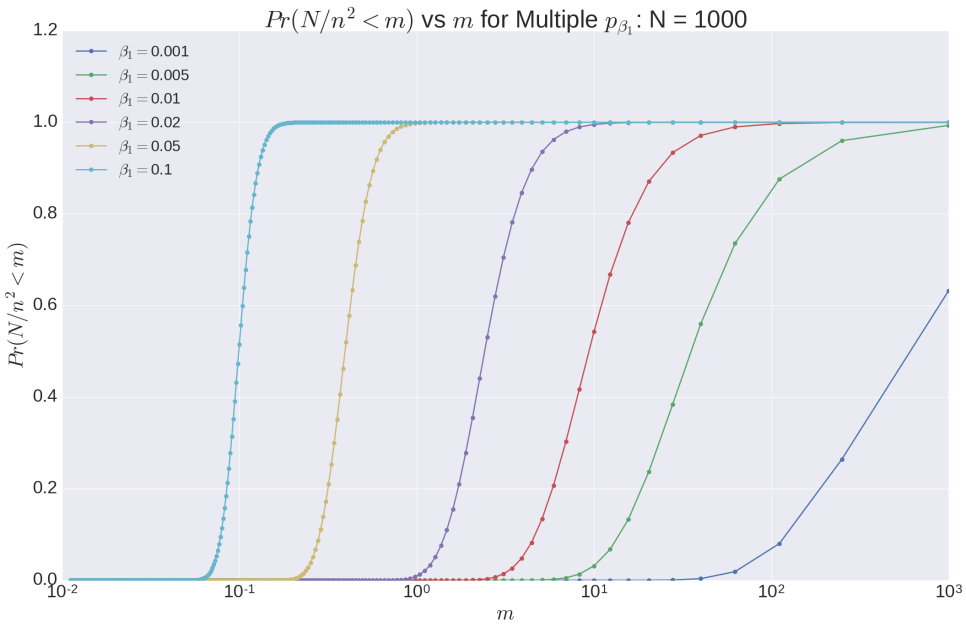
To see this difficulty, consider the case when β_2 and β_3 are infinite so that the

proposed cost is equal to the Huber cost. Assume the true samples come from a distribution with CDF $\mathrm{F}_x(x)$. For a given β_1, the probability of a sample occurring such that $|x| < \beta_1$ is $p_{\beta_1} = \mathrm{F}_x(\beta_1) - \mathrm{F}_x(-\beta_1)$. For N total samples, the number of samples satisfying this condition, say n, is distributed as a binomial distribution: $\Pr(n) = \binom{N}{n} p_{\beta_1}^n (1-p_{\beta_1})^{N-n}$. The empirical variance in this case is

$$v = \frac{N}{n^2}\left((N-n)\beta_1^2 + \sum_{\star} x_i^2\right) \tag{6.8}$$

where the starred summation is over the samples falling in the $[-\beta_1, \beta_1]$ interval. Clearly, if $n = 0$ this equation is singular. By the binomial distribution, this occurs with probability equal to $(1-p_{\beta_1})^N$. In words, if N is small and the *inlier region* is small for the given distribution, there will be a large chance of a singularity. Furthermore, even when $n > 0$, at small n the variance of v is high. The variability of n for small n will dominate the variability of v via the $\frac{N}{n^2}$ since $N \gg n$ A plot of the cumulative mass function (CMF) of $\frac{N}{n^2}$ at select values of p_{β_1} is shown in Figure (6.1) for $N = 1000$. From this plot, it is clear that $\frac{N}{n^2}$ has increasing variability for decreasing p_{β_1}. This can cause the variance to be both over-estimated and under-estimated. As a result, the value of β_1 should be restricted so that a sufficient number of samples fall into the central region of the cost function. By requiring at least 20 % of samples to fall into the central region, the variability of $\frac{N}{n^2}$ is reduced to $[0.018, 0.035]$ for the 99% confidence interval. This in contrast to a case such as when $p_{\beta_1} = 0.01$ which has a 99% confidence interval of approximately $[2.6, 100]$: a

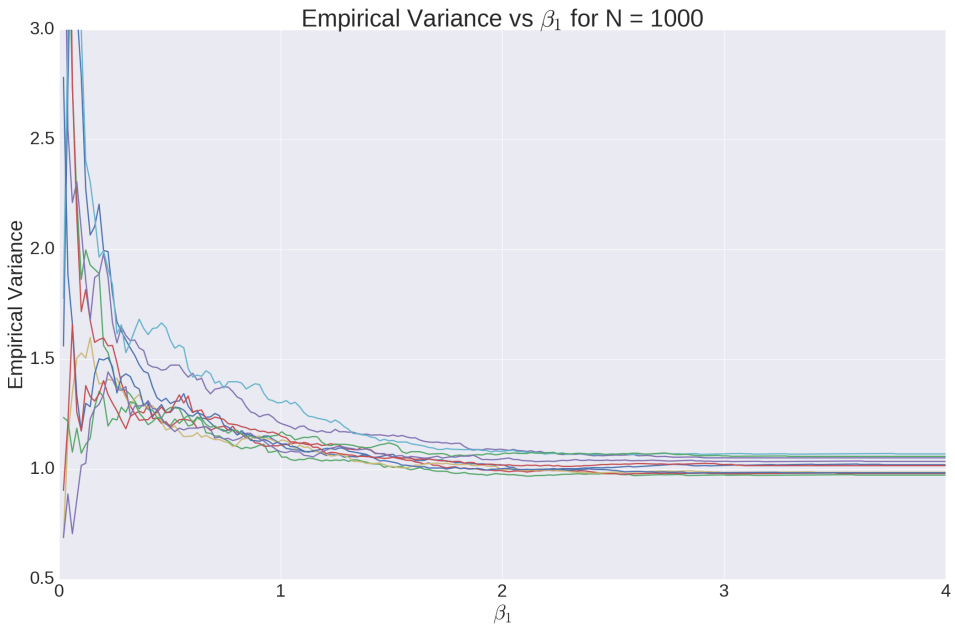
Figure 6.1: CMF of $\frac{N}{n^2}$ plotted for multiple values of p_{β_1}. When n is small ($n/N < 0.1$), this term dominates the variability in the empirical variance.



much wider range. Note that these numbers were obtained from the plotted lines in Figure (6.1).

A second and numerical example of this issue is given in Figure (6.2) and Figure (6.2). In these figures, 10 trials were performed where $N = 1000$ and $N = 10000$ samples were obtained from a Gaussian distribution (zero mean and unit variance) and used to compute the variance over a range of β_1 values (on each trial). The lines in the figure each represent a trial. In theory, the minimum variance value of β_1 is infinity (to obtain a least squares cost function). Of course, any β_1 larger than the largest sample will give the same empirical variance. Note that in the figures, especially for the case of a smaller N, there is a lot of variability when β_1

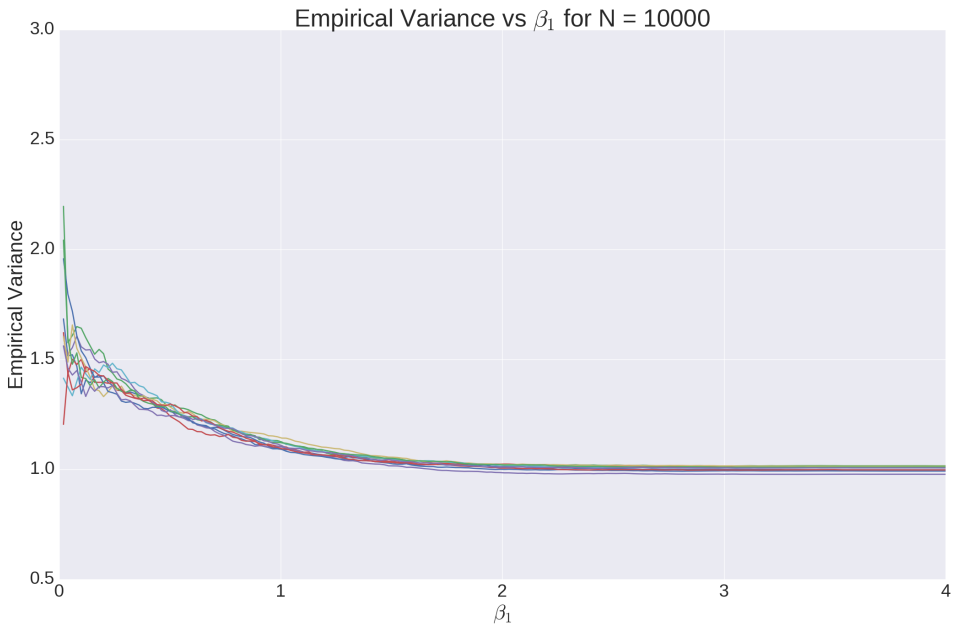
Figure 6.2: Several trials of empirical variance computed on 1000 Gaussian samples as a function of β_1 when $\boldsymbol{\beta} = [\beta_1, \infty, \infty]$.



is small. This leads to many local minima and can even lead to an empirically variance-optimal solution that is nowhere near the truly variance-optimal solution: zero instead of infinity! Again, this can simply be avoided (or at least made very improbable) by requiring at least 20% of samples to be in the central region.

The numerical issues with optimization are only partially remedied by providing a lower bound on β_1. There is still the fact that there will be many local minima and that gradient computations are unreliable due to the finite number of samples. One optimization method to find a solution is known as *particle swarm optimization* (PSO) [106]. PSO is a heuristic optimization method that searches for a global minima by heuristically proposing and modifying a large number of can-

Figure 6.3: Several trials of empirical variance computed on 10,000 Gaussian samples as a function of β_1 when $\boldsymbol{\beta} = [\beta_1, \infty, \infty]$.



didate solutions known as *particles*. The particles are initialized randomly in the search space and are iteratively adjusted pseudo-randomly based on the historical best solution of each particle individually and the best solution among all particles. The *Python PYSWARM* package implementation of PSO was used in all numerical examples below. Note that the particular parameterization used here guarantees that the transition-values between the four regions of the cost function are properly ordered by setting a lower-bound of zero on each element of $\boldsymbol{\beta}$ *without having to add any constraints*. This greatly simplifies the optimization procedure.

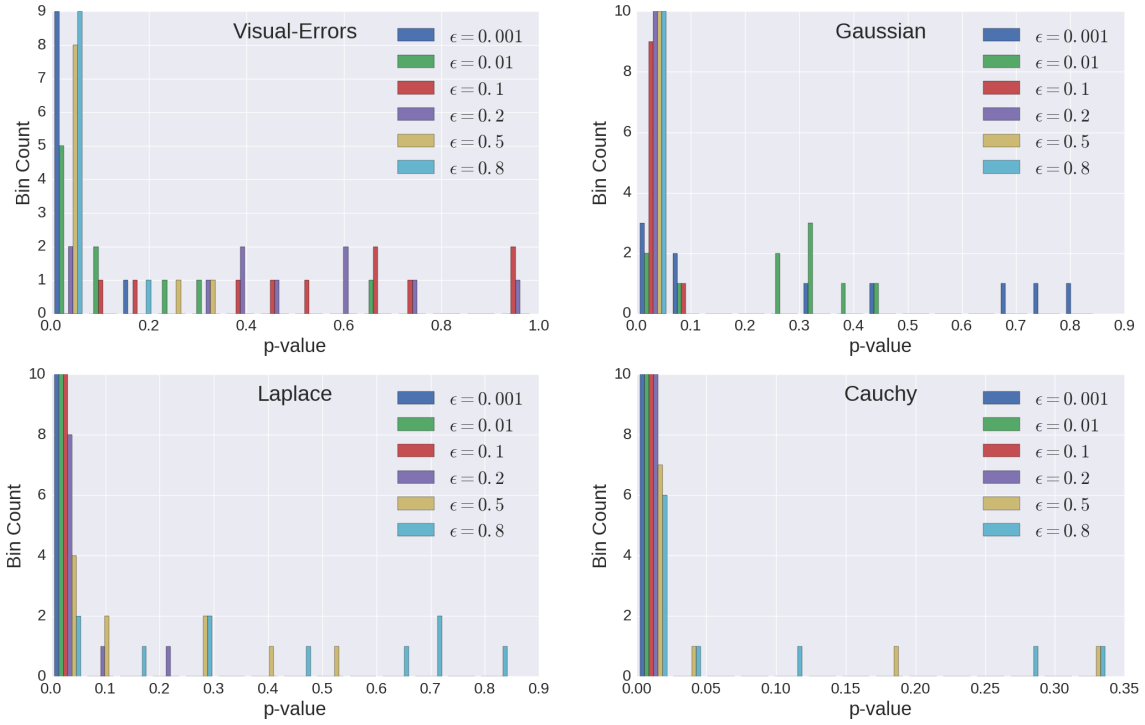
This concludes the section discussing the overall methodology for minimum variance estimators. The following subsection presents the results of several numer-

ical examples that use this concept.

6.1.1 A Numerical Example

This subsection provides numerical examples which are structured as follows. In each example, a particular true underlying distribution is used. Ten trials of generating samples, training an estimator on the samples, and evaluating the trained estimator are performed. For each trial $N = 10000$ samples are used to train the estimator. To evaluate the estimator in each trial, 400 sub-trials are performed. In each sub-trial, $n = 40$ measurements of the form $y_i = mx_i + b + \nu_i$ are simulated where m and b are parameters to be estimated, the x_i are uniformly distributed between -50 and 50, and the ν_i are samples obtained from the same distribution used to train the estimator. In all cases, the true parameters are $m = 0$ and $b = 0$. Additionally in each sub-trial, the estimate is obtained under the trained estimator and a number of other estimators. The other estimators are the Huber estimator with scale determined by the MADN (see Equation (2.137)) and several instances of the cutoff parameter (k in Equation (2.65). The values of k are set to be minimax optimal for an ε-contaminated neighborhood about a normal distribution with ε being equal to 0.001, 0.01, 0.1, 0.2, 0.5, and 0.8. Within any sub-trial, the difference in error between the trained-estimator and all six other estimators (each corresponding to an ε) are computed to form paired-differences which are then used in a WSRT. The RMS value for each estimator in each trial is then used to compute a number of statistics which are compared among the estimators. The chosen true distributions

Figure 6.4: WSRT p-values from 10 trials (on four different distributions) comparing a trained-estimator to various minimax optimal Huber estimators.

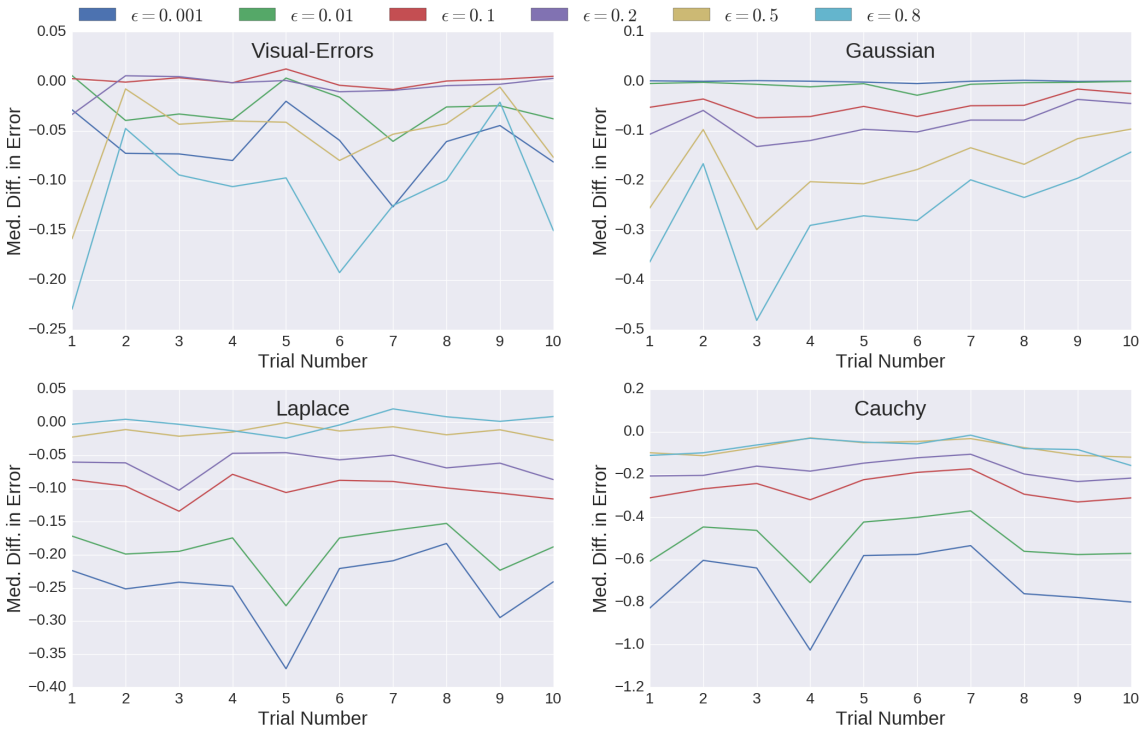


in the examples are the Gaussian, Laplace, and Cauchy for the first three. In the fourth example, the samples are generated from the dataset presented in Section 5.10.

The resulting p-values of the WSRT are plotted in Figure (6.4). In addition, the normalized difference in the median error between the trained-estimator and all others for each trial, $(\text{Med}(e_{trained}) - \text{Med}(e_{other}))/\text{Med}(e_{trained})$, is shown in Figure (6.5).

Several trends should be discussed with respect to to these two figures. For the visual-error samples, the trained-estimator performed much better than the Huber

Figure 6.5: Normalized median difference in error, $\frac{(\text{Med}(e_{trained}) - \text{Med}(e_{other}))}{\text{Med}(e_{trained})}$, between the trained estimator and all others for each of 10 trials on four different distributions.



estimators with small ε which is indicative of heavy contamination. For $\varepsilon = 0.001$, the p-values ranged from between 1.06×10^{-6} to 0.200 with the second highest p-value being 0.036. On the other hand, for $\varepsilon = 0.8$, the p-values ranged from between 7.42×10^{-10} to 0.157 with the second highest p-value being 6.64×10^{-5}. In the middle were cases like $\varepsilon = 0.1$ which was the best-tuned Huber estimator for this problem: the p-values ranged from between 0.132 to 0.987. The mixed results suggest that the two estimators are nearly equivalent. This is unsurprising as β_1 was estimated between 0.08 and 0.10 with a large β_2 and β_3 whereas the Huber cutoff for $\varepsilon = 0.1$ was 0.092 (since the MADN scale was 0.081 and $k = 1.14$ for $\varepsilon = 0.1$): they are

essentially the same cost.

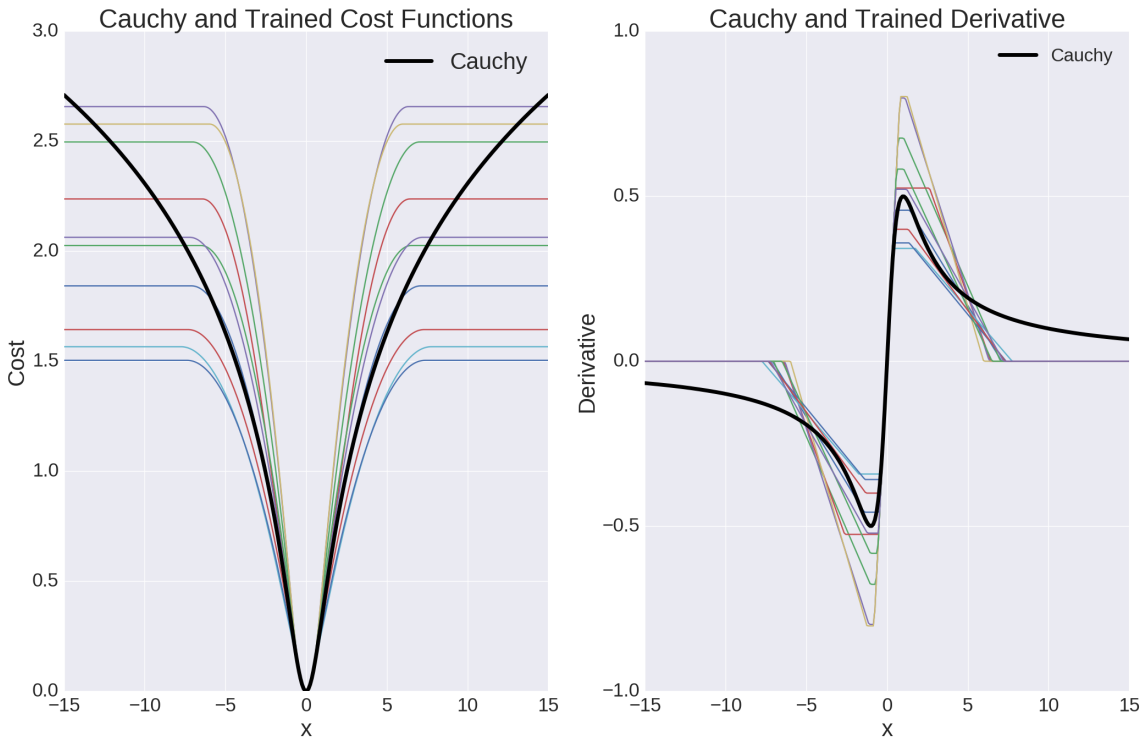
For the Gaussian errors, the Huber cost at $\varepsilon = 0.001$ and $\varepsilon = 0.01$ gave nearly identical performance to the trained-estimator as the costs were nearly the same in this case. In addition, for $\varepsilon > 0.01$ cases, there was a clear advantage for the trained estimator. The reason for this is the efficiency loss caused by truncated the quadratic inlier region. In all trials, the trained-estimator had $\beta_1 > 2.28$ which captures 97.7% of standard normal errors in the quadratic region.

For Laplace-distributed errors, the truly optimal solution is the Huber cost with $k = 0$ (i.e. an $\mathscr{L}_1$-norm). This is closely approximated by the Huber costs at $\varepsilon = 0.8$ and $\varepsilon = 0.5$. The trained estimator selected $\beta_1 \approx 0.21$, which was very close to the lower bound, and $\beta_2 > 4.1$ in each case. This means that for the unit-scale Laplace, at least 98.6% of the distribution was captured in the quadratic or $\mathscr{L}_1$ region of the cost function. As a result, the performance between the trained estimators and the Huber costs at $\varepsilon = 0.8$ and $\varepsilon = 0.5$ was nearly identical. However, the trained estimator had a clear advantage for lower ε Huber costs.

Finally, for Cauchy-distributed errors, the optimal solution is not in the family of parameterized estimators. A plot of the trained cost and derivative functions alongside a scaled-version of the Cauchy MLE cost is given in Figure (6.6). Qualitatively, the costs appear similar.

Quantitatively, the WSRT p-values show that the trained-estimator outperforms the Huber cost for all values of ε. For $\varepsilon < 0.5$, there is not a single case where

Figure 6.6: Trained cost and derivative from 10 trials of 10000 Cauchy-distributed errors plotted with the true Cauchy MLE cost and derivative.



the p-value exceeds 0.01. For $\varepsilon = 0.5$ and $\varepsilon = 0.8$, there are two and four trials respectively that had moderate p-values: between 0.05 and 0.35. A simple way to aggregate the p-values over the 10 trials is as follows. Under the null hypothesis of the WSRT, namely that the two error sources have the same distribution, we reject the null hypothesis, with α-confidence, in favor of the alternative hypothesis that the distributions have different means if the test statistic is less than α. The false rejection rate should be $1-\alpha$. The number of false rejections should follow a binomial distribution with parameters $N = 10$, $p = \alpha$. For $\alpha = 0.05$, under the null-hypothesis, there are three rejections out of ten trials for both values of ε (where the p-value is

less than $\alpha/2 = 0.025$ for a one-sided test). The binomial CMF at $n = 3$ for $N = 10$, $p = 0.05$ is 0.9990. Therefore at the 0.1% confidence level, the null hypothesis can be rejected for the aggregate test for both values of ε in question.

6.2 Auxiliary Measurements for Visual Features

This section discusses auxiliary measurements in the context of vision-aided navigation. At the start of the section, auxiliary measurements were defined as observed values that are paired with each primary measurement (i.e. feature location) and provide information about the distribution of the primary measurement. It is hypothesized that the scale of image-space feature localization errors will depend on various scores related to how well an image of a landmark matches its previous image. Several candidates for matching scores were considered. Before discussing the particular candidates, the characteristics of a good candidate are described.

A good candidate matching score should have a clear relationship to error scale. In particular, the conditional distribution $\mathrm{p}(\nu_i|z_i)$ should be have a strong dependence on the auxiliary measurement z_i. To evaluate the matching score candidates, each score is computed along with a corresponding sample of visual errors under the assumption of an autoregressive error model with $\alpha = 1$ (see Section 5.10). The paired samples $\{\nu_i, z_i\}_{i=1}^{N}$ are then binned into sets of equal probability mass: each bin contains 1/30-th of the samples giving > 10000 samples per bin. In each bin, the median of the $\mathscr{L}_1$ norm of feature localization error is computed for each bin

and plotted against the bin center. A *bad* candidate score would show little-to-no dependence of the localization error scale versus the bin center: no information of error scale is provided. The goal is to find a matching score where the relationship with the error scale is clear.

A number of proposed candidates were based on the sum of squared differences (SSD) correlation image. This is defined as

$$J(u,v) \equiv \sum_{u'} \sum_{v'} \left(I_2(u',u') - I_1(u+u', v+v') \right)^2 \tag{6.9}$$

where I_2 is a 31×31 pixel-patch centered on the localized feature in the current image and I_1 is a 36×36 pixel-patch centered on the localized feature in the previous image. The resulting image J is centered on $0,0$ and defined on the domain $\{-2,1,0,1,2\} \times \{-2,1,0,1,2\}$. The two candidates of interest that use the 5×5 pixel image J are:

1. SSD at feature location measurement: $J(0,0)$
2. $\mathscr{L}_1$-norm of centroid of SSD image: $\left| \frac{\sum_{u,v} uJ(u,v)}{\sum_{u,v} J(u,v)} \right| + \left| \frac{\sum_{u,v} vJ(u,v)}{\sum_{u,v} J(u,v)} \right|$

It is expected that a low SSD at the feature location measurement and an SSD image with a centroid at $(0,0)$ corresponds to a good feature match and hence small localization errors.

A third candidate is based on the Lucas-Kanade tracker equations and the *sandwich covariance estimator* []. The sandwich covariance estimates the variability in an M-estimate. It can be computed directly on the data without needed to assume a particular error distribution. In this case, the sandwich variance is

$$S \equiv A^{-1} V A^{-1} \tag{6.10}$$

$$A \equiv \sum_{u,v} \nabla^T I_2(u,v) \nabla I_2(u,v) + (I_2(u,v) - I_1(u,v)) \nabla^2 I_2(u,v) \tag{6.11}$$

$$V \equiv \sum_{u,v} \nabla^T I_2(u,v) (I_2(u,v) - I_1(u,v))^2 \tag{6.12}$$

where $\nabla I_2(u,v)$ is the 1×2 image gradient and $\nabla^2 I_2(u,v)$ is the 2×2 image Hessian. The trace of the sandwich covariance S gives the sandwich-variance of the computed feature location and is tested as a candidate auxiliary measurement.

The plots for the three candidates discussed above are evaluated on 100 sets of 20 images for the evaluation datasets of the three trajectory classes discussed in Section 5.1. The resulting plots for the median $\mathscr{L}_1$ norm of localization error versus the auxiliary measurement is shown in Figure (6.7), Figure (6.8), and Figure (6.9). A scatter plot of the individual samples is overlaid.

Figure (6.7) shows the results for the SSD error at the feature location measurement. The variation for the long-orbit and short-orbit trajectory classes is very clear. In the short-orbit case, the error scale varies from 0.05 to 0.17 pixels. In the long-orbit case, the error scale varies from 0.07 to 0.24 pixels. On the other hand, the trend is not very clear for the descent case.

Figure (6.8) shows the results for the SSD error at the feature location measurement. The variation for the long-orbit and short-orbit trajectory classes is very clear. In the short-orbit case, the error scale varies from 0.08 to 0.16 pixels. In the long-orbit case, the error scale varies from 0.10 to 0.22 pixels. In the descent case, the error scale varies from 0.09 to 0.22 pixels.

Figure (6.9) shows the results for the trace of the sandwich covariance. The

Figure 6.7: The median $\mathscr{L}_1$ norm of localization error versus the SSD evaluated at the feature location measurement.

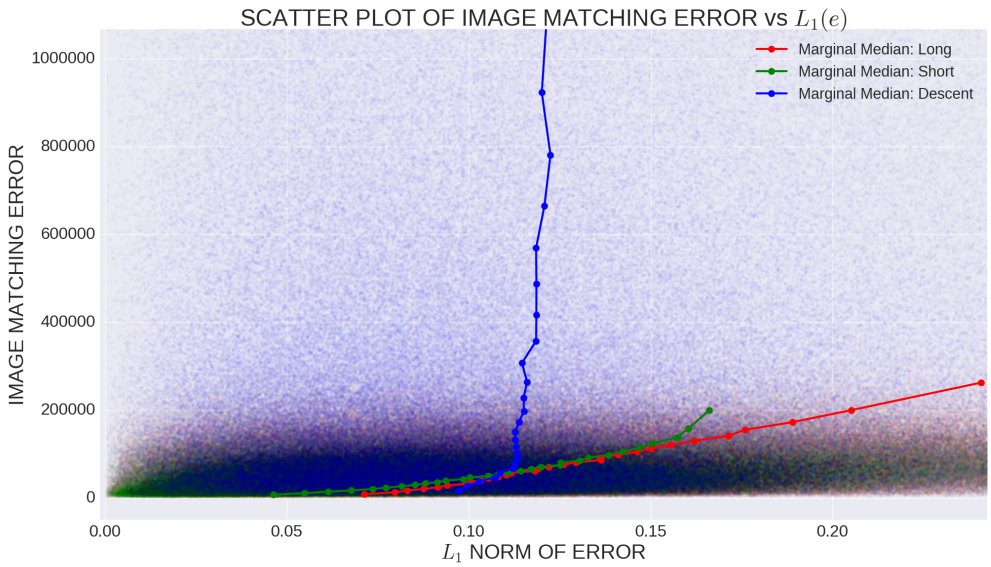


Figure 6.8: The median $\mathscr{L}_1$ norm of localization error versus the $\mathscr{L}_1$ norm of SSD centroid.

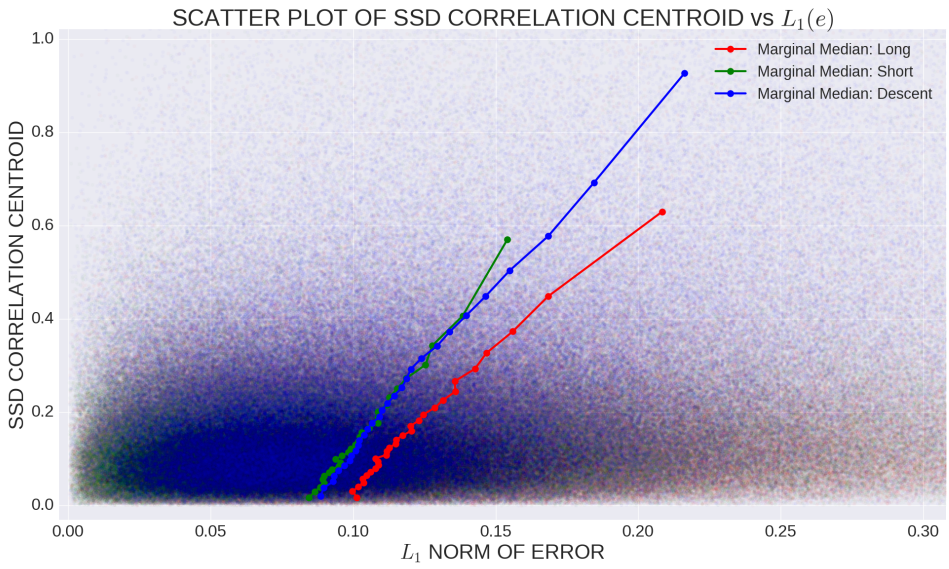
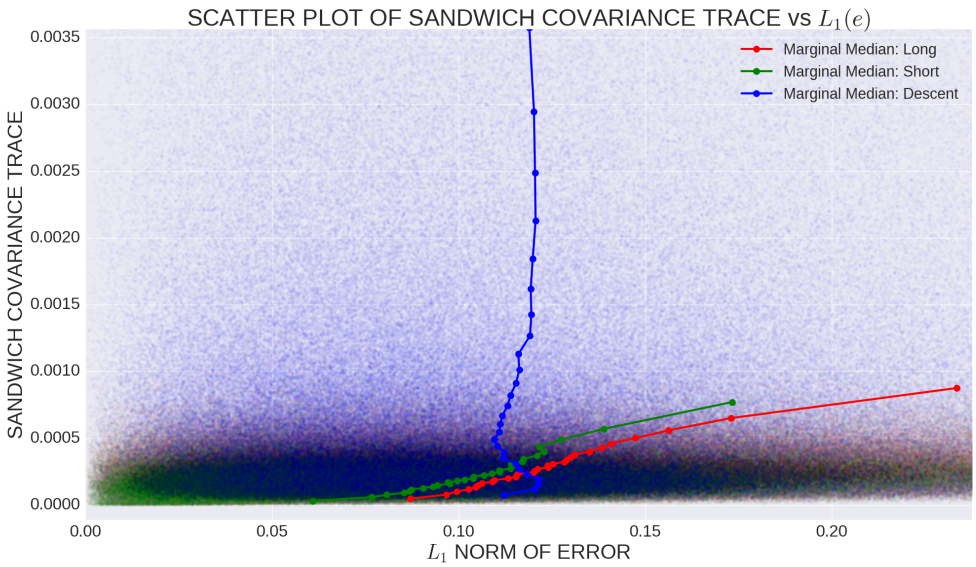


Figure 6.9: The median $\mathscr{L}_1$ norm of localization error versus the trace of the sandwich covariance.



variation for the long-orbit and short-orbit trajectory classes is very clear. In the short-orbit case, the error scale varies from 0.06 to 0.17 pixels. In the long-orbit case, the error scale varies from 0.08 to 0.24 pixels. On the other hand, the trend is not very clear for the descent case.

In summary of the above results, the SSD centroid is the best candidate across all three motion cases. It is therefore chosen for testing in the examples below.

This section has presented evidence to suggest the existence of auxiliary measurements in vision-aided navigation. This evidence motivates the next section which discusses the theoretical background for MCLE and provides simple numerical algorithms.

6.3 Maximum Conditional Likelihood Estimation

This section introduces MCLE with auxiliary measurements. The theoretical basis for this is simple. The MLE finds the estimate maximizing the likelihood defined as

$$\mathscr{L} \equiv \prod_{i=1}^{N} \mathrm{p}_{\nu_i}(y_i - \mathrm{h}_i(\boldsymbol{\theta})) \tag{6.13}$$

for the measurement model

$$y_i = \mathrm{h}_i(\boldsymbol{\theta}) + \nu_i \tag{6.14}$$

$$\nu_i \sim \mathrm{p}_{\nu_i}(\nu_i) \tag{6.15}$$

If the distribution of the error ν_i depends on a known (or observed) random variable z_i, then the conditional likelihood

$$\mathscr{C} \equiv \prod_{i=1}^{N} \mathrm{p}_{\nu_i|z_i}(y_i - \mathrm{h}_i(\boldsymbol{\theta})|z_i) \tag{6.16}$$

should me maximized.

Note that the log-conditional-likelihood is simply

$$\log \mathscr{C} \equiv \sum_{i=1}^{N} \log \mathrm{p}_{\nu_i|z_i}(y_i - \mathrm{h}_i(\boldsymbol{\theta})|z_i) \tag{6.17}$$

$$= \sum_{i=1}^{N} \log \left(\mathrm{p}_{\nu_i,z_i}(y_i - \mathrm{h}_i(\boldsymbol{\theta}), z_i)/\mathrm{p}_{z_i}(z_i)\right) \tag{6.18}$$

$$= \sum_{i=1}^{N} \log \mathrm{p}_{\nu_i,z_i}(y_i - \mathrm{h}_i(\boldsymbol{\theta}), z_i) - \log \mathrm{p}_{z_i}(z_i) \tag{6.19}$$

$$\sim \sum_{i=1}^{N} \log \mathrm{p}_{\nu_i,z_i}(y_i - \mathrm{h}_i(\boldsymbol{\theta}), z_i) \tag{6.20}$$

where the relationship between conditional and joint distributions $\mathrm{p}(x|y) = \mathrm{p}(x,y)/\mathrm{p}(y)$ is used. The last expression, although not strictly equal to the log-

conditional-likelihood, is an equivalent cost function: it is minimized by the same argument. This result has an important implication: either the joint or conditional distribution can be used to design the MCLE.

In the problems of interest to this dissertation, the analytical form for both the joint and conditional distributions are assumed to be unknown. Rather than estimating either of the distributions and then designing a cost function around the distribution, the cost function is designed directly. To do so, the method presented in Section 6.1 must be extended. In particular, the cost function parameters $\boldsymbol{\beta}$ must become a function of the auxiliary measurements z_i. Several options were considered to accomplish this.

Similar to the discussion in Section 6.1, the estimator parameters should be of low enough dimension that they can be amenable to practical optimization method and that they can be trained on the available data without over-fitting. In addition, the mapping from the auxiliary measurements to the cost function parameters should have low computational cost. One simple method to address both these needs is as follows.

First, given N samples of error and the auxiliary measurement, $\{(\boldsymbol{\nu}_i, z_i)\}_{i=1}^{N}$, select n_t tie-points at the $100\%[\frac{1}{2n_t}, \frac{3}{2n_t}, \frac{5}{2n_t}, \ldots, \frac{2n_t-1}{2n_t}]$ percentiles of the $\{z_i\}_{i=1}^{N}$ samples. Let the j'th tie-points be represented by $z_j^\star$. For any value z_i, in training or evaluation, the index j such that $|z_i - z_j^\star|$ is minimized, is the index of the tie-point associated with z_i. Then, each tie-point can have its own parameter set $\boldsymbol{\beta}_j$ associated

with it. This is analogous to estimating a conditional distribution (or cost function) by binning the auxiliary measurements and computing a PDF (or cost function) for each bin separately. Such a method has two immediate benefits. First, we can guarantee a constant number of samples, N/n_t, per bin during training. Second, the association of each measurement with a tie-point, and hence a set of cost function parameters, is computationally efficient. A third and more important benefit warrants further discussion.

First note that the binning scheme can be viewed as treating the measurements as coming from n_t different sources: each with their own distribution. The MCLE cost can then be expressed as

$$\log \mathscr{C} \sim \sum_{j=1}^{n_t} \mathscr{C}_j \tag{6.21}$$

$$\mathscr{C}_j = \sum_{i=1}^{N/n_t} \log \mathrm{p}_{\mathrm{v}_{j_i}, z_j^\star}(y_{j_i} - \mathrm{h}_{j_i}(\boldsymbol{\theta}), z_j^\star) \tag{6.22}$$

where the summation in the top equation sums over the different tie-points and the summation in the bottom equation sums up the measurements associated with the j'th tie-point. Note that $\mathscr{C}_j$ defines the MLE cost if only measurements associated with the j'th tie-point are used in the estimator. This shows that if the estimator was designed by first estimating the conditional PDFs, then each one could be estimated independently. This would reduce the problem into n_t smaller problems. However, the goal here is to design the cost directly. Nevertheless, a similar reduction in complexity can be made.

First, the β_1, β_2, β_3 parameters are designed to be minimum-variance for each

tie-point individually. Then the scale β_0, which was neglected in Section 6.1 because it did not alter the estimator output when one cost function is used for all input measurements, is estimated jointly for all tie-points. With this approach, n_t three-dimensional and one $n_t - 1$-dimensional optimization problems must be solved instead of a single $4n_t - 1$-dimensional optimization problem (the n_t β_0 parameters are unique to a common scale so β_0 can be set to an arbitrary value for one tie-point).

The single $n_t - 1$-dimensional optimization problem is still expensive. A simple heuristic, which must be validated, can eliminate the need for this optimization problem without significantly increasing the estimator variance over the optimum. In particular, consider the n_t trained cost functions with optimal variances $v_j^\star$. If a batch of measurements were available, then one could come up with n_t separate estimates of location, each with variance $\frac{v_j^\star}{n_j}$ where n_j is the number of measurements associated with the j-th tie point. Since the n_t estimates are asymptotically Gaussian, then a single location estimate can be obtained by computing a weighted average of the n_t estimates, each with weight $\frac{v_j^\star}{n_j}$. This is equivalent to solving for a single M-estimate with $\beta_0 = v_j^\star$ for each tie-point. Therefore, instead of solving the $n_t - 1$-dimensional optimization, the optimal variance resulting for each of the n_t three-dimensional optimization problems can be used to set the β_0 parameters.

This concludes the background section on the MCLE. To summarize, the method in Section 6.1 was extended by binning the auxiliary measurements into n_t bins. By design of the binning locations, each bin has approximately equal prob-

ability mass. For each bin, a three-parameter cost function is designed for minimum asymptotic variance, independent of all other bins. Two different methods of setting the scale of each cost function, necessary to tie the n_t cost functions together, are proposed. The next section gives numerical examples to evaluate these two methods and the overall concept.

6.3.1 Numerical Examples

This section presents numerical examples of the data-driven MCLE. The first numerical example is used to compare the two methods for optimizing scale which were discussed at the end of the previous section. In this example, $N = 10^6$ auxiliary measurements, z_i are generated as uniform random variables on the interval $[1, 10]$. Then error samples v_i are generated by scaling a standard normal random variable by z_i. The theoretically optimal estimator in this case is a weighted least-squares estimator with weights $1/z_i^2$. The binning of the auxiliary measurements into n_t bins centered on $z_j^\star$ suggests that the trained weights for the j-th tie-point should be between $(1/z_{j+1}^\star)^2$ and $(1/z_{j-1}^\star)^2$. Both methods discussed in the previous section are used to estimate the cost function parameters. In both methods, the β_1, β_2, and β_3 parameters for each cost are found first. In the first method, the variance from the first step for each cost is used for β_0. In the second method, the β_0 parameters are explicitly optimized. Note that $n_t = 10$ in this example and the same *Python* *PYSWARM* PSO optimization is used for all optimizations. The results are summarized in the left-hand columns of Table (6.1) . Note that the β_0 values are unique up

Table 6.1: Resulting variance and β_0 value for ten training trials under Gaussian, Laplace, and Cauchy distributions.

	Gauss		**Laplace**		**Cauchy**	
j	v	β_0	v	β_0	v	β_0
1	2.11	2.01	2.11	1.98	3.97	2.94
2	5.70	5.70	5.79	5.43	10.47	10.68
3	10.43	10.39	10.91	10.80	21.22	20.79
4	17.44	17.46	18.94	22.32	37.19	29.16
5	24.90	24.81	28.01	32.34	48.05	47.88
6	35.91	35.44	38.56	36.92	67.95	44.84
7	46.11	46.21	46.89	47.16	95.58	84.77
8	60.61	60.66	63.41	64.92	115.45	98.32
9	75.75	75.89	81.56	92.23	146.59	111.29
10	93.59	93.59	105.11	105.11	177.09	177.09

to scale and are therefore scaled to make the last element The same test is repeated for the Laplace and Cauchy distribution (using the same distribution for z_i). The results are also give in Table (6.1) .

From Table (6.1) , there is an excellent agreement between the variance and trained β_0 values for the Gaussian case. All values agree to within 5%. The agreement is good to within 20% for the Laplace case. The Cauchy has some moderate disagreement: up to about 30%.

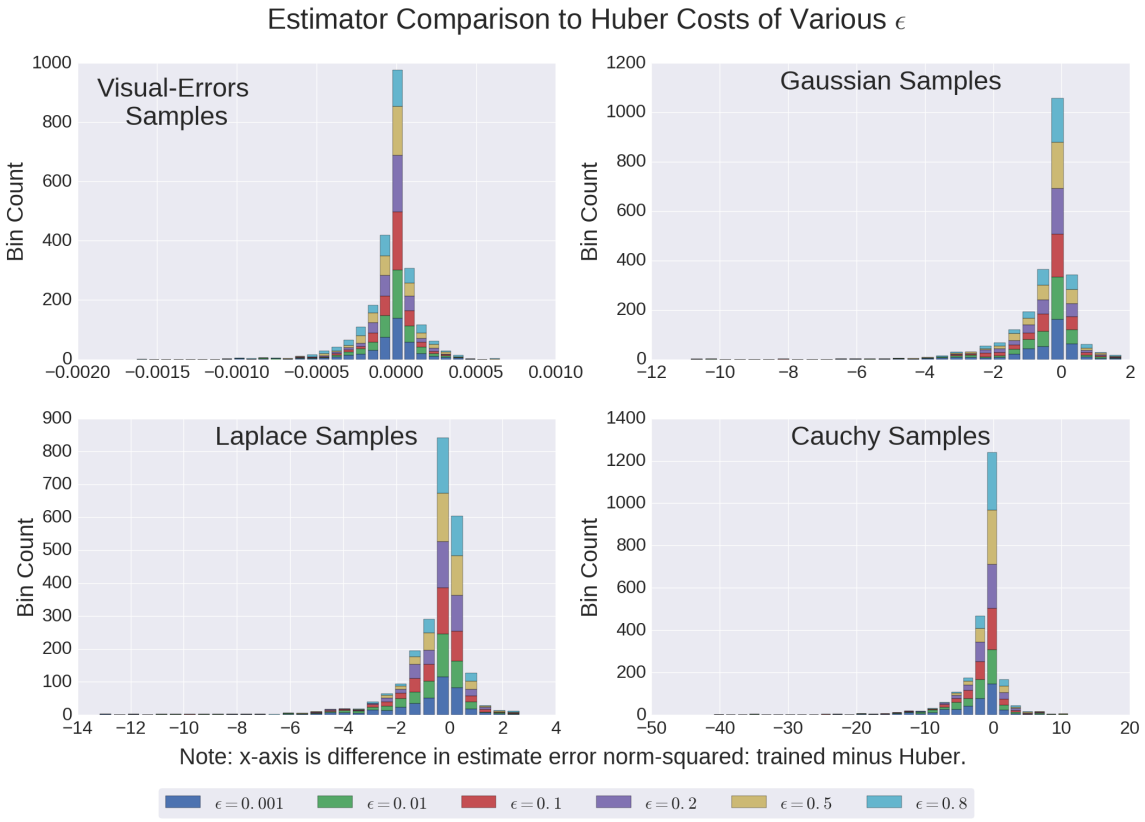
To further test the trained parameters for the three distributions above, a second simple test was performed. A total of $N = 10^6$ samples were generated on each of 100 trials from each of the three distributions. The parameters trained from the two methods were used to compute the empirical asymptotic variance on the samples for each trial. This test helps validate whether the higher-dimensional optimization

problem overfit the data. The inter-quartile range for the Gaussian samples was $[10.77, 10.92]$ and $[10.77, 10.92]$ for the first and second method respectively (where the first method uses the variance for β_0. The inter-quartile range for the Laplace samples was $[10.98, 11.31]$ and $[10.92, 11.26]$ for the first and second method respectively. And finally, the inter-quartile range for the Cauchy samples was $[21.02, 21.59]$ and $[20.48, 21.10]$ for the first and second method respectively. Again, the Gaussian samples give nearly equivalent results for the two methods, the Laplace has very close agreement, less than a 1% efficiency loss, while the Cauchy samples show approximately a 2% to 3% efficiency loss.

These results show that the method which avoids the high-dimensional optimization is sub-optimal but only slightly. Depending on the application of interest, the extra computation may justify the increased accuracy. Furthermore, it is possible that the increase in accuracy is greater than a few percent in some problems.

The next set of numerical examples evaluates the estimator performance on the same problem used in Section 6.1.1: estimating the parameters of a line given $m = 40$ measurements using 400 trials. The same line parameters and alternative estimators used in that section are used here for comparison to the trained estimators. In these examples, the Gaussian, Laplace, and Cauchy errors are considered with the trained-parameters determined above are used. In addition, an estimator trained on the empirical visual error distribution (also used in Section 6.1.1) are used in this example as well. The SSD centroid is, discussed in Section 6.2, is used as the

Figure 6.10: Difference in estimate error squared between the trained estimators and Huber estimators (various $\boldsymbol{\varepsilon}$) for four distributions.



auxiliary measurement. The results are given in Figure (6.10).

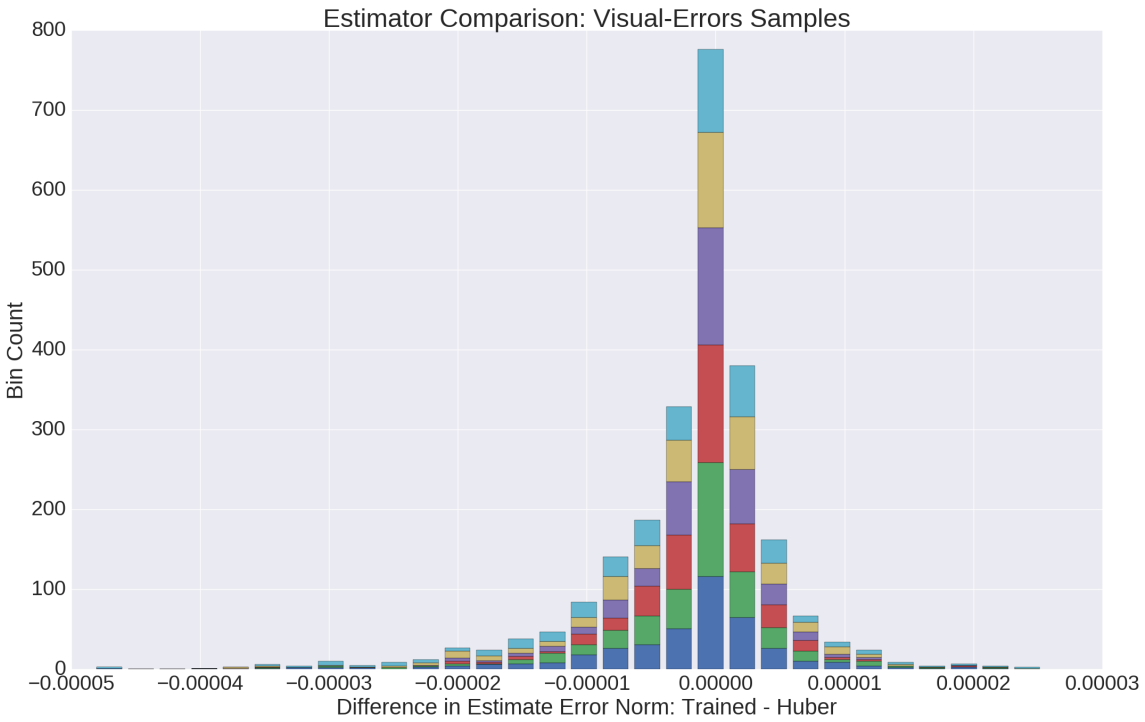
Figure (6.10) gives the difference in estimate error squared between the trained and Huber estimators. A negative difference implies that the trained-estimator had lower error and hence better performance than the corresponding Huber estimator. For the Gaussian, Laplace, and Cauchy samples, the advantage is clear. For these three distributions, a WSRT applied to the trained estimator and each Huber estimator (i.e. given $\boldsymbol{\varepsilon}$) gave p-values less than 0.0001 in each case! On the other hand, for the visual-error samples, the results are not as convincing. The WSRT p-values

were $6.74e{-}4$, 0.00271, 0.0107, 0.00391, $8.22e{-}4$, and $3.56e{-}5$ for ε equal to 0.001, 0.01, 0.1, 0.2, 0.5, and 0.8 respectively. While this is suggestive of a performance advantage, it is by no means definitive evidence.

For the visual-error samples, the test was repeated but with $m = 1000$ measurements on each of the 400 trials. In this case the p-values comparing the trained estimator to each ε-contaminated Huber cost were $3.05e{-}7$, $1.31e{-}6$, $6.61e{-}6$, $9.13e{-}5$, $1.25e{-}6$, and $1.28e{-}9$. This is much stronger evidence of the advantage of the trained estimator. The corresponding histogram is shown in Figure (6.11) The reason for the improved performance at larger m may be due to the fact the the asymptotic properties of the estimator, which is what has been trained for, did not dominate the performance at a lower value of m. This raises an interesting question with broader implications discussed in the next paragraphs. In addition, this may not have been an issue for the other distributions because of the stronger dependence of error v_i on the auxiliary measurement z_i.

The reduced performance of the estimator trained for minimum asymptotic variance on visual errors on the $m = 40$ case as compared to the $m = 1000$ case suggests that estimators may need to be trained specifically for small sample results (as opposed to asymptotic results).

Figure 6.11: Difference in estimate error squared between the trained estimators and Huber estimators (various $\boldsymbol{\varepsilon}$) for four distributions.



6.4 An Approach for Small Samples

The above numerical examples demonstrated the potential of the approach developed in Section 6.1. In particular, it was shown that estimators for general regression problems can be designed by optimizing parameters of a generic cost function for minimum asymptotic variance at the unknown distribution of interest. However, there was evidence that in some cases, the resulting estimator did not give superior performance when the number of measurements was small. This is unsurprising as the estimator was designed for minimum *asymptotic* variance. A heuristic approach was developed to deal with this problem.

For the first method, training samples from the unknown distribution were used to empirically evaluate the estimator variance for the location problem. A logical extension for applications with small samples is to use the training samples to compute an empirical variance. Let the typical number of measurements in the problem of interest be m and assume there are N training samples $\{\boldsymbol{\nu}_i, z_i\}$. Define $\lfloor N/m \rfloor$ batches of m training samples: the i'th set is simply the i'th batch of size m from $\{\boldsymbol{\nu}_i, z_i\}$. Now define the empirical estimator variance at the location problem as

$$v(\boldsymbol{\beta}) = \frac{1}{\lfloor N/m \rfloor} \sum_{i=1}^{\lfloor N/m \rfloor} \left[\operatorname{argmin} \sum_{j_i} \rho(\boldsymbol{\nu}_{j_i}, z_j | \boldsymbol{\beta}) \right]^2 \tag{6.23}$$

where the index j_i sums over the m samples in the i'th batch. The empirical variance in Equation (6.23) can be viewed as a function of the estimator parameters conditioned on the training samples. Therefore it can be optimized for the problem of interest. Note that this can be done with or without auxiliary measurements.

The downside to the above method is a higher computational cost in training due to the large number of required location problem solutions that must be found for each guess of $\boldsymbol{\beta}$. This is especially exaggerated for the MCLE framework which inevitably has a high-dimensional $\boldsymbol{\beta}$. In addition to the approach taken in Section 6.3, one way to reduce the dimensionality of $\boldsymbol{\beta}$ is to use a cost function with fewer parameters. One possibility is the so-called pseudo-Huber cost function given by Hartley and Zisserman [3]. For the purposes of this dissertation, the pseudo-Huber cost has been modified slightly to the form

$$\rho(x) \equiv 2a^2 \left(\sqrt{1 + x^2/b^2} - 1 \right) \tag{6.24}$$

The version given by Hartley and Zisserman has $a = b$ which approximates the Huber cost with cutoff parameter equal to one and scale parameter equal to b. Here an additional parameter is added so that the size of the inlier region can be set independent of the scale. This gives a two-parameter cost instead of a four-parameter cost (per auxiliary measurement bin in the MCLE case). Despite the added training cost, this is a more flexible technique as the number of measurements m can be adjusted for the problem of interest. In addition, there is no added cost to the resulting estimator.

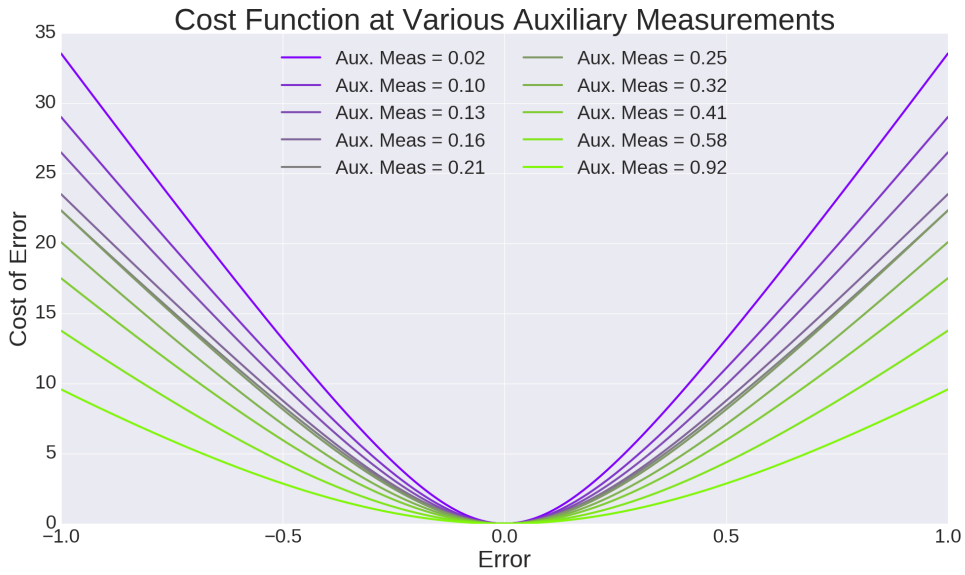
A numerical example of this is method is given below.

6.4.1 A Numerical Example

To demonstrate the small sample method of the previous section, a numerical example is performed. The paired samples $\{v_i, z_i\}$ from the empirical visual error distribution used in the last example of Section 6.3.1 (Figure (6.10) and Figure (6.11)) are used to train an estimator for minimum variance at $m = 20$ measurements. The two-parameter pseudo-Huber cost function defined in Equation (6.24) is used with 30 tie-points for the auxiliary measurements.

The resulting sixty-dimensional cost function defined in Equation (6.23) is optimized with the *SciPy* implementation of Powell's method [22, 107]. Powell's method is a derivative-free optimization method that attempts to find and proceed along conjugate directions with respect to a quadratic approximation to the cost function. This method was chosen because it does not requires derivatives and because

Figure 6.12: MCLE cost function resulting from small-sample training with thirty bins for the auxiliary measurement (only ten bins shown for clarity).

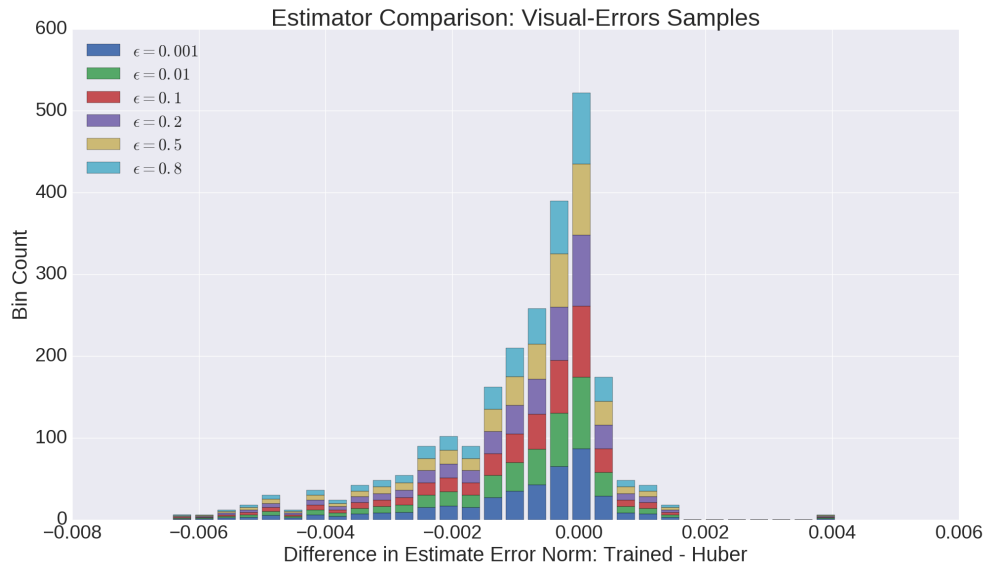


it was found to obtain an acceptable solution with a fewer number of cost function evaluations (which are more expensive than the asymptotic variance evaluation).

The resulting cost function is shown in Figure (6.12). Note that only ten of the thirty auxiliary measurement bins are shown. The trend is as expected. As the auxiliary measurement increases, the scale of the error increases (i.e. see Figure (6.8)). This is captured in training: as the auxiliary measurement increases, the cost is scaled down.

The trained MCLE cost is used on the same regression problem discussed in Section 6.3.1. The results on each trail are compared to the same set of Huber estimators used in Section 6.3.1 (i.e. of varying ε). The resulting histogram of the difference in error between the trained and Huber estimators is shown in Figure

Figure 6.13: Difference in estimate error squared between the small sample trained estimator and Huber estimators (various $\boldsymbol{\varepsilon}$).



(6.13). A WSRT was used to compare the results of the trained estimator to each Huber estimator. In all cases, the p-value was essentially-zero (10^{-36}). This shows a clear advantage of the small sample MCLE over each instance of the Huber estimator.

6.5 Application to Vision-Aided Navigation

This section presents the results of applying the methods developed above to the vision-aided navigation problem. The test cases used in this section involve a trajectory class which was not used in Section 5. Each dataset of the trajectory class contains 40 images, each equally spaced along a circular path with a diameter of one distance unit and an altitude of 10 distance units above the surface. The additional images (40 as compared to 20) and the motion give good observability of

the landmark parameters. The strong observability of this dataset encourages the dominance of the asymptotic error terms.

In the first test, a learning dataset was used to train a single Huber-Rolloff cost function. A total of 50 datasets of 40 images, each containing 200 landmarks was used to generate the samples. In total, 250,000 samples were used to train the estimator. The resulting parameters were $\alpha_1 = 0.0743$, $\alpha_2 = 0.153$, $\alpha_3 = 0.718$. Table (6.2) shows the empirical variance evaluated at both the trained estimator and several instances of Huber and Cauchy estimators. From the table, it is clear that the trained estimator has the smallest asymptotic empirical variance. Two interesting notes can be made. First, the Cauchy estimator has a theoretical asymptotic variance that is much more sensitive to scale than the Huber estimator. Second, the trained estimator has only a slight advantage as compared to the *best* Cauchy and Huber costs. As compared to least-squares, *any* of the robust cost functions provide an enormous advantage. Nevertheless, the training method gives an objective means to choose the best cost function.

Figure (6.14) shows the results for the first test case in two violin plots. The top plot contains RMS position error and the bottom plot contains RMS attitude error for each estimator as indicated on the horizontal axis. The vertical axis shows: (1) the RMS error for each dataset (horizontal lines), (2) a kernel density estimate for the RMS errors (solid contour line), (3) the inter-quartile range (solid vertical bar) and (4) the median (white dot). There are several important conclusions that can

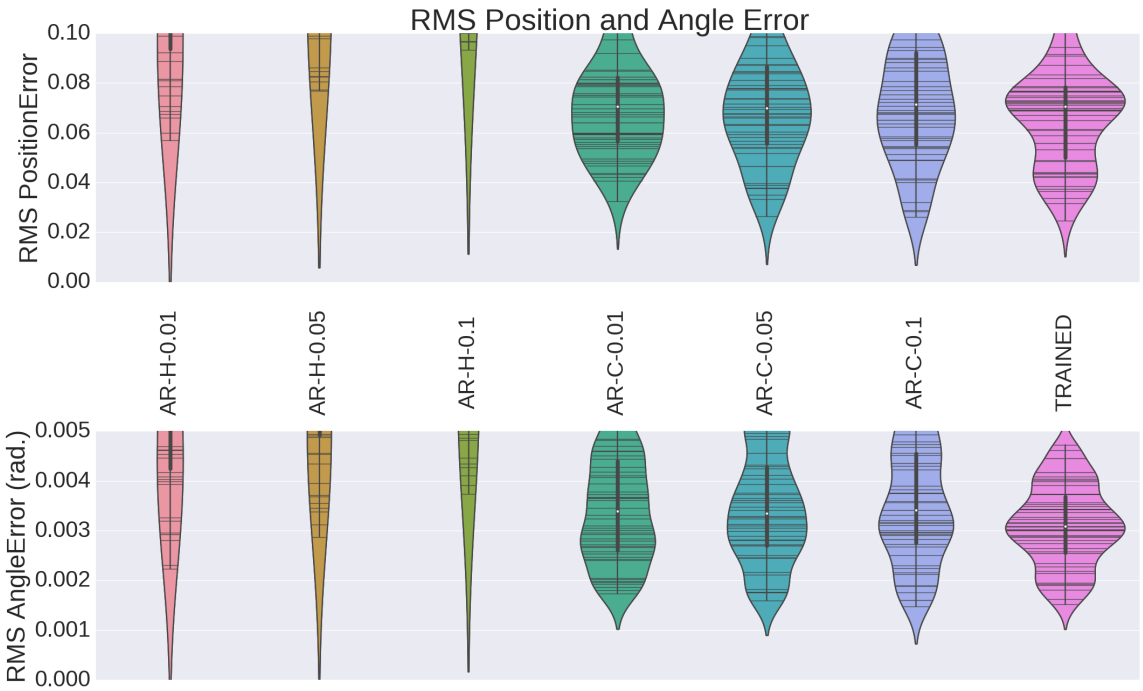
Table 6.2: Empirical variance of several estimators for a training sample.

Estimator	$N \sum_{i=1}^{N} \psi(v_i)^2 / \left(\sum_{i=1}^{N} \psi'(v_i) \right)^2$
Huber-$\sigma = 0.01$	0.0255
Huber-$\sigma = 0.05$	0.0237
Huber-$\sigma = 0.1$	0.0250
Huber-$\sigma = 0.2$	0.0339
Cauchy-$\sigma = 0.01$	0.0791
Cauchy-$\sigma = 0.05$	0.0250
Cauchy-$\sigma = 0.1$	0.0203
Cauchy-$\sigma = 0.2$	0.0213
Least Squares	1.982
Trained	0.0202

be drawn from this figure. The most obvious observation is the failure of the Huber estimators which have very large error. The Cauchy estimators all have similar error, regardless of scale. The trained estimator has slightly better performance than all of the Cauchy estimators.

The advantage of the trained estimator over the other estimators is shown to be statistically significant in Table (6.3) . This table contains WSRT p-values for the comparison of the trained estimator to the others. Even though the advantage is small over the Cauchy estimators, 5-10%, the reduction is consistent across the datasets which is why the WSRT p-value is nearly zero. Despite this important and convincing result, there are some unexplained trends. The results in Table (6.3) and Figure (6.14) are inconsistent with the theoretical variances in Table (6.2) . The theoretical variance of the Huber estimators is only 25% greater than the trained estimator but the resulting RMS errors are nearly double which is a 400%

Figure 6.14: Violin plot of RMS position error (top) and attitude error (bottom) for several autoregressive estimators including the trained Huber-Rolloff.



increase in error squared (which should be proportional to variance). Furthermore, the $\sigma = 0.01$ Cauchy estimator has similar errors to the other Cauchy estimators and to the trained estimator despite having a variance in Table (6.2) that is three to four times higher. These paradoxes are troubling and are addressed in the next test case.

The discrepancy between the theoretical scalar variances and the the RMS error of the estimators can be due to a number of reasons. One issue could be that the samples used to train the estimator are not exactly drawn for the true distribution. This is of course true. The pseudo-error samples are related to the true errors via Equation (5.69) which required a linear approximation. A second

Table 6.3: Table of WSRT p-values and percent reduction in errors for the trained estimator compared to Cauchy and Huber estimators of various scale.

	Angle		**Position**	
Est.	p	$\frac{e_{ST}-e_{AR}}{e_{ST}} \times 100\%$	p	$\frac{e_{ST}-e_{AR}}{e_{ST}} \times 100\%$
AR-0.01-H	0.000	50.581 %	0.000	55.794 %
AR-0.05-H	0.000	58.172 %	0.000	62.651 %
AR-0.1-H	0.000	64.179 %	0.000	69.612 %
AR-0.01-C	0.000	11.235 %	0.008	10.808 %
AR-0.05-C	0.000	8.886 %	0.000	7.448 %
AR-0.1-C	0.000	9.830 %	0.000	9.256 %

possible explanation is an unmodeled correlation structure. The autoregressive model assumes positive serial correlations for each individual landmark. It is possible that inter-landmark correlations exist for the case considered. For example, this could be due to a large number of landmarks being tracked on the limb of the comet model. The second test case addresses both of these explanations simultaneously.

The second test case was set up as follows. The evaluation dataset was used to obtain pseudo-error error samples in the same manner as the training dataset used to train the estimators. As both datasets were generated in the same manner, both sets of samples come from the same underlying distribution. However, the two problems are that this is not the exact distribution and there may be hidden correlation structures in the data. To test the estimators with these problems removed, a *mixed simulation* approach similar to that used in Section 5.11.7 was used here. In particular, 50 datasets were derived from the evaluation dataset using the same trajectories, landmark positions, and occlusion conditions. The difference is that the errors were simulated with a perfect $\alpha = 1$ autoregressive error process. The errors driving the

process were selected uniformly randomly from the evaluation datasets pseudo-error samples. By selecting the samples in a random manner, any correlation structure that existed in the true measurements is destroyed. In addition, the resulting dataset is described by a measurement model that is consistent with the measurements used to train the estimator. Finally, because the samples were obtained from the evaluation set, they are independent of the samples used to train the estimator which avoids any over-fitting issues.

The results for the second test case are shown in Figure (6.15) which is analogous to Figure (6.14). In addition, Table (6.4) gives the WSRT p-values and percent reduction in errors for the trained estimator compared to the other estimators. These results are consistent with the theoretical variances in Table (6.2) ! In particular, the performance of the Huber estimator is consistent regardless of scale. Compared to the train estimator, the errors in the Huber estimator are 10-20% larger which is consistent with the theoretical variance of the Huber estimator (which is 25% larger than that of the trained estimator). In addition, the Cauchy estimators are more scale sensitive with the estimator error decreasing as the scale increases from $\sigma = 0.01$ to $\sigma = 0.1$. The $\sigma = 0.1$ scale makes the Cauchy estimator most competitive with the trained estimator. Nevertheless, the trained estimator has a statistically significant improvement as the WSRT p-value < 0.0001 suggests.

The fact that the theoretical and numerical results are consistent in the mixed-simulation test but **not** on the true measurements is a key observation with several

Figure 6.15: Mixed-simulation test. Violin plot of RMS position error (top) and attitude error (bottom) for several autoregressive estimators including the trained Huber-Rolloff.

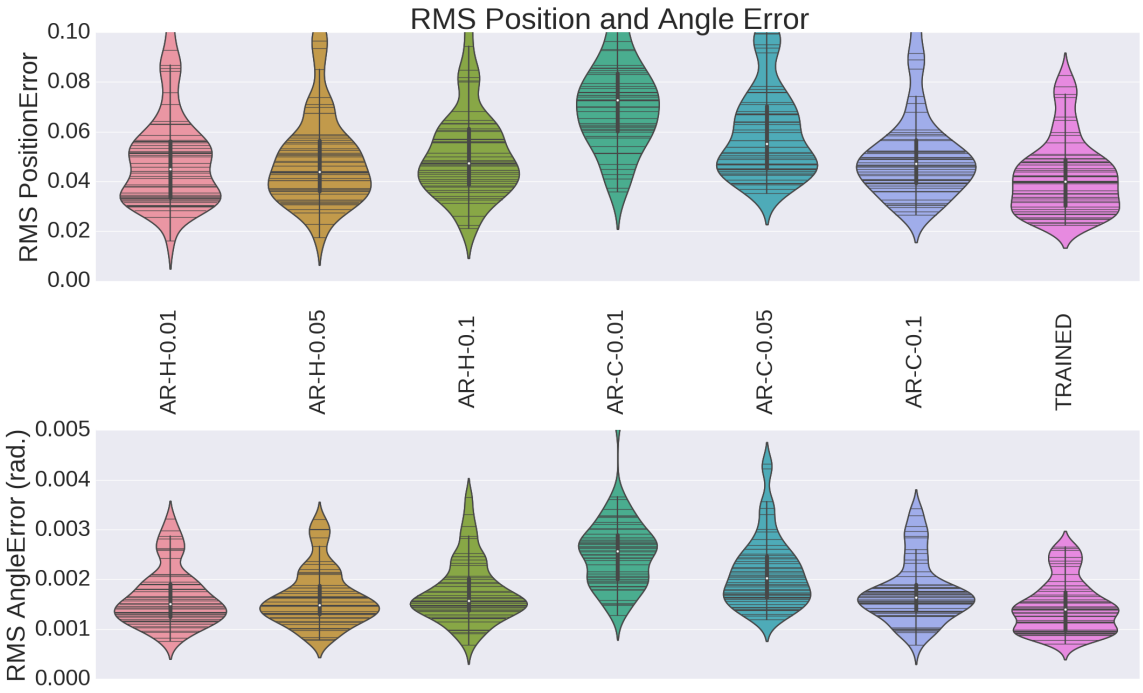


Table 6.4: Mixed-simulation test. Table of WSRT p-values and percent reduction in errors for the trained estimator compared to Cauchy and Huber estimators of various scale.

	Angle		**Position**	
Est.	p	$\frac{e_{ST}-e_{AR}}{e_{ST}} \times 100\%$	p	$\frac{e_{ST}-e_{AR}}{e_{ST}} \times 100\%$
AR-0.01-H	0.003	12.075 %	0.001	12.694 %
AR-0.05-H	0.006	13.601 %	0.002	11.683 %
AR-0.1-H	0.001	20.097 %	0.000	17.180 %
AR-0.01-C	0.000	44.040 %	0.000	42.809 %
AR-0.05-C	0.000	30.900 %	0.000	29.325 %
AR-0.1-C	0.000	17.503 %	0.000	17.731 %

implications. Most importantly, it validates the data-driven estimator design for general nonlinear regression problems when samples from the true distribution can

be obtained and all correlations are modeled. In addition, it suggests several concerning issues about vision-aided navigation problems. Despite the demonstrated improvements to visual measurement models presented in this dissertation, it is possible that further improvements are possible. The strongest evidence for this is the large error in the Huber estimator in the first test case. One possible reason for this is an unmodeled correlation structure. Such correlations can cause a *coalition of outliers* that can corrupt the estimate. In the test case presented above, such a coalition can be created by a subset of landmarks which sit on the limb of the comet model and have a bulk motion that appears inconsistent with the true comet-camera relative motion. The convex nature of the Huber estimator can cause the estimate to be corrupted even when the initial guess is simultaneously close to the truth and far from any false-solution suggested by the coalition. The issue is a strong motivation for further research which is discussed in the conclusion of this section.

6.6 Summary

This section presented a data-driven approach to M-estimation which was motivated by a theoretical result derived in Section 2. The advantage of the resulting M-estimators over the state-of-the-art Huber estimator was demonstrated on a set of canonical problems. An extension of this concept to leverage so-called auxiliary measurements was presented and termed the MCLE. Evidence of such auxiliary measurements in vision-aided navigation was presented and the advantage of the

algorithm was demonstrated on canonical problems.

The data-driven approach was applied to the vision-aided navigation problem and a statistically significant improvement was demonstrated. However, several surprising paradoxes were observed. Inconsistency between the true measurement model and the assumed measurement model used to train the estimator was identified as a likely explanation for the paradoxes. A mixed-simulation analysis that removed this inconsistency was performed. The paradoxes were **not** present in the results which was both an important success for the data-driven approach and convincing evidence that the explanation was correct. This gives strong motivation for two paths for further research. One possible path is to further improve the measurement models to capture inter-landmark correlations. Unfortunately, this is likely to be highly problem dependent which would limit the applicability of the results.

A second, and possibly far more lucrative, path is to investigate the use of high breakdown point estimators for the vision-aided navigation problem. One common estimator in the robust statistics literature is known as least trimmed squares (LTS). Given N measurements and a tuning parameter k, this algorithms computes an estimate that minimizes the k smallest square residuals [108]. For $k = N/2$, the breakdown point is 50%. This is an excellent candidate for problems that contain outlier coalitions. The downside is that computation of the estimate is more expensive, especially for problems with a large number of parameters and measurements. Approximations to the true solution lead to more computationally tractable algo-

rithms [108]. Tailoring such estimators to the vision-aided navigation problem and modifying them to handle alternative *trimmed costs* is a promising area for further research.

7 CONCLUSION

The primary goal of this dissertation was to develop methods to improve the accuracy and robustness of vision-aided navigation systems. Visual feature measurements are the distinguishing information source of such systems and MLE-like estimators are the foundation for estimation algorithms that use them. These estimators require an accurate model of visual feature measurements. This dissertation presented analysis to systematically assess the validity of the ubiquitous assumptions used in such models. Evidence against the independence assumption, the identically distributed assumption, and the Gaussian assumption, was presented. This evidence, by itself, is an important contribution of this dissertation. In addition, the evidence and associated analysis that contradicted the standard assumptions motivated the other contributions of this dissertation.

In summary, the primary contributions of this dissertation are

1. Evidence of strong positive serial error correlations in visual feature measurements (see Section 5.2.1).
2. A method to approximate the distribution of visual measurement errors under a number of structural models using a high-fidelity rendering tool (see Section 5.10.
3. Evidence to suggest that visual measurement errors are better approximated by a Cauchy distribution than a normal distribution (see Section 5.10.

4. Evidence to suggest large reductions in error when using a Cauchy distribution to derive MLEs as compared to the state-of-the-art Huber cost function (see Section 5.11.2 and Section 5.11.5).
5. The development and implementation of a set of structural models for visual feature measurements that captures serial error correlations (see Section 5.3, Section 5.7, Section 5.4, and Section 5.6).
6. Demonstration of significant performance advantages for estimators derived under models that capture serial error correlations in visual feature measurements. In tested cases, for certain motion types, a reduction in RMS angle and position error was typically 60% and 30% respectively (see Section 5.11.2 and Section 5.11.5).
7. The development, implementation and demonstration (on canonical problems) of a data-driven method to optimize a generic M-estimator for asymptotic variance using samples of an unknown distribution (see Section 6.1).
8. Evidence to suggest that visual measurement errors are not identically distributed and the identification of observable quantities that provide information about the distribution of such errors (see Section 6.2).
9. The development, implementation and demonstration (on canonical problems) of a data-driven MCLE method to optimize a generic M-estimator for asymptotic variance using samples and auxiliary measurements from an unknown joint distribution (see Section 6.3).

10. Demonstration of the data-driven approach to the vision-aided navigation problem in a controlled test case that removed possibly unknown correlation structures and insured consistency between the true and assumed measurement models.

The impact of these contributions are broad. The MLE-like estimators have been recognized as the estimator-of-choice for vision-aided navigation systems from an accuracy and robustness perspective and have been historically used in the bundle adjustment community. Early research on visual measurements for real-time use in robotics focused on sequential filtering applications due to the computational complexity of the MLE and M-estimators. As the price, size, and power consumption of computational platforms has shrunk, these estimators have become increasingly practical for real-time feedback control and motion planning. In fact, much of the recent SLAM research has focused on applying the MLE in real-time systems. Such estimators can easily incorporate the proposed models, both structural and probabilistic, with little change to the system architecture. Furthermore, the choice of additional sensors and source of information is made irrelevant by the MLE and M-estimation framework. This further broadens the applicability of the results of this dissertation. Based on these facts and the evidence contained in this dissertation, the use of the developed models is recommended for future vision-aided navigation systems.

An addition to the development and demonstration of improved measurement

models, this dissertation also presented a data-driven approach to minimum asymptotic variance M-estimator design. This method had a clear advantage in canonical problems. The flexibility of the method suggests that it can be applied to a broad range of applications. When applied to the vision-aided navigation problem, a statistically significant improvement was demonstrated. However, the results contained a number of troubling paradoxes and evidence was found to suggest that modeling errors were the cause of the paradox. In particular, in certain problems, coalitions of outliers may act together to suggest an incorrect solution. This is an entirely different issue than outliers which act independently with no shared consensus. Such problems may be addressed with a class of least-trimmed estimators which minimize a cost function on the best subset of measurements. Applying and modifying these estimators to the vision-aided navigation problem is expected to be a productive path for future research.

In summary, this dissertation has presented a set of improved measurement models for visual feature measurements and a data-driven approach to M-estimator design. The improved measurement models can be used in MLE-like estimators which are increasingly being used as the estimation framework-of-choice for vision-aided navigation systems. In addition, the data-driven M-estimator design methods resolve common issues in designing M-estimator cost functions and are applicable to a wide-range of applications. Taken together, both the improved measurement models and data-driven methods can accomplish the goal of improving the accuracy

and robustness of vision-aided navigation systems.

REFERENCES

[1] E. H. Adelson and J. R. Bergen, *The plenoptic function and the elements of early vision*. Vision and Modeling Group, Media Laboratory, Massachusetts Institute of Technology, 1991.

[2] B. Triggs, P. F. McLauchlan, R. I. Hartley, and A. W. Fitzgibbon, "Bundle adjustment: a modern synthesis," in *Vision algorithms: theory and practice*, pp. 298–372, Springer, 2000.

[3] R. Hartley and A. Zisserman, *Multiple view geometry in computer vision*. Cambridge university press, 2003.

[4] A. J. Davison, I. D. Reid, N. D. Molton, and O. Stasse, "Monoslam: Real-time single camera slam," *IEEE transactions on pattern analysis and machine intelligence*, vol. 29, no. 6, pp. 1052–1067, 2007.

[5] S. M. Kay, *Fundamentals of Statistical Signal Processing: Estimation Theory*. Upper Saddle River, NJ, USA: Prentice-Hall, Inc., 1993.

[6] R. Smith, M. Self, and P. Cheeseman, "Estimating uncertain spatial relationships in robotics," in *Autonomous robot vehicles*, pp. 167–193, Springer, 1990.

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

[8] J. Civera, A. J. Davison, and J. M. Montiel, "Inverse depth parametrization for monocular slam," *IEEE transactions on robotics*, vol. 24, no. 5, pp. 932–945, 2008.

[9] M. Montemerlo, S. Thrun, D. Koller, B. Wegbreit, *et al.*, "Fastslam: A factored solution to the simultaneous localization and mapping problem," in *Aaai/iaai*, pp. 593–598, 2002.

[10] M. Montemerlo and S. Thrun, "Fastslam 2.0," *FastSLAM: A scalable method for the simultaneous localization and mapping problem in robotics*, pp. 63–90, 2007.

[11] M. A. Fischler and R. C. Bolles, "Random sample consensus: a paradigm for model fitting with applications to image analysis and automated cartography," *Communications of the ACM*, vol. 24, no. 6, pp. 381–395, 1981.

[12] D. Nistér, "Preemptive ransac for live structure and motion estimation," *Machine Vision and Applications*, vol. 16, no. 5, pp. 321–329, 2005.

[13] A. Vedaldi, H. Jin, P. Favaro, and S. Soatto, "Kalmansac: Robust filtering by consensus," in *Tenth IEEE International Conference on Computer Vision (ICCV'05) Volume 1*, vol. 1, pp. 633–640, IEEE, 2005.

[14] H. Strasdat, J. Montiel, and A. J. Davison, "Real-time monocular slam: Why filter?," in *Robotics and Automation (ICRA), 2010 IEEE International Conference on*, pp. 2657–2664, IEEE, 2010.

[15] H. Strasdat, J. M. Montiel, and A. J. Davison, "Visual slam: why filter?," *Image and Vision Computing*, vol. 30, no. 2, pp. 65–77, 2012.

[16] G. Klein and D. Murray, "Parallel tracking and mapping for small ar workspaces," in *Mixed and Augmented Reality, 2007. ISMAR 2007. 6th IEEE and ACM International Symposium on*, pp. 225–234, IEEE, 2007.

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

[18] J. L. Junkins and P. Singla, "How nonlinear is it?," *Paper AAS*, pp. 03–286, 2003.

[19] J. Neyman and E. L. Scott, "Consistent estimates based on partially consistent observations," *Econometrica: Journal of the Econometric Society*, pp. 1–32, 1948.

[20] S. M. Kay, "Fundamentals of statistical signal processing, volume i: estimation theory," vol. 1, 1993.

[21] K. S. Miller, "On the inverse of the sum of matrices," *Mathematics Magazine*, vol. 54, no. 2, pp. 67–72, 1981.

[22] E. Jones, T. Oliphant, P. Peterson, *et al.*, "SciPy: Open source scientific tools for Python," 2001–. [Online; accessed 2016-05-08].

[23] R. P. Brent, "An algorithm with guaranteed convergence for finding a zero of a function," *The Computer Journal*, vol. 14, no. 4, pp. 422–425, 1971.

[24] J. W. Tukey, "A survey of sampling from contaminated distributions," *Contributions to probability and statistics*, vol. 2, pp. 448–485, 1960.

[25] P. J. Huber *et al.*, "Robust estimation of a location parameter," *The Annals of Mathematical Statistics*, vol. 35, no. 1, pp. 73–101, 1964.

[26] F. R. Hampel, "The influence curve and its role in robust estimation," *Journal of the American Statistical Association*, vol. 69, no. 346, pp. 383–393, 1974.

[27] F. R. Hampel, E. M. Ronchetti, P. J. Rousseeuw, and W. A. Stahel, *Robust statistics: the approach based on influence functions*, vol. 114. John Wiley & Sons, 2011.

[28] J. O. Berger, E. Moreno, L. R. Pericchi, M. J. Bayarri, J. M. Bernardo, J. A. Cano, J. De la Horra, J. Martín, D. Ríos-Insúa, B. Betrò, A. Dasgupta, P. Gustafson, L. Wasserman, J. B. Kadane, C. Srinivasan, M. Lavine, A. O'Hagan, W. Polasek, C. P. Robert, C. Goutis, F. Ruggeri, G. Salinetti, and S. Sivaganesan, "An overview of robust bayesian analysis," *Test*, vol. 3, no. 1, pp. 5–124, 1994.

[29] P. J. Huber, *Robust statistics*. Springer, 2011.

[30] D. D. Boos and R. Serfling, "A note on differentials and the clt and lil for statistical functions, with application to m-estimates," *The Annals of Statistics*, pp. 618–624, 1980.

[31] P. Kevei, "A note on asymptotics of linear combinations of iid random variables," *Periodica Mathematica Hungarica*, vol. 60, no. 1, pp. 25–36, 2010.

[32] D. Andrews, P. Bickel, and F. Hampel, "Robust estimates of location: survey and advances," 1972.

[33] P. J. Rousseeuw and C. Croux, "Alternatives to the median absolute deviation," *Journal of the American Statistical association*, vol. 88, no. 424, pp. 1273–1283, 1993.

[34] J. R. Collins and B. Wu, "Comparisons of asymptotic biases and variances of m-estimators of scale under asymmetric contamination," *Communications in Statistics-Theory and Methods*, vol. 27, no. 7, pp. 1791–1810, 1998.

[35] R. Maronna, D. Martin, and V. Yohai, *Robust statistics*. John Wiley & Sons, Chichester. ISBN, 2006.

[36] P. J. Huber, "The 1972 wald lecture robust statistics: A review," *The Annals of Mathematical Statistics*, pp. 1041–1067, 1972.

[37] R. Andersen, *Modern methods for robust regression*. No. 152, Sage, 2008.

[38] W. S. Krasker and R. E. Welsch, "Efficient bounded-influence regression estimation," *Journal of the American statistical Association*, vol. 77, no. 379, pp. 595–604, 1982.

[39] C. Mallows, "Influence functions," in *talk at NBER Conference on Robust Regression*, 1973.

[40] L. Wasserman, *All of nonparametric statistics*. Springer Science & Business Media, 2006.

[41] E. Handschin, F. C. Schweppe, J. Kohlas, and A. Fiechter, "Bad data analysis for power system state estimation," *Power Apparatus and Systems, IEEE Transactions on*, vol. 94, no. 2, pp. 329–337, 1975.

[42] C. D. Karlgaard and H. Schaub, "Adaptive huber based filtering using projection statistics: Application to spacecraft attitude estimation," in *AIAA Guidance, Navigation, and Control Conference*, 2008.

[43] H. M. Merrill and F. C. Schweppe, "Bad data suppression in power system static state estimation," *IEEE Transactions on Power Apparatus and Systems*, no. 6, pp. 2718–2725, 1971.

[44] D. W. Marquardt, "An algorithm for least-squares estimation of nonlinear parameters," *Journal of the society for Industrial and Applied Mathematics*, vol. 11, no. 2, pp. 431–441, 1963.

[45] M. J. Powell, "A hybrid method for nonlinear equations," *Numerical methods for nonlinear algebraic equations*, vol. 7, pp. 87–114, 1970.

[46] M. I. Lourakis and A. A. Argyros, "Is levenberg-marquardt the most efficient optimization algorithm for implementing bundle adjustment?," in *Computer Vision, 2005. ICCV 2005. Tenth IEEE International Conference on*, vol. 2, pp. 1526–1531, IEEE, 2005.

[47] P. W. Holland and R. E. Welsch, "Robust regression using iteratively reweighted least-squares," *Communications in Statistics-theory and Methods*, vol. 6, no. 9, pp. 813–827, 1977.

[48] R. Kümmerle, G. Grisetti, H. Strasdat, K. Konolige, and W. Burgard, "g 2 o: A general framework for graph optimization," in *Robotics and Automation (ICRA), 2011 IEEE International Conference on*, pp. 3607–3613, IEEE, 2011.

[49] P. J. Huber, "The behavior of maximum likelihood estimates under nonstandard conditions," in *Proceedings of the fifth Berkeley symposium on mathematical statistics and probability*, vol. 1, pp. 221–233, 1967.

[50] D. A. Freedman, "On the so-called huber sandwich estimator and robust standard errors," *The American Statistician*, 2012.

[51] M. H. Quenouille, "Approximate tests of correlation in time-series 3," in *Mathematical Proceedings of the Cambridge Philosophical Society*, vol. 45, pp. 483–484, Cambridge Univ Press, 1949.

[52] J. Durbin and G. S. Watson, "Testing for serial correlation in least squares regression: I," *Biometrika*, vol. 37, no. 3/4, pp. 409–428, 1950.

[53] T. J. Fisher and C. M. Gallagher, "New weighted portmanteau statistics for time series goodness of fit testing," *Journal of the American Statistical Association*, vol. 107, no. 498, pp. 777–787, 2012.

[54] G. M. Ljung and G. E. Box, "On a measure of lack of fit in time series models," *Biometrika*, vol. 65, no. 2, pp. 297–303, 1978.

[55] V. A. Epanechnikov, "Non-parametric estimation of a multivariate probability density," *Theory of Probability & Its Applications*, vol. 14, no. 1, pp. 153–158, 1969.

[56] D. W. Scott and G. R. Terrell, "Biased and unbiased cross-validation in density estimation," *Journal of the American Statistical Association*, vol. 82, no. 400, pp. 1131–1146, 1987.

[57] N. Smirnov, "Table for estimating the goodness of fit of empirical distributions," *The annals of mathematical statistics*, vol. 19, no. 2, pp. 279–281, 1948.

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

[59] E. S. Jones and S. Soatto, "Visual-inertial navigation, mapping and localization: A scalable real-time causal approach," *The International Journal of Robotics Research*, vol. 30, no. 4, pp. 407–430, 2011.

[60] S. Leutenegger, P. T. Furgale, V. Rabaud, M. Chli, K. Konolige, and R. Siegwart, "Keyframe-based visual-inertial slam using nonlinear optimization.," in *Robotics: Science and Systems*, 2013.

[61] K. Konolige, M. Agrawal, and J. Sola, "Large-scale visual odometry for rough terrain," in *Robotics Research*, pp. 201–212, Springer, 2011.

[62] K. Konolige and M. Agrawal, "Frameslam: From bundle adjustment to real-time visual mapping," *IEEE Transactions on Robotics*, vol. 24, no. 5, pp. 1066–1077, 2008.

[63] K. Konolige, J. Bowman, J. Chen, P. Mihelich, M. Calonder, V. Lepetit, and P. Fua, "View-based maps," *The International Journal of Robotics Research*, 2010.

[64] C. Tomasi and T. Kanade, *Detection and tracking of point features*. School of Computer Science, Carnegie Mellon Univ., 1991.

[65] C. Harris and M. Stephens, "A combined corner and edge detector.," in *Alvey vision conference*, vol. 15, p. 50, Citeseer, 1988.

[66] M. Trajković and M. Hedley, "Fast corner detection," *Image and vision computing*, vol. 16, no. 2, pp. 75–87, 1998.

[67] T. Tuytelaars and K. Mikolajczyk, "Local invariant feature detectors: a survey," *Foundations and Trends® in Computer Graphics and Vision*, vol. 3, no. 3, pp. 177–280, 2008.

[68] Y. Guo, M. Bennamoun, F. Sohel, M. Lu, and J. Wan, "3d object recognition in cluttered scenes with local surface features: A survey," *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, vol. 36, no. 11, pp. 2270–2287, 2014.

[69] P. E. Anuta, "Spatial registration of multispectral and multitemporal digital imagery using fast fourier transform techniques," *Geoscience Electronics, IEEE Transactions on*, vol. 8, no. 4, pp. 353–368, 1970.

[70] A. E. Johnson, A. Ansar, L. H. Matthies, N. Trawny, A. I. Mourikis, and S. I. Roumeliotis, "A general approach to terrain relative navigation for planetary landing," in *AIAA Aerospace@ Infotech Conf., Rohnert Park, CA*, 2007.

[71] B. D. Macomber, *A phase based dense stereo algorithm implemented in cuda.* PhD thesis, Texas A&M University, 2011.

[72] J. A. Leese, C. S. Novak, and B. B. Clark, "An automated technique for obtaining cloud motion from geosynchronous satellite data using cross correlation," *Journal of applied meteorology*, vol. 10, no. 1, pp. 118–132, 1971.

[73] D. I. Barnea and H. F. Silverman, "A class of algorithms for fast digital image registration," *Computers, IEEE Transactions on*, vol. 100, no. 2, pp. 179–186, 1972.

[74] B. D. Lucas, T. Kanade, *et al.*, "An iterative image registration technique with an application to stereo vision.," in *IJCAI*, vol. 81, pp. 674–679, 1981.

[75] C.-S. Fuh and P. Maragos, "Motion displacement estimation using an affine model for image matching," *Optical Engineering*, vol. 30, no. 7, pp. 881–887, 1991.

[76] R. Manmatha and J. Oliensis, "Extracting affine deformations from image patches. i. finding scale and rotation," in *Computer Vision and Pattern Recognition, 1993. Proceedings CVPR'93., 1993 IEEE Computer Society Conference on*, pp. 754–755, IEEE, 1993.

[77] J. Weng, P. Cohen, and M. Herniou, "Camera calibration with distortion models and accuracy evaluation," *IEEE Transactions on Pattern Analysis & Machine Intelligence*, no. 10, pp. 965–980, 1992.

[78] R. B. Rusu, N. Blodow, and M. Beetz, "Fast point feature histograms (fpfh) for 3d registration," in *Robotics and Automation, 2009. ICRA'09. IEEE International Conference on*, pp. 3212–3217, IEEE, 2009.

[79] L. Bo, X. Ren, and D. Fox, "Depth kernel descriptors for object recognition," in *2011 IEEE/RSJ International Conference on Intelligent Robots and Systems*, pp. 821–826, IEEE, 2011.

[80] B. Steder, R. B. Rusu, K. Konolige, and W. Burgard, "Point feature extraction on 3d range scans taking into account object boundaries," in *Robotics and automation (icra), 2011 ieee international conference on*, pp. 2601–2608, IEEE, 2011.

[81] A. E. Johnson and M. Hebert, "Using spin images for efficient object recognition in cluttered 3d scenes," *IEEE Transactions on pattern analysis and machine intelligence*, vol. 21, no. 5, pp. 433–449, 1999.

[82] P. J. Besl and N. D. McKay, "Method for registration of 3-d shapes," in *Robotics-DL tentative*, pp. 586–606, International Society for Optics and Photonics, 1992.

[83] S. Rusinkiewicz and M. Levoy, "Efficient variants of the icp algorithm," in *3-D Digital Imaging and Modeling, 2001. Proceedings. Third International Conference on*, pp. 145–152, IEEE, 2001.

[84] D. Chetverikov, D. Svirko, D. Stepanov, and P. Krsek, "The trimmed iterative closest point algorithm," in *Pattern Recognition, 2002. Proceedings. 16th International Conference on*, vol. 3, pp. 545–548, IEEE, 2002.

[85] R. A. Newcombe, S. Izadi, O. Hilliges, D. Molyneaux, D. Kim, A. J. Davison, P. Kohi, J. Shotton, S. Hodges, and A. Fitzgibbon, "Kinectfusion: Real-time dense surface mapping and tracking," in *Mixed and augmented reality (ISMAR), 2011 10th IEEE international symposium on*, pp. 127–136, IEEE, 2011.

[86] S. Izadi, D. Kim, O. Hilliges, D. Molyneaux, R. Newcombe, P. Kohli, J. Shotton, S. Hodges, D. Freeman, A. Davison, *et al.*, "Kinectfusion: real-time 3d reconstruction and interaction using a moving depth camera," in *Proceedings of the 24th annual ACM symposium on User interface software and technology*, pp. 559–568, ACM, 2011.

[87] D. T. Conway and J. L. Junkins, "Fusion of depth and color images for dense simultaneous localization and mapping," *Journal of Image and Graphics*, vol. 2, no. 1, pp. 64–69, 2014.

[88] D. Conway and J. L. Junkins, "Real-time mapping and localization under dynamic lighting for small-body landings," 2015.

[89] X.-S. Gao, X.-R. Hou, J. Tang, and H.-F. Cheng, "Complete solution classification for the perspective-three-point problem," *IEEE transactions on pattern analysis and machine intelligence*, vol. 25, no. 8, pp. 930–943, 2003.

[90] B. M. Haralick, C.-N. Lee, K. Ottenberg, and M. Nölle, "Review and analysis of solutions of the three point perspective pose estimation problem," *International*

journal of computer vision, vol. 13, no. 3, pp. 331–356, 1994.

[91] L. Quan and Z. Lan, "Linear n-point camera pose determination," *IEEE Transactions on pattern analysis and machine intelligence*, vol. 21, no. 8, pp. 774–780, 1999.

[92] A. Martinelli, "Closed-form solution of visual-inertial structure from motion," *International journal of computer vision*, vol. 106, no. 2, pp. 138–152, 2014.

[93] J. Kelly, "On the observability and self-calibration of visual-inertial navigation systems," *Technical Report*, 2008.

[94] D. Whitten, "LENS: LASR Embedded Navigation Stack." A hardware testbed for laboratory demonstrations, 2016.

[95] A. Johnson. personal communication, February 2014.

[96] M. Oren and S. K. Nayar, "Generalization of the lambertian model and implications for machine vision," *International Journal of Computer Vision*, vol. 14, no. 3, pp. 227–251, 1995.

[97] J.-Y. Bouguet, "Camera calibration toolbox for matlab," 2004.

[98] P. Furgale, J. Rehder, and R. Siegwart, "Unified temporal and spatial calibration for multi-sensor systems," in *2013 IEEE/RSJ International Conference on Intelligent Robots and Systems*, pp. 1280–1286, IEEE, 2013.

[99] P. Furgale, T. D. Barfoot, and G. Sibley, "Continuous-time batch estimation using temporal basis functions," in *Robotics and Automation (ICRA), 2012 IEEE International Conference on*, pp. 2088–2095, IEEE, 2012.

[100] P. J. Rousseeuw, "Least median of squares regression," *Journal of the American statistical association*, vol. 79, no. 388, pp. 871–880, 1984.

[101] D. Mortari, F. L. Markley, and P. Singla, "Optimal linear attitude estimator," *Journal of Guidance, Control, and Dynamics*, vol. 30, no. 6, pp. 1619–1627, 2007.

[102] G. Bradski, "Opencv computer vision library," *Dr. Dobb's Journal of Software Tools*, 2000.

[103] E. Rublee, V. Rabaud, K. Konolige, and G. Bradski, "Orb: an efficient alternative to sift or surf," in *Computer Vision (ICCV), 2011 IEEE International Conference on*, pp. 2564–2571, IEEE, 2011.

[104] J.-Y. Bouguet, "Pyramidal implementation of the affine lucas kanade feature tracker description of the algorithm," *Intel Corporation*, vol. 5, pp. 1–10, 2001.

[105] A. Quarteroni, R. Sacco, and F. Saleri, *Numerical mathematics*, vol. 37. Springer Science & Business Media, 2010.

[106] J. Kennedy, "Particle swarm optimization," in *Encyclopedia of machine learning*, pp. 760–766, Springer, 2011.

[107] M. J. Powell, "An efficient method for finding the minimum of a function of several variables without calculating derivatives," *The computer journal*, vol. 7, no. 2, pp. 155–162, 1964.

[108] P. J. Rousseeuw and K. Van Driessen, "Computing lts regression for large data sets," *Data mining and knowledge discovery*, vol. 12, no. 1, pp. 29–45, 2006.