LONG RANGE STEREO DATA-FUSION
FROM MOVING PLATFORMS

by

Gabe Sibley

A Dissertation Presented to the
FACULTY OF THE GRADUATE SCHOOL
UNIVERSITY OF SOUTHERN CALIFORNIA
In Partial Fulfillment of the
Requirements for the Degree
DOCTOR OF PHILOSOPHY
(Computer Science)

June 2007

Copyright 2007 Gabe Sibley
Abstract

This work is concerned with accurate and precise long-range depth estimation by filtering sequences of images. We investigate the problem with both stationary and mobile cameras. We find that with normally distributed sensor noise the non-linear properties of stereo triangulation, either from a motion-baseline or from a stereo-rig, can lead to statistically biased range estimates. In the static camera case we develop two novel filters to overcome this: a second order Gauss-Newton Filter, and an Iterated Sigma Point Kalman Filter. For the moving sensor case we develop a Sliding Window Filter for Simultaneous Localization and Mapping that concentrates computational resources on accurately estimating the immediate spatial surroundings by using a sliding time window of the most recent sensor measurements. The Sliding Window Filter exhibits many interesting properties, like continuous sub-mapping, lazy data association, constant time complexity, and robust estimation across multiple timesteps. Importantly, the Sliding Window Filter has the potential to approximate the optimal batch estimator in terms of optimality and efficiency. To this end we demonstrate convergence results for the Sliding Window Filter using real and simulated imagery.
# Contents

Abstract ii  
List Of Figures v  

1 Introduction 1  
   1.1 Long Range Depth Estimation from Moving Platforms 2  
   1.2 Contributions 6  

2 Bias Reduction and Filter Convergence for Long Range Stereo 11  
   2.1 Introduction 12  
   2.2 Statistical Bias in Stereo 13  
      2.2.1 Measurement Distribution 14  
      2.2.2 Derived Range Distribution 17  
      2.2.3 Bias Reduction 20  
   2.3 3D Estimation 22  
      2.3.1 3D Measurement Update 23  
      2.3.2 Gauss-Newton Measurement Update 27  
      2.3.3 Estimator Efficiency 29  
   2.4 Conclusion 31  

3 The Iterated Sigma Point Kalman Filter 33  
   3.1 Introduction 34  
   3.2 Derivation 35  
      3.2.1 Linear Error Propagation 41  
      3.2.2 Statistically Linearized Error Propagation 42  
      3.2.3 The Iterated Sigma Point Kalman Filter 44  
   3.3 Developed Example: Long Range Stereo 45  
      3.3.1 Comparison of methods 48  
         3.3.1.1 Batch Least Squares 50  
         3.3.1.2 Gauss-Newton Smoother 54  
         3.3.1.3 Iterated Extended Kalman Filter 55  
         3.3.1.4 EKF with 3D measurements 56
3.3.1.5 Unscented Kalman Filter ........................................ 56
3.3.1.6 Iterated Sigma Point Kalman Filter .......................... 57
3.4 Estimator Efficiency ................................................... 57
3.5 Long range Stereo Experiments ...................................... 60
  3.5.1 Note on Parametric Estimation .................................. 61
3.6 Conclusion ............................................................... 62

4 The Sliding Window Filter ............................................... 67
  4.1 Introduction .................................................................. 68
  4.2 Simultaneous Localization and Mapping as a Statistical Point Estimation Problem ............................................... 70
    4.2.1 Problem Formulation .............................................. 71
  4.3 The Sliding Window Filter ............................................. 77
    4.3.1 Overview ............................................................ 78
    4.3.2 The Effects of Downdating ..................................... 79
  4.4 Optimality and Efficiency ............................................. 83
  4.5 Application to Entry Descent and Landing .......................... 86
    4.5.1 Data Cleaning ....................................................... 87
    4.5.2 Stereo Convergence at the Focus of Expansion .......... 88
    4.5.3 Convergence Results ............................................. 89
  4.6 Additional Remarks on the Sliding Window Filter ............... 92
  4.7 Conclusion ............................................................... 95

5 Related Work .................................................................... 98
  5.1 Error Modeling and Bias in Stereo ................................ 98
  5.2 Simultaneous Localization and Mapping .......................... 99
  5.3 Correspondence, Data-association and Outlier Rejection ...... 108

6 Conclusion ....................................................................... 111
  6.1 Summary of Contributions ........................................... 114
  6.2 Final Remarks ............................................................ 115

Appendix A ......................................................................... 118
  A.1 Schur Complement, Gaussian Marginals, and the Matrix Inverse Lemma .................................................. 118

Appendix B ......................................................................... 123
  B.1 Coordinate Frame Relations, Spatial Operations and Kinematics .................................................. 123

Reference List ...................................................................... 126
### List Of Figures

1.1 FIDO rover at the JPL Mars Yard. An example application of dense point cloud mapping in the Mars Yard with a mobile platform is shown in Fig. 1.2.  
1.2 Dense map of the Mars Yard built using Visual Odometry. The upper left inset image is the left stereo frame. When the primary sensors are cameras, the sensor data are inherently dense, which makes data fusion computationally expensive.  
1.3 Unmanned Sea Surface Vehicle (USSV) in San Diego bay used for long range stereo experiments. Fig. 1.4 shows stereo data in a crowded bay.  
1.4 Long range stereo data taken from an USV in San Diego bay. The upper left inset image is the left stereo frame. Here the only static sensor data are on the shoreline; everything else (the other boats, the water, etc.) is in motion.  
1.5 Graphic showing Entry, Descent, and Landing (EDL) for the 2009 Mars Science Laboratory mission. In future missions the last ~200m of EDL will potentially use stereo data-fusion for landing site hazard detection and avoidance. Fig. 1.6 shows the features tracked during a simulated landing.  
1.6 Feature tracks over 25 frames from the left image of a stereo pair during EDL. Feature tracks are used to solve for platform motion and ground structure (SLAM). Here high resolution Mars Reconnaissance Orbiter (MRO) imagery is used to simulate the final phase of EDL.  
2.1 Feature measurement histogram of 10,360 measurements.
2.2 Standard model of linear perspective projection for stereo triangulation with axis aligned cameras. .............................................. 16

2.3 Range vs. Bias at 8 different ranges averaged over 10,000 trials. Clearly, bias is a strong function of range. Pixel standard deviation is 0.3 pixels in each pixel dimension, with no covariance. .............................................. 18

2.4 Range p.d.f. for $\sigma_d = 0.3$ pixels. True range is $51.08m = 1$ pixel disparity. Note that because of the tail, the mean is at $55.63m$, which is a bias of almost 10 percent. .............................................. 19

2.5 Bias-corrected stereo compared to traditional stereo. .............................................. 21

2.6 Kalman Filter of a sequence of 40 3D stereo measurements averaged over 10,000 trials. .............................................. 25

2.7 Gauss-Newton filter of a sequence of 40 image measurements averaged over 10,000 trials. .............................................. 26

2.8 Gauss-Newton and 3D EKF estimator efficiency compared against the Cramer-Rao lower bound over a sequence of 10 measurements of a feature 25m away. At each step estimator variance is found via Monte-Carlo simulation over 10,000 trials. .............................................. 30

3.1 Probability density function for disparity derived from a 3D landmark at 4m with a range standard deviation of 1.5m. The p.d.f. is clearly non-Gaussian, non-symmetric, and has a heavy tail. The difference between the p.d.f. mean and the true mean can lead to bias and apparent divergence when filtering stereo measurements. .............................................. 46

3.2 Error propagation from the state space to the measurement space for stereo ranging. Propagation of this type takes place when we compute $\hat{Z}$ and $\text{cov}(\hat{Z})$ in (3.12) or $\text{cov}(z)$ via $\text{HPH}^T$. Statistical error propagation captures the mean of the analytical p.d.f. more accurately, which is biased long due to the heavy tail. .............................................. 49

3.3 Filter convergence for a sequence of 20 measurements of a feature at 25m averaged over 1000 trials. Image noise is modeled as uncorrelated standard deviation of 0.25 pixels. .............................................. 51
3.4 Squared Filter Error vs. Measurement number for a sequence of 20 measurements of a feature at 25m averaged over 1000 trials. The improvement of the UKF over the Gauss-Newton method can be attributed to statistical error propagation. The improvement of the ISPKF over the UKF can be attributed to Iteration. Notice that filters incorporating prior information modeled as Gaussian (all but the batch filter) suffer from initialization problems – either over confidence or under confidence. This is due to modeling range uncertainty as Gaussian, which is incorrect.

3.5 Range probability density function analytically derived through the stereo triangulation equations (inverse of the sensor model). Propagating measurements that are normally distributed results in a non-Gaussian range distribution. Estimators that model the feature state as a 3D Gaussian will suffer from this mis-representation. The filter will likely exhibit either bias or divergence. Both iteration and statistical linearization can help overcome these effects. All the filters except the batch-optimal filter in Fig. 3.3 suffer from this mis-representation so some extent.

3.6 Cramer Rao Lower Bound for the ISPKF, the Gauss-Newton Smoother, and the optimal Batch filter. As expected the Batch filter tracks the CRLB, and the Gauss-Newton Smoother tracks the CRLB for the first 5 steps. However, after 5 steps the Smoother starts rolling old measurements into a prior, and the ISPKF starts to out perform the Smoother. As a recursive filter the ISPKF is nearly as efficient as the Batch filter.

3.7 Sample wall image from experiments using the FIDO Mars Rover in Fig. 3.8. The wall is approximately 6m from the robot. Over 20 frames there were 8005 stereo measurements of 899 different SIFT features.

3.8 FIDO uses rectified camera models with a resolution of 512×384 pixels, a horizontal FOV of 37.22°, a vertical FOV of 29.53° and a baseline of ~20cm.

3.9 Sample wall image from experiments using a stereo setup configured analogously to the LAGR robot in Fig. 3.10. The wall is approximately 20m from the robot.

3.10 Cameras mounted on the LAGR robot use rectified camera models with a resolution of 512×384 pixels, a horizontal FOV of 54.08° and a vertical FOV of 40.87°.
3.11 Sample image of the checker wall sequence. The wall is approximately 12m in front of the robot. For scale, each checker is a US Letter size piece of paper (8.5 in by 11 in). Over 733 frames 105 corner features were tracked from the center portion of the pattern. The lighting and camera shutter speed were deliberately set to induce image noise so that the feature tracker (a sub-pixel saddle point corner fit in this case [60]) would produce high measurement noise. This is apparent in Fig. 3.12.

3.12 Histogram of 77,049 measurements for real data from the checker wall sequence (for the horizontal pixel dimension). This sequence uses rectified camera models with a resolution of 1024x768 pixels, a horizontal FOV of 54.08°, a vertical FOV of 40.87° and baseline of ~4cm. This small baseline setup was deliberately chosen to challenge the estimators in this chapter with noisy data and small disparities.

3.13 Results from real data showing error measured against RANSAC ground truth plane for the Batch, ISPKF and Gauss-Newton filter for the checker wall sequence.

4.1 Basic structure of the Jacobians, $H$, $D$ and $L$. $H = [H_{11}^T, H_{12}^T, ..., H_{nm}^T]^T$, where $H_{ij} = \frac{\partial h_{ij}}{\partial \mathbf{x}}$. Also, $H$ has two components: $H_p$ is the Jacobian of $h$ with respect to the pose parameters, $\mathbf{x}_p$, and $H_m$ is the Jacobian of $h$ with respect to the map parameters, $\mathbf{x}_m$. Thus, $H_{p,ij} = \frac{\partial h_{ij}}{\partial \mathbf{x}_p}$, and $H_{m,ij} = \frac{\partial h_{ij}}{\partial \mathbf{x}_m}$.

4.2 The sparse structure of least squares SLAM system matrix is due to contributions from three components: the measurement information matrix $H^T R^{-1} H$, the process information matrix $D^T Q^{-1} D$, and the prior information matrix $L^T P L$. The measurement information matrix has three distinct components, $U = H_p^T R^{-1} H_p$, $W = H_m^T R^{-1} H_p$, and $V = H_m^T R^{-1} H_m$. The pattern of the system matrix in the SLAM problem depends on information contributed (or not) from the dynamical process model and any prior information. With no process model information and no prior information the problem is equivalent to Photogrammetric Bundle Adjustment (the structure of this problem is visible on the right). Including a process model makes the lower right $m \times m$ “process block” tridiagonal. Including a prior can potentially cause complete fill-in of the upper left $n \times n$ “map-block”.
4.4 Space of solutions spanned by the Sliding Window Filter. The interesting point is that the region of constant time algorithms exhibit the potential to match the more expensive offline algorithms, though of course not at loop closure.

4.5 Sliding Window Filter comes close to the Full SLAM solution. Sliding Window of 2 is close to optimal $1/n$ full batch curve. Each curve is a trial for different size time window, averaged over 50 trials, with 0.1 pixel std.dev measurement noise. 1.0m std.dev process noise. Because VO does not combine information over time, it does not reduce uncertainty as time passes.

4.6 Estimator error after 25 frames vs Distance off the optical axis, as measured from a landing vehicle descending at 10ms, capturing frames at 10hz, starting at 100m and stopping at 75m. The FOV is ~25 deg.; the stereo baseline is 1m; image resolution is 512x384.

4.7 For validation, here the SWF exhibits $1/n$ convergence for a sequence with stationary cameras and tracked fiducials.

4.8 Fiducials used for experimental analysis. Each fiducial is localized to sub-pixel accuracy (by fitting a saddle point to the checker pattern). Each fiducial has a unique bar-code that can be read automatically.

4.9 Convergence results for moving cameras for Mars EDL experiment data. 150 features are tracked over 10 frames and fused with a SWF of 3, 4, 5, and 10 frames. This plot demonstrates slightly better than $1/n$ convergence because range uncertainty is also shrinking as the cameras get closer to the wall.

4.10 Example left image from the EDL sequence. This sequence used grayscale cameras with 1024x768 resolution, 10cm baseline, 25 deg. FOV.
Chapter 1

Introduction

This work is concerned with accurate and precise 3D map estimation by filtering sequences of images. In particular, we are interested in improving the range resolution of stereo vision by exploiting temporal correspondences between images. We investigate the problem with both stationary and mobile cameras.

In robotics there are many situations in which accurate high-resolution spatial perception is useful for successfully performing a task. For instance, without prior models, good sensing is undoubtedly beneficial for tight obstacle avoidance, delicate mobile manipulation, modeling complicated terrain, subtle motion detection or moving object tracking from a moving platform. Typically this reduces to a data-fusion problem in which a sensor (usually undergoing uncertain, dynamic motion) must combine noisy measurements into a single underlying sensor-relative state estimate. In this context, some examples we study are shown in Figs. 1.1-1.6.
1.1 Long Range Depth Estimation from Moving Platforms

We are interested in estimating the depth of objects accurately out to the maximum range of the sensor, i.e. a stereo disparity of 1 pixel or less. Improving depth estimation in stereo systems is an important pursuit. For instance, better stereo range resolution will enhance an autonomous systems ability to perform tasks such as navigation, long range path planning, obstacle avoidance, mapping and localization and high-speed driving. Unbiased sensing is a prerequisite for these algorithms to perform well.

In this thesis we strive for unbiased, efficient estimators of range. Efficient estimators are those that reduce uncertainty as quickly as possible, without exhibiting over confidence in the state estimate. Such methods yield the minimum mean squared error (MMSE), and static, linear point estimation problems are able to reduce the estimator mean squared error and estimator variance by $1/n$, where $n$ is the number of distinct measurements taken over time. By reference to Fisher’s information inequality, efficient estimators are said to track the Cramer-Rao Lower Bound (CRLB).

Put differently, by combining multiple independent normally distributed measurements, we seek range estimation algorithms that reduce error as quickly as possible, extracting from each measurement the maximal amount of information about the underlying object of interest. The signature of such performance for a stationary camera and scene is a near $1/n$ reduction in scene reconstruction error, as measured against some known ground truth. Generally, error reduction by fusing multiple measurements is a central tenant of estimation theory.
Figure 1.1: FIDO rover at the JPL Mars Yard. An example application of dense point cloud mapping in the Mars Yard with a mobile platform is shown in Fig. 1.2

Figure 1.2: Dense map of the Mars Yard built using Visual Odometry. The upper left inset image is the left stereo frame. When the primary sensors are cameras, the sensor data are inherently dense, which makes data fusion computationally expensive.
Figure 1.3: Unmanned Sea Surface Vehicle (USSV) in San Diego bay used for long range stereo experiments. Fig. 1.4 shows stereo data in a crowded bay.

Figure 1.4: Long range stereo data taken from an USV in San Diego bay. The upper left inset image is the left stereo frame. Here the only static sensor data are on the shoreline; everything else (the other boats, the water, etc.) is in motion.
Figure 1.5: Graphic showing Entry, Descent, and Landing (EDL) for the 2009 Mars Science Laboratory mission. In future missions the last \( \sim 200 \)m of EDL will potentially use stereo data-fusion for landing site hazard detection and avoidance. Fig. 1.6 shows the features tracked during a simulated landing.

Figure 1.6: Feature tracks over 25 frames from the left image of a stereo pair during EDL. Feature tracks are used to solve for platform motion and ground structure (SLAM). Here high resolution Mars Reconnaissance Orbiter (MRO) imagery is used to simulate the final phase of EDL.
Our practical motivation for data fusion is to see more accurately at distance. We seek algorithms that come as close as possible to achieving the lower bound on estimation error, and ideally to understand the reasons for any remaining discrepancy from the theoretical lower bound. Note that, when the camera is in motion, the error reduction will not in general follow the $1/n$ pattern, though we can often establish the expected minimum error using the CRLB.

Again, for stationary systems, convergence results for estimating the spatial structure, or environment “map”, will ideally exhibit a signature $1/n$ reduction in MMSE, relative to some “ground truth”, where $n$ is the number of image measurements at distinct times. Typical Simultaneous Localization and Mapping or Structure from Motion experimentalists do not report such convergence results. One path to establish ground truth is to observe scenes with known geometry to which we can register our state estimates. This is precisely our strategy for the Mars Entry, Descent, and Landing experiments described later, in which we show near optimal convergence results with real data.

### 1.2 Contributions

In our investigation of long-range stereo data fusion, we begin with an analysis of filtering stereo image sequences taken from a stationary rig. Measurement errors from stereo camera systems have often been approximated as 3D Gaussians, where the mean is derived by triangulation and the covariance by linearized error propagation. However, there are two problems that arise when filtering such 3D measurements. First, stereo
triangulation suffers from a range dependent statistical bias; when filtering this leads to over-estimating the true range. Second, filtering 3D measurements derived via linearized error propagation leads to apparent filter divergence; the estimator is biased to underestimate range. To address the first issue, we examine the statistical behavior of stereo triangulation and show how to remove the bias by series expansion. The solution to the second problem is to filter with image coordinates as measurements instead of triangulated 3D coordinates. We show that bias is reduced by more than an order of magnitude, and that the variance of the estimator approaches the Cramer-Rao lower bound.

Next, we investigate the use of statistical linearization to improve iterative non-linear least squares estimators. In particular, we look at improving long range stereo by filtering feature tracks from sequences of stereo pairs. A novel filter called the Iterated Sigma Point Kalman Filter (ISPKF) is developed from first principles; this filter is shown to achieve superior performance in terms of efficiency and accuracy when compared to the Extended Kalman Filter (EKF), Unscented Kalman Filter (UKF), and Gauss-Newton filter. We also compare the ISPKF to the optimal Batch filter and to a Gauss-Newton Smoothing filter. For the long range stereo problem the ISPKF comes closest to matching the performance of the full batch MLE estimator. Further, the ISPKF is demonstrated on real data in the context of modeling environment structure from long range stereo data.

Having studied static camera configurations, we then move onto cases in which the camera moves through a static scene. We develop a new Sliding Window Filter (SWF) that is an on-line constant-time solution to the feature-based 6-degree-of-freedom full
Batch Least Squares Simultaneous Localization and Mapping (SLAM) problem. We contend that for SLAM to be useful in large environments and over extensive run-times, its computational time complexity must be constant, and its memory requirements for map storage should be at most linear. Under this constraint, the “best” algorithm will be the one that comes closest to matching the all-time maximum-likelihood estimate of the full SLAM problem, while also maintaining consistency. We start by formulating SLAM as a Batch Least Squares state estimation problem, and then show how to modify the Batch estimator into an approximate sliding window Batch/Recursive framework that achieves constant time complexity and linear space complexity. Viewing SLAM from the sliding window Least Squares perspective is very useful for understanding the structure of the problem. This perspective is general, capable of subsuming a number of common estimation techniques such as Bundle Adjustment and Extended Kalman Filter SLAM. By tuning the sliding window, the algorithm can scale from the exhaustive Batch solutions to fast incremental solutions; if the window encompasses all time, the solution is algebraically equivalent to full SLAM; if only one time step is maintained, the solution is algebraically equivalent to the Extended Kalman Filter SLAM solution. We also point out that the SWF enables other interesting properties, like continuous sub-mapping (as opposed to breaking the SLAM problem into many smaller disjoint estimation problems), lazy data association (which gives the ability to refine past data associations), undelayed or delayed landmark initialization (freedom in choosing when to initialize
landmarks in the filter), and incremental robust estimation (spreading the possibility of outlier detection out over many timesteps).

In the demonstration of SWF convergence on real data the issues of robust estimation in feature tracking invariably come up. We address the correspondence, data association and outlier rejection problems using a combination of Moravec’s consistency check[70] and robust M-estimation[41]. Importantly, to achieve convergence with real data, we find that it is crucial to refine sub-pixel correspondences during each iteration of the SWF iterative update. Bi-linearly interpolated least squares image patch matching is used to match patches to sub-pixel accuracy[59], to estimate the 2D feature variance, and to estimate the image reference variance. For image warping, we first assume landmarks live on locally planar surface patches in 3D, and we relate corresponding image patches via a projective warping homography that depends on the landmark surface plane and the camera positions.

We test the algorithm with real data captured to emulate Entry, Descent, and Landing conditions for a Mars lander. Using this imagery we are able to demonstrate the expected near $1/n$ convergence in ground structure uncertainty and estimator error. Compared to single frame stereo, such technology can allow for the accurate determination of ground structure from greater altitude during EDL, and hence allow more time for hazard avoidance maneuvers prior to touchdown.

This demonstrated convergence will be important in many other applications, such as enabling greater lookahead distance for obstacle detection, hence safe driving at higher
speeds. Ultimately, the goal is to enable high quality long range vision based depth estimation for all sorts of moving vehicles; in general, any system that wishes to estimate range from vision can benefit from temporal data fusion.
Chapter 2

Bias Reduction and Filter Convergence for Long Range Stereo

In this chapter we are concerned with improving long range stereo by filtering image sequences. Often measurement errors from stereo camera systems are approximated as 3D Gaussians, where the mean is derived by triangulation and the covariance by linearized error propagation[111, 112, 61]. However, there are two problems that arise when filtering such 3D measurements. First, stereo triangulation suffers from a range dependent statistical bias; when filtering this leads to over-estimating the true range. Second, filtering 3D measurements derived via linearized error propagation leads to apparent filter divergence; the estimator is biased to under-estimate range. To address the first issue, we examine the statistical behavior of stereo triangulation and show how to remove the bias by series expansion. The solution to the second problem is to filter with image coordinates as measurements instead of triangulated 3D coordinates. Compared to the
traditional approach, we show that bias is reduced by more than an order of magnitude, and that the variance of the estimator approaches the Cramer-Rao lower bound.

2.1 Introduction

This chapter details our efforts to enhance long range depth estimation in stationary stereo systems by filtering feature measurements from image sequences. We are interested in estimating the depth of objects accurately out to the maximum range of the sensor, i.e. a stereo disparity of 1 pixel or less. Improving depth estimation in stereo systems is an important pursuit. For instance, better stereo range resolution will enhance a robot’s ability to perform tasks such as navigation, long range path planning, obstacle avoidance, mapping and localization and high-speed driving. Unbiased sensing is a prerequisite for these algorithms to perform well.

In the balance of this chapter we will encounter two problems with 3D stereo error modeling and in turn describe their solutions. First, because of the non-linearity in stereo triangulation, we will see that range estimates produced by standard triangulation methods are statistically biased. While bias in stereo is a known phenomenon, previous research focused on how range bias is induced from uncertain camera positions[86, 112, 16], or dismissed it as insignificant [62]. Second, filtering sequences of 3D measurements from stereo leads to biased range estimates when the uncertainty of each 3D measurement is modeled by standard linearized error propagation techniques; this stems from the fact that the uncertainty model is biased.
For the former issue, analyzing the statistical behavior of stereo triangulation leads us to new triangulation equations based on series expansion; this new bias-corrected formulation is shown to be an improvement over traditional stereo triangulation by more than order of magnitude. For the latter biased filter problem, we find that formulating the filter with image coordinates as measurements leads to efficient and unbiased estimation of range. Lastly, using the Fisher information inequality we show that the combination of bias-corrected stereo and a Gauss-Newton recursive filter yield estimates that closely approach the minimum variance Cramer-Rao lower bound.

2.2 Statistical Bias in Stereo

Consider a general stereo triangulation function \( s : \mathbb{R}^4 \rightarrow \mathbb{R}^3 \)

\[
s(z) = x = \begin{bmatrix} x_0 \\ x_1 \\ x_2 \end{bmatrix}
\]  

(2.1)

where the current observation, \( z \) is the vector of pixel coordinates \( [u_1, v_1, u_2, v_2]^T \), and the pixels \( [u_1, v_1]^T \) and \( [u_2, v_2]^T \) are projections of \( x \) into the two camera image planes. Let \( x_2 \) be the range component of \( x \) - i.e. \( x_2 \) is aligned with the optical axis of the cameras.
We are interested in how a particular model of pixel measurement uncertainty will translate into range uncertainty. Before we address the issue of bias in more detail we first need to establish an appropriate observation probability density function.

### 2.2.1 Measurement Distribution

A common approximation is that many measurements of a stationary feature point, such as corner features [25, 36], follow a normal distribution [81, 62, 63, 50]. To establish how features are actually distributed we have performed the following experiment: we took a sequence of images from a stationary camera of a stationary checker board and tracked the corners over time with sub-pixel accuracy [60]; we then re-centered each feature track about zero by subtracting its mean. For each pixel dimension a histogram of all the measurements is then plotted. This histogram approximates the true distribution we should expect in measurements. Qualitatively, the histogram in Fig. 2.1 indicates that the distribution is close to Gaussian.
Figure 2.1: Feature measurement histogram of 10,360 measurements.
Figure 2.2: Standard model of linear perspective projection for stereo triangulation with axis aligned cameras.
2.2.2 Derived Range Distribution

Recall the fronto-parallel configuration, whose geometry is shown in Fig. 2.2. We will derive the range p.d.f. using this simple geometry, though the methods and results used here apply to other camera models as well. Using Fig. 2.2, the stereo equations are

\[
\mathbf{s}(z) = \mathbf{x} = \begin{bmatrix} x_0 \\ x_1 \\ x_2 \end{bmatrix} = \begin{bmatrix} u_1 b/d \\ v_1 b/d \\ (b f)/d \end{bmatrix} \tag{2.2}
\]

where the last element of \( \mathbf{x} \) is the range component, and \( d = (u_1 - u_2) \) is the disparity. Monte-Carlo simulation using these equations indicates that if image feature positions, and hence disparity, are normally distributed, then the expected range will be biased toward over estimating the true value\(^1\). The bias is empirically visible in Fig. 2.3.

Analytically, the bias can be seen by deriving the range p.d.f., \( f_{x_2}(x_2) \), from the disparity p.d.f., \( f_d(d) \) [62, 18, 45]. From (2.2) we have \( x_2 = s_2(d) = k/d \), where \( k = bf \). Since \( s_2 \) and \( s_2^{-1} \) are continuously differentiable, then

\[
f_{x_2}(x_2) = f_d(d) \left| \frac{\partial f_d(d)}{\partial x_2} \right| = f_d[s_2^{-1}(x_2)] \left| \frac{\partial s_2^{-1}(x_2)}{\partial x_2} \right|
\]

\(^1\)Throughout this chapter we use linear camera models with a resolution of 512x384 pixels, a horizontal FOV of 68.12 degrees and vertical FOV 51.3662 of degrees
Figure 2.3: Range vs. Bias at 8 different ranges averaged over 10,000 trials. Clearly, bias is a strong function of range. Pixel standard deviation is 0.3 pixels in each pixel dimension, with no covariance.
Figure 2.4: Range p.d.f. for $\sigma_d = 0.3$ pixels. True range is 51.08m = 1 pixel disparity. Note that because of the tail, the mean is at 55.63m, which is a bias of almost 10 percent.
where $|·|$ denotes absolute value of the Jacobian determinant and $s_2^{-1} = d = k/x_2$. Thus, since $f_d(d)$ is modeled as Gaussian

$$f_{x_2}(x_2) = \frac{k}{\sqrt{2\pi\sigma_d}x_2^2}exp\left(-\frac{(k/x_2 - \mu_d)^2}{2\sigma_d^2}\right)$$

(2.3)

where $\mu_d$ and $\sigma_d$ are the disparity mean and variance. The mean of (2.3) is

$$\mu_{x_2} = E[x_2] = \int_{-\infty}^{\infty} x_2 f_{x_2}(x_2) dx_2$$

which unfortunately does not appear to have an analytical solution, so we resort to numerical integration. Plots of $f_{x_2}(x_2)$ are shown in Fig. 2.4; clearly, for distant features the p.d.f. is non-Gaussian, non-symmetric and exhibits a long tail. The tail shifts the mean away from the true range and hence we see the source of bias in stereo.

### 2.2.3 Bias Reduction

Naturally, we would like an unbiased method for calculating range. Recall that the distribution on $\hat{d}$ is approximately Gaussian, the mean of which we take to approximate some true underlying state, $d$. If the true value of $d$ was known, then the true unbiased range could be calculated with $x_2 = s_2(d)$, but due to the variation in $\hat{d}$, $s_2(\hat{d})$ is, as we have
Figure 2.5: Bias-corrected stereo compared to traditional stereo.

seen, slightly biased. However, if the variation of \( \hat{d} \) around \( d \) is small, then a Taylor series expansion of \( s_2 \) may provide a better estimate [14].

\[
\bar{s}_2(d) \approx s_2(d) + \left. \frac{\partial s_2}{\partial \hat{d}} \right|_{d} (\hat{d} - d) + \frac{1}{2} (\hat{d} - d)^2 \left. \frac{\partial^2 s_2}{\partial \hat{d}^2} \right|_{d}
\]
Taking the expectation, noting that $E[\hat{d} - d] = 0$, that $E[(\hat{d} - d)^2]$ is the definition of variance, and replacing $d$ with $\hat{d}$ we get

$$s_2(d) \approx \hat{s}_2(\hat{d}) - \frac{1}{2} \text{var}(\hat{d}) \frac{\partial^2 s_2}{\partial d^2} \bigg|_{\hat{d}}$$

(2.4)

which is the new range equation that we use to correct for bias. Note that this formulation requires accurate knowledge of the measurement variance; which is reasonable. Looking at Fig. 2.5 the improvement is immediately visible for small disparities; in fact, for the ranges shown, bias is reduced by more than an order of magnitude. Note that higher order series approximation, which should theoretically provide a better estimate, will depend on higher order moments, $E[(\hat{d} - d)^n], n > 2$. But if the input distribution has negligible higher order moments, then the second term in (2.4) makes use of all the available information. By considering the variance and how it impacts the range distribution, this bias-correction method largely removes the bias from long range stereo.

### 2.3 3D Estimation

In this section we uncover another type of bias that results from filtering a sequence of 3D estimates produced by triangulation and linearized error propagation. To alleviate this we develop a non-linear Gauss-Newton iterative measurement update using image space measurements instead of 3D measurements. Finally, the statistical efficiency of the
3D measurement update and the Gauss-Newton update are compared to the Cramer-Rao lower bound.

### 2.3.1 3D Measurement Update

Let $x \in \mathbb{R}^3$, $\hat{x} \in \mathbb{R}^3$, $z_{3D} \in \mathbb{R}^3$ denote the current state, current state estimate and the current 3D observation, respectively. For the case at hand, the current observation, $z_{3D}$, is the vector found via bias-corrected stereo. The state estimate and observation are independent realizations of multivariate Gaussian distributions: $z_{3D} \sim N(s(z), R_{3D})$ and $\hat{x} \sim N(x, \hat{P})$ where $R_{3D}$ and $\hat{P}$ are the measurement and state error covariance matrices, respectively. The error covariance matrix $R_{3D}$ is found via error propagation of image errors

$$R_{3D} = \frac{\partial s}{\partial z} R \frac{\partial s}{\partial z}^T, \quad R = \begin{bmatrix} \Sigma_l & 0 \\ 0 & \Sigma_r \end{bmatrix}$$

where $s$ is the bias corrected stereo equation and $\Sigma_l$ and $\Sigma_r$ are 2x2 error covariance matrices from the left and right images, respectively.

In this scenario the sensor model $h_{3D} : \mathbb{R}^3 \rightarrow \mathbb{R}^3$ is the vector function that returns a predicted measurement for $z_{3D}$ given $x$. If $x_{ws}$ is the global position of the stereo head with orientation matrix $R_{ws}$ then the generative sensor model is, $h_{3D}(x) = R_{ws}^T x - R_{ws}^T x_{ws}$. Note that while we focus here on the stationary case and solving issues of bias and filter convergence, this formulation extends to the mobile sensor case (an issue we
are actively working on that is beyond the scope of this chapter). Following standard notation [64] the Kalman Filter update equations for this system are

\[
\hat{x}_{k+1} = \hat{x}_k + K(z_{3D} - h_{3D}(\hat{x}_k))
\]

\[
\hat{P}_{k+1} = (I - KH_k)\hat{P}_k
\]

\[
K = \hat{P}_k H_k^T (H_k \hat{P}_k H_k^T - R_{3D})^{-1}
\]

where \(H_k\) is the Jacobian of \(h_{3D}\). Filtering with this setup leads to the situation depicted in Fig. 2.6, which clearly shows what is called *apparent* divergence - i.e. convergence to the wrong result [31]. This can be explained by the fact that linearized error propagation in (2.5) gives quadratically larger range variance for more distant features. This means that the weighted average in (2.6) will always place more confidence in closer measurements and \(\hat{x}^+\) will be biased toward the short measurements. In essence, linearized error propagation leads to over confidence for shorter measurements, which in turn leads to serious filter bias.
Figure 2.6: Kalman Filter of a sequence of 40 3D stereo measurements averaged over 10,000 trials.
Figure 2.7: Gauss-Newton filter of a sequence of 40 image measurements averaged over 10,000 trials.
2.3.2 Gauss-Newton Measurement Update

Instead of using the triangulated point $z_{3D}$ as the observation, let the observation again be the vector of pixel coordinates $z = [u_1, v_1, u_2, v_2]^T$. Thus our sensor model $h : \mathbb{R}^3 \rightarrow \mathbb{R}^4$ is the vector function that projects $x$ into the left and the right images,

$$h(x) = \begin{bmatrix} h_l(x) \\ h_r(x) \end{bmatrix}$$

(2.7)

where $h_l : \mathbb{R}^3 \rightarrow \mathbb{R}^2$ and $h_r : \mathbb{R}^3 \rightarrow \mathbb{R}^2$ are the left and right camera projection functions. Depending on the camera models in use, $h_l$ and $h_r$ can be formulated in a variety of ways [105, 109]. We only require that these functions and their first derivatives are available, and otherwise leave them unspecified.

For convenience we choose to formulate the measurement update as an iterative Gauss-Newton method, which is equivalent to an iterated Extended Kalman Filter [5]. To integrate prior information about $\hat{x}$ we write the current state estimate and and current observation as a single measurement vector

$$Z = \begin{bmatrix} z \\ \hat{x} \end{bmatrix}, \quad g(x) = \begin{bmatrix} h(x) \\ x \end{bmatrix}$$

For the first measurement, the filter is initialized with $\hat{x} = z_{3D}$ and $\hat{P} = R_{3D}$ which are calculated by bias-corrected triangulation and linearized error propagation in as in
section 2.3.1. Since the current observation and state estimate are realizations of independent normal distributions we have \( Z \sim N(g(x), C) \) where \( C = \begin{bmatrix} R & 0 \\ 0 & P \end{bmatrix} \). Given the measurement \( Z \), we can write the likelihood function

\[
L(x) = \frac{1}{\sqrt{(2\pi)^d|C|}} \exp \left( -\frac{1}{2} (Z - g(x))^T C^{-1} (Z - g(x)) \right)
\] (2.8)

where \(| \cdot |\) is the determinant. The maximum likelihood estimate for this expression is \( \hat{x}^+ = \arg\max_x L(x) \), whose solution is equivalent the solution minimizing the negative log-likelihood, \( \arg\min_x \ell(x) \),

\[
\ell(x) = \frac{1}{2}(Z - g(x))^T C^{-1} (Z - g(x)) + k
\] (2.9)

where \( k \) is a constant. If we let \( S^T S = C^{-1} \) and

\[
r(x) = S(Z - g(x))
\] (2.10)

then (2.9) is a non-linear least squares problem to minimize \( r(x)^T r(x) \). The Gauss-Newton method to solve non-linear problems of this form is the sequence of iterates [20]

\[
x_{i+1} = x_i - (J(x_i)^T J(x_i))^{-1} J(x_i)^T r(x_i)
\] (2.11)
where $J$ is the Jacobian of (2.10). Noting that $J = -SG_i$ where $G_i$ is the Jacobian of $g(x_i)$, (2.11) becomes

$$x_{i+1} = (G_i^T C^{-1} G_i)^{-1} G_i^T C^{-1} (Z - g(x_i) + G_i x_i)$$

which is the familiar normal equation solution. Once iterated to convergence the covariance $\hat{P}^+$ can then be approximated using $\hat{P}^+ = (G_i^T C^{-1} G_i)^{-1}$. As noted in [5], this is equivalent to the iterated Extended Kalman Filter measurement update.

Filtering with this setup leads to the situation depicted in Fig 2.7. Typically, the measurement update converges after 3 to 4 iterations. Compared to the 3D measurement update, the fact that the Gauss-Newton (IEKF) method converges without bias is not surprising considering that we avoid the intermediate stereo triangulation and linearized error propagation for calculating the 3D error covariance matrix.

### 2.3.3 Estimator Efficiency

Having derived a bias-corrected estimator it is important to address its efficiency, that is, how well it approximates a minimal variance estimate of the parameters. The information inequality, $\text{cov}_x(x) \geq I_z(x)^{-1}$, defines such a bound, which is called Cramer-Rao lower bound[18]. Here the Fisher information matrix $I_z(x)$ is given by the symmetric matrix
Figure 2.8: Gauss-Newton and 3D EKF estimator efficiency compared against the Cramer-Rao lower bound over a sequence of 10 measurements of a feature 25m away. At each step estimator variance is found via Monte-Carlo simulation over 10,000 trials.
whose $ij^{th}$ element is the covariance between first partial derivatives of the measurement log-likelihood function,

$$
I_{z,i,j}(x) = \text{cov}_x \left( \frac{\partial \ell_z}{\partial x_i}, \frac{\partial \ell_z}{\partial x_j} \right)
$$

(2.12)

The measurement log-likelihood function is $\ell_z(x) = \frac{1}{2}(z - h(x))^T R^{-1} (z - h(x)) + k$.

For a multivariate normal distribution (3.20) reduces to[95, 112]

$$
I_{z,i,j}(x) = \frac{\partial h}{\partial x_i}^T R^{-1} \frac{\partial h}{\partial x_j}
$$

For $n$ independent identically distributed (i.i.d.) measurements the Fisher information is simply $nI$. An estimator that achieves the CRLB is said to be efficient. Fig. 2.8 shows range variance convergence for the Gauss-Newton estimator; this demonstrates that the Gauss-Newton stereo estimator is efficient.

### 2.4 Conclusion

In our efforts to improve long range stereo by filtering image sequences we have come across two problems: the first is that stereo triangulated range estimates are statistically biased. To address this we have re-expressed the stereo triangulation equations using a second order series expansion. This new formulation reduces bias in stereo triangulation by more than an order of magnitude. The second problem is that temporal filtering of 3D stereo measurements also leads to biased estimates. The solution to this problem is to filter with image coordinates as measurements instead of triangulated 3D coordinates.
Finally, using the Fisher information inequality we show that the bias-corrected Gauss-Newton stereo estimator approaches the minimum variance Cramer-Rao lower bound. While the scope of this chapter is constrained to address stereo bias and estimator efficiency, our ultimate goal is to filter feature points from a moving platform. This is a task that requires a solid solution to the simultaneous localization and mapping problem, which we address in Ch. 4. In the next chapter we derive an Iterated Sigma Point Kalman Filter that also addresses the bias and convergence issues raised in this chapter.
Chapter 3

The Iterated Sigma Point Kalman Filter

This chapter investigates the use of statistical linearization to improve iterative nonlinear least squares estimators. Statistical linearization has a beneficial bias-correction property, like what we saw in Ch. 2, but without the need to compute second order derivatives. Just as in Ch. 2 the goal here is to improve long range stereo by filtering feature tracks from sequences of stereo pairs.

A novel filter called the Iterated Sigma Point Kalman Filter (ISPKF) is developed; this filter is shown to achieve superior performance in terms of efficiency and accuracy when compared to the Extended Kalman Filter (EKF), Unscented Kalman Filter (UKF), and Gauss-Newton filter. We also compare the ISPKF to the optimal Batch filter and to a Gauss-Newton Smoothing filter. For the long range stereo problem the ISPKF comes closest to matching the performance of the full batch MLE estimator. Further, the ISPKF is demonstrated on real data in the context of modeling environment structure from long range stereo data.
3.1 Introduction

This chapter introduces the Iterated Sigma Point Kalman Filter (ISPKF), which is a principled extension to statistical linearization methods such as the Unscented Kalman Filter (UKF), Central Difference Kalman Filter (CDKF), and Sigma Point Kalman Filter (SPKF) [107, 44, 49, 106]. In the seminal work [106] the Iterated Sigma Point Kalman Filter was discussed as future work as an “ad-hoc improvement” to non-iterated methods. In contrast, we highlight the fundamental importance of iteration in solving non-linear least squares problems. To this end we develop the ISPKF directly from the underlying probability density function, grounding it in non-linear optimization theory and Newton’s Method. For non-linear problems we emphasize that the measurement update for methods that do not iterate, such as the EKF, UKF or SPKF, cannot be expected to achieve the Maximum Likelihood solution, whereas iterative methods are provably convergent and have a rich convergence theory [20].

After deriving the ISPKF, we compare it to a number of techniques including an optimal batch non-linear least squares solution, a Gauss-Newton filter, an Extended Kalman Filter, (as in [91]) a Gauss-Newton Smoother, and an Unscented Kalman Filter. We find that the ISPKF both converges faster and with greater accuracy when compared to the other estimators.
We are interested in improving the range resolution of stereo to the point where we can accurately estimate depth from disparities on the order of one pixel or less. Our approach is to filter measurement sequences of image features over time. Temporal filtering of such measurements can improve depth estimation, which is important for mobile robots. For instance, better stereo range resolution increases a robot’s ability to perform tasks such as navigation, long range path planning, obstacle avoidance, mapping and localization and high-speed driving.

Stereo is an interesting problem on which to apply the ISPKF because it is non-linear, suffers from inherent statistical bias, and because the true range probability distribution is non-Gaussian. When using traditional filters, such as the Extended Kalman Filter, these issues can cause apparent divergence (e.g. convergence to the wrong value). While the difficulties with bias in stereo can be largely overcome without recourse to statistical linearization, [91] we show that using statistical linearization results in significant improvements. Further, we show that the iterative nature of the new estimator greatly improves upon results of non-iterative statistical linearization methods like the UKF.

Finally, we demonstrate the ISPKF’s capabilities in the real-world context of building accurate environment models from long range stereo data.

3.2 Derivation

In this section we derive the measurement update equations for the Iterated Sigma Point Kalman Filter. For completeness we start from first principles with the joint probability
density function describing our problem, work through Bayes’s rule, Newton’s method, Gauss-Newton iterated non-linear least squares, the Iterated Extended Kalman Filter, linearized error propagation, statistically linearized error propagation, and finally arrive at the ISPKF.

We are interested in estimating the value of an unknown parameter vector $\mathbf{x}$ from noisy measurements of related quantities $\mathbf{z}$. We consider the estimate vector $\hat{\mathbf{x}}$ and observation vector $\mathbf{z}$ to be independent realizations of multivariate normal distributions

$$
\mathbf{z} \sim N(h(\mathbf{x}), \mathbf{R})
$$

$$
\hat{\mathbf{x}} \sim N(\mathbf{x}, \mathbf{P})
$$

where $h: \mathbb{R}^n \rightarrow \mathbb{R}^m$ is a non-linear measurement function relating the state to the measurements; and $\mathbf{R}$ and $\mathbf{P}$ are the measurement and state error covariance matrices, respectively.

From the Bayesian perspective we have $P(\mathbf{x}|\mathbf{z}) = \eta P(\mathbf{z}|\mathbf{x})P(\mathbf{x})$ where $\eta = P(\mathbf{z})^{-1}$ amounts to a normalizing factor, $P(\mathbf{x})$ is the prior, and $P(\mathbf{z}|\mathbf{x})$ is the likelihood. Note that in this chapter we ignore any system dynamics in the process model that might be acting on the prior and focus on the distinct problem of the measurement update. The task of maximum a posterior (MAP) estimation is to find the $\mathbf{x}$ that maximizes the scalar quantity $P(\mathbf{z}|\mathbf{x})P(\mathbf{x})$. Recall that the distributions on $\mathbf{x}$ and $\mathbf{z}$ are normal,
\[ P(z|x) = \frac{1}{\sqrt{(2\pi)^n|R|}} \exp \left( -\frac{1}{2} (z - h(x))^T R^{-1} (z - h(x)) \right) \]

\[ P(x) = \frac{1}{\sqrt{(2\pi)^n|P|}} \exp \left( -\frac{1}{2} (\hat{x} - x)^T P^{-1} (\hat{x} - x) \right) \]

where \(| \cdot |\) is the determinant. The solution that maximizes \(P(z|x)P(x)\) is equivalent to minimizing its negative log, which reduces to the quadratic,

\[ \ell(x) = \frac{1}{2} \left[ (z - h(x))^T R^{-1} (z - h(x)) \right. \]

\[ + (\hat{x} - x)^T P^{-1} (\hat{x} - x) \left. \right] + k \]

where \(k\) is a constant which may be dropped. An algebraically equivalent way to look at MAP is to consider the prior estimate as a pseudo-observation and then to write a new observation vector and function,

\[ Z = \begin{bmatrix} z \\ \hat{x} \end{bmatrix}, \quad g(x) = \begin{bmatrix} h(x) \\ x \end{bmatrix}, \quad C = \begin{bmatrix} R & 0 \\ 0 & P \end{bmatrix} \]

which gives (ignoring \(k\))

\[ \ell(x) \approx \frac{1}{2} \left[ (Z - g(x))^T C^{-1} (Z - g(x)) \right]. \quad (3.1) \]
If we let $S^T S = C^{-1}$ and

$$f(x) = S(Z - g(x)) \tag{3.2}$$

then (3.1) is clearly a non-linear least squares problem of the form

$$\ell(x) = \frac{1}{2} ||f(x)||^2.$$

Newton’s solution to such optimization problems is the iterative sequence

$$x_{i+1} = x_i - (\nabla^2 \ell(x_i))^{-1} \nabla \ell(x_i). \tag{3.3}$$

For small residual problems a useful approximation to (3.3) is the Gauss-Newton method, which approximates the Hessian $\nabla^2 \ell(x_i)$ by $f'(x_i)^T f'(x_i)$. Thus, since the gradient of (3.2) is $\nabla \ell(x_i) = f'(x_i)^T f(x_i)$, the Gauss-Newton method defines the sequence of iterates [20]

$$x_{i+1} = x_i - (f'(x_i)^T f'(x_i))^{-1} f'(x_i)^T f(x_i) \tag{3.4}$$

where $f'(x_i)$ is the Jacobian of (3.2). Noting that $f'(x_i) = -SG_i$ where $G_i$ is the Jacobian of $g(x_i)$, (3.4) becomes

$$x_{i+1} = (G_i^T C^{-1} G_i)^{-1} G_i^T C^{-1} (Z - g(x_i) + G_i x_i) \tag{3.5}$$
which (when \( g(x) \) is first approximated by its first order expansion) is also the \textit{normal equation} solution to (3.1). This sequence is iterated to convergence. After convergence the updated covariance \( \hat{P}_{k+1} \) is approximated using

\[
\hat{P}_{k+1} = (G_i^T C^{-1} G_i)^{-1} = (H_i^T R^{-1} H_i + P^{-1})^{-1}
\]  

where \( H_i \) is the Jacobian of \( h(x_i) \) (for notational simplicity the \( k \) is dropped on the current covariance \( P \)). At this point we have derived the Gauss-Newton measurement update and made explicit the connection to Newton’s method. In contrast to one shot linearization methods like the Extended Kalman Filter or the Unscented Kalman Filter, the Gauss-Newton method is locally convergent to the MAP estimate for near zero-residual problems [20]. In fact the discrete EKF is algebraically equivalent to a \textit{single} iteration of the Gauss-Newton method [5]. The Gauss-Newton method is simply on the other side of the \textit{matrix inversion lemma}. To see this we will need two forms of the matrix inversion lemma: 1) 

\[
(H^T R^{-1} H + P^{-1})^{-1} H^T R^{-1} = P H^T (H P H^T + R)^{-1}
\]

and 2) 

\[
(H^T R^{-1} H + P^{-1})^{-1} = P - P H^T (H P H^T + R)^{-1} H P
\]
Expanding (3.5) and making use of the matrix inversion lemma we obtain, $x_{i+1}$

$$x_{i+1} = x_i + (H_i^T R^{-1} H_i + P^{-1})^{-1} \left[ H_i^T R^{-1}(z - h(x_i)) + P^{-1}(\hat{x}_k - x_i) \right]$$

$$= \hat{x}_k + PH_i^T (H_i PH_i^T + R)^{-1}(z - h(x_i)) - PH_i^T (H_i PH_i^T + R)^{-1} H_i (\hat{x}_k - x_i) \quad (3.7)$$

$$= \hat{x}_k + PH_i^T (H_i PH_i^T + R)^{-1}(z - h(x_i) - H_i (\hat{x}_k - x_i)) \quad (3.8)$$

$$= \hat{x}_k + K(z - h(x_i) - H_i (\hat{x}_k - x_i)) \quad (3.9)$$

which is the Iterated Extended Kalman Filter measurement update; $K = PH_i^T (H_i PH_i^T + R)^{-1}$ is the Kalman gain. Once the sequence over $i$ is iterated to convergence the next time estimate $\hat{x}_{k+1}$ is set to $x_i$. Similarly, after convergence, the covariance update (3.6) can be manipulated into the familiar EKF form $\hat{P}_{k+1} = (I - KH_i)P$. On the first iteration $x_i = \hat{x}, i = 1$, and (3.9) reduces to the Extended Kalman Filter. The point here is that iteration is not just an extension to the EKF; on the contrary, for near zero-residual problems, the EKF is a sub-optimal approximation of the provably convergent Gauss-Newton method [20].
3.2.1 Linear Error Propagation

Notice that in (3.9) there are numerous linearized error propagation terms like $PH^T$ and $HPH^T$. If we have a linear function $z = Hx$ like the linearized equations above, then using the linearity of expectation we can propagate covariance in $x$ through to $z$

$$
cov(z) = E[(\hat{z} - z)(\hat{z} - z)^T] = E[(H\hat{x} - Hx)(H\hat{x} - Hx)^T] = H\text{cov}(x)H^T = HPH^T
$$

and the cross-covariance is

$$
cov(x, z) = E[(\hat{x} - x)(\hat{z} - z)^T] = E[(\hat{x} - x)(H\hat{x} - Hx)^T] = \text{cov}(x)H^T = PH^T
$$

Note also that $\text{cov}(z, x) = \text{cov}(x, z)^T = H\text{cov}(x) = HP$. With these in mind the IEKF update in (3.7) and (3.9) is

$$
x_{i+1} = \hat{x} + \text{cov}(x, z)(\text{cov}(z) + R)^{-1} \left( z - h(x_i) - \text{cov}(x, z)^TP^{-1}(\hat{x} - x_i) \right) \quad (3.10)
$$
This equation is based on linearization of the observation function $h$, and linearized error propagation of the random variables $\hat{x}$ and $z$.

A more accurate method of error propagation could be employed, such as statistically linearized error propagation [31]. There are various methods that use this technique, such as the UKF, SPKF, CDKF and LRKF [107, 106, 44, 49, 53]. However, these methods do not iterate the sequence to convergence, and typically end their analysis with comparison to the Extended Kalman Filter. This practice misses the direct and intuitive relationship between the Iterated Extended Kalman Filter and Newton’s method. One would never iterate the Newton method just once. Hence, it is important to extend the statistical linearized filters so that iteration is possible.

### 3.2.2 Statistically Linearized Error Propagation

Statistical linear error propagation is generally more accurate than error propagation via first order Taylor series expansion [31, 47, 106]. The idea behind statistical linear error propagation is simple: select from the distribution on $x$ a set of regression points $\mathcal{X}$, such that they maintain certain properties of the input distribution (such as the mean and covariance). These points are then mapped through $h$

$$Z_i = h(\mathcal{X}_i)$$

creating a set of transformed regression points, from which the mean and covariance are computed. Recent research has led to a number of filters that employ statistical
linearization [47, 107, 44], which can all be understood as examples of the so-called Sigma Point approach [106]. In this method the Sigma Points (regression points) are selected to lie on the principle component axes of the input covariance, plus one extra point for the mean of the distribution

\[ \mathcal{X}_0 = \hat{x} \]
\[ \mathcal{X}_i = \hat{x} + (\sqrt{(L + \lambda)P})_i \quad i = 1, \cdots, L \]
\[ \mathcal{X}_i = \hat{x} - (\sqrt{(L + \lambda)P})_i \quad i = L + 1, \cdots, 2L \]

\[ w_i^{(m)} = \frac{\lambda}{L + \lambda} \]
\[ w_i^{(c)} = \frac{\lambda}{L + \lambda} + (1 - \alpha^2 + \beta) \]
\[ w_i^{(m)} = w_i^{(c)} = \frac{\lambda}{L + \lambda} \quad i = 1, \cdots, 2L \]

where \( \lambda = \alpha^2(L + \kappa) - L \), \( L \) is the dimension of the state space, and the parameters \( \alpha, \beta, \) and \( \kappa \) are tuning parameters\(^1\).

The mean and covariance are computed from the Sigma points \( \mathcal{X} \) and transformed Sigma points \( \mathcal{Z} \) using

\[ \tilde{\mathcal{Z}} = \sum_{i=0}^{2L} w_i^{(m)} \mathcal{Z}_i \]

\(^1\)For more on setting these see [106]. In the examples here \( \alpha = 10^{-4}, \beta = 2, \) and \( \kappa = 0 \), which is appropriate for Gaussian distributions [107]. The parameter \( \alpha \) controls the distance of the Sigma Points away from \( \hat{x} \) along the principle components of \( P \); \( \beta \) can be used to include information about higher order moments; and \( \kappa \geq 0 \) helps guarantee the positive-semi-definiteness of the covariance matrix.
\[
\text{cov}(Z) = \sum_{i=0}^{2L} w_i^{(c)}(Z_i - \bar{Z})(Z_i - \bar{Z})^T
\]

\[
\text{cov}(\mathcal{X}, Z) = \sum_{i=0}^{2L} w_i^{(c)}(\mathcal{X}_i - \hat{x})(Z_i - \bar{Z})^T
\] (3.12)

Selecting the regression points in this way is the key insight behind the Sigma Point approach. After pushing the Sigma points through \( h \), we compute the first and second order statistics of the transformed points.

### 3.2.3 The Iterated Sigma Point Kalman Filter

From (3.7) we see that in (3.10) we have expressed the Iterated Extended Kalman Filter equations so that they do not depend on the sensor model Jacobian, but are instead expressed in terms of propagated error terms. This allows replacing the linear error propagation terms with statistical error propagation terms. By replacing the linearized error propagation terms in (3.10) with statistically linearized error propagation terms (3.12) we get

\[
x_{i+1} = \hat{x} + \text{cov}(\mathcal{X}, Z)(\text{cov}(Z) + R)^{-1}
\]

\[
\left( z - h(x_i) - \text{cov}(\mathcal{X}, Z)^T P^{-1}(\hat{x}_k - x_i) \right)
\] (3.13)
\[ P_{k+1} = P_k - \text{cov}(\mathcal{X}, Z)(\text{cov}(Z) + R)^{-1}\text{cov}(\mathcal{X}, Z)^T \] (3.14)

The Iterated Sigma Point Kalman Filter measurement update is defined by (3.11), (3.12), (3.13) and (3.14), which completes the derivation. This filter utilizes the benefit of statistical linearization and iteration.

### 3.3 Developed Example: Long Range Stereo

We have applied the ISPKF to the problem of filtering a sequence of feature measurements from a Stereo Vision system. Stereo is an interesting problem because it exposes the inability of linear error propagation to faithfully transform a probability density through a non-linear function. This is apparent in Fig. 3.1 which shows the shape of the analytically propagated disparity probability density function – it is not Gaussian.

Consider a standard camera model that maps a 3D point \( \mathbf{x} = [x, y, z]^T \) to pixel locations in an axis aligned stereo rig. Standard perspective projection gives

\[
\begin{bmatrix}
  u_l \\
  v_l \\
  u_r \\
  v_r
\end{bmatrix} = h(\mathbf{x}) =
\begin{bmatrix}
  f x/z \\
  f y/z \\
  f(b - x)/z \\
  f y/z
\end{bmatrix}
\]
Figure 3.1: Probability density function for disparity derived from a 3D landmark at 4m with a range standard deviation of 1.5m. The p.d.f. is clearly non-Gaussian, non-symmetric, and has a heavy tail. The difference between the p.d.f. mean and the true mean can lead to bias and apparent divergence when filtering stereo measurements.
where \([u_l, v_l, u_r, v_r]^T\) is the measurement of the pixels in the left and right images, \(b\) is the baseline, and \(f\) is the focal length\(^2\). For rectified imagery, the disparity, \(d = u_l - u_r\), is inversely related to range, \(d = s(z) = bf/z\), and clearly range is inversely related to disparity \(z = s(z)^{-1} = bf/d\). If both the left pixel \([u_l, v_l]^T\) and right pixel \([u_r, v_r]^T\) are realizations of normal distributions, then the measured disparity will also be normally distributed. Analogously, notice that in (3.9) that the covariance of the measurement distribution, derived from the state covariance \(P\), is represented by the linear approximation \(H_i PH_i^T\). To understand the impact of this approximated measurement distribution better we use the range p.d.f., \(f_z(z)\), which we have modeled as Gaussian, to derive analytically the disparity p.d.f., \(f_d(d)\). This gives the derived distribution [18],

\[
f_d(d) = f_z(s(d)^{-1}) \left| \frac{\partial s(d)^{-1}}{\partial d} \right|
\]

which expands to

\[
f_d(d) = \frac{fb}{\sqrt{2\pi\sigma_z^2}} e^{\exp} \left( \frac{-(fb/d - \mu_z)^2}{2\sigma_z^2} \right) \tag{3.15}
\]

Where \(\sigma_z\) is the range standard deviation and \(\mu_z\) is the range mean. This non-Gaussian p.d.f. is shown in Fig. 3.1 and on the y-axis of Fig. 3.2. Clearly, this predicted measurement disparity p.d.f. is non-symmetric and has a long tail (which causes bias in stereo). Using statistical linearization to compute these distributions is generally more accurate that linearized error propagation when the function \(h\) is non-linear [31, 106].

\(^2\)Throughout this portion of the chapter we use linear camera models with a resolution of 512×384 pixels, a horizontal FOV of 68.12° and vertical FOV of 51.37°. Unless otherwise stated, we use uncorrelated image measurement noise with a standard deviation of 0.25 pixels.
This is apparent in Fig. 3.2 which shows that statistical error propagation captures the mean of the disparity probability density function more accurately than linear error propagation.

3.3.1 Comparison of methods

In this section we derive and compare a number of methods that estimate the 3D position of a distant feature given stereo measurements. Consider some general stereo triangulation function \( r : \mathbb{R}^4 \rightarrow \mathbb{R}^3 \)

\[
r(z) = x = \begin{bmatrix} x \\ y \\ z \end{bmatrix}
\]  

(3.16)

All the filters we compare are initialized with \( r(z_1) \), the triangulated stereo value from the first measurement. The initial state error covariance matrix \( P \) is found via error propagation of image errors

\[
P = \frac{\partial r}{\partial z} R \frac{\partial r}{\partial z}^T, \quad R = \begin{bmatrix} \Sigma_l & 0 \\ 0 & \Sigma_r \end{bmatrix}
\]  

(3.17)

where \( \Sigma_l \) and \( \Sigma_r \) are the left and right \( 2 \times 2 \) measurement error covariance matrices, respectively.
Figure 3.2: Error propagation from the state space to the measurement space for stereo ranging. Propagation of this type takes place when we compute $\bar{Z}$ and $\text{cov}(Z)$ in (3.12) or $\text{cov}(z)$ via $\text{HPH}^T$. Statistical error propagation captures the mean of the analytical p.d.f. more accurately, which is biased long due to the heavy tail.
We compare the following methods: the optimal Batch non-linear least squares solution (3.3.1.1), an Iterated EKF (3.3.1.3), a Gauss-Newton Smoother (3.3.1.2), an EKF with 3D measurements (3.3.1.4), the UKF (3.3.1.5), and finally the Iterated Sigma Point Kalman Filter (3.3.1.6).

It is well known that stereo triangulation suffers from a statistical bias, and recently solutions for bias removal have been proposed [91, 86] However, in the context of temporal filtering bias removal is not enough – we must also make sure the underlying estimation machinery is robust to the non-linear nature of the problem at hand, or else the estimator may exhibit apparent divergence, as we will see with the “3D EKF” described below.

### 3.3.1.1 Batch Least Squares

If we have measurements $z_j$ for $j = 1, 2, ..., k$ up to time $k$ then we can lump all the measurements together

$$z = \begin{bmatrix} z_1 \\ z_2 \\ \vdots \\ z_k \end{bmatrix}, \quad h = \begin{bmatrix} h_1(x) \\ h_2(x) \\ \vdots \\ h_k(x) \end{bmatrix},$$
Figure 3.3: Filter convergence for a sequence of 20 measurements of a feature at 25m averaged over 1000 trials. Image noise is modeled as uncorrelated standard deviation of 0.25 pixels.
\[
R = \begin{bmatrix}
R_1 & 0 & \ldots & 0 \\
\vdots & R_2 & \ddots & \vdots \\
\vdots & \ddots & 0 & \vdots \\
0 & \ldots & 0 & R_k
\end{bmatrix}
\]

Where the \(j^{th}\) observation function \(h_j\) is

\[
h_j(x) = \begin{bmatrix}
h_{left}(x) \\
h_{right}(x)
\end{bmatrix}
\]  

(3.18)

and \(h_{left} : \mathbb{R}^3 \to \mathbb{R}^2\) and \(h_{right} : \mathbb{R}^3 \to \mathbb{R}^2\) are the left and right camera projection functions. Depending on the camera models in use, \(h_{left}\) and \(h_{right}\) can be formulated in a variety of ways [105, 109]. We only require that these functions and their first derivatives are available, and otherwise leave them unspecified.

From this we can write a large batch least squares problem

\[
\ell_z(x) \approx \frac{1}{2}(z - h(x))R^{-1}(z - h(x))
\]  

(3.19)

the solution to which is the maximum likelihood estimate (MLE) given all measurements at all times. The Gauss-Newton Iteration for this problem is

\[
x_{i+1} = (H_i^TR^{-1}H_i)^{-1}H_i^TR^{-1}(z - h(x_i) + H_ix_i).
\]
Figure 3.4: Squared Filter Error vs. Measurement number for a sequence of 20 measurements of a feature at 25m averaged over 1000 trials. The improvement of the UKF over the Gauss-Newton method can be attributed to statistical error propagation. The improvement of the ISPKF over the UKF can be attributed to Iteration. Notice that filters incorporating prior information modeled as Gaussian (all but the batch filter) suffer from initialization problems – either over confidence or under confidence. This is due to modeling range uncertainty as Gaussian, which is incorrect.
As usual after convergence the covariance is approximated as

\[ \hat{P}_{k+1} = (H_i^T R^{-1} H_i)^{-1}. \]

Notice that for identical and independently distributed Gaussian measurements this is exactly the inverse Fisher Information matrix, and hence the batch estimator matches the Cramer-Rao lower bound. In the maximum likelihood sense this is the best unbiased estimate possible. In Fig. 3.3 and Fig. 3.4 we expect this estimator to perform the best.

### 3.3.1.2 Gauss-Newton Smoother

Including a prior in the batch estimator results in a formulation identical to the Gauss-Newton recursive filter in (3.5) except that the measurement vector \( z \) is much larger (it contains all measurements from all times). This batch/recursive filter can use any number of past measurements, such as a sliding window over the previous \( m \) measurements. The longer the window the closer the results will approximate the MLE batch solution.

A history window of \( m = 1 \) is equivalent to the Gauss-Newton recursive filter (3.5); a window with all measurements is equivalent to the batch method in Sec. (3.3.1.1). Results for smoothing with a window of \( m = 5 \) are shown in Fig. 3.3 and Fig. 3.4. As expected, the Smoother matches the Batch estimator up until the time window starts dropping older measurements at \( m = 5 \).
Figure 3.5: Range probability density function analytically derived through the stereo triangulation equations (inverse of the sensor model). Propagating measurements that are normally distributed results in a non-Gaussian range distribution. Estimators that model the feature state as a 3D Gaussian will suffer from this mis-representation. The filter will likely exhibit either bias or divergence. Both iteration and statistical linearization can help overcome these effects. All the filters except the batch-optimal filter in Fig. 3.3 suffer from this mis-representation so some extent.

### 3.3.1.3 Iterated Extended Kalman Filter

An Iterated Extended Kalman Filter is algebraically equivalent to a Gauss-Newton Smoother with a time window of $m = 1$. The results for this filter are plotted in Fig. 3.3 and Fig. 3.4. Initially the filter exhibits apparent divergence along the lines of the 3D EKF. This
can be attributed to the inaccuracy of modeling range uncertainty as a Gaussian, when the true range distribution is biased long, as shown in Fig. 3.5.

### 3.3.1.4 EKF with 3D measurements

In order to see the deleterious effects linearized error propagation can have on a filter, consider a Kalman Filter with 3D observations, \( z_{3D} \), as output from stereo triangulation. The state estimate and observation are independent realizations of multivariate normal distributions: 

\[
\begin{align*}
\mathbf{z}_{3D} &\sim \mathcal{N}(\mathbf{r}(\mathbf{z}), \mathbf{R}_{3D}) \\
\hat{\mathbf{x}} &\sim \mathcal{N}(\mathbf{x}, \hat{\mathbf{P}})
\end{align*}
\]

where \( \mathbf{R}_{3D} \) and \( \hat{\mathbf{P}} \) are the measurement and state error covariance matrices, respectively. The measurement error covariance matrix \( \mathbf{R}_{3D} \) is found via error propagation of image errors just like (3.17).

Notice that using this formulation introduces an extra linear error propagation into the equations. This has serious implications for filter performance – indeed from Fig. 3.3 we see that the 3D EKF converges to a biased value. Filtering sequences of 3D measurements from stereo leads to biased range estimates when the uncertainty of each 3D measurement is modeled by standard linear error propagation techniques; this emphasizes the fact that linear error propagation does not faithfully represent the transformed distribution.

### 3.3.1.5 Unscented Kalman Filter

The Unscented Kalman Filter is equivalent to the Iterated Sigma Point Filter where only one iteration is carried out during the measurement update. This filter is shown in Fig. 3.3 and Fig. 3.4.
3.3.1.6 Iterated Sigma Point Kalman Filter

Applying the Iterated Sigma Point Filter developed in Sec. 3.3 to the stereo problem is straightforward. Like the other filters the state is initialized via stereo triangulation and error propagation. The iterated measurement update typically converges within 9 iterations. As can be seen in Fig. 3.3 and Fig. 3.4 the ISPKF comes closest to matching the batch solution, followed by the Gauss-Newton Smoother.

3.4 Estimator Efficiency

It is also important to address the efficiency of the ISPKF; that is, how well it approximates a minimal variance estimate of the parameters. The information inequality, \( \text{cov}_x(x) \geq I_z(x)^{-1} \), defines such a bound, which is called Cramer-Rao lower bound [18]. Here the Fisher information matrix \( I_z(x) \) is given by the symmetric matrix whose \( i^{th}, j^{th} \) element is the covariance between first partial derivatives of the log-likelihood function (3.19),

\[
I_z(x)_{i,j} = \text{cov}_x \left( \frac{\partial \ell_z}{\partial x_i}, \frac{\partial \ell_z}{\partial x_j} \right) \tag{3.20}
\]

which, for a multivariate normal distribution, reduces to [95, 112]

\[
I_z(x)_{i,j} = \frac{\partial h^T}{\partial x_i} R^{-1} \frac{\partial h}{\partial x_j}.
\]

For \( n \) independent identically distributed measurements the Fisher information is simply \( nI \). Qualitatively, an estimator that comes close to the CRLB is efficient. Fig. 3.6 shows
Figure 3.6: Cramer Rao Lower Bound for the ISPKF, the Gauss-Newton Smoother, and the optimal Batch filter. As expected the Batch filter tracks the CRLB, and the Gauss-Newton Smoother tracks the CRLB for the first 5 steps. However, after 5 steps the Smoother starts rolling old measurements into a prior, and the ISPKF starts to outperform the Smoother. As a recursive filter the ISPKF is nearly as efficient as the Batch filter.
Figure 3.7: Sample wall image from experiments using the FIDO Mars Rover in Fig. 3.8. The wall is approximately 6m from the robot. Over 20 frames there were 8005 stereo measurements of 899 different SIFT features.

Figure 3.8: FIDO uses rectified camera models with a resolution of $512 \times 384$ pixels, a horizontal FOV of $37.22^\circ$, a vertical FOV of $29.53^\circ$ and a baseline of $\sim20$cm.
comparison between the ISPKF, the Gauss-Newton smoother and the CRLB, and demonstrates that the ISPKF is efficient. Note that this is also an indication of consistency, since the estimator is not over-confident (e.g. it does not dip below the CRLB).

### 3.5 Long range Stereo Experiments

To verify that the ISPKF is indeed converging to a reasonable estimate we have performed the following experiments. We took a sequence of 20 stereo images of a highly textured large flat wall, triangulated and tracked SIFT features on this wall [57]. We then fit a plane to all of the triangulated 3D points from all of the frames using RANSAC with an inlier threshold of 5cm [24]. If the stereo rig is well calibrated then the plane fit should give a reasonable ‘ground truth’ against which to measure error. Fig. 3.8 and 3.10 show sample images from two of the robots used in two different experiments. Fig. 3.11 shows a sample image from a sequence of 733 frames of a large checkerboard pattern measured from approximately 12m. In this case features are extracted with sub-pixel accuracy using a saddle point least squares fit [60]. This experimental setup is meant to ensure that features will follow a normal distribution, and for purposes of clarity we report results for this sequence.

The error plot in Fig. 3.13 for the checker wall sequence shows that filtering with the ISPKF improves the planar-error – e.g. the structure estimate is improved with filtering. The Gauss-Newton and Batch estimators are also plotted. While all the estimators show improvement, it is difficult to distinguish difference in their performance.
Figure 3.9: Sample wall image from experiments using a stereo setup configured analogously to the LAGR robot in Fig. 3.10. The wall is approximately 20m from the robot.

### 3.5.1 Note on Parametric Estimation

Generally, the problem with using simple parametric distributions (like Gaussians) is that error-propagation through non-linear functions can transform the p.d.f. into highly non-Gaussian shapes. For instance, in stereo, the input p.d.f. is quite nearly Gaussian (see for instance the measurement histogram in Fig. 3.12) while the transformed range p.d.f. is not. Since our state space representation assumes the normal distribution, we are likely to face difficulties. The underlying problem is that in state space the distribution we are tracking is not Gaussian - even if our sensor measurements are. There are many solutions, such as non-parametric density estimation [93], or tracking higher order moments to estimate more accurately the a posteriori density [96], to name just two. In general comparison to Monte-Carlo methods should give more insight into the ISPKF’s performance.
3.6 Conclusion

This chapter develops the Iterated Sigma Point Kalman Filter and emphasizes the importance of iteration in solving non-linear least squares problems. We develop the ISPKF directly from first principles, grounding it in non-linear optimization theory and Newton’s Method. This derivation shows that the Iterated Sigma Point Kalman Filter (ISPKF) is more than just an extension to non-iterated statistical linearization methods.

To establish the usefulness of the ISPKF we compare against a number of methods in the context of filtering long range stereo range measurements. Stereo is an interesting problem on which to apply the ISPKF because triangulation is an inherently biased, non-linear problem. We compare the ISPKF to the optimal batch non-linear least squares solution, a Gauss-Newton smoother, the Iterated Extended Kalman Filter, an Extended
Figure 3.11: Sample image of the checker wall sequence. The wall is approximately 12m in front of the robot. For scale, each checker is a US Letter size piece of paper (8.5 in by 11 in). Over 733 frames 105 corner features were tracked from the center portion of the pattern. The lighting and camera shutter speed were deliberately set to induce image noise so that the feature tracker (a sub-pixel saddle point corner fit in this case [60]) would produce high measurement noise. This is apparent in Fig. 3.12.
Figure 3.12: Histogram of 77,049 measurements for real data from the checker wall sequence (for the horizontal pixel dimension). This sequence uses rectified camera models with a resolution of 1024x768 pixels, a horizontal FOV of 54.08°, a vertical FOV of 40.87° and baseline of ~4cm. This small baseline setup was deliberately chosen to challenge the estimators in this chapter with noisy data and small disparities.
Figure 3.13: Results from real data showing error measured against RANSAC ground truth plane for the Batch, ISPKF and Gauss-Newton filter for the checker wall sequence.
Kalman Filter with 3D measurements, and the Unscented Kalman Filter. This comparison shows that the ISPKF out performs all but the full batch solution. For experimental validation we have also applied ISPKF to filtering long range stereo data using three stereo rigs.

This chapter concludes our work on stereo data-fusion from stationary cameras. The estimation machinery developed here leads naturally into Structure from Motion (SFM) and Simultaneous Localization and Mapping (SLAM) problems. Hence, in the next chapter we discuss stereo data-fusion from moving robot platforms.
Chapter 4

The Sliding Window Filter

Extending the work with stationary cameras in Ch. 2 and Ch. 3, in this chapter we address long range stereo data-fusion from mobile platforms. Recall that in Ch. 3 we saw that, while statistically linearized error propagation leads to improved performance, so too does smoothing. With that in mind, the work in this chapter is a natural extension to the Gauss-Newton smoother in Ch. 3. Applying the Unscented Transform is to the filter developed here is possible and is left as future work. Moving forward, we now develop a new Sliding Window Filter (SWF) that is an on-line constant-time solution to the feature-based 6-degree-of-freedom full Batch Least Squares Simultaneous Localization and Mapping (SLAM) problem. We contend that for SLAM to be useful in large environments and over extensive run-times, its computational time complexity must be constant, and its memory requirements should be at most linear. Under this constraint, the “best” algorithm will be the one that comes closest to matching the all-time maximum-likelihood estimate of the full SLAM problem, while also maintaining consistency. We start by formulating SLAM as a Batch Least Squares state estimation
problem, and then show how to modify the Batch estimator into an approximate Sliding Window Batch/Recursive framework that achieves constant time complexity and linear space complexity. We argue that viewing SLAM from the Sliding Window Least Squares perspective is very useful for understanding the structure of the problem. This perspective is general, capable of subsuming a number of common estimation techniques such as Bundle Adjustment and Extended Kalman Filter SLAM. By tuning the sliding window, the algorithm can scale from the exhaustive Batch solutions to fast incremental solutions; if the window encompasses all time, the solution is algebraically equivalent to full SLAM; if only one time step is maintained, the solution is algebraically equivalent to the Extended Kalman Filter SLAM solution.

The Sliding Window Filter enables other interesting properties, like continuous sub-mapping, the ability to revise past data associations, and incremental robust estimation.

We test The performance of the algorithm with sequences of real images. Experiments show that the SWF approaches the performance of the optimal batch estimator, even for small windows on the order of 3-5 frames.

### 4.1 Introduction

For a mobile robot, there are many situations in which accurate high-resolution local *spatial awareness* is a prerequisite for successfully performing a task. For instance, good sensing is undoubtedly useful for tight obstacle avoidance, delicate mobile manipulation,
sensing complicated terrain, subtle motion detection or moving object tracking from a moving platform.

This is a *data-fusion* problem in which a sensor undergoing uncertain, dynamic motion, must combine noisy measurements into a single underlying sensor-relative state estimate. *Data-fusion is the fundamental process of combining information over time in order to reduce uncertainty.* In Robotics this is most commonly tackled under the banner of Simultaneous Localization and Mapping (SLAM); in Computer Vision the problem is usually referred to as Structure from Motion (SFM).

We contend that for SLAM to be useful in large environments and over extensive run-times, its computational time complexity must be *constant*, and its memory requirements should be at most *linear*. Under this constraint, the “best” algorithm will be the one that comes closest to matching the all-time maximum-likelihood estimate of the full SLAM problem, while also maintaining consistency.

Section 4.2.1 formulates SLAM as a Batch Least Squares state estimation problem, and then section 4.3 shows how to modify the full SLAM estimator into an approximate sliding window filter that achieves constant time complexity and linear space complexity. Viewing SLAM from the sliding window least squares perspective is very useful for understanding the structure of the problem; this perspective is also general, encompassing techniques such as Sparse Bundle Adjustment, GraphSLAM, and Extended Kalman Filter SLAM. We conclude with some early results indicating that SWF SLAM comes close to matching the performance of Full SLAM.
4.2 Simultaneous Localization and Mapping as a Statistical Point Estimation Problem

It is useful to approach SLAM from the traditional statistical point estimation perspective because it helps reveal the underlying structure of the problem. This is apparent for a number of reasons. First, because it highlights the fundamental minimization principle at work in least squares, which is sometimes harder to see from the filtering perspective. Second, starting with the underlying probability density functions that describe our problem, it clearly shows the Gaussian probabilistic nature of SLAM - that is, SLAM is simply tracking a normal distribution through a large state space; a state space that changes dimension as we undertake the fundamental probabilistic operations of removing parameters via marginalization, and adding parameters via error propagation and conditioning. A third reason to derive SLAM from statistical point estimation is because it exposes a rich body of theory about the convergence of least squares estimators. Further, starting from least squares one can easily see the connection to many important concepts like Newton’s method, Fisher Information, the Cramer Rao Lower bound, information graphs, graphical models, belief propagation and probabilistic belief networks to name just a few. All these concepts have intuitive derivations starting from traditional statistical point estimation.
4.2.1 Problem Formulation

The parameter vector we wish to estimate is composed of a map of 3D landmark positions $x_{m_i}$, $1 \leq i \leq n$ and a temporal sequence of robot poses $x_{p_j}$, $1 \leq j \leq m$. $x = [x_m^T, x_p^T]^T = [x_{m_1}^T, ..., x_{m_n}^T, x_{p_1}^T, ..., x_{p_m}^T]^T$. Each pose $x_{p_j}$ is a six element column vector containing a 3D position and an Euler angle $x_{p_j} = [x_{p_j}, y_{p_j}, z_{p_j}, r_{p_j}, p_{p_j}, q_{p_j}]^T$. Landmarks are represented by their 3D position, $x_{m_i} = [x_{m_i}, y_{m_i}, z_{m_i}]^T$. The state dimension is thus $|x| = (3n + 6m)$ and grows as the robot path increases and as new landmarks are observed.

The process model, $f_j : \mathbb{R}^6 \rightarrow \mathbb{R}^6$, for a single step describes each pose in terms of the previous pose: $x_{p_{j+1}} = f_j(x_{p_j}, u_{j+1}) + w_{j+1}$, where $u_{j+1}$ is an input command to the robot. The noise vector $w_{j+1}$ is additive and follows a normal distribution $w_{j+1} \sim N(0, Q_{j+1})$, so that $x_{p_{j+1}} \sim N(f_j(x_{p_j}, u_{j+1}), Q_{j+1})$. A simple and useful kinematic process model for $f_j$ is the compound operation, $\oplus$, which is described in [94]. The $6 \times 6$ Jacobian of $f_j$, $F_j = \frac{\partial f_j}{\partial x_{p_j}} | _{x_{p_j}, u_{j+1}}$, which we will need in a moment, is also derived in [94]. Concatenating individual process models together, the p.d.f describing the robot path, $x_p = [x_{p_1}^T, ..., x_{p_m}^T]^T$, is $p(x_p) = \mathcal{N}(\mu_p, Q)$, where

$$\mu_p = f(x) = \begin{bmatrix} x_{\Pi_0} \\ f_1(x_{p_1}, u_2) \\ \vdots \\ f_m(x_{p_{m-1}}, u_m) \end{bmatrix}, \quad Q = \begin{bmatrix} Q_1 & & \\ & \ddots & \\ & & Q_m \end{bmatrix}.$$
The generative sensor model, \( h_{ij} : \mathbb{R}^{\left| x_{m_i} \right| + \left| x_{p_j} \right|} \rightarrow \mathbb{R}^{\left| z_{ij} \right|} \), returns the expected value the sensor will give when the \( i^{th} \) landmark is observed from the \( j^{th} \) pose: 
\[
z_{ij} = h_{ij}(x_{m_i}, x_{p_j}) + v_{ij}.
\]
Here \( \left| z_{ij} \right| \) is the dimension of a single measurement - in this paper the sensor model is for stereo observations, so \( \left| z_{ij} \right| = 4 \). We assume 
\[
v_{ij} \sim N(0, R_{ij})
\]
so that 
\[
z_{ij} \sim N(h_{ij}, R_{ij}),
\]
where \( R_{ij} \) is the observation error covariance matrix. Concatenating all the observations, measurement functions and measurement covariances together into the column vectors 
\[
z = [z_{11}^T, z_{12}^T, ..., z_{nm}^T]^T,
\]
\[
h = [h_{11}^T, h_{12}^T, ..., h_{nm}^T]^T,
\]
and block diagonal matrix \( R \), gives 
\[
z \sim N(h, R),
\]
which defines the measurement likelihood \( p(z|x) \). Treating the process information as observations the we get the likelihood 
\[
p(z, u|x) = \mathcal{N}(\mu_z, \Sigma),
\]
where
\[
\mu_z = \begin{bmatrix} f(x) \\ h(x) \end{bmatrix}, \quad \Sigma = \begin{bmatrix} Q & 0 \\ 0 & R \end{bmatrix}
\]

Suppose we are also given prior information about the first pose and the map, 
\[
p(x_\pi) = \mathcal{N}(\mu_\pi, \Pi^{-1}),
\]
where
\[
x_\pi = \begin{bmatrix} x_m \\ x_{p_1} \end{bmatrix}, \quad \mu_\pi = \begin{bmatrix} \hat{x}_m \\ \hat{x}_{p_1} \end{bmatrix}, \quad \Pi = \begin{bmatrix} \Pi_m & \Pi_{mp} \\ \Pi_{mp}^T & \Pi_p \end{bmatrix}
\]
where \( \Pi_p \) is the \( 6\times6 \) initial pose information matrix, \( \Pi_m \) is \( 3n\times3n \) map prior information matrix, and \( \Pi_{mp} \) is the \( 3n\times6 \) map-to-poses information matrix.
With the above definitions and assumptions we can finally write the posterior probability of the system, \( p(x|z, u) = p(z, u|x)p(x) \). Our goal is to compute the value of \( x \) which maximizes this density - i.e. the maximum a posteriori estimate. To do so it is first convenient to lump the sensor model, process model, and prior information terms together as follows,

\[
g(x) = \begin{bmatrix} g_r(x) \\ g_f(x) \\ g_z(x) \end{bmatrix} = \begin{bmatrix} x - \hat{x} \\ x - f(x) \\ z - h(x) \end{bmatrix}, \quad C^{-1} = \begin{bmatrix} \Pi & 0 & 0 \\ 0 & Q^{-1} & 0 \\ 0 & 0 & R^{-1} \end{bmatrix};
\]

then the negative logarithm of \( p(x|z, u) \) gives the proportional non-linear least squares problem \( \ell(x) = \frac{1}{2}(g(x)^T C^{-1} g(x)) = \frac{1}{2}||r(x)||^2 \) (to see this let \( r = S g(x) \) where \( S^T S = C^{-1} \)).

The Gauss-Newton method for such problems defines the sequence of iterates, \( x_{i+1} = x_i - (r'(x_i))^{-1} r'(x_i) r(x_i) \), which is locally quadratically convergent to the MAP estimate for near zero-residual problems[20]. Noting that \( r'(x_i) = S G_i \), where \( G_i \) is the Jacobian of \( g(x_i) \), we get the system of equations \( G_i^T C^{-1} G_i \delta x_i = -G_i^T C^{-1} g(x_i) \), which is the fundamental least squares form of the SLAM problem. Note that for notational convenience we will often omit the index \( i \).

It is interesting to note here that for many problems the Gauss-Newton method is algebraically identical to the Iterated Extended Kalman Filter. In fact, as we will see later, the Sliding Window Filter over single time step and all landmarks is exactly the IEKF SLAM solution.
Figure 4.1: Basic structure of the Jacobians, $H$, $D$ and $L$. $H = [H_{11}^T, H_{12}^T, ..., H_{nm}^T]^T$, where $H_{ij} = \frac{\partial h_{ij}}{\partial x}$. Also, $H$ has two components: $H_p$ is the Jacobian of $h$ with respect to the pose parameters, $x_p$, and $H_m$ is the Jacobian of $h$ with respect to the map parameters, $x_m$. Thus, $H_{p,ij} = \frac{\partial h_{ij}}{\partial x_p}$, and $H_{m,ij} = \frac{\partial h_{ij}}{\partial x_m}$.

Figure 4.2: The sparse structure of least squares SLAM system matrix is due to contributions from three components: the measurement information matrix $H^T R^{-1} H$, the process information matrix $D^T Q^{-1} D$, and the prior information matrix $L^T L L$. The measurement information matrix has three distinct components, $U = H_p^T R^{-1} H_p$, $W = H_m^T R^{-1} H_p$, and $V = H_m^T R^{-1} H_m$.

As $\delta x_i \to 0$ the term $(G_i^T C^{-1} G_i)$ converges to the Hessian of the likelihood function (this is the approximation the Gauss-Newton method makes over the full second order Newton method[20]). It turns out that for maximum-likelihood estimation with normal distributions (or MAP estimation when prior information is included like in our formulation) the term $(G_i^T C^{-1} G_i)$ is also the Fisher Information matrix, and it’s inverse approximates the system covariance.
Expanding the Jacobian, \( G = \left[ \frac{\partial g}{\partial x}, \frac{\partial f}{\partial x}, \frac{\partial z}{\partial x} \right]^T \), we see that the system matrix, \( G^T C^{-1} G = L^T \Pi L + D^T Q^{-1} D + H^T R^{-1} H \), has a sparse structure due to the form of \( D, L \) and especially \( H \). Likewise, the RHS vector, \(-G^T C^{-1} g(x)\), expands to \( L^T \Pi (\dot{x} - x) + D^T Q^{-1} (f(x) - x) + H^T R^{-1} (z - h(x)) \). The structure of these Jacobians are shown in Fig. 4.1. Also, the four primary blocks of the Hessian (system) matrix, the “process block”, \( \Lambda_p \), the “map block”, \( \Lambda_m \), and the “map-to-poses block”, \( \Lambda_{mp} \), are clearly shown in Fig. 4.2. Note in particular the tri-diagonal structure in the process information block, \( E = \)

\[
\begin{bmatrix}
F_1 Q^{-1}_{21} F^T_1 & -F^T_2 Q^{-1}_{12} & 0 & \cdots & 0 \\
-Q^T_2 F_1 & Q^{-1}_{21} + F_2 Q^{-1}_{12} F^T_1 & \cdots & \vdots & \vdots \\
0 & \ddots & \ddots & \ddots & 0 \\
\vdots & \ddots & \ddots & \ddots & Q^{-1}_{m-1} + F_{m-1} Q^{-1}_{m-2} F^T_{m-1} & -F^T_{m-1} Q^{-1} \\
0 & \cdots & 0 & \ddots & \ddots \end{bmatrix}
\]

which encodes the conditional dependence between temporally adjacent poses. Notice also that the sensor information matrix terms \( U, V \) and \( W \) can be broken apart into individual contributions from each pose

\[
U = U_1 + U_2 + \ldots + U_m, \quad \text{where } U_j = \sum_{k} H^T_{p_{ij}} R_{ij}^{-1} H_{p_{ij}}
\]

\[
V = V_1 + V_2 + \ldots + V_m, \quad \text{where } V_j = \sum_{k} H^T_{m_{ij}} R_{ij}^{-1} H_{m_{ij}}
\]

\[
W = W_1 + W_2 + \ldots + W_m, \quad \text{where } W_j = \sum_{k} H^T_{m_{ij}} R_{ij}^{-1} H_{p_{ij}}
\]

The task is to solve \( G_t^T C^{-1} G_t \delta x_i = -G_t^T C^{-1} g(x_i) \) which can also be expressed as the \( 2 \times 2 \) system of equations.
Figure 4.3: The pattern of the system matrix in the SLAM problem depends on information contributed (or not) from the dynamical process model and any prior information. With no process model information and no prior information the problem is equivalent to Photogrammetric Bundle Adjustment (the structure of this problem is visible on the right). Including a process model makes the lower right \( m \times m \) “process block” tridiagonal. Including a prior can potentially cause complete fill-in of the upper left \( n \times n \) “map-block”.

\[
\begin{bmatrix}
\Lambda_m & \Lambda_{mp} \\
\Lambda_{mp} & \Lambda_p \\
\end{bmatrix}
\begin{bmatrix}
\delta x_m \\
\delta x_p \\
\end{bmatrix}
= 
\begin{bmatrix}
g_m \\
g_p \\
\end{bmatrix}
\]

where \( g_p \) and \( g_m \) are the RHS vector corresponding to the robot path and map, respectively. Taking advantage of this sparse structure, the system of equations is typically solved by forward-then-backward substitution, either of the path-onto-the-map or of the map-onto-the-path [104, 92].

Depending on the process noise and the prior, the system matrix \( G^T C^{-1} G \) can take on different sparsity patterns that affect the complexity of finding a solution. The possible sparsity patterns are shown in figure 4.3. For instance, an infinite process noise covariance would mean the motion model does not contribute information to the system
\((Q^{-1} = 0 \implies E = 0)\), which would reduce the process-block of the system matrix to block diagonal, which is \(O(m + n^3)\) to solve. Similarly, without prior information \((\Pi = 0)\) the map-block is also block diagonal, which is \(O(m^3 + n)\) to solve. Without information from the motion model and without prior information the problem is equivalent to the Bundle Adjustment problem in Photogrammetry, which can be solved in either \(O(m^3 + n)\) or \(O(m + n^3)\)[10]. It is interesting to note that in this form (no motion model, no prior), the first optimal solution to the SLAM problem using cameras appears to have been developed by Brown circa 1958[9]. Brown was also the originator of what has come to be known as the Tsai camera model[105].

\subsection{4.3 The Sliding Window Filter}

For SLAM to operate usefully over long periods of time, perhaps over the course of a robot's life time, its computational complexity must be \(O(1)\), and hence the parameter vector cannot grow indefinitely. The simplest way to bound computational complexity is to reduce the size of the state vector by, say, removing the oldest pose parameters or distant landmark parameters. However, if we directly remove parameters from the system equation we might lose information about how the parameters interact. The correct way to remove parameters from a multi-dimensional normal distribution is to marginalize them out. Marginalizing out the parameters we wish to remove turns out to be equivalent to applying the Schur complement to the least squares equations[46, 92]. For example, given the system
\[
\begin{bmatrix}
\Lambda_a & \Lambda_b \\
\Lambda_b^T & \Lambda_c
\end{bmatrix}
\begin{bmatrix}
x_a \\
x_b
\end{bmatrix}
=
\begin{bmatrix}
g_a \\
g_b
\end{bmatrix},
\]

reducing the parameters \(x_a\) onto the parameters \(x_b\) gives
\[
\begin{bmatrix}
\Lambda_a & \Lambda_b \\
0 & \Lambda_c - \Lambda_b^T \Lambda_a^{-1} \Lambda_b
\end{bmatrix}
\begin{bmatrix}
x_a \\
x_b
\end{bmatrix}
=
\begin{bmatrix}
g_a \\
g_b - \Lambda_b^T \Lambda_a^{-1} g_a
\end{bmatrix},
\]

where the term \(\Lambda_b^T \Lambda_a^{-1} \Lambda_b\) is called the Schur complement of \(\Lambda_a\) in \(\Lambda_b\). After reducing, the smaller system \(\Lambda_c - \Lambda_b^T \Lambda_a^{-1} \Lambda_b = g_b - \Lambda_b^T \Lambda_a^{-1} g_a\) can be solved for updates to \(x_b\).

### 4.3.1 Overview

We now give a brief synopsis of the SWF algorithm. First, after completing \(m-1\) steps, the command \(u_m\) is used to drive the system forward via the process model, \(x_{p m} = f_{m-1}(x_{p m-1}, u_m)\), which adds six new pose parameters to \(x_p\). Recall that in the Gauss-Newton method the covariance matrix is approximated by the inverse of the Hessian matrix[5]. Thus, after applying the process model but before incorporating any new measurements, we can use the Gauss-Newton method to compute an updated information matrix, which is simply the Hessian associated with the MLE solution. This operation only affects the process-block of the information matrix and can be computed in constant time.
Next, if there are now more than \( n \) poses active (for a \( n \)-window SWF), then the we marginalize out the oldest pose parameters using the Schur complement. If \( n = 1 \) then this step is algebraically equivalent to the EKF SLAM timestep, and there is only ever a single active pose. After marginalizing out the oldest pose we may optionally marginalize out invisible landmarks, as detailed in the next section. Note that marginalizing affects the RHS of the system equations - specifically the first 6 parameters of \( g_p \) and potentially all of \( g_m \).

Next, any new landmarks are added to \( x_m \) (currently initial values are computed via stereo), and \( \Lambda \) is extended (with zeros) appropriately.

Finally, the sparse normal equations are solved for an update to \( x \) using all the observations from all the active poses. The interesting part the SWF depends on when and if poses and landmarks are downdated (marginalized out), which we discuss in a moment.

### 4.3.2 The Effects of Downdating

Marginalizing out parameters induces conditional dependencies between the parameters that the removed-parameters were conditionally dependent on. Hence, downdating a set of pose parameters will induce conditional dependencies between all the landmarks that were visible from that pose. This is depicted graphically in Fig. ?? for a system that starts \textit{without} any prior information. From this we see that it is important to consider how downdating affects the system of equations. It is well known that the information matrix can be interpreted as an adjacency matrix of an undirected graph where non-zero
off-diagonal entries encode conditional dependencies between parameters. The $i^{th}$, $j^{th}$ $3 \times 6$-block of the $\Lambda_{mp}$ matrix encodes map-to-poses conditional dependencies and is non-zero only if the $i^{th}$ map landmark was visible from the $j^{th}$ pose.

Studying this structure we see that downdating the oldest pose causes fill-in in three places: 1) between any landmarks that were visible from the downdated pose, 2) between the parameters of the next-oldest-pose (the pose one time step after the pose being down-dated), and 3) between the next-oldest-pose and all landmarks seen by the downdated pose. Notice that only $\Pi$ experiences additional fill-in.

This is important because it means that the lower right block of the full SLAM problem is still block tri-diagonal, and the $\Lambda_{mp}$ block is only changed along the first 6 columns - exactly where it overlaps with $\Pi_{mp}$. Hence, when solving we can still take advantage of any sparsity patterns that may exist in $\Lambda_{mp}$, just like as in Bundle Adjustment (see the Appendix in[37] for more on this exploit). Note that the patterns in $\Lambda_{mp}$ are problem specific, depending on how the robot went about observing the environment. For instance, a robot sentry that always sees the same landmarks will have a completely dense $\Lambda_{mp}$ block, while for a robot exploring and sensing new landmarks, it will have a sparse banded pattern. This fact can hamper methods that rely on predictability in $\Lambda_{mp}$ in order to achieve computational efficiency[19].

When landmarks are observed from a pose it adds pose-to-landmarks conditional dependence information to the system matrix. This information encodes a soft spatial
rigidity between the parameters. Downdating a pose preserves this soft spatial rigidity by transferring its structure into a map of conditionally dependent landmarks.

By downdating poses we have succeeded in removing the $O(m^3)$ cost of carrying the complete robot path in the state estimate. Interestingly, this process is in some sense the opposite of sparsification in Sparse Extended Information Filters[98] and Exactly Sparse Delayed State Filters[98, 22] - one might call it map “densification”.

The next thing is to bound the growth of the map by a constant, which we will do by downdating landmarks that are no longer visible from any pose currently in the state vector. It is crucial that the landmark is no longer visible because except for the oldest map-to-pose terms, all the cross-information terms in $\Lambda_{mp}$ will be zero, which means that the Schur complement onto the remaining system will once again only affect the prior, $\Pi$.

With a sensor that has a limited range and under the reasonable condition that the spatial density of landmarks in the world is roughly uniform, then the number of map-to-pose information terms is constant. Downdating poses at a fixed rate and landmarks when they become “invisible” results in a bounded size state vector and hence a constant time complexity SLAM estimation algorithm. Unlike other techniques this is true regardless of how the robot explores the environment[100].

The key points here are that 1) downdating preserves information by transforming the way in which the system probability distribution is represented; 2) downdating both pose and “invisible” landmark parameters only ever affects the system prior, and not
the general sparsity pattern of the system equations - the prior term $\Pi$ “catches” all the information we marginalize out.

Note that just by choosing when to downdate poses and landmarks the Least Squares SLAM algorithm can scale from the full Batch solution, to the Extended Kalman Filter solution, to the incremental $O(1)$ SWF solution. That these algorithms are subsumed within one framework testifies to the generality of the simple least squares approach.

It is interesting to note what happens if we simply delete parameters from the estimator instead of marginalizing them out. For a sliding window of size $n$, the error converges like $1/n$ just as we would expect the batch estimator to do. However, after $n$ steps, the error stops converging as we delete information from the back of the filter. With such deleting and a sliding window of $n = 1$ it is interesting to note that we end up with a solution that is nearly identical to previous forms of Visual Odometry\cite{63, 80, 81} (though we should expect slightly better results since the state estimate of the landmarks is being improved with each observation, that is, there is some data fusion, even with “deleting”). The graph in Fig. 4.5 shows the MSE performance for this type of Visual Odometry compared to the batch solution, as well as the SWF solution. The space of solutions spanned by the SWF is depicted graphically in Fig. 4.3.2.
Figure 4.4: Space of solutions spanned by the Sliding Window Filter. The interesting point is that the region of constant time algorithms exhibit the potential to match the more expensive offline algorithms, though of course not at loop closure.

### 4.4 Optimality and Efficiency

If (3.4) converges to the global minimum of $\ell(x)$ then the resultant $x$ is the minimal variance $MAP$ estimate [20]. This is the “best” or “optimal” estimate possible in the non-linear least squares sense with normally distributed measurements. Unlike the Extended Kalman Filter, the Gauss-Newton method has a known convergence theory, which, for near zero-residual problems like SLAM, is locally q-quadratically convergent[20]. The
Figure 4.5: Sliding Window Filter comes close to the Full SLAM solution. Sliding Window of 2 is close to optimal $1/n$ full batch curve. Each curve is a trial for different size time window, averaged over 50 trials, with 0.1 pixel std.dev measurement noise, 1.0m std.dev process noise. Because VO does not combine information over time, it does not reduce uncertainty as time passes.

Signature of an optimal estimator is a $1/n$ reduction in mean squared error of the state estimate vs. ground truth, where here $n$ is the number of time steps.

Another important factor to address is estimator efficiency, that is, how well the estimator approximates a minimal variance estimate of the parameters. The information inequality, $\text{cov}_x(\mathbf{x}) \geq \mathcal{I}(\mathbf{x})^{-1}$, defines the minimal variance bound, which is called Cramer-Rao lower bound[18]. Here the matrix $\mathcal{I}(\mathbf{x})$ is the Fisher information matrix.
defined by the symmetric matrix whose $i^{th}$, $j^{th}$ element is the covariance between first partial derivatives of the log-likelihood function (2.2)

$$\mathcal{I}(x)_{i,j} = \text{cov}_x \left( \frac{\partial \ell}{\partial x_i}, \frac{\partial \ell}{\partial x_j} \right)$$

(4.1)

For a multivariate normal distribution (4.1) reduces to $\mathcal{I}_{i,j}(x) = G^T C^{-1} G$, which is equivalent to the Hessian matrix in the Gauss-Newton method. This is why in least squares problems like SLAM the Hessian (and inverse covariance) is the Information Matrix[95]. Iterative batch solutions to non-linear problems generally give better parameter estimates than both non-iterative methods and first order recursive methods. For example, because they do not iterate, methods like the Extended Kalman Filter have no guarantee of converging to the local cost minimum of the objective function, while iterative techniques like the Gauss-Newton method have a well established convergence theory[20]. For convergence, one would have to use the iterative Extended Kalman Filter (which is algebraically equivalent to the Gauss-Newton method[5]). Iteration is fundamental.

In SLAM, batch estimators are beneficial because new observations can often improve old pose estimates, which can in turn lead to better estimate for the current pose. For instance, new landmark measurements can help estimate not only where the current pose, but also previous poses. Recursive methods marginalize out old poses and hence cannot improve previous pose estimates. On the other hand, batch methods maintain
the old pose information necessary to re-evaluate and re-linearize the objective function using all measurements, which improves a significant portion of the robot path estimate. Estimating the robot path and re-considering past measurements both lead to better (smaller residual, less variance) parameter estimates. Batch methods also produce smooth trajectories (hence techniques that optimize over more than one time step are called *smoothers*).

The covariance estimate in Recursive Least Squares like the EKF is based on the assumption that second derivative terms in the true Hessian are negligible. The closer the parameter estimate to the true value the better this covariance approximation will be. Since iterative batch least squares solutions give better parameter estimates, the approximated covariance matrix (and information matrix) is also more accurate. In the non-linear least squares framework, estimating over all the parameters in the system, considering the entire measurement history, and iterating the update are crucial for attaining an optimal, efficient and consistent estimate.

### 4.5 Application to Entry Descent and Landing

In this section we test the SWF with real data captured to emulate Entry Descent and Landing conditions for a Mars lander. Using this imagery we are able to demonstrate a signature $1/n$ reduction in ground structure uncertainty and estimator error. Such technology can thus allow for the accurate determination of ground structure from greater
altitude during EDL, and hence allow more time for hazard avoidance maneuvers prior to touchdown.

4.5.1 Data Cleaning

When dealing with real data one must invariably handle the issues of Data association, the correspondence problem, and outlier rejection. Our approach to these problems is two fold: first we employ a coarse outlier rejection scheme to remove gross outliers, and second we use robust least squares m-estimation to reduce sensitivity to any remaining outliers in the inlier set.

A very practical inlier selection mechanism is the consistency check of[70], which has recently been re-discovered and extended by[39]. This method relies on knowing 3D structure from stereo at each frame. Given putative correspondences between two 3D data sets, this method can quickly and effectively find the largest set of consistent correspondences that belong to a single rigid body transform - hence it is a useful first cut pass at discovering the dominant motion, which is usually the static scene. Since this method detects the largest consistent set of correspondences, it is capable of finding the correct inlier set even in cases where the inlier set is less than 50 percent of the data (it just has to be the largest consistent subset). It is also possible to extend Moravec’s basic method to handle the greater uncertainty that plagues long range stereo[39]. In our work, we have found it sufficient to first employ Moravec’s method, followed by robust M-estimation using Huber’s error function in the core of the SWF estimator[85, 41].
Figure 4.6: Estimator error after 25 frames vs Distance off the optical axis, as measured from a landing vehicle descending at 10ms, capturing frames at 10hz, starting at 100m and stopping at 75m. The FOV is $\sim$25 deg.; the stereo baseline is 1m; image resolution is 512x384.

4.5.2 Stereo Convergence at the Focus of Expansion

Here we note our motivation for using stereo cameras, as opposed to a monocular setup. Clearly, a monocular camera can only triangulate given multiple views with a sufficient baseline. This necessitates horizontal or vertical translation. With monocular cameras there is no depth information at the focus of expansion, which is in the center of the image for forward motion. For robots, like many animals in Nature, motion along the line of site is likely to be the predominant type of movement. We favor stereo vision systems for this simple and practical reason (note however that, given an initialization
routine, the SWF will work just fine with monocular cameras - especially if the inverse depth formulation is used[69], or if homogeneous coordinates are used[37, 104]).

Following the above argument, and in the context of Entry Descent and Landing, the limitations of downward pointing monocular cameras for landing vehicles are readily apparent. This leads us to prefer wide baseline, long range stereo for EDL, Hazard Detection and Avoidance (HDA) and ultimately safe and precise landing.

In contrast to monocular setups, stereo can establish range estimates along the line of sight. Fig. 4.6 shows ground range uncertainty vs distance from the viewing axis (straight down) as measured from a landing vehicle descending at 10ms, capturing frames at 10hz. The ability to establish range along the primary viewing and traveling direction and is a fundamental advantage stereo has over monocular sequences.

4.5.3 Convergence Results

We tested the convergence of the SWF with real data captured to emulate Mars Entry Descent and Landing. A large flat surface was covered with orbital imagery from the Mars Hi-Rise camera, which has sub-meter per pixel resolution. A stereo rig was then “flown” at this surface to generate descent imagery. A still from this sequence is shown in Fig. 4.10.

To aid algorithm development, and to help establish ground truth, separate sequences were taken of the wall in which fiducials were added. This setup is shown in Fig. 4.7. To establish ground truth we computed the SWF solution over all frames (which matches
Figure 4.7: For validation, here the SWF exhibits $1/n$ convergence for a sequence with stationary cameras and tracked fiducials.

The batch least squares SLAM solution, tracks the CRLB, and is the best one can do within the least squares Gaussian framework. We then used the converged result to find the wall plane equation. From this we were able to determine that the wall was indeed very flat (with average residuals of 0.1mm), and hence that using a plane as a model is sufficient. Convergence results for this engineered stationary sequence are shown in Fig. 4.8.

In future, it is envisioned that the nominal EDL baseline will be approximately 1m, and that, depending on camera field of view and resolution, stereo ground ranging will
Figure 4.8: Fiducials used for experimental analysis. Each fiducial is localized to sub-pixel accuracy (by fitting a saddle point to the checker pattern). Each fiducial has a unique bar-code that can be read automatically.

commence anywhere between 300m to 50m. To emulate these conditions within a confined laboratory we used a smaller baseline of \( \sim 10\text{cm} \) and moved the cameras 10m to 1m from the landing surface. Hence, modulo image noise and footprint, this should compare to a sequence taken from 100m to 10m.

To test the SWF under the nominal case, where the cameras are moving and there are no fiducials, we moved the camera along a “vertical” trajectory from 10m to 1m, capturing images every \( \sim 15\text{cm} \) (or 6in). Scaled by an order of magnitude, this represents a frame every 1.5m. Given a framerate of 15hz, means the descent velocity is 10\( \text{m/s} \), which is reasonable for a lander with a parachute.
During this experiment it became apparent that we needed to improve the accuracy of the feature tracker to achieve $1/n$ convergence. This was accomplished by adding a homographic warping to each feature track, assuming that each feature lies on a locally planar patch, with the range defined by the landmark state estimate and plane normal by the optical axis of the first frame in which the feature is tracked. Using this inferred scene plane, we are able to warp image patches from frame $n$ to frame 1. We then perform Lucas-Kanade least squares translation only patch alignment to compute the sub-pixel feature location\[59\]. For optimal results, the sub-pixel patch location is refined after each SWF iteration.

Convergence results for 10 frames from this sequence are shown in Fig. 4.9. This sequence tracks the same 150 features throughout; we hence use it to check for the signature $1/n$ convergence. It is apparent that we have achieved slightly better than $1/n$ convergence - which is to be expected given that range uncertainty naturally reduces as the distance to the surface shrinks. Note that the 3, 4 and 5 frame filters come close to matching the all time filter.

4.6 Additional Remarks on the Sliding Window Filter

In theory, the Sliding Window Filter offers a method to achieve the optimal all-time $MAP$ estimate at reduced complexity, potentially even at constant complexity. A key question that remains to be answered is how much information is lost when we transform the system p.d.f. via downdating? If we can quantify this information loss then we can
Figure 4.9: Convergence results for moving cameras for Mars EDL experiment data. 150 features are tracked over 10 frames and fused with a SWF of 3, 4, 5, and 10 frames. This plot demonstrates slightly better than $1/n$ convergence because range uncertainty is also shrinking as the cameras get closer to the wall.

possibly show a bounded equivalence between the full SLAM solution and a constant time approximation based on downdating.

Notice that even though the Sliding Window Filter technique focuses on the spatially immediate estimation process, it also incrementally builds a map structure (occupying $O(n)$ memory space) that is useful for non-local problems, such as loop closure, place recognition, navigation, topological planning, etc. Most of this map is not active in the estimator - it is passive. So while the SWF does not address loop closure per-se, it does build a representation amenable to solving such problems. Fundamentally, the
Figure 4.10: Example left image from the EDL sequence. This sequence used grayscale cameras with 1024x768 resolution, 10cm baseline, 25 deg. FOV.

information necessary to compute the full batch solution is not lost - it is passive and available for a batch solution. Though of course such a solution cannot be computed in constant time.

Ultimately, we believe that loop closure is best cast as a distinct problem from local estimation. In this light, combining Sliding Window Filter with new methods for recognizing loop closures is an interesting idea[78].

Decoupling loop closure from the core SLAM estimator allows concentrating computational resources on improving the local result, which is crucial for applications that
require spatially high-resolution, dense structure estimates. With high bandwidth sensors (like cameras) focusing on the local problem is clearly important for computational reasons; this is especially true if we wish to fuse all of the sensor data (or a significant portion thereof). Even with this local focus, it is a remarkable fact that once a loop closure is identified, global optimization over the map structure left behind by the SWF should match the global batch solution.

4.7 Conclusion

Tackling SLAM from the classic Statistical Point Estimation perspective gives the problem an intuitive, elegant simplicity - especially when compared against the algebraically equivalent smoothing-filters.

This chapter describes a SLAM solution that concentrates computational resources on accurately estimating the immediate spatial surroundings by using a sliding time window of the most recent sensor measurements. Focusing computation on improving the local result is crucial for applications that require spatially high-resolution, dense structure estimates. With high bandwidth sensors (like cameras) this is clearly beneficial for computational reasons, and it especially true if we wish to fuse all of the sensor data (or a significant portion thereof).

The ultimate goal is to develop a filter that can quickly and optimally fuse all data from a sequence of (stereo) images into a single underlying statistically accurate and precise spatial representation. Such a capability is highly desirable for mobile robots,
though it is a computationally intense *dense data-fusion* problem. The SWF is useful in this context because it can scale from exhaustive batch solutions to fast incremental solutions. For instance if the window encompasses all time, the solution is algebraically equivalent to full SLAM; if only one time step is maintained, the solution is algebraically equivalent to the Extended Kalman Filter SLAM solution; if robot poses and environment landmarks are slowly marginalized out over time such that the state vector ceases to grow, then the filter becomes constant time, like Visual Odometry. Hence, by tuning a few parameters, the sliding window algorithm can scale from exhaustive Batch solutions to fast incremental solutions. The SWF has many other interesting properties; for instance, continuous sub-mapping, lazy data association, undelayed or delayed initialization, and incremental robust estimation.

Ideally, we would like a constant time algorithm that closely approximates the all-time maximum-likelihood estimate as well as the minimum variance Cramer Rao Lower Bound - e.g. we would like an estimator that achieves some notion of statistical optimality (quickly converges), efficiency (quickly reduces uncertainty) and consistency (avoids over-confidence). We find that approaching this problem from the Statistical Point Estimation point of view results in a simple, yet general, take on the SLAM problem; we think this is a useful contribution.

The Sliding Window Filter is generally applicable to a wide variety of sensors and environments, capable of working with or without a motion model and with or without proprioceptive sensors. This flexibility allows solving bearing only SLAM problems,
such as Bundle Adjustment, as well as traditional dynamic state space problems, like indoor SLAM with lasers. Further the approach is applicable to highly observable configurations, such as when using GPS like sensors. No special care is necessary to incorporate asynchronous update rates of various sensors.

This generality makes applying the Sliding Window Filter attractive for many problems. Some interesting examples would be 1) using orbital imagery, inertial sensing and a star tracker on-board a space craft to improve orbit estimation around small bodies like asteroids[1], 2) using stereo, odometry and inertial sensing for mobile manipulation [89] 3) using stereo and inertial sensors to land an autonomous helicopter [88] 4) safe and precise landing for planetary landers[2, 87].
Chapter 5

Related Work

We address three bodies of literature in this section. First, in relation to the second order Bias-Reduction filter in Ch. 2 and the Iterated Sigma Point Kalman Filter in Ch. 3, we discuss the small set of publications about uncertainty modeling and bias in stereo. Second, we investigate the more extensive SLAM literature as it relates to the Sliding Window Filter in Ch. 4. Finally, we look at a selection of techniques that are useful when handling real data. These are related to probabilistic data-association, and robust estimation in the presence of outliers. These techniques are also useful for SLAM in dynamic environments.

5.1 Error Modeling and Bias in Stereo

While bias in stereo is a known phenomenon, it is not well studied in the literature. Previous research focused on how range bias is induced from uncertain camera positions[86, 111, 112, 16], or dismissed it as insignificant [62].
Bias reduction for least squares estimators is typically done via a second order expansion [14, 91]. More recently, second order filter have been replaced with Unscented Filters, which get the second order bias correction “for free” [107, 44, 49, 106]. Ch. 3 describes one such estimator, the Iterated Sigma Point Kalman Filter. Non-parametric sampling methods such as Particle Filters can overcome problems with parametric techniques[43], though our knowledge they have not been applied to the stereo ranging problem.

5.2 Simultaneous Localization and Mapping

The Sliding Window Filter was inspired by the results from the Photogrammetry community, dating back to the late 1950’s[9], and later derivatives like the Variable State Dimension Filter[65, 66], Visual Odometry[63], and of course Extended Kalman Filter SLAM[94]. The techniques of Photogrammetry were gradually adopted or rediscovered as Visual Odometry and Shape from Motion in the computer vision community[63, 104] and Simultaneous Localization and Mapping in the robotics community[58, 100]. These are all least squares estimators - often expressing algebraically equivalent solutions. In older work, mapping and localization have been addressed many times[13, 8, 72], though contemporary approaches have gradually evolved toward a more rigorous probabilistic point of view[100]. The first real time vision based EKF SLAM implementation is to Davison[17].

Photogrammetric Bundle Adjustment (BA) is the original image based batch Maximum Likelihood solution to the full SLAM problem from the iterative non-linear Least
Squares perspective. However, BA does not typically include prior information or a process model, which can be useful for SLAM. Over the decades, incremental/recursive BA algorithms have been formulated [67]. To facilitate faster run-times these derivations usually assume that cross-correlations can be ignored and hence they do not accurately capture the temporal evolution of the system probability density function.

Exactly Sparse Delayed State Filters (ESDSF) are a view-based approach inspired by both the VSDF and Sparse Extended Information Filters[98, 23]. ESDSFs are efficient approximations to the EKF SLAM solution, though rely on scan-matching raw data, so the assumption of independent measurement noise in the sensor model may be violated. In practice, to avoid “double counting” the data, sensor measurements must be carefully overlapped, which ultimately results in a restriction on how sensor data is acquired[23].

GraphSLAM is an offline technique nearly identical to Bundle Adjustment[100]. In some sense, the Sliding Window Filters are the opposite of GraphSLAM[100] and Delayed State Filters: where these methods factor the map onto the path, the SWF slowly factors the path onto the map. This has important implications for the run time complexity as the algorithm progresses. In GraphSLAM, as the map is downdated onto the path, the induced structure in the path block, $\Lambda_p$, can grow to be arbitrarily complex. This stems from the fact that there are an infinite variety of paths through an environment - and usually we will not know how the robot is going to move beforehand. On the other hand, downdating the path onto the map only ever induces a structure with a bounded complexity as there is limited number of landmark-to-landmark conditional dependencies
induced. Fundamentally, while there is an infinite variety of paths through the environment, there is just one environment. This point is a crucial distinction between methods that downdate onto the path and methods that do not. Interestingly, GraphSLAM allows the powerful capability of lazy data association (the ability to revise old data association in light of new information)[35, 40]. The SWF also enjoys this capability, which enables searching more (not all) of the space of possible maps (from all possible data associations). Lazy data association also allows the SWF to track the state p.d.f. through ambiguous situations, and is hence an alternative to multi-modal tracking techniques.

The Variable State Dimension Filter[65, 66] is more recent attempt to combine the benefits of batch least squares with those of recursive estimation. Of all the literature surveyed here, the VSDF is most similar to our work. The VSDF is a mixed formulation, taking inspiration from the Sparse Levenberg-Marquardt method used in Bundle Adjustment[71, 37], and also from the traditional Extended Kalman Filter used in SLAM[94]. For computational efficiency, the VSDF ignores conditional dependencies that are induced from marginalizing out old parameters, and like Bundle Adjustment it also ignores conditional dependences that exist between adjacent pose parameters - especially the block tridiagonal matrix structure of the process block, $\Lambda_p$. In comparison, the least squares formulation for full SLAM captures this information naturally. Neglecting conditionl dependncies can be detrimental; in SLAM it can lead to divergence [79].

Techniques that are nearly identical to Bundle Adjustment are GraphSLAM [100], Delayed State Filters (DSFs)[23, 56], and Smoothing and Mapping (SAM)[19]. The first
two of these techniques downdate the map onto the robot path, a well known approach in Photogrammetry. SAM tries to solve the system equations efficiently by variable re-ordering to accelerate solving the least squares system equations, which is also a well known technique in Photogrammetry[104]. The success of this approach depends critically on the structure of the least square system matrix, which generally cannot be known beforehand since it depends on how the robot goes about observing the world. General re-ordering algorithms that work for arbitrary system equations are known to be NP-complete[110]. GraphSLAM is an off-line solution and is typically tackled with canned numerical sparse solvers[83, 30]. Generally, Delayed State Filters can be solved online with complexity that is cubic in the number of poses.

In some sense, Sliding Window Filters are the opposite of GraphSLAM and Delayed State Filters. Where these methods factor the map onto the path, the Sliding Window Filter slowly factors the path onto the map. This has important implications for the run time complexity as the algorithm progresses. In GraphSLAM, as the map is downdate onto the path, the induced structure in the path block can grow to be arbitrarily complex. This stems from the fact that there are an infinite variety of paths through an environment and usually we will not know how the robot is going to move beforehand. On the other hand, downdating the path onto the map only ever induces a structure with a bounded complexity as there is limited number of landmark-to-landmark conditional dependencies induced. Fundamentally, while there is an infinite variety of paths through the environment, there is just one environment. This point is a crucial distinction between methods
that downdate onto the path and methods that do not. Interestingly, GraphSLAM allows the powerful capability of lazy data association (the ability to revise old data association in light of new information) [35, 40]. The SWF also enjoys this capability, which enables searching more (not all) of the space of possible maps (from all possible data associations). Lazy data association also allows the SWF to track the state PDF through ambiguous situations, and is hence an alternative to multi-modal tracking techniques.

Like the SWF, the Sparse Extended Information Filter (SEIF) is another solution to the SLAM problem that aims for constant time complexity [98]. In EIFs, poses are rolled up into the prior and only the most recent pose is ever active in the state vector. The conditional dependencies that this induces are then dealt with via a “sparsification” routine that deletes weak links between parameters. Hence, like the VSDF, SEIF ends up ignoring conditional dependencies that are induced when the pose history is rolled up into the most current pose estimate. Because SEIFs and DSFs use the information formulation, the state and covariance is not directly accessible, and calculating them requires inverting the entire system information matrix (an \( n^3 \) operation). SEIFs address this via an approximation that searches for sets of conditionally independent parameters (a so-called Markov blanket) [99] which can then be extracted without solving for the remaining parameters. Note that probabilistic data-association relies on knowing the state and covariance, which thus makes data association difficult in Information Filters (or expensive at least). Since the SWF maintains a bounded size map, covariance extraction, and hence probabilistic data association, are constant time operations.
Other algorithms closely related are the Thin Junction Tree Filters\cite{82}(TJTF), and TreeMap\cite{29}, both of which roll up the process information into the prior and hence do not smooth over the robot path. Both techniques make explicit the connection to graphical modeling (as does GraphSLAM) as an underlying tool for thinking about the SLAM problem (and statistical models in general). TJTF in particular is a beautiful example of modern inference techniques on graphical models.

Understanding the connection between traditional least squares and graphical models can help us choose appropriate tools for solving problems\cite{73}. Graphical Models, like Least Squares, are useful for understanding problem structure precisely because they are intuitive. Graphical Models also lend themselves to integrating disparate estimation problems into a single framework \cite{46}.

Further, in Graphical models, Information Filters or plain Least Squares like the Gauss-Newton method developed here, the concept of smoothing an $n^{th}$ order Markov model is readily apparent - it is clear that the process model encodes conditional dependencies between adjacent poses, and that the estimation operates over the $n$ poses in the path. An equivalent solution can be found via the Kalman Smoother\cite{5, 104}. However, the Kalman Smoother relies on potentially non-intuitive notions like “backwards filtering” and “future estimates”, and is generally less accessible. With least squares and the intuitive relationship to graphical models, these concepts are simplified.

The computation and storage costs of the most well known SLAM solution, Extended Kalman Filter stochastic mapping, is $O(n^2)$ in the number of landmarks\cite{94}. This has
prompted extensive research aimed at reducing these costs in a principled way while still maintaining the beneficial properties of the EFK solution[15, 7, 82, 98, 28].

Most approaches make use of the observation that conditional dependencies between landmarks are often negligible, and hence that the information matrix (inverse covariance matrix) is nearly sparse [98, 82]. This leads naturally to approximate de-correlation techniques where only a subset of cross-correlation terms are maintained at any given time [48, 98, 82].

As a non-smoothing, non-iterative solution to a non-linear least squares problem, the EKF is sub-optimal when compared to the full batch SLAM solution. Methods that take the EKF as their baseline for comparison are unlikely match the optimal estimator. Both EKF SLAM and SEIF SLAM are special examples of least squares SLAM. However, unlike the least squares SLAM, neither the EKF nor the SEIF are both optimal and efficient. A Gauss-Newton batch estimator over $n$ time steps is algebraically equivalent to an “order $n$” Iterated Extended Kalman Smoother[4]. The traditional EKF SLAM solution is simply an order 1 filter. For significantly non-linear problems, higher order filters will yield better results than lower order filters. That the least squares formulation is not the standard approach for SLAM is perhaps due to the historical need for fast recursive algorithms in online estimation, which lead naturally to the Kalman Filter. Unfortunately, the Kalman Filter is only optimal for linear problems. To improve the Kalman Filter beyond linear problems one has to add both iterative linearization and smoothing, which in many ways defeats the reason for starting with recursive filter in the first place. The
convoluted concepts of “backward time filtering” etc. in the Kalman Smoothers testify to the difficulty of using the KF as a basis for developing optimal non-linear estimators.

It has been our experience that the EKF perspective often leads to a lack of appreciation for the essence of the least squares problem at hand - i.e. the EKF perspective obscures its very origins as a batch Least Squares problem. For these reasons, we prefer the least squares perspective.

Non-parametric techniques have also been applied to SLAM, the most notable is the Particle Filter[27], which represents the posterior state distribution with a set of samples. Clearly, with infinite samples we can represent any distribution. Particle Filter can track multiple modes of a distribution with a small set of particles via the use of Selective Importance Re-sampling [21]. The scalability of these filters is poorly understood (in terms of the number of particles needed for convergence). Even highly optimized hybrids like FastSLAM [68] are still $O(m \log(n))$ during exploration (where $m$ is the number of particles). One solution is to factor the problem into disjoint estimators, where one estimator assumes the output of the other is true - this is the basic idea behind Rao-Blackwellized Particle Filters[74] (which FastSLAM is an instance of). It is not clear how the number of particles should increase as the map grows. As we increase the dimensionality of the problem, the number of particles is likely increase more than linearly. Further, in practice, Particle Filters often diverge do to the so-called particle impoverishment problem.

To bound the computational complexity of EKF SLAM, many authors suggest the use of relative sub-maps where each sub-map computes an independent EKF solution
and sub-maps are related to each other via a global map of maps[15, 54, 7, 11]. Julier has shown that to bound robot covariance requires maintaining and updating $O(n)$ cross-correlation terms between the sub-maps[48]. This view is supported by Frese[28], who describes an approach using multi-grid relaxation methods that solve the non-linear least squares minimization problem in the context of relaxation on a graph[28]. The Compressed Extended Kalman Filter (CEKF) [32] estimates a sub-portion of the map and can achieve $O(1)$ updates within a sub-map. However, propagating changes to the entire map still requires $O(n^2)$ in the number of landmarks. Interestingly, the SWF is essentially a continuous sub-map method (though of course the cost of closing loops is not addressed by the SWF). Other constant time algorithms [52, 77, 52] use the idea of “postponement”, which is similar to the Delayed State approach.

Laser scan matching, which is typically implemented using Iterated Closest Point algorithm[6], has been instrumental in fielding SLAM algorithms[33, 34]. Scan matching is essentially Visual Odometry for lasers. It is important to note that combining scan matching with SLAM can lead to “double counting the data”, this is because scan matching uses sensor data to compute the incremental pose estimate, while SLAM also uses this same data to refine the pose and map estimate. In practice this difficulty is often resolved by only using some measurements for scan matching and some for SLAM. Note that the Sliding Window Filter overcomes this difficulty, simultaneously performing incremental alignment as well as updating the map.
5.3 Correspondence, Data-association and Outlier Rejection

Data Association is a fundamental issue in SLAM since false data associations, which are outliers, are likely to cause divergence. Most SLAM algorithms perform Probabilistic Data Association (PDA) using statistical validation gating on an individual landmark basis. This work builds off the earlier work from the target tracking and motion estimation communities [3, 26], where a threshold Mahalanobis distance is used to reject or accept associations. More recent Batch Probabilistic Data Association (BPDA) has been used [75], [55] that exploits the geometric relationship between groups of landmarks. In the SLAM context there are two known implementations of BPDA: Joint Compatibility Branch and Bound (JCBB) [75], which uses tree-search to test possible associations and Combined Constraint Data Association (CCDA) [76], which uses a graph to search for and test possible data associations. It is interesting to contrast Batch methods that search for solutions with sampling methods like RANSAC that randomly generate and test solutions.

In computer vision ego motion estimation is often used for video stabilization, and hence temporal correspondence must be found. It has been tackled from the optical flow perspective [42, 59, 97], and also as a homography estimation problem [38]. Like Visual Odometry and vision based SLAM, these methods rely on tracking salient features, and there is a wide variety of feature extraction and tracking methods [12, 90, 101, 114, 57, 36, 25]. In their basic form, all these techniques suffer from false data-association, and all suffer in the presence of moving objects.
To overcome these issues, one turns to robust estimation techniques that are able to segment out the largest self-consistent portion of the input data, which is usually the static part of the environment.

Robust regression and outlier detection are often used to improve the performance of parametric estimators like least squares[85], which quickly degrades in the presence of outliers. Perhaps the most popular techniques are Random Sampling Consensus (RANSAC)[24] and Least Median Squares (LMS)[84]. Both RANSAC and LMS require large number of samples to compute parameter hypothesis, prior knowledge of the outlier ratio and additional, difficult to obtain, inlier threshold for hypothesis evaluation.

MLESAC (maximum likelihood estimate sampling consensus) is a natural extension to RANSAC and LMS that scores each hypothesis sample based on its probabilistic likelihood[103]. Guided-MLESAC uses domain knowledge to guide the sampling process[102]. Like RANSAC and LMS, MLESAC and Guided-MLESAC can require large number of samples to find an inlier solution with high probability.

Techniques that overcome these drawbacks include Adaptive Scale Residual Consensus[108], and a new technique that works by segmenting the 3rd and 4th order distributions of the residual histogram[113]. These methods can correctly identify the largest residual-consensus cluster with greater than 50 percent noise (80 percent and 70 percent, respectively). The most important aspect of these new methods is that they also learn the scale of the inlier noise, and hence give information about the underlying reference
variance of the system. Presently, we simply estimate the reference variance (using only the inlier data) as the estimator measurement variance after convergence.

A very practical inlier selection mechanism is the consistency check of[70], which has recently been re-discovered and extended by[39]. This method relies on knowing 3D structure from stereo at each frame. Given putative correspondences bewtween two 3D data sets, this method can quickly and effectively find the largest set of consistent correspondences that belong to a single rigid body transform - hence it is a useful first cut pass at discovering the dominant motion, which is usually the static scene. Since this method detects the largest consistent set of correspondences, it is capable of finding the correct inlier set even in cases where the inlier set is less than 50 percent of the data (it just has to be the largest consistent subset). It is also possible to extend Morvec’s basic method to handle the greater uncertainty that plagues long range stereo[39]. In our work, we have found it sufficient to first employ Morvec’s method, followed by robust M-estimation using Huber’s error function in the core of the SWF estimator[85, 41].
Chapter 6

Conclusion

Our aim throughout has been to derive accurate and precise stereo range estimates by filtering sequences of images. We study both stationary and mobile cameras. Though our work is widely applicable to other sensors as well, we focus on robots with stereo-vision, inertial and odometric sensors. This choice of sensor leads us to develop new methods to handle statistical bias present in stereo sensing.

In Ch. 2 our investigation of long-range stereo data fusion began with an analysis of filtering stereo image sequences taken from a stationary rig. We saw that traditionally, measurement errors from stereo camera systems have been approximated as 3D Gaussians, where the mean is derived by triangulation and the covariance by linearized error propagation. Unfortunately, there are two problems that arise when filtering such 3D measurements. First, stereo triangulation suffers from a range dependent statistical bias; when filtering this leads to over-estimating the true range. Second, filtering 3D measurements derived via linearized error propagation leads to apparent filter divergence; the estimator is biased to under-estimate range. To address the first issue, we examine
the statistical behavior of stereo triangulation and show how to remove the bias by series expansion. The solution to the second problem is to filter with image coordinates as measurements instead of triangulated 3D coordinates. Compared to the traditional approach, we show that bias is reduced by more than an order of magnitude, and that the variance of the estimator approaches the Cramer-Rao lower bound.

In Ch. 3 we explore the use of statistical linearization to improve iterative non-linear least squares estimators, and we were able improve long range stereo by filtering feature tracks from sequences of stereo pairs. We developed a novel filter called the Iterated Sigma Point Kalman Filter (ISPKF); and show that it achieves superior performance in terms of efficiency and accuracy when compared to the Extended Kalman Filter (EKF), Unscented Kalman Filter (UKF), and Gauss-Newton filter. We also compared the ISPKF to the optimal Batch filter and to a Gauss-Newton Smoothing filter. For the long range stereo problem the ISPKF comes closest to matching the performance of the full batch MLE estimator. Further, the ISPKF is demonstrated on real data in the context of modeling environment structure from long range stereo data.

In Ch. 4 we address long range stereo estimation from a moving platforms. We develop a new Sliding Window Filter (SWF) that is an on-line constant-time solution to the feature-based 6-degree-of-freedom full Batch Least Squares Simultaneous Localization and Mapping (SLAM) problem. We contend that for SLAM to be useful in large environments and over extensive run-times, its computational time complexity must be constant, and its memory requirements should be at most linear. Under this
constraint, the “best” algorithm will be the one that comes closest to matching the all-
time maximum-likelihood estimate of the full SLAM problem, while also maintaining
consistency. We start by formulating SLAM as a Batch Least Squares state estimation
problem, and then show how to modify the Batch estimator into an approximate sliding
window Batch/Recursive framework that achieves constant time complexity and linear
space complexity. Viewing SLAM from the sliding window Least Squares perspective
is very useful for understanding the structure of the problem. This perspective is gen-
eral, capable of subsuming a number of common estimation techniques such as Bundle
Adjustment and Extended Kalman Filter SLAM. By tuning the sliding window, the al-
gorithm can scale from the exhaustive Batch solutions to fast incremental solutions; if
the window encompasses all time, the solution is algebraically equivalent to full SLAM;
if only one time step is maintained, the solution is algebraically equivalent to the Ex-
tended Kalman Filter SLAM solution. We also point out that the SWF enables other
interesting properties, like continuous sub-mapping, lazy data association, undelayed or
delayed landmark initialization, and incremental robust estimation. We test the algorithm
with real data captured to emulate Entry Descent and Landing conditions for a Mars lan-
der. Using this imagery we are able to demonstrate a signature $1/n$ reduction in ground
structure uncertainty and estimator error. Such technology can enhance ground structure
determination from greater altitude during EDL, and hence allow more time for hazard
avoidance maneuvers prior to touchdown.
Beyond EDL scenarios, it is worth pointing out that optimal, efficient and consistent long range depth estimation is a generally useful capacity. For instance, we have applied the SWF to long range obstacle detection to improve ground vehicle navigation, and believe that it will be useful in other contexts as well.

### 6.1 Summary of Contributions

1. We developed a new Bias-Corrected Gauss-Newton filter for long range stereo. Reduced bias in stereo triangulation by more than an order of magnitude.

2. We show that temporal filtering of 3D stereo measurements leads to apparent divergence. The solution to this problem is to filter with image coordinates as measurements instead of triangulated 3D coordinates.

3. Using the Fisher information inequality we show that the bias-corrected Gauss-Newton stereo estimator approaches the minimum variance Cramer-Rao lower bound.

4. A novel filter called the Iterated Sigma Point Kalman Filter (ISPKF) is developed from first principles; this filter is shown to achieve superior performance in terms of efficiency and accuracy when compared to the Extended Kalman Filter (EKF), Unscented Kalman Filter (UKF), and Gauss-Newton filter.
5. We introduce the Sliding Window Filter (SWF) that is an on-line constant-time solution to the feature-based 6-degree-of-freedom \textit{full} Batch Least Squares Simultaneous Localization and Mapping (SLAM) problem.

6. We demonstrate signature $1/n$ convergence of the Sliding Window Filter with real imagery emulating Mars Entry Descent and Landing.

\subsection*{6.2 Final Remarks}

The bulk of this thesis has focused on accurate long range stereo estimation, from both static and moving platforms. At this point we would like to highlight that, beyond long range stereo, there are two fundamental and well known sub-problems addressed in this work. The first is known as the correspondence, or data association problem, which asks how to establish correspondence between measurements of the same physical phenomenon when the measurements are taken at different times or different perspectives. Note that in this context we consider data-association, the correspondence problem, and inlier-outlier segmentation as facets of the same problem (and we conflate them intentionally). Assuming a correspondence is known, the second is the SLAM problem that addresses the actual estimation of the robot location and environment structure.

We have endeavored to show that the Sliding Window Filter is a solid solution for SLAM (discounting the loop closure problem). For the correspondence problem, we currently use an extended version of Moravec’s rigid body consistency check, followed
by robust M-estimation within the core of the SLAM engine. Undoubtedly, more work is warranted for this later problem.

We feel it is important to appreciate that in the presence of uncertainty, the problems of correspondence and SLAM are intricately intertwined since an accurate solution to the SLAM problem requires the correct data association, while finding such a data-association often relies on a good SLAM estimate... hence they must be addressed simultaneously within a single framework; the Sliding Window Filter is one such framework.

The Sliding Window Filter for SLAM that concentrates computational resources on accurately estimating the immediate spatial surroundings. Focusing computation on improving the local result is crucial for applications that require spatially high-resolution, dense structure estimates. With high bandwidth sensors (like cameras) this is clearly beneficial for computational reasons, and it especially true if we wish to fuse all of the sensor data (or a significant portion thereof).

The work presented here is founded on (conceptually simple) Recursive Least Squares. The goal is a constant time algorithm that closely approximates the all-time maximum-likelihood estimate as well as the minimum variance Cramer Rao Lower Bound - e.g. we would like an estimator that achieves some notion of statistical optimality (quickly converges), efficiency (quickly reduces uncertainty) and consistency (avoids over-confidence). We find that approaching this problem from the Statistical Point Estimation point of view results in a simple, yet general, take on the SLAM problem; we think this is a useful contribution.
Beyond performance, the Sliding Window Filter has a number of interesting properties that recommend its use in a large range of situations. For instance, during exploration, the Sliding Window Filter is a statistically optimal, efficient and consistent estimator, yet it has constant time complexity. SWF for SLAM is also an example of an online algorithm that allows changing past data-association decisions - so called lazy data association. This possibility is intriguing, because it allows the filter to overcome ambiguity without having to represent multiple modes in the posterior distribution.
Appendix A

A.1 Schur Complement, Gaussian Marginals, and the Matrix Inverse Lemma

In this section we look at some properties of block matrices and how they are used in recursive filtering [51]. First we look at the problem of computing the inverse of a matrix in terms its sub-matrices. This derivation leads to the Matrix Inversion Lemma, and makes use of an operation called the Schur complement, which for normal distributions in Canonical form turns out to be equivalent to marginalization.

The Matrix Inverse Lemma and Marginalization via the Schur Complement are useful for understanding recursive linear estimation theory. For instance, deriving the Kalman Filter from Bayes rule is greatly simplified by reference to the Matrix Inversion Lemma, and the update step in state estimation is an application of marginalization.
Let us say $M$ is a large matrix

$$M = \begin{bmatrix} A & B \\ C & D \end{bmatrix}$$

that we wish to invert, and we know that $A$ and $D$ are square and invertable. The first thing is to notice the two following simple matrix multiplications that allow us to triangularize $M$: first, the following left multiplication creates an upper-right triangular system,

$$\begin{bmatrix} I & 0 \\ -CA^{-1} & I \end{bmatrix} \begin{bmatrix} A & B \\ C & D \end{bmatrix} = \begin{bmatrix} A & B \\ 0 & \Delta A \end{bmatrix}$$

and the following right multiplication creates a lower-left triangular system,

$$\begin{bmatrix} A & B \\ C & D \end{bmatrix} \begin{bmatrix} I & -A^{-1}B \\ 0 & I \end{bmatrix} = \begin{bmatrix} A & 0 \\ C & \Delta A \end{bmatrix}$$

The term $\Delta_A = D - CA^{-1}B$ is called the Schur Complement of $A$ in $M$. Similarly, we can complement $D$ instead of $A$,

$$\begin{bmatrix} I & -BD^{-1} \\ 0 & I \end{bmatrix} \begin{bmatrix} A & B \\ C & D \end{bmatrix} = \begin{bmatrix} \Delta_D & 0 \\ \Delta_D & C \end{bmatrix}$$

and
\[
\begin{bmatrix}
A & B \\
C & D
\end{bmatrix}
\begin{bmatrix}
I & 0 \\
-D^{-1}C & I
\end{bmatrix}
= \begin{bmatrix}
\Delta_D & B \\
0 & D
\end{bmatrix}
\]

where \( \Delta_D = A - BD^{-1}C \) is the Schur Complement of \( D \) in \( M \).

Combining the above gives two different ways to block diagonalize \( M \):

\[
\begin{bmatrix}
I & 0 \\
-C A^{-1} & I
\end{bmatrix}
\begin{bmatrix}
A & B \\
C & D
\end{bmatrix}
\begin{bmatrix}
I & -A^{-1}B \\
0 & I
\end{bmatrix}
= \begin{bmatrix}
A & 0 \\
0 & \Delta_A
\end{bmatrix}
\]

and

\[
\begin{bmatrix}
I & -BD^{-1} \\
0 & I
\end{bmatrix}
\begin{bmatrix}
A & B \\
C & D
\end{bmatrix}
\begin{bmatrix}
I & 0 \\
-D^{-1}C & I
\end{bmatrix}
= \begin{bmatrix}
D & 0 \\
0 & \Delta_D
\end{bmatrix}.
\]

Using these we can re-express the original matrix \( M \) in terms of a lower-left block triangular component, a block diagonal component, and an upper-right block triangular component. That is,
\[
\begin{bmatrix}
A & B \\
C & D
\end{bmatrix}
= \begin{bmatrix}
I & 0 \\
CA^{-1} & I
\end{bmatrix}
\begin{bmatrix}
A & 0 \\
0 & \Delta_A
\end{bmatrix}
\begin{bmatrix}
I & A^{-1}B \\
0 & I
\end{bmatrix}
= \begin{bmatrix}
I & BD^{-1} \\
D & 0
\end{bmatrix}
\begin{bmatrix}
I & 0 \\
0 & \Delta_D
\end{bmatrix}
\begin{bmatrix}
I & 0 \\
D^{-1}C & I
\end{bmatrix}.
\]

which greatly simplifies computing the inverse since the middle term is block diagonal. For instance,

\[
\begin{bmatrix}
A & B \\
C & D
\end{bmatrix}^{-1}
= \begin{bmatrix}
I & -A^{-1}B \\
0 & I
\end{bmatrix}
\begin{bmatrix}
A^{-1} & 0 \\
0 & \Delta_A^{-1}
\end{bmatrix}
\begin{bmatrix}
I & 0 \\
-CA^{-1} & I
\end{bmatrix}
= \begin{bmatrix}
A^{-1} + A^{-1}B\Delta_D^{-1}CA^{-1} & -A^{-1}B\Delta_D^{-1} \\
-\Delta_D^{-1}CA^{-1} & \Delta_D^{-1}
\end{bmatrix}
\]

and equivalently,
\[
\begin{bmatrix}
A & B \\
C & D
\end{bmatrix}^{-1} = \begin{bmatrix}
I & 0 \\
-D^{-1}C & I
\end{bmatrix} \begin{bmatrix}
\Delta_D^{-1} & 0 \\
0 & D^{-1}
\end{bmatrix} \begin{bmatrix}
I & -BD^{-1}
\end{bmatrix}
\] (A.3)

\[
= \begin{bmatrix}
\Delta_A^{-1} & -\Delta_A^{-1}BD^{-1} \\
-D^{-1}C\Delta_A^{-1} & D^{-1} + D^{-1}C\Delta_A^{-1}BD^{-1}
\end{bmatrix}
\] (A.4)

Equating various terms of (A.2) and (A.4) yield the different forms of the Matrix Inverse Lemma, one of which is

\[
(A - BD^{-1}C)^{-1} = A^{-1} + A^{-1}B(D - CA^{-1}B)^{-1}CA^{-1}.
\]

This lemma is one of the primary tricks used for manipulating linear least squares equations (for instance, it is used to get between the Gauss-Newton method and the Kalman filter). The Schur Complement is the same as downdating, when applied to an inverse covariance matrix it is equivalent to Marginalizing a normal distribution [46].
Appendix B

B.1 Coordinate Frame Relations, Spatial Operations and Kinematics

This section discusses two tools for reasoning about spatial relationships, namely the compound and inverse operators. Generally, objects are represented by a pose vector which is an entity comprised of a 3D position and an orientation. A pose always has 6 degrees of freedom, though it may have more elements (i.e. a quaternion has 4 parameters to represent 3 degrees of rotational freedom, giving a pose vector with 7 elements). This detail has consequences. For example, it is worth remembering that all 3-parameter representations of orientation will have a singularity associated with them (e.g. gimbal-lock for Euler angles), and all 4 or more parameter representations are redundant, leading to other problems, such as rank deficient covariance matrices.

In general there is no one “best” representation of orientation, and it is often up to the practitioner to decide what is the right one for the application. One approach is to use quaternions to represent pose globally, and then use Euler angles to represent small local
changes. This gets the best of both worlds: a non-singular global representation and a minimal local representation.

Note that using local parameterizations like this buys us some of the benefits enjoyed by so-called error state Kalman filters, namely that the least squares optimization takes place around a state vector whose optimal solution will be near zero. In our case this has the effect of keeping the optimization machinery far away from the singularities of $SO^3$.

Therefore *local* pose, $x_{ij}$ is represented by a six parameter column vector comprised of a 3D point, $p_{ij} = [x_{ij} \ y_{ij} \ z_{ij}]^T$ and a 3 element Euler Roll-Pitch-Yaw angle $\theta_{ij} = [r_{ij} \ p_{ij} \ q_{ij}]^T$. The $+x$-axis is forward, $+y$-axis is right and $+z$-axis is down; roll is about $x$-axis, pitch about $y$-axis and yaw is about $z$-axis. Dropping the subscripts and abbreviating $\sin$ and $\cos$, a rotation matrix is parameterized by $\theta$ such that $R_\theta = R_z R_y R_x$.

where

$$R_x = \begin{bmatrix} 1 & 0 & 0 \\ 0 & cr & -sr \\ 0 & sr & cr \end{bmatrix}, \quad R_y = \begin{bmatrix} cp & 0 & sp \\ 0 & 1 & 0 \\ -sp & 0 & cp \end{bmatrix}, \quad R_z = \begin{bmatrix} cq & -sq & 0 \\ sq & cq & 0 \\ 0 & 0 & 1 \end{bmatrix}.$$

Say, for example, that $x_{wr}$ denotes the robot pose in the world coordinate frame and $x_{rs}$ denotes a sensor in the robot coordinate frame. The compound operator, $\oplus$, can be used to find the sensor pose in the world coordinate frame, e.g. $x_{ws} = x_{wr} \oplus x_{rs}$ (notice that the two $r$’s are adjacent).
More generally, given two spatial relationships, \( x_{ij} \) and \( x_{jk} \) the compound operator, \( \oplus \), gives the pose of object \( k \) in coordinate frame \( i \),
\[
x_{jk} = x_{ij} \oplus x_{jk} = \begin{bmatrix}
R_{\theta_{ij}} p_{jk} + p_{ij} \\
\gamma(R_{\theta_{ij}} R_{\theta_{jk}})
\end{bmatrix},
\]
(B.1)
where \( R_{\theta_{ij}} \) and \( R_{\theta_{jk}} \) are the 3x3 rotation matrices defined by the Euler angle components of \( x_{ij} \) and \( x_{jk} \), respectively. The function \( \gamma \) maps from a rotation matrix to the appropriate Euler angle
\[
\gamma(R) = \begin{bmatrix}
\arctan(R_{32}/R_{33}) \\
-\arcsin(R_{31}) \\
\arctan(R_{21}/R_{11})
\end{bmatrix}
\]
which is well defined for attitude angles between \(-90^\circ\) and \(90^\circ\). The compounding operation is associative but not commutative.

The inverse operation, \( \ominus \), inverts a spatial relation. For instance \( x_{ij} \) gives object \( j \)’s pose in \( i \)’s coordinate frame, and \( \ominus x_{ij} \) gives object \( i \)’s pose in \( j \)’s coordinate frame. The inverse operator is
\[
x_{ji} = \ominus x_{ij} = \begin{bmatrix}
-R_{\theta_{ij}} p_{ij} \\
\gamma(R_{\theta_{ij}}^T)
\end{bmatrix}
\]
Spatial operations are useful in a variety of problems. For instance, in estimation, they simplify writing sensor models and process models. In manipulator robotics, spatial operations can be used for forward and inverse kinematics. The benefit of thinking in terms of operations on spatial relations is the intuitive appeal, and the fact that the underlying linear-algebra does not need to be considered over and over again.

Jacobians of spatial operations are needed in many places - e.g. linear error propagation, Kalman filtering applications, least squares normal equations, non-linear optimization, etc. These Jacobians are involved, but tractable; it is recommended to find them with the aid of symbolic solvers, such as Maple or Mathematica.
Bibliography


