2005-03-18

Structure from Motion Using Optical Flow Probability Distributions

Paul Clark Merrell
Brigham Young University - Provo

STRUCTURE FROM MOTION USING OPTICAL FLOW PROBABILITY DISTRIBUTIONS

by

Paul C. Merrell

A thesis submitted to the faculty of
Brigham Young University in partial fulfillment
of the requirements for the degree of

Master of Science.

Department of Electrical and Computer Engineering
Brigham Young University
April 2005

BRIGHAM YOUNG UNIVERSITY

GRADUATE COMMITTEE APPROVAL

of a thesis submitted by

Paul C. Merrell

This thesis has been read by each member of the following graduate committee and by majority vote has been found to be satisfactory

Date	D. J. Lee, Chair
Date	Randal W. Beard
Date	Bryan S. Morse

BRIGHAM YOUNG UNIVERSITY

As chair of the candidate's graduate committee, I have read the thesis of Paul C. Merrell in its final form and have found that (1) its format, citations, and bibliographical style are consistent and acceptable and fulfill university and department style requirements; (2) its illustrative materials including figures, tables, and charts are in place; and (3) the final manuscript is satisfactory to the graduate committee and is ready for submission to the university library.

Date

D. J. Lee
Chair, Graduate Committee

Accepted for the Department

Michael A. Jensen
Graduate Coordinator

Accepted for the College

Douglas M. Chabries
Dean, Ira A. Fulton
College of Engineering and Technology

ABSTRACT

STRUCTURE FROM MOTION USING OPTICAL FLOW PROBABILITY DISTRIBUTIONS

Paul C. Merrell

Department of Electrical and Computer Engineering

Master of Science

Several novel structure from motion algorithms are presented that are designed to more effectively manage the problem of noise. In many practical applications, structure from motion algorithms fail to work properly because of the noise in the optical flow values. Most structure from motion algorithms implicitly assume that the noise is identically distributed and that the noise is white. Both assumptions are false. Some points can be track more easily than others and some points can be tracked more easily in a particular direction. The accuracy of each optical flow value can be quantified using an optical flow probability distribution. By using optical flow probability distributions in place of optical flow estimates in a structure from motion algorithm, a better understanding of the noise is developed and a more accurate solution is obtained.

Two different methods of calculating the optical flow probability distributions are presented. The first calculates non-Gaussian probability distributions and the second calculates Gaussian probability distributions. Three different methods for calculating structure from motion are presented that use these probability distributions. The first method works on two frames and can handle any kind of noise. The second method works on two frames and is restricted to only Gaussian noise. The final method works on multiple frames and uses Gaussian noise.

A simulation was created to directly compare the performance of methods that use optical flow probability distributions and methods that do not. The simulation results show that those methods which use the probability distributions better estimate the camera motion and the structure of the scene.

ACKNOWLEDGEMENTS

I would like to thank my adviser, Dr. D. J. Lee for his constant support and enthusiasm. I would also like to thank my family whom I dearly love, for their encouragement and support.

Contents

List of Figures

Chapter 1

Introduction

Structure from motion (SFM) is the technique of reconstructing the three-dimensional structure of a scene from a set of two-dimensional images captured from a camera moving within the scene. SFM is one of the most well-studied and important problems in computer vision. The problem has been studied for over two decades since the publication of Lonquet-Higgins's eight point algorithm [14] in 1981. It remains an important problem because of its numerous applications. For example, in robotics, a working method would allow a robot with only a camera to map out its environment and to detect and then avoid any obstacles in its path. In computer graphics, a working algorithm would allow a complete three-dimensional model of a scene to be created by simply moving a camera around the scene avoiding the difficult task of modeling it by hand.

Despite the considerable amount of research that has focused on the SFM problem, it remains a difficult and challenging problem. In a controlled environment, many SFM algorithms work well, but in many practical applications, the results are unsatisfactory. This poor performance is largely attributed to the noise that corrupts much of the data in real applications. This noise is poorly understood and is ineffectively handled by existing SFM algorithms.

1.1 Literature Review

A wide variety of different methods have been proposed for calculating structure from motion. Most methods work by tracking the motion of several feature points in the images. This motion is called the *optical flow*. From the optical flow, the motion of the camera and the three-dimensional structure of the scene is estimated. There are a few methods that do not use optical flow. They are called direct methods because they work directly from the image data without an intermediate calculation of the optical flow [8, 19].

Some SFM algorithms are designed for only two frames and others use multiple frames. One set of methods could be considered a combination of a two-frame and multiple-frame approach. These methods use two frames to find one intermediate solution. Then two more frames are used to find another intermediate solution and so on. Finally, all the intermediate solutions are then fused together or Kalman filtered to produce the final solution [3, 20, 24]. The accuracy of these methods largely depends on the accuracy of the intermediate solutions.

Another set of methods is based upon projective geometry [7, 11]. Projective geometry is used to calculate SFM without any camera calibration. However, in most applications, something is known about the camera calibration. The camera calibration may be imperfect, but the assumption that nothing is known about the camera calibration is unreasonable.

Another set of methods is based on the concept of factorization. Under orthographic projection, SFM can be framed as an optimal fixed-rank approximation problem, which is solved using a factorization method [27]. Factorization-based methods are appealing because they are simple and robust. However, the original factorization method is only valid when orthographic projection is an accurate approximation to true perspective projection. The original method has been extended to use more accurate approximations to true perspective projection such as weak perspective or paraperspective projections [21].

Another approach is to transform the SFM problem into a linear estimation problem [13]. Linear estimation is preferred over non-linear estimation because it is computationally efficient and more stable and reliable. Unfortunately, these methods are known to be biased. The translation estimate is biased towards the optical axis of the camera [26].

One of the most difficult aspects of SFM is the problem of noise. Noise corrupts many of the optical flow estimates and small errors in the optical flow estimates may lead to large errors in the estimation of the camera motion. One possible solution is to use some kind of outlier rejection [2, 9, 28] to remove any data that appears to be inaccurate because it is inconsistent with the other data. However, this approach may remove some valuable information. A better approach would be to identify those optical flow values that are less accurate and then to rely less heavily on the less accurate values rather than discarding them completely.

The accuracy and reliability of each optical flow value is best assessed from an optical flow probability distribution. There are several ways of calculating optical flow probability distributions, which will be examined in detail later. SFM should be calculated from the optical flow distributions rather than the optical flow estimates, because the probability distributions provide more information about the noise.

Most SFM algorithms simply assume that the noise in the optical flow estimates is identically distributed and white. Other types of noise have been largely overlooked in all but a handful of methods that do consider other kinds of noise. Most of these methods are factorization methods [1, 10, 12, 18], which are numerically robust but use an imperfect camera model. Zucchelli et al. [29] also consider non-identically distributed and nonwhite Gaussian noise, but instead of using a factorization method, they formulate the SFM problem as a nonlinear least squares minimization problem. The least squares problem is solved using a Gauss-Newton iteration. The Gauss-Newton iteration requires an initial estimate of the camera motion and then finds a better solution iteratively. This method is highly sensitive to the location of the initial estimate. If it is given a poor initial estimate, the solution often will converge to a local minimum, not to the global minimum.

The method presented here most closely resembles the optimal SFM methods of Soatto and Brockett [23] and Chiuso et al. [4], which are specifically designed to address the issue of noise. Their method is the optimal solution to the SFM problem when there is identically-distributed white Gaussian noise on each of the optical flow values. Their result will be extended by providing the optimal solution in the more general case where the noise is not required to be identically distributed and white.

Dellaert et al. [5] have developed a SFM algorithm where the exact motion of each feature point is unknown, but it is known to be one of a few possible values.

Similarly, the new approach presented here assumes the exact motion is unknown but could be one of a large number or a continuum of possible values. In addition, the new approach has the advantage of knowing the probability of each value.

1.2 Thesis Outline

Section 2 presents two different methods for calculating the optical flow probability distributions. Section 3 presents several different methods of calculating SFM that use the probability distributions. Some of the methods are designed to use only two frames and others are designed for multiple frames. One of the methods is designed to use any kind of probability distributions and the others are designed to only use Gaussian probability distributions. Section 4 presents the results from a number of different simulations that test the various methods. Experiments were conducted that use simulated data as well as data taken from computer-generated and real video sequences.

Chapter 2

Optical Flow Probability Distributions

There are several different ways of calculating optical flow probability distributions. One method could be described as a correlation-based method. In most cases, it is more accurate, but takes a longer time to compute. Another method is taken from the work of Simoncelli et al. [22]. It is a gradient-based approach, which makes some assumptions about the spatial gradients in the image. One assumption is that the spatial gradients are smooth. In order for the images to agree with this assumption, it is often necessary to blur the images before they are processed. The correlation-based method does not require this. Both methods assume that the motion of the image is a simple planar translation. This assumption is approximately correct except at depth discontinuities. The gradient-based method may be more accurate when this assumption is incorrect.

The gradient-based method only calculates the mean values and covariance matrices of the probability distributions and so it assumes that the noise is Gaussian. The correlation-based method calculates probability distributions that may be non-Gaussian. This has both advantages and disadvantages. The non-Gaussian probability distributions more accurately describe the true noise, but due to the mathematics of the SFM problem the non-Gaussian probability distributions are more difficult to manage.

There is an additional approach that has been suggested, which is to estimate the covariance matrix as the inverse of the Hessian matrix [25], but this method will not be discussed in any detail.

2.1 Correlation-Based Approach

The optical flow probability distributions are calculated at a total of *n* feature points. These feature points are chosen to be those points that have high spatial gradients in both the vertical and horizontal directions. Points with high gradients are the easiest points to track. Let $\mathbf{p}_i$ be the position of the *i*-th feature point. The image intensity at position $\mathbf{p}_i$ and at time *t* can be modeled as a signal plus white noise:

$$I(\mathbf{p}_i, t) = S(\mathbf{p}_i, t) + N(\mathbf{p}_i, t) \tag{1}$$

where $I(\mathbf{p}_i,t)$ represents the measured image intensity, $S(\mathbf{p}_i,t)$ represents the signal, and $N(\mathbf{p}_i,t)$ represents the noise. Over a sufficiently small time step, the change in the signal can be expressed as a simple translation:

$$S(\mathbf{p}_i + \mathbf{U}_i, t) = S(\mathbf{p}_i, t + dt) \tag{2}$$

where $\mathbf{U}_i$ is the optical flow vector between the two frames. If the first image is shifted by $\mathbf{u}_i$ and then the next image is subtracted from the shifted image, then

$$\begin{aligned} I(\mathbf{p}_i + \mathbf{u}_i, t) - I(\mathbf{p}_i + t + dt) &= S(\mathbf{p}_i + \mathbf{u}_i, t) - S(\mathbf{p}_i + \mathbf{U}_i, t + dt) \\ &\quad + N(\mathbf{p}_i + \mathbf{u}_i, t) - N(\mathbf{p}_i, t + dt) \end{aligned} . \tag{3}$$

If $\mathbf{u}_i = \mathbf{U}_i$, then the shifted difference contains only noise:

$$I(\mathbf{p}_i + \mathbf{U}_i, t) - I(\mathbf{p}_i, t + dt) = N(\mathbf{p}_i + \mathbf{U}_i, t) - N(\mathbf{p}_i, t + dt) . \tag{4}$$

The noise must be much smaller than the signal for there to be any chance of recovering the true optical flow. The shifted difference will be small if **u**$_i$ is equal to or close to **U**$_i$ and will be large otherwise. The probability that a particular optical flow **u**$_i$ is equal to the true optical flow value **U**$_i$ is equivalent to the probability that the magnitude of the shifted difference is equal to the magnitude of the noise. The shifted difference is a Gaussian random process. The probability that **u**$_i$ is equal to **U**$_i$ based on the image intensities is proportional to

$$P[\mathbf{U}_i = \mathbf{u}_i \mid I(\mathbf{p}_i + \mathbf{u}_i, t), I(\mathbf{p}_i, t + dt)] \propto \frac{1}{\sqrt{2\pi\sigma^2}} e^{-\frac{(I(\mathbf{p}_i + \mathbf{u}_i, t) - I(\mathbf{p}_i, t+dt))^2}{2\sigma^2}}, \quad (5)$$

for some σ^2, which depends upon the amount of noise in the image. This probability is found using a single point, but more than one point can be used. Points near the point **p**$_i$ are likely to have moved close to the same amount as **p**$_i$. Let B(**p**$_i$) be a set of points near **p**$_i$. By repeating the same analysis over the set B(**p**$_i$), and by assuming that the noise is independent, the optical flow probability is estimated as

$$P[\mathbf{U}_i = \mathbf{u}_i \mid I] \propto \prod_{\mathbf{p} \in B(\mathbf{p}_i)} \frac{1}{\sqrt{2\pi\sigma^2}} e^{-\frac{(I(\mathbf{p} + \mathbf{u}_i, t) - I(\mathbf{p}, t+dt))^2}{2\sigma^2}}. \quad (6)$$

To calculate a probability distribution, this calculation is repeated over a range of possible values of **u**$_i$. These probabilities are normalized, so that the probabilities for all possible values of **u**$_i$ sum to one.

2.2 Gradient-Based Approach

The following method for calculating optical flow distributions will be presented without a derivation. For a complete derivation, see [22]. The first step is to construct the following matrix and vectors using functions of the position **p**$_i$, the spatial gradients

$\mathbf{I}_x(\mathbf{p}_i,t) = \dfrac{\partial \mathbf{I}(\mathbf{p}_i,t)}{\partial x}$ and $\mathbf{I}_y(\mathbf{p}_i,t) = \dfrac{\partial \mathbf{I}(\mathbf{p}_i,t)}{\partial y}$, and the temporal-derivative of the image $\mathbf{I}_t(\mathbf{p}_i,t) = \dfrac{\partial \mathbf{I}(\mathbf{p}_i,t)}{\partial t}$:

$$\mathbf{M}(\mathbf{p}_i) = \begin{bmatrix} \mathbf{I}_x^2(\mathbf{p}_i,t) & \mathbf{I}_x(\mathbf{p}_i,t)\mathbf{I}_y(\mathbf{p}_i,t) \\ \mathbf{I}_x(\mathbf{p}_i,t)\mathbf{I}_y(\mathbf{p}_i,t) & \mathbf{I}_y^2(\mathbf{p}_i,t) \end{bmatrix}^{-1}, \tag{7}$$

$$\mathbf{b}(\mathbf{p}_i) = \begin{bmatrix} \mathbf{I}_x(\mathbf{p}_i,t)\mathbf{I}_t(\mathbf{p}_i,t) \\ \mathbf{I}_y(\mathbf{p}_i,t)\mathbf{I}_t(\mathbf{p}_i,t) \end{bmatrix}, \qquad \mathbf{f}_s(\mathbf{p}_i) = \begin{bmatrix} \mathbf{I}_x(\mathbf{p}_i) \\ \mathbf{I}_y(\mathbf{p}_i) \end{bmatrix}.$$

Each of these quantities is only calculated from the position $\mathbf{p}_i$. In the calculation, it would be best to use all of the points in the set $B(\mathbf{p}_i)$ that neighbor $\mathbf{p}_i$. The positions closer to $\mathbf{p}_i$ are more likely to have moved in the same direction as $\mathbf{p}_i$. They are more valuable and should be given greater weight. Let $\omega(\mathbf{p})$ be the weight attached to the position $\mathbf{p} \in B(\mathbf{p}_i)$ so that the positions closer to $\mathbf{p}_i$ are given more weight. These values are then used to calculate the covariance matrix of the optical flow, Ω_i, from the equation

$$\Omega_i = \left[\sum_{\mathbf{p} \in B(\mathbf{p}_i)} \frac{\omega(\mathbf{p})\mathbf{M}(\mathbf{p})}{\sigma_1 \|\mathbf{f}_s(\mathbf{p})\|^2 + \sigma_2} + \Omega_p^{-1} \right]^{-1} \tag{8}$$

with Ω_p being the covariance matrix of the prior distribution of the optical flow and with σ_1 and σ_2 being the variances associated with two different sources of noise. One source of noise is a product of the incorrect assumption that the motion of the image is a simple planar translation. σ_1 describes the errors introduced from the failure of this planarity assumption. σ_2 describes the errors introduced by an inaccurate temporal derivative, possibly from noise in the image intensities. These parameters may need to be adjusted based upon the quality of the images and the characteristics of the scene. In a typical image sequence, approximate values for each parameter have been found empirically to be $\sigma_1 = 0.08$, $\sigma_2 = 1.0$, and $\Omega_p = 2.0 \cdot \mathbf{I}$. The mean value of the optical flow $\mathbf{u}_i$ is given as

$$\mathbf{u}_i = -\Omega_i \cdot \sum_{\mathbf{p} \in B(\mathbf{p}_i)} \frac{\omega(\mathbf{p})\mathbf{b}(\mathbf{p})}{\sigma_1 \left\| \mathbf{f}_s(\mathbf{p}) \right\|^2 + \sigma_2}. \tag{9}$$

Up to this point, the image intensity has been treated as a single value. This would accurately describe a black and white camera, but typical cameras have red, green, and blue intensities. Let $\mathbf{M}_1(\mathbf{p}_i)$ be the value of $\mathbf{M}(\mathbf{p}_i)$ for the red image intensity and $\mathbf{M}_2(\mathbf{p}_i)$ and $\mathbf{M}_3(\mathbf{p}_i)$ be the values of $\mathbf{M}(\mathbf{p}_i)$ for the blue and green image intensities, and similarly let $\mathbf{f}_1(\mathbf{p})$, $\mathbf{f}_2(\mathbf{p})$, and $\mathbf{f}_3(\mathbf{p})$ be the values of $\mathbf{f}_s(\mathbf{p}_i)$ for the red, blue, and green image intensities. Ω_i can be recalculated as

$$\Omega_i = \left[\sum_{k=1}^{3} \sum_{\mathbf{p} \in B(\mathbf{p}_i)} \frac{\omega(\mathbf{p})\mathbf{M}_k(\mathbf{p})}{\sigma_1 \left\| \mathbf{f}_k(\mathbf{p}) \right\|^2 + \sigma_2} + \Omega_p^{-1} \right]^{-1}. \tag{10}$$

Likewise, if $\mathbf{b}_1(\mathbf{p}_i)$, $\mathbf{b}_2(\mathbf{p}_i)$, and $\mathbf{b}_3(\mathbf{p}_i)$ are the values of $\mathbf{b}(\mathbf{p}_i)$ for the red, green, and blue image intensities then $\mathbf{u}_i$ can be recalculated as

$$\mathbf{u}_i = -\Omega_i \cdot \sum_{k=1}^{3} \sum_{\mathbf{p} \in B(\mathbf{p}_i)} \frac{\omega(\mathbf{p})\mathbf{b}_k(\mathbf{p})}{\sigma_1 \left\| \mathbf{f}_k(\mathbf{p}) \right\|^2 + \sigma_2}. \tag{11}$$

The mean values $\mathbf{u}_i$ and the covariance matrix Ω_i define a Gaussian probability distribution.

Chapter 3

Structure from Motion

This section presents several different SFM algorithms that use probability distributions. The first is a method that is designed to use any kind of probability distributions and is designed for two frames. The second only uses Gaussian distributions and is designed for two frames. The third uses Gaussian distributions and is designed for multiple frames. By restricting the second and third methods to Gaussian distributions, they become simpler computationally. Allowing non-Gaussian probability distributions makes the task much more difficult so that a genetic algorithm is needed to solve the problem. The first method may take a very long time to find an accurate solution. However, given sufficient time, the first method will usually be more accurate than the second because the non-Gaussian probability distributions more accurately describe the noise.

All of these methods are nonlinear because they are iterative. Linear methods do have some advantages over nonlinear methods. The Gaussian methods in Sections 3.2 and 3.3 can easily be made linear simply by stopping after a single iteration and will work better than existing linear SFM algorithms that do not use optical flow probability distributions.

3.1 Structure from Motion with Non-Gaussian Noise

After using the correlation-based approach to find the non-Gaussian optical flow probability distributions, the goal is to find the probability of a given camera rotation and translation. The camera translation will be represented by the vector **a**, the camera rotation will be represented by the vector **b**, and the image data will be represented by **I**. The inverse depth, meaning the inverse of the distance from the camera to the objects in the scene, will be represented by the vector **λ**. The probability of a given translation, rotation, and depth value can be found by taking the expected value for all possible optical flow values:

$$P(\mathbf{a},\mathbf{b},\lambda \mid \mathbf{I}) = \iint \cdots \int P(\mathbf{a},\mathbf{b},\lambda,\mathbf{u}_1,\mathbf{u}_2,\mathbf{u}_3 \ldots \mid \mathbf{I}) d\mathbf{u}_1 d\mathbf{u}_2 d\mathbf{u}_3 \ldots. \qquad (12)$$

After applying Bayes' rule repeatedly and assuming the optical flow values are independent of one another,

$$P(\mathbf{a},\mathbf{b},\lambda \mid \mathbf{I}) = \iint \cdots \int P(\mathbf{a},\mathbf{b},\lambda \mid \mathbf{I},\mathbf{u}_1,\mathbf{u}_2,\mathbf{u}_3 \ldots) \prod_{i=1}^{n} P(\mathbf{u}_i \mid \mathbf{I}) d\mathbf{u}_i . \qquad (13)$$

The estimate of **a**, **b**, and λ is only based upon the optical flow values so $P(\mathbf{a},\mathbf{b},\lambda \mid \mathbf{I},\mathbf{u}_1,\mathbf{u}_2,\mathbf{u}_3 \ldots) = P(\mathbf{a},\mathbf{b},\lambda \mid \mathbf{u}_1,\mathbf{u}_2,\mathbf{u}_3 \ldots)$. Applying Bayes' rule several more times,

$$P(\mathbf{a},\mathbf{b},\lambda \mid \mathbf{I}) = \iint \cdots \int \frac{P(\mathbf{a},\mathbf{b},\lambda) P(\mathbf{u}_1,\mathbf{u}_2,\mathbf{u}_3,\ldots \mid \mathbf{a},\mathbf{b},\lambda)}{P(\mathbf{u}_1,\mathbf{u}_2,\mathbf{u}_3,\ldots)} \prod_{i=1}^{n} P(\mathbf{u}_i \mid \mathbf{I}) d\mathbf{u}_i . \quad (14)$$

Each of the optical flow values is independent of the other values:

$$P(\mathbf{a},\mathbf{b},\lambda \mid \mathbf{I}) = \iint \cdots \int P(\mathbf{a},\mathbf{b},\lambda) P(\mathbf{u}_1,\mathbf{u}_2,\mathbf{u}_3,\ldots \mid \mathbf{a},\mathbf{b},\lambda) \prod_{i=1}^{n} \frac{P(\mathbf{u}_i \mid \mathbf{I})}{P(\mathbf{u}_i)} d\mathbf{u}_i . \quad (15)$$

In this two-frame method, the depths of each object is fixed. The depths of the objects in the scene are independent of the camera motion meaning that $P(\mathbf{a},\mathbf{b},\lambda) = P(\mathbf{a},\mathbf{b}) P(\lambda)$.

The vector λ contains the depth at each feature point λ_i. The depth of each feature point is independent of the depth of the other points meaning that $P(\lambda)=\prod_{i=1}^{n}P(\lambda_i)$.

$$P(\mathbf{a},\mathbf{b},\lambda\,|\,\mathbf{I})=\iint\cdots\int P(\mathbf{a},\mathbf{b})P(\mathbf{u}_1,\mathbf{u}_2,\mathbf{u}_3,\ldots\,|\,\mathbf{a},\mathbf{b},\lambda)\prod_{i=1}^{n}\frac{P(\lambda_i)P(\mathbf{u}_i\,|\,\mathbf{I})}{P(\mathbf{u}_i)}d\mathbf{u}_i\ .(16)$$

The true optical flow vector at the image position $\mathbf{p}_i=\begin{bmatrix}x_i & y_i\end{bmatrix}^T$ can be written as a function of the translation $\mathbf{a}=\begin{bmatrix}a_x & a_y & a_z\end{bmatrix}^T$, the rotation $\mathbf{b}=\begin{bmatrix}b_x & b_y & b_z\end{bmatrix}^T$, and the inverse depth λ_i

$$\mathbf{U}_i(\mathbf{a},\mathbf{b},\lambda_i)=\begin{bmatrix}\left(-a_xf_o+x_ia_z\right)\lambda_i+b_x\dfrac{x_iy_i}{f_o}-b_y\left(\dfrac{x_i^2}{f_o}+f_o\right)+b_zy_i\\ \left(-a_yf_o+y_ia_z\right)\lambda_i+b_x\left(\dfrac{y_i^2}{f_o}+f_o\right)-b_y\dfrac{x_iy_i}{f_o}-b_zx_i\end{bmatrix}\qquad(17)$$

where f_o is the focal length of the camera [6]. There is only one possible optical flow value for any given camera motion and inverse depth so

$$P(\mathbf{u}_i\,|\,\mathbf{a},\mathbf{b},\lambda_i)=\begin{cases}1, & \mathbf{u}_i=\mathbf{U}_i(\mathbf{a},\mathbf{b},\lambda_i)\\ 0, & \mathbf{u}_i\neq\mathbf{U}_i(\mathbf{a},\mathbf{b},\lambda_i)\end{cases},\qquad(18)$$

$$P(\mathbf{u}_1,\mathbf{u}_2,\mathbf{u}_3,\ldots\,|\,\mathbf{a},\mathbf{b},\lambda_i)=\prod_{i=1}^{n}\begin{cases}1, & \mathbf{u}_i=\mathbf{U}_i(\mathbf{a},\mathbf{b},\lambda_i)\\ 0, & \mathbf{u}_i\neq\mathbf{U}_i(\mathbf{a},\mathbf{b},\lambda_i)\end{cases}.\qquad(19)$$

Equation (16) can therefore be rewritten as

$$P(\mathbf{a},\mathbf{b},\lambda\,|\,\mathbf{I})=P(\mathbf{a},\mathbf{b})\prod_{i=1}^{n}\frac{P(\lambda_i)P\left(\mathbf{u}_i(\mathbf{a},\mathbf{b},\lambda_i)\,|\,\mathbf{I}\right)}{P(\mathbf{u}_i(\mathbf{a},\mathbf{b},\lambda_i))}.\qquad(20)$$

The goal is to calculate the most probable translation, rotation, and inverse depth based on the image data. For every possible camera translation and rotation there is a set of depth values that are the most probable. The other less probable depth values can be ignored. So the most probable rotation and translation values are the values of **a** and **b** that maximize the probability

$$P(\mathbf{a},\mathbf{b} \mid \mathbf{I}) = P(\mathbf{a},\mathbf{b})\prod_{i=1}^{n} \max_{\lambda_i \geq 0} \frac{P(\lambda_i)P\left(\mathbf{u}_i(\mathbf{a},\mathbf{b},\lambda_i) \mid \mathbf{I}\right)}{P(\mathbf{u}_i(\mathbf{a},\mathbf{b},\lambda_i))}. \tag{21}$$

The depth and the inverse depth must be greater than zero (since it is impossible to see objects behind the camera). The prior distribution of the camera motion $P(\mathbf{a},\mathbf{b})$ is based upon any prior knowledge of how the camera moves. Many things may be known about the camera motion that may improve the estimate. For example, it may be known that the camera only travels in the forward or near-forward directions. This knowledge can be incorporated into the prior distribution and will provide a more accurate estimate. The prior depth distributions $P(\lambda_i)$ are based on any prior knowledge of how far away the objects are expected to be from the camera. $P(\mathbf{u}_i(\mathbf{a}, \mathbf{b}, \lambda_i) \mid \mathbf{I})$ is the optical flow probability distribution defined in Equation (6). The prior optical flow distribution $P(\mathbf{u}_i(\mathbf{a}, \mathbf{b}, \lambda_i))$ can be calculated using the prior distributions of the camera motion and the prior distribution of the depth using Equation (17).

The most probable translation and rotation vectors are found using a genetic algorithm. This is a very time-consuming process and it is the main disadvantage of this method, but this method is accurate when given sufficient time.

3.2 Two-Frame Gaussian Structure from Motion

If the optical flow probability distributions are Gaussian, a structure from motion algorithm can be developed that is much simpler computationally. In the first image, the i-th feature point is located on the image plane at the position $\mathbf{p}_i$. In the second image, the i-th feature point has moved to the position $\mathbf{p}_i'$. The optical flow between the two images is a random variable called $\mathbf{U}_i$, which has a mean value of $\mathbf{u}_i$ and a covariance matrix of Ω_i. The mean values and covariance matrices can be calculated using one of the methods described earlier. The optical flow relates the two positions by the equation $\mathbf{p}_i' = \mathbf{p}_i + \mathbf{U}_i$.

Each position $\mathbf{p}_i$ is located on the image plane. The distance from the optical center of the camera to the image plane is equal to the focal length of the camera, f_o. If a coordinate system is defined with the origin at the optical center of the camera and the z-axis pointing in the direction the camera is facing, then each vector $\mathbf{p}_i$ in this three-dimensional coordinate system can be written as $\mathbf{p}_i = [p_x \quad p_y \quad f_o]^T$. To make the mathematics of the problem simpler, the positions $\mathbf{p}_i$ and $\mathbf{p}_i'$ will be projected onto a unit sphere centered at the optical center of the camera. This projection is shown in Figure 1. The projection of the position $\mathbf{p}_i$ onto a unit sphere will be called $\mathbf{x}_i$. $\mathbf{x}_i$ and $\mathbf{x}_i'$ are calculated as

$$\mathbf{x}_i = \frac{\mathbf{p}_i}{\|\mathbf{p}_i\|}, \qquad \mathbf{x}_i' = \frac{\mathbf{p}_i'}{\|\mathbf{p}_i'\|} = \frac{\mathbf{p}_i + \mathbf{U}_i}{\|\mathbf{p}_i + \mathbf{U}_i\|} \quad . \tag{22}$$

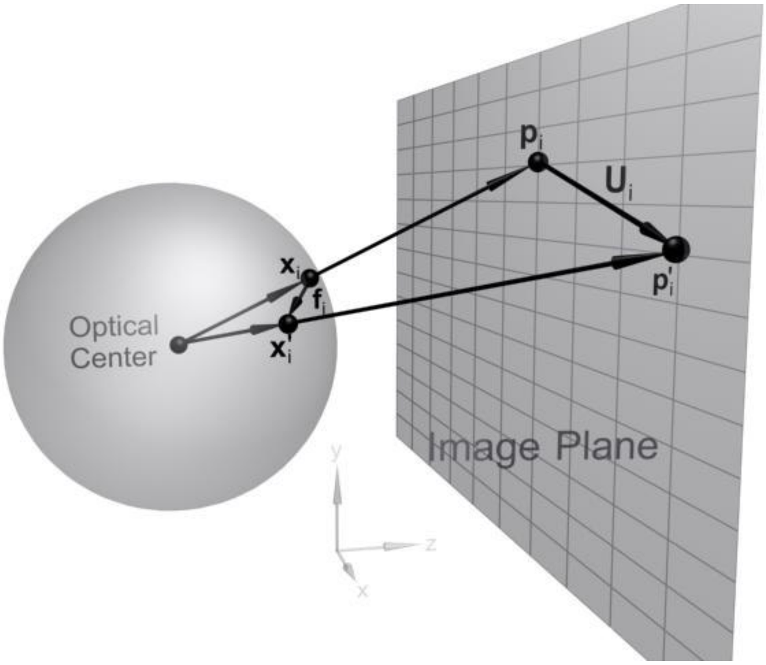

Figure 1: Diagram of the Projection onto a unit sphere

$\mathbf{x}_i'$ can be written as a function of $\mathbf{U}_i$. An approximate linear relationship between $\mathbf{U}_i$ and $\mathbf{x}_i'$ needs to be found. Because the random variable $\mathbf{U}_i$ is likely to be very close to its mean value of $\mathbf{u}_i$, $\mathbf{x}_i'(\mathbf{U}_i)$ can be approximated using a Taylor series as

$$\mathbf{x}_i'(\mathbf{U}_i) \approx \mathbf{x}_i'(\mathbf{u}_i) + \left.\frac{\partial \mathbf{x}_i'(\mathbf{U}_i)}{\partial \mathbf{U}_i}\right|_{\mathbf{U}_i=\mathbf{u}_i} (\mathbf{U}_i - \mathbf{u}_i)\,. \tag{23}$$

If the matrix $\mathbf{G}_i^*$ is defined as

$$\mathbf{G}_i^* = \begin{bmatrix} \frac{1}{\|\mathbf{p}_i+\mathbf{u}_i\|} - \frac{(p_x+u_x)^2}{\|\mathbf{p}_i+\mathbf{u}_i\|^3} & -\frac{(p_x+u_x)(p_y+u_y)}{\|\mathbf{p}_i+\mathbf{u}_i\|^3} \\ -\frac{(p_x+u_x)(p_y+u_y)}{\|\mathbf{p}_i+\mathbf{u}_i\|^3} & \frac{1}{\|\mathbf{p}_i+\mathbf{u}_i\|} - \frac{(p_y+u_y)^2}{\|\mathbf{p}_i+\mathbf{u}_i\|^3} \\ -\frac{f_o(p_x+u_x)}{\|\mathbf{p}_i+\mathbf{u}_i\|^3} & -\frac{f_o(p_y+u_y)}{\|\mathbf{p}_i+\mathbf{u}_i\|^3} \end{bmatrix} \tag{24}$$

and the vector $\mathbf{g}_i$ as

$$\mathbf{g}_i = \frac{\mathbf{p}_i + \mathbf{u}_i}{\left\|\mathbf{p}_i + \mathbf{u}_i\right\|} - \mathbf{G}_i^*\mathbf{u}_i, \tag{25}$$

then

$$\mathbf{x}_i' \approx \mathbf{G}_i^*\mathbf{U}_i + \mathbf{g}_i. \tag{26}$$

The motion of the feature points on the unit sphere will be represented by the vector $\mathbf{f}_i = \mathbf{x}_i' - \mathbf{x}_i$, which is approximated by

$$\mathbf{f}_i \approx \mathbf{G}_i^*\mathbf{U}_i + \mathbf{g}_i - \mathbf{x}_i. \tag{27}$$

A new vector $\mathbf{y}_i$ is defined as $\mathbf{y}_i = \mathbf{f}_i \times \mathbf{x}_i$. The "hat" operator will be used to indicate the skew-symmetric matrix that performs the cross-product between two vectors, so that $\hat{\mathbf{x}}\mathbf{y} = \mathbf{x} \times \mathbf{y}$. If the matrix $\mathbf{G}_i$ is defined as $\mathbf{G}_i = -\hat{\mathbf{x}}_i\mathbf{G}_i^*$, then

$$\mathbf{y}_i \approx \mathbf{G}_i\mathbf{U}_i - \hat{\mathbf{x}}_i\mathbf{g}_i. \tag{28}$$

The translational motion of the camera will be represented by a unit vector **a** pointed in the direction of translation. The rotation of the camera will be represented by a vector **b**, where the direction of **b** is the axis about which the camera is rotated and the magnitude ||**b**|| is the amount the camera is rotated in radians. The inverse depth of the *i*-th feature point is λ_i. The depth can only be calculated to an unknown scale factor. The scale factor is arbitrarily chosen to be the distance traveled by the camera. So λ_i^{-1} will be the distance of the *i*-th feature point from the camera divided by the distance traveled by the camera.

3.2.1 Cost Function

A slightly different camera model from the camera model used in Equation (17) for non-Gaussian noise will be used based on spherical projection. The following approximate relationship exists between each of the terms that have been defined [23]

$$\mathbf{y}_{\mathrm{i}} + \hat{\mathbf{x}}_i \mathbf{a} \lambda_i - \hat{\mathbf{x}}_i^2 \mathbf{b} = 0 . \tag{29}$$

After substituting $\mathbf{G}_i \mathbf{U}_i - \hat{\mathbf{x}}_i \mathbf{g}_i$ in for $\mathbf{y}_i$,

$$\mathbf{G}_{\mathrm{i}} \mathbf{U}_i - \hat{\mathbf{x}}_i \mathbf{g}_i + \hat{\mathbf{x}}_i \mathbf{a} \lambda_i - \hat{\mathbf{x}}_i^2 \mathbf{b} = 0 . \tag{30}$$

Both sides of the equation are multiplied by the left pseudo-inverse of $\mathbf{G}_{\mathrm{i}}$, which is $\left(\mathbf{G}_i^T \mathbf{G}_i\right)^{-1} \mathbf{G}_i^T$

$$\mathbf{U}_i + (\mathbf{G}_i^T \mathbf{G}_i)^{-1} \mathbf{G}_i^T (-\hat{\mathbf{x}}_i \mathbf{g}_i + \hat{\mathbf{x}}_i \mathbf{a} \lambda_i - \hat{\mathbf{x}}_i^2 \mathbf{b}) = 0 . \tag{31}$$

Equation (31) is only true when there is no noise in the optical flow estimates. In practice, there is always noise in the optical flow estimates. The value of $\mathbf{U}_{\mathrm{i}}$ can be calculated for any given **a**, **b**, and λ_i values from equation (31). This $\mathbf{U}_{\mathrm{i}}$ value has a known probability based upon the optical flow probability distributions. From the probability distributions a probability can be calculated for every possible combination of **a**, **b**, and λ_i values. The goal is to find the **a**, **b**, and λ_i values that are the most probable. Since the noise in $\mathbf{U}_{\mathrm{i}}$ is not identically distributed and is not white, the proper measure to minimize is the covariance weighted squared-error or the Mahalanobis distance. The cost function r(**a**,**b**, λ) is defined as

$$r(\mathbf{a}, \mathbf{b}, \lambda) = \sum_{i=1}^{n} \left\| \mathbf{u}_i + (\mathbf{G}_i^T \mathbf{G}_i)^{-1} \mathbf{G}_i^T (-\hat{\mathbf{x}}_i \mathbf{g}_i + \hat{\mathbf{x}}_i \mathbf{a} \lambda_i - \hat{\mathbf{x}}_i^2 \mathbf{b}) \right\|_{\Omega_i^{-1}}^2 , \tag{32}$$

where the weighted-norm $\left\| \cdot \right\|_{\Omega_i^{-1}}$ is given by $\left\| \mathbf{x} \right\|_{\Omega_i^{-1}}^2 = \mathbf{x}^T \Omega_i^{-1} \mathbf{x}$. By minimizing this cost function, the **a**, **b**, and λ_i values that are the most probable will be found. By expanding the weighted-norm and rearranging a few terms, it is found that

$$r(\mathbf{a}, \mathbf{b}, \lambda) = \sum_{i=1}^{n} \left(\mathbf{y}_i + \hat{\mathbf{x}}_i \mathbf{a} \lambda_i - \hat{\mathbf{x}}_i^2 \mathbf{b} \right)^T \mathbf{G}_i (\mathbf{G}_i^T \mathbf{G}_i)^{-1} \Omega_i^{-1} (\mathbf{G}_i^T \mathbf{G}_i)^{-1} \mathbf{G}_i^T \left(\mathbf{y}_i + \hat{\mathbf{x}}_i \mathbf{a} \lambda_i - \hat{\mathbf{x}}_i^2 \mathbf{b} \right) . \tag{33}$$

By introducing a new weighting matrix $\mathbf{W}_i = \mathbf{G}_i(\mathbf{G}_i^T\mathbf{G}_i)^{-1}\Omega_i^{-1}(\mathbf{G}_i^T\mathbf{G}_i)^{-1}\mathbf{G}_i^T$ Equation (33) is rewritten as

$$r(\mathbf{a},\mathbf{b},\lambda) = \sum_{i=1}^{n} \left\| \mathbf{y}_i + \hat{\mathbf{x}}_i\mathbf{a}\lambda_i - \hat{\mathbf{x}}_i^2\mathbf{b} \right\|_{\mathbf{W}_i}^2 . \tag{34}$$

The key difference between previous work in SFM and the new method is the addition of the weighting matrix **W**$_i$. The weighting matrix gives the more accurate data more weight so it is considered to be more valuable. With the addition of this new matrix, the optimal solutions for the translation, rotation, and depth all change to incorporate the new weights.

3.2.2 Depth and Rotation Estimation

In order to minimize the cost function $r(\mathbf{a},\mathbf{b},\lambda)$ in (34), it will be necessary to use generalized least-squares repeatedly. For any two matrices **A** and **W** and any vector **x**, the vector **c** that minimizes the weighted-norm $\left\|\mathbf{x} - \mathbf{A}\mathbf{c}\right\|_{\mathbf{W}}$ is found using generalized least-squares to be $\mathbf{c} = \left(\mathbf{A}^T\mathbf{W}\mathbf{A}\right)^{-1}\mathbf{A}^T\mathbf{W}\mathbf{x}$. From this method, the optimal value for λ_i for any arbitrary value of **a** and **b** is found to be

$$\lambda_i = \frac{-\mathbf{a}^T\hat{\mathbf{x}}_i\mathbf{W}_i(\mathbf{y}_i - \hat{\mathbf{x}}_i^2\mathbf{b})}{\mathbf{a}^T\hat{\mathbf{x}}_i\mathbf{W}_i\hat{\mathbf{x}}_i\mathbf{a}} . \tag{35}$$

Since this solution is the best possible solution for any **a** and **b**, it can be placed back into equation (34), so that the values of **a** and **b** that minimize the cost function

$$r_o(\mathbf{a},\mathbf{b}) = \sum_{i=1}^{n} \left\| \mathbf{y}_i - \frac{\hat{\mathbf{x}}_i\mathbf{a}\mathbf{a}^T\hat{\mathbf{x}}_i\mathbf{W}_i(\mathbf{y}_i - \hat{\mathbf{x}}_i^2\mathbf{b})}{\mathbf{a}^T\hat{\mathbf{x}}_i\mathbf{W}_i\hat{\mathbf{x}}_i\mathbf{a}} - \hat{\mathbf{x}}_i^2\mathbf{b} \right\|_{\mathbf{W}_i}^2 \tag{36}$$

are the same values of **a** and **b** that minimize the cost function r(**a**, **b**, λ).

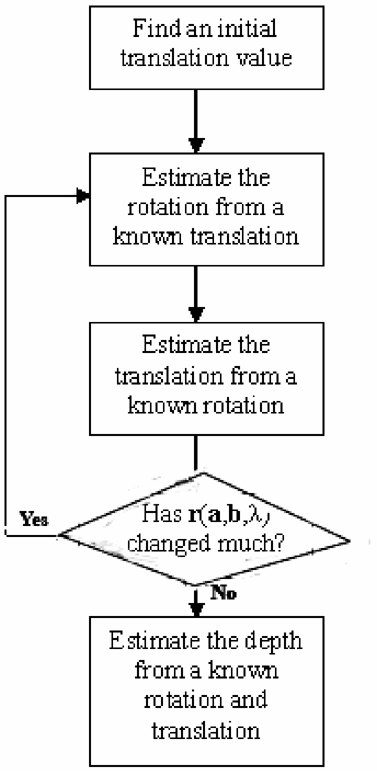
Find an initial translation value
Estimate the rotation from a known translation
Estimate the translation from a known rotation
Has r(a,b,λ) changed much?
Yes
No
Estimate the depth from a known rotation and translation

Figure 2: Two-Frame Flowchart

Ideally, a solution for **a** and **b** would be found simultaneously. Unfortunately, the only closed form solution for **a** that has been found requires a known estimate of **b** and likewise the closed form solution for **b** requires a known estimate value for **a**. As outlined in the flowchart in Figure 2, the plan will be to pick an initial value for **a** and find the optimal value for **b** based upon that **a** value. Next, the optimal value for **a** based upon the calculated **b** value is found. Then the optimal value for **b** based upon the new **a** value is found. This process repeats itself iteratively until after a few iterations the cost function does not change significantly.

First, the solution for **b** based upon a known **a** will be examined. The cost function in Equation (36) can be simplified by defining a new matrix **Q**i as $\mathbf{Q}_i = \mathbf{I} - \frac{\hat{\mathbf{x}}_i \mathbf{a}\mathbf{a}^T \hat{\mathbf{x}}_i \mathbf{W}_i}{\mathbf{a}^T \hat{\mathbf{x}}_i \mathbf{W}_i \hat{\mathbf{x}}_i \mathbf{a}}$,

$$r_o(\mathbf{a},\mathbf{b}) = \sum_{i=1}^{n} \left\| \mathbf{Q}_i \left(\mathbf{y}_i - \hat{\mathbf{x}}_i^2 \mathbf{b} \right) \right\|_{\mathbf{W}_i}^2 . \tag{37}$$

This equation can be modified in much the same way that Equation (32) was modified to produce Equation (34)

$$r_o(\mathbf{a},\mathbf{b}) = \sum_{i=1}^{n} \left\| \mathbf{y}_i - \hat{\mathbf{x}}_i^2 \mathbf{b} \right\|_{\mathbf{Q}_i^T \mathbf{W}_i \mathbf{Q}_i}^2 . \tag{38}$$

The solution for **b** is found using generalized least-squares to be

$$\mathbf{b} = \left(\sum_{i=1}^{n} \hat{\mathbf{x}}_i^2 \mathbf{Q}_i^T \mathbf{W}_i \mathbf{Q}_i \hat{\mathbf{x}}_i^2 \right)^{-1} \sum_{i=1}^{n} \hat{\mathbf{x}}_i^2 \mathbf{Q}_i^T \mathbf{W}_i \mathbf{Q}_i \mathbf{y}_i . \tag{39}$$

3.2.3 Translation Estimation

For the moment, let us consider a different way of weighting the norms in the cost function. Let us create a new cost function $r_1(\mathbf{a},\mathbf{b},\lambda) = \sum_{i=1}^{n} \left\| \mathbf{y}_i + \hat{\mathbf{x}}_i \mathbf{a} \lambda_i - \hat{\mathbf{x}}_i^2 \mathbf{b} \right\|_{w_i}^2$, which uses the new weight $w_i = \left\| \hat{\mathbf{x}}_i \mathbf{a} \right\|^2$. This cost function no longer uses the optical flow distributions, because the covariance matrices Ωi are no longer used. This new cost function will be used to find a solution that does not use optical flow distributions, but then the result will be extended to obtain a method that does use them. Much of this derivation is taken from [23], until the part where it is extended to better handle the noise.

The optimal value for λ_i for arbitrary translation and rotation vectors using the new cost function is

$$\lambda_i = \frac{-\mathbf{a}^T \hat{\mathbf{x}}_i (\mathbf{y}_i - \hat{\mathbf{x}}_i^2 \mathbf{b})}{\mathbf{a}^T \hat{\mathbf{x}}_i^2 \mathbf{a}} . \tag{40}$$

This solution is plugged back into the cost function so that the cost function no longer depends on λ_i:

$$r_1(\mathbf{a}, \mathbf{b}) = \sum_{i=1}^{n} \left\| (\hat{\mathbf{x}}_i \mathbf{a})^{\perp} (\mathbf{y}_i - \hat{\mathbf{x}}_i^2 \mathbf{b}) \right\|_{w_i}^2 . \tag{41}$$

where $\mathbf{x}^{\perp}$ represents the matrix that projects another vector onto the plane perpendicular to $\mathbf{x}$. The norm of $(\hat{\mathbf{x}}_i \mathbf{a})^{\perp} \mathbf{y}_i$ is equal to the norm of $\frac{\hat{\mathbf{x}}_i \mathbf{a}}{\left\| \hat{\mathbf{x}}_i \mathbf{a} \right\|} \times \mathbf{y}_i$

$$r_1(\mathbf{a}, \mathbf{b}) = \sum_{i=1}^{n} \frac{\left\| \hat{\mathbf{x}}_i \mathbf{a} \times (\mathbf{y}_i - \hat{\mathbf{x}}_i^2 \mathbf{b}) \right\|_{w_i}^2}{\left\| \hat{\mathbf{x}}_i \mathbf{a} \right\|^2} . \tag{42}$$

$\mathbf{x}_i$ points out from the center of the unit sphere and $\mathbf{y}_i$ lies on the surface of the sphere tangent to $\mathbf{x}_i$, so $\mathbf{x}_i$ and $\mathbf{y}_i - \hat{\mathbf{x}}_i^2 \mathbf{b}$ are orthogonal. This equation can be simplified by recognizing that $(\hat{\mathbf{x}}_i \mathbf{a}) \times = \mathbf{a}\mathbf{x}_i^T - \mathbf{x}_i \mathbf{a}^T$ and by removing the weights from the weighted-norm

$$\begin{aligned} r_1(\mathbf{a}, \mathbf{b}) &= \sum_{i=1}^{n} \frac{\left\| \mathbf{x}_i \mathbf{a}^T (\mathbf{y}_i - \hat{\mathbf{x}}_i^2 \mathbf{b}) \right\|_{w_i}^2}{\left\| \hat{\mathbf{x}}_i \mathbf{a} \right\|^2} . \\ &= \sum_{i=1}^{n} \left\| \mathbf{x}_i \mathbf{a}^T (\mathbf{y}_i - \hat{\mathbf{x}}_i^2 \mathbf{b}) \right\|^2 \end{aligned} \tag{43}$$

Since $\mathbf{x}_i$ is a unit vector and $\mathbf{a}^T (\mathbf{y}_i - \hat{\mathbf{x}}_i^2 \mathbf{b})$ is a scalar value,

$$
\begin{aligned}
r_1(\mathbf{a},\mathbf{b}) &= \sum_{i=1}^{n}\left\|\mathbf{a}^T(\mathbf{y}_i - \hat{\mathbf{x}}_i^2\mathbf{b})\right\|^2 \\
&= \sum_{i=1}^{n}\mathbf{a}^T\left\|(\mathbf{y}_i - \hat{\mathbf{x}}_i^2\mathbf{b})\right\|^2\mathbf{a} \\
&= \mathbf{a}^T\left(\sum_{i=1}^{n}(\mathbf{y}_i - \hat{\mathbf{x}}_i^2\mathbf{b})(\mathbf{y}_i - \hat{\mathbf{x}}_i^2\mathbf{b})^T\right)\mathbf{a}
\end{aligned}
\quad . \qquad (44)
$$

The solution for **a** that minimizes this cost function is the minimum-eigenvalue eigenvector of the matrix $\sum_{i=1}^{n}(\mathbf{y}_i - \hat{\mathbf{x}}_i^2\mathbf{b})(\mathbf{y}_i - \hat{\mathbf{x}}_i^2\mathbf{b})^T$.

This result can be extended so that it can handle noise calculated from optical flow distributions that are not identically-distributed. The amount noise in the value $\mathbf{a}^T(\mathbf{y}_i - \hat{\mathbf{x}}_i^2\mathbf{b}) = \mathbf{a}^T(\mathbf{G}_i\mathbf{u}_i - \hat{\mathbf{x}}_i\mathbf{g}_i - \hat{\mathbf{x}}_i^2\mathbf{b})$ needs to be calculated, so that those values with less noise can be given greater weight. The covariance matrix of $\mathbf{u}_i$ is equal to Ω_i and so the covariance matrix of $\mathbf{a}^T(\mathbf{G}_i\mathbf{u}_i - \hat{\mathbf{x}}_i\mathbf{g}_i - \hat{\mathbf{x}}_i^2\mathbf{b})$ is equal to $\mathbf{a}^T\mathbf{G}_i\Omega_i\mathbf{G}_i^T\mathbf{a}$. By modifying the cost function in (44), so that the more accurate values are given greater weight, a new cost function is formed as

$$
r_2(\mathbf{a}_k,\mathbf{b}) = \sum_{i=1}^{n}\left\|\mathbf{a}^T(\mathbf{y}_i - \hat{\mathbf{x}}_i^2\mathbf{b})\right\|^2_{\left(\mathbf{a}_{k-1}\mathbf{G}_i\Omega_i\mathbf{G}_i^T\mathbf{a}_{k-1}^T\right)^{-1}} . \qquad (45)
$$

There is a small dilemma. The amount of noise depends on the parameter that we are trying to estimate **a**. The estimation of **a** depends upon the amount of noise and the amount of noise depends upon the estimation of **a**. There is a simple solution to this dilemma. An iterative method is already required to calculate **a** and **b**. The solution is to simply use the value of **a** calculated from the previous iteration to estimate the amount of noise in the current iteration. To clarify the notation, $\mathbf{a}_k$ will be the value of **a** on the *k*-th iteration. The amount of noise on the k-th iteration is equal to $\mathbf{a}_{k-1}\mathbf{G}_i\Omega_i\mathbf{G}_i^T\mathbf{a}_{k-1}^T$. So equation (45) should read

$$r_2(\mathbf{a}_k,\mathbf{b}) = \sum_{i=1}^{n}\left\|\mathbf{a}_k^T(\mathbf{y}_i - \hat{\mathbf{x}}_i^2\mathbf{b})\right\|^2_{\left(\mathbf{a}_{k-1}\mathbf{G}_i\Omega_i\mathbf{G}_i^T\mathbf{a}_{k-1}^T\right)^{-1}}, \tag{46}$$

$$r_2(\mathbf{a}_k,\mathbf{b}) = \sum_{i=1}^{n}\mathbf{a}_k^T\left\|(\mathbf{y}_i - \hat{\mathbf{x}}_i^2\mathbf{b})\right\|^2_{\left(\mathbf{a}_{k-1}\mathbf{G}_i\Omega_i\mathbf{G}_i^T\mathbf{a}_{k-1}^T\right)^{-1}}\mathbf{a}_k, \tag{47}$$

$$r_2(\mathbf{a}_k,\mathbf{b}) = \mathbf{a}_k\left(\sum_{i=1}^{n}\frac{(\mathbf{y}_i - \hat{\mathbf{x}}_i^2\mathbf{b})(\mathbf{y}_i - \hat{\mathbf{x}}_i^2\mathbf{b})^T}{\mathbf{a}_{k-1}\mathbf{G}_i\Omega_i\mathbf{G}_i^T\mathbf{a}_{k-1}^T}\right)\mathbf{a}_k^T. \tag{48}$$

The solution for $\mathbf{a}_k$ that minimizes the new cost function $r_2(\mathbf{a}_k,\mathbf{b})$ is the minimum-eigenvalue eigenvector of the matrix $\sum_{i=1}^{n}\frac{(\mathbf{y}_i - \hat{\mathbf{x}}_i^2\mathbf{b})(\mathbf{y}_i - \hat{\mathbf{x}}_i^2\mathbf{b})^T}{\mathbf{a}_{k-1}\mathbf{G}_i\Omega_i\mathbf{G}_i^T\mathbf{a}_{k-1}^T}$.

3.3 Multi-Frame Gaussian Structure from Motion

Multi-frame SFM is quite similar to two-frame SFM. All of the vectors and matrices remain essentially the same except they are larger to accommodate multiple frames. Let us assume that there are a total of N+1 frames. $\mathbf{U}_{it}$ is the optical flow value of the i-th feature point at time t. $\mathbf{U}_{i1}$ is the optical flow from the first frame to the second frame, $\mathbf{U}_{i2}$ is the optical flow from the second frame to the third frame, and so on. $\mathbf{U}_i$ will now represent a vector of size $2N$ that contains all of the optical flow values $\mathbf{U}_{i1}$ through $\mathbf{U}_{iN}$. $\mathbf{x}_{it}$ is the position of the i-th feature point at time t projected onto a unit sphere according to Equation (22). $\mathbf{f}_{it}$ is the projection of $\mathbf{U}_{it}$ onto a unit sphere according to Equation (27) at time t and $\mathbf{y}_{it}$ is defined as $\mathbf{y}_{it} = \mathbf{f}_{it} \times \mathbf{x}_{it}$. $\mathbf{y}_i$ is a $3N$ dimensional vector that contains the vectors $\mathbf{y}_{i1}$ through $\mathbf{y}_{iN}$. $\mathbf{G}_{it}$ is the matrix defined according to Equation (28). All of the matrices $\mathbf{G}_{i1}$ through $\mathbf{G}_{iN}$ can be combined to form a $3N \times 2N$ block diagonal matrix $\mathbf{G}_i$ so that the relationship $\mathbf{y}_i = \mathbf{G}_i\mathbf{U}_i + \mathbf{g}_i$ holds for some $\mathbf{g}_i$. Nothing substantial has changed from the two-frame method except the vectors and matrices are larger.

The translational motion of the camera at time t will be represented by the vector $\mathbf{a}_t$ and the rotational motion will be represented by $\mathbf{b}_t$. All of the $\mathbf{a}_t$ and $\mathbf{b}_t$ vectors can be combined into the larger vector $\mathbf{a}$ and $\mathbf{b}$. The depth still can only be calculated to an unknown scale factor. The first translation vector $\mathbf{a}_1$ will be arbitrarily choosen to be a unit vector, so that now $\|\mathbf{a}_t\|$ is equal to the speed of the camera at time t divided by the speed of the camera at the first frame and λ_i^{-1} is equal to the distance from the camera to the i-th feature point divided by distance traveled by the camera from between the first and the second frames.

3.3.1 Constant Depth

The depth of each feature point is not constant over time, but often the depth only changes slightly. The depth will change slightly as the camera approaches or recedes from the objects in the scene. However, if most of the objects are not too close to the camera, it can be assumed that the depth is constant over time without significantly affecting the performance of the algorithm. First, an algorithm that assumes the depth is constant will be presented, and then a more difficult method that does not assume constant depth will be considered.

The relationship between the optical flow values and the camera motion is the same for multiple frames as it is for two frames. By adding time indices to Equation (29), it can be rewritten as

$$\mathbf{y}_{\mathrm{it}} + \hat{\mathbf{x}}_{it}\mathbf{a}_t\lambda_{it} - \hat{\mathbf{x}}_{it}^2\mathbf{b}_t = 0\,. \qquad (49)$$

If $\hat{\mathbf{x}}_i$ is a $3N \times 3N$ block diagonal matrix whose blocks are the matrices $\hat{\mathbf{x}}_{i1}$ through $\hat{\mathbf{x}}_{iN}$ then

$$\mathbf{y}_{\mathrm{i}} + \hat{\mathbf{x}}_i \mathbf{a} \lambda_i - \hat{\mathbf{x}}_i^2 \mathbf{b} = 0, \tag{50}$$

which is exactly the same as Equation (29) except each of the vectors and matrices are larger than they were in Equation (29). Since this equation is the identical, the solution for the rotation for a given translation can be derived in the same way it was in Equations (29) through (39) as

$$\mathbf{b} = \left(\sum_{i=1}^{n} \hat{\mathbf{x}}_i^2 \mathbf{Q}_i^T \mathbf{W}_i \mathbf{Q}_i \hat{\mathbf{x}}_i^2 \right)^{-1} \sum_{i=1}^{n} \hat{\mathbf{x}}_i^2 \mathbf{Q}_i^T \mathbf{W}_i \mathbf{Q}_i \mathbf{y}_i \,. \tag{51}$$

The depth estimation is also identical:

$$\lambda_i = \frac{-\mathbf{a}^T \hat{\mathbf{x}}_i \mathbf{W}_i (\mathbf{y}_i - \hat{\mathbf{x}}_i^2 \mathbf{b})}{\mathbf{a}^T \hat{\mathbf{x}}_i \mathbf{W}_i \hat{\mathbf{x}}_i \mathbf{a}}. \tag{52}$$

Like the two-frame SFM method, the rotation and depth are estimated from a known translation. However, instead of estimation the translation from a known rotation, the translation will be estimated from known depth values. The reason for this change between two frames and multiple frames is that in the two-frame case the depth values are highly unreliable. Each depth value is only calculated from a single optical flow value. If that one optical flow value is inaccurate, the depth value will also be inaccurate. What makes multi-frame SFM so appealing is that all of the optical flow values from many frames can be used to produce a fairly accurate depth value.

As outlined in Figure 3, the multi-frame SFM algorithm works by first finding an initial estimate of the translation according to Equation (48). This initial translation

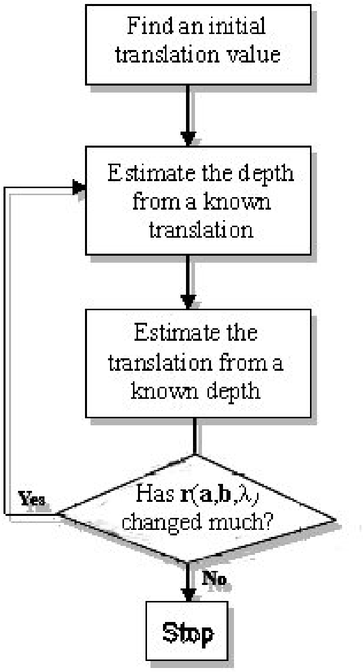
Find an initial translation value
Estimate the depth from a known translation
Estimate the translation from a known depth
Has r(a,b,λ) changed much?
Yes
No
Stop

Figure 3: Multi-Frame Flowchart

estimate is used to calculate the optimal rotation and depth of each feature point from Equations (51) and (52). The depth then is used to calculate the optimal translation and rotation. Then the translation is used to calculate the optimal rotation and depth. This process repeats until a fairly good estimate of the translation, rotation, and depth is obtained.

Notice that the rotation is recalculated in each step. There is a good reason for this. An alternative worth considering is to calculate the depth from a known rotation and translation. However, this is the wrong approach. The best approach would be to solve all three parameters simultaneously and use no known values. The next best alternative is to use one known parameter to solve for the other two parameters. The worst approach would be to use two parameters to solve for the remaining parameter.

The only remaining step is to find the optimal value for translation and rotation from known depth values. The vector **d** and the matrix $\mathbf{P}_i$ are defined as

$$\mathbf{d} = \begin{bmatrix} \mathbf{a} \\ \mathbf{b} \end{bmatrix}, \qquad \mathbf{P}_i = \left[\hat{\mathbf{x}}_i \lambda_i \quad -\hat{\mathbf{x}}_i^2\right], \tag{53}$$

so that the weighted-norm in (34) can be written as $\sum_{i=1}^{n} \left\| \mathbf{y}_i + \mathbf{P}_i \mathbf{d} \right\|_{\mathbf{W}_i}^2$ and the solution for the combined translation and rotation vector **d** is found to be

$$\mathbf{d} = -\left(\sum_{i=1}^{n} \mathbf{P}_i^T \mathbf{W}_i \mathbf{P}_i \right)^{-1} \sum_{i=1}^{n} \mathbf{P}_i^T \mathbf{W}_i \mathbf{y}_i \ . \tag{54}$$

3.3.2 A Better Depth Model

In reality, the inverse depths of the feature points λ_i are not constant over time. The depths will change as the camera approaches or recedes from the objects in the scene. The depth of the *i*-th feature point at time *t* will be called λ_{it}. A coordinate system can be defined around the initial position and orientation of the camera. The origin of this coordinate system is located at the position of the camera on the first frame, the *z*-axis is the direction the camera is facing at the first frame, and the *x*-axis as the horizontal direction of the camera in the first frame. In this three-dimensional coordinate system, the *i*-th feature point is located at the position $\mathbf{x}_{i1}\lambda_{i1}^{-1}$. At time *t*, the camera will be located at the position $\mathbf{v}_t = \sum_{\tau=1}^{t} \mathbf{a}_\tau$. (This equation is not entirely correct because it ignores the rotation of the camera. Each of the translation vectors should be rotated about each of the rotation vectors from the frames that precede it.) The distance from the camera to the *i*-th feature point at time *t* is equal to $\left\| \mathbf{x}_{i1}\lambda_{i1}^{-1} - \mathbf{v}_t \right\|$ and so λ_{it} can be written as a function of the inverse depth at the first frame λ_{i1}

$$\lambda_{it}(\lambda_{i1}) = \frac{1}{\left\| \mathbf{x}_{i1}\lambda_{i1}^{-1} - \mathbf{v}_t \right\|} . \tag{55}$$

Since the depth at any future time can be calculated from the depth on the first frame then it is only necessary to solve for the depth at the first frame. The other depth values will immediately follow. Unfortunately the relationship between λ_{i1} and λ_{it} is nonlinear. To avoid nonlinear estimation, a Taylor series approximation is used. Let λ_o be a value near the correct value of λ_{i1}. In practice, λ_o will simply be the value of λ_{i1} taken from the previous iteration. Using a Taylor series and neglecting the higher order terms λ_{it} is approximated as

$$\lambda_{it}(\lambda_{i1}) \approx \lambda_{it}(\lambda_o) + \frac{\partial \lambda_{it}(\lambda_{i1})}{\partial \lambda_{i1}} (\lambda_{i1} - \lambda_o)$$

$$\lambda_{it}(\lambda_{i1}) \approx \frac{1}{\left\| \mathbf{x}_{i1}\lambda_0^{-1} - \mathbf{v}_t \right\|} + \frac{\lambda_o^{-3} - \mathbf{v}_t^T \mathbf{x}_{i1} \lambda_o^{-2}}{(\lambda_o^{-2} - 2\mathbf{v}_t^T \mathbf{x}_{i1} \lambda_o^{-1} + \mathbf{v}_t^T \mathbf{v}_t)^{3/2}} (\lambda_{i1} - \lambda_o) . \tag{56}$$

If the vectors $\mathbf{m}_i$, λ_i, and $\mathbf{q}_i$ are defined as

$$\mathbf{m}_i = \begin{bmatrix} \dfrac{\lambda_o^{-3} - \mathbf{v}_1^T \mathbf{x}_{i1} \lambda_o^{-2}}{(\lambda_o^{-2} - 2\mathbf{v}_1^T \mathbf{x}_{i1} \lambda_o^{-1} + \mathbf{v}_1^T \mathbf{v}_1)^{3/2}} \\ \dfrac{\lambda_o^{-3} - \mathbf{v}_2^T \mathbf{x}_{i1} \lambda_o^{-2}}{(\lambda_o^{-2} - 2\mathbf{v}_2^T \mathbf{x}_{i1} \lambda_o^{-1} + \mathbf{v}_2^T \mathbf{v}_2)^{3/2}} \\ \vdots \\ \\ \end{bmatrix}, \qquad \lambda_i = \begin{bmatrix} \lambda_{i1} \\ \lambda_{i2} \\ \vdots \end{bmatrix}, \tag{57}$$

$$\mathbf{q}_i = \begin{bmatrix} \dfrac{1}{\left\| \mathbf{x}_{i1}\lambda_o^{-1} - \mathbf{v}_1 \right\|} - \dfrac{\lambda_o^{-2} - \mathbf{v}_1^T \mathbf{x}_{i1} \lambda_o^{-1}}{(\lambda_o^{-2} - 2\mathbf{v}_1^T \mathbf{x}_{i1} \lambda_o^{-1} + \mathbf{v}_1^T \mathbf{v}_1)^{3/2}} \\ \dfrac{1}{\left\| \mathbf{x}_{i1}\lambda_o^{-1} - \mathbf{v}_2 \right\|} - \dfrac{\lambda_o^{-2} - \mathbf{v}_2^T \mathbf{x}_{i1} \lambda_o^{-1}}{(\lambda_o^{-2} - 2\mathbf{v}_2^T \mathbf{x}_{i1} \lambda_o^{-1} + \mathbf{v}_2^T \mathbf{v}_2)^{3/2}} \\ \vdots \\ \\ \end{bmatrix},$$

then the vector λ$_i$ that contains the inverse depth at each time step is approximately equal to

$$\lambda_i \approx \mathbf{m}_i \lambda_{i1} + \mathbf{q}_i\,. \tag{58}$$

3.3.3 Translation, Rotation, and Depth Estimation

Let $\mathbf{a}'$ be a 3N × N block diagonal comprised of the vectors $\mathbf{a}_1$ through $\mathbf{a}_N$. Since λ$_i$ is no longer a scalar quantity, Equation (50) is rewritten as

$$\mathbf{y}_\mathrm{i} + \hat{\mathbf{x}}_i \mathbf{a}' \lambda_i - \hat{\mathbf{x}}_i^2 \mathbf{b} = 0\,. \tag{59}$$

Substituting in the new value for λ$_i$,

$$\mathbf{y}_\mathrm{i} + \hat{\mathbf{x}}_i \mathbf{a}'(\mathbf{m}_i \lambda_{i1} + \mathbf{q}_i) - \hat{\mathbf{x}}_i^2 \mathbf{b} = 0\,.$$

The solution for the depth is found using generalized least squares to be

$$\lambda_i = \frac{-\mathbf{m}_i^T \mathbf{a}'^T \hat{\mathbf{x}}_i \mathbf{W}_i (\mathbf{y}_i + \hat{\mathbf{x}}_i \mathbf{a}' \mathbf{q}_i - \hat{\mathbf{x}}_i^2 \mathbf{b})}{\mathbf{m}_i^T \mathbf{a}'^T \hat{\mathbf{x}}_i \mathbf{W}_i \hat{\mathbf{x}}_i \mathbf{a}' \mathbf{m}_i}\,. \tag{60}$$

Plugging this solution back in the cost function and solving for the rotation using generalized least squares, it is found that

$$\mathbf{b} = \left(\sum_{i=1}^{n} \hat{\mathbf{x}}_i^2 \mathbf{Q}_i^T \mathbf{W}_i \mathbf{Q}_i \hat{\mathbf{x}}_i^2 \right)^{-1} \sum_{i=1}^{n} \hat{\mathbf{x}}_i^2 \mathbf{Q}_i^T \mathbf{W}_i \mathbf{Q}_i (\mathbf{y}_i + \hat{\mathbf{x}}_i \mathbf{a}' \mathbf{q}_i) \tag{61}$$

where $\mathbf{Q}_i$ is redefined as $\mathbf{Q}_i = \mathbf{I} - \dfrac{\hat{\mathbf{x}}_i \mathbf{a}' \mathbf{m}_i \mathbf{m}_i^T \mathbf{a}'^T \hat{\mathbf{x}}_i \mathbf{W}_i}{\mathbf{m}_i^T \mathbf{a}'^T \hat{\mathbf{x}}_i \mathbf{W}_i \hat{\mathbf{x}}_i \mathbf{a}' \mathbf{m}_i}$.

The only missing piece is a method of finding the optimal value for translation and rotation from known depth values. The matrix λ_i' and the matrix $\mathbf{P}_\mathrm{i}$ are defined as

$$\lambda_i' = \begin{bmatrix} \lambda_{i1} & 0 & \dots \\ \lambda_{i1} & 0 & \dots \\ \lambda_{i1} & 0 & \dots \\ 0 & \lambda_{i2} & \dots \\ 0 & \lambda_{i2} & \dots \\ 0 & \lambda_{i2} & \dots \\ \vdots & \vdots & \ddots \end{bmatrix}, \qquad \mathbf{P}_i = [\hat{\mathbf{x}}_i \lambda_i' \quad -\hat{\mathbf{x}}_i^2]. \tag{62}$$

Similar to equation (54), the solution for the combined translation and rotation vector **d** is found to be

$$\mathbf{d} = -\left(\sum_{i=1}^{n} \mathbf{P}_i^T \mathbf{W}_i \mathbf{P}_i \right)^{-1} \sum_{i=1}^{n} \mathbf{P}_i^T \mathbf{W}_i \mathbf{y}_i \ . \tag{63}$$

3.3.4 Smoothness Constraint

A better estimate can be obtained with prior knowledge of how the camera typically moves or of what is expected to be in the scene that the camera will be observing. For example, it may be known that because the camera is attached to an airplane and the camera can only travel in the forward or near-forward directions. In other applications, the camera may not be able to rotate very quickly and the rotation will be small. All of this knowledge can be used to improve the estimate. In a typical application, it can be assumed that the camera will not change directions quickly relative to frame rate. It can be assumed that both the rotation and translation of the camera in one frame will be close to their new values in the next frame. By introducing two new terms into the cost function in (34), a requirement is added that the motion of the camera be somewhat smooth over many frames. The new cost function is given by

$$
\begin{aligned}
r(\mathbf{a},\mathbf{b},\lambda) = \quad & \sum_{i=1}^{n} \left\| \mathbf{y}_i + \hat{\mathbf{x}}_i \mathbf{a}'(\mathbf{m}_i \lambda_{i1} + \mathbf{q}_i) - \hat{\mathbf{x}}_i^2 \mathbf{b} \right\|_{\mathbf{W}_i}^2 \\
& + s_a \sum_{t=1}^{N-1} \left\| \mathbf{a}_{t+1} - \mathbf{a}_t \right\|^2 + s_b \sum_{t=1}^{N-1} \left\| \mathbf{b}_{t+1} - \mathbf{b}_t \right\|^2
\end{aligned}
\tag{64}
$$

where s_a and s_b are two constants that can be adjusted based upon how smooth the camera translation and rotation is expected to be. A new $3(N\text{-}1) \times 3N$ matrix $\mathbf{H}$ and a new $6(N\text{-}1) \times 3N$ matrix $\mathbf{H}'$ are formed using the equations

$$
\mathbf{H} = \begin{bmatrix} 1 & 0 & 0 & -1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & -1 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & -1 & 0 \\ 0 & 0 & 0 & \ddots & 0 & 0 & \ddots \end{bmatrix}, \qquad \mathbf{H}' = \begin{bmatrix} s_a \mathbf{H} \\ s_b \mathbf{H} \end{bmatrix}. \tag{65}
$$

The solution for the optimal translation and rotation from equation (63) is modified to be

$$
\mathbf{d} = -\left(\sum_{i=1}^{n} \mathbf{P}_i^T \mathbf{W}_i \mathbf{P}_i + \mathbf{H}'^T \mathbf{H}' \right)^{-1} \sum_{i=1}^{n} \mathbf{P}_i^T \mathbf{W}_i \mathbf{y}_i \tag{66}
$$

and the solution for the rotation from a known depth is modified in a similar fashion

$$
\mathbf{b} = \left(\sum_{i=1}^{n} \hat{\mathbf{x}}_i^2 \mathbf{Q}_i^T \mathbf{W}_i \mathbf{Q}_i \hat{\mathbf{x}}_i^2 + s_b^2 \mathbf{H}^T \mathbf{H} \right)^{-1} \sum_{i=1}^{n} \hat{\mathbf{x}}_i^2 \mathbf{Q}_i^T \mathbf{W}_i \mathbf{Q}_i (\mathbf{y}_i + \hat{\mathbf{x}}_i \mathbf{a}' \mathbf{q}_i). \tag{67}
$$

Chapter 4

Results

Two simulations were created to compare the new methods to other similar methods. The two simulations are identical except one uses two frames and the other uses five frames. On each trial, fifty feature points are selected at random locations. Random translation and rotation vectors are chosen and then used to calculate the optical flow values at each feature point. The optical flow values are calculated exactly and then these optical flow values are corrupted by noise at each time step. The variance of the optical flow noise in the *x* and *y* directions is randomly assigned a value between 0.25 and 1.75 times a mean noise value. The correlation coefficient is randomly chosen to be between -1 and 1. The corrupted optical flow values are then used in several different methods for comparison. Some aspects of this simulation are known to be unrealistic. The simulation assumes that the exact covariance matrix of the noise is known. In practice, this must be estimated using Equation (10).

The results from the two-frame simulation are shown in Figures 4 and 5. In this simulation, the new method that uses probability distributions is compared with another method that does not [23]. The noise level along the *x*-axis is equal to the average variance of the noise and is measured in focal lengths. The translation error is measured as the size of the angle between the estimated translation vector and the true translation vector. The rotation error is measured as the norm of the difference between the estimated rotation vector and the true rotation vector. The result for both the rotation and translation estimation show the new method gives better results across all noise levels.

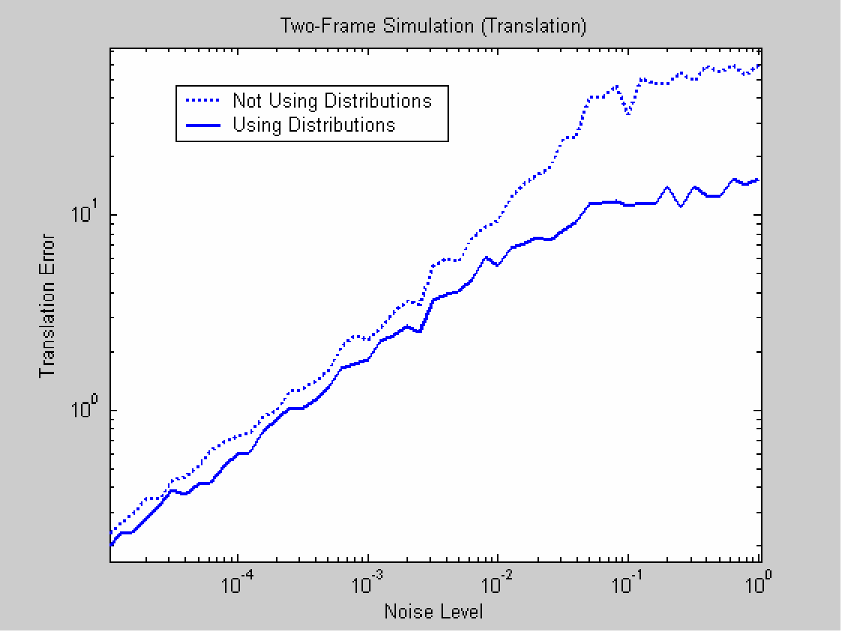

Figure 4: Two-Frame Simulation – Translation

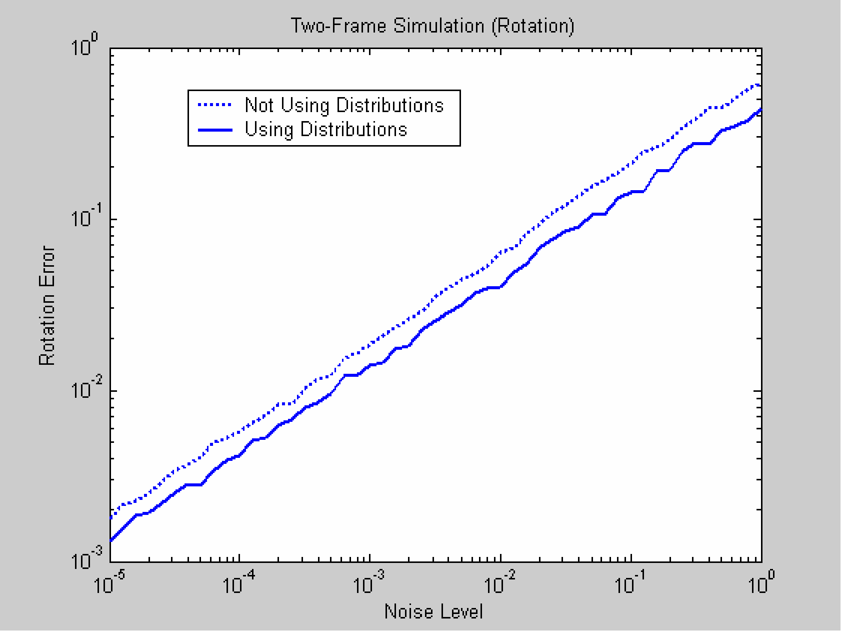

Figure 5: Two-Frame Simulation – Rotation

Figures 6, 7, and 8 show the median translation, rotation, and depth error after 400 trials for four different multiple-frame methods. The translation error is equal to the norm of the difference between the estimated translation vector and the true translation and is measured in terms of the distance the camera travel between the first and second frames. The rotation error is equal to the norm of the difference between the estimated rotation vector and the true rotation vector. The inverse depth error is equal to the average error in the inverse depth calculation for all fifty points. The inverse depth is measured in terms of the inverse distance the camera traveled between the first and second frames.

Two of the four tested methods assume constant depth and two do not. Two of the four methods use the optical flow probability distributions, and two methods do not use the probability distributions and replace the matrix $\mathbf{W}_i$ with the identity matrix. A fifth method, which is a two-frame method, is included in the depth error results of Figure 8. This uses the two-frame method of Soatto and Brockett [23]. This method has similarities with the other four methods that were tested. However, the two-frame method was never designed to work on multiple frames. The reason it is included is to show how much better results can be found using multiple frames instead of just two frames. The results in Figures 6, 7, and 8 show that the methods that use probability distributions perform better. The results also show that the methods which do not assume a constant depth perform much better at low noise levels. However, at high noise levels, the dynamic depth methods become unstable and perform more poorly than the methods that assume constant depth. The results are plotted on a log-log scale. So the differences between the methods may appear to be smaller than they truly are.

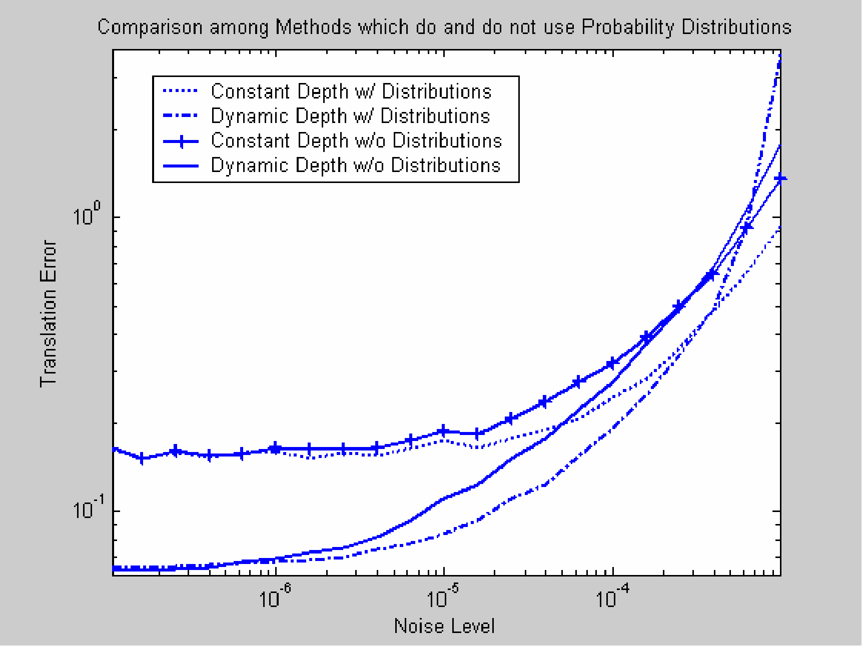

Figure 6: Translation errors for different methods that do and do not use probability distributions.

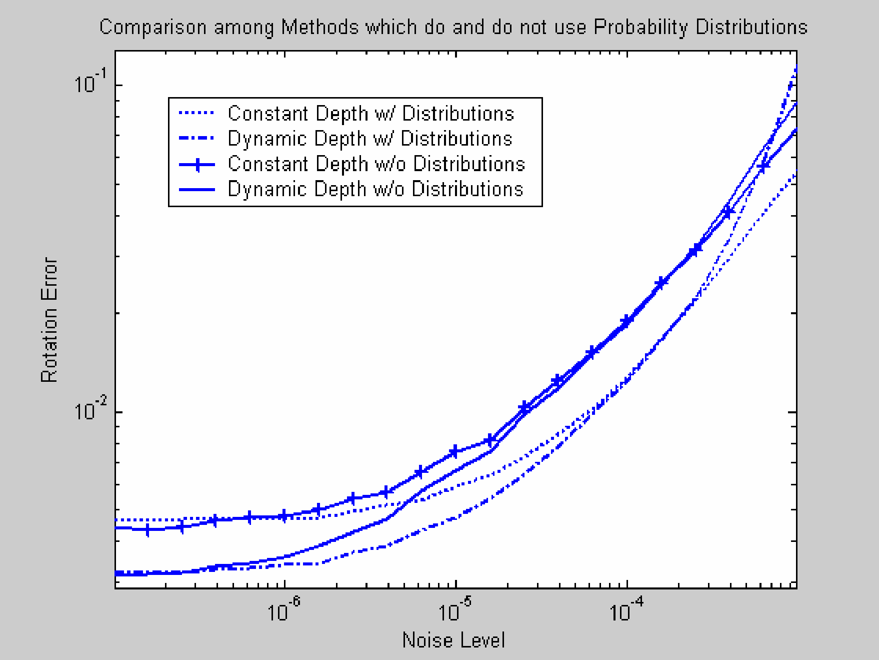

Figure 7: Rotation errors for different methods that do and do not use probability distributions.

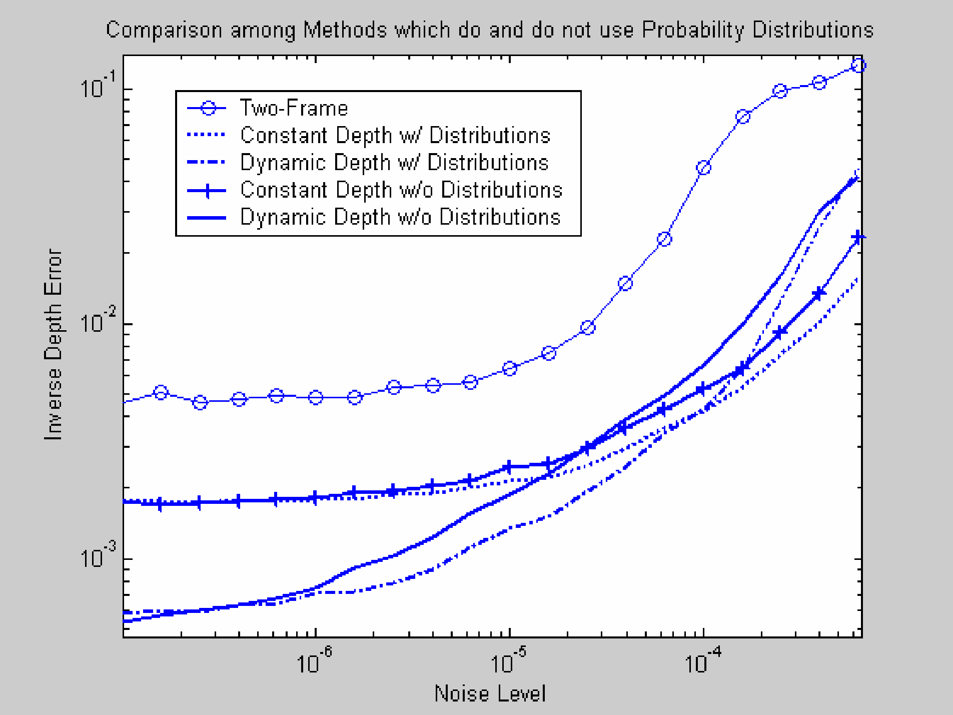

Figure 8: Depth errors for different methods
that do and do not use probability distributions.

To better test the methods presented here, a comparison is needed with another method that uses multiple frames and that uses probability distributions. There are only a handful of methods that fit this criterion. Zucchelli et al. [29] present a method that fits this criterion and which is similar because it formulates the problem as a least squares optimization problem. In some ways, it is difficult to compare these different methods. All of the methods require an initial translation estimate, but the Zucchelli method is particularly sensitive to the starting location. In fact, the authors suggest running the same algorithm fifteen to twenty times with different initial values each time. The algorithms presented in Section 3 need only one initial value and to run only once. As a compromise, the simulation uses only one initial value, but it is given an initial value close to the true value so that the Zucchelli method will work relatively better.

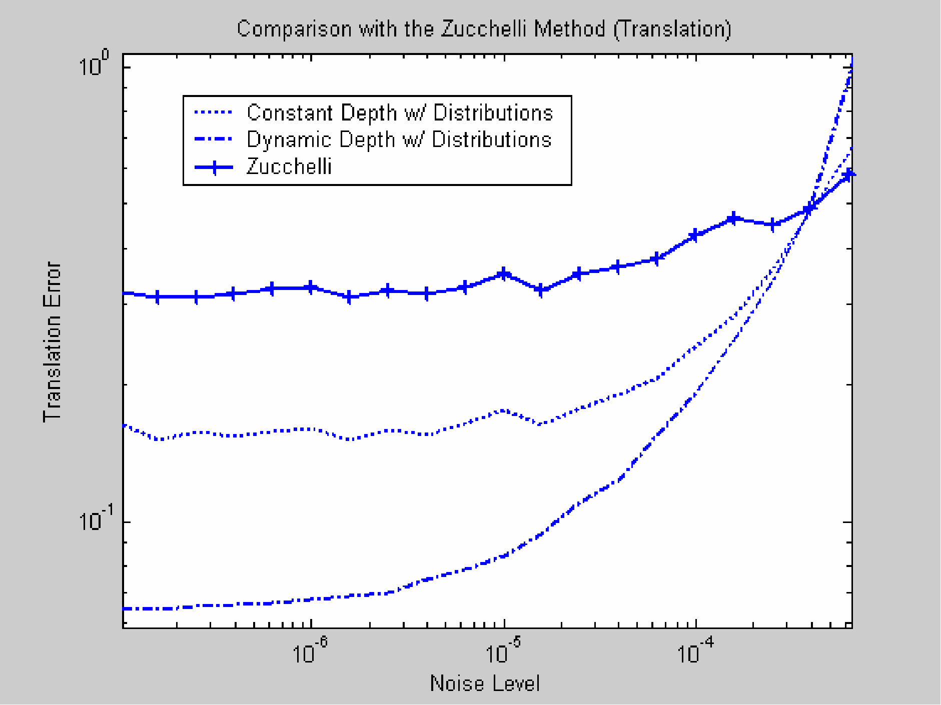

Figure 9: Translation error comparison with Zucchelli's Method.

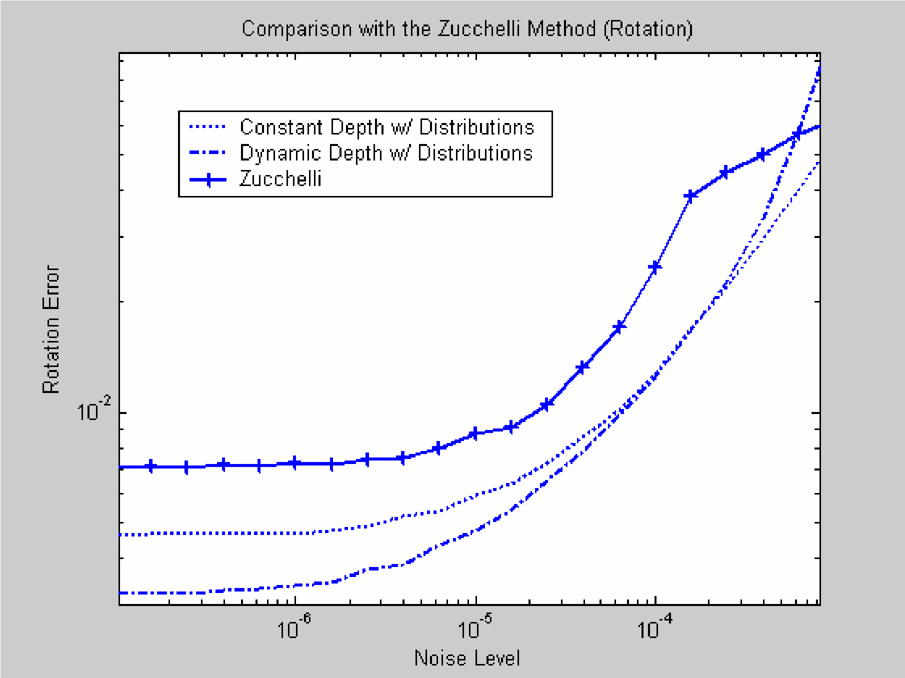

Figure 10: Rotation error comparison with Zucchelli's Method.

The results in Figures 9 and 10 show that the new method perform better except at very high noise levels. There are several reasons why the new methods work better. The Zucchelli method uses a camera model that is less accurate than the one in Equation (29). One of the reasons spherical projection is used is to be able to find a more accurate camera model. The Zucchelli method uses a Gauss-Newton iteration. Like our method, it requires an initial translation value. However, it is very sensitive to the location of the initial translation value. If there is a poor initial value, the solution will often converge to a local minimum instead of the global minimum. The Zucchelli method assumes the depth is constant, which partially explains why it does not perform as well when there is little noise.

The calculation of optical flow is difficult and sometimes unreliable. Both the high noise levels and the low noise levels in the simulation are realistic values under different circumstances. However, once a certain noise level is reached nearly all methods perform unsatisfactory. The fact that one method performs better than another method at a high noise level is not as important as the fact that both methods perform so poorly that their estimate has little value.

The methods were tested on two different scenes: a computer generated scene and a scene from a real image sequence taken from a camera onboard a UAV headed towards a tree. Figure 11 shows one frame from the computer generated sequence. Computer generated images are useful because the true depths of the objects in the image are known precisely. Figure 12 shows the true inverse depth, λ_i, at each point on the image. The darker areas indicate an object that is further away from the camera and lighter areas indicate an object that is closer. From the same scene, two different image sequences were created one where the camera moves horizontally and the other where the camera moves forward. Figure 13 shows the recovered inverse depth when the camera is moving sideways, which is close to the true inverse depth. For comparison, Figure 14 shows the recovered inverse depth when probability distributions are not used. Figure 13 is significantly closer to the true inverse depth. Figure 15 shows the recovered inverse depth when the camera is traveling forward. These methods only recover the depth at the feature points on the image. The other points on the image can only be found through interpolation, which is why the images have a tiled appearance. The centers of the tiles are the location of the feature points.

Figure 16 shows one frame of a video taken from a UAV approaching a tree. The recovered inverse depth is shown in Figure 17, which shows that the tree has been accurately detected in front of the rest of the scene.

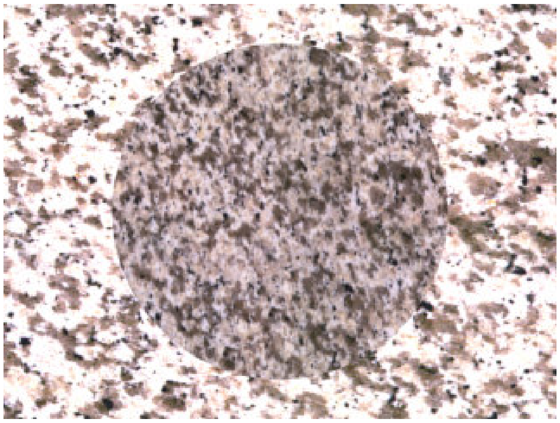

Figure 11: One Frame from a computer-generated video.

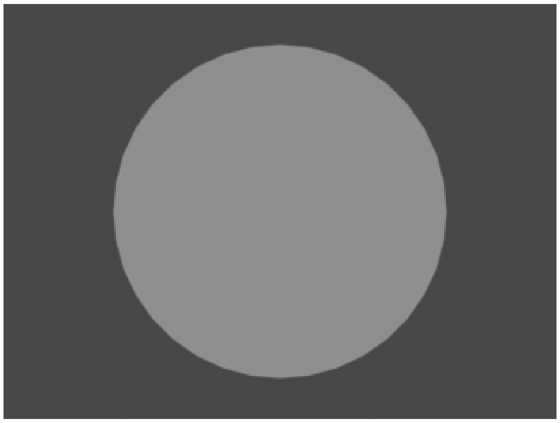

Figure 12: True inverse depth.

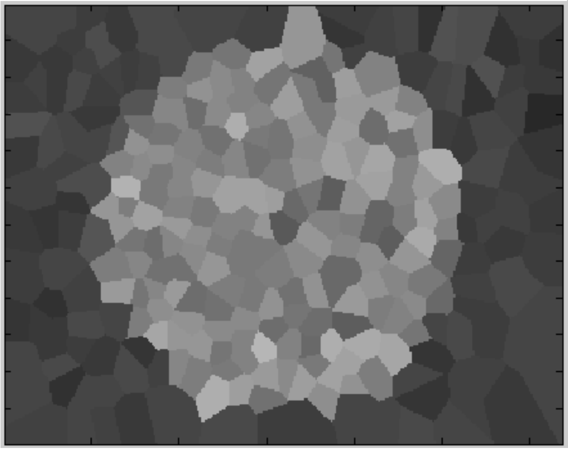

Figure 13: Recovered inverse depth using the multiple frames with distributions method.

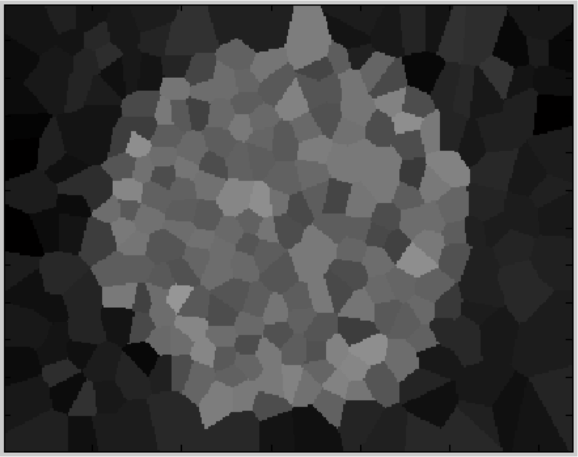

Figure 14: Recovered inverse depth using the multiple frames without distributions method.

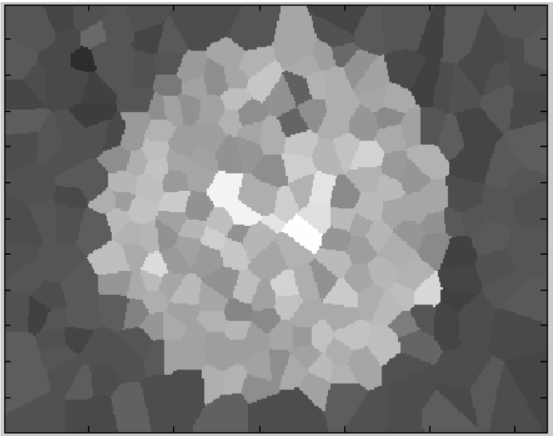

Figure 15: Recovered inverse depth with the camera moving forward.

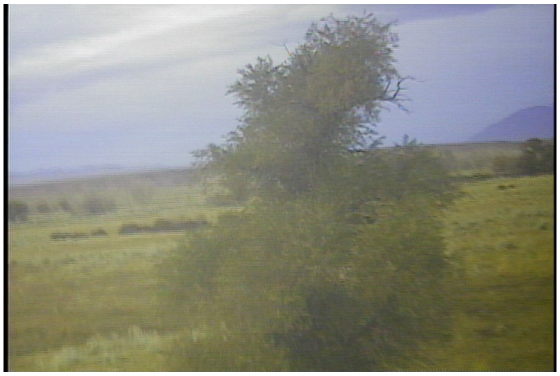

Figure 16: One frame from a video from a camera approaching a tree.

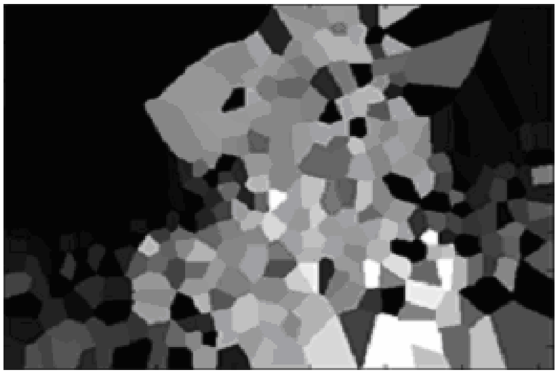

Figure 17: Recovered Inverse Depth

Chapter 5

Conclusion

5.1 Future Research

There are several areas where future research should be directed. One possible improvement is to find a method that is as simple computationally as the Gaussian methods but can handle non-Gaussian noise. Another improvement that could be made is to solve for the translation, rotation, and depth simultaneously. However, this is a highly complex nonlinear estimation problem. There also may be a slightly better way to estimate the translation in Section 3.2.3.

Structure from motion, even with the improvements developed in this thesis, can be difficult under certain circumstances. This problem could be made easier and more reliable by using additional information. An easy way more information could be included is to use more than one camera, that is to have two moving cameras rigidly attached to one another. Notice that this would be different from stereo vision, which uses two stationary cameras, and different from structure from motion, which uses a single moving camera. This idea is particularly appealing because a well-designed algorithm should accentuate the strengths of both stereo and structure from motion. Structure from motion has the advantage of being able to find corresponding points easily, but has the disadvantage of not being able to find the depth easily from the corresponding points. Stereo vision has the opposite problem. Stereo vision has the advantage of easily telling the depth from corresponding points but is not able to find corresponding points as easily. By combining the two approaches, it may be possible to use the strengths of both methods to produce a very reliable algorithm. The idea of combining stereo vision and structure from motion has been considered in the past in several methods [30, 31, 32], but these methods could all be improved by applying the techniques described here to better understand and manage the noise.

5.2 Major Contributions

This thesis makes several contributions to the field of structure from motion. First, it contributes the correlation-based method for computing optical flow probability distributions. It contributes the first structure from motion algorithm that considers non-Gaussian noise.

The Gaussian structure from motion algorithms are based on the work Soatto and Brockett [23]. However, this thesis makes several modifications and improvements to their original method. One contribution is to weight the data, so that the more valuable data is given greater value, which is accomplished by adding the weighting matrices $\mathbf{W}_i$ to Equation (34). The derivations of the matrices $\mathbf{W}_i$ and $\mathbf{G}_i$ in Equations (23) through (34) are also new. The solutions for the translation, rotation, and depth from [23] all had to be modified to incorporate the new weights $\mathbf{W}_i$.

The work of Soatto and Brockett was only a two-frame method. Another important contribution of this thesis is to extend their method to use multiple frames. The use of multiple frames necessitates the addition of a better depth model that does not assume constant depth. This depth model is unnecessary in the original two-frame method. Another contribution is the addition of a smoothness constraint.

5.3 Summary

A structure from motion algorithm that provides a more rigorous treatment of the noise in the SFM problem has been presented. By using optical flow probability distributions, a better understanding of the noise is obtained and the noise can be managed more effectively. Two different methods for calculating optical flow probability distributions were presented, as well as methods to calculate structure from motion assuming non-Gaussian and Gaussian noise using two frames or multiple frames. The experimental results show that methods which use optical flow probability distributions better estimate the camera motion and the three-dimensional structure of the scene. The experimental results also show that our method compares favorably with other SFM algorithms that also use probability distributions.

Bibliography

[1] H. Aanaes, R. Fisker, K. Astrom, and J. M. Carstensen. Robust Factorization, *IEEE Trans. on Pattern Analysis and Machine Intelligence*, vol. 24, pp. 1215-1225, 2002.

[2] P. Beardsley, P. Torr, and A. Zisserman. 3D Model acquisition from extended image sequences. *Eur. Conf. on Computer Vision (ECCV)*, pp. II:683-695, 1996.

[3] T. J. Broida and R. Chellappa, Estimating the kinematics and structure of a rigid object from a sequence of monocular images, *IEEE Trans. Pattern Anal. Mach. Intell.* Vol. 13, pp. 497-513, 1991.

[4] A. Chiuso, R. Brocket, and S. Soatto, Optimal Structure from Motion: Local Ambiguities and Global Estimates, *Int. J. of Computer Vision* vol. 39, pp. 195-228, 2000.

[5] F. Dellaert, S. M. Seitz, C. E. Thrope, and S. Thrun. Structure from motion without correspondence, *Proc. Conf. Computer Vision and Pattern Recognition*, vol. 2, pp. 557-564, 2000.

[6] Z. Duric and A. Rosenfeld. Shooting a smooth video with a shaky camera, *Machine Vision and Applications*, Vol. 13, pp. 303-313, 2003.

[7] O. D. Faugeras and B. Mourrain, What can be seen in three dimensions with an uncalibrated stereo rig? *Eur. Conf. on Computer Vision (ECCV)*, pp. 563-576, 1992.

[8] P. Favaro, Hailin Jin, S. Soatto. A semi-direct approach to structure from motion, *Int. Conf. on Image Analysis and Processing*. pp.250-255, 2001.

[9] D. Forsyth, S. Ioffe, and J. Haddon. Bayesian structure from motion. In *Int. Conf. on Computer Vision (ICCV)*, pp. 660-665, 1999.

[10] A. Gruber and Y. Weiss. Multibody factorization with uncertainty and missing data using the EM algorithm. *Proc. Conf. Computer Vision and Pattern Recognition*. vol. 1, pp. I:707-714, 2004.

[11] R. I. Hartley, Estimation of relative camera positions for uncalibrated cameras, *Eur. Conf. on Computer Vision (ECCV),* pp. 579-587, 1992.

[12] M. Irani and P. Anandan. Factorization with Uncertainty. *Proc. European Conf. Computer Vision 2000,* pp. 539-553, 2000.

[13] A. Jepson and D. Heeger. Linear subspace methods for recovering rigid motion. *Spatial Vision in Humans and Robots*, Cambridge University Press: New York, 1992.

[14] H. C. Lonquet-Higgins. A computer algorithm for reconstructing a scene from two projections. *Nature*, 293: 133-135.

[15] Q. T. Luong and O. D. Faugeras. The fundamental matrix: Theory, algorithms, and stability analysis, *Int. J. of Computer Vision*, vol. 17, pp. 43-76, 1996.

[16] P.C. Merrell, D.J. Lee, and R.W. Beard, Statistical Analysis of Multiple Optical Flow Values for Estimation of Unmanned Air Vehicles Height Above Ground,

SPIE Optics East, Robotics Technologies and Architectures, Intelligent Robots and Computer Vision XXII, vol. 5608-28, Philadelphia, PA USA, October 25-28, 2004.

[17] P.C. Merrell, D.J. Lee, and R.W. Beard, Obstacle Avoidance for Unmanned Air Vehicles Using Optical Flow Probability Distributions, SPIE Optics East, Robotics Technologies and Architectures, Mobile Robots XVII, vol. 5609-04, Philadelphia, PA USA, October 25-28, 2004.

[18] D. D. Morris and T. Kanade. A unified factorization algorithm for points, line segments, and planes with uncertainty models. *Int. Conf. on Computer Vision*, pp. 696-702, 1998.

[19] J. Oliensis. Direct multi-frame structure from motion for hand-held cameras, *Int. Conf. on Pattern Recognition*. vol. 1, pp. 889-895, 2000.

[20] J. Oliensis and J. I. Thomas, Incorporating motion error in multi-frame structure from motion, *IEEE Trans. Pattern Anal. Mach. Intell.* Vol 21, pp. 665-671, 1999.

[21] C. J. Poelman and T. Kanade. A paraperspective factorization method for shape and motion recovery, *IEEE Trans. Pattern Analysis and Machine Intelligence,*vol.19, pp.206-218, 1997.

[22] E. P. Simoncelli, E. H. Adelson, and D. J. Heeger, Probability distributions of optical flow, *Proc. Conf. Computer Vision and Pattern Recognition*, pp. 310-315, 1991.

[23] S. Soatto and R. Brocket, Optimal Structure from Motion: Local Ambiguities and Global Estimates, *Proc. Conf. Computer Vision and Pattern Recognition,* pp. 282-288, 1998.

[24] S. Soatto and P. Perona, Rucursive 3D visual-motion estimation using subspace constraints, *Int. J. Computer Vision.* Vol. 22, pp. 235-259, 1997.

[25] Z. Sun, A. M. Tekalp and V. Ramesh, Error Characterization of the factorization method, *Computer Vision and Image Understanding*, 2001, Vol. 82, No. 2.

[26] I. Thomas and E. Simoncelli. Linear Structure from Motion. *Technical Report IRCS 94-26*, University of Pennsylvania, 1994.

[27] C. Tomasi and T. Kanade. Shape and motion from image streams under orthography: a factorization method, *Int. J. of Computer Vision*, vol. 9, pp.137-154, 1992.

[28] Z. Zhang. Determining the epipolar geometry and its uncertainty – a review. *Int. J. of Computer Vision*, 27 (2): 161-195.

[29] M. Zucchelli, J. Santos-Victor, and H. I. Christensen, Maximum likelihood structure from motion estimation integrated over time, *Int. Conf. on Pattern Recognition*, vol. 4, pp. 260-263, 2002.

[30] P. K. Ho and R. Chung. Stereo-motion with stereo and motion in complement, *IEEE Trans. Pattern Analysis and Machine Intelligence,* vol. 22, pp. 215-220, 2000.

[31] G. P. Stein and A. Shashua. Direct Estimation of motion and extended scene structure from a moving stereo rig, *Proc. Conf. Computer Vision and Pattern Recognition,* pp. 211-218, 1998.

[32] Z. Zhang and O. D. Faugeras. Three dimensional motion computation and segmentation in a long sequence of stereo frames. *Int. J. of Computer Vision*, vol. 7, pp. 211-241, 1992.