The Performance of Camera Translation Direction Estimators[1]

A. Mark Earnshaw & Steven D. Blostein
Department of Electrical & Computer Engineering
Queen's University, Kingston, Ontario K7L 3N6
earnshaw@eleceng.ee.queensu.ca
sdb@eleceng.ee.queensu.ca

Abstract

A linear method using optical flow to recover the translation direction of a moving camera has been previously presented. This algorithm produced a biased estimate and the original authors proposed a dithering technique for eliminating this bias. A more detailed explanation of the source of this bias and methods for removing it are presented here, together with a comprehensive error analysis of the different approaches which derives expressions for the expected mean vectors and covariance matrices of the translation direction estimates. This analysis also includes a necessary modification to the Cramér-Rao lower bound (CRLB) which specifies the minimum possible covariance matrix for the estimates produced by the different techniques. In addition, it is apparent that unbiased estimates of a least-squares matrix do not necessarily result in unbiased estimates of the corresponding eigenvalues and eigenvectors. A simple nonlinear extension to the linear algorithm which produces optimal (in the sense that the mean vector is unbiased and the covariance matrix approaches the CRLB) translation direction estimates is described. Sample numerical results are used to compare the various techniques. Finally, some remaining problems meriting further investigation are discussed.

[1]This research has been partially supported by a grant from the Information Technology Research Centre of Ontario

1 Introduction

In certain computer vision problems, there is a need to recover the motion parameters of a mobile camera with respect to a stationary environment. Such a situation might occur, for example, with a video camera mounted on the end-effector of a robot. A similar problem involves estimating the velocity parameters of a moving object relative to a fixed camera.

In this area of research, it would be useful to have a method which could easily recover a camera's motion parameters from the observed visual data. One approach involves processing the observed optical flow data. Heeger and Jepson [13] illustrated how the estimation of translation, rotation, and depth could be decoupled into separate problems which could then be solved in order. They later managed to reformulate the original nonlinear translation recovery method as a linear problem [15]. However, the resulting translation direction estimate tended to be biased towards the optical axis. In [16] and [17], Jepson and Heeger performed a preliminary analysis of this bias and proposed a solution for eliminating it.

This paper presents a detailed analysis of the bias and proposes alternative methods for overcoming it. We also explain why bias compensation is not as straightforward as initially thought. Existing motion recovery algorithms are reviewed in Section 2, and Section 3 discusses the assumed nature of the observations. Jepson and Heeger's algorithms are covered in Section 4, while our new method of bias compensation is outlined in Section 5. Section 6 contains an additional recently developed anti-biasing technique [21]. An extensive error analysis is performed in Section 7 for each of the algorithms. Section 8 shows how the rotation and depth estimates may be used to optimize the translation direction value. Finally, sample simulation results may be found in Section 9, and concluding remarks are made in Section 10.

2 Background

A variety of approaches have been applied to the motion recovery problem. One option involves tracking features over an image sequence and using this data to estimate the motion parameters with either a batch [7] [18] [23] [25] [27] [30] [31] or recursive technique [4] [6] [5]

[9] [11] [32]. In the former case, a nonlinear error function is usually minimized to find the optimal set of parameters. Recursive methods generally formulate a state space system model and update this model with the errors between predicted and observed feature locations.

An alternative approach is to estimate the instantaneous motion parameters at a particular moment. These algorithms typically utilize *optical flow* which is defined as the movement of differing intensity patterns across the image plane. The *motion field* is the actual quantity used in the derivations, but this information can usually not be measured directly and must be approximated by the observed optical flow. This paper assumes that flow data has already been calculated by some method (for example, see [2] [10] [14, chap. 12] [20] [24]). Barron *et al* [3] contains an extensive comparison of several flow generation algorithms.

Bruss and Horn [8] [14, chap. 17] developed a linear least-squares method for recovering motion from optical flow for the cases of pure translation and pure rotation. They also formulated the nonlinear equations for the situation with both types of motion. Adiv [1] presented an algorithm for estimating motion and structure for several independent objects by grouping adjacent image sections with compatible motion parameters together to represent a common object. As already mentioned, Jepson and Heeger [13] [15] [16] [17] have shown how to decouple this problem into three linear subproblems.

3 Optical Flow Observations

The instantaneous optical flow at a point on the image plane of a moving camera is:

$$\mathbf{u}_i = \frac{1}{Z_i}\mathrm{A}_i\mathrm{T} + \mathrm{B}_i\Omega \tag{1}$$

$$\mathrm{A}_i = \begin{bmatrix} -f & 0 & x_i \\ 0 & -f & y_i \end{bmatrix} \qquad \mathrm{B}_i = \begin{bmatrix} x_i y_i & -(1+x_i^2) & y_i \\ (1+y_i^2) & -x_i y_i & -x_i \end{bmatrix}$$

where T and Ω represent the translational and rotational velocities of the camera, (x_i, y_i) specify the image plane coordinates of the i^{th} pixel, and Z_i is the corresponding unknown depth. Note that A_i and B_i are completely defined by the camera geometry.

The model we used for noisy measurements assumes additive Gaussian noise with an identical distribution over the entire image. The corrupted observations are thus:

$$\begin{bmatrix} (\tilde{\mathbf{u}}_i)_x \\ (\tilde{\mathbf{u}}_i)_y \end{bmatrix} = \begin{bmatrix} (\mathbf{u}_i)_x \\ (\mathbf{u}_i)_y \end{bmatrix} + \begin{bmatrix} (\Delta\mathbf{u}_i)_x \\ (\Delta\mathbf{u}_i)_y \end{bmatrix}$$

$(\Delta\mathbf{u}_i)_x$ and $(\Delta\mathbf{u}_i)_y$ are noise components in the x and y directions. These terms are assumed to be independent with identical zero-mean Gaussian distributions $N(0, \sigma_n^2)$. The standard deviation of the noise σ_n is a specified percentage ρ of the average flow vector length.

4 Jepson & Heeger's Algorithm

In [15], [16] and [17], Jepson and Heeger presented a method for determining the translation direction for a moving camera based on the optical flow data. The appeal of this approach was the use of a *linear* system of equations. This algorithm is briefly reviewed here.

4.1 Original Biased Method

The observed optical flow vectors may be used to derive linear constraint vectors τ_i which are (in the noiseless case) perpendicular to the translation direction T.

$$\begin{bmatrix} \tau_1^{\mathrm{T}} \\ \vdots \\ \tau_n^{\mathrm{T}} \end{bmatrix} \mathrm{T} = \begin{bmatrix} 0 \\ \vdots \\ 0 \end{bmatrix} \tag{2}$$

These constraint vectors are calculated as a convolution over a local patch of twisted flow vectors, $\mathbf{q}_j$. Each $\mathbf{q}_j$ is simply the cross-product of an optical flow vector $\mathbf{u}_j$ with the corresponding sampling direction vector $\mathbf{s}_j$.

$$\begin{aligned} \tau_i &= \sum_j c_{ij}\mathbf{q}_j \\ \mathbf{q}_j &= \mathbf{u}_j \times \mathbf{s}_j \\ \mathbf{s}_j &= \begin{bmatrix} x_j & y_j & f \end{bmatrix} \end{aligned}$$

(x_j, y_j) specify image plane coordinates, and f is the focal length of the camera lens. The coefficients c_{ij} represent a normalized convolution mask centered at position i.

$$\sum_j c_{ij}^2 \quad = \quad 1$$

These coefficients are selected to ensure that (2) holds in the noiseless case regardless of the value of Ω (see [16] for details). Typically, the same symmetric mask is used for all i.

If the number of constraint vectors n is larger than 3, T may be found via least-squares to be the eigenvector corresponding to the minimum eigenvalue of the matrix D where:

$$\mathrm{D} \quad \equiv \quad \sum_{i=1}^{n} \tau_i \tau_i^{\mathrm{T}}$$

When noisy optical flow measurements were used, the recovered translation directions were noticeably biased towards the optical axis. Jepson and Heeger [16] [17] explained this by illustrating that the covariance matrix $\Sigma_{\tilde{\tau}_i}$ for a noisy constraint $\tilde{\tau}_i$ represented a flattened ellipsoid rather than a sphere. The covariance matrix for a noisy twisted flow vector is:

$$\Sigma_{\tilde{\mathbf{q}}_i} \quad = \quad \begin{bmatrix} (p_{xx})_i & (p_{xy})_i & (p_{xz})_i \\ (p_{xy})_i & (p_{yy})_i & (p_{yz})_i \\ (p_{xz})_i & (p_{yz})_i & (p_{zz})_i \end{bmatrix} \quad = \quad \sigma_n^2 \begin{bmatrix} f^2 & 0 & -fx_i \\ 0 & f^2 & -fy_i \\ -fx_i & -fy_i & (x_i)^2 + (y_i)^2 \end{bmatrix} \tag{3}$$

If (x_i, y_i) are the image plane coordinates of the center of the i^{th} patch (*i.e.* the coordinates of $\tilde{\tau}_i$) and a symmetric convolution mask is used, then the relevant covariance matrix is:

$$\Sigma_{\tilde{\tau}_i} \quad = \quad \begin{bmatrix} (\tilde{\sigma}_{xx})_i & (\tilde{\sigma}_{xy})_i & (\tilde{\sigma}_{xz})_i \\ (\tilde{\sigma}_{xy})_i & (\tilde{\sigma}_{yy})_i & (\tilde{\sigma}_{yz})_i \\ (\tilde{\sigma}_{xz})_i & (\tilde{\sigma}_{yz})_i & (\tilde{\sigma}_{zz})_i \end{bmatrix} \quad = \quad \sigma_n^2 \begin{bmatrix} f^2 & 0 & -fx_i \\ 0 & f^2 & -fy_i \\ -fx_i & -fy_i & \alpha_i + \beta_i \end{bmatrix} \tag{4}$$

$$\alpha_i \quad = \quad (x_i)^2 + (y_i)^2 \qquad\qquad \beta_i \quad = \quad \sum_{j=1}^{n} (c_{ij})^2 \left[(x_j - x_i)^2 + (y_j - y_i)^2 \right]$$

σ_n is the standard deviation of the optical flow noise. If the same convolution mask is used for calculating each τ_i, β_i will be a constant independent of the patch location.

4.2 Dithering Method

The covariance matrix in (4) corresponds to a flattened ellipsoid which is much narrower in the mean sampling direction $\mathbf{s}_i$ than at right angles to this direction. This aspherical shape accounts for the observed bias in the translation direction estimates. Jepson and Heeger reasoned that by adding extra noise or *dithering*, the covariance matrix could be altered to represent an approximately spherical shape. This noise is added only along $\mathbf{s}_i$ since it is this dimension of the ellipsoid which is much smaller than the other two.

The eigenvalues of the covariance matrix for $\tilde{\tau}_i$ may easily be found to be:

$$\begin{aligned}
\lambda_{i,1} &= \sigma_n^2 f^2 \\
\lambda_{i,2} &= \sigma_n^2 \left(\gamma_i + \sqrt{\gamma_i^2 - f^2 \beta} \right) \\
\lambda_{i,3} &= \sigma_n^2 \left(\gamma_i - \sqrt{\gamma_i^2 - f^2 \beta} \right)
\end{aligned}$$

where γ_i is defined as:

$$\gamma_i \equiv \frac{1}{2} \left(f^2 + \alpha_i + \beta \right)$$

If the field of view is not excessively wide, α_i and β will be small compared to f^2. Thus, $\lambda_{i,1}$ and $\lambda_{i,2}$ will be close together in value. However, $\lambda_{i,3}$ will be significantly smaller than the other two eigenvalues. This illustrates the flattened nature of the covariance ellipsoid.

The unnormalized eigenvectors corresponding respectively to the above eigenvalues are:

$$\left[\begin{array}{ccc} \mathbf{v}_{i,1} & \mathbf{v}_{i,2} & \mathbf{v}_{i,3} \end{array} \right] = \left[\begin{array}{c|c|c} y_i & -x_i f & x_i \\ -x_i & -y_i f & y_i \\ 0 & x_i^2 + y_i^2 & f \end{array} \right]$$

Note that the eigenvector $\mathbf{v}_{i,3}$ represents the mean sampling direction for the patch which produced τ_i. Thus, by adding noise in this direction, the resulting covariance matrix will represent an approximation to a sphere. The required variance for this dither noise is:

$$\sigma_{d,i}^2 = \sigma_n^2 (f^2 - \lambda_{i,3}) \tag{5}$$

The expected value of the dithered least-squares matrix can be approximated as:

$$E[\bar{\mathrm{D}}] \approx \mathrm{D} + \kappa \mathrm{I}_3$$

where κ is a positive constant whose value depends upon the amount of noise present. If $\{\lambda_1, \lambda_2, \lambda_3\}$ are the eigenvalues of D, then $\{\lambda_1 + \kappa, \lambda_2 + \kappa, \lambda_3 + \kappa\}$ will be the eigenvalues of $\bar{\mathrm{D}}$. Thus, the relative ordering of the eigenvalues will not be changed. Both D and $E[\bar{\mathrm{D}}]$ will have the same eigenvectors.

4.3 Thresholding Constraint Vectors

Since each τ_i vector is a linear combination of the local twisted flow vectors, the noise component of $\tilde{\tau}_i$ is additive in nature. Each τ_i represents the normal to a plane which contains T. Therefore, the orientation of τ_i is the only quantity being considered. However, a longer τ_i will have its orientation changed less by a given amount of noise than will a shorter τ_i. In fact, when the length of τ_i is in a similar range to the length of the noise component, that particular constraint vector becomes extremely unreliable for estimating T. Consequently, the solution accuracy can be improved by ignoring all $\tilde{\tau}_i$ vectors whose lengths are less than a specified threshold.

4.4 Rotation & Depth Calculation

If $\mathbf{p}_i$ is a unit vector chosen perpendicular to the vector $\mathrm{A}_i\hat{\mathrm{T}}$ in (1) where $\hat{\mathrm{T}}$ is the translation estimate, a least-squares estimate for the camera's rotational velocity may be obtained [13].

$$\hat{\Omega} = \left[\sum_i \mathrm{B}_i^{\mathrm{T}} \mathbf{p}_i \mathbf{p}_i^{\mathrm{T}} \mathrm{B}_i\right]^{-1} \left[\sum_i \mathrm{B}_i^{\mathrm{T}} \mathbf{p}_i \mathbf{p}_i^{\mathrm{T}} \mathbf{u}_i\right]$$

Once the quantities $\hat{\mathrm{T}}$ and $\hat{\Omega}$ have been estimated, a relative depth map can be reconstructed from the optical flow data using least-squares.

$$\hat{Z}_i = \frac{(\mathrm{A}_i\hat{\mathrm{T}})^{\mathrm{T}}(\mathrm{A}_i\hat{\mathrm{T}})}{(\mathrm{A}_i\hat{\mathrm{T}})^{\mathrm{T}}(\mathbf{u}_i - \mathrm{B}_i\hat{\Omega})}$$

5 New Unbiasing Algorithm

As illustrated in Section 4.2, dithering may be used to overcome the bias in the translation direction estimates. However, it is possible to develop a more elegant explanation for the source of the bias and an alternative method for negating its effect.

We first define the following quantities where τ_i is a column vector representing the i^{th} exact constraint vector and $\Delta\tau_i$ represents the accompanying noise vector.

$$\begin{aligned} \mathrm{A}^{\mathrm{T}} &\equiv \begin{bmatrix} \tau_1 & \dots & \tau_n \end{bmatrix} \\ \mathrm{D} &\equiv \mathrm{A}^{\mathrm{T}}\mathrm{A} = \sum_i \tau_i \tau_i^{\mathrm{T}} \\ \tilde{\mathrm{D}} &\equiv (\mathrm{A}+\Delta\mathrm{A})^{\mathrm{T}}(\mathrm{A}+\Delta\mathrm{A}) \end{aligned}$$

$\tilde{\mathrm{D}}$ is the quantity which can be calculated from the noisy observations. It is desired to use this matrix to generate an unbiased estimate of the true least-squares matrix D.

Consider the expected value of $\tilde{\mathrm{D}}$. The noise components $\Delta\tau_i$ and $\Delta\mathrm{A}$ are both zero-mean.

$$\begin{aligned} E[\tilde{\mathrm{D}}] &= E[(\mathrm{A}+\Delta\mathrm{A})^{\mathrm{T}}(\mathrm{A}+\Delta\mathrm{A})] \\ &= E[\mathrm{A}^{\mathrm{T}}\mathrm{A}] + E[\mathrm{A}^{\mathrm{T}}\Delta\mathrm{A}] + E[\Delta\mathrm{A}^{\mathrm{T}}\mathrm{A}] + E[\Delta\mathrm{A}^{\mathrm{T}}\Delta\mathrm{A}] \\ &= \mathrm{A}^{\mathrm{T}}\mathrm{A} + E\left[\sum_i \Delta\tau_i \Delta\tau_i^{\mathrm{T}}\right] \\ &= \mathrm{D} + \sum_i \Sigma_{\tilde{\tau}_i} \end{aligned}$$

The $\Sigma_{\tilde{\tau}_i}$ terms are the covariance matrices of the constraint vectors as given in (4).

Clearly, $\tilde{\mathrm{D}}$ provides a *biased* estimate of D. Now consider the following quantity.

$$\acute{\mathrm{D}} \equiv \tilde{\mathrm{D}} - \sum_i \Sigma_{\tilde{\tau}_i}$$

This provides an unbiased estimate since $E[\acute{\mathrm{D}}] = E[\mathrm{D}]$. $\tilde{\mathrm{D}}$ can be obtained from the observed optical flow, and the $\Sigma_{\tilde{\tau}_i}$ terms can be calculated from known or estimated quantities. Thus, $\acute{\mathrm{D}}$ may be easily computed and used to generate an estimate for T. A similar operation was also proposed by Kanatani [19] to obtain unbiased estimates in a least-squares situation.

6 Whitening Method

The main disadvantage of the two anti-biasing algorithms discussed above is their requirement for a reasonably accurate estimate of σ_n. Kanatani [19] proposed a renormalization method which iteratively estimates this information, although this may be computationally expensive. A preferable approach is MacLean and Jepson's [21] anti-biasing technique which does not require knowledge of the noise variance.

As shown in the previous section, the expected value of the noisy least-squares matrix is:

$$E[\tilde{\mathrm{D}}] \;=\; \mathrm{D} + \sum_i \Sigma_{\tilde{\tau}_i}$$

Now define the following known quantity.

$$\mathrm{M} \;\equiv\; \frac{1}{\sigma_n^2} \sum_i \Sigma_{\tilde{\tau}_i}$$

Let $\mathrm{M}^{1/2}$ be the matrix square root of M. Both of these matrices will be symmetric positive definite. We can use this quantity to transform the noisy least-squares matrix as shown.

$$\begin{aligned} \check{\mathrm{D}} &\equiv \mathrm{M}^{-1/2}\tilde{\mathrm{D}}\mathrm{M}^{-1/2} \\ E[\check{\mathrm{D}}] &= \mathrm{M}^{-1/2}\mathrm{D}\mathrm{M}^{-1/2} + \sigma_n^2\mathrm{I} \end{aligned}$$

Adding a multiple of the identity matrix will not affect the eigenvectors of the matrix or the ordering of the eigenvalues in this situation. Consequently, the eigenvectors of $E[\check{\mathrm{D}}]$ are independent of the noise variance σ_n^2 so this quantity need not be estimated.

If $\check{\mathrm{T}}$ is the minimum eigenvalue eigenvector of $\check{\mathrm{D}}$, the translation direction estimate in the original coordinate frame may be calculated as:

$$\mathrm{T} \;=\; \mathrm{M}^{-1/2}\check{\mathrm{T}}$$

T must usually be renormalized since $\mathrm{M}^{-1/2}$ is not a length preserving transformation.

7 Error Analysis

In this section, we show how to approximate the mean vectors and covariance matrices of the translation direction estimates. It is first necessary to examine the least-squares matrices.

The unbiasing method is considered initially since it is simpler to analyze.

7.1 Least-Squares Matrix – Unbiasing Method

The means of the least-squares matrices $\tilde{\mathrm{D}}$ and $\acute{\mathrm{D}}$ have already been calculated.

$$E[\tilde{\mathrm{D}}] = \mathrm{D} + \sum_i \Sigma_{\tilde{\tau}_i} \qquad\qquad E[\acute{\mathrm{D}}] = \mathrm{D}$$

Now suppose $\tilde{\mathrm{D}}$ is rewritten as a vector $\tilde{\mathbf{d}}$ with the columns of $\tilde{\mathrm{D}}$ being stacked vertically. It is then possible to calculate a covariance matrix for $\tilde{\mathbf{d}}$.

$$\Sigma_{\tilde{\mathbf{d}}} = [\tilde{\sigma}_{i,j}] = E[\tilde{\mathbf{d}}\tilde{\mathbf{d}}^{\mathrm{T}}] - E[\tilde{\mathbf{d}}]E[\tilde{\mathbf{d}}^{\mathrm{T}}]$$

For illustrative purposes, we shall only consider one entry in the covariance matrix – the remaining entries may be calculated in a similar fashion.

$$\tilde{\sigma}_{2,6} = E[\tilde{d}_2\tilde{d}_6] - E[\tilde{d}_2]E[\tilde{d}_6] \tag{6}$$

Letting $\tau_i = \begin{bmatrix} x_i & y_i & z_i \end{bmatrix}^{\mathrm{T}}$ and $\Delta\tau_i = \begin{bmatrix} \delta x_i & \delta y_i & \delta z_i \end{bmatrix}^{\mathrm{T}}$, the second term in (6) may be evaluated as:

$$\begin{aligned} E[\tilde{d}_2]E[\tilde{d}_6] &= \left\{\sum_i x_i y_i + (\tilde{\sigma}_{xy})_i\right\}\left\{\sum_j y_j z_j + (\tilde{\sigma}_{yz})_j\right\} \\ &= \sum_i\sum_j x_i y_i y_j z_j + x_i y_i (\tilde{\sigma}_{yz})_j + (\tilde{\sigma}_{xy})_i y_j z_j + (\tilde{\sigma}_{xy})_i (\tilde{\sigma}_{yz})_j \end{aligned}$$

and the first term as:

$$\begin{aligned} E[\tilde{d}_2\tilde{d}_6] &= E\left[\left\{\sum_i (x_i+\delta x_i)(y_i+\delta y_i)\right\}\left\{\sum_j (y_j+\delta y_j)(z_j+\delta z_j)\right\}\right] \qquad (7) \\ &= E\Bigg[\sum_i\sum_j x_i y_i y_j z_j + x_i y_i y_j \delta z_j + x_i y_i \delta y_j z_j + x_i y_i \delta y_j \delta z_j + x_i \delta y_i y_j z_j \\ &\quad + x_i \delta y_i y_j \delta z_j + x_i \delta y_i \delta y_j z_j + x_i \delta y_i \delta y_j \delta z_j + \delta x_i y_i y_j z_j + \delta x_i y_i y_j \delta z_j + \delta x_i y_i \delta y_j z_j \\ &\quad + \delta x_i y_i \delta y_j \delta z_j + \delta x_i \delta y_i y_j z_j + \delta x_i \delta y_i y_j \delta z_j + \delta x_i \delta y_i \delta y_j z_j + \delta x_i \delta y_i \delta y_j \delta z_j\Bigg] \end{aligned}$$

The δ perturbations are zero-mean so any term with only one δ will go to zero when the expectation is taken. Those terms with two δ_i's or two δ_j's can be expressed as the appropriate

entry of $\Sigma_{\tilde{\tau}_i}$. Terms with three δ's can be written as a sum of third moments of the distributions of the twisted flow vectors. However, all odd moments of a Gaussian distribution are zero [22, p. 107] so these terms will disappear when the expectation is taken. Finally, the term with four δ's can be expressed as:

$$\begin{aligned} E[\delta x_i \delta y_i \delta y_j \delta z_j] &= \left\{\sum_k (c_{ik})^2 (p_{xy})_k\right\}\left\{\sum_m (c_{im})^2 (p_{yz})_m\right\} \\ &\quad + \left\{\sum_k c_{ik}c_{jk}(p_{xy})_k\right\}\left\{\sum_m c_{im}c_{jm}(p_{yz})_m\right\} \\ &\quad + \left\{\sum_k c_{ik}c_{jk}(p_{xz})_k\right\}\left\{\sum_m c_{im}c_{jm}(p_{yy})_m\right\} \\ &= (\tilde{\sigma}_{xy})_i(\tilde{\sigma}_{yz})_j + \left\{\sum_k c_{ik}c_{jk}(p_{xy})_k\right\}\left\{\sum_m c_{im}c_{jm}(p_{yz})_m\right\} \\ &\quad + \left\{\sum_k c_{ik}c_{jk}(p_{xz})_k\right\}\left\{\sum_m c_{im}c_{jm}(p_{yy})_m\right\} \end{aligned}$$

where the p terms are entries from $\Sigma_{\tilde{\mathbf{q}}_k}$ as given in (3). Thus, (7) can be simplified to:

$$\begin{aligned} E[\tilde{d}_2\tilde{d}_6] &= \sum_i\sum_j x_i y_i y_j z_j + x_i y_i (\tilde{\sigma}_{yz})_j + (\tilde{\sigma}_{xy})_i y_j z_j + (\tilde{\sigma}_{xy})_i(\tilde{\sigma}_{yz})_j \\ &\quad + x_i y_j E\left[\delta y_i \delta z_j\right] + x_i z_j E\left[\delta y_i \delta y_j\right] + y_i y_j E\left[\delta x_i \delta z_j\right] + y_i z_j E\left[\delta x_i \delta y_j\right] \\ &\quad + \left\{\sum_k c_{ik}c_{jk}(p_{xy})_k\right\}\left\{\sum_m c_{im}c_{jm}(p_{yz})_m\right\} + \left\{\sum_k c_{ik}c_{jk}(p_{xz})_k\right\}\left\{\sum_m c_{im}c_{jm}(p_{yy})_m\right\} \end{aligned}$$

The first four terms are subtracted off when calculating the entry in the covariance matrix.

$$\begin{aligned} \tilde{\sigma}_{2,6} &= \sum_i\sum_j x_i y_j E\left[\delta y_i \delta z_j\right] + x_i z_j E\left[\delta y_i \delta y_j\right] + y_i y_j E\left[\delta x_i \delta z_j\right] + y_i z_j E\left[\delta x_i \delta y_j\right] \\ &\quad + \left\{\sum_k c_{ik}c_{jk}(p_{xy})_k\right\}\left\{\sum_m c_{im}c_{jm}(p_{yz})_m\right\} + \left\{\sum_k c_{ik}c_{jk}(p_{xz})_k\right\}\left\{\sum_m c_{im}c_{jm}(p_{yy})_m\right\} \end{aligned}$$

Since the $\tilde{\tau}$ vectors are calculated as a weighted local sum of twisted flow vectors, there will be some dependency between $\tilde{\tau}_i$ and $\tilde{\tau}_j$ if their convolution masks overlap. In such a situation, the resulting "cross-covariance" can be calculated with the following formula.

$$\begin{aligned} E\left[\delta x_i \delta y_j\right] &= E\left[\left\{\sum_k c_{ik}(\Delta\mathbf{q}_k)_x\right\}\left\{\sum_m c_{jm}(\Delta\mathbf{q}_m)_y\right\}\right] \\ &= E\left[\sum_k c_{ik}c_{jk}(\Delta\mathbf{q}_k)_x(\Delta\mathbf{q}_k)_y\right] \end{aligned} \tag{8}$$

$$
\begin{aligned}
&= \sum_k c_{ik} c_{jk} E\left[(\Delta \mathbf{q}_k)_x (\Delta \mathbf{q}_k)_y\right] \\
&= \sum_k c_{ik} c_{jk} (p_{xy})_k
\end{aligned}
$$

For the unbiased vector $\acute{\mathbf{d}}$, the covariance matrix $\Sigma_{\acute{\mathbf{d}}}$ will be the same as $\Sigma_{\tilde{\mathbf{d}}}$ since subtracting a constant vector simply shifts the mean and does not affect the covariance.

7.2 Least-Squares Matrix – Dithering Method

It is first necessary to compute the mean of the dithered least-squares matrix $\bar{\mathrm{D}}$. Let $\Delta\tau_i$ represent the noise perturbation as before and $\Delta\bar{\tau}_i$ the dithering component.

$$
\begin{aligned}
E[\bar{\mathrm{D}}] &= E\left[\sum_i (\tau_i + \Delta\tau_i + \Delta\bar{\tau}_i)(\tau_i + \Delta\tau_i + \Delta\bar{\tau}_i)^{\mathrm{T}}\right] \\
&= \sum_i E\left[\tau_i\tau_i^{\mathrm{T}} + \tau_i\Delta\tau_i^{\mathrm{T}} + \tau_i\Delta\bar{\tau}_i^{\mathrm{T}} + \Delta\tau_i\tau_i^{\mathrm{T}} + \Delta\tau_i\Delta\tau_i^{\mathrm{T}}\right. \\
&\qquad \left. + \Delta\tau_i\Delta\bar{\tau}_i^{\mathrm{T}} + \Delta\bar{\tau}_i\tau_i^{\mathrm{T}} + \Delta\bar{\tau}_i\Delta\tau_i^{\mathrm{T}} + \Delta\bar{\tau}_i\Delta\bar{\tau}_i^{\mathrm{T}}\right]
\end{aligned}
$$

$\Delta\tau_i$ and $\Delta\bar{\tau}_i$ are both zero-mean so any terms containing only one of these quantities will have an expectation of zero. $\Delta\tau_i$ and $\Delta\bar{\tau}_i$ are mutually independent so the expectation of their product can be factored into the product of their expectations which will also be zero.

$$
E[\bar{\mathrm{D}}] = \sum_i \left\{\tau_i\tau_i^{\mathrm{T}} + E\left[\Delta\tau_i\Delta\tau_i^{\mathrm{T}}\right] + E\left[\Delta\bar{\tau}_i\Delta\bar{\tau}_i^{\mathrm{T}}\right]\right\} = \mathrm{D} + \sum_i \left\{\Sigma_{\tilde{\tau}_i} + \Sigma_{\bar{\tau}_i}\right\}
$$

$\Sigma_{\tilde{\tau}_i}$ represents the covariance matrix of the undithered $\tilde{\tau}_i$ vector as given in (4). $\Sigma_{\bar{\tau}_i}$ is the covariance matrix of the dithered component which may be easily derived from the standard deviation of the dithering noise and the direction in which it is applied.

$$
\Sigma_{\bar{\tau}_i} = \begin{bmatrix} (\bar{\sigma}_{xx})_i & (\bar{\sigma}_{xy})_i & (\bar{\sigma}_{xz})_i \\ (\bar{\sigma}_{xy})_i & (\bar{\sigma}_{yy})_i & (\bar{\sigma}_{yz})_i \\ (\bar{\sigma}_{xz})_i & (\bar{\sigma}_{yz})_i & (\bar{\sigma}_{zz})_i \end{bmatrix} = \frac{\sigma_{d,i}^2}{x_i^2 + y_i^2 + f^2} \begin{bmatrix} x_i^2 & x_i y_i & x_i f \\ x_i y_i & y_i^2 & y_i f \\ x_i f & y_i f & f^2 \end{bmatrix}
$$

The derivation for the covariance matrix of the dithered vector $\bar{\mathbf{d}}$ proceeds in a similar fashion as before. Noting that $\Delta\tau_i$ and $\Delta\bar{\tau}_i$ are mutually independent and that the individual

dithering terms are independent, the sample covariance matrix entry can be shown to be:

$$\begin{aligned}\bar{\sigma}_{2,6} &= \tilde{\sigma}_{2,6} + \sum_k \{x_k y_k(\bar{\sigma}_{yz})_k + x_k z_k(\bar{\sigma}_{yy})_k + y_k y_k(\bar{\sigma}_{xz})_k + y_k z_k(\bar{\sigma}_{xy})_k \\ &\quad + (\tilde{\sigma}_{xy})_k(\bar{\sigma}_{yz})_k + (\tilde{\sigma}_{xz})_k(\bar{\sigma}_{yy})_k + (\tilde{\sigma}_{yy})_k(\bar{\sigma}_{xz})_k + (\tilde{\sigma}_{yz})_k(\bar{\sigma}_{xy})_k \\ &\quad + (\bar{\sigma}_{xy})_k(\bar{\sigma}_{yz})_k + (\bar{\sigma}_{xz})_k(\bar{\sigma}_{yy})_k\}\end{aligned}$$

7.3 Least-Squares Matrix – Whitening Method

Let m_{kl} represent the (k,l) entry of the symmetric matrix $\mathrm{M}^{-1/2}$. It can easily be shown that entry (i,j) of $\check{\mathrm{D}}$ may be calculated as:

$$\check{d}_{ij} = \sum_{k=1}^{3}\sum_{l=1}^{3} m_{il} m_{kj} \tilde{d}_{lk}$$

If the least-squares matrix is converted to vector form, this can be written as:

$$\check{\mathbf{d}} = \mathrm{N}\tilde{\mathbf{d}}$$

where N is a symmetric 9×9 matrix defined as:

$$\begin{aligned}\mathrm{N} &\equiv \left[\, n_{ij} \,\right] = \left[\, m_{pq} m_{rt} \,\right] \\ p &= ((i-1) \bmod 3) + 1 \qquad & r &= \text{floor}\,\{(j+2)/3\} \\ q &= ((j-1) \bmod 3) + 1 \qquad & t &= \text{floor}\,\{(i+2)/3\}\end{aligned}$$

The covariance matrix of $\check{\mathbf{d}}$ may now be found to be:

$$\begin{aligned}\Sigma_{\check{\mathbf{d}}} &= E[\check{\mathbf{d}}\check{\mathbf{d}}^{\mathrm{T}}] - E[\check{\mathbf{d}}]E[\check{\mathbf{d}}^{\mathrm{T}}] \\ &= \mathrm{N}E[\tilde{\mathbf{d}}\tilde{\mathbf{d}}^{\mathrm{T}}]\mathrm{N}^{\mathrm{T}} - \mathrm{N}E[\tilde{\mathbf{d}}]E[\tilde{\mathbf{d}}^{\mathrm{T}}]\mathrm{N}^{\mathrm{T}} \\ &= \mathrm{N}\Sigma_{\tilde{\mathbf{d}}}\mathrm{N}^{\mathrm{T}}\end{aligned}$$

7.4 Translation Direction – Covariance Matrix

It is possible to approximate the covariance matrices for the eigenvalues and eigenvectors of $\tilde{\mathrm{D}}$ using matrix perturbation theory [29]. Weng [28] used a Taylor series expansion to derive

the first-order perturbation terms for eigenvalues and eigenvectors as:

$$\begin{aligned}\Delta\lambda_i &\approx \delta\lambda_i = \mathbf{v}_i^{\mathrm{T}}\Delta_{\tilde{\mathrm{D}}}\mathbf{v}_i \\ \Delta\mathbf{v}_i &\approx \delta\mathbf{v}_i = \mathrm{V}\Delta_i\mathrm{V}^{\mathrm{T}}\Delta_{\tilde{\mathrm{D}}}\mathbf{v}_i\end{aligned}$$

where the columns of V are the eigenvectors of the true D, $\Delta_{\tilde{\mathrm{D}}}$ represents the perturbation in the $\tilde{\mathrm{D}}$ matrix, and Δ_1 is defined as (Δ_2 and Δ_3 are defined similarly):

$$\Delta_1 \equiv \begin{bmatrix} 0 & 0 & 0 \\ 0 & \frac{1}{\lambda_1-\lambda_2} & 0 \\ 0 & 0 & \frac{1}{\lambda_1-\lambda_3} \end{bmatrix}$$

Note that $\Delta_{\tilde{\mathrm{D}}}\mathbf{v}_i$ may be rewritten as:

$$\Delta_{\tilde{\mathrm{D}}}\mathbf{v}_i = \begin{bmatrix} (v_x)_i\mathrm{I}_3 & (v_y)_i\mathrm{I}_3 & (v_z)_i\mathrm{I}_3 \end{bmatrix}\Delta_{\tilde{\mathbf{d}}} = \mathrm{U}_i\Delta_{\tilde{\mathbf{d}}}$$

The first-order eigenvalue and eigenvector perturbations are both zero-mean. The covariance matrix for $\tilde{\mathbf{v}}_i$ may therefore be approximated as:

$$\begin{aligned}\Sigma_{\tilde{\mathbf{v}}_i} &= E\left[\Delta\mathbf{v}_i\Delta\mathbf{v}_i^{\mathrm{T}}\right] \\ &\approx E\left[\left(\mathrm{V}\Delta_i\mathrm{V}^{\mathrm{T}}\mathrm{U}_i\Delta_{\tilde{\mathbf{d}}}\right)\left(\Delta_{\tilde{\mathbf{d}}}^{\mathrm{T}}\mathrm{U}_i^{\mathrm{T}}\mathrm{V}\Delta_i\mathrm{V}^{\mathrm{T}}\right)\right] \\ &= \mathrm{V}\Delta_i\mathrm{V}^{\mathrm{T}}\mathrm{U}_i E\left[\Delta_{\tilde{\mathbf{d}}}\Delta_{\tilde{\mathbf{d}}}^{\mathrm{T}}\right]\mathrm{U}_i^{\mathrm{T}}\mathrm{V}\Delta_i\mathrm{V}^{\mathrm{T}} \\ &= \mathrm{V}\Delta_i\mathrm{V}^{\mathrm{T}}\mathrm{U}_i\Sigma_{\tilde{\mathbf{d}}}\mathrm{U}_i^{\mathrm{T}}\mathrm{V}\Delta_i\mathrm{V}^{\mathrm{T}}\end{aligned}$$

The covariance matrices for the translation directions obtained from the unbiasing and dithering approaches may be found by substituting $\Sigma_{\acute{\mathbf{d}}}$ or $\Sigma_{\bar{\mathbf{d}}}$ respectively for $\Sigma_{\tilde{\mathbf{d}}}$. Unfortunately, since the T estimate must be renormalized after applying the whitening algorithm, this covariance matrix approximation is not applicable for the whitening situation.

The (i,j) entry of the 3×3 eigenvalue covariance matrix may be approximated as:

$$\begin{aligned}(\Sigma_{\tilde{\lambda}})_{i,j} &= E\left[\Delta\lambda_i\Delta\lambda_j\right] \\ &\approx E\left[\left(\mathbf{v}_i^{\mathrm{T}}\mathrm{U}_i\Delta_{\tilde{\mathbf{d}}}\right)\left(\Delta_{\tilde{\mathbf{d}}}^{\mathrm{T}}\mathrm{U}_j^{\mathrm{T}}\mathbf{v}_j\right)\right] \\ &= \mathbf{v}_i^{\mathrm{T}}\mathrm{U}_i E\left[\Delta_{\tilde{\mathbf{d}}}\Delta_{\tilde{\mathbf{d}}}^{\mathrm{T}}\right]\mathrm{U}_j^{\mathrm{T}}\mathbf{v}_j \\ &= \mathbf{v}_i^{\mathrm{T}}\mathrm{U}_i\Sigma_{\tilde{\mathbf{d}}}\mathrm{U}_j^{\mathrm{T}}\mathbf{v}_j\end{aligned}$$

7.5 Translation Direction – Mean Vector

During simulation experiments, it was observed that the translation direction estimates were still visibly biased, despite using unbiased estimates of the least-squares matrices. It is simple to show that $\hat{\mathrm{T}}$ is asymptotically unbiased (*i.e.* as the number of τ_i vectors increases, the expected value of the T estimate approaches the true value). However, an unbiased estimate of D does not necessarily imply that the corresponding estimate of T is unbiased.

7.5.1 Second-Order Eigenvector Perturbations

It can be shown (see Appendix B) that extending the method of Weng [28] leads to the expected values of the second-order perturbations of the eigenvalue λ_1 and its corresponding eigenvector $\mathbf{v}_1 = \begin{bmatrix} v_x & v_y & v_z \end{bmatrix}^{\mathrm{T}}$.

$$\begin{aligned} E[\delta^2\lambda_1] &= \mathbf{v}_1^{\mathrm{T}}\mathrm{E}_1\mathbf{v}_1 \\ E[\delta^2\mathbf{v}_1] &= \mathrm{G}_1\mathrm{E}_1\mathbf{v}_1 - \mathrm{G}_1\mathrm{G}_1\mathrm{F}_1\mathbf{v}_1 \end{aligned} \tag{9}$$

where G_1 is defined in terms of quantities specified in Section 7.4.

$$\mathrm{G}_1 \equiv \mathrm{V}\Delta_1\mathrm{V}^{\mathrm{T}}$$

E_1 and F_1 are both symmetric 3×3 matrices with elements as defined in Appendix B.

The first-order eigenvector perturbations are linear functions of the perturbation in the least-squares vector $\mathbf{d}$. Since this latter quantity is zero-mean, the expected values of the first-order perturbations will also be zero. However, the expected values of the second-order expansion terms are clearly nonzero. The biased mean of the recovered translation direction vector can thus be approximated reasonably accurately as (in unnormalized form):

$$E[\hat{\mathrm{T}}] \approx \mathbf{v}_1 + E[\delta^2\mathbf{v}_1]$$

The first term on the righthand side represents the desired value, whereas the second term represents an unwanted bias. Consequently, an unbiased estimate of the least-squares matrix D does not necessarily imply an unbiased estimate of the translation direction T.

7.5.2 Whitening Renormalization

As mentioned earlier, it is necessary to renormalize the whitened translation direction estimates after transforming back to the original coordinate space. Since the lengths of the resulting vectors depend on their orientations, each vector must be scaled by a different value to have unit magnitude. Thus, the estimates are not weighted equally when the mean is calculated. The result is that the renormalization process introduces yet another bias.

Due to the nature of the renormalization equation, it is not possible to derive an analytical estimate of this bias. However, a numerical estimation process can be used to yield a value which agrees reasonably well with simulation data. By sampling a large number of evenly distributed candidate vectors over the unit sphere in whitened space, transforming each back to original space, renormalizing, and finally taking their average with each vector weighted by its corresponding pdf value, we can generate a suitable estimate of the biased mean.

Let $\Sigma_{\check{\mathbf{v}}}$ be the estimated covariance matrix for the minimum eigenvalue eigenvector $\check{\mathbf{v}}$ of the whitened least-squares matrix. Let $\mathbf{w}_1$, $\mathbf{w}_2$, and $\mathbf{w}_3$ be the eigenvectors of $\Sigma_{\check{\mathbf{v}}}$. It transpires that $\mathbf{w}_1$ equals the expected value of $\check{\mathbf{v}}$ and that the corresponding eigenvalue is zero. Thus, we can assume that the perturbation of $\check{\mathbf{v}}$ has a two-dimensional zero-mean Gaussian distribution. We now need to convert the unit sphere candidate vectors to the $\mathbf{w}_2 - \mathbf{w}_3$ plane which is tangent to the unit sphere. This is accomplished by extending each $\check{\mathbf{v}}_i$ to intersect the plane and recording its new coordinates $\check{\mathbf{w}}_i$ as defined by $\mathbf{w}_2$ and $\mathbf{w}_3$.

$$\begin{aligned} \mathrm{W} &\equiv \begin{bmatrix} \mathbf{w}_2 & \mathbf{w}_3 \end{bmatrix} \\ \check{\mathbf{w}}_i &= \frac{\mathrm{W}^{\mathrm{T}}\check{\mathbf{v}}_i}{\check{\mathbf{v}}_i^{\mathrm{T}}\mathbf{w}_1} \end{aligned}$$

$\check{\mathbf{w}}$ has a zero-mean Gaussian distribution with covariance matrix:

$$\Sigma_{\check{\mathbf{w}}} = \mathrm{W}^{\mathrm{T}}\Sigma_{\check{\mathbf{v}}}\mathrm{W}$$

7.6 Cramér-Rao Lower Bound

There are two important quantities to consider when evaluating the performance of a given estimator. The mean indicates whether or not the estimator is biased, and the covariance ma-

trix quantifies the reliability of the estimator. A smaller covariance matrix is more desirable since any individual estimate will be more likely to be closer to the mean. The *Cramér-Rao lower bound* (CRLB) represents the minimum possible covariance matrix for any estimator. This implies that the difference between any estimator's covariance matrix and the CRLB will be a positive semi-definite matrix (*i.e.* the eigenvalues will be non-negative).

We would like to use a set of observations (the τ_i's) to estimate a parameter vector (T). Suppose we stack the noisy constraint vectors vertically to yield a single observation vector.

$$\tilde{\tau} \equiv \begin{bmatrix} \tilde{\tau}_1 \\ \vdots \\ \tilde{\tau}_n \end{bmatrix}$$

We now need to express the probability distribution of $\tilde{\tau}$ in terms of T. Let us first consider the mean of $\tilde{\tau}_i$. Assuming $\Omega = 0$, a flow vector may be written in terms of T as follows.

$$\mathbf{u}_j = \frac{1}{Z_j} \begin{bmatrix} -f & 0 & x_j \\ 0 & -f & y_j \\ 0 & 0 & 0 \end{bmatrix} \begin{bmatrix} T_x \\ T_y \\ T_z \end{bmatrix} = \mathrm{A}_j \mathrm{T}$$

The twisted flow vector may also be expressed as a linear function of T.

$$\mathbf{q}_j = \mathbf{u}_j \times \mathbf{s}_j = -\mathbf{s}_j \times \mathbf{u}_j = \mathrm{S}_j \mathbf{u}_j = \mathrm{S}_j \mathrm{A}_j \mathrm{T}$$

$$\mathrm{S}_j = \begin{bmatrix} 0 & f & -y_j \\ -f & 0 & x_j \\ y_j & -x_j & 0 \end{bmatrix}$$

The cross-product operation has been replaced by multiplication with a skew-symmetric matrix. The mean μ_i of each $\tilde{\tau}_i$ vector is computed as a convolution over the $\mathbf{q}_j$ vectors.

$$\mu_i = \sum_j c_{ij} \mathbf{q}_j = \left\{ \sum_j c_{ij} \mathrm{S}_j \mathrm{A}_j \right\} \mathrm{T} = \mathrm{H}_i \mathrm{T}$$

Consequently, the mean for the observation vector may be written as:

$$\mu = \begin{bmatrix} \mathrm{H}_1 \\ \vdots \\ \mathrm{H}_n \end{bmatrix} \mathrm{T} = \mathrm{H}\mathrm{T}$$

We now need to consider the covariance matrix of $\tilde{\tau}$ which is made up of 3×3 blocks.

$$(\Sigma_{\tilde{\tau}})_{ij} = \begin{cases} \Sigma_{\tilde{\tau}_i} & i = j \\ 0 & \tilde{\tau}_i \text{ and } \tilde{\tau}_j \text{ are independent} \\ \mathrm{C}_{ij} & \tilde{\tau}_i \text{ and } \tilde{\tau}_j \text{ are dependent} \end{cases}$$

$\tilde{\tau}_i$ and $\tilde{\tau}_j$ are dependent if their convolution masks overlap. C_{ij} is calculated as in (8):

$$\mathrm{C}_{ij} = \sum_k c_{ik} c_{jk} \Sigma_{\tilde{\mathbf{q}}_k}$$

where $\Sigma_{\tilde{\mathbf{q}}_k}$ is the covariance matrix of the twisted flow vector $\tilde{\mathbf{q}}_k$ as given in (3).

If the observation vector is assumed to be normally distributed, it can be modeled as:

$$\tilde{\tau} = \mathrm{H}\mathrm{T} + \Psi$$

where T is the true translation value and Ψ is a zero-mean Gaussian vector with covariance matrix $\Sigma_{\tilde{\tau}}$. In this situation, the Cramér-Rao lower bound is known to be the inverse of the Fisher information matrix J [12].

$$\mathrm{J} \equiv \mathrm{H}^{\mathrm{T}} \Sigma_{\tilde{\tau}}^{-1} \mathrm{H} \tag{10}$$

The bound in (10) is not quite correct since T is constrained to have unit magnitude. Gorman and Hero [12] developed the necessary modifications to the Fisher matrix for situations where equality and/or inequality constraints apply. For our particular problem with an unbiased estimator, the constrained lower bound B_{C} is:

$$\begin{aligned} \mathrm{B}_{\mathrm{C}} &= \mathrm{Q}_{\mathrm{T}} \mathrm{J}^{-1} \\ \mathrm{Q}_{\mathrm{T}} &\equiv \mathrm{I}_3 - \mathrm{J}^{-1} [\nabla \mathrm{G}_{\mathrm{T}}]^{\mathrm{T}} \left\{ [\nabla \mathrm{G}_{\mathrm{T}}] \mathrm{J}^{-1} [\nabla \mathrm{G}_{\mathrm{T}}]^{\mathrm{T}} \right\}^{\dagger} [\nabla \mathrm{G}_{\mathrm{T}}] \end{aligned}$$

where the † operator represents the Moore-Penrose pseudo-inverse. In this case, the quantity within parentheses is a scalar so it is only necessary to take its reciprocal. $\nabla \mathrm{G}_{\mathrm{T}}$ is the gradient matrix of the equality constraint which applies in this problem.

$$\begin{aligned} \mathrm{G}_{\mathrm{T}} &= T_x^2 + T_y^2 + T_z^2 - 1 = 0 \\ \nabla \mathrm{G}_{\mathrm{T}} &= \begin{bmatrix} 2T_x & 2T_y & 2T_z \end{bmatrix} \end{aligned}$$

However, it has already been shown that the estimators being considered are not unbiased. For a constrained biased estimate with mean $\mathbf{m}_\mathrm{T}$ (which may vary depending upon the estimator), the final lower bound is [12]:

$$\mathrm{B_B} = \nabla\mathbf{m}_\mathrm{T}\mathrm{B_C}\nabla\mathbf{m}_\mathrm{T}^\mathrm{T}$$

The rows of $\nabla\mathbf{m}_\mathrm{T}$ are the gradient vectors of the corresponding elements of $\mathbf{m}_\mathrm{T}$ with respect to the elements of T. Therefore, it must be possible to express $\mathbf{m}_\mathrm{T}$ as a function of T. This is achievable for the unbiasing method as demonstrated in the previous section.

$$\mathbf{m}_\mathrm{T} = \frac{1}{|\,\mathrm{T}+\mathbf{t}\,|}(\mathrm{T}+\mathbf{t})$$

The vector $\mathbf{t} = \begin{bmatrix} t_x & t_y & t_z \end{bmatrix}^\mathrm{T} = E[\delta^2\mathrm{T}]$ represents the expected second-order perturbation in the translation direction estimate. Define the normalization factor as η.

$$\eta \equiv \frac{1}{|\,\mathrm{T}+\mathbf{t}\,|} = \left\{(T_x+t_x)^2+(T_y+t_y)^2+(T_z+t_z)^2\right\}^{-1/2}$$

Apply the product rule for differentiation to find the first column of $\nabla\mathbf{m}_\mathrm{T}$ (the other two columns can be calculated similarly).

$$\frac{\partial\mathbf{m}_\mathrm{T}}{\partial T_x} = \eta\frac{\partial(\mathrm{T}+\mathbf{t})}{\partial T_x}+(\mathrm{T}+\mathbf{t})\frac{\partial\eta}{\partial T_x} = \eta\frac{\partial\mathrm{T}}{\partial T_x}+\eta\frac{\partial\mathbf{t}}{\partial T_x}+(\mathrm{T}+\mathbf{t})\frac{\partial\eta}{\partial T_x}$$

Clearly, $\partial\mathrm{T}/\partial T_x = \begin{bmatrix} 1 & 0 & 0 \end{bmatrix}$. Referring to (9), the second differential term in the above expression can be calculated.

$$\frac{\partial\mathbf{t}}{T_x} = \mathrm{G_1}\frac{\partial\mathrm{E_1}}{\partial T_x}\mathrm{T}+\mathrm{G_1E_1}\frac{\partial\mathrm{T}}{\partial T_x}-\mathrm{G_1G_1}\frac{\partial\mathrm{F_1}}{\partial T_x}\mathrm{T}-\mathrm{G_1G_1F_1}\frac{\partial\mathrm{T}}{\partial T_x} \tag{11}$$

Since the two matrices $\mathrm{E_1}$ and $\mathrm{F_1}$ are already known, their derivatives, although complicated, can be easily found through the use of a symbolic math program such as **Maple**.

Now consider the derivative of the normalization factor.

$$\frac{\partial\eta}{\partial T_x} = -\frac{1}{2}\left\{(T_x+t_x)^2+(T_y+t_y)^2+(T_z+t_z)^2\right\}^{-3/2} \left\{\frac{\partial(T_x+t_x)^2}{\partial T_x}+\frac{\partial(T_y+t_y)^2}{\partial T_x}+\frac{\partial(T_z+t_z)^2}{\partial T_x}\right\}$$

The individual terms within the parentheses can be calculated as:

$$\begin{aligned}
\frac{\partial (T_x + t_x)^2}{\partial T_x} &= 2\,(T_x + t_x)\left(1 + \frac{\partial t_x}{\partial T_x}\right) \\
\frac{\partial (T_y + t_y)^2}{\partial T_x} &= 2\,(T_y + t_y)\,\frac{\partial t_y}{\partial T_x} \\
\frac{\partial (T_z + t_z)^2}{\partial T_x} &= 2\,(T_z + t_z)\,\frac{\partial t_z}{\partial T_x}
\end{aligned}$$

where the derivatives of the elements of $\mathbf{t}$ have already been derived in (11). Thus, $\partial\eta/\partial T_x$ can be expressed in terms of quantities which have already been calculated.

8 Estimation Optimization

During simulations, it was observed that the estimates for T were not always particularly accurate. Conversely, the maximum eigenvalue eigenvectors $\mathbf{v}_3$ were always almost exactly perpendicular to the true T. Thus, it seemed as though the linear constraint method could identify the plane containing T, but not necessarily the translation direction vector itself.

The eigenvectors of D can be used as a basis to express an arbitrary unit vector $\mathbf{r}$.

$$\begin{aligned}
\mathbf{r} &= a\mathbf{v}_1 + b\mathbf{v}_2 + c\mathbf{v}_3 \\
1 &= a^2 + b^2 + c^2
\end{aligned}$$

The least-squares error associated with using this arbitrary $\mathbf{r}$ to approximate T is:

$$e^2 = a^2\lambda_1 + b^2\lambda_2 + c^2\lambda_3$$

Clearly, e^2 can be minimized by letting $a = 1$ and $b = c = 0$, if the eigenvalues are sorted in ascending order. Similarly, e^2 can be maximized by setting $c = 1$ and $a = b = 0$. Maximizing the error implies that we are choosing a direction ($\mathbf{v}_3$) which is perpendicular to T.

The above problem is primarily a concern in cameras with a narrow field of view. Let θ represent the viewing angle for a given imaging system. Each of the τ_i vectors is ideally perpendicular to T. In addition, the method by which the constraint vectors are constructed ensures that each τ_i is essentially perpendicular to $\mathbf{s}_i$ (the corresponding mean sampling

direction). This implies that $\mathbf{s}_i$ is normal to the plane containing τ_i. Since $\mathbf{s}_i$ can vary a maximum angle of $\theta/2$ from the vertical, τ_i can vary a maximum of $\theta/2$ above or below the horizontal image plane. All of the $\tilde{\tau}_i$ vectors must therefore lie on or near the planar region defined by the intersection of the plane perpendicular to T and the exterior volume between two cones aligned with the optical axis and joined at their tips (located at the origin of the image plane) whose interior angles are $180^\circ - \theta$.

The ideal scenario occurs when T is roughly perpendicular to the image plane. In this situation, the τ_i vectors can lie at any orientation in an unrestricted planar region and the translation direction estimate which is perpendicular to them can be accurately identified. Conversely, the worst case occurs when T is almost parallel to the image plane. In such an instance, the planar region containing the τ_i's is in the shape of a bow-tie. If the viewing angle θ is relatively small, it becomes more difficult to identify the best translation estimate which is perpendicular to the constraint vectors. The linear constraint method proposed by Jepson and Heeger seems to break down under the condition of a narrow viewing angle.

The original motivation for developing the linear constraint algorithm was to bypass the need for optimizing a nonlinear error function over the entire possible range for $\hat{\mathrm{T}}$ (*i.e.* the unit sphere). However, since $\mathbf{v}_3$ is known to be almost exactly perpendicular to T, it is possible to constrain the translation direction estimate to lie in the plane defined by $\mathbf{v}_1$ and $\mathbf{v}_2$. In fact, $\hat{\mathrm{T}}$ can be written in terms of a single unknown rotation angle ϕ around $\mathbf{v}_3$.

$$\hat{\mathrm{T}} \quad = \quad \cos(\phi)\mathbf{v}_1 + \sin(\phi)\mathbf{v}_2$$

It is now simply necessary to determine the value of ϕ which minimizes some error function.

Our search technique first calculates $\hat{\mathrm{T}}$ using the current value of ϕ. The corresponding rotational velocity estimate $\hat{\Omega}$ is then computed. These two parameters are utilized to generate a depth map. All of this information is then used in (1) to compute the optical flow corresponding to the estimated values. The error function is formed from the summed squared difference between the observed ($\mathbf{u}$) and expected ($\hat{\mathbf{u}}$) optical flow.

$$e^2 \quad = \quad \sum_i (\mathbf{u}_i - \hat{\mathbf{u}}_i)^{\mathrm{T}} (\mathbf{u}_i - \hat{\mathbf{u}}_i)$$

A modified bisection algorithm is used to narrow the search interval repeatedly until a predetermined width is achieved. The search interval for ϕ is initialized as $[-\xi, \xi]$ (*e.g.* $\xi = 20°$) since the optimal $\hat{\mathrm{T}}$ is generally in the neighborhood of $\mathbf{v}_1$.

Although this nonlinear optimization uses a significantly reduced search space, it can still be quite expensive as compared to the linear constraint method. Consequently, it may be desirable to sacrifice accuracy for speed if suboptimal estimates of T can be tolerated. It should be noted, however, that an error of only several degrees in the translation direction estimate can dramatically affect the accuracy of the rotational velocity estimate.

As mentioned earlier, a camera with a large viewing angle is unlikely to suffer from the ambiguity problem inherent in estimating T using only linear constraints. In such a case, the nonlinear optimization should not be necessary. Conversely, $\mathbf{v}_1$ will be a less reliable estimate when a narrow field of view is involved. Such a situation can also occur when estimating the motion of a moving object occupying only part of the imaging array (and therefore subtending a relatively small angle). In these instances, it may be desirable to use the nonlinear search as an additional step to refine the quality of the motion estimates.

9 Experimental Results

In this section, we compare sets of sample results where the translation direction is recovered using the methods outlined in Sections 4, 5, 6 and 8. It should be noted that the values presented here are only representative since they are dependent upon both the scene structure and camera motion.

9.1 Results from Simulated Data

Figure 1 illustrates the depth map used for simulation purposes, and Figure 2 shows sample optical flow from the scene for a 16×16 imaging array. The simulated camera had a focal length of 16 mm and an imaging array measuring 6.4 mm (or 64 pixels) on each side. These values correspond to a viewing angle of approximately 22.6°. The interpixel spacing

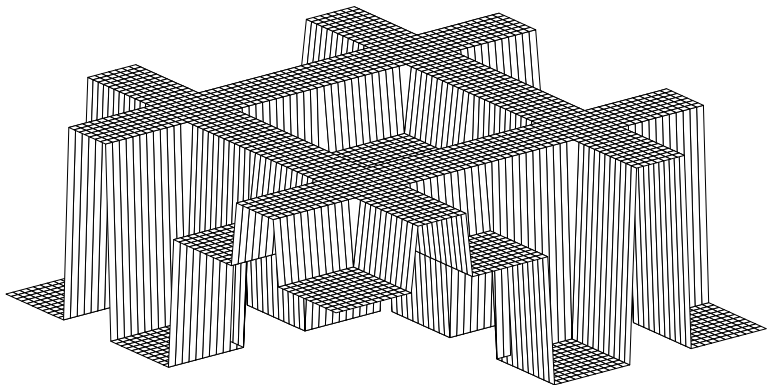

Figure 1: Depth map

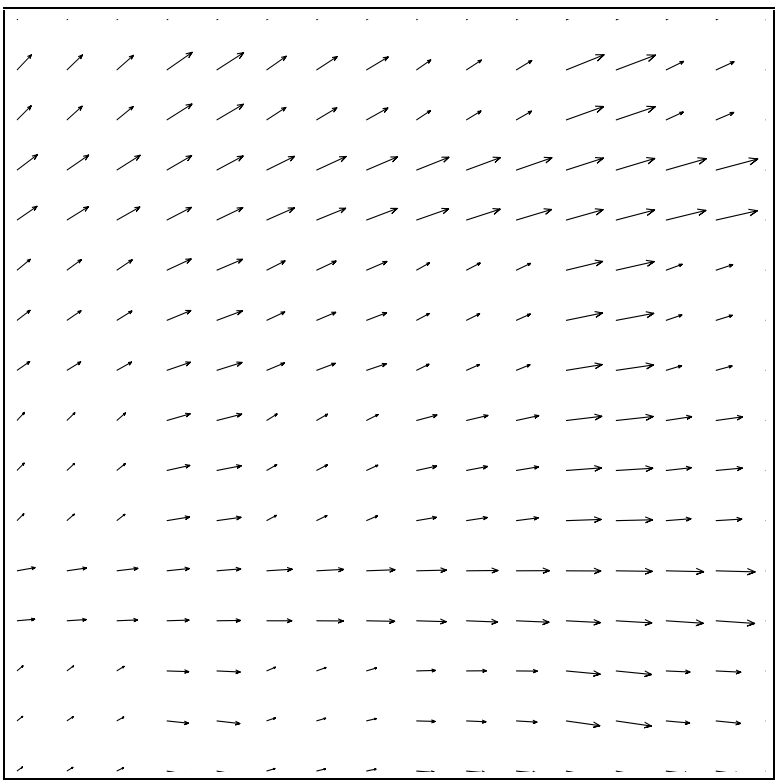

Figure 2: Sample optical flow for $T = \begin{bmatrix} -10 & 0 & 20 \end{bmatrix}$ and $\Omega = \begin{bmatrix} -0.05 & 0 & -0.1 \end{bmatrix}$

(IPS) between adjacent τ_i vectors was varied between 2, 4 and 8 pixels to investigate the effects of reducing the number of constraint vectors (an IPS of 4 implies that constraint vectors are located every four pixels in the horizontal and vertical directions). A 7×7 convolution mask was used to generate the constraint vectors in all experiments. Each simulation run consisted of 1000 trials – this number was selected to ensure that sufficiently accurate statistical estimates could be computed from the simulation results. The mean vector and covariance matrix for the translation direction estimates as calculated by each algorithm were determined, and the corresponding predicted values were also calculated via the methods outlined in Section 7. The noise percentages tested were 5%, 10% and 15%.

The first set of results were based on a translational velocity of $\mathrm{T} = \begin{bmatrix} -10 & 0 & 20 \end{bmatrix}$ (cm/s) and a rotational velocity of $\Omega = \begin{bmatrix} -0.05 & 0.0 & -0.1 \end{bmatrix}$ (rad/s). Tables 1 through 5 contain the mean translational velocity estimates for the five methods: no bias compensation, unbiasing, dithering, whitening, and optimizing (the optimizing algorithm used the unbiasing eigenvectors to define the search space). We are currently only interested in translation information so the rotational velocities were not recovered. The ideal normalized translation direction vector is $\mathrm{T} = \begin{bmatrix} -0.4472 & 0.0000 & 0.8944 \end{bmatrix}$. The error column contains the distances between the tips of the estimated and ideal translation vectors. As expected, the biased estimates are noticeably biased towards the z axis. The unbiasing method performs slightly worse than dithering for this particular example, whereas the whitening method is somewhat better. As can be seen, the optimizing technique provides the best estimate out of the five methods at lower noise levels. In addition, more accurate estimates are obtained when a larger number of constraint vectors are used, which is as expected.

Tables 6 and 7 show predicted (from Sections 7.4 and 7.5) and simulated mean vectors and covariance matrices for the translation direction estimates. These values are for 5% noise and an IPS of 4. All other parameters are the same as before. The predicted means include the bias terms discussed in Section 7.5 so they do not match the ideal value. The constrained Cramér-Rao lower bound as derived in Section 7.6 has been calculated assuming a biased estimator mean equal to that of the unbiasing technique. As a consequence, this bound is only

IPS	Noise	Mean Translation Estimate			Error
1	5%	−.2737	−.0005	.9618	0.1862
	10%	−.1858	.0308	.9821	0.2775
	15%	−.1225	.0415	.9916	0.3415
2	5%	−.2747	−.0011	.9615	0.1851
	10%	−.1821	.0277	.9829	0.2809
	15%	−.1188	.0373	.9922	0.3447
4	5%	−.2740	.0045	.9617	0.1859
	10%	−.1488	.0370	.9882	0.3150
	15%	−.0958	.0458	.9943	0.3682
8	5%	−.2963	.0045	.9551	0.1627
	10%	−.1371	.0201	.9904	0.3253
	15%	−.0678	.0285	.9973	0.3941

Table 1: No bias compensation

IPS	Noise	Mean Translation Estimate			Error
1	5%	−.4500	−.0000	.8930	0.0031
	10%	−.4878	−.0043	.8729	0.0461
	15%	−.4171	.0036	.9089	0.0336
2	5%	−.4546	.0002	.8907	0.0083
	10%	−.4637	−.0015	.8860	0.0186
	15%	−.2768	.0123	.9608	0.1833
4	5%	−.4865	−.0013	.8737	0.0445
	10%	−.1994	.0285	.9795	0.2636
	15%	−.0885	.0372	.9954	0.3745
8	5%	−.5074	−.0018	.8617	0.0685
	10%	−.1401	.0139	.9900	0.3220
	15%	−.0097	.0232	.9997	0.4506

Table 2: Unbiasing compensation

IPS	Noise	Mean Translation Estimate			Error
1	5%	−.4476	−.0003	.8943	0.0005
	10%	−.4723	−.0022	.8814	0.0284
	15%	−.4110	.0036	.9116	0.0402
2	5%	−.4508	.0000	.8926	0.0040
	10%	−.4066	.0059	.9136	0.0453
	15%	−.3057	.0122	.9520	0.1532
4	5%	−.4491	−.0013	.8935	0.0024
	10%	−.3049	−.0122	.9523	0.1541
	15%	−.1394	.0241	.9899	0.3232
8	5%	−.4270	−.0010	.9042	0.0224
	10%	−.2829	.0145	.9590	0.1771
	15%	−.0940	−.0059	.9956	0.3674

Table 3: Dithering compensation

IPS	Noise	Mean Translation Estimate			Error
1	5%	−.4488	−.0000	.8936	0.0017
	10%	−.4542	−.0005	.8909	0.0078
	15%	−.4627	−.0008	.8865	0.0175
2	5%	−.4500	.0002	.8930	0.0032
	10%	−.4625	−.0017	.8866	0.0172
	15%	−.4727	−.0034	.8812	0.0289
4	5%	−.4582	−.0007	.8889	0.0123
	10%	−.4342	.0000	.9008	0.0145
	15%	−.2473	.0134	.9688	0.2137
8	5%	−.4632	−.0005	.8863	0.0180
	10%	−.3665	−.0030	.9304	0.0884
	15%	−.2237	−.0094	.9746	0.2377

Table 4: Whitening compensation

IPS	Noise	Mean Translation Estimate			Error
1	5%	−.4471	−.0000	.8945	0.0001
	10%	−.4461	.0006	.8950	0.0014
	15%	−.3919	.0077	.9200	0.0614
2	5%	−.4470	.0001	.8945	0.0003
	10%	−.4184	.0030	.9083	0.0321
	15%	−.3833	.0072	.9236	0.0706
4	5%	−.4469	−.0002	.8946	0.0004
	10%	−.3624	.0159	.9319	0.0941
	15%	−.3498	.0187	.9366	0.1078
8	5%	−.4447	−.0001	.8957	0.0028
	10%	−.3073	.0102	.9516	0.1515
	15%	−.2870	.0063	.9579	0.1725

Table 5: Optimized estimates

Method	Pred/Sim	Mean Translation Estimate		
Ideal	P	−0.4472	0.0000	0.8944
Biased	P	−0.2739	0.0046	0.9617
	S	−0.2740	0.0045	0.9617
Unbiasing	P	−0.4841	−0.0012	0.8750
	S	−0.4865	−0.0013	0.8737
Dithering	P	−0.4431	−0.0000	0.8965
	S	−0.4491	−0.0013	0.8935
Whitening	P	−0.4570	−0.0010	0.8895
	S	−0.4582	−0.0007	0.8889
Optimized	S	−0.4469	−0.0002	0.8946

Table 6: Predicted and simulated translation direction estimates (5% noise, IPS=4)

Method	Covariance Matrix ($\times 10^{-2}$)					
	Predicted			Simulated		
Biased	0.1470	0.0125	0.0404	0.1666	0.0120	0.0484
	0.0125	0.0169	0.0034	0.0120	0.0181	0.0035
	0.0404	0.0034	0.0111	0.0484	0.0035	0.0143
Unbiasing	1.0290	0.0624	0.5145	1.2491	0.0648	0.8102
	0.0624	0.0390	0.0312	0.0648	0.0480	0.0427
	0.5145	0.0312	0.2573	0.8102	0.0427	0.5600
Dithering	4.0900	0.0951	2.0291	4.8203	0.0836	1.8541
	0.0951	0.1604	0.0472	0.0836	0.1546	0.0543
	2.0291	0.0472	1.0067	1.8541	0.0543	1.3659
Whitening				0.5584	0.0588	0.3157
				0.0588	0.0425	0.0326
				0.3157	0.0326	0.1835
Optimized				0.0044	0.0032	0.0022
				0.0032	0.0373	0.0016
				0.0022	0.0016	0.0011
Constrained CRLB	0.0290	0.0016	0.0161			
	0.0016	0.0243	0.0009			
	0.0161	0.0009	0.0089			

Table 7: Predicted and simulated covariance matrices for T estimates (5% noise, IPS=4)

IPS	Noise	# Swaps	Mean Translation Estimate			Error
1	5%	0	−.4470	.0001	.8945	0.0003
	10%	3	−.4474	.0004	.8943	0.0004
	15%	112	−.4448	.0010	.8956	0.0029
2	5%	0	−.4470	.0001	.8945	0.0002
	10%	54	−.4475	.0007	.8943	0.0008
	15%	200	−.4286	.0013	.9035	0.0207
4	5%	1	−.4471	−.0002	.8945	0.0003
	10%	277	−.4461	.0017	.8950	0.0021
	15%	187	−.4199	.0060	.9076	0.0309
8	5%	2	−.4478	−.0004	.8941	0.0008
	10%	313	−.4420	−.0043	.8970	0.0072
	15%	205	−.4005	−.0008	.9163	0.0516

Table 8: Optimized estimates for 1000 trials (with eigenvector swapping)

approximate when applied to the other anti-biasing algorithms. The primary appeal of the unbiasing and whitening methods over dithering is their smaller covariance matrices. Thus, estimates made using these techniques are more likely to be closer to the corresponding mean vector. Note that the optimized method has the smallest covariance matrix of those listed and is the only method whose covariance matrix is close to the CRLB. In addition, its mean direction vector is the closest to the ideal of the five estimation algorithms. Thus, optimizing appears to produce the best results for this particular case, although it is computationally more expensive than the other techniques.

One problem noticed during experimentation was that the distributions (assumed to be Gaussian) of the two smallest least-squares eigenvalues often had significant overlap, especially in the noisier cases. This implied that in a number of the simulation trials, the wrong eigenvector was selected. Possible solutions include ignoring any trial in which λ_1 and λ_2 are too close together in value, or evaluating the error between the observed and expected optical flow using both $\mathbf{v}_1$ and $\mathbf{v}_2$ as estimates for T and then swapping the eigenvectors if necessary. Table 8 contains the optimized translation estimates for simulation runs where the unbiasing eigenvectors were exchanged when appropriate. Comparing these results to those in Table 5 indicates that a significant improvement in accuracy has been achieved.

9.2 Results from Real Data

In addition to simulated data, optical flow calculated from image sequences was used to test the motion estimation algorithms. These image sequences and flow-generation programs are discussed in [3], and two sample images are shown in Figures 3 and 4. The tree sequence contains images of a picture so there is no true depth in the scene. There were two such sequences – one in which the camera moved parallel to the scene (translating sequence) and one with the camera moving towards the picture (diverging sequence). The Yosemite Valley sequence is a synthetically generated set of images with translation in a forward direction, although the camera is angled slightly towards the ground. The flow fields from this latter sequence were cropped to eliminate the moving clouds since their motion was independent

Figure 3: Sample image from tree sequence

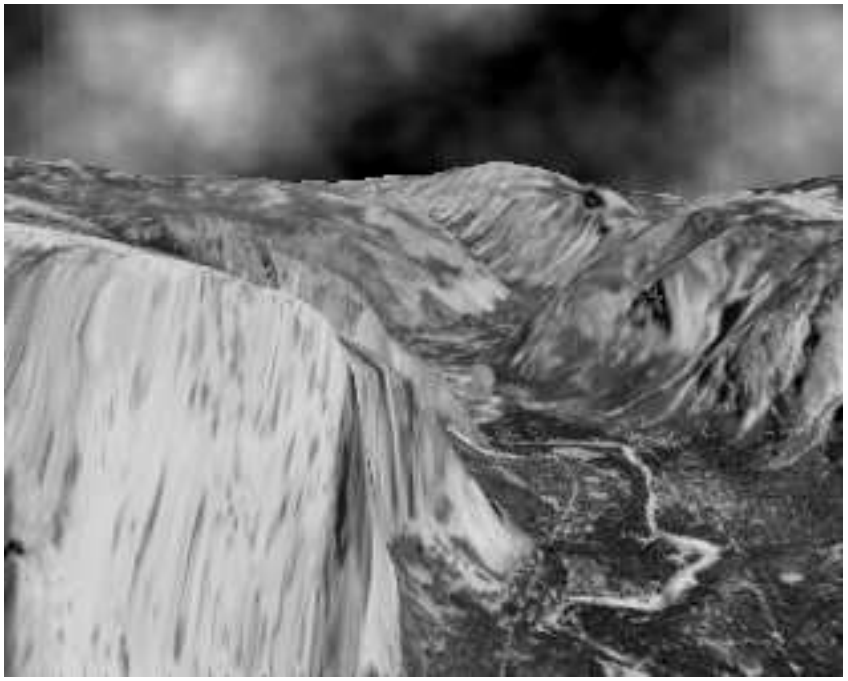

Figure 4: Sample image from Yosemite Valley sequence

Flow Type	Mask	IPS	Method	Translation Dir. Estimate			Error
Correct Flow	15×15	16	B/U/D	-1.0000	-0.0000	0.0000	0.0000
			Opt.	-1.0000	-0.0000	0.0000	0.0000
			White	-1.0000	-0.0000	0.0000	0.0000
	31×31	32	B/U/D	-1.0000	-0.0000	0.0000	0.0000
			Opt.	-1.0000	-0.0000	0.0000	0.0000
			White	-1.0000	-0.0000	0.0000	0.0000
Estimated Flow	15×15	16	Biased	-0.9151	0.1610	-0.3696	0.4120
			Unbias	-0.9524	0.1799	-0.2461	0.3085
			Dither	-0.9484	0.3147	-0.0403	0.3214
			Opt.	-0.9782	0.2078	0.0048	0.2090
			White	-0.9581	0.1792	-0.2233	0.2894
	31×31	32	Biased	-0.0008	-0.1424	0.9898	1.4136
			Unbias	-0.5931	-0.0314	-0.8045	0.9021
			Dither	-0.9622	0.0135	-0.2719	0.2748
			Opt.	-0.2575	-0.0716	-0.9636	1.2186
			White	-0.9487	0.0398	-0.3136	0.3203

Table 9: Results for translating tree sequence ($\mathrm{T} = \begin{bmatrix} -1 & 0 & 0 \end{bmatrix}$, $\rho = 0.035$)

of the camera translation. It should be noted that the translation recovery process requires knowledge of the field of view and this information was available only for these three image sequences.

The translation direction estimates for the three image sequences are contained in Tables 9, 10, and 11. The estimated flow was computed using Uras' method [26] which is a differential algorithm. Since the correct flow fields were available for the image sequences being considered, appropriate values for ρ were estimated by analyzing the differences between the correct and recovered flow fields. When the correct flow was used as input, ρ was set to zero since no noise was present, and thus the biased, unbiasing, and dithering methods (B/U/D in the table) were all identical. In the Yosemite sequence, the true translation value was taken to be the optimized estimate from the correct data. As can be seen, the optimizing technique usually provided the best estimate. The sole exception (in Table 9) was likely due to the incorrect initialization of the search space with the corresponding unbiasing estimate.

Flow Type	Mask	IPS	Method	Translation Dir. Estimate			Error
Correct Flow	15×15	16	B/U/D	-0.0031	-0.0200	0.9998	0.0202
			Opt.	-0.0003	0.0022	1.0000	0.0022
			White	0.2488	0.0042	0.9686	0.2508
	31×31	32	B/U/D	-0.2210	-0.0005	0.9753	0.2224
			Opt.	-0.0003	-0.0122	0.9999	0.0122
			White	-0.5639	0.0160	0.8257	0.5904
Estimated Flow	15×15	16	Biased	0.2300	-0.0527	0.9718	0.2376
			Unbias	-0.0425	0.1146	0.9925	0.1225
			Dither	-0.7881	-0.5861	0.1880	1.2743
			Opt.	-0.1149	-0.0506	0.9921	0.1258
			White	-0.9998	0.0070	0.0170	1.4021
	31×31	32	Biased	-0.0768	-0.1342	0.9880	0.1551
			Unbias	-0.0829	-0.1351	0.9874	0.1590
			Dither	-0.0153	-0.1062	0.9942	0.1075
			Opt.	-0.0442	-0.1377	0.9895	0.1450
			White	-0.5771	-0.0986	0.8107	0.6153

Table 10: Results for diverging tree sequence ($\mathrm{T} = \begin{bmatrix} 0 & 0 & 1 \end{bmatrix}$, $\rho = 0.157$)

The incorrect estimates from the correct flow of the diverging tree sequence illustrate the need for varying depths within the image. Adjacent flow vectors from an area of constant depth will ideally be very similar unless the motion contains a rotational component. The sum of the convolution mask elements is zero so using this mask on almost identical flow vectors will produce a constraint vector with a significantly smaller relative magnitude.

One problem observed with using optical flow calculated from actual images involved the varying degrees of confidence for the flow vectors across the image. Some of the available flow-generation programs possessed the capability of thresholding unreliable data. However, complete coverage is required if a single convolution mask is to be used to generate the τ constraint vectors. Thus, thresholding during the flow-generation stage was not used. An alternative approach using a flow field with less than 100% coverage would have required calculating an individual convolution mask and accompanying covariance matrix for each constraint vector. This is technically feasible, but computationally expensive.

Flow Type	Mask	IPS	Method	Translation Dir. Estimate			Error
Correct Flow	15×15	16	B/U/D	-0.0042	0.3544	0.9351	0.0084
			Opt.	0.0030	0.3585	0.9335	0.0000
			White	-0.0042	0.3544	0.9351	0.0084
	31×31	32	B/U/D	-0.0041	0.3552	0.9348	0.0080
			Opt.	0.0032	0.3582	0.9336	0.0000
			White	-0.0041	0.3552	0.9348	0.0080
Estimated Flow	15×15	16	Biased	-0.2005	0.1338	0.9705	0.3054
			Unbias	-0.1973	0.1451	0.9696	0.2949
			Dither	-0.1865	0.1314	0.9736	0.2985
			Opt.	-0.1281	0.3209	0.9384	0.1365
			White	-0.0158	0.7793	0.6264	0.5213
	31×31	32	Biased	-0.1647	0.1387	0.9765	0.2797
			Unbias	-0.1639	0.1410	0.9763	0.2773
			Dither	-0.1780	0.1325	0.9751	0.2924
			Opt.	-0.0829	0.3481	0.9338	0.0867
			White	-0.0611	0.3992	0.9148	0.0785

Table 11: Results for Yosemite Valley sequence ($\rho = 0.148$)

10 Discussion

Alternative approaches for eliminating the bias in translation direction estimates obtained using linear constraints have been presented. The whitening algorithm appeared to produce the best results of the three linear anti-biasing methods. The unbiasing technique has a similar level of performance as the previously proposed method of dithering the constraint vectors in terms of the corresponding mean vectors. An error analysis indicated that dithering should result in a larger covariance matrix for the translation direction estimate as compared to the unbiasing and whitening methods. This prediction was supported by the simulation results which were presented. In addition, a simple nonlinear search technique for optimizing the motion parameter estimates was discussed. This method generally possessed the smallest covariance matrix of all five translation estimation methods and approached the CRLB in simulation experiments.

One important result from the error analysis is that an unbiased estimate of a least-squares matrix does not necessarily yield unbiased eigenvalue and eigenvector estimates.

The anti-biasing algorithms can be further improved through the use of a better noise model for the observed optical flow. The current noise model assumes that the optical flow noise variance is known or can be estimated reasonably accurately, which may not be the case (although σ_n^2 is not required for the whitening algorithm). The anti-biasing methods require the covariance matrices of the τ constraints which are calculated from the probability densities of the flow vectors. Although the covariance matrices are known for simulations, there are superior methods for estimating them for real image sequences [20] [24] since it may not be realistic to assume uniform noise across the image. It might also be useful to use the estimated variance of the optical flow vectors or some other confidence measure [2] to determine the relative reliability of the constraint vectors.

Another potential area for further investigation involves developing better solutions to the problem of eigenvalue confusion described at the end of Section 9.1. This may be somewhat dependent upon the camera parameters being used, but was found to be a significant problem in the unbiasing and dithering methods during experimentation.

Parameters which can be varied include the mask size, interpixel spacing, thresholding limits, and the width of the Gaussian distributions used in the convolution mask. At present, there is no real basis for selecting good values for these parameters. Varying the first three quantities will directly affect the number of constraint vectors which are available. Increasing the number of τ_i vectors may improve the translation estimate or it may introduce additional noise, so a suitable compromise must be attained.

A Calculating Mask Coefficients

This appendix is based on information presented by Jepson and Heeger [16] [17] and explains how to calculate the coefficients for the convolution mask used to compute the τ_i constraint vectors from the twisted flow vectors. The coefficients are obtained by perturbing a DOG

(difference of Gaussians) mask with a unique quadratic polynomial.

The basis for quadratic polynomials is $\{1, x, y, x^2, y^2, xy\}$. The following matrix contains this basis evaluated at each of the m locations within the mask.

$$\mathrm{F} = \begin{bmatrix} 1 & \dots & 1 \\ x_1 & \dots & x_m \\ y_1 & \dots & y_m \\ x_1^2 & \dots & x_m^2 \\ y_1^2 & \dots & y_m^2 \\ x_1 y_1 & \dots & x_m y_m \end{bmatrix}$$

(x_i, y_i) specify the relative coordinates of the i^{th} pixel within the convolution mask.

In order for the coefficient mask to be orthogonal to all possible quadratic samples (this condition is necessary to eliminate the dependence of the observation data on the camera's rotational velocity), the following equation must be satisfied.

$$\mathrm{F} \begin{bmatrix} c_1 \\ \vdots \\ c_m \end{bmatrix} = \begin{bmatrix} 0 \\ \vdots \\ 0 \end{bmatrix}$$

This implies that C must lie in the null space of F.

Let σ_1^2 and σ_2^2 be the variances of the two zero-mean Gaussian functions for the DOG mask with $\sigma_1 < \sigma_2$.

$$G_j(x_i, y_i) = \frac{1}{2\pi\sigma_j} \exp\left(\frac{x_i^2 + y_i^2}{\sigma_j^2}\right)$$

Each unnormalized mask coefficient can then be calculated as:

$$\begin{aligned} c(x_i, y_i) &= G_1(x_i, y_i) - G_2(x_i, y_i)[1 - p(x_i, y_i)] \\ p(x_i, y_i) &= a_1 + a_2 x_i + a_3 y_i + a_4 x_i^2 + a_5 y_i^2 + a_6 x_i y_i \end{aligned}$$

where the a_i coefficients of the quadratic polynomial are unknown, but may be calculated

as:

$$\begin{bmatrix} a_1 \\ \vdots \\ a_6 \end{bmatrix} = \mathrm{H}^{-1}\mathbf{b}$$

with H and $\mathbf{b}$ being defined as:

$$\begin{aligned} \mathrm{H} &= \mathrm{FG} \\ \mathrm{G} &= \left[g_{ij} \right] = \left[G_2(x_i, y_i) f_{ji} \right] \\ \mathbf{b} &= \mathrm{F} \begin{bmatrix} G_2(x_1, y_1) - G_1(x_1, y_1) \\ \vdots \\ G_2(x_m, y_m) - G_1(x_m, y_m) \end{bmatrix} \end{aligned}$$

For the experimental results presented in this paper, the following variances were used for a $p \times p$ mask with a pixel width of w.

$$\begin{aligned} \sigma_1 &= \frac{pw}{3} \\ \sigma_2 &= 2\sigma_1 \end{aligned}$$

This selection of values was somewhat arbitrary, but did not seem to significantly affect the accuracy of the results.

B Second-Order Eigen-Perturbations

This appendix extends the derivation contained in Appendix A of Weng *et al* [28].

The equation for a perturbed eigenvalue/eigenvector pair is:

$$(\mathrm{D} + \epsilon \mathrm{B})X(\epsilon) = (\lambda_1 + \epsilon p_1 + \epsilon^2 p_2 + \ldots)X(\epsilon) \tag{12}$$

where ϵ, B, $X(\epsilon)$, and V_1 are defined as:

$$\begin{aligned} \epsilon &\equiv \max\{b_{ij}\} \\ \mathrm{B} &\equiv \frac{1}{\epsilon}\Delta\mathrm{D} \end{aligned}$$

$$
\begin{aligned}
X(\epsilon) &\equiv X + \epsilon \mathrm{V}_1 \mathbf{g}_1 + \epsilon^2 \mathrm{V}_1 \mathbf{g}_2 + \ldots \\
\mathrm{V}_1 &\equiv \begin{bmatrix} \mathbf{v}_2 & \mathbf{v}_3 \end{bmatrix}
\end{aligned}
$$

Equating the coefficients for the second-order term in (12) yields:

$$
\mathrm{D}\mathrm{V}_1 \mathbf{g}_2 + \mathrm{B}\mathrm{V}_1 \mathbf{g}_1 = \lambda \mathrm{V}_1 \mathbf{g}_2 + p_1 \mathrm{V}_1 \mathbf{g}_1 + p_2 X \tag{13}
$$

p_1 and $\mathbf{g}_1$ have already been calculated by Weng.

$$
\begin{aligned}
p_1 &= \mathbf{v}_1^{\mathrm{T}} \mathrm{B} \mathbf{v}_1 \\
\mathbf{g}_1 &= \Phi_1 \mathrm{V}_1^{\mathrm{T}} \mathrm{B} X \\
\Phi_1 &\equiv \begin{bmatrix} \frac{1}{\lambda_1 - \lambda_2} & 0 \\ 0 & \frac{1}{\lambda_1 - \lambda_3} \end{bmatrix}
\end{aligned}
$$

Noting that the columns of V_1 are eigenvectors of D, we can rewrite (13) as:

$$
\begin{aligned}
\mathrm{V}_1 \Lambda_1 \mathbf{g}_2 + \mathrm{B}\mathrm{V}_1 \mathbf{g}_1 &= \lambda \mathrm{V}_1 \mathbf{g}_2 + p_1 \mathrm{V}_1 \mathbf{g}_1 + p_2 X \qquad (14) \\
\Lambda_1 &\equiv \begin{bmatrix} \lambda_2 & 0 \\ 0 & \lambda_3 \end{bmatrix}
\end{aligned}
$$

Premultiplying both sides of (14) by the eigenvector X^{T} reduces the number of terms.

$$
X^{\mathrm{T}} \mathrm{B} \mathrm{V}_1 \mathbf{g}_1 = p_2 X^{\mathrm{T}} X \tag{15}
$$

Solving (15) for p_2 yields:

$$
\begin{aligned}
p_2 &= \frac{1}{X^{\mathrm{T}} X} X^{\mathrm{T}} \mathrm{B} \mathrm{V}_1 \mathbf{g}_1 \\
&= \frac{1}{X^{\mathrm{T}} X} X^{\mathrm{T}} \mathrm{B} \mathrm{V}_1 \Phi_1 \mathrm{V}_1^{\mathrm{T}} \mathrm{B} X \\
&= \mathbf{v}_1^{\mathrm{T}} \mathrm{B} \mathrm{V}_1 \Phi_1 \mathrm{V}_1^{\mathrm{T}} \mathrm{B} \mathbf{v}_1
\end{aligned}
$$

Thus, the second-order perturbation of the eigenvalue λ_1 is:

$$
p_2 \epsilon^2 = \mathbf{v}_1^{\mathrm{T}} \left\{ \Delta \mathrm{D} \mathrm{V}_1 \Phi_1 \mathrm{V}_1^{\mathrm{T}} \Delta \mathrm{D} \right\} \mathbf{v}_1 \tag{16}
$$

Now we would like to find the second-order eigenvector perturbation. We have (14).

$$
\mathrm{V}_1 \Lambda_1 \mathbf{g}_2 + \mathrm{B}\mathrm{V}_1 \mathbf{g}_1 = \lambda \mathrm{V}_1 \mathbf{g}_2 + p_1 \mathrm{V}_1 \mathbf{g}_1 + p_2 X
$$

Noting that the eigenvectors are orthonormal, we can premultiply through by $\mathrm{V}_1^{\mathrm{T}}$.

$$\Lambda_1 \mathbf{g}_2 + \mathrm{V}_1^{\mathrm{T}} \mathrm{B} \mathrm{V}_1 \mathbf{g}_1 = \lambda_1 \mathrm{I}_2 \mathbf{g}_2 + p_1 \mathrm{I}_2 \mathbf{g}_1 \tag{17}$$

Solving (17) for $\mathbf{g}_2$ yields:

$$\mathbf{g}_2 = \Phi_1 \left(\mathrm{V}_1^{\mathrm{T}} \mathrm{B} \mathrm{V}_1 - \mathbf{v}_1^{\mathrm{T}} \mathrm{B} \mathbf{v}_1 \mathrm{I}_2 \right) \mathbf{g}_1$$

The second-order eigenvector perturbation is therefore:

$$\epsilon^2 \mathrm{V}_1 \mathbf{g}_2 = \mathrm{V}_1 \Phi_1 \left\{ \left(\mathrm{V}_1^{\mathrm{T}} \Delta \mathrm{D} \mathrm{V}_1 - \mathbf{v}_1^{\mathrm{T}} \Delta \mathrm{D} \mathbf{v}_1 \right) \Phi_1 \mathrm{V}_1^{\mathrm{T}} \Delta \mathrm{D} \right\} X \tag{18}$$

We now want to consider the expected values of these second-order perturbations which will be nonzero, unlike the expected values of the first-order perturbations. There is no elegant method to accomplish this, other than multiplying out individual elements of the matrices within parentheses in (16) and (18). The resulting matrix entries contain terms of the form $\delta d_{ij} \delta d_{km}$. The expected values of these are simply the appropriate entries from the covariance matrix of the least-squares vector $\mathbf{d}$. Thus, the expected values of the second-order eigenvalue and eigenvector perturbations can be found to be:

$$\begin{aligned} E[\delta^2 \lambda_1] &= \mathbf{v}_1^{\mathrm{T}} \mathrm{E}_1 \mathbf{v}_1 \\ E[\delta^2 \mathbf{v}_1] &= \mathrm{G}_1 \mathrm{E}_1 \mathbf{v}_1 - \mathrm{G}_1 \mathrm{G}_1 \mathrm{F}_1 \mathbf{v}_1 \end{aligned}$$

where the following matrices have been defined:

$$\begin{aligned} \mathrm{E}_1 &\equiv \left[\; e_{ij} \; \right] \\ \mathrm{F}_1 &\equiv \left[\; f_{ij} \; \right] \\ \mathrm{G}_1 &\equiv \mathrm{V}_1 \Phi_1 \mathrm{V}_1^{\mathrm{T}} \end{aligned}$$

Note that E_1 is a symmetric 3×3 matrix. The $\sigma_{i,j}$'s are elements in the covariance matrix of the least-squares vector $\mathbf{d}$.

$$\begin{aligned} e_{1,1} &= g_{1,1}\sigma_{1,1} + g_{2,2}\sigma_{2,2} + g_{3,3}\sigma_{3,3} + 2g_{1,2}\sigma_{1,2} + 2g_{1,3}\sigma_{1,3} + 2g_{2,3}\sigma_{2,3} \\ e_{1,2} &= g_{1,1}\sigma_{1,2} + g_{1,2}\sigma_{2,2} + g_{1,3}\sigma_{2,3} + g_{2,1}\sigma_{1,5} + g_{2,2}\sigma_{2,5} + g_{2,3}\sigma_{3,5} \end{aligned}$$

$$+ g_{3,1}\sigma_{1,6} + g_{3,2}\sigma_{2,6} + g_{3,3}\sigma_{3,6}$$

$$e_{1,3} = g_{1,1}\sigma_{1,3} + g_{1,2}\sigma_{2,3} + g_{1,3}\sigma_{3,3} + g_{2,1}\sigma_{1,6} + g_{2,2}\sigma_{2,6} + g_{2,3}\sigma_{3,6}$$
$$+ g_{3,1}\sigma_{1,9} + g_{3,2}\sigma_{2,9} + g_{3,3}\sigma_{3,9}$$

$$e_{2,2} = g_{1,1}\sigma_{2,2} + g_{2,2}\sigma_{5,5} + g_{3,3}\sigma_{6,6} + 2g_{1,2}\sigma_{2,5} + 2g_{1,3}\sigma_{2,6} + 2g_{2,3}\sigma_{5,6}$$

$$e_{2,3} = g_{1,1}\sigma_{2,3} + g_{1,2}\sigma_{5,3} + g_{1,3}\sigma_{6,3} + g_{2,1}\sigma_{2,6} + g_{2,2}\sigma_{5,6} + g_{2,3}\sigma_{6,6}$$
$$+ g_{3,1}\sigma_{2,9} + g_{3,2}\sigma_{5,9} + g_{3,3}\sigma_{6,9}$$

$$e_{3,3} = g_{1,1}\sigma_{3,3} + g_{2,2}\sigma_{6,6} + g_{3,3}\sigma_{9,9} + 2g_{1,2}\sigma_{3,6} + 2g_{1,3}\sigma_{3,9} + 2g_{2,3}\sigma_{6,9}$$

F_1 is also a symmetric 3×3 matrix.

$$f_{1,1} = v_x^2\sigma_{1,1} + v_y^2\sigma_{1,5} + v_z^2\sigma_{1,9} + 2v_xv_y\sigma_{1,2} + 2v_xv_z\sigma_{1,3} + 2v_yv_z\sigma_{1,6}$$

$$f_{1,2} = v_x^2\sigma_{2,1} + v_y^2\sigma_{2,5} + v_z^2\sigma_{2,9} + 2v_xv_y\sigma_{2,2} + 2v_xv_z\sigma_{2,3} + 2v_yv_z\sigma_{2,6}$$

$$f_{1,3} = v_x^2\sigma_{3,1} + v_y^2\sigma_{3,5} + v_z^2\sigma_{3,9} + 2v_xv_y\sigma_{3,2} + 2v_xv_z\sigma_{3,3} + 2v_yv_z\sigma_{3,6}$$

$$f_{2,2} = v_x^2\sigma_{5,1} + v_y^2\sigma_{5,5} + v_z^2\sigma_{5,9} + 2v_xv_y\sigma_{5,2} + 2v_xv_z\sigma_{5,3} + 2v_yv_z\sigma_{5,6}$$

$$f_{2,3} = v_x^2\sigma_{6,1} + v_y^2\sigma_{6,5} + v_z^2\sigma_{6,9} + 2v_xv_y\sigma_{6,2} + 2v_xv_z\sigma_{6,3} + 2v_yv_z\sigma_{6,6}$$

$$f_{3,3} = v_x^2\sigma_{9,1} + v_y^2\sigma_{9,5} + v_z^2\sigma_{9,9} + 2v_xv_y\sigma_{9,2} + 2v_xv_z\sigma_{9,3} + 2v_yv_z\sigma_{9,6}$$

References

[1] G. Adiv. "Determining three-dimensional motion and structure from optical flow generated by several moving objects". *IEEE Trans. on Pattern Analysis and Machine Intelligence*, 7(4):384–401, 1985.

[2] P. Anandan. "A computational framework and an algorithm for the measurement of visual motion". *Int. Journal of Computer Vision*, 2(3):283–310, 1989.

[3] J. Barron, D. Fleet, and S. Beauchemin. "Performance of optical flow techniques". *Int. Journal of Computer Vision*, 12(1):43–77, 1994.

[4] S. D. Blostein and R. M. Chann. "The use of image plane velocity measurements in recursive 3-D motion estimation from a monocular image sequence". In *1994 Canadian Conference on Electrical and Computer Engineering*, volume II, pages 797–801, 1994.

[5] T. J. Broida, S. Chandrashekhar, and R. Chellappa. "Recursive 3-D motion estimation from a monocular image sequence". *IEEE Trans. on Aerospace & Electronic Systems*, 26(4):639–656, 1990.

[6] T. J. Broida and R. Chellappa. "Estimation of object motion parameters from noisy images". *IEEE Trans. on Pattern Analysis and Machine Intelligence*, 8(1):90–99, 1986.

[7] T. J. Broida and R. Chellappa. "Estimating the kinematics and structure of a rigid object from a sequence of monocular images". *IEEE Trans. on Pattern Analysis and Machine Intelligence*, 13(6):497–513, 1991.

[8] A. R. Bruss and B. K. P. Horn. "Passive navigation". *Computer Vision, Graphics, and Image Processing*, 21(1):3–20, 1983.

[9] K.-C. Chang, H.-J. Lee, and C.-G. Chung. "Adaptively estimating motion parameters for tracking a maneuvering target using images". *Int. Journal of Robotics & Automation*, 4(1):43–52, 1989.

[10] D. J. Fleet. *Measurement of Image Velocity.* Kluwer Academic Publishers, 1992.

[11] D. B. Gennery. "Visual tracking of known three-dimensional objects". *Int. Journal of Computer Vision*, 7(3):243–270, 1992.

[12] J. D. Gorman and A. O. Hero. "Lower bounds for parametric estimation with constraints". *IEEE Trans. on Information Theory*, 36(6):1285–1301, 1990.

[13] D. J. Heeger and A. D. Jepson. "Subspace methods for recovering rigid motion I: Algorithm and implementation". *Int. Journal of Computer Vision*, 7(2):95–117, 1992.

[14] B. K. P. Horn. *Robot Vision.* McGraw-Hill Book Company, 1986.

[15] A. D. Jepson and D. J. Heeger. "A fast subspace algorithm for recovering rigid motion". In *IEEE Workshop on Visual Motion (1991)*, pages 124–131, 1991.

[16] A. D. Jepson and D. J. Heeger. "Linear subspace methods for recovering translational direction". Technical Report RBCV-TR-92-40, University of Toronto, April 1992.

[17] A. D. Jepson and D. J. Heeger. "Linear subspace methods for recovering translational direction". In L. Harris and M. Jenkin, editors, *Spatial Vision in Humans and Robots*, pages 39–62. Cambridge University Press, 1993.

[18] C. Jerian and R. Jain. "Polynomial methods for structure from motion". *IEEE Trans. on Pattern Analysis and Machine Intelligence*, 12(12):1150–1166, 1990.

[19] K. Kanatani. "Renormalization for unbiased estimation". In *4th International Conference on Computer Vision 1993*, pages 599–606, 1993.

[20] M. R. Luettgen, W. C. Karl, and A. S. Willsky. "Efficient multiscale regularization with applications to the computation of optical flow". *IEEE Trans. on Image Processing*, 3(1):41–64, 1994.

[21] W. J. MacLean, 1994. private communication.

[22] P. S. Maybeck. *Stochastic Models, Estimation, and Control*, volume 1. Academic Press, Inc., 1979.

[23] H. Shariat and K. E. Price. "Motion estimation with more than two frames". *IEEE Trans. on Pattern Analysis and Machine Intelligence*, 12(5):417–434, 1990.

[24] E. P. Simoncelli, E. H. Adelson, and D. J. Heeger. "Probability distributions of optical flow". In *1991 IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, pages 310–315, 1991.

[25] G. J. Tseng and A. K. Sood. "Analysis of long image sequence for structure and motion estimation". *IEEE Trans. on Systems, Man, and Cybernetics*, 19(6):1511–1526, 1989.

[26] S. Uras, F. Girosi, A. Verri, and V. Torre. "A computational approach to motion perception". *Biological Cybernetics*, 60(1):79–87, 1988.

[27] J. Weng, T. S. Huang, and N. Ahuja. "3-D motion estimation, understanding, and prediction from noisy image sequences". *IEEE Trans. on Pattern Analysis and Machine Intelligence*, 9(3):370–389, 1987.

[28] J. Weng, T. S. Huang, and N. Ahuja. "Motion and structure from two perspective views: Algorithms, error analysis, and error estimation". *IEEE Trans. on Pattern Analysis and Machine Intelligence*, 11(5):451–476, 1989.

[29] J. Wilkinson. *The Algebraic Eigenvalue Problem.* Oxford University Press, 1965.

[30] Y. Yasumoto and G. Medioni. "Robust estimation of three-dimensional motion parameters from a sequence of image frames using regularization". *IEEE Trans. on Pattern Analysis and Machine Intelligence*, 8(4):464–471, 1986.

[31] G. Young, R. Chellappa, and T. Wu. "Monocular motion estimation using a long sequence of noisy images". In *1991 International Conference on Acoustics, Speech, and Signal Processing*, pages 2437–2440, 1991.

[32] G.-S. J. Young and R. Chellappa. "3-D motion estimation using a sequence of noisy stereo images: Models, estimation, and uniqueness results". *IEEE Trans. on Pattern Analysis and Machine Intelligence*, 12(8):735–759, 1990.