Three Algorithms for 2–Image and $\geq$ 2–Image Structure from Motion

John Oliensis (oliensis@research.nj.nec.com)
NEC Research Institute
4 Independence Way
Princeton, N.J. 08540
and
Yacup Genc
Siemens Research
Princeton, N.J. 08540

Abstract

We describe three approaches to 2–image and $\geq$ 2–image structure from motion. First, we present a new approximation to the least–squares image–reprojection error for 2 images. It depends only on the motion unknowns and is much more accurate than previous approximations such as the (weighted) coplanarity, especially for forward camera motions. We use this error to compute tight, rigorous upper and lower bounds on the true error and to study its properties experimentally. We demonstrate that the true error has many local minima for forward motions even when the motion is large. We propose and experimentally test a second approach, which is potentially more robust than bundle adjustment. Last, we describe algorithms for $\geq$ 2 images that reconstruct from the measured *2D* affine deformations of image patches.

Keywords: structure–from–motion, two–image structure from motion, least–squares error,rotational invariance, angular error, local minima, affine image flow.

1 Introduction

Two–image structure from motion (SFM) turns out to be surprisingly accurate and robust [40][29][25][37][36], and [26] has recently pointed out that it has advantages over reconstructing from more than two images. Thus, it is important to develop good two–image algorithms.

Current two–image approaches are slow or nonoptimal. The most accurate approach minimizes the image reprojection error in the structure as well as the motion, which makes it slow, since the minimization is in many variables. Another standard and faster approach minimizes the weighted coplanarity error in the five motion unknowns (see, e.g., [29][40]). We refer to this as *Algorithm WC*. Since it minimizes an approximation to the true error, it is nonoptimal, but [29][40] have shown that it often gives nearly optimal results, at least when the camera translates in a direction different enough from the viewing direction.

In this paper, we propose three approaches to 2–image and $\geq$ 2–image SFM. First, we present an approximation to the true least–squares error for two images which is much more accurate than the weighted coplanarity (WC) error and which depends only on the five motion unknowns. Minimizing the new approximate error gives an algorithm that is as fast as *Algorithm WC* and more accurate. The new error is a good approximation even for forward camera motions, which is important, since we show that the true error is often most complex for forward motions (see also [25][4]). One can use the new approximation to get tight, easily computable, and rigorous bounds on the true error. We use this to study the true error experimentally, without needing to calculate it explicitly, and give the first demonstration that it has many local minima for forward motions even when the true motion is large.

Second, we present an algorithm which applies either to two images or more than two. It has an accuracy comparable to *Algorithm WC,* but it is potentially more robust even than bundle adjustment, and it is potentially faster than *Algorithm WC*. Motivated partly by its usefulness as a multi–image approach, we experimentally study its accuracy and range of applicability.

Last, we present a new approach that reconstructs from the measured *2D* affine deformations of image patches. ([2][19][18] have described approaches that reconstruct from the affine deformation plus the *affine parallax* ([19]), which in effect requires computing a second derivative of the motion

flow. Our technique only requires measuring first derivatives.) This approach, which is really a class of algorithms, applies to two or more than two images. One motivation for the approach is that image regions where the intensity changes smoothly can account for a large fraction of the image, and often one can only recover the affine deformations of such regions. Also, one can measure the affine deformations more robustly than higher order properties of the correspondence or motion flow.

We described some of these results in [23].

1.1 Overview of the Paper

Section 2.1 derives the standard *WC* error and estimates how well it approximates the true error. Section 2.2 introduces the *angular* least–squares error and derives our new approximation to this error. Section 3 uses our approximation to study the shape of the angular least–squares error surface for large motion.

Section 4 motivates and describes our second algorithm, and Section 4.3 reports our experiments with it. Finally, Section 5 describes our approach to reconstructing from affine deformations, and Section 6 concludes.

1.2 Initial Definitions

Without loss of generality, we take the focal length to be 1. We use MATLAB notation: a semi-colon separates entries in a coumn vector, a comma or space separates entries in a row vector, and a colon indicates a range of indices. Let the column vector $\mathbf{p}_{im} \equiv (x;y)^i_m \equiv (x^i_m;y^i_m)$ denote the m-th point in the i-th image, where $m = \{1,2,..N_p\}$ and $i = \{0,1\}$. Let $\mathbf{p}_m \equiv (x;y)_m \equiv (x^0_m;y^0_m) = \mathbf{p}_{0m}$. Define $\mathbf{P}_m \equiv \begin{pmatrix} X_m; & Y_m; & Z_m \end{pmatrix}$ to be the m–th 3D point in the coordinate system of the zeroth image.

Let $\mathbf{T} \equiv (T_x;T_y;T_z)$ be the translation and $\hat{\mathbf{T}} \equiv \mathbf{T}/|\mathbf{T}|$ the translation direction, and let R be the rotation. We define the motion of the 3D point $\mathbf{P}$ under R and $\mathbf{T}$ by $\mathbf{P'} = R(\mathbf{P} - \mathbf{T})$. Let F represent the essential matrix: F acting on a 3D vector $\mathbf{V}$ gives $F\mathbf{V} = R(\hat{\mathbf{T}} \times \mathbf{V})$.

Given a vector $\mathbf{V}$, define $[\mathbf{V}]_2$ as the length–2 vector consisting of the first two components of $\mathbf{V}$. Similarly, for a matrix M, define $[M]_2$ as the 2×2 submatrix consisting of entries for the first

two indices. Let $\underline{\mathbf{V}}$ denote the 2D image point corresponding to the 3D point $\mathbf{V}$: $\underline{\mathbf{V}} \equiv [\mathbf{V}]_2 / V_z$. For 2D vectors $\mathbf{v}_1$, $\mathbf{v}_2$, we use the notation $\mathbf{v}_1 \times \mathbf{v}_2$ to mean $v_{1x}v_{2y} - v_{1y}v_{2x}$. Define the 3D point $\bar{\mathbf{v}}_1$ corresponding to $\mathbf{v}_1$ by $\bar{\mathbf{v}}_1 \equiv \left[\begin{array}{cc} \mathbf{v}_1; & 1 \end{array} \right]$. Let $R * \mathbf{v}_1$ denote the image point obtained from $\mathbf{v}_1$ after a rotation: $R * \mathbf{v}_1 \equiv \underline{(R\bar{\mathbf{v}}_1)}$.

Let $\mathbf{e} \equiv \underline{\mathbf{T}}$ denote the epipole in the zeroth image. Let $\theta_{\rm FOV}$ be an angle in radians characterizing the field of view (FOV); for simplicity, we assume the FOV is a circle centered on the view direction. Let E_T denote the least–squares image–reprojection error as a function of $\hat{\mathbf{T}}$ after minimizing over R.

2 Approximations to the Least–Squares Error

2.1 The Weighted–Coplanarity Error

We derive the standard weighted–coplanarity approximation to the least–square error [41][29] and present a new estimate of its goodness. Let the motion be given, and consider the m–th point. We denote the error for this point, the differences between the image projections of $\mathbf{P}_m$ and the observed image points, by the length–4 column vector $\boldsymbol{\delta}_m \equiv \left(\underline{\mathbf{P}_m} - \mathbf{p}_{0m}; \underline{R\left(\mathbf{P}_m - \mathbf{T}\right)} - \mathbf{p}_{1m} \right)$. Define E_m as the result of minimizing the square of this error over $\mathbf{P}_m$: $E_m \equiv \inf_{\mathbf{P}_m} |\boldsymbol{\delta}_m|^2$. (The infimum or inf is the greatest lower bound.) Equivalently,

$$E_m = \inf_{\boldsymbol{\delta}_m} |\boldsymbol{\delta}_m|^2 \tag{1}$$

where one minimizes over $\boldsymbol{\delta}_m$ subject to the coplanarity constraint

$$C_m + \mathbf{L}_m \cdot \boldsymbol{\delta}_m + \boldsymbol{\delta}_m^T Q \boldsymbol{\delta}_m = 0,$$

and where $C_m \equiv \bar{\mathbf{p}}_{1m} F \bar{\mathbf{p}}_{0m}$ is the coplanarity error for the *observed* points $\mathbf{p}_{\{0,1\}m}$,

$$Q \equiv \frac{1}{2} \left[\begin{array}{cc} 0 & [F]_2^T \\ [F]_2 & 0 \end{array} \right],$$

and $\mathbf{L}_m \equiv \left(\left[F^T \bar{\mathbf{p}}_{1m} \right]_2 ; [F \bar{\mathbf{p}}_{0m}]_2 \right)$.

Following [41], we compute the minimum by finding the stationary points of

$$|\boldsymbol{\delta}_m|^2 + \lambda_m \left(C_m + \mathbf{L}_m \cdot \boldsymbol{\delta}_m + \boldsymbol{\delta}_m^T Q \boldsymbol{\delta}_m \right),$$

where λ_m is a lagrangian parameter. Differentiating this with respect to $\boldsymbol{\delta}_m$ gives

$$\boldsymbol{\delta}_m = -\frac{(1+\lambda_m Q)^{-1}\lambda_m \mathbf{L}_m}{2},$$

$$E_m = \frac{\lambda_m^2}{4}\mathbf{L}_m^T (1+\lambda_m Q)^{-2}\mathbf{L}_m, \tag{2}$$

where one can solve for λ_m by enforcing the constraint

$$C_m - \frac{1}{2}\mathbf{L}_m^T (1+\lambda_m Q)^{-1}\lambda_m \mathbf{L}_m + \frac{\lambda_m^2}{4}\mathbf{L}_m^T (1+\lambda_m Q)^{-1} Q (1+\lambda_m Q)^{-1}\mathbf{L}_m = 0. \tag{3}$$

Note that $C_m \sim O(\eta) \ll 1$, where $O(\eta)$ represents the size of the noise. We now expand the above expressions in powers of C_m. To lowest order,

$$\lambda_m \approx \lambda_{0m} \equiv \frac{2C_m}{|\mathbf{L}_m|^2},$$

$$E_m \approx \frac{\lambda_m^2}{4}|\mathbf{L}_m|^2 \approx \frac{C_m^2}{|\mathbf{L}_m|^2} \equiv E_{0m}.$$

E_{0m} is the standard weighted–coplanarity error used, e.g., in [41][29].

We obtain the first–order corrections E_{1m}, λ_{1m} by expanding (2) and the constraint (3) to the next lowest order. This gives

$$E_{1m} \approx E_m - E_{0m} \approx \frac{2\lambda_{0m}\lambda_{1m}}{4}|\mathbf{L}_m|^2 - \frac{\lambda_{0m}^3}{2}\mathbf{L}_m^T Q \mathbf{L}_m, \tag{4}$$

$$0 \approx C_m - \frac{\lambda_m |\mathbf{L}_m|^2}{2} + \frac{\lambda_m^2}{2}\mathbf{L}_m^T Q \mathbf{L}_m + \frac{\lambda_m^2}{4}\mathbf{L}_m^T Q \mathbf{L}_m. \tag{5}$$

Substituting $\lambda_m \longrightarrow 2C/|\mathbf{L}_m|^2 + \lambda_{1m} + O(C_m^3)$ into (5) gives

$$\frac{\lambda_{1m}\mathbf{L}_m^T\mathbf{L}_m}{2} = \frac{3\lambda_{0m}^2}{4}\mathbf{L}_m^T Q \mathbf{L}_m = \frac{3C_m^2}{|\mathbf{L}_m|^2}\mathbf{L}_m^T Q \mathbf{L}_m$$

$$\lambda_{1m} = \frac{6C_m^2}{|\mathbf{L}_m|^3}\mathbf{L}_m^T Q \mathbf{L}_m.$$

Substituting λ_{1m} into (4) gives

$$E_{1m} = 2C_m^3 \frac{\mathbf{L}_m^T Q \mathbf{L}_m}{|\mathbf{L}_m|^3}.$$

Thus, the weighted–coplanarity error approximates the true error up to a relative factor of $O\left(C\mathbf{L}^T Q\mathbf{L}/\left|\mathbf{L}\right|\right) \sim O\left(\eta\mathbf{L}^T Q\mathbf{L}/\left|\mathbf{L}\right|\right)$.

Note that Q vanishes when both epipoles are at infinity, so that the first–order correction E_1 vanishes, in agreement with the result in [41] that the *WC* error becomes exact. For forward motion, typically $\mathbf{L}^T Q\mathbf{L} \sim |\mathbf{L}| \sim O(1)$, so the *WC* error is correct up to a relative error of $O(\eta)$.

In the next section, we define an approximate error that is correct up to a relative error $O\left(\eta^2\right)$.

2.2 Angular Error

We represent the m–th image points in the zeroth and first image by the length–3 *unit* vectors $\hat{\mathbf{p}}_{0m}$ and $\hat{\mathbf{p}}_{1m}$. Consider the *angular* least–squares error

$$\begin{aligned} \boldsymbol{\delta}_{\theta,m} &\equiv \left(\frac{\mathbf{P}_m}{|\mathbf{P}_m|} - \hat{\mathbf{p}}_{0m}; \frac{R\left(\mathbf{P}_m - \mathbf{T}\right)}{\left|R\left(\mathbf{P}_m - \mathbf{T}\right)\right|} - \hat{\mathbf{p}}_{1m}\right), \\ E_{\theta,m} &\equiv \inf_{P_m} |\boldsymbol{\delta}_{\theta,m}|^2, \;\; E_\theta\left(R,\mathbf{T}\right) \equiv \sum_m E_{\theta,m}\left(R,\mathbf{T}\right). \end{aligned} \tag{6}$$

By using this form of the error, one implicitly assumes that the error in the *direction* of a measured image point is independent of the point's position.[1] When using the standard image–plane error (1), one implicitly assumes this for the *pixel error* in the image plane.

The two forms of the error are nearly equal for small FOV, with $\theta_{\mathrm{FOV}} < 1$, and it is not clear which is more realistic for larger FOV. For real cameras, the image noise depends on the lens and on the properties of the world (its texture, contrast, etc.) as well as on the surface on which the camera focuses the image. Though the standard pixel–based error gives a good model of image formation onto a plane, a direction–based error gives a better model of the world and lens. For instance, the image–plane error implies that the angular resolution of image measurements becomes arbitrarily good for image points at large angles to the viewing direction, which is inconsistent with the properties of physical lenses. Taking the angular resolution to be constant, as (6) does, gives a much better approximation to the physics of lenses. Also, if the camera has a general position and general orientation, it is more realistic to assume that the scene features have the same angular

[1] Instead of (6), one can use use $E_{LS\theta} \equiv \sum_m w_m E_{LS\theta,m}$, with a different weighting w_m for each 3D point, without changing any of the subsequent analysis.

scale at different orientations than to assume that they have the same pixel scale. The properties of the world do not depend on the imaging surface inside the camera!

2.2.1 Approximating $E_{\theta,m}(R, \mathbf{T})$

Let the motion R and $\mathbf{T}$ be given, and consider the m–th point. Let $\hat{\mathbf{p}}'_{1m} \equiv R^{-1}\hat{\mathbf{p}}_{1m}$ be the unrotated point in image 1. Rewrite the $\hat{\mathbf{p}}_{1m}$–dependent term in the error (6) as

$$\left|(\mathbf{P}_m - \mathbf{T}) / |\mathbf{P}_m - \mathbf{T}| - R^{-1}\hat{\mathbf{p}}_{1m}\right|^2 = \left|(\mathbf{P}_m - \mathbf{T}) / |\mathbf{P}_m - \mathbf{T}| - \hat{\mathbf{p}}'_{1m}\right|^2$$

(This simplifying step is our main reason for using the angular error.) We organize the minimization of (6) over $\mathbf{P}_m$ by first fixing a unit vector $\hat{\mathbf{n}}_m$ and minimizing over all $\mathbf{P}_m$ with $\hat{\mathbf{P}}_m \cdot \hat{\mathbf{n}}_m = \mathbf{0}$—i.e., over all $\mathbf{P}_m$ on the great circle perpendicular to $\hat{\mathbf{n}}_m$—and then minimizing the results over $\hat{\mathbf{n}}_m$. We follow the standard practice of neglecting the constraint that the depths are positive.

We first assume that $\mathbf{P}_m \not\sim \mathbf{T}$. Then, for fixed $\hat{\mathbf{n}}_m$, one can select $\hat{\mathbf{b}}_m = (\mathbf{P}_m - \mathbf{T}) / |\mathbf{P}_m - \mathbf{T}|$ *independently* of $\hat{\mathbf{a}}_m = P_m / |\mathbf{P}_m|$, essentially anywhere on the great circle perpendicular to $\hat{\mathbf{n}}_m$. Minimizing (6) independently over $\hat{\mathbf{a}}_m$ and $\hat{\mathbf{b}}_m$ gives

$$E_{\theta,m} = \inf_{\mathbf{n}_m} 2\left(2 - |\hat{\mathbf{n}}_m \times \hat{\mathbf{p}}_{0m}| - \left|\hat{\mathbf{n}}_m \times \hat{\mathbf{p}}'_{1m}\right|\right). \tag{7}$$

Let $\hat{\mathbf{p}}_{am}$ denote either $\hat{\mathbf{p}}_{0m}$ or $\hat{\mathbf{p}}'_{1m}$. We have $|\hat{\mathbf{n}}_m \times \hat{\mathbf{p}}_{am}| = \cos(\alpha_{ma})$, where α_{ma} is the angle between the measured point $\hat{\mathbf{p}}_{am}$ and the epipolar great circle. For the true $\hat{\mathbf{n}}_m$, α_{ma} is the noise in the measured image–point direction, and, near the minimizing value of $\hat{\mathbf{n}}_m$, we expect $\alpha_{ma} \sim O(\eta) \ll 1$. For small α_{ma}, one can expand

$$2(1 - |\hat{\mathbf{n}}_m \times \hat{\mathbf{p}}_{am}|) \approx |\hat{\mathbf{n}}_m \cdot \hat{\mathbf{p}}_{am}|^2 + O\left(\eta^4\right). \tag{8}$$

For the minimization over $\mathbf{P}_m$, this approximation gives

$$E_{\theta,m} = \underline{E}_{\theta,m} + O\left(\eta^4\right),$$

$$\underline{E}_{\theta,m} \equiv \inf_{\hat{\mathbf{n}} \perp \mathbf{T}} \left(\hat{\mathbf{n}}^T S_{\theta,m} \hat{\mathbf{n}}\right), \quad S_{\theta,m} \equiv \hat{\mathbf{p}}_{0m}\hat{\mathbf{p}}_{0m}^T + \hat{\mathbf{p}}'_{1m}\hat{\mathbf{p}}'^T_{1m}. \tag{9}$$

Since $\hat{\mathbf{n}}_m \perp \mathbf{T}$, one can compute the minimum in (9) exactly as the least eigenvalue of a 2×2

matrix:

$$\begin{aligned} \underline{E}_{\theta,m} &= A_{\theta m}/2 - \sqrt{A_{\theta m}^2/4 - B_{\theta m}}, \qquad (10) \\ A_{\theta m} &\equiv \hat{\mathbf{p}}_{0m}^T \left(1 - \hat{\mathbf{T}}\hat{\mathbf{T}}^T\right) \hat{\mathbf{p}}_{0m} + \hat{\mathbf{p}}_{1m}^T R \left(1 - \hat{\mathbf{T}}\hat{\mathbf{T}}^T\right) R^{-1} \hat{\mathbf{p}}_{1m}, \\ B_{\theta m} &\equiv \left(\hat{\mathbf{T}} \cdot \hat{\mathbf{p}}_{0m} \times R^{-1} \hat{\mathbf{p}}_{1m}\right)^2. \end{aligned}$$

Define $\underline{E}_\theta(R, \mathbf{T}) \equiv \sum_m \underline{E}_{\theta,m}$. (10) and $\underline{E}_\theta$ give our approximation to the least–squares error. We refer to the strategy of minimizing $\underline{E}_\theta(R, \mathbf{T})$ as *Algorithm A1.*

We derived (7) and (10) assuming that $\mathbf{P}_m \not\sim \mathbf{T}$. We now show that the exceptional case $\hat{\mathbf{P}}_m = \hat{\mathbf{T}}$ does not change these results. Fix $\hat{\mathbf{n}}_m \perp \hat{\mathbf{T}}$ as before. The infimum of $|\boldsymbol{\delta}_{\theta,m}|^2$ over all $\mathbf{P}_m \perp \hat{\mathbf{n}}$ is at least as big as

$$2\left(2 - |\hat{\mathbf{n}}_m \times \hat{\mathbf{p}}_{0m}| - \left|\hat{\mathbf{n}}_m \times \hat{\mathbf{p}}'_{1m}\right|\right), \qquad (11)$$

in (7), since we obtained (7) by adding an extra freedom to adjust parameters to lower the error. Also, $|\boldsymbol{\delta}_{\theta,m}|^2$ is clearly a continuous function of $\hat{\mathbf{a}}_m$ and $\hat{\mathbf{b}}_m$, and for any values $\hat{\mathbf{a}}_m, \hat{\mathbf{b}}_m \neq \hat{\mathbf{T}}$, one can find a $\mathbf{P}_m$ that yields these values. Consider an infinite sequence of values for $\hat{\mathbf{a}}_m, \hat{\mathbf{b}}_m \neq \hat{\mathbf{T}}$ such that $|\boldsymbol{\delta}_{\theta,m}|^2$ converges to (11). There exists a corresponding sequence of values for $\mathbf{P}_m$ such that $|\boldsymbol{\delta}_{\theta,m}|^2$ converges to (11) as a function of $\mathbf{P}_m$. This shows that the infimum of $|\boldsymbol{\delta}_{\theta,m}|^2$ with respect to $\mathbf{P}_m$ for fixed $\hat{\mathbf{n}}_m$ is less than or equal to (11). Combining this with the previous inequality in the opposite direction, we find that the infimum equals (11).

2.2.2 Goodness of the approximation (10).

The approximation $|\hat{\mathbf{n}} \cdot \hat{\mathbf{p}}_{am}|^2 \leq 2(1 - |\hat{\mathbf{n}} \times \hat{\mathbf{p}}_{am}|)$. Thus, $\underline{E}_{\theta,m} \leq E_{\theta,m}$ and

$$\underline{E}_{\theta,T} \equiv \inf_R \sum_m \underline{E}_{\theta,m} \leq \inf_R \sum_m E_{\theta,m}(R, \mathbf{T}) \equiv E_{\theta,T}.$$

i.e., $\underline{E}_{\theta,T}$ gives a lower bound on the true error $E_{\theta,T}$. One can get an upper bound on $E_{\theta,T}$ by plugging the values of $\hat{\mathbf{n}}_m$ and R giving the minimum of $\underline{E}_{\theta,T}$ into $\sum_m E_{\theta,m}(R, \mathbf{T})$. Thus, *one can use our approximation (10) to compute strict upper and lower bounds on the true error.*

Instead of computing and plugging in the $\hat{\mathbf{n}}_m$, one can get a larger–than–necessary but easy–to–compute upper bound on $E_{\theta,m}$ as follows. Assume we have computed the minimizing rotation $R_{\min} \equiv \arg\min_R \sum_m \underline{E}_{\theta,m}(R, \mathbf{T})$. Since $2(1 - |\hat{\mathbf{n}}_m \times \hat{\mathbf{p}}_{am}|)$ is a concave–up function of $|\hat{\mathbf{n}}_m \cdot \hat{\mathbf{p}}_{am}|^2$, it is easy to show that

$$E_{\theta,m}(R_{\min}, \mathbf{T}) = 2(1 - |\hat{\mathbf{n}}_m \times \hat{\mathbf{p}}_{0m}|) + 2\left(1 - \left|\hat{\mathbf{n}}_m \times \hat{\mathbf{p}}'_{1m}(R_{\min})\right|\right) \leq 2\left(1 - \sqrt{1 - \underline{E}_{\theta,m}(R_{\min}, \mathbf{T})}\right)$$

$$\longrightarrow \quad E_{\theta,T} \leq 2\sum_m \left(1 - \sqrt{1 - \underline{E}_{\theta,m}(R_{\min}, \mathbf{T})}\right).$$

We use this upper bound in our experiments below.

For *all* α_a, the ratio of the approximation to the true error is larger than $1/2$, that is, $1 \geq |\hat{\mathbf{n}} \cdot \hat{\mathbf{p}}_{am}|^2 / 2(1 - |\hat{\mathbf{n}} \times \hat{\mathbf{p}}_{am}|) \geq 1/2$. Even for very large $\alpha_a \leq 53°$, the approximation has a relative error of less than 20%, and for $\alpha_a \leq 11.5°$, the relative error is less than 1%. For a more typical and reasonable noise of $1°$, the *relative* error is just 7.6×10^{-5}. Thus, our approximations (8) and (10) are extremely good *even for forward motion*, in contrast with the standard approximation used in [40][29]. This is important, since the error is most complex for forward motion, as we show below [4][25].

2.2.3 Exact Structure Estimate

Given the motion, [6] showed that one could compute the exact estimate of the structure from (1) by finding the zeros of a 6–th degree polynomial. For the exact angular error $E_\theta(\mathbf{T}, R)$, one can also compute the exact structure by solving a 6–th degree polynomial.

Without loss of generality, take $\hat{\mathbf{T}} = \hat{\mathbf{z}}$ and write $\hat{\mathbf{n}} = (\ \cos\theta;\quad \sin\theta;\quad 0\ )$. Define

$$(\sin\alpha_0 \cos\beta_0; \sin\alpha_0 \sin\beta_0; \cos\alpha_0) \equiv \hat{\mathbf{p}}_{0m}$$

and $(\sin\alpha_1 \cos\beta_1; \sin\alpha_1 \sin\beta_1; \cos\alpha_1) \equiv R^{-1}\hat{\mathbf{p}}_{1m}$. We must maximize the error

$$|\hat{\mathbf{n}} \times \hat{\mathbf{p}}_{0m}| + \left|\hat{\mathbf{n}} \times R^{-1}\hat{\mathbf{p}}_{1m}\right| = \sum_{a=0}^{1} \sqrt{\cos^2\alpha_a + \sin^2\alpha_a \sin^2(\theta - \beta_a)}$$

with respect to $\hat{\mathbf{n}}$. Differentiating with respect to θ gives

$$0 = \sum_a \frac{\sin^2 \alpha_a \sin(\theta - \beta_a) \cos(\theta - \beta_a)}{\sqrt{\cos^2 \alpha_a + \sin^2 \alpha_a \sin^2(\theta - \beta_a)}} = \frac{1}{2} \sum_a \frac{\sin^2 \alpha_a \sin 2(\theta - \beta_a)}{\sqrt{1 - \frac{1}{2} \sin^2 \alpha_a (1 + \cos 2(\theta - \beta_a))}}$$

Defining $\theta' \equiv \theta - \beta_0$ and $\delta\beta \equiv \beta_1 - \beta_0$ gives

$$0 = \frac{\sin^2 \alpha_0 \sin 2\theta'}{\sqrt{1 - \frac{1}{2} \sin^2 \alpha_0 \left(1 + \cos 2\theta'\right)}} + \frac{\sin^2 \alpha_1 \sin 2\left(\theta' - \delta\beta\right)}{\sqrt{1 - \frac{1}{2} \sin^2 \alpha_1 \left(1 + \cos 2\left(\theta' - \delta\beta\right)\right)}}.$$

Let $x \equiv \cos\left(2\theta'\right)$. Squaring gives

$$\frac{\sin^4 \alpha_0 \left(1 - x^2\right)}{1 - \frac{1}{2} \sin^2 \alpha_0 (1 + x)} = \frac{\sin^4 \alpha_1 \left(1 - \left(x \cos(2\delta\beta) + \left(1 - x^2\right)^{1/2} \sin(2\delta\beta)\right)^2\right)}{1 - \frac{1}{2} \sin^2 \alpha_1 \left(1 - x \cos(2\delta\beta) - (1 - x^2)^{1/2} \sin(2\delta\beta)\right)}.$$

Simplifying gives a polynomial of degree 6 in x.

3 Experiments: the Least–Squares Error Surface

This section applies our approximation and bounds to study the least–squares error surface experimentally. Since we previously analyzed the surface for sideways translations in [25], we focus here on the shape of the surface for forward motion. (That is, we study the surface for candidate translations that are forward; the true translation can be in any direction). We demonstrate that the forward–motion error surface is often complex and has local minima. [4] recently verified this experimentally for infinitesimal motion, confirming our predictions in [25], but this paper gives the first demonstration for large motions. What makes our study feasible is that we avoid a costly minimization over the structure unknowns but still obtain exact, tight bounds on the true error.

We report results for noiseless synthetic sequences generated using the measured ground–truth structure from two real motion sequences: the UMASS/Martin–Marietta rocket–field sequence [5] and the UMASS puma sequence [17]. The depths for the Rocket–Field sequence vary from 17 to 67, and for the PUMA sequence they vary from 13–32. We use a variety of translations and, for simplicity, take the rotation to be zero. (We set the *true* rotation to be zero; the error surfaces in the figures incorporate a minimization over candidate rotations. [25] argues that the qualitative

Structure	$\mathbf{T}$	$\underline{E}_{\theta,T}$ Range	$\max(\Delta E_{\theta,T})$	$\max(\Delta E_{\theta,T}/\underline{E}_{\theta,T})$	Minima (#	Goodness)
Rocket	$(3,0,3)^T$	$[2\times10^{-4}, 6.2\times10^{-3}]$	2×10^{-6}	6×10^{-4}	2	10^3
Rocket	$(1,0.5,2.5)^T$	$[2\times10^{-3}, 2\times10^{-4}]$	1.5×10^{-7}	8×10^{-5}	2	10^3
PUMA	$(2,3,1)^T$	$[1\times10^{-3}, 1.6\times10^{-2}]$	4×10^{-6}	3×10^{-4}	3	10^2
PUMA	$(-1,1,3)^T$	$[0, 1\times10^{-2}]$	1.5×10^{-6}	1.8×10^{-4}	1	10^2
PUMA	$(4,0,6)^T$	$[4\times10^{-4}, 1.3\times10^{-2}]$,	5×10^{-6}	3.6×10^{-4}	2	10^1

Table 1: Results for 5 synthetic two–frame sequences generated using the ground–truth structure for the Rocket–Field and PUMA sequences. $\underline{E}_{\theta,T}$ is our approximate error, $\Delta E_{\theta,T}$ is the difference between our upper and lower bounds on the true error $E_{\theta,T}$. (The lower bound is $\underline{E}_{\theta,T}$, and the upper bound is as described in Section 2.2.2.) "Goodness" describes the approximate ratio of the scale of the local minima to $\Delta E_{\theta,T}$.

features of the error surface are independent of the true rotation; see [39] for experimental support for this.)

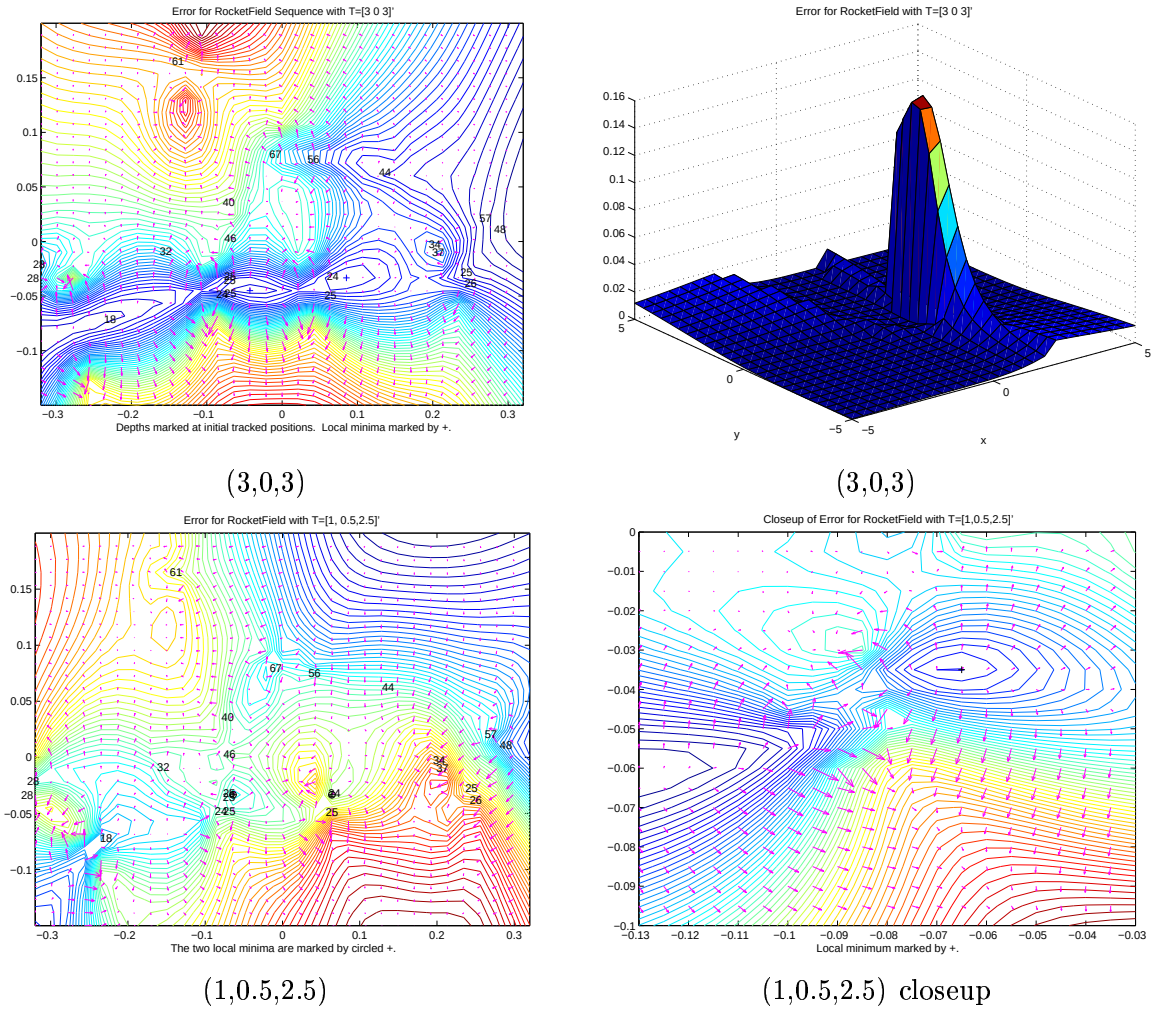
Figure 1 shows results for the Rocket–Field sequence, and Figures 2–3 show results for the PUMA sequence. For the Rocket–Field experiment with $\mathbf{T} = (3,0,3)^T$, the local increase in the error from each local minimum is at least three orders of magnitude bigger than the difference between our upper and lower bounds on the true error. Thus, the true error does have these minima.

Denote the difference between our upper and lower bounds by $\Delta E_{\theta,T}$. While $\underline{E}_{\theta,T}$ in the Figure ranges between 2×10^{-4} and 6.2×10^{-3}, $\Delta E_{\theta,T}$ satisfies $\Delta E_{\theta,T} \leq 2\times10^{-6}$, and $\Delta E_{\theta,T}/\underline{E}_{\theta,T} \leq 6\times10^{-4}$ over this region. Thus, $\underline{E}_{\theta,T}$ gives an excellent approximation to $E_{\theta,T}$.

Table 1 shows similar results for our other experiments. (The number of local minima displayed excludes the global minimum.)

Note that the error is most complex for forward motion candidates. (The trough for large sideways motion in the Rocket–Field error in the upper right plot in Figure 1 reflects the fact that the scene is nearly planar: the eigenvalues of $\sum_m \left(\mathbf{P}_m - \bar{\mathbf{P}}\right)\left(\mathbf{P}_m - \bar{\mathbf{P}}\right)^T$ are 67.4, 31.1, 6.0 for this sequence, where $\bar{\mathbf{P}} \equiv \sum_m \mathbf{P}_m/N_p$, and the relative smallness of the third eigenvalue reveals the scene's planarity. The planar two–fold ambiguity produces a local minimum which would be at $\mathbf{e} \approx (5,12)^T$ for an exactly planar scene, and this causes the trough. For nonplanar scenes, the error is smoother for large sideways motions, as shown for the PUMA sequences in Figures 2,3.

.



(3,0,3) (3,0,3)

(1,0.5,2.5) (1,0.5,2.5) closeup

Figure 1: Contour and surface plots of the error $\underline{E}_{\mathbf{T}}$ as a function of $\mathbf{e}$, with the structure derived from the Rocket–Field sequence. The arrows indicate the local gradient direction; "red" denotes a large error. The error $\underline{E}_{\mathbf{T}}$ has 2 forward local minima for $\mathbf{T} = (3, 0, 3)^T$ and 2 for $T = (1, 0.5, 2.5)^T$. The local minima are marked by '+'. Depths are shown at the positions of the tracked points in the zeroth image.

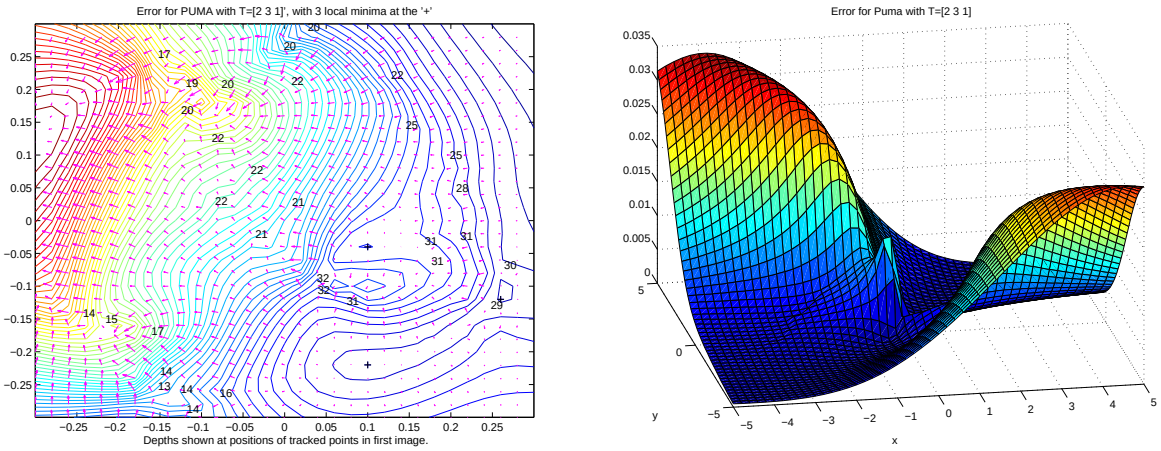
The PUMA sequence has eigenvalues 38.7, 18.6, 9.7.)

4 *Algorithm A2*

The second algorithm we present, which we call *Algorithm A2*, is an extension of the multi–image algorithm of [27][28][33][31]. It works by cycling between recovering the rotation and the translation, but, unlike previous approaches, it handles the errors consistently between the two stages. This guarantees that it converges correctly if it starts near the correct motion [29].

The main contribution of this section is experimental. We address two questions:

.



(a) (b)

Figure 2: Contour and surface plots of the error $\underline{E}_{\mathbf{T}}$ as a function of $\mathbf{e}$, with the structure derived from the PUMA sequence. The arrows indicate the local gradient direction; "red" denotes a large error. The error with $\mathbf{T} = (2, 3, 1)^T$ has 3 forward local minima marked by '+'. Depths are shown at the positions of the tracked points in the zeroth image.

1. Since *Algorithm A2* minimizes approximations to the true least–squares error that are worse than that of (10), how does its performance compare to the optimal one?

2. Since the algorithm is based on a small–motion approximation, how well does it deal with large motions?

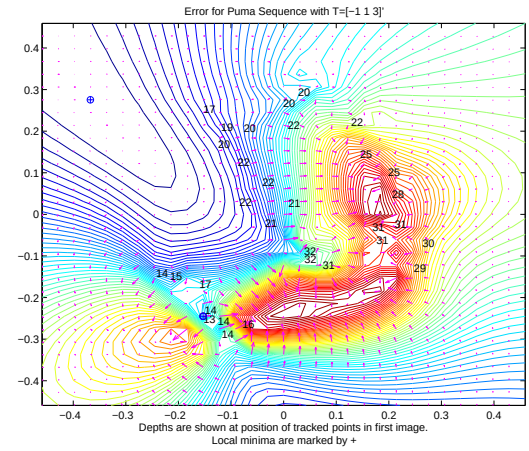
4.1 Discussion and Motivation

Despite its approximations, *Algorithm A2* has potential advantages over *Algorithm A1* or the standard weighted–coplanarity (WC) approach in [40][29].

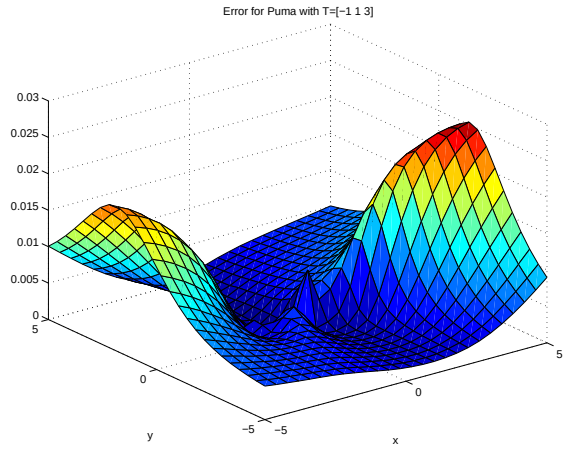
It allows one to treat any number of images (and motions in any number of directions [33]) using a single approach, while *Algorithms A1* and WC can only handle two images.

In the translation–recovery stage, it minimizes a "simpler" error than $\underline{E}_\theta$ or the WC error. Srinivasan has shown that one can quickly compute the *global* minimum of this error using a Fourier technique, at least over the forward–motion region of the error surface [37][36]. This is unlikely to be possible for the more "complex" $\underline{E}_\theta$ or true error. As discussed in the previous sections, the error surface can have many local minima in the forward–motion region and avoiding these is crucial. Thus, with an implementation of Srinivasan's method [36], *Algorithm A2* might be *more* robust than *Algorithm A1* or even than a full bundle adjustment, since the latter are probably

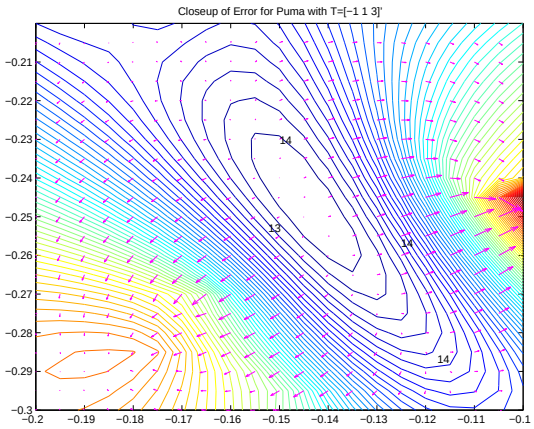
.



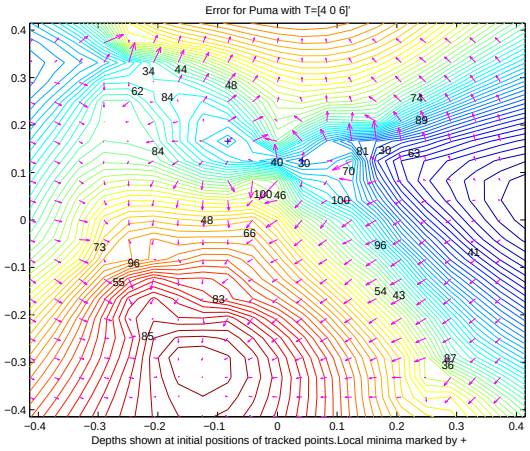
(-1,1,3)



(-1,1,3)



(-1,1,3) closeup



(4,0,6)

Figure 3: Contour and surface plots of the error $\underline{E}_{\mathbf{T}}$ for the PUMA sequence. The error has 1 forward local minima for $\mathbf{T} = (-1, 1, 3)^T$ and 2 for $\mathbf{T} = (4, 0, 6)^T$. They are marked by '+'. Depths are shown at the positions of the tracked points in the zeroth image.

limited to minimization by local search. (We have not yet implemented Srinivasan's method within our approach.)

The way that *Algorithm A2* organizes the minimization may make it more efficient and faster than other techniques. Its $\hat{\mathbf{T}}$–recovery stage is insensitive to first–order errors in its rotation estimation. As a result, the algorithm gives accurate results for $\hat{\mathbf{T}}$ even if the R–estimate has a moderately large error, and it typically requires just a few cycles of rotation/translation recovery to compute $\hat{\mathbf{T}}$ accurately. Since the rotation–recovery stage is linear and fast, the algorithm does most of its work in the $\hat{\mathbf{T}}$–recovery stage. This consists of a minimization over $\hat{\mathbf{T}}$, i.e., over two unknowns,

and it is likely to be to be much faster than a full minimization over all 5 motion unknowns as in *Algorithms A1* or *WC*. Thus, the algorithm may be faster than previous ones.

More precisely, we expect this at least for the most computationally demanding part of the minimization. A minimization routine does most of its work in the opening and middle stages of minimization, while it is still far from the minimum and the error surface is complex and cannot be modeled as quadratic around the minimum.[2] It is in these computationally intensive stages that *Algorithm A2* may be faster than its competitors, since it focuses on the variables (the $\hat{\mathbf{T}}$) in which the error surface has its most important variations. In the late stages of minimization, when one can approximate the surface as quadratic and Gauss–Newton is stable, a straightforward simultaneous minimization over R, $\mathbf{T}$ may well be faster than *Algorithm A2*. One may want to switch from *A2* to such an approach at the end of the minimization.

Though *Algorithm A2* incorporates a small–motion assumption, its main requirement is that the rotations not be too large: [27][25] argue that the algorithm does handle large translations. One of our main experimental conclusions in this section is that the algorithm deals well with large motions. Even if *Algorithm A2* turns out to be less useful for two images than *Algorithms A1* or *WC,* this result supports its usefulness as a multi–image technique [27][28].

The small–rotation assumption in *Algorithm A2* offers an advantage. In real sequences, the rotation are usually moderate, since a large rotation will tend to remove 3D points from the field of view or make them invisible due to occlusion. Also, [25] has shown that the minimization over R for fixed $\hat{\mathbf{T}}$ tends to have local minima at medium–to–large R, which could slow algorithms that do unrestricted minimizations over R if they pass near these minima. *Algorithm A2*, which confines its minimization to small–to–moderate R, is more likely to avoid slowdowns caused by these minima.

We argued in [25] that the small–translation formulation of *Algorithm A2* only diminishes its large–translation performance for forward translations. One of the main experimental conclusions of this section is that *Algorithm A2* gives nearly optimal results for both sideways and forward motions. For forward motions, in fact, it gives better results than the standard *WC* approach in [40][29].

Lastly, the fact that *Algorithm A2* minimizes a "simple" error and incorporates an explicit

[2] If the starting estimate is good enough, one may be able to exploit a quadratic approximation from the beginning. *Algorithm A2* might not be useful in such cases.

small–motion assumption might make it more "reliable," in the sense of [26].

4.2 *Algorithm A2*: Description

4.2.1 Rotation Recovery Stage

Neglecting noise, the exact image displacements between the two images are

$$\mathbf{d}_m \equiv \mathbf{p}_{1m} - \mathbf{p}_{0m} = \frac{T_z Z_m^{-1}(\mathbf{p}_m - \mathbf{e})}{1 - Z_m^{-1} T_z} + \mathbf{f}\left(R, \mathbf{p}_{1m}\right), \tag{12}$$

where $\mathbf{f}(R, \mathbf{p}_{1m}) \equiv \mathbf{p}_{1m} - R^{-1} * \mathbf{p}_{1m}$. Define the first–order rotational flows $\mathbf{r}_m^{(1)}$, $\mathbf{r}_m^{(2)}$, $\mathbf{r}_m^{(3)}$:

$$\left[\begin{array}{ccc} \mathbf{r}_m^{(1)}, & \mathbf{r}_m^{(2)}, & \mathbf{r}_m^{(3)} \end{array} \right] \equiv \left[\begin{pmatrix} -x_m y_m \\ -\left(1 + y_m^2\right) \end{pmatrix}, \begin{pmatrix} 1 + x_m^2 \\ x_m y_m \end{pmatrix}, \begin{pmatrix} -y_m \\ x_m \end{pmatrix} \right].$$

For small rotations, $R\mathbf{V} \approx \mathbf{V} + \boldsymbol{\omega} \times \mathbf{V}$, where $\boldsymbol{\omega}$ is the "log" of the rotation and $\mathbf{V}$ is any 3D vector, and $\mathbf{f}$ is approximately given by

$$\mathbf{f}\left(R, p_{1m}\right) \approx \sum_{a=1}^{3} \omega^a \mathbf{r}_m^{(a)} + O\left(\omega^2, \omega Z^{-1} \left|\mathbf{T}\right|\right), \tag{13}$$

and $\omega, Z^{-1}, |\mathbf{T}|$ in the correction term represent the approximate scales of the rotations, inverse depths, and translation. Define 4 length–N_p vectors $\Psi^{(1,2,3)}$, Υ, with elements

$$\Psi_m^{(a)} \equiv \frac{\mathbf{p}_{0m} - \mathbf{e}}{|\mathbf{p}_{0m} - \mathbf{e}|} \times \mathbf{r}_m^{(a)},$$

and

$$\Upsilon_m \equiv \frac{\mathbf{p}_{0m} - \mathbf{e}}{|\mathbf{p}_{0m} - \mathbf{e}|} \times \mathbf{d}_m.$$

Define $\Psi \equiv \left[\begin{array}{ccc} \Psi^{(1)} & \Psi^{(2)} & \Psi^{(3)} \end{array} \right]$. We have

$$\Upsilon_c = \Psi_c \boldsymbol{\omega} + O\left(\omega^2, \left|\delta \hat{\mathbf{T}}\right| Z^{-1} \left|\mathbf{T}\right|\right),$$

where the subscript c indicates that the quantities are computed for the current estimate of $\mathbf{T}$ or $\mathbf{e}$, and $\left|\delta \hat{\mathbf{T}}\right|$ denotes the error in the recovered $\hat{\mathbf{T}}$. We compute $\boldsymbol{\omega}$ from the above equation, rotate image 1 to compensate for the recovered rotation, and then reapply the translation–recovery step.

4.2.2 Translation Recovery

We briefly summarize our technique for translation recovery, which we described previously in [27][28][31]. Starting from the previous estimate of the translation, our technique refines this estimate by minimizing the least–squares error for infinitesimal motion. We have argued in [25] that the infinitesimal–motion error is a good approximation to the true one when the candidate $\mathbf{e}$ is not close to the image points.

For the first few cycles of translation/rotation recovery, we enhance speed and robustness as follows. When the prevous estimate for $\mathbf{e}$ is outside the FOV, our technique conducts two separate minimizations to avoid the "flipped" or "reflected" local minimum of [25]. For speed, we minimize initially just over $\hat{\mathbf{T}}$ within the plane of $\hat{\mathbf{z}}$ and the previous estimate for $\hat{\mathbf{T}}$. The initial estimate of the $\mathbf{T}$–$\hat{\mathbf{z}}$ plane is likely to be trustworthy even if the bas–relief ambiguity [3][33][38] makes the estimate of $\hat{\mathbf{T}}$ within this plane inaccurate.

4.2.3 Initialization

We have considered two methods for providing initial estimates of the translation and rotation. The first is the standard linear "8-point" algorithm [20] as improved by Hartley [7]. It can deal with motions of any size but works best for large motions [32]. The second is an improved version [28][21] of the linear subspace technique of [14][15], which essentially eliminates the earlier techniques's bias toward recovering $\mathbf{e}$ within the FOV. It can deal with translations of any size but requires small–to–moderate rotations. For small motions, it in effect minimizes the exact projective least–squares error and, thus, should have less bias than the "8-point" approach. Unlike the "8–point" approach, it works for multiple images.

We have arbitrarily used the second method in many of our experiments. But, it cannot deal with planar scenes, and, somewhat surprisingly, our experiments indicate that the "8-point" algorithm does as well even for small translations, except when the translation is forward. We propose a possible explanation for this below. For practical implementations of our algorithm, one should probably initialize using the "8–point" algorithm for nonplanar scenes and homography computation for planar ones.

Though *A2* incorporates a small–motion assumption, it can give accurate results even for very

large translations when initialized using the "8–point" algorithm. As shown in [32], the "8–point" algorithm gives accurate and reliable results when the translation is large and the depth range of the scene is not too small. The translation–recovery step in *A2* can deal with arbitrarily large translations as long as the rotation is known accurately enough: if the rotation is zero, minimizing the infinitesimal–motion error gives $\mathbf{T}$ *exactly* up to noise, even for large translations. Thus, even if the motion is large, *A2* can give good results after compensation of the rotation computed initially by the "8-point" approach.

4.3 Experiments

4.3.1 Synthetic Sequences

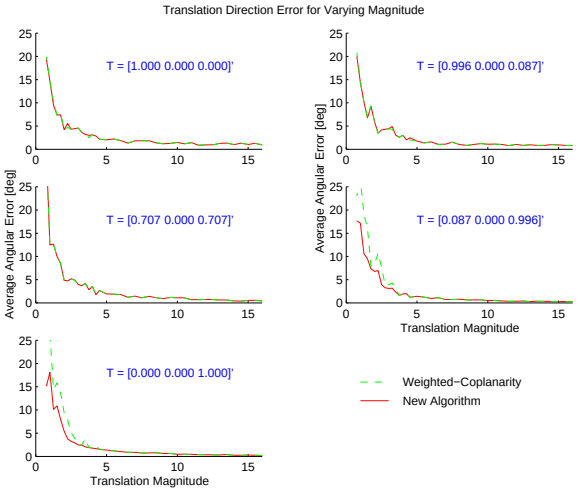
In the following experiments, we chose the rotations randomly up to a maximum of about 22°. The structure consisted of 30 randomly selected points, and the noise was 1 pixel Gaussian, assuming a 512×512 image and the specified FOV. For each translation tested, we created 30 sequences, with different structures, rotations, and noise for each sequence. The error reported for each $\mathbf{T}$ represents the average result over these 30 sequences.

We compared *A2* to *WC*, which gives close–to–optimal results [40][29]. Figure 4 shows results for a FOV of 60° and 3D points with $20 \leq Z \leq 100$. For five different directions of the true $\mathbf{T}$, we plot the average angular error in the recovered $\mathbf{T}$ as a function of the magnitude of $\mathbf{T}_{\text{true}}$ as $|\mathbf{T}_{\text{true}}|$ varies from 0.75 to 16 units. For the most part, the results of *A2* and *WC* are indistinguishable. *A2* appears to do slightly better for the more difficult small–translation trials, at least when $\hat{\mathbf{T}} \sim \hat{\mathbf{z}}$. Both algorithms do poorly in these trials, however. One could have improved the performance of *A2* by using the exact method of [37][36] to avoid local minima.

Figure 5 shows how the algorithms' performance varies as a function of the translation direction. The FOV and depth range were as before. Again, *A2* does slightly better than *WC* when $|\mathbf{T}|$ is small and $\hat{\mathbf{T}} \sim \hat{\mathbf{z}}$, but otherwise gives identical results.

We also studied the convergence behavior of our algorithm. For the experiments in Figure 4, the bottom plot in Figure 6 shows how many cycles of rotation/translation recovery our algorithm needed to converge. Convergence typically takes less than 4 cycles even for large translations. For an additional set of experiments, the top plot in Figure 6 shows how the error in the recovered $\hat{\mathbf{T}}$

Figure 4: Angular error in recovered translation directions for varying magnitudes at fixed directions ($\hat{T} = (\cos\theta, 0, \sin\theta)^T$ with $\theta = 0°$, $\theta = 5°$, $\theta = 45°$, $\theta = 85°$ and $\theta = 90°$).



decreases with the number of iterations. The error shown is the average result over 300 trials with random translations varying in slant between 0° and 90° and in magnitude between 0.75 and 16. The FOV and depth range were as before. Typically, just 2–3 iterations are enough to give a good estimate of the translation direction.

We also tested our initial linear estimator from [28] against the "8-point" algorithm. Figure 7 shows the results for a variety of translation directions and magnitudes. The FOV and depth range were as before. The "8–point" algorithm does better at large translations. Unexpectedly, it also does as well as our linear estimator even for small translations, except for $\hat{\mathbf{T}} \sim \hat{\mathbf{z}}$. This is surprising since for small motions the linear estimator minimizes nearly the exact projective least–squares error, while the "8–point" algorithm in effect minimizes a biased projective least–squares error. We also compared an iterative version of our linear estimator to the "8-point" algorithm. For this version, we repeat a two–step cycle of linear rotation recovery followed by linear translation recovery, until convergence. Figure 8 shows that this iterated approach again does slightly better than (a single run of) the "8-point" algorithm for $\hat{\mathbf{T}} \sim \hat{\mathbf{z}}$ but otherwise performs nearly identically. These results have a natural explanation. For large sideways translations, the rotation must be large to keep the scene points in view. The better performance of the "8–point" approach for such translations is probably due to the fact that our linear estimator assumes the rotations are infinitesimal.

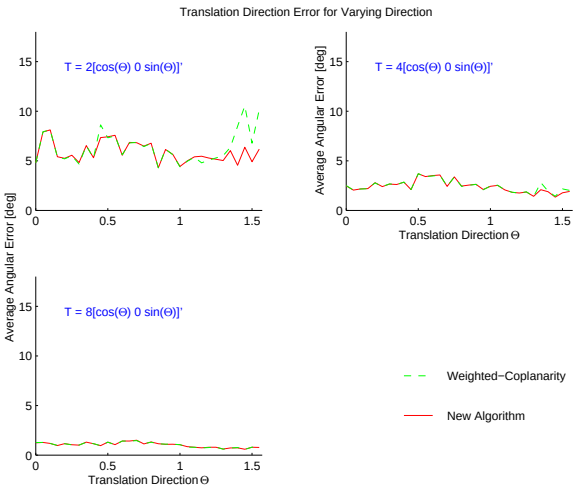


Figure 5: Angular error in translation direction recovery for varying translation directions ($\hat{T} = (\cos(\theta), 0, \sin(\theta))^T$, $\theta \in [0°, 90°]$) at fixed translation magnitudes of 2, 4, 8.

We tested the two algorithms on planar scenes. The planes were chosen with random tilts, with random slants between 20° and 75°, and so that the minimum depth of the 3D points was 20. The FOV was 60°. Figure 9 shows the results.[3] When $\hat{\mathbf{T}} \sim \hat{\mathbf{z}}$, *A2* again does slightly better than *WC* for small $|\mathbf{T}|$, but it now appears to do slightly worse for large $|\mathbf{T}|$. For other $\hat{\mathbf{T}}$ directions, its performance is nearly the same. Surprisingly, our algorithm achieved the same converged results whether started with the "8–point" algorithm or our initial linear estimator. Since these initial estimators both give poor results for planar scenes, this indicates that our approach converges quite stably for these scenes.

We also tested the algorithms with very large motion to the side of the scene. For each image pair, we chose the depths of the 3D points in the range $\bar{d} - 40$ to $\bar{d} + 40$, where $\bar{d}$ varied from 50 to 70 for different pairs. The translation was $\begin{pmatrix} 4\left(\bar{d} + W_{\max}\right) & 0 & -\left(\bar{d} + W_{\max}/2\right) \end{pmatrix}^T$, where $W_{\max}$ characterizes the width of the structure. The rotations were in the range 60°–90°. Figure 10 shows the results for 1000 image pairs created in this way. *A2* gives results close to those of *WC* even for this large translational motion. Surprisingly, it again converged to the same results when started from either initial linear estimator, indicating its stability.

Finally, we conducted experiments with varying FOV (Figure 11) and scene depth (Figure 12).

[3] To account for the well known two–fold ambiguity in reconstructing planar scenes, the error plotted in this Figure is computed as the minimum of the two errors between the recovered $\hat{\mathbf{T}}$ and the two valid possibilities for the ground truth $\hat{\mathbf{T}}$.

Figure 6: Number of iterations for convergence in the new algorithm.

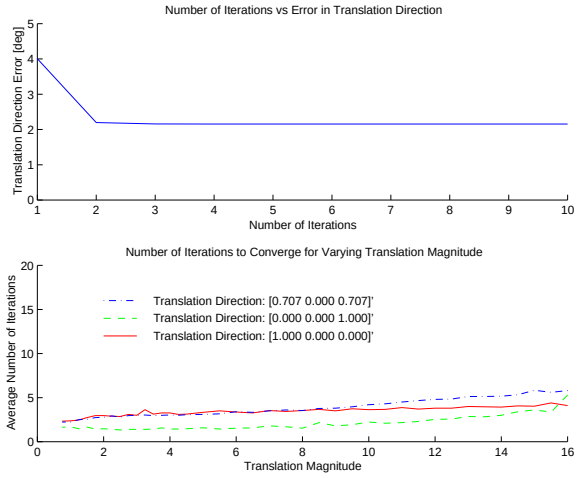
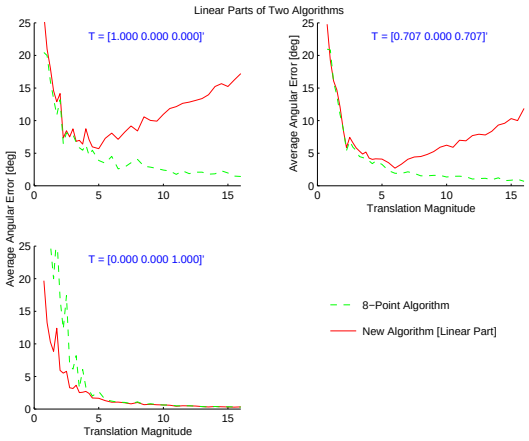
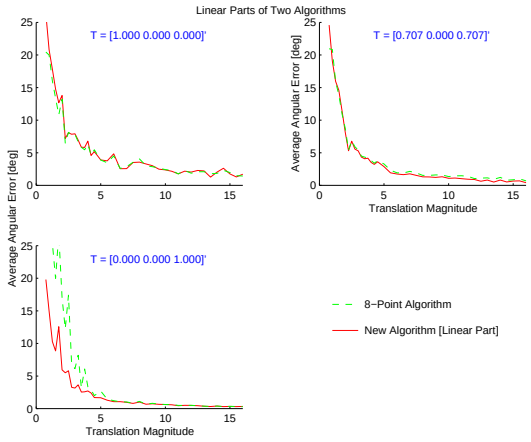


Figure 7: Angular error in translation direction recovery with linear algorithms for varying translation magnitudes at fixed translation directions ($\hat{T} = (\cos\theta, 0, \sin\theta)^T$ with $\theta = 0°$, $\theta = 45°$ and $\theta = 90°$).



For the FOV test, we created 1000 image pairs in the same way as in the first two experiments except that the FOV varied randomly in the range 50°–80°. We also chose the translations randomly with magnitudes of 2–6 units. For the varying scene–depth test, we kept the field of view fixed at 60°, again varied the translations in the range $2 \leq |\mathbf{T}| \leq 6$, and randomly chose $\bar{d}$ in the range 50–80 units. (As before, for any given sequence we chose the 3D depths in the range $\bar{d} - 40$ to $\bar{d} + 40$.) Note that the average error does not decrease with the FOV, though larger FOV makes it easier to distinguish rotations from translations. This is due to the fact that we scaled the size of the image noise by the FOV [39].

Figure 8: Angular error in translation direction recovery with linear algorithms (iterative for the new approach) for varying translation magnitudes at fixed translation directions ($\hat{T} = (\cos\theta, 0, \sin\theta)^T$ with $\theta = 0°$, $\theta = 45°$ and $\theta = 90°$).



4.3.2 Experiments on Real Images

We also ran *A2* and *WC* on the CASTLE data set (available from CMU). This sequence consists of 11 images with 28 feature points tracked over the sequence. From the provided calibration, we calculated that the FOV was 9.2°. Based on a multi–image reconstruction and the provided ground truth, the 3D points vary in depth from 90–104, in units where the maximum translation from the first camera position was 4 units. Figure 13 shows one of the images with the tracked feature points marked. Figure 14 shows the results[4] obtained by *WC* and *A2* for the angular errors in the recovered $\hat{\mathbf{T}}$ for all 55 distinct image pairs. The two algorithm perform essentially identically. Over all pairs, the average error in $\hat{\mathbf{T}}$ is 1.35° and the standard deviation is 1.44°.

5 *Algorithm A3*

In reconstructing from intensity images, one confronts several dilemmas. Small image regions often contain too little information for reliable correspondence recovery, but one can't compute detailed correspondence directly from big image regions—usually the most one can recover directly are their 2D affine deformations and perhaps their *affine parallax* [19]. Similarly, though it may seem easiest to determine the correspondence in regions where the intensity is changing sharply, these often

[4]Since the motion in this sequence is almost purely translational, for our algorithm we did not compensate for the rotation. Compensating for the rotations gives similar results; the difference in average is just 0.08°.

Figure 9: Angular error in translation direction recovery for planar scenes. The translation directions are given by $\hat{T} = (\cos\theta, 0, \sin\theta)^T$, with $\theta = 0°$, $\theta = 45°$ and $\theta = 90°$.

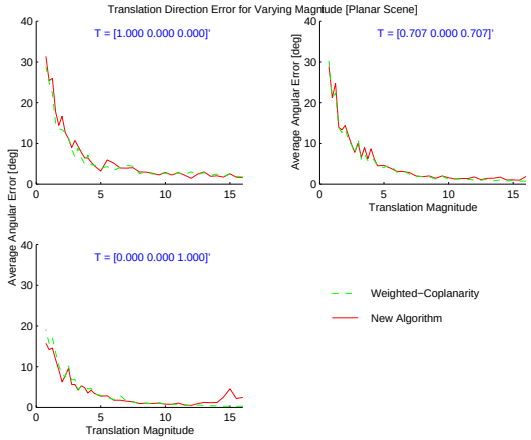
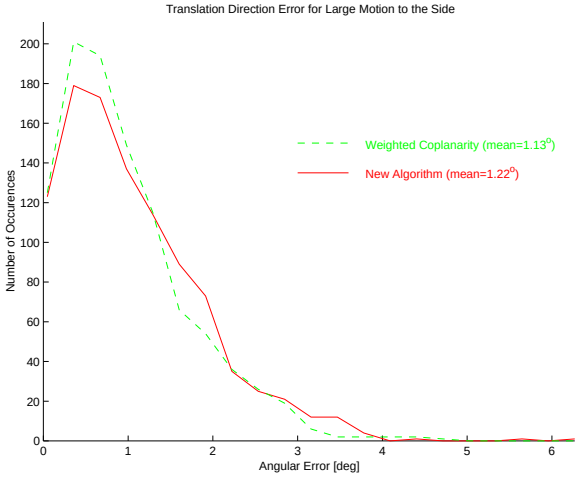


Figure 10: Angular error in translation direction recovery for a motion to the side of the scene.



constitute a small fraction of the image, and they tend to occur at occlusion boundaries, which can compromise the recovered correspondence. Regions where the intensity changes smoothly cover more of the image and are less likely to be affected by occlusion, but, again, one can seldom recover much more than their affine deformations.

Motivated by this dilemma, we propose in this section an algorithm that recovers the motion and structure from the recovered *2D* affine deformations over ≥ 3 distinct image patches. [2][19][18] have described methods for reconstructing from the affine deformations plus affine parallax. These in effect require computing a *second* derivative of the flow, whereas our approach only requires computing its first derivatives. Also, we show how to generalize our approach to a multi–image

Figure 11: Angular error in translation direction recovery for varying field of view.

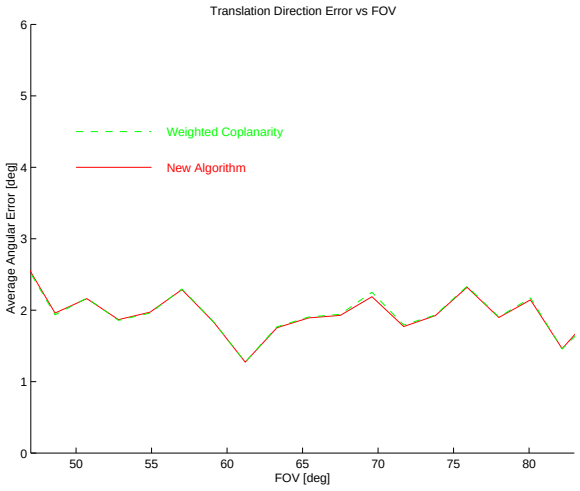
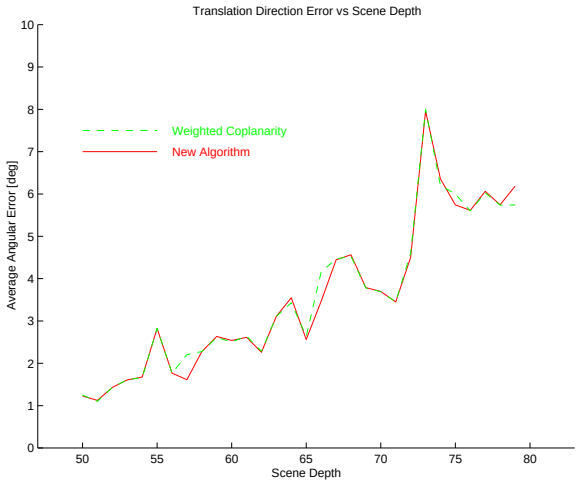


Figure 12: Angular error in translation direction recovery for varying scene depth.



algorithm.

Our method is a generalization of the subspace technique [16] [1] for infinitesimal motion. We begin by describing the equivalent of the linear subspace technique [14]. For small rotations, we have:

$$(\mathbf{p} - \mathbf{e}_{\text{true}}) \times \mathbf{d} \approx \sum_a (\mathbf{p} - \mathbf{e}_{\text{true}}) \times \omega^a \mathbf{r}^{(a)}(\mathbf{p}). \tag{14}$$

(We consider $\mathbf{p}$ and $\mathbf{d}$ as continuous and eliminate the subscript m.)

Recovering the affine deformation of a image patch is essentially equivalent to recovering the flow and its first derivatives with respect to the image coordinates. If the affine deformation $(A, \mathbf{B})$ is defined by $\mathbf{p}_1 = A\mathbf{p}_0 + \mathbf{B}$, where $\mathbf{p}_0$ and $\mathbf{p}_1$ are points in the zeroth and first images, then

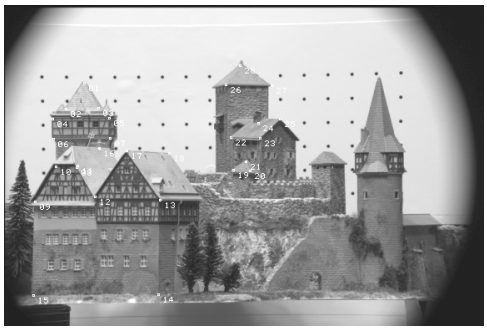
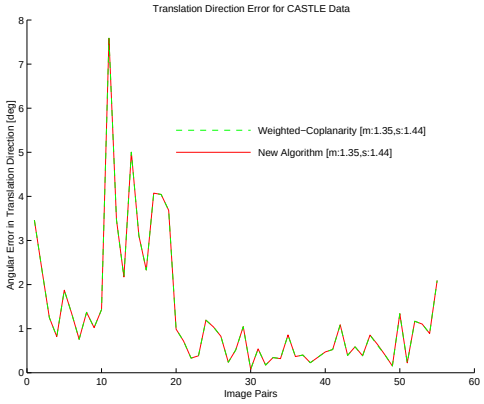


Figure 13: A frame in CASTLE data with overlayed feature points.

Figure 14: Performance of approximate and the full minimization algorithms on CASTLE data.



$\mathbf{d} = (A - \mathbf{1}_2)\,\mathbf{p}_0 + \mathbf{B}$, where $\mathbf{1}_2$ is the 2×2 identity matrix. Also, $\nabla_k \mathbf{d} = (A - \mathbf{1}_2)\,\hat{\mathbf{k}}$, where the 2D vector $\hat{\mathbf{k}}$ is defined by $\hat{\mathbf{k}}_a \equiv \delta_{ak}$. Conversely, one can compute A and $\mathbf{B}$ from the 6 quantites $\mathbf{d}$, $\nabla_k \mathbf{d}$.

Differentiating (14) yields

$$\begin{aligned} \nabla_k \left((\mathbf{p} - \mathbf{e}_{\text{true}}) \times \mathbf{d} \right) &= \hat{\mathbf{k}} \times \mathbf{d} + (\mathbf{p} - \mathbf{e}_{\text{true}}) \times \nabla_k \mathbf{d} \\ &\approx \sum_a \omega^a \nabla_k \left((\mathbf{p} - \mathbf{e}_{\text{true}}) \times \mathbf{r}^{(a)} \right). \end{aligned}$$

Let N_R be the number of image regions over which the affine deformation has been recovered. Let $\mathbf{p}_h$ give the position of the h–th region, and let $\mathbf{d}_h$ and $\nabla \mathbf{d}|_{\mathbf{p}_h}$ characterize the flow for this region.

Define the length–$3N_R$ vector Υ_L by:

$$\begin{aligned}
\Upsilon_{L,3h-2} &= (\mathbf{p}_h - \mathbf{e}) \times \mathbf{d}_h, \\
\Upsilon_{L,3h-1} &= \nabla_x \left((\mathbf{p} - \mathbf{e}) \times \mathbf{d} \right) |_{\mathbf{p}_h}, \\
\Upsilon_{L,3h} &= \nabla_y \left((\mathbf{p} - \mathbf{e}) \times \mathbf{d} \right) |_{\mathbf{p}_h}.
\end{aligned}$$

Similarly, define 3 length–$3N_R$ vectors $\Psi_L^{(1,2,3)}$ by

$$\begin{aligned}
\Psi_{L,3h-2}^{(a)} &= (\mathbf{p}_h - \mathbf{e}) \times \mathbf{r}_h^{(a)}, \\
\Psi_{L,3h-1}^{(a)} &= \nabla_x \left((\mathbf{p} - \mathbf{e}) \times \mathbf{r}_h^{(a)} \right) |_{\mathbf{p}_h}, \\
\Psi_{L,3h}^{(a)} &= \nabla_y \left((\mathbf{p} - \mathbf{e}) \times \mathbf{r}_h^{(a)} \right) |_{\mathbf{p}_h},
\end{aligned}$$

where $\mathbf{r}_h^{(a)} \equiv \mathbf{r}^{(a)}(\mathbf{p}_h)$.

Independent of $\mathbf{e}$, $(\mathbf{p}_h - \mathbf{e}) \times \mathbf{r}_h^{(a)} =$

$$c_1 + c_2 x_h + c_3 y_h + c_4 x_h^2 + c_5 x_h y_h + c_6 y_h^2,$$

where $\mathbf{p}_h \equiv (x_h; y_h)$, and

$$\begin{aligned}
\nabla_x \left((\mathbf{p} - \mathbf{e}) \times \mathbf{r}_h^{(a)} \right) |_{\mathbf{p}_h} &= c_2 + 2c_4 x_h + c_5 y_h, \\
\nabla_y \left((\mathbf{p} - \mathbf{e}) \times \mathbf{r}_h^{(a)} \right) |_{\mathbf{p}_h} &= c_3 + c_5 x_h + 2c_6 y_h,
\end{aligned}$$

with the same coefficients. Using Householder matrices [31][29], define a matrix H_L that annihilates the six length–$3N_R$ rotational–flow vectors $\bar{\Psi}^{(a)}$, where $\bar{\Psi}^{(1)}_{3h-2:3h} = \left[\begin{array}{ccc} 1; & 0; & 0 \end{array} \right]$, $\bar{\Psi}^{(2)}_{3h-2:3h} = \left[\begin{array}{ccc} x_h; & 1; & 0 \end{array} \right]$, $\bar{\Psi}^{(3)}_{3h-2:3h} = \left[\begin{array}{ccc} y_h; & 0; & 1 \end{array} \right]$, $\bar{\Psi}^{(4)}_{3h-2:3h} = \left[\begin{array}{ccc} x_h^2; & 2x_h; & 0 \end{array} \right]$, $\bar{\Psi}^{(5)}_{3h-2:3h} = \left[\begin{array}{ccc} x_h y_h; & y_h; & x_h \end{array} \right]$, $\bar{\Psi}^{(6)}_{3h-2:3h} = \left[\begin{array}{ccc} y_h^2; & 0; & 2y_h \end{array} \right]$.

Then

$$H_L \Upsilon_L \left(\mathbf{e}_{\text{true}} \right) \approx 0$$

gives a linear system of equations which one can solve for $\mathbf{e}_{\text{true}}$.

The algorithm is:

1. Compute the $\bar{\Psi}^{(a)}$ and H for the image regions over which one computes the affine deformations.

2. Compute $\mathbf{d}_h$ and $\nabla\mathbf{d}_h$ from the affine deformations.

3. Solve for $\mathbf{e}$ from $H_L\Upsilon_L(\mathbf{e}) = 0$, where one computes Υ_L from the $\mathbf{p}$, $\mathbf{d}$, and $\nabla\mathbf{d}$.

As described for the feature–point approach of [27][28], one should modify this algorithm by introducing a scaling by $O(\theta_{\text{FOV}})$ to reduce the bias for typical FOV with $\theta_{\text{FOV}} \leq 90°$.

This linear algorithm has the usual problems of the linear subspace technique [25][27]: most important, it does not exploit all the available constraints for known calibration. One can get a nonlinear, iterative algorithm that does, and which also has reduced bias, by starting from:

$$\frac{\mathbf{p}-\mathbf{e}_{\text{true}}}{|\mathbf{p}-\mathbf{e}_{\text{true}}|}\times\mathbf{d} \approx \sum_a \frac{\mathbf{p}-\mathbf{e}_{\text{true}}}{|\mathbf{p}-\mathbf{e}_{\text{true}}|}\times\omega^a\mathbf{r}^{(a)}(\mathbf{p}).$$

Define a matrix $H_A(\mathbf{e})$ to eliminate the 3 vectors $\Psi_A^{(1,2,3)}$ defined by:

$$\begin{aligned}
\Psi^{(a)}_{A,3h-2} &= \frac{\mathbf{p}_h-\mathbf{e}}{|\mathbf{p}_h-\mathbf{e}|}\times\mathbf{r}_h^{(a)},\\
\Psi^{(a)}_{A,3h-1} &= \nabla_x\left(\frac{\mathbf{p}-\mathbf{e}}{|\mathbf{p}-\mathbf{e}|}\times\mathbf{r}_h^{(a)}\right)|_{\mathbf{p}_h},\\
\Psi^{(a)}_{A,3h} &= \nabla_y\left(\frac{\mathbf{p}-\mathbf{e}}{|\mathbf{p}-\mathbf{e}|}\times\mathbf{r}_h^{(a)}\right)|_{\mathbf{p}_h}.
\end{aligned}$$

Define Υ_A similarly to Υ_L, but with $(\mathbf{p}-\mathbf{e}) \longrightarrow (\mathbf{p}-\mathbf{e})/|\mathbf{p}-\mathbf{e}|$. One recovers $\mathbf{e}$ by minimizing

$$\Upsilon_A^T H_A^T H_A \Upsilon_A.$$

This algorithm generalizes to multiple images and multiple motions as in [27].

A Multi–Image Algorithm. Following [29], we define a multi–image, general–motion algorithm from affine deformations. Start from

$$\mathbf{d}^i \approx Z^{-1}(T_z^i\mathbf{p} - \left[\mathbf{T}^i\right]_2) + \sum_a \omega^{i,a}\mathbf{r}^{(a)}(\mathbf{p}),$$

where the superscript i indexes the images, and the displacements are with respect to image 0. Differentiating yields

$$\nabla_k \mathbf{d}^i = \boldsymbol{\nabla}_k Z^{-1}(T_z^i \mathbf{p} - [\mathbf{T}]_2) + Z^{-1} T_z^i \hat{\mathbf{k}} + \sum_a \omega^{i,a} \nabla_k \mathbf{r}^{(a)}.$$

Let N_I be the number of images, and define a $(N_I - 1) \times 6N_R$ matrix $\mathbf{D}_A$, where the i–th row equals

$$\left[d^i_{x1}, \nabla_x d^i_{x1}, \nabla_y d^i_{x1}, d^i_{x2}, \nabla_x d^i_{x2}, \dots d^i_{y1}, \nabla_x d^i_{y1}, \nabla_y d^i_{y1}, d^i_{y2}, \dots \right]$$

(the subscripts $\{1, 2, \dots\}$ indicate the image regions over which the affine deformation has been recovered). Define the three length–$6N_R$ structure vectors

$$\Phi_x^T \equiv \left[-Z_1^{-1}, -\nabla_x Z_1^{-1}, -\nabla_y Z_1^{-1}, -Z_2^{-1}, -\nabla_x Z_2^{-1}, \dots, \underbrace{0, 0, \dots 0}_{3N_R} \right],$$

$$\Phi_y^T \equiv \left[\underbrace{0, 0, \dots 0}_{3N_R}, -Z_1^{-1}, -\nabla_x Z_1^{-1}, -\nabla_y Z_1^{-1}, -Z_2^{-1}, -\nabla_x Z_2^{-1}, -\nabla_y Z_2^{-1}, \dots \right],$$

$$\Phi_z^T \equiv \left[x_1 Z_1^{-1}, Z_1^{-1} + x_1 \nabla_x Z_1^{-1}, x_1 \nabla_y Z^{-1}, x_2 Z_2^{-1}, \dots y_1 Z_1^{-1}, y_1 \nabla_x Z_1^{-1}, Z_1^{-1} + y_1 \nabla_x Z_1^{-1}, y_2 Z_2^{-1} \dots \right].$$

Defining as before a matrix H to eliminate the rotational contribution, one obtains the approximate factorization

$$\mathbf{D}_A H^T \approx \{T_x\}\, \Phi_x^T + \{T_y\}\, \Phi_y^T + \{T_z\}\, \Phi_z^T,$$

where, e.g., $\{T_x\}$ is the column vector $\left[\; T_x^1; \quad T_x^2; \quad \dots \; \right]$. The algorithm is:

1. Compute $\mathbf{D}_A$ from the recovered affine deformations via $\mathbf{d}$, $\nabla \mathbf{d}$.
2. Compute H using Householder matrices.
3. Use the singular value decomposition to compute the best rank–3 factorization of $\mathbf{D}_A H^T = \sum_{a=1}^3 M^a S^{aT}$, where M^a, S^a are column vectors.
4. Recover the Z^{-1}, ∇Z_h^{-1} from the three S^a by solving the linear system

$$\sum_{a=1}^{3} S^a U_{ab} = \Phi_{b,}$$

which consists of of $18N_R$ equations in the $3N_R + 9$ unknowns Z_h^{-1}, ∇Z_h^{-1}, and the 9 parameters U_{ab}.

5. Recover the translations from

$$\left[\begin{array}{ccc} \{T_x\} & \{T_y\} & \{T_z\} \end{array} \right] = \left[\begin{array}{ccc} M^1 & M^2 & M^3 \end{array} \right] U^{-T}.$$

As discussed in [29], one should modify this algorithm by right–multiplying $\mathbf{D}_A H^T$ by the matrix $C^{-1/2}$ defined in that paper. This reduces the bias due to singling out the zeroth image. Also, one can adapt this algorithm to exploit affine deformations in combination with tracked points, lines, and intensities along the lines of [22].

Projection Algorithms. All the above algorithms can be converted directly into projective ones, see [24].

6 Conclusion

We presented three algorithms for 2–image and $\geq$ 2–image structure from motion. The first gives fast and essentially optimal reconstructions from point features. We used this algorithm to study the least–squares error surface, and demonstrated that local minima occur frequently for forward motion, even when the motion is large. (Our results also verify the existence of the "flipped" minimum of [25] [4].)

The second applies to multi–image as well as 2–image sequences. It is potentially faster than the first, and our experiments show that it gives nearly optimal results even for large motions. This success provides further supporting evidence that the algorithm is effective for multiple images (see also [27]), and suggests that it is worth pursuing as a two–image approach. An important question for future study is how much implementing the method of [37][36] will improve its performance.

Finally, the third algorithm (or class of algorithms) reconstructs from 2D affine deformations recovered over distinct image patches. We described multi–image extensions of this approach.

References

[1] G. Adiv, "Inherent ambiguities in recovering 3-D motion and structure from a noisy flow field," **PAMI** 11, 477-489, 1989.

[2] P. Anandan and S. Avidan, "Integrating Local Affine in Global Projective Images in the Joint Image Space," *ECCV* 907–921fs, 2000.

[3] P. Belhumeur, D. Kriegman, and A. Yuille "The Bas–Relief Ambiguity," *CVPR* 1060–1066, 1997.

[4] A. Chiuso, R. Brockett, and S. Soatto, "Optimal Structure from Motion: Local Ambiguities and Global Estimates," Washington University Technical Report, 1999.

[5] R. Dutta, R. Manmatha, L.R. Williams, and E.M. Riseman, "A data set for quantitative motion analysis," *CVPR*, 159-164, 1989.

[6] R. I. Hartley and P. Sturm, "Triangulation," *CVIU* 68, 146-157, 1997.

[7] R. I. Hartley, "In Defense of the Eight–Point Algorithm," **PAMI** 19, 580–593, 1997.

[8] R. I. Hartley, "In Defense of the 8–point Algorithm," *ICCV* 1064–1070, 1995.

[9] R. Hartley, "Euclidean Reconstruction from Uncalibrated Views," in *Second Workshop on Invariants*,1993, 187–202.

[10] R. I. Hartley, "Self–Calibration from Multiple Views with a Rotating Camera," *ECCV* 471–478, 1994.

[11] R. I. Hartley, "Lines and points in three views — a unified approach," *IUW* 1009–1016, 1994.

[12] R.I. Hartley and P. Sturm, "Triangulation," *IUW*, 957-966, 1994.

[13] D.J. Heeger and A.D. Jepson, "Subspace methods for recovering rigid motion I: Algorithm and implementation," **IJCV** 7, 95-117, 1992.

[14] A.D. Jepson and D.J. Heeger, "Linear subspace methods for recovering translational direction," in *Spatial Vision in Humans and Robots*, Cambridge, 39–62, 1993.

[15] A.D. Jepson and D.J. Heeger, "A fast subspace algorithm for recovering rigid motion," *Motion Workshop* , Princeton, N.J., 124-131, 1991.

[16] A.D. Jepson and D.J. Heeger, "Subspace methods for recovering rigid motion II: Theory," U. of Toronto TR RBCV-TR-90-36, 1990.

[17] R. Kumar and A.R. Hanson, "Sensitivity of the Pose Refinement Problem to Accurate Estimation of Camera Parameters," *ICCV*, 365-369, 1990.

[18] J. M. Lawn and R. Cipolla, "Reliable extraction of camera motion using constraints on the epipole," in *ECCV* II:161–173, 1996.

[19] J. M. Lawn and R. Cipolla, "Robust egomotion estimation from affine motion–parallax," in *ECCV* I: 205–210, 1994.

[20] H. C. Longuet–Higgins, "A computer algorithm for reconstructing a scene from two projections," **Nature**, 293: 133–135, 1981.

[21] W. J. MacLean, "Removal of translation bias when using subspace methods," *ICCV* 753–758, 1999. Also, PhD thesis, 1996.

[22] J. Oliensis and M. Werman, "Structure from Motion using Points, Lines, and Intensities," *CVPR* vol. 2, 599–606, 2000.

[23] J. Oliensis and Y. Genc, "New Algorithms for Two-Frame Structure from Motion," *ICCV* 737–744, 1999.

[24] J. Oliensis and Y. Genc, "Fast Algorithms for Projective Multi–Frame Structure from Motion," *ICCV* 536–543,1999, and **PAMI**, to appear.

[25] J. Oliensis, "A New Structure from Motion Ambiguity," *CVPR* 185–191, 1999, and **PAMI** 22:7, 685–700, 2000.

[26] J. Oliensis, "A Critique of Structure from Motion Algorithms," NECI TR 1997, to appear as a discussion paper in *CVIU*.

[27] J. Oliensis, "Recovering Heading and Structure for Constant–Direction Motion," NEC TR 1997.

[28] J. Oliensis, "Computing the Camera Heading from Multiple Frames," *CVPR* 203–210, 1998.

[29] J. Oliensis, "A Multi-frame Structure from Motion Algorithm Under Perspective," **IJCV** 34:2/3, 163–192, 1999.

[30] J. Oliensis, "Multiframe Structure from Motion in Perspective," *Workshop on the Representations of Visual Scenes*, 77–84, June 1995.

[31] J. Oliensis, "A Linear Solution for Multiframe Structure from Motion," *IUW*, 1225–1231. 1994.

[32] J. Oliensis, "Rigorous Bounds for Two–Frame Structure from Motion," *ECCV* 184-195, 1996.

[33] J. Oliensis, "Structure from Linear and Planar Motions," *CVPR* 335–342, 1996.

[34] S. Soatto and R. Brocket, "Optimal Structure from Motion: Local Ambiguities and Global Estimates," *CVPR* 1998, 282–288.

[35] S. Soatto and R. Brocket, "Optimal and Suboptimal Structure from Motion," Harvard University TR, 1997.

[36] S. Srinivasan, "Extracting Structure from Optical Flow Using the Fast Error Search Technique," **IJCV** 37(3): 203–230, 2000.

[37] S. Srinivasan, "Fast Partial Search Solution to the 3D SFM Problem," *ICCV* 528–535, 1999.

[38] R. Szeliski and S.B. Kang, "Shape ambiguities in structure from motion," **PAMI** 19, 506–512, 1997.

[39] T. Y. Tian, C. Tomasi, and D. J. Heeger, "Comparison of Approaches to Egomotion Computation" *CVPR* 315–320, 1996.

[40] Zhengyou Zhang, "On the optimization criteria for two–frame structure from motion," **PAMI** 20:7, 717-729, 1998.

[41] Zhengyou Zhang, "Understanding the Relationship Between the Optimization Criteria in Two–View Motion Analysis," *ICCV* 772–777, 1998.