Fast, Robust, and Consistent Camera Motion Estimation

Tong Zhang and Carlo Tomasi*
Computer Science Department
Stanford University, Stanford, CA, 94305
tomasi@cs.stanford.edu, tzhang@watson.ibm.com

Abstract

Previous algorithms that recover camera motion from image velocities suffer from both bias and excessive variance in the results. We propose a robust estimator of camera motion that is statistically consistent when image noise is isotropic. Consistency means that the estimated motion converges in probability to the true value as the number of image points increases. An algorithm based on reweighted Gauss-Newton iterations handles 100 velocity measurements in about 50 milliseconds on a workstation.

1 Introduction

In principle, the instantaneous velocities measured in the images produced by a moving camera determine the rotation and direction of translation of the camera as well as depth, *i.e.*, the distances of scene points from the camera's center of projection. While large, finite camera motions yield a better baseline for scene depth measurements, the small motions implied by the use of image velocities are more easily measured, and are sufficiently good for camera motion estimation.

In the absence of noise, conditions for the existence and uniqueness of a solution are well understood. However, small errors (henceforth referred to as *noise*) in the image velocities can produce large errors in the estimates of depth and motion. Similarly, a single outlier can play havoc with the solution.

In addition, bias and variance can be amplified by inappropriate transformations of the problem formulation. When previous researchers have seeked insights and designed algorithms through linearizations or various algebraic transformations of the original problem, they have exposed themselves to this danger. Bias, often severe, appeared in the solutions, and variance was greater than necessary.

This paper addresses these issues, and proposes an algorithm for the computation of camera motion from sparse image velocities. The algorithm produces solutions with lower bias and smaller variance than previous methods, and tolerates a moderate amount of outliers. Furthermore, our method produces consistent estimates of camera motion, in the sense that these estimates become as close as desired to their true values, with probability one, as the number of image points is increased. Our algorithm is fast, and can process 100 image velocities per frame at 20 frames per second on a workstation.

*Supported by NSF grant IRI-9509149, DoD MURI contract DAAH04-96-1-0007, and by a grant of the Charles Lee Powell Foundation.

2 Problem Statement

The image velocity due to camera motion under perspective projection is given by

$$\mathbf{u}(\mathbf{x}) = d(\mathbf{x})A(\mathbf{x})\mathbf{t} + B(\mathbf{x})\omega + \mathbf{n}(\mathbf{x}) \ . \qquad (1)$$

Here, $\mathbf{u}(\mathbf{x})$ is the image velocity at image position $\mathbf{x} = (x_1, x_2, 1)^T$, $\mathbf{t}$ is the camera's translational velocity, ω is its rotational velocity, $d(\mathbf{x})$ is the reciprocal of scene depth at $\mathbf{x}$, measured along the optical axis. The camera's focal length is 1, without loss of generality. The term $\mathbf{n}(\mathbf{x})$ denotes noise, and

$$A(\mathbf{x}) = \begin{bmatrix} 1 & 0 & -x_1 \\ 0 & 1 & -x_2 \end{bmatrix},$$

$$B(\mathbf{x}) = \begin{bmatrix} -x_1x_2 & 1+x_1^2 & -x_2 \\ -1-x_2^2 & x_1x_2 & x_1 \end{bmatrix}.$$

The image motion analysis problem is to estimate the 3D motion parameters $\mathbf{t}$ and ω and the vector $\mathbf{d}$ of inverse depths $d(\mathbf{x})$ from a collection of velocity vectors $\mathbf{u}(\mathbf{x})$. Because $\mathbf{t}$ and d appear as a product in equation (1), their absolute magnitudes cannot be determined. We therefore add the constraint $||\mathbf{t}|| = 1$. We define our estimator as

$$\hat{\mathbf{t}}, \hat{\omega}, \hat{\mathbf{d}} = \arg\min_{\mathbf{t},\omega,\mathbf{d}} \frac{1}{m} \sum_{\{\mathbf{x}\}} ||\mathbf{r}(\mathbf{x})||_p^p \ , \qquad (2)$$

In this expression,

$$\mathbf{r}(\mathbf{x}) = \mathbf{u}_* - d(\mathbf{x})A(\mathbf{x})\mathbf{t} - B(\mathbf{x})\omega \qquad (3)$$

is the residual, $\{\mathbf{x}\}$ is the set of the m image locations where velocity measurements $\mathbf{u}_*$ are available, and $||\cdot||_p$ is the

0-7695-0149-4/99 $10.00 © 1999 IEEE

p-norm, with p a number between 1 and 2. More generally, we can define the estimator

$$\hat{\mathbf{t}}, \hat{\omega}, \hat{\mathbf{d}} = \arg\min_{\mathbf{t},\omega,\mathbf{d}} \frac{1}{m} \sum_{\{\mathbf{x}\}} f(\mathbf{r}(\mathbf{x})) \tag{4}$$

where $f(\cdot)$ is any rotationally symmetric and convex function, called the *loss function*.

The residual $\mathbf{r}(\mathbf{x})$ is a vector with two components. However, for each point $\mathbf{x}$, a reference frame can be chosen in which one component of $\mathbf{r}(\mathbf{x})$ can be zeroed exactly, even in the presence of noise: If we define the unit *epipolar* vector

$$\mathbf{e} = \frac{A(\mathbf{x})\mathbf{t}}{\|A(\mathbf{x})\mathbf{t}\|} ,$$

from equation (3) we see that $\mathbf{e}^T\mathbf{r}(\mathbf{x}) = \mathbf{e}^T\mathbf{u}_* - d(\mathbf{x})\|A(\mathbf{x})\mathbf{t}\| - \mathbf{e}^T B(\mathbf{x})\mathbf{x}$ can be set to zero by letting

$$d(\mathbf{x}) = \frac{\mathbf{e}^T(B(\mathbf{x})\mathbf{x} - \mathbf{u}_*)}{\|A(\mathbf{x})\mathbf{t}\|} . \tag{5}$$

Thus, no loss of generality is incurred if $\mathbf{r}(\mathbf{x})$ is replaced by its component orthogonal to $\mathbf{e}$:

$$r(\mathbf{x}) = \tau^T\mathbf{r}(\mathbf{x}) = \tau^T(\mathbf{u}_* - B(\mathbf{x})\mathbf{x}) \tag{6}$$

where $\tau = [e_2, -e_1]^T$ is a unit-norm vector orthogonal to $\mathbf{e}$. This residual $r(\mathbf{x})$ is independent of $\mathbf{t}$ and $d(\mathbf{x})$, and only depends on the rotation ω.

3 Previous Work

In the absence of noise, image velocities at five points yield a finite number of depth and motion solutions [12]. With more points, the solution is essentially unique [7]. Image velocity measurements at more than five points are necessary if the data are noisy. Bruss and Horn [2] algebraically eliminate depth and obtain a residual $\mathbf{r}(\mathbf{x})$ that is bilinear in camera rotation and translation. They then simplify the expression of the residual for computational purposes, effectively replacing the residual $\mathbf{r}(\mathbf{x})$ in equation (3) with

$$\mathbf{r}'(\mathbf{x}) = \mathbf{r}(\mathbf{x})\|A(\mathbf{x})\mathbf{t}\| . \tag{7}$$

We will see in section 6 that this simplification introduces bias. Later MacLean *et al.* [13] derived the same residual as Bruss and Horn by a different route. In either case, a least-squares estimate of both depth and rotation can be obtained as a function of translation. These estimates are substituted back into the residual to obtain a function of translation alone, which is found by nonlinear minimization.

Rieger and Lawton [16] and later Hildreth [6] and Prazdny [14] proposed a method based on motion parallax. Jepson and Heeger built upon these previous efforts and proposed a series of subspace methods for estimating egomotion [5, 9, 10]. Given optical flow at m image points, they construct a set of $m - 6$ constraint vectors that are orthogonal to the camera translation velocity. From these, translational velocity is computed without requiring iterative numerical optimization. However, this method is biased, and does not make use of all of the available information ($m - 6$ linear constraints versus m bilinear constraints). We will see in section 6 that this increases variance.

From a problem formulation equivalent to equation (1), Zhuang *et al.* [21] proposed a linear algorithm for egomotion estimation based on the so-called *epipolar constraint* on the velocity field:

$$\mathbf{t}^T(\mathbf{x} \times \mathbf{u}) + \mathbf{x}^T K\mathbf{x} = 0 \tag{8}$$

where $K = \omega^T\mathbf{t}\, I - (\omega\mathbf{t}^T + \mathbf{t}\omega^T)/2$.

Since camera motion is a nonlinear function of image measurements, its estimates are usually biased. For instance, a nonlinear transformation $T(n)$ from a scalar input noise variable n to an output perturbation can be written as

$$T(n) = \sum_{k=0}^{\infty} c_k n^k \approx c_0 + c_1 n + c_2 n^2$$

where the c_k are Taylor coefficients. Even when $c_0 = 0$ and $E[n] = 0$, we have bias $E[T(n)] \approx c_2 E[n^2]$, which is nonzero if $c_2 \neq 0$. Kanatani [11] analyzed the bias of image motion analysis, and proposed a method that subtracts an estimate of the output bias from the solution. However, his bias cancellation assumes a narrow field of view, which leads to poor camera motion estimates in the first place. Furthermore, Kanatani's results, even when unbiased, exhibit unnecessarily high variance, as will be shown in section 6.

The somewhat disappointing results obtained in the literature, together with an analysis [19] based on the Cramer-Rao lower bound [15], led researchers to believe [19] that computation of camera motion from instantaneous image velocities is unlikely to succeed. In this paper we show that this is not so. We propose an algorithm for the computation of camera translation and rotation from motion field measurements that is consistent and fast. "Consistent" here means that as more image points are used, the estimates of camera motion converge in probability to their true values. This is a strong result, since for sufficiently many points it guarantees that both bias and variance are almost certainly small. In addition, we reduce sensitivity to outliers by using a less-than-quadratic error norm.

4 Consistency

If $(\hat{\mathbf{t}}, \hat{\omega})_m$ is the motion estimate obtained from m noisy measurements, and $(\mathbf{t}^*, \omega^*)$ are the true motion parameters,

consistency means that for any positive ϵ,

$$\lim_{m\to\infty} \mathcal{P}\left[\|(\hat{\mathbf{t}},\hat{\omega})_m - (\mathbf{t}^*,\omega^*)\| < \epsilon\right] = 1 .$$

In words, whatever the allowed, positive error ϵ, a sufficient ($m \to \infty$) number of measurements will almost certainly ($\mathcal{P}[\cdot] = 1$) yield an estimate $(\hat{\mathbf{t}},\hat{\omega})_m$ that is within ϵ of the true value $(\mathbf{t}^*,\omega^*)$.

To show that (2) and (4) are consistent estimators, two assumptions are required, one technical and one essential. The technical assumption is that the sample distribution of image noise approaches the true distribution as $m \to \infty$. This ergodicity assumption presents no problem. The essential assumption is that the true noise distribution is the same for all image points, and isotropic, *i.e.*, symmetric around the origin. Because of the aperture problem, positional uncertainty in images is not generally isotropic. However, a feature detector like the one in [18] leads to uncertainties that are very close to isotropic. With these assumptions, we have the following result.

Theorem. *If the errors on the image velocity measurements are independent, identically distributed, and isotropic, and if $f(\cdot)$ is rotationally symmetric and convex, then the estimator (4) is consistent.*

The proof of this theorem is very technical, and is omitted here for lack of space. Full details can be found in [20], and a proof of a similar flavor has been given in [8] under different assumptions and for different problems. These proofs use the central limit theorem as their main tool.

To be sure, expressions (2) or (4) have been the starting point for many if not most motion estimator algorithms. However, as mentioned above, past approaches have manipulated (2) or (4) one way or another to simplify computation. For instance, the Bruss-Horn approach is equivalent to the minimization of (7). The term $\|A(\mathbf{x})\mathbf{t}\|$ breaks rotational symmetry, and biases results toward a translation direction such that $\|A(\mathbf{x})\mathbf{t}\|$ with $\|\mathbf{t}\| = 1$ is small. By the definition of $A(\mathbf{x})$, if the coordinates (x_1, x_2) are small compared to 1, increasing the third component of $\mathbf{t}$ tends to give a smaller value of $\|A(\mathbf{x})\mathbf{t}\|$; on the other hand, if many of the coordinates (x_1, x_2) are large compared to 1, decreasing the third component of $\mathbf{t}$ tends to give a smaller value of $\|A(\mathbf{x})\mathbf{t}\|$. Since the former situation occurs when the camera's field-of-view (fov) is small, and the second when fov is large, the translation direction computed by Bruss and Horn biases toward forward motion with a small fov and toward side motion with a large fov. Both effects are shown by the simulations in section 6. The methods in [5, 9, 10] suffer from the same problem because they start from the same residual. This will be illustrated in section 6. Similarly, concerning [21], the term $\mathbf{t}^T(\mathbf{x}\times\mathbf{u})$ in equation (8) again breaks the rotational symmetry, and leads to similar problems. Although the method obtained in [11] is statistically more sensible, it starts from an inconsistent formula. Simulations in section 6 show that while the bias is not totally removed, this formulation yields a greater variance than (4).

5 The Algorithm

In this section, we develop an efficient numerical algorithm to solve the minimization problem (2). We first introduce the algorithm for the Least Squares case, and then extend it to p-norms. Finally, we describe a simple technique for global convergence, and discuss computation times.

Least Square Formulation. Let $p = 2$ in the estimator (2). The starting point of our algorithm is a Gauss-Newton updating procedure, which converges fast, and is simple and stable.

Given a residual $R(\theta)$, where $\theta = (\mathbf{t},\omega,\mathbf{d})$ is the unknown parameter, Gauss-Newton determines a descent step $\Delta\theta_k$ from the current estimate θ_k by approximating $R(\theta)$ with a truncated Taylor series

$$R(\theta_k + \Delta\theta_k) \approx R(\theta_k) + \mathbf{g}_k^T + \frac{1}{2}\Delta\theta_k H_k \Delta\theta_k \quad (9)$$

where $\mathbf{g}_k$ and H_k are gradient and Hessian of R at θ_k:

$$\mathbf{g}_k = J_k\rho_k \quad \text{and} \quad H_k = J_k^T J_k + \sum_{i=1}^{2m} \rho_i H_{ik}$$

where J_k is the Jacobian of $\rho(\theta) = [\mathbf{r}_1,\ldots,\mathbf{r}_m]^T$, and H_{ik} is the Hessian of $\rho_i(\theta)$. If the residual ρ at the solution is small, the approximation $H_k \approx J_k^T J_k$ can be made. Differentiating (9) with respect to $\Delta\theta_k$ and setting the result to zero yields the linear system

$$J_k\Delta\theta_k = -\rho_k , \quad (10)$$

which is the basic step of the Gauss-Newton method. For our problem, the system (10) has m linear equations of the form

$$d_k(\mathbf{x})A(\mathbf{x})\Delta\mathbf{t}_k + A(\mathbf{x})\mathbf{t}_k\Delta d_k(\mathbf{x}) + B(\mathbf{x})\Delta\omega_k = \mathbf{r}_k(\mathbf{x}) . \quad (11)$$

The constraint $\|\mathbf{t}\| = 1$, by differentiation, yields the additional linear equation $\mathbf{t}_k^T\Delta\mathbf{t}_k = 0$.

Much better performance than straight Gauss-Newton is obtained by observing that θ is separable [1, 4, 3, 17] as $\theta = (\mathbf{t},(\omega,\mathbf{d}))$, in the sense that if $\mathbf{t}_k$ is known then ω_k and $d_k(\mathbf{x})$ can be simply computed by first solving the linear problem

$$\omega_k = \arg\min_{\omega} \frac{1}{m}\sum_{\{\mathbf{x}\}} f(\tau_k(\mathbf{x})(\mathbf{u}(\mathbf{x}) - B(\mathbf{x})\omega)) \quad (12)$$

(see (6)). Once $\mathbf{t}_k$ and ω_k are available, inverse depth $d_k(\mathbf{x})$ can be computed from (5).

Thus, while the Gauss-Newton iteration (11) yields estimates of $\Delta\mathbf{t}_k$, $\Delta\omega_k$, $\Delta d_k(\mathbf{x})$, only $\Delta\mathbf{t}_k'$ is used, while $\Delta\omega_k$, $\Delta d_k(\mathbf{x})$ are re-estimated via (12) and (5) for greater accuracy. As discussed in [1, 4, 3], this amounts to doing Gauss-Newton in $\mathbf{t}$, a three-dimensional vector, rather than in the $m+6$-dimensional vector $(\mathbf{t}, \omega, \mathbf{d})$. The computational advantage is obvious.

Robust Formulation. Some degree of robustness can be achieved by using a function $f(x)$ in (4) that grows more slowly than x^2: $\lim_{x\to\infty} f(x)/x^2 = 0$. This is equivalent to introducing a weight factor $w(x)$ so that $f(x) = w^2(x)x^2$ and $\lim_{x\to\infty} w(x) = 0$:

$$\min_{\mathbf{t},\omega,\mathbf{d}} \sum_{\{\mathbf{x}\}} w^2(\mathbf{x})\|\mathbf{r}(\mathbf{x})\|_2^2 \tag{13}$$

where $w(\mathbf{x}) = w(\|\mathbf{r}(\mathbf{x})\|_2)$. The initial weights $w(\mathbf{x})$ are set to 1, and computation interleaves estimates of $\Delta\mathbf{t}_k$, $\Delta\omega_k$, $\Delta d_k(\mathbf{x})$ with recomputation of the $w(\mathbf{x})$.

Global Optimization. The above algorithm converges to a local minimum very quickly, usually within three to five iterations. In order to find the global minimum, we start from 15 initial points evenly spread on the unit hemisphere for $\mathbf{t}$. The number of iterations can be reduced by eliminating a fraction of branches having the largest residues at each step, as well as merging two branches when they become close to each other. Each branch is stopped when $\mathbf{t}$ changes by less than $0.05°$. A typical run requires a total of about 40 Gauss-Newton steps, and converges essentially all the time to the correct minimum.

Running Time. Each reweighted Gauss-Newton update requires $O(m)$ time. Around $90m$ floating point operations (flops) are required per point and iteration. With 100 selected features ($m = 100$), the total number of flops for the global optimization is about $40 \times 90 \times 100 = 3.6 \times 10^5$. In our experiments (with $m = 100$) on a Sun Sparc workstation, global optimization, written in C, takes on average less than 0.05 seconds per image.

6 Simulations and Experiments

In this section, the following algorithms are compared: (i) Bruss-Horn: the algorithm in [2]. (ii) Jepson-Heeger: the linear subspace method in [9, 10]. (iii) Kanatani: the renormalization method in [11]. (iv) RM-L2: the least square formulation described in this paper. (v) RM-L1.2: the robust formulation in this paper with $f(x) = |x|^{1.2}$.

100 features are used unless stated otherwise. Multiple runs are reported for each experiments. The directions computed by each run are plotted on a hemisphere, which is scaled to be uniform from $0° - 90°$ in the radial direction. The true camera translation and rotation directions are denoted by the symbols $\circ$ and $\times$, respectively. The computed camera translation directions are plotted as a dot for each run. Average computed translation direction and rotation axis are denoted as $\triangleright$ and $+$. The size of rotation in degrees is also reported ("$|R|$" as the computed average, "true $|R|$" as the true size of rotation) on each plot. The tables in each figure report the errors using the format "error mean $\pm$ error standard deviation." For translation direction, error is defined as the angle in degrees; for rotation, error is defined as the 2-norm in degrees of the difference between computed and true rotation.

Simulations. The scene is randomly synthesized 3D positions with depth uniformly distributed between 1 and 4 focal lengths. The projected image coordinates $\mathbf{x}$ are uniformly distributed in an image of size 512×512. A hundred runs are made for each experiment. For all simulations, we fix the translation direction as $[4, -3, 5]$ and rotation direction as $[-1, 2, 0.5]$. Noise is Gaussian, and is measured by the signal to noise ratio (SNR): $(E\|\mathbf{u}\|_2^2)^{1/2} : (E\|\mathbf{n}\|_2^2)^{1/2}$.

Our first experiment, in figure 2, assumes fov=50°. Noise $\sigma = 0.5$ pixel, which leads to a 6:1 SNR. With relatively small fov, the Bruss-Horn and Jepson-Heeger formulations compute a translation direction biased towards the center, as predicted. The other three algorithms give results that are comparable to one another.

The experiment in figure 3 changes fov to 150°. Noise standard deviation is still $\sigma = 0.5$ pixel, with SNR around 10:1. Except RM-L2 and RM-L1.2, all algorithms give estimation of translation that is biased toward the side.

The experiment in figure 4 studies robustnes, with 50° fov. Noise is a mixture of Gaussians, in which 90% of the points has a 6:1 SNR, and the other 10% has 1:1 SNR. Bruss-Horn and Jepson-Heeger are still biased, while the other three methods are relatively unbiased. However, Kanatani gives a greater variance than RM-L2, which in turn gives a greater variance than RM-L1.2.

The experiment in figure 5 verifies our theoretical claim of consistency. The data are generated in exactly the same way as in the third experiment, except that the sample size increases from 100 to 2000. The bias of Bruss-Horn and Jepson-Heeger cannot be corrected by adding more data.

Experiments with Real Images. The last dataset, figure 6, is a pair of real images (block dataset) taken with a 16mm lens on a Panasonic camera (sensor area 8.8mm × 6.6mm). The field of view is thus approximately 31°, which is fairly narrow. The image resolution is 720×480 with a horizontal to vertical aspect ratio of $1.25 : 1$. The distance between the camera center and the V block is about 40.6cm, and the distance to the background is about 58.5cm. This pair only contains lateral motion, without any rotation. 150 features are automatically tracked by the method in [18]. Figure 1 shows one image.

7 Conclusions

The theory, simulations, and experiments in this paper emphasize the importance of consistency in the design of algorithms for the recovery of camera motion. Algebraic manipulation of the standard residual can lead to both bias and excessive variance in the results. Direct minimization of the standard residual, on the other hand, leads to smaller variance and negligible bias, and the use of p-norms increases robustness to a moderate amount of outliers. Consistency and robustness are not expensive, and a method using Gauss-Newton for quadratic convergence and special techniques that exploit the bilinear nature of the problem can work at near frame rate without special hardware. More systematic experiments are in progress to test the validity of the underlying assumption. We are also applying the principle of consistency to more general structure from motion algorithms.

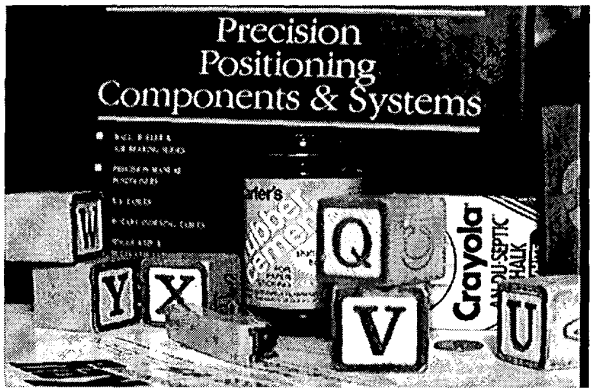


Figure 1: One of the block images.

References

[1] C. Böckmann. A modification of the trust-region Gauss-Newton method to solve separable nonlinear least squares problems. *J. Math. Systems Estim. Control*, 5:111–114, 1995.

[2] A. R. Bruss and B. K. P. Horn. Passive navigation. *CVGIP*, 21:3–20, 1983.

[3] D. M Gay and L. Kaufman. Tradeoffs in algorithms for separable nonlinear least squares. *Computational and applied mathematics, I*, pages 179–183, 1992.

[4] G. H. Golub and V. Pereyra. The differentiation of pseudo-inverses and nonlinear least squares problems whose variables separate. *SIAM J. on Numerical Analysis*, 10(2):413–432, 1973.

[5] D. J. Heeger and A. D. Jepson. Subspace methods for recovering rigid motion i: Algorithm and implementation. *IJCV*, 7(2):95–118, 1992.

[6] E. C. Hildreth. Recovering heading for visually-guided navigation. *Vision Res.*, 32(6):1177–1192, 1992.

[7] B. K. P. Horn. Motion fields are hardly ever ambiguous. *IJCV*, 1:259–274, 1987.

[8] P. J. Huber. *Robust Statistics*. Wiley, 1981.

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

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

[11] K. Kanatani. 3-d interpretation of optical flow by renormalization. *IJCV*, 11(3):267–282, 1993.

[12] E. Kruppa. Zur Ermittlung eines Objektes aus zwei Perspektiven mit innerer Orientierung. *Sitz.-Ber. Akad. Wiss., Wien, Math. Naturw. Kl., Abt. IIa.*, 122:1939–1948, 1913.

[13] W. J. MacLean, A. D. Jepson, and R. C. Frecker. Recovery of egomotion and segmentation of independent object motion using the em algorithm. In *Proc. 5th British Machine Vision Conference*, pages 13–16, 1994.

[14] K. Prazdny. On the information in optical flows. *CGIP*, 22:239–259, 1983.

[15] C. R. Rao. *Linear Statistical Inference and Its Applications*. Wiley, 1973.

[16] J. H. Rieger and D. T. Lawton. Processing differential image motion. *JOSA*, A(2):254–360, 1985.

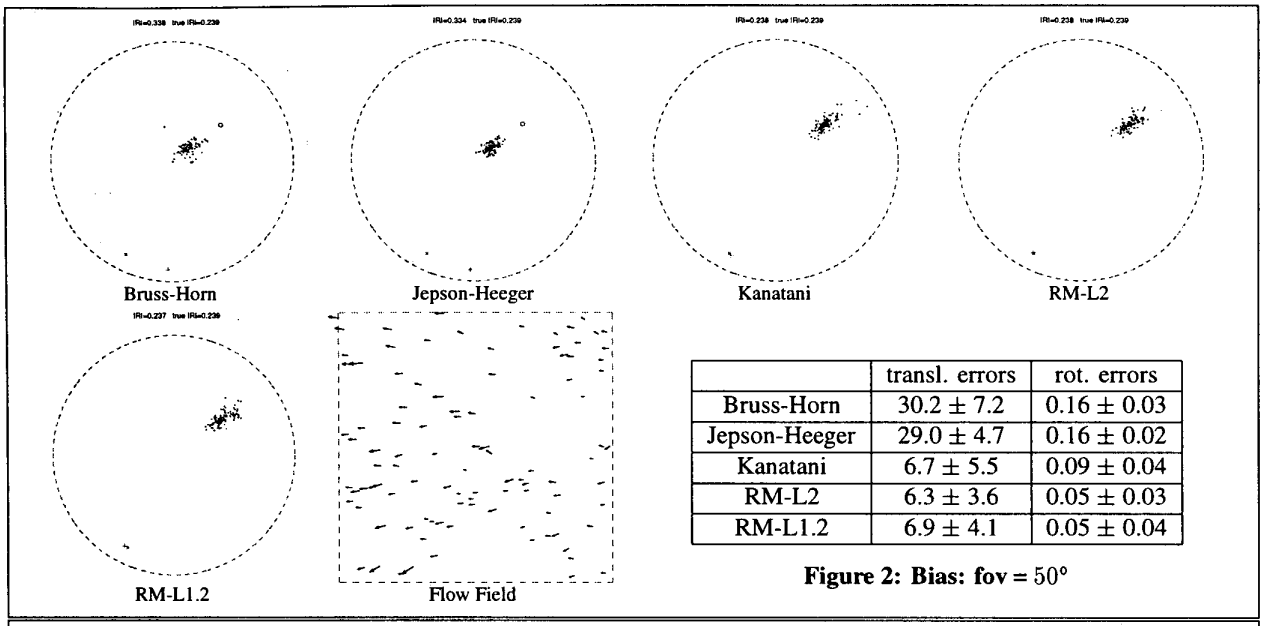
[17] A. Ruhe and P. Å. Wedin. Algorithms for separable nonlinear least squares problems. *SIAM Review*, 22(3):318–337, 1980.

[18] J. Shi and C. Tomasi. Good features to track. In *CVPR94*, pages 593–600, 1994.

[19] J. Weng, N. Ahuja, and T. S. Huang. Optimal motion and structure estimation. *PAMI*, 15(9):864–884, 1993.

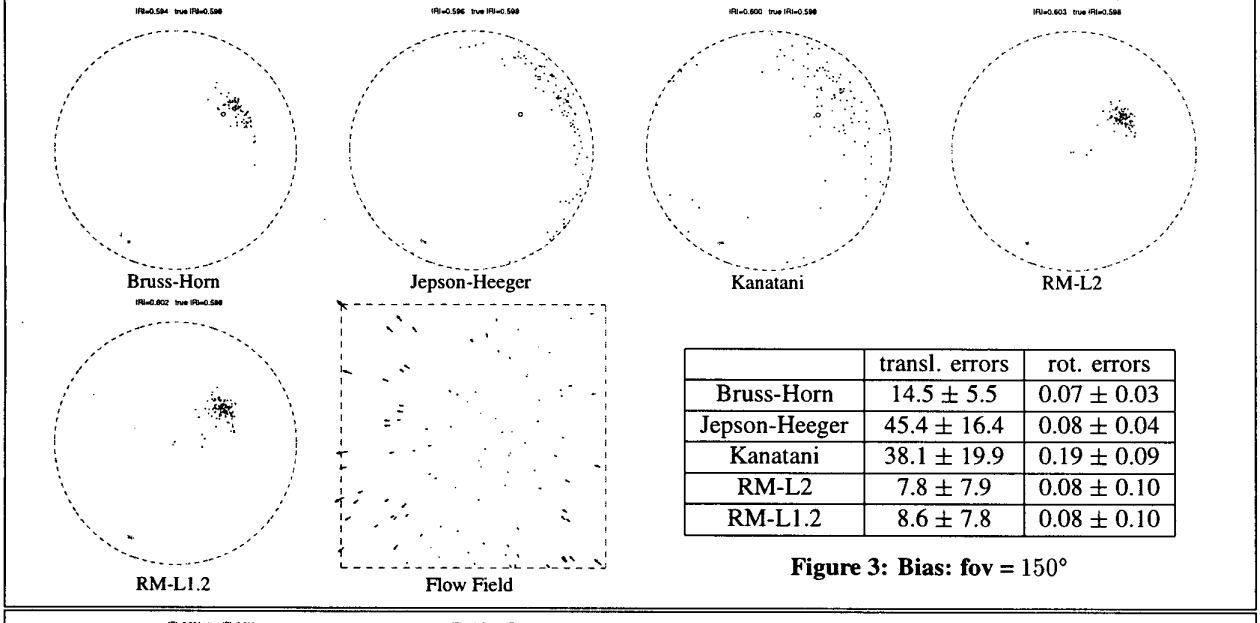
[20] T. Zhang. *Methods for Computational and Statistical Estimation with Applications*. PhD thesis, Stanford University, November 1998.

[21] X. Zhuang, T.S. Huang, N. Ahuja, and R.M. Haralick. A simplified linear optic flow-motion algorithm. *CVGIP*, 42(3):334–344, 1988.



	transl. errors	rot. errors
Bruss-Horn	30.2 ± 7.2	0.16 ± 0.03
Jepson-Heeger	29.0 ± 4.7	0.16 ± 0.02
Kanatani	6.7 ± 5.5	0.09 ± 0.04
RM-L2	6.3 ± 3.6	0.05 ± 0.03
RM-L1.2	6.9 ± 4.1	0.05 ± 0.04

Figure 2: Bias: fov = 50°



	transl. errors	rot. errors
Bruss-Horn	14.5 ± 5.5	0.07 ± 0.03
Jepson-Heeger	45.4 ± 16.4	0.08 ± 0.04
Kanatani	38.1 ± 19.9	0.19 ± 0.09
RM-L2	7.8 ± 7.9	0.08 ± 0.10
RM-L1.2	8.6 ± 7.8	0.08 ± 0.10

Figure 3: Bias: fov = 150°

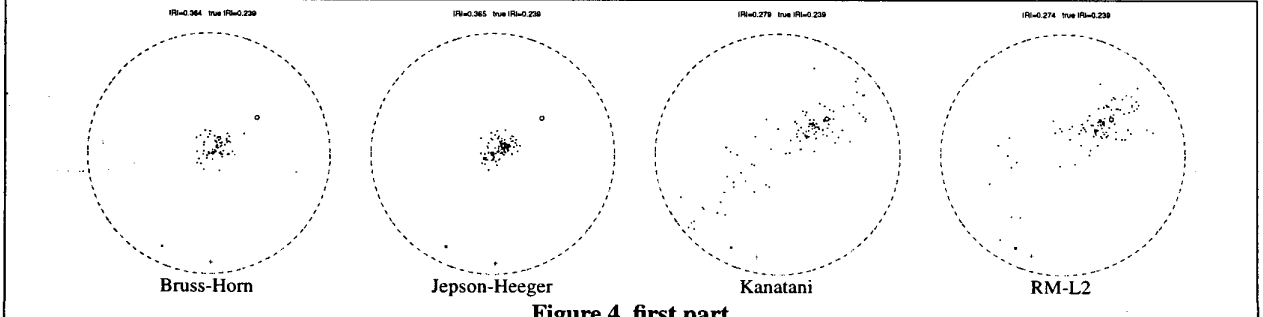
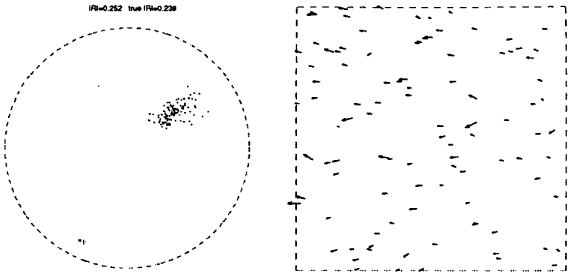
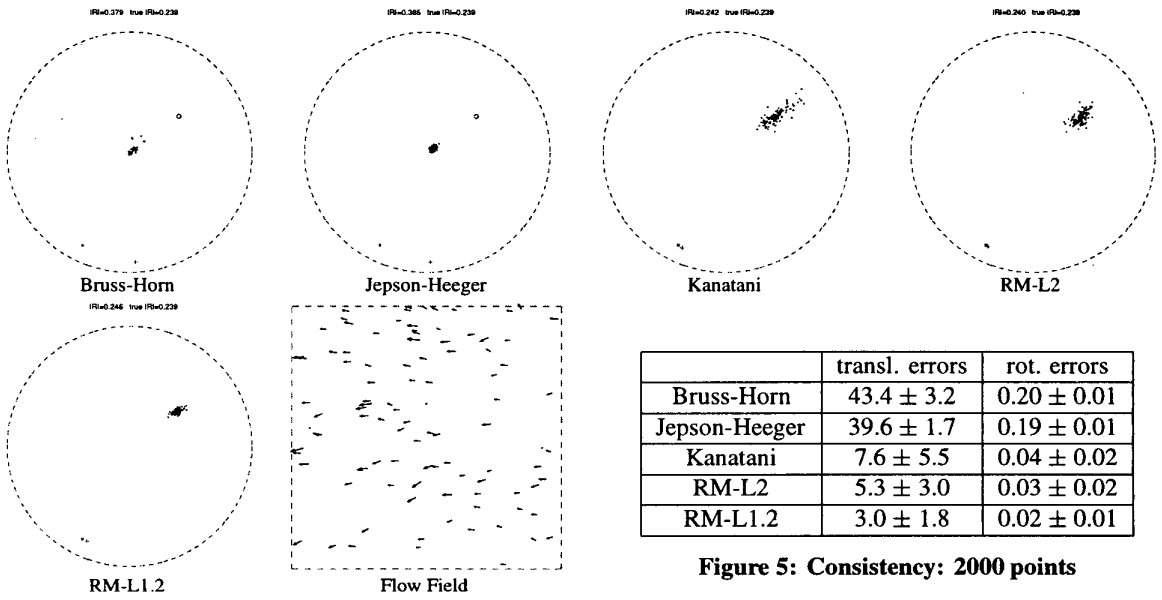


Figure 4, first part.



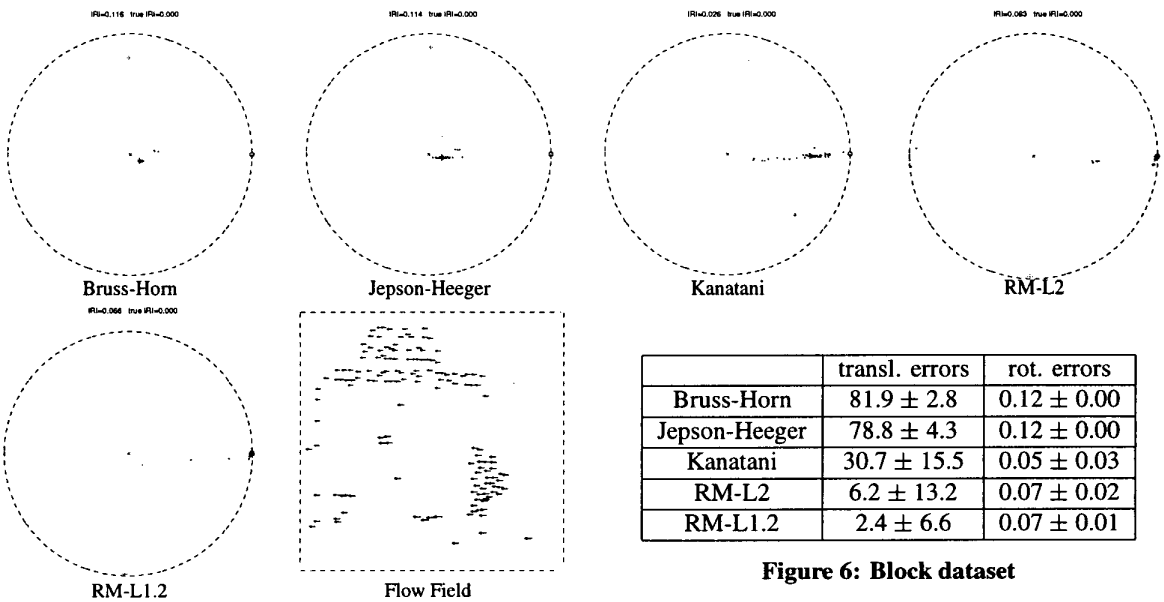
	transl. errors	rot. errors
Bruss-Horn	39.3 ± 7.8	0.20 ± 0.04
Jepson-Heeger	39.0 ± 7.6	0.19 ± 0.04
Kanatani	32.1 ± 24.9	0.17 ± 0.10
RM-L2	22.0 ± 19.9	0.15 ± 0.12
RM-L1.2	9.5 ± 5.2	0.08 ± 0.05

Figure 4, ct'd: Robustness: fov= 50°



	transl. errors	rot. errors
Bruss-Horn	43.4 ± 3.2	0.20 ± 0.01
Jepson-Heeger	39.6 ± 1.7	0.19 ± 0.01
Kanatani	7.6 ± 5.5	0.04 ± 0.02
RM-L2	5.3 ± 3.0	0.03 ± 0.02
RM-L1.2	3.0 ± 1.8	0.02 ± 0.01

Figure 5: Consistency: 2000 points



	transl. errors	rot. errors
Bruss-Horn	81.9 ± 2.8	0.12 ± 0.00
Jepson-Heeger	78.8 ± 4.3	0.12 ± 0.00
Kanatani	30.7 ± 15.5	0.05 ± 0.03
RM-L2	6.2 ± 13.2	0.07 ± 0.02
RM-L1.2	2.4 ± 6.6	0.07 ± 0.01

Figure 6: Block dataset