Spectrum analysis of motion parallax in a 3-D cluttered scene and application to egomotion

Richard Mann
Dept. of Computer Science
University of Waterloo
Waterloo, Canada

Michael S. Langer
School of Computer Science
McGill University
Montreal, Canada

Previous methods for estimating observer motion in a rigid 3D scene assume that image velocities can be measured at isolated points. When the observer is moving through a cluttered 3D scene such as a forest, however, point-wise measurements of image velocity are more challenging to obtain because multiple depths and hence multiple velocities are present in most local image regions. This paper introduces a method for estimating egomotion which avoids point-wise image velocity estimation as a first step. In its place, the direction of motion parallax in local image regions is estimated, using a spectrum-based method, and these directions are then combined to directly estimate 3D observer motion. There are two advantages to this approach. First, the method can be applied on a wide range of 3D cluttered scenes, including those for which pointwise image velocities cannot be measured because only normal velocity information is available. Second, the egomotion estimates can be used as a posterior constraint on estimating pointwise image velocities, since known egomotion parameters constrain the candidate image velocities at each point to a 1D rather than a 2D space. © 2004 Optical Society of America

1. Introduction

When an observer moves through a 3D scene, nearby surfaces move at different image speed than do distant surfaces. This *motion parallax* effect informs an observer both about its own motion relative the scene, called *egomotion*, and about the spatial layout and depth of surfaces in the scene. This paper concentrates on a specific natural scenario that has been relatively neglected in the past, in which the scene is composed of a very large number of distinct surfaces at a large range of depths. We refer to such scenarios as 3D cluttered scenes. For such scenes, most local image regions contain multiple visible surfaces, and hence multiple depths. Psychophysical studies have shown that humans observers can estimate their direction of heading in such scenes.[1–3]

A canonical natural example of a 3D cluttered scene is the woods. Helmholtz observed that, from a fixed vantage point in the woods and using one eye only, an observer has difficulty segmenting the scene into distinct visible surfaces [4, p.295]. The difficulty arises from the abundance of depth discontinuities, surface textures, shadows etc, which are confounded in the retinal image projection. He also claimed that as soon as the observer begins to move, he immediately perceives the 3-D layout of the scene and the positions of surfaces both relative to each other and to himself.

Helmholtz'es scenario of a 3D cluttered scene presents a challenge to traditional computational models of egomotion and 3D structure from motion. Traditional models assume a two step process in which the observer first measures point-wise image velocities by optical flow or feature tracking, and second combines these image velocities to estimate egomotion and 3D structure. Although this two-step process may seem attractive from a computational standpoint since it partitions the problem into independent modules, it is unclear *a priori* that this is the best computational approach to solving the problem in all scenarios. In particular, the first step of measuring pointwise image velocities is complex in a 3D cluttered scene, since multiple image velocities can occur within many local regions. To obtain a dense image velocity field, a vision system would need to choose from a variety of methods such as optical flow,[5] occlusion boundary detection,[6,7] layered motion,[8] and even transparency..[9] We emphasize that, in the traditional two-step approach, these motion estimation methods are applied prior to the estimation of egomotion and scene structure.

In this paper we present an alternative approach which applies to 3D cluttered scenes. Rather than estimating pointwise image velocities as a first step, the observer instead estimates the direction of motion parallax in a small number of local regions. These estimates are obtained using the local power spectrum of the image sequence. The directions of motion parallax are then used to estimate the observer 3D egomotion.

There are two advantages of our method for the case of 3D cluttered scenes. First, our method can estimate egomotion in scenes for which no previous egomotion method exists. We present one such example, namely scenes consisting entirely of shaded cylinders at random 3D orientations. For such scenes, only normal image velocities can be computed and so egomotion methods that are based on 2D image velocities cannot be applied. Although there do exist egomotion methods that only require normal velocities to be computed,[10–13] these methods make very restrictive assumptions on the observer motion, namely that the rotation component is either zero or is known to great accuracy. Our method makes no such assumption. Hence, our method is able to estimate egomotion in situations for which no previous method exists.

A second advantage of our method concerns the subsequent problem of estimating 3D scene geometry, i.e. the depths of visible surfaces. Methods for estimating depth from motion require pointwise image velocities. How does our method help to estimate such pointwise image velocities for the case of cluttered 3D scenes? While we do not address and solve that problem in this paper, we do make a key observation that the parameters of egomotion which our method *does* compute strongly constrain the solution to the problem of estimating pointwise image velocities. Specifically, once the observer's rotation and translation are known, candidate image velocities at each point are constrained to a line (an epipolar line[14]) in velocity space. Thus, estimating egomotion *prior to* estimating image velocities simplifies the latter problem. This argument applies in principle regardless of which pointwise image velocity method is used, be it optical flow or a more general method that allows for occlusions or transparency.

A preliminary version of this work has been presented previously[15]

2. Background

We begin by reviewing motion parallax and how it can be used to estimate egomotion and 3D structure.[16] When an observer moves through a rigid scene, the instantaneous retinal image velocity of a visible scene point depends on the observer's instantaneous 3D motion relative to the scene. Let (X, Y, Z) be a coordinate system with the observer at the origin, Z is the optical axis and X, Y are the horizontal and vertical directions in the observer's frame. Let the observer's 3D instantaneous translation velocity in this coordinate system be $\mathbf{T} = (T_x, T_y, T_z)$ and let the observer's instantaneous angular velocity be $\mathbf{\Omega} = (\Omega_x, \Omega_y, \Omega_z)$. Let the image plane be at depth $Z = f$ and let $Z(x, y)$ be the depth of any surface point visible at image position (x, y).

The image velocity vector $\vec{v}$ at (x, y) is the sum of two component vectors,

$$\vec{v} = \vec{v}_\Omega + \vec{v}_T$$

which are due to the observer's rotation $\mathbf{\Omega}$ and translation $\mathbf{T}$ respectively. The rotation component is

$$\vec{v}_\Omega = \begin{bmatrix} xy/f & -f - x^2/f & y \\ f + y^2/f & -xy/f & -x \end{bmatrix} \begin{bmatrix} \Omega_x \\ \Omega_y \\ \Omega_z \end{bmatrix} \tag{1}$$

and the translation component is

$$\vec{v}_T = \frac{T_z}{Z(x, y)} \begin{bmatrix} x - x_T \\ y - y_T \end{bmatrix} . \tag{2}$$

The special image position

$$(x_T, y_T) \;=\; \frac{f}{T_z}(T_x, T_y) \tag{3}$$

is called the *axis of translation* (AOT). In the special case of lateral motion, namely where $T_z = 0$, the AOT is at infinity in the image plane. In this case, the translation component is written slightly differently,

$$\vec{v}_T = \frac{f}{Z(x,y)} \begin{bmatrix} -T_x \\ -T_y \end{bmatrix} \quad . \tag{4}$$

There are two important observations about these equations. First, there is a depth–speed ambiguity in recovering depth $Z(x, y)$ and the translation vector $\mathbf{T}$. Multiplying both $Z(x, y)$ and $\mathbf{T}$ by a constant does not change the velocity field. For this reason, the vision system can at best estimate these variables up to the unknown scale factor.

Second, if the observer knows $\mathbf{\Omega}$ and the direction of $\mathbf{T}$ (and the parameter f) then the observer could compute immediately the rotation vector $\vec{v}_\Omega$ and the direction of the translation vector $\vec{v}_T$ at each image position (x, y). These values are given directly from the above equations. To compute the depth $Z(x, y)$, the observer would only need to know the magnitude of $\vec{v}_T$ at each (x, y).

To compute depth $Z(x, y)$ given $\mathbf{\Omega}$ and the direction of $\mathbf{T}$, the observer needs to know the image velocity at (x, y). But now we see that this velocity is constrained to a 1D line, namely the line passing through $\vec{v}_\Omega$ and in the direction of $\vec{v}_T$, both of which are given by $\mathbf{\Omega}$ and the direction of $\mathbf{T}$. Thus, knowing $\mathbf{T}$ and $\mathbf{\Omega}$ *prior to* estimating pointwise image velocities simplifies this latter problem, by reducing the search space from 2D to 1D. This helps to motivate the reordering of computations which we present in this paper, namely to compute $\mathbf{T}$ and $\mathbf{\Omega}$ prior to estimating pointwise image velocities, rather than the other way around.

Before we introduce our method, we review a traditional approach in which an observer estimates image velocities prior to estimating observer motion.

This particular approach is then reformulated into our approach.

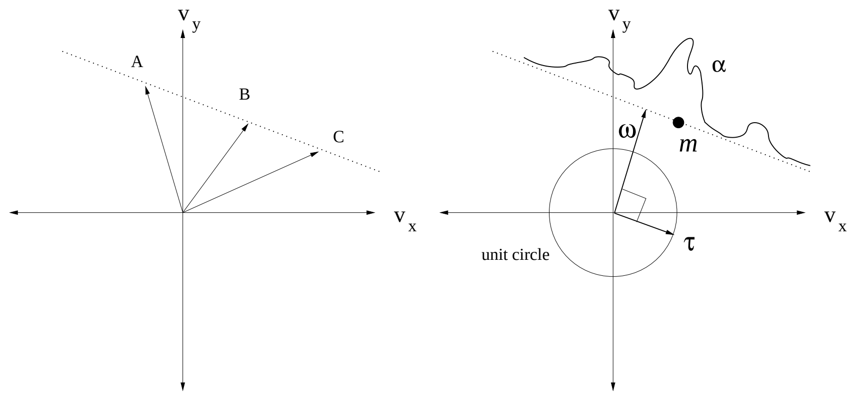
How can an observer estimate the rotation vector $\mathbf{\Omega}$ and the direction of $\mathbf{T}$ (AOT), given pointwise image velocities? A classical idea[16] is to use motion parallax of pairs of points that straddle a depth discontinuity. Assume first that the image projections of two points are an infinitesimal distance apart. Because the rotation component of the velocity field does not depend on depth, the difference of the velocity vectors of the two points depends only on their translation components. But because the two points lie at different depths, the translation components of the two velocity vectors have different magnitude. Since both of these translation components point away from the AOT, the difference of the two velocity vectors must also point away from (or toward) the AOT. Thus the AOT lies on a line that passes through the depth discontinuity and whose direction is defined by the velocity difference vector. By computing a set of such lines and finding their intersection, a vision system can estimate the AOT. Rieger and Lawton[17] presented the first computer vision implementation of this idea.

The method we present uses a similar idea, namely to isolate a direction vector that points to the AOT. The novelty of our method is that it does not rely on pre-computed pointwise image velocities, but instead estimates this direction vector directly from image intensities.

3. Model of motion parallax

Our method is based on a linear model of motion parallax in a local image region. We first motivate this model using informal arguments, and then present the formal model itself. In Appendix A, we analyze the accuracy of the model by providing bounds on the deviation of the variables in the model as a function of scene geometry and observer motion.

Consider three surfaces in a local image region and one point A, B, C on each of these surfaces. For each pair of points AB and AC, there is a corresponding pair of velocity vectors. According the differential motion idea,[16,17]



(a) (b)

Fig. 1. (a) Image velocity vectors at three points A, B, C in a local image region. The three points lie on distinct surfaces. From Rieger and Lawton's arguments, the differences of any pair of these vectors should point to the AOT. It follows that the vectors must lie on a line in velocity space. (b) Motion parallax line (dotted). $\vec{\tau}$ is a unit vector in the direction of the translation component. $\vec{\omega}$ is the component of the rotation vector that is perpendicular to $\vec{\tau}$. α is a set of component speeds in the direction-of-translation. $\vec{m} = (m_x, m_y)$ is the mean velocity vector which is used in Sec. 4.

the difference vectors $\vec{v}(A) - \vec{v}(B)$ and $\vec{v}(A) - \vec{v}(C)$ should point roughly towards or away from the AOT. Specifically, if three points are near to each other and are far from the AOT, then these two difference vectors are roughly parallel. It follows that the image velocity vectors at A, B, C fall roughly on a line in velocity space (v_x, v_y). See Figure 1a. We refer to the line as the *motion parallax line* for the local image region containing A, B, C.

Another way to motivate the model is to use the smoothness of the rotation field and of the direction of the translation field. The rotation field is a second order polynomial of image position so it is clearly smooth, and hence locally constant. The direction of the translation field is a field of unit vectors that point away from the AOT. With the exception of the singularity at the AOT, the direction of the unit vectors is also smooth and hence locally constant.

Fig. 2 illustrates these two smoothness properties. Each plot shows a 7×7

sampling of the respective field, with neighboring samples differing in position by about 4 degrees of visual angle. The field of view is a 30° wide which is typical for a video camera. If we partition the field of view into square regions, with one region per sample point in the figure, then we expect the rotation fields and direction of translation fields *within each region* to be near constant. The reason is that the fields themselves are smooth. Since the fields change very little across regions from sample point to sample point (except near the AOT), we would expect them to change little within regions.

Of course, the image velocity field $\vec{v}$ is not expected to be smooth since the translation component of velocity depends on inverse depth via Eqs. (2) and (4). Depth is expected to be discontinuous often in cluttered 3D scenes, and depth differences lead to velocity discontinuities. The key, however is that since the *direction* of the translation component is roughly constant within regions, the translation component vectors will be roughly parallel within regions. This again leads to a line model of image velocities within regions, namely a near constant rotation component plus a translation component whose direction is near fixed but whose speed varies over image position.

With these motivating arguments behind us, we now formally state the motion parallax line model. Consider a image region i that is a centered at (x_i, y_i). For the image velocity vector $\vec{v}$ at any position (x, y) in this region, define $\vec{\tau}$ to be the unit vector that is parallel to the translation component $\vec{v}_T$ and that points away from the AOT. Let $\vec{\tau}_i$ be the $\vec{\tau}$ vector at (x_i, y_i). We refer to $\vec{\tau}_i$ as the *motion parallax direction* for region centered at (x_i, y_i). The model uses $\vec{\tau}_i$ to approximate all $\vec{\tau}$ vectors in region i. We define the *motion parallax line* for region i to be:

$$\vec{v}(x,y) \ = \ \vec{\omega}_i \ + \ \alpha(x,y) \ \vec{\tau}_i \tag{5}$$

The 2D vector $\vec{\omega}_i$ is the component of the rotation vector $\vec{v}_\Omega$ at (x_i, y_i) that is perpendicular to $\vec{\tau}_i$. The geometry is illustrated in Fig.1b.

In Appendix A, we present bounds on how well this line indeed describes the image velocities in a local region centered at (x_i, y_i). We also apply these

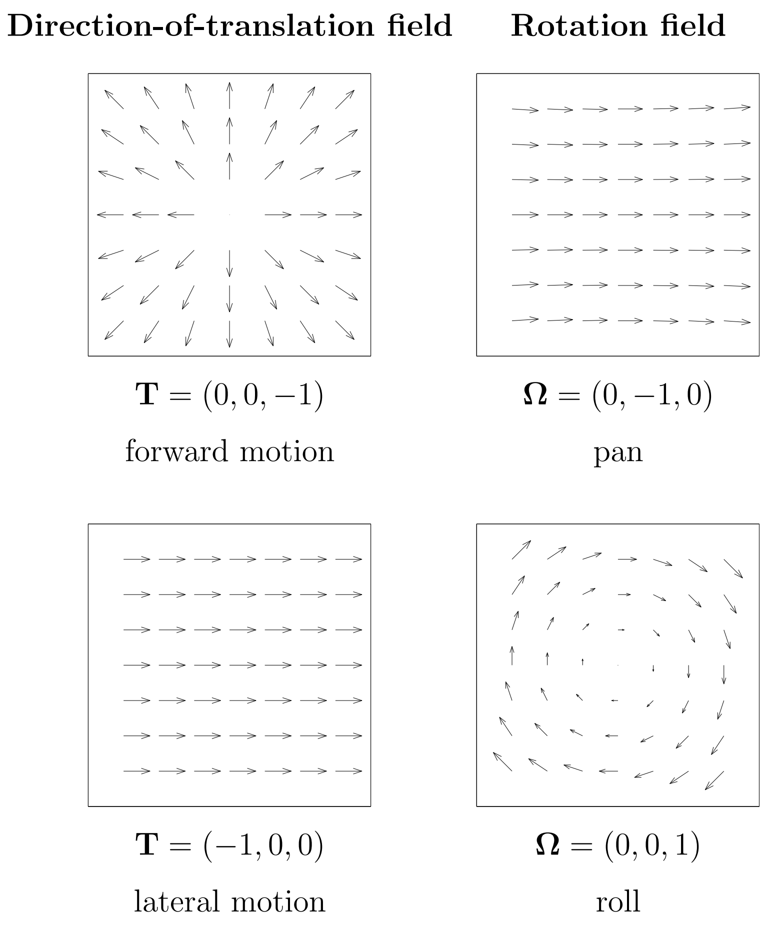

Direction-of-translation field
Rotation field
$\mathbf{T} = (0, 0, -1)$
forward motion
$\boldsymbol{\Omega} = (0, -1, 0)$
pan
$\mathbf{T} = (-1, 0, 0)$
lateral motion
$\boldsymbol{\Omega} = (0, 0, 1)$
roll

Fig. 2. 7×7 sampling of a 30 degree field of view. (left) direction-of-translation (normalized) vectors. (right) rotation vectors. The motion parallax line model assumes that the fields shown are constant over a image region with diameter equal to distance between samples.

bounds using the scene parameters from our experiments. We show that the motion parallax line is a valid model provided that there is a sufficient range of depths and that the region width is small relative to the distance to the AOT.

In the following section we present a method for estimating the motion parallax line for an image region. This method does *not* rely on pre-computed pointwise image velocities.

4. Method for estimation of motion parallax direction

Our method for estimating the direction of motion parallax $\vec{\tau}_i$ for a region i generalizes the optical snow method,[18] which is a power spectrum method for estimating a motion parallax line. The optical snow method assumes that the rotation component of the image velocities is parallel to the translation component, in particular, it assumes $\vec{\omega}_i = 0$ in Eq. (5). This assumption is not valid under general camera rotation, however. To apply the method we need to generalize it to the case that $\vec{\omega}_i \neq 0$.

To generalize the optical snow method, we add an initial step which is to transform the power spectrum of the local image region. This initial step amounts to motion compensation. It shears the power spectrum, effectively subtracting an estimate of the mean image velocity in that image region. This mean velocity estimate is obtained using a multiscale least-squares technique. By subtracting the mean velocity, the motion parallax line of Eq. (5) is shifted so that it passes through the origin. Once this initial step is performed, the assumption of the optical snow method holds and so this method can be applied to estimate the direction of motion parallax $\vec{\tau}_i$ for the region.

4.A. Background: motion parallax in the frequency domain

We begin with a brief review of the power spectrum model which is the basis of the optical snow method and our generalization of this method. See Ref. [18] for more details.

When an image region contains an intensity pattern that translates over

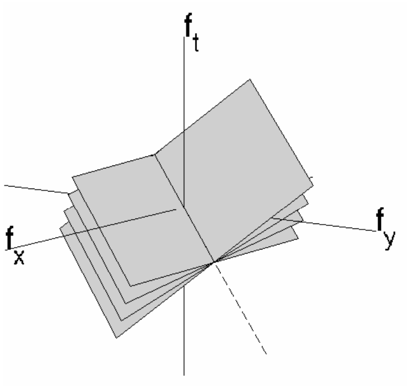


Fig. 3. **Bowtie.** Motion parallax in frequency domain.

time with velocity (v_x, v_y), non-zero values in the power spectrum are confined to a *motion plane*,[19]

$$v_x f_x + v_y f_y + f_t \; = \; 0 \tag{6}$$

where (f_x, f_y) are the spatial frequencies and f_t is the temporal frequency. When multiple image velocities are present in a region and these velocities lie on a line in velocity space, Eq. (5) applies and can be substituted into Eq. (6) to obtain a one parameter (α) family of motion planes,

$$(\vec{\omega}_i + \alpha \; \vec{\tau}_i) \;\; \cdot (f_x, f_y) + f_t \; = \; 0 \; . \tag{7}$$

As illustrated in Fig. 3, this family of motion planes intersects at a common line that passes through the origin in the 3D frequency domain. This family of planes is referred to as a *bowtie* and the line of intersection of the planes is referred to as the *axis of the bowtie.* The axis of the bowtie is in direction $(-\tau_y, \; \tau_x, \; \sqrt{\omega_x^2 + \omega_y^2})_i$. Thus, to estimate the direction of motion parallax $\vec{\tau}_i$ and the vector $\vec{\omega}_i$ for image region i, it is sufficient to estimate this bowtie axis.

The optical snow method estimates the bowtie axis for the special case that $\vec{\omega} = 0$. In this case, the bowtie axis lies in the (f_x, f_y) plane. The method begins with a brute force search of candidate directions $\vec{\tau}_i$ on the unit circle in the (f_x, f_y) plane. The directions are sampled every 5 degrees. For each direction, a goodness of fit is computed for how well the power spectrum resembles a bowtie whose axis is in that direction. The direction of best fit is found, and a local binary search refines this initial $\vec{\tau}_i$ estimate.

4.B. Motion compensation in local image regions

The optical snow method assumes that the bowtie axis is contained in the (f_x, f_y) plane, i.e. $\vec{\omega}_i = 0.$. Here we generalize that method for the case that $\vec{\omega}_i \neq 0$. We do so by performing motion compensation in the frequency domain, effectively subtracting off the mean velocity of the image region. This shifts the motion parallax line of Eq. (5) so that it passes through the origin and the assumptions of the optical snow method are then met.

The motion compensation step is performed in the frequency domain, but the basic idea is best explained in the space-time domain. Suppose we were to shear the image region i in xyt by shifting image frame t by (tm_x, tm_y) pixels. This is equivalent to subtracting some vector (m_x, m_y) from each image velocity. If (m_x, m_y) lies on the motion parallax line of Eq. (5), then this image shear shifts the motion parallax line so that it passes through the origin which is what we want.

The problem, therefore, is to find a vector $(m_x, m_y)_i$ for region i that lies on the motion parallax line of that region. We estimate a vector $(m_x, m_y)_i$ such that if the image region were shifted by $(m_x, m_y)_i$ pixels per frame, then the frame-to-frame image intensity variations would be minimized in the sense of minimizing the sum of squared intensity differences

$$\sum_{(x,y,t) \ \in \ \text{region } i} \{ \ I(x - m_x \ t, y - m_y \ t, t+1) - I(x,y,t) \}^2$$

where the sum is over a region i which we now assume is of size $M \times M$ pixels$\times$ T frames. This technique is similar to classical motion compensation techniques in computer vision,[20, 21] except that the sum is performed over T frames rather than two frames.

Rather than performing motion compensation is the *xyt* domain, we perform it in the frequency domain. Using Parseval's theorem, the above minimization can be performed [22, p.61] by finding the vector (m_x, m_y) that minimizes

$$\sum_{(f_x, f_y, f_t)} (m_x f_x + m_y f_y + f_t)^2 \mid \hat{I}(f_x,\ f_y,\ f_t) \mid^2 \tag{8}$$

where the sum is over $f_x, f_y \in \{-M/2, \ldots, M/2 - 1\}$ and $f_t \in \{-T/2, \ldots, T/2 - 1\}$. This is a least squares fit of a plane in (f_x, f_y, f_t), where the error is measured in the f_t direction.

4.C. Signal processing details

Prior to computing the Fourier transform of a local image region, the mean intensity of the region of this region is subtracted. The intensities in the $M \times M \times T$ region are then multiplied by a cone window in (x, y) and a triangular window in t. The spatial window gives more weight to pixels near the center point of the region. This is consistent with the bounds derived in Appendix A in the sense that deviations in the model that result from position differences are more severe as the region is larger. The temporal window is used because the observer's motion parameters $\mathbf{T}, \mathbf{\Omega}$ may vary slowly over the T frames. The central frame is taken as the representative value and is weighted more heavily. (For the experiments we present, the camera motion parameters $\mathbf{T}, \mathbf{\Omega}$ were constant over the T frames.)

Low spatial frequencies are avoided for the motion compensation step, for the same reason there are avoided in the optical snow method,[18] namely the image window and space-time occlusions spread power locally in the frequency domain.[18, 23–25] This spreading of power is problematic at low spatial frequencies since the power of the image is highest there i.e. natural images[26] have a power spectrum that tends to fall off as $1/(f_x^2 + f_y^2)$. At low spatial frequencies, spreading of the power spectrum transfers power between motion planes of very different orientations. This would corrupt the estimates of mean velocity needed for motion compensation. The spreading of power is less significant for higher spatial frequencies since the spreading occurs over motion planes of similar slope.

A third signal processing issue is aliasing. Because the image sequence is sampled in xyt, the Fourier transform of the local image region is a periodic function, $\hat{I}(f_x \bmod M,\ f_y \bmod M,\ f_t \bmod T)$. Any spatial or temporal power in the continuous pre-sampled image that is above the Nyquist frequency ($\frac{M}{2}$ in space, $\frac{T}{2}$ in time) is aliased to frequencies below the Nyquist frequency. Spatial aliasing is typically insignificant for real optical systems because spatial blurring attenuates high spatial frequencies prior to sampling. Temporal aliasing may be significant, however, for digital images if high image speeds are present. Temporal aliasing occurs if there is energy at spatial frequencies (f_x, f_y) and velocities (v_x, v_y) such that

$$\mid v_x f_y + v_y f_y \mid > \frac{T}{2}. \tag{9}$$

The motion plane corresponding to this image velocity wraps around at the Nyquist frequency, $f_t = \pm\frac{T}{2}$, so that high positive speeds appear in the power spectrum as high negative speeds, and vice-versa. Temporal aliasing does occur in our image sequences.

To address temporal aliasing, we iteratively estimate the mean velocity vector (m_x, m_y) using a multi-scale method.[20,22] The least squares minimization is first performed using a roughly one octave band of spatial frequencies $\sqrt{f_x^2 + f_y^2} \in [\frac{M}{16}, \frac{M}{8}]$. (Spatial frequencies below $\frac{M}{16}$ are not used because of windowing and occlusion issues – see above). Because this one octave band consists of relatively low frequencies, it is less likely to meet the temporal aliasing condition of Eq. 9. At iteration n, an n octave band $[\frac{M}{16},\ \frac{M}{16} \cdot 2^n\ ]$ is used. Typically just 2-3 iterations are needed for convergence.

At each iteration, the original power spectrum is sheared in the f_t direction. Let $(\hat{m}_x, \hat{m}_y)$ be the current estimate of the motion compensation vector. Before the first iteration, $(\hat{m}_x, \hat{m}_y) = (0, 0)$. After each iteration, $(\hat{m}_x, \hat{m}_y)$ is updated by adding the compensation vector computed in the least squares

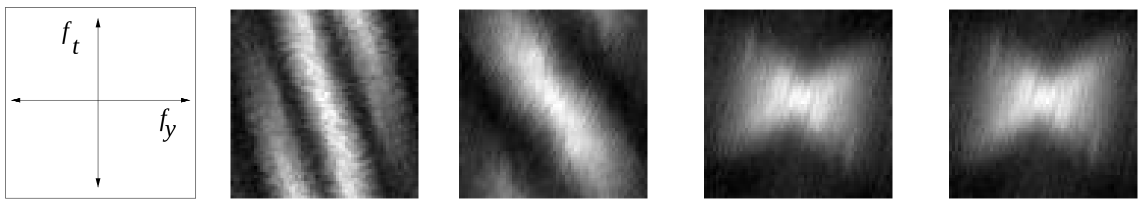


Fig. 4. Example of motion compensation by iterative shearing. Power spectrum is projected to (f_y, f_t) plane. Log-brightness is plotted. For this example, $\vec{\tau}$ is near the f_x direction and so bowtie is visible. Temporal aliasing is evident in the original projected spectrum (left) but disappears after motion compensation (right).

minimization:

$$\hat{I}_{sheared}(f_x,\ f_y,\ f_t)\ :=\ \hat{I}(f_x,\ f_y,\ (f_t - \hat{m}_x f_x - \hat{m}_y f_y) \bmod T\ )$$

where f_t takes values in 0 to $T-1$, so that the mod operation is well-defined. An example of the sheared power spectrum after each iteration is shown in Fig. 4.

5. Experiments: estimation of motion parallax direction

How well can the direction of motion parallax $\vec{\tau}_i$ be estimated? We address this question first using image sequences that are rendered with computer graphics (OpenGL) so that results can be compared to precise ground truth. We also consider several natural image sequences.

5.A. Synthetic sequences

We use three sets of synthetic scenes (see top row of Fig. 5), which vary in information content and in degree of difficulty.[1] The first consists of squares distributed randomly in a 3D view volume. Each square is texture mapped

[1]Examples of our image sequences are available online at: www.cs.uwaterloo.ca/˜mannr/parallax.

with a low pass noise pattern which simulates texture and shading on real surfaces. These scenes are intended to model natural cluttered scenes. We have also carried out experiments using checkerboard texture maps and similar results were obtained.[15]

The second set of scenes consists of a semi-transparent planar surface in front of an opaque planar surface. An additive transparency model was used with constant opacity α. Both surfaces were texture mapped with a noise pattern. We predicted our method would perform better in these scenes since there is abundant texture and depth differences in local image regions, but no occlusions and hence relatively little spreading of power (see Sec. 4.C).

The third set of scenes consists of long, thin cylinders, distributed randomly within a view volume in the squares case. The cylinders are shaded but not texture mapped and so intensity varies only in one dimension for each cylinder, namely perpendicular to its axis. This implies that only normal velocities can be measured for each cylinder, that is, the aperture problem occurs.[27,28] As such pointwise image velocities cannot be measured directly from intensities from these sequences and so egomotion methods that rae based on pointwise image velocities cannot be applied. Clearly, this set of scenes is especially challenging.

We next provide more details on how we rendered these three types of synthetic scenes. For the squares scenes, the squares are all the same size, namely a width of 0.5 units, and have random orientation and position. Squares are distributed randomly in a cubic view volume whose sides are 40 units. The camera had a near and far clipping planes of $Z_{min} = 5$ and $Z_{max} = 50$, where the image projection plane is $Z = 1$ and, at time 0, the cube contained the view volume is between $Z = 0$ and $Z = 40$. Ambient lighting was used.

For the transparent layers scenes, the two layers are at depths 10 and 20, respectively. Both planes were texture mapped with $1/f$ noise. Again ambient lighting was used. The opacity α of the front plane was constant in each sequence, and was varied uniformly from 0.2 to 0.8 over the twenty sequences used. The compositing model is equivalent to

$$I(x, y, t) = \alpha I_{fore}(x, y, t) + (1 - \alpha) I_{back}(x, y, t)$$

where $I_{fore}(x, y, t)$ and $I_{back}(x, y, t)$ are the image sequences are due to the foreground and background layers on their own.

For the cylinder scenes, each cylinder has radius 0.1 and the same view volume as the squares. The cylinder scenes have Lambertian surface reflectance and are illuminated by a single collimated light source parallel to the camera's optical axis (Z). We chose this lighting condition because it avoids sharp visible shadows which would provide point features to track. Instead, our lighting presents a aperture problem similar to what one finds under true diffuse lighting. We are choosing a scenario in which motion measurement is intentionally difficult for traditional methods, in order to illustrate an advantage of our method.

For each of the scenes types, we considered three camera motions:

(i) forward translation and no rotation:
$\mathbf{T} = (0, 0, -0.25), \quad \mathbf{\Omega} = (0, 0, 0).$

(ii) forward translation and pan:
$\mathbf{T} = (0, 0, -0.05), \quad \mathbf{\Omega} = (0, 0.234, 0).$

(iii) lateral motion and roll:
$\mathbf{T} = (-0.05, 0, 0), \quad \mathbf{\Omega} = (0,\ 0,\ 1.25).$

The translations are stated in world units per frame, the rotations are in units of degrees, and the optical axis points in the negative Z direction (standard in OpenGL).

For each camera motion and scene type, twenty images sequences were rendered. Each sequence is $T = 32$ frames and $N \times N = 256 \times 256$ pixels. This pixel grid spans a 30×30 degree field of view.

To render the sequences with forward motion, we performed a preprocessing step to remove any object that is in front of or straddling the near clipping plane in any frame of the sequence. This "carves out" a space directly in front of the camera's path, while leaving objects on the periphery. (See Fig. 5(top).) We rendered scenes in this manner to remove any artifacts of clipping. This includes objects that are partially drawn, and also objects that vanish as the camera moves forward. We note that removing objects in this way makes the egomotion problem *more* difficult, since it reduces the amount of parallax. Nonetheless, as we will see, enough depth structure remains to compute the directions of motion parallax and egomotion.

Finally, to mimic optical blur that arises from real cameras and thereby reduce spatial aliasing, each image frame was rendered at twice ($N = 512$) the desired spatial resolution. Each frame was then blurred with a Gaussian kernel ($\sigma = 1$), and the blurred frame was sub-sampled by a factor of two to yield the desired spatial resolution. No temporal blur was computed, however, since real cameras do suffer from temporal aliasing.

5.B. Synthetic sequences: Results

For all sequences, the direction of motion parallax $\vec{\tau}_i$ was estimated in a 7×7 grid of overlapping image regions, each of size $M \times M = 64 \times 64$, which is a width of approximately 8 degrees. The regions were overlapped because of the triangular window (recall Sec. 4.B).

Figure 5 shows examples of results for pure forward motion. The top row shows the first frame of three sequences. The middle row shows the motion parallax lines, estimated in each image region and for that sequence. The bottom row shows the power spectrum, summed and projected along the estimated bowtie axis for each region after motion compensation. Like the results in Ref. [18], the "bowtie" signature is clearly visible in each region. Note that for the transparent scene, two distinct motion planes are visible in the bowties, corresponding to two depth layers.

The amount of parallax in each region is indicated by the range of slopes in the projected bowtie. Regions at the image edge have largest range of slopes,

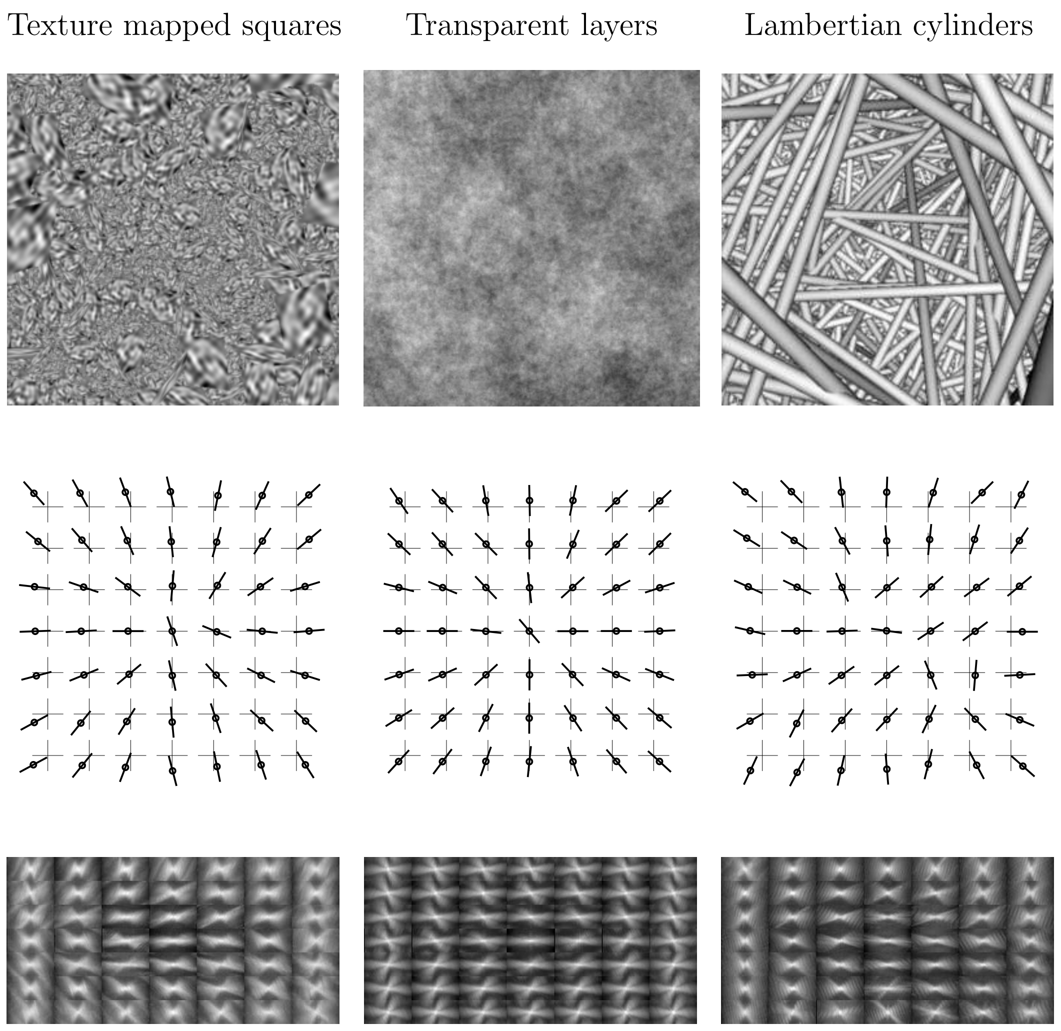


Fig. 5. Example of results for forward motion only. (top row) First frame of sequence. (middle row) Estimated motion parallax lines. The circle shows the computed motion compensation vector (m_x, m_y). (bottom row) Projected bowties for each image region after motion compensation.

while regions near the center (the AOT) have smallest range of slopes. We expect better estimates of $\vec{\tau}_i$ when a higher range of slopes is present.

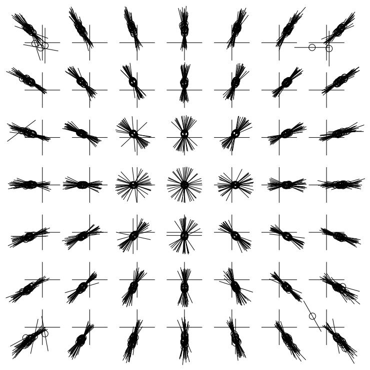
Figures 6, 7, and 8 show the pooled results for the squares, transparent, and cylinders scenes, respectively, and for each type of camera motion. The left column of each figure presents the estimated motion parallax lines. One line segment is shown for each of the regions and for each of the 20 sequences. Each line segment is centered at the estimated (m_x, m_y) vector and is oriented in the estimated $\vec{\tau}_i$ direction.

The right column of each figure presents the mean absolute angular error for the data in the left column, that is, the difference between the estimated and true $\vec{\tau}_i$. A mean absolute angular error of 45 degrees indicates chance performance. The reason is that, in our experiments, we defined $\vec{\tau}$ up to a 180 degree ambiguity (pointing away or toward the AOT). The absolute angular error per trial is bounded between 0 and 90 degrees, and so a uniform error in this interval would have mean of 45 degrees.

Although the general pattern of errors is similar for each scene type, the magnitudes of the errors are smallest for transparent scenes and largest for cylinders. This is what we expected. The small errors for the transparent scenes are due to lack of occlusions. The large errors for the cylinders are due to the lack of texture, i.e. only normal velocities. Note that even though the cylinder errors are large, they are well below the chance level of 45 degrees. As we will see in Sec. 6, they are sufficiently low to estimate egomotion.

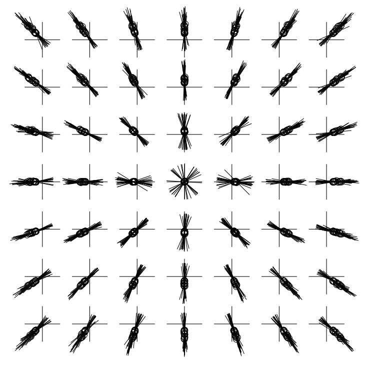
Let us next examine how the results vary with camera motion. For the sake of brevity, we concentrate our discussion on the squares scenes (the top row of Figures 6–8).

Figure 6 shows the results for forward translation and no rotation. The AOT is in the center of the image and the true $\vec{\tau}_i$ vectors are oriented radially from the AOT. Thus the motion parallax line for each region passes through the origin. The only exception is for regions containing or very near to the AOT where the motion parallax line model does not hold. In particular, for the



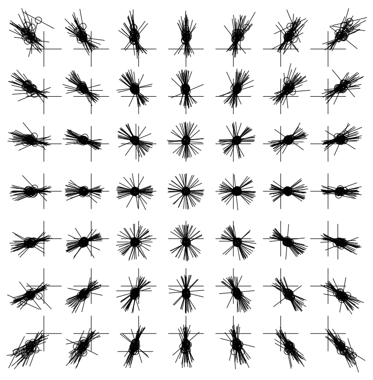
12	7	5	5	7	5	9
5	7	7	8	8	6	4
7	7	17	21	12	8	7
7	9	28	*	25	11	7
8	8	13	25	9	6	5
5	7	8	9	11	5	8
12	7	6	7	6	5	7

(a) Texture mapped squares.



5	4	4	3	4	4	5
4	3	4	2	3	3	4
6	3	3	8	4	4	5
5	4	10	*	11	3	3
3	4	5	6	4	3	4
5	3	5	3	4	3	4
6	5	5	3	3	4	4

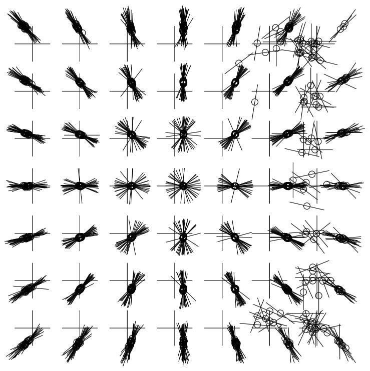
(b) Transparent layers.



18	16	17	10	16	19	21
13	16	20	16	19	17	15
13	10	25	30	30	15	14
11	12	22	*	27	19	8
13	16	35	29	27	16	13
19	16	18	19	18	16	15
15	15	17	12	14	9	9

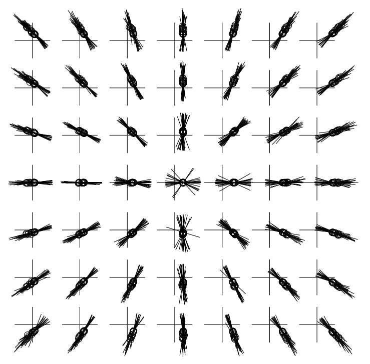
(c) Lambertian cylinders.

Fig. 6. Estimated directions of motion parallax for synthetic sequences under motion (i), forward translation. (left) Estimated motion parallax lines. (right) Average mean absolute angular errors of $\vec{\tau}_i$ (in degrees) over twenty runs.



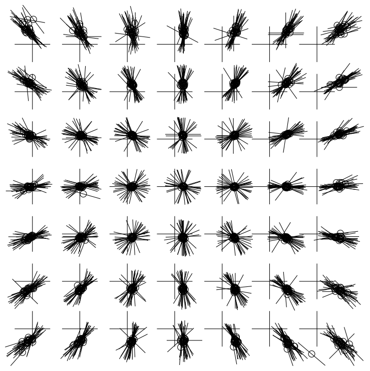
6	4	9	9	9	18	44
6	8	12	7	8	7	31
7	8	19	30	14	10	22
6	14	26	*	20	7	15
6	10	17	26	16	9	14
7	7	11	10	8	7	21
7	8	8	9	5	22	36

(a) Texture mapped squares.



4	6	4	3	4	4	5
4	3	4	3	4	6	5
4	3	3	8	7	7	6
3	1	7	*	7	6	6
3	4	6	14	7	5	4
4	3	4	6	4	5	6
6	3	4	4	4	4	6

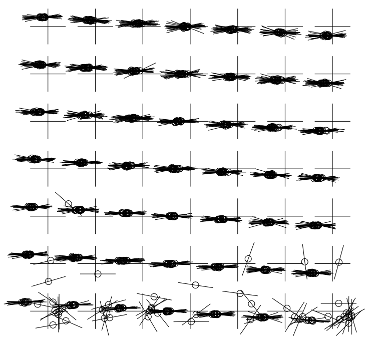
(b) Transparent layers.



9	13	18	11	14	15	15
12	23	13	13	15	18	9
15	26	27	28	31	14	8
13	15	36	*	27	10	8
11	24	29	27	31	18	11
14	12	23	16	18	17	16
10	12	15	17	11	12	15

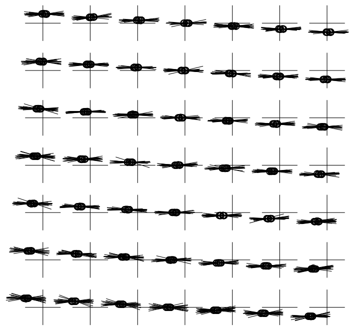
(c) Lambertian cylinders.

Fig. 7. Estimated directions of motion parallax for synthetic sequences under motion (ii), forward translation plus pan. (left) Estimated motion parallax lines. (right) Average mean absolute angular errors of $\vec{\tau}_i$ (in degrees) over twenty runs.



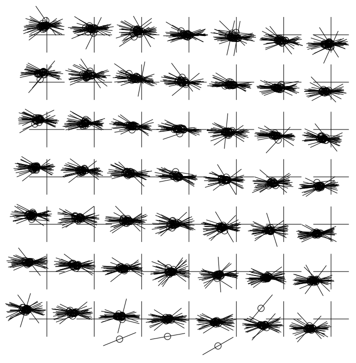
4	5	6	6	6	5	4
4	5	6	7	4	6	5
4	6	5	3	4	4	4
3	2	4	4	4	3	4
4	3	2	3	4	5	4
4	4	3	3	6	8	8
20	12	13	5	9	13	31

(a) Texture mapped squares.



9	4	5	5	6	7	5
9	5	6	4	7	5	6
10	6	4	5	5	4	5
9	6	6	4	3	2	4
9	9	7	4	6	9	7
10	8	10	3	7	6	5
8	12	9	7	6	10	8

(b) Transparent layers.



23	13	21	12	20	15	12
20	14	16	15	15	20	22
16	19	20	14	14	20	19
22	16	12	18	16	26	22
17	15	17	19	17	18	28
30	18	21	17	30	22	22
23	24	27	15	22	21	22

(c) Lambertian cylinders.

Fig. 8. Estimated directions of motion parallax for synthetic sequences under motion (iii), lateral translation plus roll. (left) Estimated motion parallax lines. (right) Average mean absolute angular errors of $\vec{\tau}_i$ (in degrees) over twenty runs.

central region of the image, a motion parallax line is not even defined. The angular error for this region is thus marked with an asterisk.

Figure 7 shows the data when the observer motion is forward translation and a pan to the right. The AOT is again at the center of the image but now the mean velocity vector is non-zero in the central region. Instead the mean velocity in the central region is equal to the pan component. As in Fig. 6, $\vec{\tau}_i$ is not well defined in the central region and so the average angular error is indicated by an asterisk.

Away from the AOT, the error behavior in Fig. 7a differs from Fig. 6a. For the regions on the left edge of 7a, errors in are relatively small whereas for the right edge the errors are relatively large. How can we account for these errors? At the left edge, the translation components (specifically τ_x) of the image velocities are to the left whereas the pan component is to the right. This yields a velocity cancellation on the left edge such that the average velocities are near zero. On the right edge of the image though, both the translation and the pan components are rightward and thus their sum yields large rightward velocities. Thus, the regions on the left vs. right edges of Fig. 7 have a similar absolute range of velocities but, because of the additive pan component, they have a different mean velocity. We next argue that this difference in mean velocity affects the errors.

For a given absolute range of velocities, a low mean velocity tends to yield better estimates of the motion parallax line than a higher mean velocity. To understand why, consider two sets of velocities corresponding to two sets of motion planes in the frequency domain. Suppose arbitrarily that in one set the slopes range from 12 to 14 (mean of 13) and in the other set the slopes range from -1 to 1 (mean of 0). In both cases, the absolute range of slopes is 2. However, in the former case, the motion planes are covered by far fewer samples in the 3D frequency domain than the latter case. The result is an undersampling of the bowtie in the former case. This undersampling cannot be undone by motion compensation and it leads to a relatively poor estimate of the bowtie axis.

It follows that the larger errors on the right edge of Figure 7a relative to the left edge are as expected. Moreover, as one moves away from the AOT to the right, errors initially fall because motion parallax increases thus providing information for estimating $\vec{\tau}_i$. Near the right edge of the image, however, errors rise again because the mean velocity error becomes high. Near the left edge, errors fall where the mean velocity is near zero. (Note that similar rising errors away from the AOT are seen at the corners in 6a.)

Similar errors in estimating $\vec{\tau}_i$ are found in Figure 8. The observer is translating laterally to the left and rotating in a counter-clockwise roll. This yields a true $\vec{\tau}_i$ that is horizontal for all image regions, along with a clockwise rotation field. Errors in the $\vec{\tau}_i$ estimates increase noticeably at the bottom edge of the image, where the rotation component of the velocity field is large and parallel to the translation component, yielding a large mean velocity component.

5.C. Real sequences

Fig. 9 shows direction of motion parallax estimates for two real image sequences. The format is similar to Fig. 5. The first frame of the sequence is shown (top) along with estimated motion parallax lines (middle) and projected bowties (bottom).

The left sequence ("bush") has horizontal (lateral) motion. The image sequence was obtained using a Hitachi MPEG MP-EG10W camera moving horizontally on a sliding platform. Images were captured at 30 frames per second, at a resolution of 320×240 pixels. Images were cropped to 240×240, and 32 frames, or approximately one second of video, were used. This data is a subset of the sequence used in Ref. [18].

The right sequence ("robot") of Fig. 9 was generated from a consumer video camera (Canon Optura Pi) mounted on a wheeled robot (AmigoBot from ActivMedia Robotics). The robot drove on a circular path, curving to

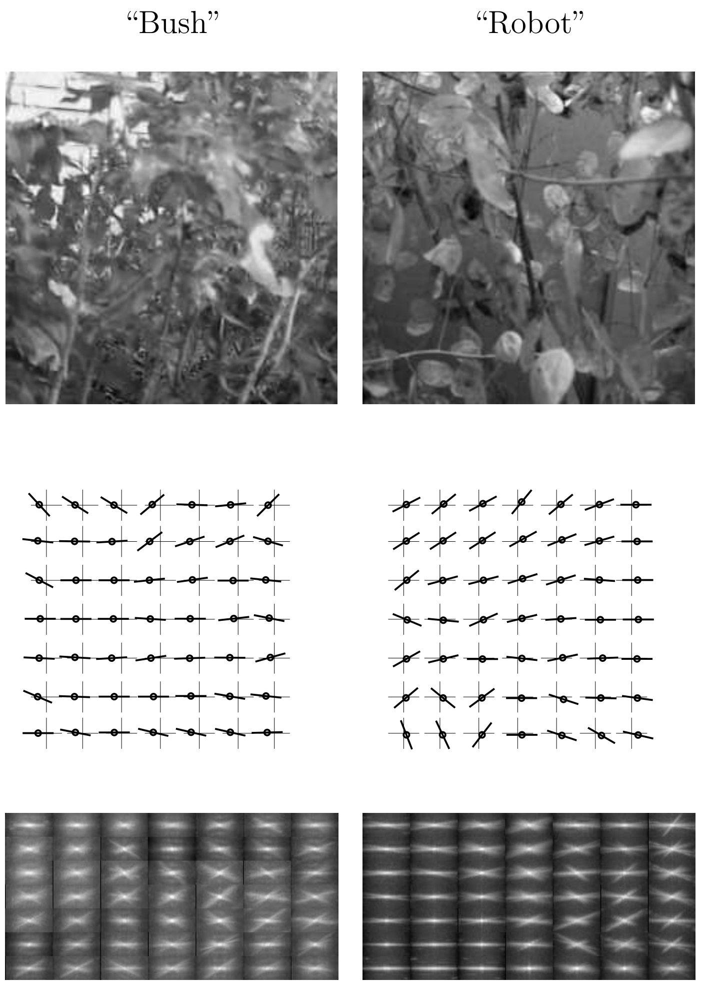


Fig. 9. Estimation of motion parallax direction for real sequences. (left) "Bush" sequence. Camera is translating laterally to left. (right) "Robot" sequence. Camera is translating forward to the left and panning to the left.

the left. The camera was pointing about 20° to the right of $\mathbf{T}$. Thus, the AOT was about 20° to the left of the image center. The standard 4:3, 640×480 image frame was cropped to a square, blurred, and subsampled by two to give a 240×240 image. The camera field of view of the cropped image was approximately 30 degrees. Hence, the AOT is just outside the cropped image. The camera captures 30 frames per second, non-interlaced. Again, 32 frames, or approximately one second of video, were used.

Numerical data are not given since the camera motion was not fully calibrated. Note that bowties are visible in many but not all regions. In particular, in the sequence on the right, only a small number of surfaces are visible in certain regions. In addition errors for that sequence are higher near the left edge we get closer to the AOT.

6. Estimation of egomotion

6.A. Method

Once the directions of motion parallax $\vec{\tau}_i$ have been estimated for each region i, the egomotion vectors $\mathbf{T}$ and $\mathbf{\Omega}$ can be estimated using standard methods.

Our method is based on the Heeger and Jepson's subspace method.[29] Each $\vec{\tau}_i$ estimate defines a 2D plane (in XYZ space) on which the true $\mathbf{T}$ vector must lie. We find the unit vector that minimizes the sum of squared distances to all these $\vec{\tau}_i$ planes and take this to be our estimate of $\mathbf{T}$. Once we have the translation $\mathbf{T}$, we find the rotation $\mathbf{\Omega}$ by solving a second least squares problem. The details are given in Appendix B. We also present results that were based on a robust estimation method.[30,31] Robust estimation can be viewed as a weighted version of least squares, where the weighting is data-dependent. Estimates that closely fit the model are assigned greater weights and contribute more to the estimate. Details of this method are given in Appendix C.

Tables 1–3 show the egomotion computation results for the three motion cases described in Section 5. For each case we report both the mean of the estimates and the average errors for **T** and **Ω** over the twenty sequences. The **T** error is reported as the average of the angle between the estimated unit vector **T** and the true unit vector **T**. The **Ω** error is reported in two ways: as the standard deviation of each of the $\Omega_x, \Omega_y, \Omega_z$ components, and as the average angular error between the estimated **Ω** and true **Ω**. The latter error is undefined in Table 1 since there is no rotation.

		$(T_x,$	$T_y,$	$T_z)$	$(\Omega_x,$	$\Omega_y,$	$\Omega_z)$
Scene		(0,	0,	-1)	(0,	0,	0)
Squares	mean	(-0.002,	0.000,	-1.000)	(0.003,	0.000,	-0.004)
	std.dev.				(0.011,	0.011,	0.019)
	average err.		0.6 degrees			n/a	
Layers	mean	(-0.001,	0.001,	-1.000)	(-0.002,	-0.000,	-0.001)
	std.dev.				(0.004,	0.004,	0.010)
	average err.		0.3 degrees			n/a	
Cylinders	mean	(0.003,	0.007,	-1.000)	(-0.008,	0.003,	-0.011)
	std.dev.				(0.032,	0.045,	0.055)
	average err.		1.7 degrees			n/a	

Table 1. Egomotion computation results for case (i), forward motion.

For Table 1 (forward motion), both translation and rotation are accurately determined for all three scene types. The translation direction is accurate to about one degree and the estimated rotation vector is near the true value of zero.

For Table 2 (forward motion + pan), the translation estimates are again accurate to roughly one degree for all three scene types, but the rotation error rises to about 15 degrees for the squares and cylinders scenes.

		$(T_x,$	$T_y,$	$T_z)$	$(\Omega_x,$	$\Omega_y,$	$\Omega_z)$
Scene		(0,	0,	-1)	(0,	0.234,	0)
Squares	mean	(0.012,	0.004,	-1.000)	(-0.003,	0.240,	0.008)
	std.dev.				(0.035,	0.018,	0.068)
	average err.		1.4 degrees			15.7 degrees	
Layers	mean	(-0.002,	-0.000,	-1.000)	(0.000,	0.224,	0.001)
	std.dev.				(0.006,	0.005,	0.003)
	average err.		0.4 degrees			1.4 degrees	
Cylinders	mean	(0.003,	0.007,	-1.000)	(-0.002,	0.210,	-0.003)
	std. dev.				(0.034,	0.042,	0.048)
	average err.		1.8 degrees			13.7 degrees	

Table 2. Egomotion computation results for case (ii), forward motion plus pan.

For Table 3 (lateral translation + roll), the errors remain low for the transparent layers scene, but the errors are now large for the squares and cylinder cases. Using the robust method, the errors were reduced for the squares scenes, but remain high for the cylinder scenes. Here the method comes up against a well-known limitation of egomotion estimation. For a narrow field of view, a pan+tilt rotation can produce a similar image velocity fields as a lateral translation. This problem known as the rotation-translation ambiguity.[32] In the case of the motion in Table 3, there is an ambiguity between translation along the X axis and rotation about the Y axis. This explains the relatively large bias in the estimate of Ω_y for the squares and cylinders. Presumably this bias is less significant in the transparent layers case because the estimates of $\vec{\tau}_i$ are accurate enough to avoid the ambiguity.

To investigate the translation-rotation ambiguity further, we consider the performance of the egomotion method for squares sequences generated with varying degrees of forward and lateral translation. The translation was given by $\mathbf{T} = (0, \sin\theta, \cos\theta)$, which varies from the Z direction ($\theta = 0$) to the

		$(T_x,\ T_y,\ T_z)$	$(\Omega_x,\ \Omega_y,\ \Omega_z)$
Scene		(-1, 0, 0)	(0, 0, 1.25)
Squares	mean	(-0.993, -0.011, 0.122)	(0.004, -0.472, 1.166)
	std.dev.		(0.017, 0.539, 0.078)
	average err.	11.8 degrees	19.9 degrees
	(robust)	(6.2 degrees)	(6.5 degrees)
Layers	mean	(-1.000, 0.002, -0.018)	(0.000, -0.037, 1.211)
	std.dev.		(0.002, 0.056, 0.006)
	average err.	4.2 degrees	2.6 degrees
	(robust)	(1.9 degrees)	(2.2 degrees)
Cylinders	mean	(-0.988, 0.015, 0.156)	(-0.005, -0.331, 1.237)
	std.dev.		(0.024, 0.501, 0.071)
	average err.	26.0 degrees	14.4 degrees
	(robust)	(23.8 degrees)	(11.3 degrees)

Table 3. Egomotion computation results for case (iii), lateral motion plus roll.

Y direction (θ =90 degrees). The rotation was fixed at $\mathbf{\Omega} = (0, 0.2344, 0)$ which was the rotation vector used in Table 2. Figure 10 shows the average (absolute) angular error of $\mathbf{T}$ and $\mathbf{\Omega}$ as θ varies. As expected, the average angular error increases towards lateral motion ($\theta = 90$). This pattern of errors is thus consistent with Rieger and Lawton's method, as implemented by Heeger and Jepson in Ref. [29, Fig. 6a].

Finally Table 4 shows egomotion results for the real sequences shown in Figure 9. For the 'bush" sequene, the camera motion is roughly lateral and horizontal. For the "robot" sequence, the camera motion is roughly forward and to the left. For both of these real sequences, the "ground truth" of $\mathbf{T}$ was not determined using precise instrumentation, and is accurate to no better than a few degrees. Moreover, for the robot sequence, only the direction of the the angular velocity $\mathbf{\Omega}$ but not its magnitude was known. Given this margin of error, the estimates are acceptable and consistent with the synthetic scene

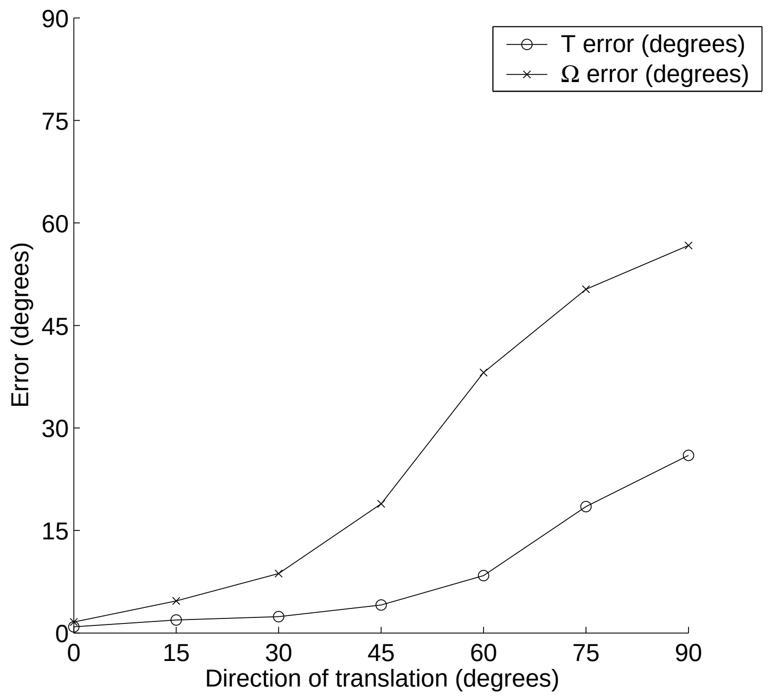


Fig. 10. Egomotion computation results ($\mathbf{T}$, $\mathbf{\Omega}$) for a variant of case (ii), forward motion + pan. The translation direction moves from forward ($\theta = 0$) to upward ($\theta =$90 degrees). $\|\mathbf{T}\| = 0.05$, $\mathbf{\Omega} = (0, 0.2344, 0)$. Note that the translation speed is reduced from that in Fig. 6. The data is for squares scenes, and least squares estimation was used. Errors were averaged over ten runs.

results presented above.

Scene		$(T_x,$ $T_y,$ $T_z)$	$(\Omega_x,$ $\Omega_y,$ $\Omega_z)$
Bush	approx. ground truth	(-1, 0, 0)	(0, 0, 0)
	computed value	(-0.796, 0.020, -0.605)	(-0.012, -0.188, -0.028)
	approx. angular err.	37 degrees	n/a
(robust)	computed value	(-0.998, 0.002, -0.063)	(-0.012, -0.091, -0.033)
	angular err.	4 degrees	n/a
Robot	approx. ground truth	(-0.383, 0, -0.924)	(0, 1, 0)
	computed value	(0.392, 0.048, 0.919)	(0.019, 0.044, -0.023)
	approx. angular err.	3 degrees	34 degrees
(robust)	computed value	(0.370, 0.020, 0.929)	(0.017, 0.042, -0.008)
	angular err.	1 degrees	24 degrees

Table 4. Egomotion results for real sequences in Fig. 9. Angular errors are approximate only, since camera motion was not accurately calibrated.

7. Conclusion

The key contribution of this paper is to show how the direction of motion parallax can be estimated in local image regions, prior to estimating pointwise image intensities, and that these motion parallax directions can be combined to estimate egomotion ($\mathbf{T}$ and $\mathbf{\Omega}$). The advantage of ordering the computations in this way is that the estimated egomotion provides strong constraints on the pointwise image velocity field, reducing the search for image velocities to a 1D set rather than 2D set. We argue that this simplification of "optical flow" computation is very important 3D cluttered scenes, where the high density of depth discontinuities makes it difficult to estimate a dense image velocity field. Such a dense image velocity field is needed for the moving observer to estimate a dense depth map.

Our method for estimating the direction of local motion parallax is based on a particular model that assumes the spread of (unknown) image velocities in a local region is dominated by the translation component of the velocity field, rather than by the rotation component. As we showed in Appendix A, this assumption in turn requires that there is a large range of depths in each local region. If this assumption does not hold, then the model does not hold, and the method cannot be expected to give reliable estimates. In particular, if the scene consists of a smooth ground plane, then we would not expect our method to be effective. This is not a problem, of course, since there are other egomotion methods which are explicitly designed for planar surfaces[?,33] and perform well in such scenes. Moreover, when surfaces are smooth and textured, many traditional egomotion methods which only assume pre-computed optical flow do fine.[34] The case to be made for our method is that it is designed for 3D cluttered scenes, where a dense field of image velocities is difficult to compute. Our method provides a way to deal with the complexity of 3D cluttered scenes, both for computing egomotion and for subsequently computing image velocities and thereby depth.

A. Bounds on motion parallax line model

Suppose the image region is a disk of radius R_i in the image plane ($Z = f$). From the arguments of Longuet-Higgins and Pradzny,[16] the model of Eq. (5) clearly holds in the limit as $R_i \rightarrow 0$. A pointwise analysis of the error of the model for $R_i > 0$ was given by Rieger and Lawton.[17] Here we extend that analysis and present bounds on the model as a function of the radius R_i and other relevant variables

For any variable V defined in region i, let $V(x, y)$ and V_i be the values of the variable at (x, y) and (x_i, y_i), respectively. We define the *deviation* ΔV of the variable V in region i to be the largest difference between V and V_i in the region:

$$\Delta V \;\equiv\; max\{ \;\| V(x,y) - V_i \;\|_2 \;\;:\;\; \| \;(x,y) - (x_i, y_i) \;\|_2 \;\; < R_i \;\}$$

A.1. Deviation in rotation component

We first consider the deviation in the components of $\vec{v}_\Omega$. From Eq. (1), the roll component of $\vec{v}_\Omega$ is

$$\vec{v}_{roll} \;=\; (-y, x)\;\Omega_z$$

where Ω_z is in radians per frame. Consider a step (δ_x, δ_y) away from (x_i, y_i). The resulting difference in $\vec{v}_{roll}$ is $(-\delta_y, \delta_x)\Omega_z$. The length of this difference vector is $\Omega_z \delta$ where $\delta = \sqrt{\delta_x^2 + \delta_y^2}$ is the step size. Since the largest step size is R_i, the deviation of $\vec{v}_{roll}$ is $R_i\Omega_z$.

The pan component of $\vec{v}_\Omega$ is

$$\vec{v}_{pan} = (-f - x^2/f, -xy/f)\;\Omega_y\;.$$

A small step of length δ in the x direction produces a vector difference $(-2x_i/f, -y_i/f)\;\delta$, and a small step of length δ in the y direction produces a vector difference of $(0, -x_i/f)\;\delta$. We get a bound on the deviation of the pan component by noting that $\| \;(x_i, y_i) \;\|_2 \;/f$ is the tangent of the visual angle from $(0, 0)$ to (x_i, y_i) and that this value is well below 1 for a typical video camera (30 degree field of view). Specifically, $\Delta\vec{v}_{pan}$ goes from 0 for a region at the image center up to about $R_i\Omega_y$ for a region at the image corners.

Because of symmetry between pan and tilt, we immediately get that $\Delta\vec{v}_{tilt} = R_i\Omega_x$. Combining the bounds gives

$$\Delta\vec{v}_{\Omega} \ < \ R_i \ \parallel \mathbf{\Omega} \parallel_2 .$$

Note that this is conservative. The pan and tilt components of this bound vanish to zero at the image center.

A.2. Deviation in direction of translation component

We next consider the deviation of $\vec{\tau}$. In the case of lateral motion ($T_z = 0$), $\vec{\tau}$ is constant across the entire image, and so $\Delta\vec{\tau}_i = 0$. For non-lateral motion, $\vec{\tau}$ is constant along any line through (x_T, y_T) in the image. Thus $\vec{\tau}$ varies only in the direction perpendicular to such a line. Because of radial symmetry of the $\vec{\tau}$ field about AOT, without loss of generality we can calculate $\Delta\vec{\tau}$ by letting $y_i = y_T$ and taking a step $(0, R_i)$ from (x_i, y_T) to $(x_i, y_T + R_i)$. This step produces a change of $\vec{\tau}$ by an angle $\arctan(R_i/(x_i - x_T))$ radians. Invoking the radial symmetry of the $\vec{\tau}$ field about AOT, and defining d_i to be the distance from AOT to (x_i, y_i), that is,

$$d_i \equiv \parallel (x_i, y_i) - (x_T, y_T) \parallel$$

and noting that $|\arctan(\theta)| < |\theta|$, we get

$$\Delta\vec{\tau} = R_i/d_i \quad \text{radians.}$$

A.3. Deviation in translation component

Two bounds can be derived depending on whether the motion is lateral or not. Both of these bounds depend on the deviation in inverse depths, which we denote $\Delta\frac{1}{Z}$.

For lateral motion, Eq. (4) immediately gives:

$$\Delta\vec{v}_T \ = \ f \ \parallel \mathbf{T} \parallel_2 \ \Delta\frac{1}{Z} .$$

For non-lateral motion, we expand $\vec{v}_T - \vec{v}_{T,i}$ using Eq. (2) and then add and subtract $\frac{T_z}{Z(x,y)}(x_i, y_i)$. Grouping terms gives:

$$\vec{v}_T - \vec{v}_{T,i} = \frac{T_z}{Z(x,y)}(x - x_i, y - y_i) + (\frac{T_z}{Z(x,y)} - \frac{T_z}{Z(x_i,y_i)})(x_i - x_T, y_i - y_T)$$

Applying the triangle inequality then gives:

$$\Delta\vec{v}_T \ > \ T_z \ d_i \ \Delta\frac{1}{Z} \ .$$

A.4. Discussion

We emphasize that the calculation of the bounds was conservative in that the bounds are based on worst case scenarios. In several cases, we expect the motion parallax model of Eq. 5 to provide a much better fit than the bounds would imply. In particular, only the differences in the rotation vector that are perpendicular to the motion parallax line cause deviation from the model, and this deviation is negligible under many natural conditions. For example, if there is no camera roll component ($\Omega_z = 0$) then $\Delta\vec{v}_\Omega$ decreases linearly to zero at the AOT and so the model holds better than the bounds predict. Another example is if a laterally moving camera rotates so as to track a visible point in the scene.[35,36] The rotation vectors may be large but nonetheless are near parallel to $\vec{\tau}_i$, and so deviation in the rotation vectors is confined to the $\vec{\tau}_i$ direction which supports rather than contradicts the model.

A.5. Example: when the motion parallax model holds

For the model of Eq. (5) to hold in region R_i and for there to be motion parallax present, all velocity vectors $\vec{v}$ must lie near the line of Eq. (5) and there must a spread of speeds along this line. For these two conditions to be satisfied, it is sufficient that both $\Delta\vec{\tau}$ is small and that $\Delta\vec{v}_\Omega \ll \Delta\vec{v}_T$.

To illustrate these bounds, we use the parameters from our experiments of Sec. 5.A. Each local image region has radius of about 4 degrees, so $R_i \approx \frac{1}{15}$ radians. Camera translation direction is either pure forward or pure lateral. For forward translation, $d_i = 0$ at the image center and rises to $d_i \approx \frac{1}{3}$ at the four image corners. Thus, at the four image corners, $\Delta\vec{\tau} \approx \frac{1}{5}$ radians or 12 degrees. The deviation depends on d_i^{-1}, e.g. it is doubled for regions that are half the distance to the AOT. Nonetheless, the majority of regions have $\Delta\vec{\tau}$ of less than 24 degrees. Considering this is a worst case measure, such a value supports the model.

For the second condition ($\Delta\vec{v}_\Omega \ll \Delta\vec{v}_T$), we must choose a value for $\Delta\frac{1}{Z}$. Here we must approximate since, for the forward translation sequences, the surface corresponding to (x_i, y_i) may vary between frames as may the set of depths of visible surfaces in the region. For frame 1 of the cylinder and textured square sequences, $Z_{min} = 5, \quad Z_{max} = 40$, and the net forward motion over the sequence is 8 units ($f = 1$). The maximum value of $\Delta\frac{1}{Z}$ is $(\frac{1}{5} - \frac{1}{40}) = 0.2$ (at the beginning of the sequence). Using these numbers we take a value 0.1 as typical for $\Delta\frac{1}{Z}$, keeping in mind below that the value would be lower if there were an insufficient range of depths in the region.

We now address two of the observer motions:

- forward motion + pan:

  $T_z = .25, \qquad \| \mathbf{\Omega} \|_2 = .234$ deg/frame $\approx \frac{1}{250}$ radians/frame.

  At the image corners,

  $$\Delta\vec{v}_\Omega = \Delta\vec{v}_{pan} = \frac{1}{250} * \frac{1}{15} \approx .0003$$

  $$\Delta\vec{v}_T > .25 * .1 * \frac{1}{3} \approx .001$$

  so that $\Delta\vec{v}_T \gg \Delta\vec{v}_\Omega$ provided there is reasonable depth variation in a region. Moreover, these bounds are both linear functions of d_i, so the model still holds well for regions that are half the distance from the corner to the image center. The difficulties with the model thus arise near the AOT since $\Delta\vec{\tau}$ can become large there, or when there is a limited range of depths in the region.

- lateral motion + roll:

$\| \mathbf{T} \|_2 = .05, \qquad \| \mathbf{\Omega} \|_2 = 1.25 * \frac{2\pi}{360} \approx .02$ radians. Thus,

$$\Delta \vec{v}_\Omega \approx \frac{1}{15} * .02 \approx .001$$
$$\Delta \vec{v}_T \; = \; .05 * .1 \approx .005 \; .$$

The latter again a reasonable range of depths ($f\Delta\frac{1}{Z} = 0.1$).

B. Least squares egomotion computation

Here we provide a brief description of the egomotion computation. Our method follows the notation and least squares formulation of Heeger and Jepson.[29]

For any image region i, let $\mathbf{p}_i = [x_i, y_i, f]$ be the center of the region and let $\mathbf{t}_i = [\tau_{x,i}, \tau_{y,i}, 0]$ be a 3D vector corresponding to $\vec{\tau}_i = (\tau_{x,i}, \tau_{y,i})$. Since the true $\vec{\tau}_i$ points toward the AOT, it follows that the translation vector $\mathbf{T}$ must lie in the plane spanned by $\mathbf{p}_i$ and $\mathbf{t}_i$. Thus, each region supplies a single constraint on the camera translation $\mathbf{T}$, namely:

$$(\mathbf{t}_i \times \mathbf{p}_i) \cdot \mathbf{T} = 0 \; .$$

Since the estimates of $\vec{\tau}_i$ have non-zero error and since there is a unresolvable scale ambiguity in $\mathbf{T}$, we compute a (unit) vector $\mathbf{T}$ that minimizes:

$$\text{argmin}_{\,\mathbf{T}} \sum_i \rho\Big(\, unit\left(\hat{\mathbf{t}}_i \times \mathbf{p}_i\right) \cdot \mathbf{T}\Big) \tag{10}$$

where $unit()$ takes the unit vector and $\hat{\mathbf{t}}_i$ is the estimated value of $\mathbf{t}_i$. $\rho(.)$ is an error measure, and for least squares, $\rho(e_i) = e_i^2$. Robust error measures are described in Appendix C. The solution to the above may be found by least squares. In particular, if we represent the estimate of each region by the vector $\mathbf{c}_i = \hat{\mathbf{t}}_i \times \mathbf{p}_i$ then the optimal $\mathbf{T}$ is in the direction of the smallest eigenvector of the matrix

$$\mathbf{C} = \sum_i \mathbf{c}_i \mathbf{c}_i^T \; .$$

From the estimate of $\vec{\tau}_i$, we estimate $\vec{\omega}_i$ by taking the vector $\mathbf{m}_i$ which was computed by the motion compensation step and subtracting the component that is in the direction of the $\hat{\tau}_i$,

$$\hat{\omega}_i := \mathbf{m}_i - (\mathbf{m}_i \cdot \hat{\tau}_i)\hat{\tau}_i$$

where $\hat{\omega_i}$ is the vector perpendicular to $\hat{\tau}_i$.

Instead of using the estimated values of $\hat{\tau}_i$ however, we used the *predicted* values using the translation direction $\mathbf{T}$ from above. We found this to give significantly lower errors in $\hat{\omega}_i$.

The observer's angular velocity $\mathbf{\Omega}$ can then be estimated. We choose $\mathbf{\Omega}$ to be a vector such that the rotation component of the velocity field at (x_i, y_i) – which earlier we called $\mathbf{B}_i\mathbf{\Omega}$ – lies as close as possible to the estimated motion parallax line, in the least squares sense of minimizing:

$$\text{argmin}_{\,\mathbf{\Omega}} \sum_i \rho\left(\ \|\hat{\omega}_i\| - unit(\hat{\omega}_i) \cdot \mathbf{B}_i\mathbf{\Omega}\ \right)$$

where $\rho(e_i) = e_i^2$ as above.

The above method of computing $\mathbf{T}$ exhibits significant bias towards the optical axis.[29] To avoid this bias, we applied a "whitening" transformation to the least squares solution described above. The details of the algorithm are as follows.[37] We first compute a "whitening matrix":

$$\mathbf{M} = \sum_i \begin{bmatrix} f & 0 & -x_i \\ 0 & f & -y_i \\ -x_i & -y_i & x_i^2 + y_i^2 \end{bmatrix}$$

This matrix reweights the data based on the position of the image regions in the scene. Next use $\mathbf{M}$ to reweight the least squares matrix, $\mathbf{C}_w = \mathbf{M}^{-1/2}\mathbf{C}\mathbf{M}^{1/2}$. Choose $\mathbf{x}$ corresponding to the smallest eigenvector of $\mathbf{C}_w$. Finally set $\mathbf{T} = \mathbf{M}^{-1/2}\mathbf{x}$.

C. Robust egomotion computation

Here we present a brief summary of our robust method for finding $\mathbf{T}$ and $\mathbf{\Omega}$. We use a form of robust estimation called *M-estimation*.[38] Like least squares, M-estimation performs an optimal fit of the parameters, except that the method reduces the influence of "outliers". (For a review of robust methods in Computer vision, see Meer *et. al.*[30] and Stewart.[31])

Our robust method follows the notation and formulation of Black and Anandan.[39] We use the following error measure:

$$\rho(e_i) = \frac{e_i^2}{\sigma^2 + e_i^2}$$

where e_i is the error of the ith estimate and σ is a scale term, intended to specify the expected error in the estimates. When the error is small we have $e_i^2 \ll \sigma^2$, and $\rho(e_i) \approx e_i^2$. In this case the estimate is referred to as an "inlier", and the behavior is the same as least squares. When the error is large, $e_i^2 \gg \sigma^2$ and $\rho(e_i) \approx 1$, the data is an "outlier", and the weight of that estimate is limited. Various forms of ρ may be used, but usually have similar behavior.

With the new error measure, Eq. (10) becomes nonlinear. A simple solution is to use "iteratively reweighted least squares" (IRLS).[31] The current errors are used to define a *weighting function*,

$$W(e_i) = \frac{1}{e_i}\frac{\partial}{\partial e_i}\rho(e_i) = \frac{2\sigma}{(\sigma^2 + e_i^2)^2}$$

The weighting function limits the influence of estimates that are a poor fit to the model.

The iterative algorithm is as follows. Let each estimate $\hat{\tau}_i$ have a weight $0 \leq w_i \leq 1$.

1. Begin with all weights equal, ie. $w_i = 1$.

2. Find the least squares solution for $\mathbf{T}$ using weights w_i on each $\vec{\tau}_i$ vector.

3. Reassign the weights: $w_i \leftarrow W(e_i)$

4. Repeat step 2. until convergence.

A similar process is used to find $\mathbf{\Omega}$. We used a *coarse-to-fine* optimization strategy, starting with a large value of σ and decreasing it to the size of the expected error in the estimates.

References

1. J. H. Rieger and L. Toet, “Human visual navigation in the presence of 3D rotations,” Biological Cybernetics **52**, 377–381 (1985).
2. W. H. Warren and D. J. Hannon, “Eye movements and optical flow,” Journal of the Optical Society of America A **7**(1), 160–169 (1990).
3. L. Li and W. H. Warren, “Perception of heaing during rotation: sufficiency of dense motion parallax and reference objects,” Vision Research **40**, 3873–3894 (2000).
4. H. L. F. von Helmholtz, *Treatise on Physiological Optics.* (Dover, New York, 1925).
5. J. Barron, D. Fleet, and S. Beauchemin, “Performance of Optical Flow Techniques,” International Journal of Computer Vision **12**(1), 43–77 (1994).
6. K. Mutch and W. Thompson, “Analysis of Accretion and Deletion at Boundaries in Dynamic Scenes,” IEEE Transactions on Pattern Analysis and Machine Intelligence **7**(2), 133–138 (1985).
7. M. J. Black and D. J. Fleet, “Probabilistic detection and tracking of motion discontinuities,” International Journal of Computer Vision **38**(3), 229–243 (2000).
8. J. Wang and E. Adelson, “Layered Representation for Motion Analysis,” in *IEEE Conf. on Computer Vision and Pattern Recognition*, pp. 361–366 (1993).
9. J. Bergen, P. Burt, R. Hingorani, and S. Peleg, “A Three-Frame Algorithm for Estimating Two-Component Image Motion,” IEEE Transactions on Pattern Analysis and Machine Intelligence **14**(9), 886–896 (1992).
10. B. Horn and E. Weldon, Jr., “Direct Methods for Recovering Motion,” International Journal of Computer Vision **2**(1), 51–76 (1988).
11. N. Hatsopoulos and J. Warren, W.H., “Visual navigation with a neural network.” Neural Networks **4**, 303–317 (1991).
12. Y. Aloimonos and Z. Duric, “Estimating The Heading Direction Using Normal Flow,” International Journal of Computer Vision **13**(1), 33–56 (1994).
13. D. Sinclair, A. Blake, and D. Murray, “Robust Estimation of Egomotion from Normal Flow,” International Journal of Computer Vision **13**(1), 57–69 (1994).
14. H. Longuet-Higgins, “A Computer Algorithm for Reconstructing a Scene from Two Projections,” Nature **293**, 133–135 (1981).
15. R. Mann and M. S. Langer, “Estimating camera motion through a 3D cluttered scene,” in *First Canadian Conference on Computer and Robot Vision* (London, Canada, 2004).
16. H. Longuet-Higgins and K. Prazdny, “The Interpretation of a Moving Retinal Image,” Proceedings of the Royal Society of London B **B-208**, 385–397 (1980).
17. J. H. Rieger and D. T. Lawton, “Processing Differential Image Motion,” J. Opt. Soc. Am. A **2**, 254–260 (1985).

18. M. S. Langer and R. Mann, “Optical Snow,” International Journal of Computer Vision **55**(1), 55–71 (2003).
19. A. Watson and A. Ahumada, “Model of Human Visual-Motion Sensing,” J. Opt. Soc. Am. A **2**(2), 322–342 (1985).
20. B. Lucas and T. Kanade, “Optical Navigation by the Method of Differences,” in *International Joint Conf. on Artificial Intelligence*, pp. 981–984 (1985).
21. D. J. Fleet, *Measurement of Image Velocity* (Kluwer Academic Press, Norwell, MA, 1992).
22. E. P. Simoncelli, “Distributed Analysis and Representation of Visual Motion,” Ph.D. thesis, Massachusetts Institute of Technology (1993).
23. D. J. Fleet and K. Langley, “Computational analysis of non-Fourier motion,” Vision Research **34**(22), 3057–3079 (1994).
24. S. Beauchemin and J. Barron, “The Frequency Structure of 1D Occluding Image Signals,” IEEE Transactions on Pattern Analysis and Machine Intelligence **22**(2), 200–206 (2000).
25. S. Beauchemin and J. Barron, “On the Fourier Properties of Discontinuous Motion,” Journal of Mathematical Imaging and Vision **13**(3), 155–172 (2000).
26. D. J. Field, “Relations between the statistics of natural images and the response properties of cortical cells,” J. Opt. Soc. Am. A **4**, 2379 (1987).
27. S. Ullman, *The Interpretation of Visual Motion* (MIT Press, 1979).
28. E. Hildreth, “The Computation of the Velocity Field,” Proceedings of the Royal Society of London **B-221**, 189–220 (1984).
29. D. J. Heeger and A. D. Jepson, “Subspace methods for recovering rigid motion,” International Journal of Computer Vision **7**(2), 95–117 (1992).
30. P. Meer, D. Mintz, D. Kim, and A. Rosenfeld, “Robust Regression Methods for Computer Vision: A Review,” International Journal of Computer Vision **6**(1), 59–70 (1991).
31. C. Stewart, “Robust Parameter Estimation in Computer Vision,” SIAM Rev **41**(3), 513–537 (1999).
32. G. Adiv, “Inherent ambiguities in recovering 3-D motion and structure from a noisy flow field,” IEEE Trans. on Pattern Analysis and Machine Intelligence **11**, 477–489 (1989).
33. M. Irani, B. Rousso, and S. Peleg, “Recovery of Egomotion using Region Alignment,” IEEE Trans. on Pattern Analysis and Machine Intelligence **19**(3), 268–272 (1997).
34. E. C. Hildreth and C. S. Royden, *High-Level Motion Processing*, chap. Computing Observer Motion from Optical Flow, pp. 269–294 (MIT Press, Cambridge, MA, USA, 1998).
35. M. Lappe and J. P. Rauschecker, “A neural network for the processing of optical flow from egomotion in man and higher mammals,” Neural Computation **5**, 374–391 (1993).
36. J. A. Perrone and L. S. Stone, “A Model of Self-motion Estimation Within Primate

Extrastriate Visual Cortex," Vision Research **34**(21), 2917–2938 (1994).

37. W. MacLean, "Removal of Translation Bias when using Subspace Methods," in *International Conference on Computer Vision (ICCV)*, pp. 753–758 (1999).
38. P. Huber, *Robust Statistics* (Wiley, 1981).
39. M. Black and P. Anandan, "The Robust Estimation of Multiple Motions: Parametric and Piecewise-Smooth Flow-Fields," Computer Vision and Image Understanding **63**(1), 75–104 (1996).