Motion parallax in cluttered scenes: 
frequency domain PCA

M. S. Langer\(^1\), V. Chapdelaine-Couture\(^2\), R. Mann\(^3\), and S. Roy\(^2\)

\(^1\) McGill University (School of Computer Science), Montreal, Quebec, H3A 2A7 Canada
\(^2\) University of Montreal (DIRO), Montreal, Quebec, H3T 1J4 Canada
\(^3\) University of Waterloo (School of Computer Science), Waterloo, Ontario, N2L 3G1 Canada

Abstract

When an observer moves through a rigid 3D cluttered scene such as a forest, the set of image velocities in each local image region is constrained to lie near a line in velocity space. Estimating the directions of such lines in different regions is useful for computing the observer’s 3D motion, in particular, since each of these lines points towards the observer’s heading direction. Recently we introduced a frequency domain method for estimating the motion parallax line of a local region. The method was based on a bowtie distribution of the 3D power spectrum. Here we introduce a simpler method which is based on a 2D principal components analysis of the sum of squared normalized power spectrum. The new method avoids motion compensation, which is typically used to subtract out the average velocity in the local region. The new method thus demonstrates that the computational problem of estimating the average velocity in a local region can be treated independently of the problem of estimating the direction of motion parallax in the region.

Index Terms

(EDICS) Motion detection and estimation: Optical flow, Transform based approaches, parametric models

I. INTRODUCTION

When an observer moves through a rigid scene, nearby points move with a different image velocity than distant points, a phenomenon known as motion parallax. This paper addresses the problem of estimating motion parallax for an observer moving through a rigid 3D cluttered scene. Cluttered scenes are abundant in nature, for example, when trees or bushes are present.

When an observer moves through a rigid 3D cluttered scene, the resulting pointwise image velocity field is difficult to estimate because there are a large number of depth discontinuities and a large number of depth layers. This clutter tends to violate the assumptions of standard image measurement methods, namely that the velocity field is smooth [1] or that only a small number of independent layers are present [3], [12], [13], [6], [4], [24].

Despite the violation of these standard assumptions, there still exist strong constraints on the image velocity field. When a local image region contains visible surfaces at a wide range of depths, the image velocity vectors lie near a motion parallax line [18],

\[(v_x, v_y) = (\omega_x + \alpha \tau_x, \omega_y + \alpha \tau_y)\]

(1)

where, without loss of generality, \((\tau_x, \tau_y)\) is a unit vector, and \((\tau_x, \tau_y)\) and \((\omega_x, \omega_y)\) are orthogonal. Eq. (1) can be derived from the equations of egomotion introduced by Longuet-Higgins and Prazdny[15] as follows.
Let \((X, Y, Z)\) be a coordinate system with the observer at the origin. Let \(Z\) be the optical axis and let \(X, Y\) be the horizontal and vertical directions in the observer’s frame. The observer’s 3D instantaneous translation velocity in this coordinate system is a 3D vector \(\mathbf{T} = (T_x, T_y, T_z)\) and the observer’s instantaneous angular velocity is a 3D vector \(\mathbf{\Omega} = (\Omega_x, \Omega_y, \Omega_z)\). 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 \((x, y)\). Then the image velocity vector \(\mathbf{v}\) at \((x, y)\) is the sum of two component vectors, 

\[
\mathbf{v} = \mathbf{v}_\Omega + \frac{\mathbf{v}_T}{Z(x, y)}
\]

which are due to the observer’s rotation \(\mathbf{\Omega}\) and translation \(\mathbf{T}\) respectively, where

\[
\mathbf{v}_\Omega = \begin{bmatrix}
x y/f & -f - x^2/f & y \\
x + y^2/f & -xy/f & -x
\end{bmatrix} \begin{bmatrix}
\Omega_x \\
\Omega_y \\
\Omega_z
\end{bmatrix}
\]

and

\[
\mathbf{v}_T = T_z \begin{bmatrix}
x - x_T \\
y - y_T
\end{bmatrix}
\]

The special image position

\[
(x_T, y_T) = \frac{f}{T_z}(T_x, T_y)
\]

is called the axis of translation (AOT). In the case of lateral motion \((T_z = 0)\), the AOT is at infinity in the image plane and the translation component is written

\[
\mathbf{v}_T = \begin{bmatrix}
-x_T \\
-f T_x \\
-f T_y
\end{bmatrix}
\]

Eq. (1) is easily derived from Eq. (2). Consider a local image region which is centered at \((x_0, y_0)\). In this local region, the vectors \(\mathbf{v}_\Omega\) and \(\mathbf{v}_T\) are approximately constant since \((x, y) \approx (x_0, y_0)\), and the approximation improves as the region shrinks (see Appendix of [18]). Because \(\mathbf{v}_\Omega\) and \(\mathbf{v}_T\) are approximately constant, Eq. (2) defines a line in velocity space \((v_x, v_y)\) such that \(1/Z(x, y)\) is the free parameter for that line. Eq. (1) is merely another way of parameterizing this line. The key difference between these two parameterizations is that the \(\tau\) and \(\omega\) vectors in Eq. (1) are defined to be orthogonal and \(\tau\) is defined to be of unit length, whereas the \(\mathbf{v}_\Omega\) and \(\mathbf{v}_T\) are defined independently of each other via Eqns. (3) and (4).

The main contribution of this paper is to introduce a new efficient method for estimating the parameters \((\tau_x, \tau_y)\). These parameters define the direction of motion parallax. These parameters are significant because, for any local region, the vector \((\tau_x, \tau_y)\) points from that region toward the AOT [15], [20], [9]. If the observer can estimate the direction of motion parallax in several local regions, it can in turn estimate the direction in which it is heading through the scene.

The new method is based on frequency domain properties of motion. The classical motion plane property [23], [10] states that an image that is translating with uniform image velocity \((v_x, v_y)\) pixels per frame produces a plane
of energy in the 3D spatio-temporal frequency domain:

\[ v_x f_x + v_y f_y + f_t = 0 \quad . \tag{7} \]

The intuition behind this motion plane equation is that when an image translates with this velocity, each spatial frequency component of an image translates with this velocity as well and this yields a linear relationship between the temporal frequency \( f_t \) at a point \((x, y)\), and the spatial frequencies.

Motion parallax occurs when multiple velocities are present. Substituting Eq. (1) into Eq. (7) then yields a family of planes in the frequency domain [14],

\[
(\omega_x + \alpha \tau_x) f_x + (\omega_y + \alpha \tau_y) f_y + f_t = 0 \quad .
\]

These planes form a bowtie pattern (see Fig. 1). The planes intersect at a common line called the axis of the bowtie (see dotted line of Fig. 1). The axis passes through the origin and is in direction \((-\tau_y, \tau_x, \sqrt{\tau_x^2 + \tau_y^2})\) in the 3D frequency domain. The \((f_x, f_y)\) component of the bowtie axis is \((-\tau_y, \tau_x)\) which is perpendicular to the direction of motion parallax \((\tau_x, \tau_y)\).

![Fig. 1](image)

Fig. 1. If the velocity vectors in a local region lie on a motion parallax line (Eq. 1) then the 3D power spectrum has a bowtie distribution. Each plane has a corresponding \(\alpha\) value in Eq. 1). The dotted line where the planes intersect is the bowtie axis.

We recently introduced a two stage frequency-based method for finding this bowtie axis for any local image region containing a range of depths [17], [18]. The first stage uses a frequency based least squares method for estimating the mean velocity in the region [16], [22]. This first stage fits a single motion plane to the 3D power spectrum. From this mean velocity estimate, the power spectrum is sheared in the \(f_t\) direction. (Details are reviewed in Sec. II). This shearing amounts to motion compensation. Following this shearing, the bowtie axis now lies in the \(f_t = 0\) plane, namely, the bowtie axis is now \((-\tau_y, \tau_x, 0)\). The second stage of the method finds this bowtie
axis using a brute force template match [14]. Although this method yielded good estimates of the bowtie axis, it was computationally slow.

Recently, we introduced a different method for finding the bowtie axis which is based on a 3D principal components analysis [5]. The idea is that frequency components near the bowtie axis tend to have large power because all motion planes contribute to these frequencies, and this concentration of power can be estimated using PCA. The basic argument was presented in [5] along with preliminary experimental results to support the argument.

In the present paper, we develop the PCA method further by showing how to estimate the direction of motion parallax \((\tau_x, \tau_y)\) with first performing motion compensation. A brief overview of the paper is as follows. In Sec. II, we develop the PCA method and relate it to previous frequency domain methods for estimating image motion. We also show how to collapse the PCA method from 3D to 2D, namely from the spatio-temporal frequency domain to the spatial frequency domain only. This step allows us to avoid the motion compensation step used in our previous methods, and thereby estimate the \((\tau_x, \tau_y)\) more directly, while skipping the estimation of \((\omega_x, \omega_y)\). In Sec. III, we provide technical details on how to relate the PCA problem to the problem of estimating the \((f_x, f_y)\) component of the bowtie axis. In Sec. IV, we present experimental results.

## II. AVOIDING MOTION COMPENSATION

A common computational approach to estimating image velocities is to perform motion compensation [16]. One finds a single image velocity vector \((v_x, v_y)\) that best explains the image intensity changes from frame to frame in some local image region, in the sense of minimizing some measure of the intensity differences. This velocity vector serves as an estimate on the mean velocity in a region. In our previous methods [18], [5], motion compensation was used to shear the amplitude spectrum such that the resulting sheared bowtie axis lies in the \((f_x, f_y)\) plane. This reduced the search for the bowtie axis from a line in 3D to a line in 2D.

In this section, we compare two frequency domain formulations of motion compensation. We show that these two formulations are equivalent, once motion compensation has been computed. A surprising result then emerges, namely that the PCA analysis can be collapsed from 3D to 2D and moreover that the 2D analysis can be performed without motion compensation. This theoretical result allows us to estimate the bowtie axis parameters \((\tau_x, \tau_y)\) without first performing motion compensation.

The first frequency domain formulation of motion compensation [10], [22] finds the vector \((v_x, v_y)\) that minimizes the sum of weighted distances to a motion plane, such that the distance is measured in the \(f_t\) direction only. Consider an \(N \times N\) image and \(T\) frames. From the motion plane equation, we define the sum of weighted squared distances from the motion plane in the \(f_t\) direction:

\[
\epsilon_1(v_x, v_y) \equiv \sum_{f_x, f_y, f_t} | W_{x,y,t} |^2 (f_t + v_x f_x + v_y f_y)^2
\]  

(8)
where the weights are
\[ W_{x,y,t} \equiv W(f_x, f_y, f_t) \]
and the spatial and temporal frequencies are below the Nyquist limits
\[ f_x, f_y \in \{-\frac{N}{2}, \ldots, \frac{N}{2} - 1\}, \quad f_t \in \{-\frac{T}{2}, \ldots, \frac{T}{2} - 1\}. \tag{9} \]

Eq. (8) can be re-expressed as a matrix product. Let \( F \) be the \( N^2T \times 3 \) matrix that lists the \( N^2T \) frequency triplets \((f_x, f_y, f_t)\) as defined in (9), and let \( W \) be the corresponding \( N^2T \times N^2T \) diagonal matrix of weights \( W_{x,y,t} \). Define the 3 × 3 matrix:
\[
F^T W^2 F \equiv \begin{pmatrix}
\sum f_x^2 W_{x,y,t}^2 & \sum f_x f_y W_{x,y,t}^2 & \sum f_x f_t W_{x,y,t}^2 \\
\sum f_x f_y W_{x,y,t}^2 & \sum f_y^2 W_{x,y,t}^2 & \sum f_y f_t W_{x,y,t}^2 \\
\sum f_x f_t W_{x,y,t}^2 & \sum f_t f_y W_{x,y,t}^2 & \sum f_t^2 W_{x,y,t}^2 \\
\end{pmatrix} \tag{10}
\]
where each summation \( \sum \) is over \( f_x, f_y, f_t \) as defined in (9).

Finally, for any \((v_x, v_y)\) in Eq. (8), let \( v \equiv (v_x, v_y, 1)^T \) be a vector normal to the corresponding motion plane. Eq. (8) may be written:
\[
\epsilon_1(v_x, v_y) \equiv v^T F^T W^2 F v \tag{11}
\]
The first formulation of the motion compensation problem is to find the vector \((v_x, v_y)\) that minimizes \( \epsilon_1(v_x, v_y) \).

This formulation needs to be modified slightly to account for temporal aliasing, which occurs when there are high image speeds present, i.e. \(|v_x^2 + v_y^2|\) is large. Temporal aliasing causes motion planes to wrap around at the Nyquist frequency (see Fig. 4). This aliasing is not accounted for by the distance function of Eq. (11). The traditional way to perform motion compensation is to use a multiscale method [16], [2]. For example, when minimizing \( \epsilon_1(v_x, v_y) \), an initial estimate of \((v_x^0, v_y^0)\) is obtained using low spatial frequencies only. The weighting function \( W(f_x, f_y, f_t) \) is then sheared iteratively in the \( f_t \) direction, with wraparound where it exceeds Nyquist limit. At the \( n^{th} \) iteration, we write this shear transformation as:
\[
W^n(f_x, f_y, f_t) \leftarrow W(f_x, f_y, (f_t + v_x^n f_x + v_y^n f_y)). \tag{12}
\]
Interpolation is typically needed since \( f_t + v_x^n f_x + v_y^n f_y \) is typically not an integer. Also, note that the temporal frequency on the right hand side of (12) need not be constrained to lie in \([-T/2, T/2]\] since \( W(f_x, f_y, f_t) \) is periodic with period \( T \).

The minimization of (11) is then computed using the sheared spectrum \( W^n(f_x, f_y, f_t) \) to obtain an incremental \((v_x, v_y)\), and this incremental velocity is added to the currently velocity estimate,
\[
(v_x^{n+1}, v_y^{n+1}) \leftarrow (v_x^n, v_y^n) + (v_x, v_y). 
\]
Higher spatial frequencies are included in each iteration, since temporal aliasing is less likely to occur when the motion has been partly compensated. The computation terminates when the motion is fully compensated, that is, when the incremental velocity \((v_x, v_y)\) is sufficiently close to \((0, 0)\).

In the multiscale method, each iterative estimate of \((v_x, v_y)\) uses the sheared \(W(f_x, f_y, f_t)\) to minimize \(\epsilon_1(v_x, v_y)\). Higher frequencies are included in each iteration, since temporal aliasing is less likely to occur when the motion has been partly compensated. The computation terminates when the motion is fully compensated, that is, when the velocity \((v_x, v_y)\) that best fits the sheared weights is sufficiently close to \((0, 0)\). Such a velocity vector approximately minimizes \(\epsilon_1(v_x, v_y)\) for the sheared weights.

The second frequency domain method estimates mean image velocity in a region by minimizing the perpendicular distance to a motion plane, rather than the distance in the 2D plane. We will use this fact next, when we discuss the eigenvectors of \(W\).

Thus the sum of squared weighted perpendicular distances is

\[
e_2(v) = \frac{1}{v^T v} \sum_{f_x, f_y, f_t} W_{x,y,t}^2 |f_t + v_x f_x + v_y f_y|^2
\]

or equivalently,

\[
e_2(v) = (v^T F^T W^2 F v) / (v^T v)
\]

If the weights \(W(f_x, f_y, f_t)\) have been sheared by motion compensation, then \(e_2\) has a local minimum at \((0, 0, 1)\). To see why, recall that \(\epsilon_1(v)\) has a minimum at \((v_x, v_y) = (0, 0)\) after motion compensation. Equivalently, when \(W(f_x, f_y, f_t)\) is restricted to the plane \(f_t = 1\), then \(W(f_x, f_y, f_t)\) has a minimum at \((0, 0, 1)\). But the unit sphere, \(||v|| = 1\), is tangent to the \(f_t = 1\) plane at \((0,0,1)\). Because \(W(f_x, f_y, f_t)\) is smooth, it follows that \(W(f_x, f_y, f_t)\) attains a local minimum at \((0,0,1)\) on the unit sphere, just as it attains a minimum there on the \(f_t = 1\) plane. We will use this fact next, when we discuss the eigenvectors of \(F^T W^2 F\).

Consider Eq. (14) on its own and allow \(v\) to be any vector in \(\mathbb{R}^3\), rather than just a vector restricted to the 2D plane \((v_x, v_y, 1)\). Then \(e_2(v)\) of Eq. (14) has the form of a Rayleigh quotient. Moreover, since the matrix \(F^T W^2 F\) is real and symmetric, there exists an orthogonal set of three eigenvectors. Assuming \(W_{x,y,t}\) has been motion compensated, we have just seen that \(e_2(v)\) has a minimum at \((v_x, v_y, 1) = (0, 0, 1)\). It follows that this minimum must be an eigenvector of \(F^T W^2 F\). Moreover, the remaining two eigenvectors can be assumed to be
orthogonal to (0,0,1) and thus they lie in \((f_x, f_y)\) planes, and so Eq. (10) must have the form:

\[
\begin{pmatrix}
\sum f_x^2 W_{xyt} & \sum f_x f_y W_{xyt} & 0 \\
\sum f_x f_y W_{xyt} & \sum f_y^2 W_{xyt} & 0 \\
0 & 0 & \sum f_t^2 W_{xyt}
\end{pmatrix}.
\]

(15)

where again each of the summations \(\sum\) is over \(f_x, f_y, f_t\), below the Nyquist limit as in (9). It follows that the \((f_x, f_y)\) components of the remaining two eigenvectors are identical to the eigenvectors of the following \(2 \times 2\) matrix, which is just the upper left part of (15):

\[
\begin{pmatrix}
\sum f_x^2 W_{xyt} & \sum f_x f_y W_{xyt} \\
\sum f_x f_y W_{xyt} & \sum f_y^2 W_{xyt}
\end{pmatrix}.
\]

(16)

Two interesting observations follow immediately. First, the sum over \(f_t\) in Eq. (16) can be treated as an inner loop over \(W_{xyt}^2\). Second, although we have assumed that weighting function \(W_{xyt}\) has been sheared via motion compensation, the inner loop sum \(\sum f_t W_{xyt}^2\) does not depend on this shearing since the shearing is in the \(f_t\) direction only. That is, in (12), for any fixed \((f_x, f_y)\) the operation

\[f_t \leftarrow (f_t + v_x^n f_x + v_y^n f_y) \mod T\]

merely translates the \(f_t\) values with wraparound, but this does not change the sum of the values over \(f_t\). Thus, although we have assumed \(W_{xyt}\) has been sheared by motion compensation, the matrix (16) does not change if we used the original (non-sheared) \(W_{xyt}\). In particular, we can compute the eigenvectors of (15) without first performing motion compensation. This is a surprising result. It is also useful since, as we shall argue in the next section, these eigenvectors provide an estimate of \((\tau_x, \tau_y)\)

### III. The Principal Eigenvector and Bowtie Axis

For any Raleigh quotient, the eigenvector with largest eigenvalue is called the *principal eigenvector*. The principal eigenvector defines the “best fit line” that passes through the origin, in that the sum of weighted orthogonal distances to this line is minimized [19]. For the 2D matrix (16), the principal eigenvector is the best fit 2D line in the 2D plane \((f_x, f_y)\).

We wish to choose \(W_{xyt}\) such that \(\sum f_t W_{xyt}^2\) is relatively large for spatial frequencies \((f_x, f_y)\) near the bowtie axis line \(\pm(\tau_y, -\tau_x)\) and relatively small for spatial frequencies far from this line. The intuition is that such a \(W_{xyt}\) would yield a best fit line that is roughly aligned with the bowtie axis line.

To choose \(W_{xyt}\), we consider the distribution of power within constant \((f_x, f_y)\) columns of the 3D frequency domain, with \(f_t\) varying within a column. Assuming a bowtie is present, columns that intersect the bowtie axis will have large power \(|\hat{I}(f_x, f_y, f_t)|^2\) for \(f_t\) at or near the bowtie axis, but small power for \(f_t\) away from the bowtie axis. For columns that do not intersect the bowtie axis, the power will be spread out over \(f_t\).
With this distribution property in mind, we define $W_{xyt}$ to be the normalized power spectrum

$$W_{xyt} \equiv \frac{|\hat{I}(f_x, f_y, f_t)|^2}{\sum_{f_t} |\hat{I}(f_x, f_y, f_t)|^2}.$$  \hspace{1cm} (17)

Clearly $0 \leq W_{xyt} \leq 1$ and, for fixed $(f_x, f_y)$,

$$\sum_{f_t} W_{xyt} = 1 .$$

We can think of $W_{xyt}$ as the whitened power spectrum [10], so that each spatial frequency $(f_x, f_y)$ has the same total normalized power.

With the above definition of $W_{xyt}$ as the normalized power spectrum, we now define the sum of squared normalized power

$$SSNP \equiv \sum_{f_t} W_{xyt}^2 ,$$

which is a function of spatial frequency $(f_x, f_y)$. In Sec. IV, we will show several examples of SSNP for different motion parallax sequences.

We expect to find relative large values of SSNP for spatial frequencies $(f_x, f_y)$ near the bowtie axis line $(-\tau_y, \tau_x)$. Two extreme examples illustrate why. One extreme is an $(f_x, f_y)$ such that $|\hat{I}(f_x, f_y, f_t)|$ is non-zero for only one value of $f_t$, which is the ideal case that the $(f_x, f_y)$ column intersects the bowtie axis. In this case, $W_{xyt}$ is 1 for that $f_t$ and is zero for all other $f_t$, and so $SSNP = 1$. The other extreme is an $(f_x, f_y)$ such that $|\hat{I}(f_x, f_y, f_t)|$ has the same value for all $f_t$. Since there are $T$ values of $f_t$, we would have $W_{xyt} = 1/T$ for all $f_t$ and so $SSNP = \frac{T}{T} = \frac{1}{T}$. It is easy to see that the general case lies between these two extremes and so

$$SSNP \in \left[\frac{1}{T}, 1\right].$$  \hspace{1cm} (18)

As argued above, the left and right limits tend to correspond, respectively, to frequencies $(f_x, f_y)$ far from and to near to the bowtie axis line. Thus, this choice of $W_{xyt}$ should yield a principal eigenvector of (16) that is roughly aligned with the bowtie axis line $(-\tau_y, \tau_x)$.

IV. EXPERIMENTS

The experiments described below show that the values of the SSNP are indeed larger near the bowtie axis, and that the principal eigenvector of (16) is a good estimate of the vector $(-\tau_y, \tau_x)$.

A. Synthetic motion layers

The first experiment uses synthetic videos that exactly obey the model of Eq. (1). The results here serve as a baseline for what we can expect in experiments for which the model does not hold exactly, namely when the motion parallax line of Eq. (1) is not exact. Each video is defined by shifting a set of opaque layers from frame to frame
according to the model of Eq. (1). We use parameters \((\tau_x, \tau_y) = (1, 1)\) and \((\omega_x, \omega_y) = (0, -3)\), and we consider up to five moving layers, \(\alpha \in \{1, 2, \ldots, 5\}\). Each layer is a set of opaque 2D tiles. These are dropped at frame \(t = 0\) and at randomly chosen positions in a 2D image plane. When a tile in layer \(i\) is dropped, the intensities for that tile replace any intensity values that may have previously been assigned to the corresponding pixel. In particular, far layers are dropped first, so that near layers occlude far tiles. For frames \(t > 0\), each tile’s position is defined according to the model of Eq. (1).

In addition to capturing occlusion relations between layers, the size and density of tiles in each layer is chosen according to the laws of linear perspective. The width of each tile in layer \(\alpha\) is chosen to be \(2\alpha\) (and hence the area of each tile is \(4\alpha^2\)). The number of tiles in layer \(\alpha\) is set to be inversely proportional to \(\alpha^2\), so that there would be roughly as many pixels covered by each layer. This latter constraint doesn’t quite hold in practice, since far layers are occluded by near layers, but this is the situation in real scenes as well.

For the purposes of illustrating the size and visibility of the tiles, Figure 2 shows an example of a single frame, in which each tile is given a single random intensity. For the actual videos that we used in the experiments, the intensities in each tile were cropped from a 1/f noise pattern. The 1/f noise pattern was defined in the standard way using spectral synthesis [7]: set the amplitude spectrum to \(1/\sqrt{f_x^2 + f_y^2}\), choose a random phase spectrum, and take the inverse Fourier transform. We do not use 1/f noise in Fig. 2 because then tiles would be much less visible in the figure. (The tiles with 1/f intensities are visible in the video because of the motion cues allows better segmentation.)

![Fig. 2. (a) Tiles in one frame of the synthetic video with five layers \((n = 5)\). (b) Intensity of each tile in (a) is drawn from intensities of a equal sized but randomly position square in a 1/f noise image such as shown here.](image)

Before taking the 3D Fourier transform of a video sequence, we multiply it by a 3D Hanning window to reduce boundary artifacts in the Fourier transform. Once the 3D Fourier transform of the windowed image sequence is computed, we compute the SSNP for all \((f_x, f_y)\).

Fig. 3 shows the results. Each plot shows the SSNP as a function of \((f_x, f_y)\). The average is taken over 100
Fig. 3. Example plots of the SSNP as a function of $(f_x, f_y)$, averaged over 100 videos with varying number of frames $T$ and layers $n$. Left column shows the $\alpha$ values of the layers for each row, so that $n = 5$ for the top row, $n = 2$ for the middle two rows, and $n = 1$ for the bottom row. Each column corresponds to a fixed number of frames $T$. The intensities in each plot have been normalized to have the same maximum value for illustration purposes.

Three effects can be observed in these data. The first is the expected one. For larger values of $T$ (say $T \geq 8$) and for the case of more than one motion layer (upper three rows), there are larger values of the SSNP in the direction of the bowtie axis, namely from the top left corner to the bottom right corner of each plot. This corresponds to a bowtie axis line $f_y = -f_x$, which is in direction $(-\tau_y, \tau_x) = (-1, 1)$. The ratio of largest to smallest values increases with $T$, as illustrated by the increase of the contrast in the plots, which is consistent with (18).

A second effect is that there are larger values of the SSNP along lines that are parallel to the bowtie axis line. (These appear in the upper two rows only). These parallel lines are a result of temporal aliasing, as sketched in Fig. 4(b). When a motion plane reaches the temporal Nyquist frequency, aliasing occurs and the motion plane wraps around and can then intersect other motion planes below the Nyquist frequency. This intersection of aliased and non-aliased motion planes can yield larger values of the SSNP, just as bowtie axis yields larger values. Notice that the size of the aliasing effects can vary from case to case. For example, the aliasing effect is most apparent in the second row.
A third effect occurs for small values of $T_s$, namely a set of parallel bands are produced (see Fig. 3 left column). The cause of the bands is reminiscent to what we discussed in Sec. III. We briefly explain the idea for the case $T = 2$. For any spatial frequency $(f_x, f_y)$, the normalized powers $W_{x,y}$ are $p$ and $1-p$ for $f_t = 0$ and 1, respectively, where $0 \leq p \leq 1$. Thus the squared normalized powers $W_{x,y}^2$ are $p^2$ and $(1-p)^2$, respectively, and so SSNP is $p^2 + (1-p)^2$. For different $(f_x, f_y)$, this sum can vary between 0 and 1 in a rather complicated manner which need not be directly related to the bowtie axis. For example, this banding occurs in the case of one motion plane only (Fig. 3 bottom row), where there is no bowtie axis! In this case, the bands have twice the spatial frequency in the $x$ direction as in the $y$ direction. This ratio makes sense since the single image velocity $(v_x, v_y)$ is $(2, -1)$, i.e.

$$\alpha(\tau_x, \tau_y) + (\omega_x, \omega_y) = 2 \cdot (1, 1) + (0, -3) = (2, -1).$$

Banding also results when more than one motion plane is present. We spare the readers further analysis of the bands, however, since banding is an artifact that we wish to avoid. We can avoid it simply by using larger values of $T$. The reason is that the more finely quantized $f_t$ variable is better able to represent the single motion plane. As is evident in the figures, the banding disappears as $T$ increases.

The next question is whether the larger values of the SSNP near the bowtie axis cause the principal eigenvector to be aligned with the bowtie axis, as desired. To address this question, we compute the angular error between the direction of the principal eigenvector and the bowtie axis for each of 100 videos in each condition of Fig. 3, in particular, the first three rows. To reduce the aliasing effects discussed above, we only used spatial frequencies up to half the spatial Nyquist limit $\| (f_x, f_y) \| < \frac{N}{4}$.

Table I shows the median error for each condition. The angular errors are large for $T = 2$, presumably because of the banding effects, but become quite small for larger values of $T$. For this reason, we will consider only $T \geq 8$.
TABLE I
MEDIAN OF ABSOLUTE ANGULAR DIFFERENCES (IN DEGREES) BETWEEN THE BOWTIE AXIS DIRECTION AND THE COMPUTED PRINCIPAL EIGENVECTOR, FOR LAYERED MOTION SCENES ILLUSTRATED IN FIG. 2

<table>
<thead>
<tr>
<th>Layering</th>
<th>2</th>
<th>4</th>
<th>8</th>
<th>16</th>
<th>32</th>
</tr>
</thead>
<tbody>
<tr>
<td>five layers ($\alpha = 1, \ldots, 5$)</td>
<td>19.7</td>
<td>6.0</td>
<td>2.6</td>
<td>2.5</td>
<td>2.5</td>
</tr>
<tr>
<td>two layers ($\alpha = 2,4$)</td>
<td>14.8</td>
<td>3.9</td>
<td>3.2</td>
<td>4.6</td>
<td>5.9</td>
</tr>
<tr>
<td>two layers ($\alpha = 4,5$)</td>
<td>17.8</td>
<td>4.6</td>
<td>2.4</td>
<td>2.5</td>
<td>2.9</td>
</tr>
</tbody>
</table>

in subsequent experiments.

Note that if the estimates were uniformly random then we would expect a mean error of 45 degrees. The maximum error possible on any trial is 90 degrees since the direction of the bowtie axis is only defined up to a sign ambiguity.

B. Synthetic cluttered 3D scene

As discussed in Sec. I, the motion parallax model of Eq. (1) describes a constraint on velocity vectors in a local image region. In the previous experiment, we considered a set of videos in which the motion parallax model holds exactly. In more typical scenarios, the model holds only approximately [18]. In the second experiment, we address such a scenario. We use synthetic sequences from [18]. The scenes are rendered with OpenGL. Each scene consists of squares distributed randomly in a 3D cube and oriented randomly as well. Each square is texture mapped with a 1/f noise pattern which simulates texture and shading on real surfaces. One frame from one of the sequences is shown in Fig. 5.

For each scene, we rendered a set of motion sequences that were generated by a virtual camera moving through the scene. For each sequence, the camera had near and far clipping planes of $Z_{\text{min}} = 5$ and $Z_{\text{max}} = 50$, and projection plane was $Z = 1$. The image spanned a $30 \times 30$ degree field of view. The squares were all the same 3D size, namely a width of 0.5 units. As described in [18], the scenes were rendered at double resolution, blurred and subsampled to remove spatial aliasing effects. The final sequences were $256 \times 256$ pixels and 32 frames.

We considered two camera motions, and twenty image sequences for each. The two camera motions were:

1) forward translation plus pan:
   \[ T = (0, 0, 0.25), \quad \Omega = (0, 0.234, 0). \]

2) lateral translation plus roll:
   \[ T = (-0.05, 0, 0), \quad \Omega = (0, 0, 1.25). \]

The translations are stated in world units per frame, and the rotations are in units of degrees.

For each sequence, the direction of motion parallax $\vec{\omega}_i$ was estimated for a set of 49 local overlapping image regions $i$, which defined a $7 \times 7$ grid. Each local region was of size $M \times M = 64 \times 64$ pixels and $T$ frames. The direction of motion parallax was estimated using the method described in this paper. Specifically, for each region,
Fig. 5. Example frame from sequences used in the second experiment. The scene consists of a set of randomly textured square tiles at random 3D orientation and distributed uniformly in a 3D cube.

Fig. 6. SSNP for varying number of image frames ($T$). Left column shows forward translation plus pan motion. Right column shows lateral translation plus roll motion.

The mean intensity of each region was subtracted out. The region was Hanning windowed in space and time to remove boundary effects. Then, the Fourier transform of each region was computed and used to compute SSNP for each spatial frequency ($f_x, f_y$). Finally, the 2D principal eigenvector for each region was computed based on the SSNP.

Fig. 6 shows the SSNP for a single example of each condition. This figure is analogous to Fig. 3 except that here we show the data for $T$ frames of a single image sequence, whereas Fig. 3 showed the data averaged over 100 sequences. The left column illustrates the case for forward translation + pan. The bowtie axes for the different regions are arranged on concentric circles, i.e. perpendicular to the motion parallax. Larger values on the bowtie axis are more visible near the boundary of the image, since there is more parallax in these regions. direction which
is radial from the center of the image. The right column of Fig. 6 shows the data for lateral translation plus roll. Here the direction of motion parallax \((r_x, r_y)\) is horizontal, and so the bowtie axis direction is vertical for each local region.

Fig. 7 illustrates the angular errors for the case of forward translation and pan. The average angular error between the true and computed direction of motion parallax is shown as a function of distance from the center of the image region. The reason for measuring the errors as a function of this distance is that the observer’s translation motion is forward and so the center of the image is the axis of translation AOT (recall Sec. I). One expects angular errors to be inversely related to distance from AOT, since there tends to be more motion parallax for regions farther from the AOT [18]. Since the local regions are of width 64 pixels and there is a 32 pixel overlap between regions, the closest region is at a distance of 32 pixels from the center, the second closest region is at a distance \(32 \times \sqrt{2}\) pixels, etc.

![Fig. 7. Average angular errors are shown for as a function of distance from the direction of heading (AOT) for the case of forward+pan motion and \(T = 16\). Average error is inversely related to distance from AOT, since the motion parallax increases with distance from AOT [18].](image)

Figure 8 shows a histogram of estimates of the direction of motion parallax for the case of lateral translation and roll, for \(T = 16\). The histogram shows pooled data over all 49 local regions. The true direction of motion parallax is horizontal for each local region.

### C. Real sequences

The next experiment shows how the method performs on real image sequences, each frame having \(256 \times 256\) pixels. A scene composed of plants was shot under two different camera motions. Sequence A was shot by a camera undergoing a lateral translation only. Sequence B was shot by a camera undergoing lateral translation while being gradually tilted, which produces a rotation component perpendicular to the direction of motion parallax. The true motion parallax direction for both sequences is roughly horizontal. Both sequences contain 128 frames. We use
Fig. 8. Histogram of errors in the estimates of the bowtie axis direction for the case of lateral+roll motion, and $T = 16$. The histogram contains 980 data points (49 regions $\times$ 20 sequences).

Figure 9(a) shows a single frame from sequence A. Figure 9(b) and (c) show plots of the SSNP for several image regions (see dotted lines in Fig. 9(a)) from sequences A and B respectively, with different temporal windows $T$. As $T$ is increased, the bowtie axis becomes more visible in the plots. For both sequences, the true bowtie axis is vertical line since motion parallax is horizontal. In addition, Figure 10 shows that increasing the width $N$ of the regions lowers the errors. This is expected when the observer’s translation is lateral[18]. Finally, note that the errors are bigger for the real sequences than for the synthetic ones. There are several possible reasons why: the surfaces in the real sequences have much less texture on them than those in the synthetic sequences; also, the ground truth direction of motion parallax and the camera rotation axis was only approximately known for the real sequence.

D. Performance Comparison

Computation time for any region is roughly proportional to the number of frames $T$. For a region of width $N = 64$ pixels and for $T = 32$, the following times are typical. The 3D FFT took about 0.3 seconds. The PCA computation including the computation of the SSNP took an additional 0.08 seconds. This is much faster than the method of [18] which used motion compensation followed by a brute force search for the bowtie axis. That method typically used about 6 seconds total per local region (0.65 sec for motion compensation and over 5 sec for the brute force bowtie search). The new method is also must faster than the previous 3D PCA method [5]. The reason is that the 3D PCA method requires motion compensation as a first stage (0.65 seconds, as quoted above). Quoted values are on an AMD Athlon 1.6 GHz cpu, running Matlab version 6.5.

The accuracy of the new 2D PCA method is near identical to the 3D PCA method [5]. It was shown previously [5] that the 3D PCA method has similar accuracy to the method of [18]. Thus there is no tradeoff of accuracy for the decrease in computation time for the new method.
V. CONCLUSIONS

We have addressed the problem of estimating the direction of motion parallax in local image regions, for the case of an observer moving through a rigid 3D cluttered scene. As in [18], we have taken a frequency domain approach, exploiting a bowtie distribution of power that arises from the intersection of a set of motion planes. As in [5], we have used a PCA method. The key contribution of the present paper is to show how to collapse the 3D PCA problem to a 2D PCA problem and thereby avoid motion compensation.

The experiments showed that the method performs better when there are several different velocities present in the local image region. The reason is that power is spread out over more motion planes, and so the bowtie axis has a greater concentration of power since the motion planes all overlap there. The method still yields good results when only two depth layers are present in a region, which is often the case over local image regions in a cluttered scene.

We also found that the estimates of the direction of motion parallax were more accurate when the number of frames was larger. For \( T = 2 \) or 4 frames only, the estimates were susceptible to artifacts that arose from limited sampling of the temporal frequency \( f_t \). For \( T \geq 8 \) frames, these artifacts seem to be avoided.

The key feature of the new method is that it avoids motion compensation. This significantly speeds up the estimate of the motion parallax direction, which allows the vision system to use these estimates more quickly. Once the motion parallax directions for different local regions have been estimated, the visual system can use them to
immediately estimate the observer’s direction of translation. This heading estimate can be done with traditional methods [20], which require as input only the directions of local motion parallax.

**ACKNOWLEDGEMENTS**

The authors wish to thank NSERC and FCAR for funding this research.

**REFERENCES**


