A Fast Subspace Algorithm for Recovering Rigid Motion

Allan D. Jepson
University of Toronto

David J. Heeger
NASA-Ames Research Center

Abstract

The image motion field for an observer moving through a static environment depends on the observer's translational and rotational velocities along with the distances to surface points. Given such a motion field as input we present a new algorithm for computing the observer's motion and the depth structure of the scene. The approach is a further development of subspace methods. This class of methods involves splitting the equations describing the motion field into separate equations for the observer's translational direction, the rotational velocity, and the relative depths. The resulting equations can then be solved successively, beginning with the equations for the translational direction. Here we show how this first step can be simplified considerably. The consequence is that the observer's velocity and the relative depths to points in the scene can all be recovered by successively solving three linear problems.

1 Introduction.

The basic problem we consider is how to obtain reliable information on the motion of a camera, along with distances to various points in its environment, from measurements of image motion (optical flow) alone. Here we pursue subspace methods which have been recently introduced for solving this problem (see [4, 5, 6]). The general approach involves splitting the problem into three subproblems, each of which can be solved in the following order. First, we obtain constraints which involve only the translational direction, \( \overrightarrow{T} \), of the camera. These equations are independent of the camera's angular velocity, \( \overrightarrow{\Omega} \), and do not involve knowing the distances to points in the scene. Secondly, given the resulting estimate for the translational direction, a set of linear equations can be obtained which involve only the rotational velocity as an unknown. Finally, given estimates of both the translational direction and rotational velocities of the camera, several methods are available for obtaining reliable information about the (relative) distances to various scene points (see [1, 10]).

Unlike many previously proposed algorithms, our approach to motion analysis applies to the general case of arbitrary motion with respect to an arbitrary scene. There is no assumption of smooth or planar surfaces. The results in our previous work [5, 6] demonstrate that our approach can be stable with respect to random errors in the flow field measurements, and that it performs quite favorably when compared with other proposed approaches. It is simple to compute and it is highly parallel, not requiring iteration and not requiring an initial guess. For a brief review of the existing literature see [6].

In this paper we are primarily concerned with simplifying the first step of the subspace methods in which the direction of translation is estimated independently of the rotational velocity and depths. The input data is taken to be a discrete set of optical flow vectors, say \( \overrightarrow{u}(\overrightarrow{x}_k) \), for \( k = 1, \ldots, K \), where \( \overrightarrow{x}_k \) denotes the image position for the \( k \)-th sample. It is convenient to collect these two-vectors into a single \( 2K \)-dimensional vector, \( \overrightarrow{\sigma} \). With this notation, the constraints on \( \overrightarrow{T} \) take the simple form

\[
\overrightarrow{\psi}_i(\overrightarrow{T}) \cdot \overrightarrow{\sigma} = 0, \quad \text{for } i = 1, \ldots, K - 3, \tag{1}
\]

where \( \cdot \) denotes the usual vector inner-product. In previous work [4, 5, 6] we show how to compute the constraint vectors \( \overrightarrow{\psi}_i \), and show that these vectors are typically nonlinear functions of \( \overrightarrow{T} \). In our previous work we propose that the nonlinear problem (1) can be solved for \( \overrightarrow{T} \) simply by sampling the constraints on a mesh distributed over a hemisphere of possible orientations for \( \overrightarrow{T} \), and then seeking points of least square error[4, 5, 6].

Here we introduce an alternative method for finding the camera's translational direction, which avoids sampling \( \overrightarrow{T} \)-space at many points, and results in a linear system for the translational direction. The key observation to make is that it is possible to redefine the constraint vectors \( \overrightarrow{\psi}_i(\overrightarrow{T}) \) in equation (1) in such
a way that the first \( K - 6 \) of them depend linearly on \( T' \), while the remaining three constraint vectors are typically nonlinear functions of \( T' \). From (1) we see that the first \( K - 6 \) vectors now lead to linear constraints on the translational direction. The step in our previous method which involved sampling constraints of the form (1) over \( T' \)-space can be replaced by the construction of these linear constraints, followed by a standard linear least squares solver. As a result this new method represents a huge savings in the computational resources required to obtain an estimate for \( T' \).

There is, of course, a price to pay for this short cut. We are not using all of the available information to obtain this estimate of \( T' \) (i.e. we are omitting the three nonlinear constraints). Therefore we can expect that the new approach will be more sensitive to errors in the input. Fortunately, we can show that for general scenes containing a rich depth structure this linear approach still does provide a robust estimate. This conclusion is supported by preliminary experiments on the expected accuracy of the approach.

We first sketch the new algorithm for recovering the translational velocity given only optical flow data. The approach for recovering the rotational velocity and the inverse depths remains the same as in [6]. The theoretical justification for our new algorithm is discussed in subsequent sections, and the results of an implementation are presented in Section 6.

2 Basic Algorithm.

The basic situation we consider is an observer moving through a stationary environment. In the observer's coordinate frame this is equivalent to the scene undergoing a rigid motion, which is completely characterized by a translational velocity \( T' \) and an angular velocity \( \Omega \). In particular, the instantaneous velocity of the point \( X(t) \) is

\[
\frac{dX}{dt} = \dot{T}' + \Omega \times X. \tag{2}
\]

Here we take \( X \equiv (X_1, X_2, X_3) \) to be a right-handed coordinate system fixed on the observer, with the nodal point of the imaging system at the origin. We set the focal length to \( f \), and denote the image point at \((X_1, X_2, f)\) by image coordinates \( \bar{x} \equiv (x_1, x_2, 1) = (X_1/f, X_2/f, 1) \) (for vector operations later in this paper it is convenient to write \( \bar{x} \) as a 3-vector). We put the transducer surface in front of the nodal point to avoid the need to reflect the image coordinates.

We assume that we are given the optical flow (image velocity) at a set of image positions, \( \{\bar{x}_k\} \), for a single frame of the image sequence. The optical flow data at each point is first expanded into the 3-vector

\[
q(\bar{x}_k) = Q(\bar{x}_k)u(\bar{x}_k), \tag{3a}
\]

where

\[
Q(\bar{x}) = \begin{pmatrix} 0 & 1 & 0 \\ -1 & 0 & x_3 \\ x_2 & -x_1 & 0 \end{pmatrix} \tag{3b}
\]

Notice that this preprocessing step is local; each image velocity is transformed separately.

The sample points \( \{\bar{x}_k\} \) are subdivided into \( N \) (usually overlapping) patches. In the computed examples we use sample points from small square regions in the image (e.g. a \( 3 \times 3 \) patch is sufficient). More generally, let the \( n^{th} \) image patch consist of sample points \( \{\bar{x}_k\}_{k=1}^N \). For each patch we define a particular coefficient vector, \( \tilde{c}_i \equiv (c_{i1}, \ldots, c_{iN})^T \). The details of the computation of \( \tilde{c}_i \) are presented further below. One important property of this coefficient vector is that it is independent of any affine transformation of the sampling points, and depends only on the particular pattern of sample points in the patch. Therefore, in situations where the patches are all chosen to be the same, the same coefficient vector \( \tilde{c} \) can be used for all the patches.

Given the transformed optical flow vectors \( q(\bar{x}_k) \), and the (precomputed) coefficient vector \( \tilde{c}_i \) for the \( n^{th} \) image patch, we next build the translation constraint vector

\[
\tilde{r}_n \equiv \begin{pmatrix} \tilde{c}_1 \cdot (q_1(\bar{x}_1), \ldots, q_1(\bar{x}_K))^T \\ \tilde{c}_2 \cdot (q_2(\bar{x}_1), \ldots, q_2(\bar{x}_K))^T \\ \vdots \\ \tilde{c}_N \cdot (q_N(\bar{x}_1), \ldots, q_N(\bar{x}_K))^T \end{pmatrix}. \tag{4}
\]

That is, the \( i^{th} \) component of \( \tilde{r}_n \) is obtained by taking the inner product of the coefficient vector \( \tilde{c}_i \) with the vector formed from the \( i^{th} \) component of \( q(\bar{x}) \) sampled over the image patch.

An important special case for the above computation is when the optical flow is sampled on a regularly spaced grid, and the image patches are all taken to have the same sampling pattern (e.g. a \( 3 \times 3 \) square grid). In such a case it is convenient to use the center position of the patch, say \( \bar{z} \), to denote the patch instead of the arbitrary index \( n \). Also, since the sampling patterns are all the same, the coefficient vectors \( \tilde{c} \) are the same for each patch. Given this small shift in notation, the computation in equation (4) is equivalent to a convolution operation applied to the three "images" \( q_i(\bar{z}) \), for \( i = 1, 2, 3 \). The coefficients of the
convolution mask are given by the coefficient vector \( \vec{c} \), and the same mask is used for all three images. The results of these three convolution operations contain the components of the translation constraint vector, namely \( \tau_i(\vec{x}) \), for \( i = 1, 2, 3 \), respectively. Therefore the computation of the translation constraint vectors \( \tau_i(\vec{x}) = \vec{c}_i \) is an extremely simple one given regularly spaced optical flow data. However, for generality we continue to consider the use of different sampling patterns across the image and therefore continue to index patches by \( n \) in the analysis below.

The particular construction of the coefficient vectors \( \vec{c}_n \) ensures that the resulting translation constraint vector, \( \tau_n \), is perpendicular to the true translational direction. Taken over all \( N \) image patches, each with its own \( \vec{c}_n \), we now find that the translational direction \( \vec{T} \) of the observer must satisfy the linear equation

\[
\begin{pmatrix}
\tau_{1T} \\
\tau_{2T} \\
\vdots \\
\tau_{NT}
\end{pmatrix} \vec{T} = \vec{0}.
\]

We can solve this in a least squares sense by accumulating the following 3 \( \times \) 3 symmetric matrix

\[
D \equiv \sum_{n=1}^{N} \tau_n \tau_n^T.
\]

The least squares estimate for the translational direction is then given by the eigenvector for the smallest eigenvalue of this 3 \( \times \) 3 matrix \( D \).

In summary, the algorithm requires a linear transformation to be applied to the optical flow data. For a regular sampling pattern, this operation is conveniently implemented by first transforming the optical flow data \( \vec{u}(\vec{x}) \) at each sample point to a particular vector, \( \vec{q}(\vec{x}) \). Three images, \( q_i(\vec{x}) \), are formed from each component of this vector. Each of these images is then convolved with the same precomputed mask, and the three resulting images together provide an image of translation constraint vectors. That is, for each spatial position there is a translation constraint vector \( \tau(\vec{x}) \) known to be perpendicular to the observer’s translational direction. These constraint vectors are finally compiled into the three by three matrix \( D \), from which the translational direction can easily be obtained. It is also possible, of course, to avoid explicitly forming any of these intermediate images, and instead directly build the matrix \( D \) from the results of an equivalent linear transformation applied to the optical flow for each image patch. We choose to present the algorithm in the expanded form primarily to emphasize the simplicity of the computations involved.

The following three sections briefly discuss why this magic works. In particular the construction of the coefficient vectors \( \vec{c}_n \) is of central importance. We also briefly sketch some results on the expected robustness of the approach. Alternatively, you may avoid these terse technical sections by skipping directly to the example in Section 6.

3 Samples of the Motion Field.

In this section we review the well known relationship between three dimensional motion and the motion of image points on a receptor surface. For theoretical work in later sections we need to derive the image motion for both planar and spherical receptor surfaces. To do this it is convenient to assume that the transducer array is organized on the receptor surface specified by \( \vec{X} = P(\vec{v}) \), with \( \vec{p}(\vec{v}) \equiv r_0(\vec{s}) \vec{s} \) and \( \vec{s} \) is on the unit sphere (i.e. \( ||\vec{s}|| = 1 \) where \( || \cdot || \) refers to the usual Euclidean norm). Here \( r_0(\vec{s}) \) gives the distance, and \( \vec{s} \) specifies the direction, from the nodal point to the point \( \vec{p}(\vec{v}) \) on the receptor surface. For a scene point \( \vec{X} \), its corresponding image point is then simply given by \( \vec{p}(\vec{v}) \) where \( \vec{s} = \vec{X}/||\vec{X}|| \). As above we put the transducer surface in front of the nodal point. For example, the planar receptor surface \( X_3 = f \) is given by

\[
\vec{p}(\vec{v}) = \frac{f}{s_3} \vec{s} = \frac{f \vec{X}}{||\vec{X}||}
\]

for \( s_3 > 0 \), where the vector \( \vec{s} = (s_1, s_2, s_3)^T \).

We use the 3-vector \( \vec{v}(\vec{s}) \) to denote the image flow on a general receptor surface. For the special case of a planar surface, the third component of \( \vec{s} \) is identically zero, and we use the 2-vector \( \vec{u}(\vec{s}) \) to denote the first two components of \( \vec{v} \). For rigid motion, as described by (2), it can be shown that the image velocity \( \vec{v}(\vec{s}) \) is given by (see [7]),

\[
\vec{v}(\vec{s}) \equiv \frac{d\vec{p}}{dt} = \frac{1}{||\vec{X}||} R(\vec{s}) P(\vec{v}) \vec{s} + R(\vec{s}) \left[ \left[ \vec{X} \vec{s} \right] \right].
\]

Here \( P(\vec{v}) \) is the projection operator onto the plane perpendicular to \( \vec{s} \), that is, \( P(\vec{s}) \equiv I - \vec{s} \vec{s}^T \), and \( R(\vec{s}) \) is the Jacobian matrix, \( R(\vec{s}) \equiv \frac{d\vec{p}}{d\vec{s}}(\vec{s}) \). For the planar receptor surface given above, we have

\[
R(\vec{s}) = \frac{f}{s_3} \begin{pmatrix} 1 & 0 & -s_1/s_3 \\ 0 & 1 & -s_2/s_3 \\ 0 & 0 & 0 \end{pmatrix}.
\]
Substitution of (9) into (8) provides the standard motion field equations, for example, as provided in [9].

Note that for the special case of the unit sphere, \( \vec{r}(\vec{s}) = \vec{\xi} \), and \( R(\vec{\xi}) = I \). This puts (8) into a particularly simple form. For much of the theoretical work it is convenient to make use of this simplicity by considering the motion field mapped onto the unit sphere.

In general, suppose we have \( K \) distinct sample points on a patch of a planar image, say \( \{ \vec{r}(\vec{s}_k) \}_{k=1}^K \). As in Section 1, we collect the optical flow observations into the vector

\[
\vec{\vec{O}}^T = (\vec{n}^T(\vec{s}_1), \ldots, \vec{n}^T(\vec{s}_K)) \in \mathbb{R}^{2K},
\]

(10)

where \( \vec{n} \) is obtained from the first two components of \( \vec{v} \). From equation (8) it follows that the observation vector \( \vec{O} \) satisfies

\[
\vec{O} = C(\vec{T}) \left( \begin{array}{c} \vec{p} \\ \vec{\Omega} \end{array} \right) \equiv A(\vec{T}) \vec{p} + B \vec{\Omega},
\]

(11)

where

\[
\vec{p}^T = \left( \frac{1}{||\vec{X}(\vec{s}_1)||}, \ldots, \frac{1}{||\vec{X}(\vec{s}_K)||} \right),
\]

and

\[
C(\vec{T}) \equiv \left( \begin{array}{cc} A(\vec{T}) & B \end{array} \right) \in \mathbb{R}^{2K \times (K+3)}.
\]

Here \( A(\vec{T}) \) and \( B \) are \( 2K \times K \) and \( 2K \times 3 \) dimensional matrices, respectively. The matrix \( A(\vec{T}) \) can be shown to be

\[
A(\vec{T}) = \left( \begin{array}{cccc} R'(\vec{s}_1) P(\vec{s}_1) & \ldots & 0 \\ \vdots & \ddots & \vdots \\ 0 & \ldots & R'(\vec{s}_K) P(\vec{s}_K) \end{array} \right)
\]

with \( R'(\vec{s}) \) equal to the \( 2 \times 3 \) matrix formed using the first two rows of \( R(\vec{s}) \) in (9). Similarly, the matrix \( B \) in (11) is defined by

\[
B \vec{\Omega} \equiv \left( \begin{array}{c} R'(\vec{s}_1)(\vec{\Omega} \times \vec{s}_1) \\ R'(\vec{s}_K)(\vec{\Omega} \times \vec{s}_K) \end{array} \right),
\]

which is to be satisfied for all \( \vec{\Omega} \). The particular coefficients of \( B \) can of course be derived from this equation, but we omit these details (see [4, 7]).

Equation (11) expresses the image velocities at various sample points as a function of the relative motion parameters and the relative depths. We have simply written this relation in a particularly convenient form. A (well-known) observation is that \( \vec{T} \) and \( \vec{p} \) can only be determined up to a mutual scale factor. In the sequel we abuse the notation slightly, and consider \( \vec{T} \) to be the translational direction (i.e. \( ||\vec{T}|| = 1 \)), and \( \vec{p} \) to be the vector of inverse relative depths.

4 Subspace Constraint for \( \vec{T} \).

The central idea of our subspace approach is as follows. In order for the optical flow vectors in an image patch to be consistent with a rigid motion having translational direction \( \vec{T} \), we see from equation (11) that the observation vector \( \vec{O} \) must be in the range of the matrix \( C(\vec{T}) \). (The range of a matrix is simply the subspace spanned by its columns.) That is, we require the subspace consistency constraint

\[
\vec{O} \in \text{range}[C(\vec{T})].
\]

(12)

This constraint only depends on the translational direction \( \vec{T} \) and on the particular image samples used to define \( C(\vec{T}) \). It does not depend on the rotational velocity, \( \vec{\Omega} \), nor on the vector of inverse depths, \( \vec{p} \).

For situations where the \( 2K \times (K+3) \) matrix \( C(\vec{T}) \) has more rows than columns (i.e. for \( K \geq 4 \) sample points) the range of \( C(\vec{T}) \) is a strict subspace of the space of observations, \( \vec{O} \in \mathbb{R}^{2K} \), and (12) provides nontrivial conditions for the compatibility \( \vec{T} \) and \( \vec{O} \).

In order to write the subspace consistency condition in a more convenient form, suppose the vector \( \vec{\hat{O}}(\vec{T}) \in \mathbb{R}^{2K} \) is known to be perpendicular to \( \text{range}[C(\vec{T})] \). Then a necessary condition for (12) is that the component of \( \vec{O} \) in the direction \( \vec{\hat{O}}(\vec{T}) \) must vanish. That is, we require \( \vec{O} \cdot \vec{\hat{O}} = 0 \). In previous work [4, 5, 6] we precomputed a basis \( \{ \vec{\hat{O}}(\vec{T}) \}_{n=1}^{2K} \) for the entire subspace perpendicular to \( \text{range}[C(\vec{T})] \). Then our subspace consistency condition for \( \vec{O} \) is equivalent to the constraint that \( \vec{\hat{O}} \cdot \vec{O} = 0 \).

Fortunately, this constraint depends nonlinearly on \( \vec{T} \), and therefore we proposed a dense sampling approach for locating its solution.

Here we take a different approach by giving up on having sufficient conditions for subspace consistency. Instead, for each image patch we chose only one constraint vector \( \vec{\hat{O}}(\vec{T}) \) from the space of perpendicular vectors. The advantage of this approach is that we can use a constraint vector \( \vec{\hat{O}}(\vec{T}) \) that depends linearly on \( \vec{T} \), and which then leads to a linear constraint on the translational direction.

To continue we require a computationally convenient characterization of a suitable constraint vector \( \vec{\hat{O}}(\vec{T}) \). Due to space restrictions we cannot show a detailed derivation of such a vector. Instead we only summarize its construction below, and then use the result to supply some intuition for its properties.

Define \( M(\vec{T}) \) to be the \( 3 \times 2 \) matrix which maps the optical flow vectors from the planar image surface to
the equivalent vectors for the unit sphere. In particular, a short calculation shows that
\[
M(\vec{s}) \equiv \frac{g_3}{f} \begin{pmatrix} 1 - s_1^2 & -s_1 s_2 \\ -s_1 s_2 & 1 - s_2^2 \\ -s_1 s_3 & s_1 s_3 \end{pmatrix}.
\] (13)

The general structure of a perpendicular vector \( \vec{v}'(\vec{T}) \) is shown in [7] to be
\[
\vec{v}'(\vec{T}) = \left( -\frac{c_1}{s_3}\{\vec{s}_1 \times \vec{T}\}^T M(\vec{s}_1), \ldots, \frac{c_k}{s_3}\{\vec{s}_k \times \vec{T}\}^T M(\vec{s}_k) \right).
\] (14)

Here \( s_{3k} \equiv s_{3k} \) is the third component of \( \vec{s}_k \). As we sketch below, for a suitably chosen coefficient vector \( \vec{c}' \equiv (c_1, \ldots, c_K)^T \), this vector \( \vec{v}' \) is perpendicular to the subspace range \([C(\vec{T})]\).

We are interested in the vectors \( \vec{v}'(\vec{T}) \) which depend linearly on \( \vec{T} \). It is shown in [8] that a suitable constraint on \( \vec{c}' \) is
\[
F\vec{c} = 0,
\] (15)
where \( F \) is a particular \( 6 \times K \) matrix. The structure of \( F \) is most easily described in terms of a basis for the quadratic polynomials on the image plane, say \( \{1, z_1, x_2, z_1 x_2, x_2^2\} \). These basis polynomials are evaluated at the image sample points, \( \vec{z}_k = \vec{z}(\vec{s}_k) \), with \( \vec{z}(\vec{s}) \) given in (7). The \( k \text{th} \) column of \( F \) is given by the values taken by the basis polynomials at \( \vec{s}_k \), namely
\[
F_{z_k} = (1, z_{1k}, x_{2k}, z_{1k} x_{2k}, x_{2k}^2)^T.
\]
with \( \vec{s}_k = (z_{1k}, x_{2k}, 1) \). Moreover, for generic sampling patterns, \( F \) is of full rank, so we can expect equation (15) to have an \( K - 6 \) dimensional space of solutions.

In summary, we can compute a suitable perpendicular vector \( \vec{v}'(\vec{T}) \) by first computing a solution \( \vec{c}' \) to (15). Given such a coefficient vector \( \vec{c}' \), we can then use (14) to define corresponding constraint vectors \( \vec{v}'(\vec{T}) \) for all \( \vec{T} \). Since \( \vec{c}' \) does not depend on \( \vec{T} \) we see from this construction that \( \vec{v}'(\vec{T}) \) is a linear (and homogeneous) function of \( \vec{T} \), as desired. Moreover, the constraint on \( \vec{c}' \) in equation (15) is simply that it must be perpendicular to the samples of all quadratic polynomials. The \( 6 \)-dimensional space of quadratic polynomials is invariant under affine deformations. Therefore the solution vectors of (15), namely \( \vec{c}' \), can also be taken to be invariant of affine deformations of the sampling pattern. As mentioned in section 2, this invariance is important for the convolution form of our algorithm.

Some intuition for the above construction can be obtained by considering the inner-product of \( \vec{v}'(\vec{T}) \) with the observation vector
\[
\vec{d} = C(\vec{T}_0)^T \left( \begin{array}{c} \vec{P} \\ \vec{O} \end{array} \right) = A(\vec{T}_0)\vec{F} + B\vec{O}.
\] (16)
That is, \( \vec{d} \) corresponds to rigid motion with translational velocity \( \vec{F}_0 \). We wish to show that \( \vec{v}'(\vec{T}) \) is perpendicular to range \([C(\vec{T}_0)]\) or, equivalently, that \( \vec{v}'(\vec{T}_0) : \vec{d} = 0 \) for all \( \vec{F} \) and \( \vec{O} \) in (16). By (14),
\[
\vec{v}'(\vec{T}) : \vec{d} = \sum_{k=1}^{K} \frac{c_k}{s_{3k}} (\vec{s}_k \times \vec{T})^T M(\vec{s}_k) \vec{u}(\vec{s}_k)
\]
\[
= \sum_{k=1}^{K} \frac{c_k}{s_{3k}} (\vec{s}_k \times \vec{T})^T (P(\vec{s}_k)\vec{P}_0 \vec{F}_k + \vec{O} \times \vec{s}_k).
\]
In the last step we have used the fact that \( M(\vec{s}) \) maps the optical flow vectors, \( \vec{u} \), for the planar receptor surface to the equivalent vectors, \( \vec{v}' \), for the unit sphere. As a consequence, we can now use the simpler expressions for the motion field on the sphere. This provides
\[
\vec{v}'(\vec{T}) : \vec{d} = \sum_{k=1}^{K} \frac{c_k}{s_{3k}} (\vec{s}_k \times \vec{T})^T \left( P(\vec{s}_k)\vec{P}_0 \vec{F}_k + \vec{O} \times \vec{s}_k \right).
\] (17)

The first term in square brackets above corresponds to the translational component of the flow, while the second term represents the rotational component.

First consider only the rotational component in equation (17), that is,
\[
\vec{v}'(\vec{T}) : B\vec{O} = \sum_{k=1}^{K} \frac{c_k}{s_{3k}} (\vec{s}_k \times \vec{T})^T \left( \vec{O} \times \vec{s}_k \right)
\]
\[
= \sum_{k=1}^{K} c_k (\vec{z}_k \times \vec{T})^T (\vec{O} \times \vec{z}_k).
\]
Here \( \vec{z}_k = \vec{z}(\vec{s}_k) = (1/s_{3k})\vec{s}_k \), as in equation (7). But, for a fixed \( \vec{T} \) and \( \vec{O} \), \( (\vec{z} \times \vec{T})^T (\vec{O} \times \vec{z}) \) is a quadratic polynomial. The coefficient vector \( \vec{c}' \) has been chosen to be orthogonal to all quadratic polynomials (i.e. \( \vec{c}' \) satisfies (15)). Therefore \( \vec{v}'(\vec{T}) : B\vec{O} = 0 \) for all \( \vec{F} \) and \( \vec{O} \). That is, \( \vec{v}'(\vec{T}) \) annihilates the rotational component of any (rigid) motion field.

Secondly, the translational component in equation (17) is annihilated term by term when \( \vec{T} \) agrees with
the actual translational velocity $\vec{T}_0$. In particular, one can show that $(\vec{s}_k \times \vec{T}_0)\vec{P}(\vec{s}_k)\vec{T}_0 = 0$ for all $k$. This illustrates another part of the construction of $\vec{P}(\vec{T})$, namely that the terms $(\vec{s}_k \times \vec{T})\vec{M}(\vec{s}_k)$ in (14) generate 2-vectors which are perpendicular to the translational component of any flow field for which the actual translational direction is also $\vec{T}$. 

In summary we have shown that $\vec{P}(\vec{T}_0) \cdot \vec{c} = 0$ for any choice of $\vec{c}$ and $\vec{c}$. It follows that (for any $\vec{T}_0$) the vector $\vec{P}(\vec{T}_0)$ is perpendicular to range[$\vec{C}(\vec{T}_0)$], as desired. Moreover, we have motivated the particular form of $\vec{P}(\vec{T})$ given in (14) in terms of its application to the rotational and translational components of the motion field.

Finally, the algorithm described in section 2 can be derived as follows. By the linearity of $\vec{P}(\vec{T})$, we have

$$\vec{r} \cdot \vec{T} \equiv \vec{P}(\vec{T}) \cdot \vec{c} = \sum_{j=1}^{3} \left[ \vec{P}(\vec{c}_j) \cdot \vec{c} \right] T_j.$$ 

Here $\vec{c}_j$ is the $j^{th}$ column of the $3 \times 3$ identity matrix. From the above equation we can identify the $j^{th}$ component of $\vec{r}$ to be

$$T_j = \vec{P}(\vec{c}_j) \cdot \vec{c} = \sum_{k=1}^{K} \frac{1}{s_k} (\vec{s}_k \times \vec{c}_j)\vec{M}(\vec{s}_k).$$

The term in square brackets reduces to the $j^{th}$ row of the matrix $Q(\vec{s}_k)$ introduced in equation (3b). Therefore, the appropriate $\vec{r}$ vector is equivalent to the one provided by the algorithm, as described by equations (3) and (4).

5 Stability Properties.

Before we discuss some results based on this approach we need to briefly consider the sort of information we can expect from a single constraint of the form $\vec{r} \cdot \vec{T} = 0$. Let $\vec{c}$ be as in (16), corresponding to an actual translational direction of $\vec{T}_0$. From equation (17), and the fact that $\vec{P}(\vec{T}_0)$ annihilates the rotational component of the flow field, a short calculation shows that

$$\vec{r} = \sum_{k=1}^{K} \frac{c_k}{s_k^3} p_k (\vec{T}_0 \times \vec{s}_k).$$

Two important results can be derived from (18).

First, we consider situations for which all the sample points $\{\vec{s}_k\}_{k=1}^{K}$ correspond to image points for the same planar surface in the scene. In this case the inverse depth values $p_k$ are given by a linear function of $\vec{s}_k$. That is, $p_k = \vec{c} \cdot \vec{s}_k$ for some constant 3-vector $\vec{c}$. Substitution of this expression for the inverse depths into (18), and use of $\vec{s}_k = (1/s_k)\vec{c}_k$, provides

$$\vec{r} = \sum_{k=1}^{K} c_k \left( (\vec{c} \cdot \vec{s}_k) (\vec{T}_0 \times \vec{s}_k) \right) = 0.$$

Notice that the term in square brackets is a quadratic polynomial in $\vec{s}_k$, so (15) ensures that $\vec{r}$ vanishes. Therefore our algorithm obtains no constraint on the translational direction from motion field samples corresponding to a single planar surface.

The second consequence of equation (18) is for sampling patterns $\{\vec{s}_k\}_{k=1}^{K}$ which span a small visual angle. In particular, if $\vec{s}_k$ is the mean direction of such a sampling pattern, then $\vec{r}$ is roughly in the direction $\vec{T}_0 \times \vec{s}_0$ (see (18)). Therefore the constraint $\vec{r} \cdot \vec{T} = 0$ simply forces $\vec{T}$ to be near the plane that contains both the true translational direction, $\vec{T}_0$, and the mean viewing direction $\vec{s}_0$. (Similar results have been obtained by Maybank [11], and later by ourselves [7], for the nonlinear constraints on $\vec{T}$ provided in (1).)

In order to recover $\vec{T}_0$ robustly we require at least two such constraint planes, intersecting at a sufficiently large angle. This might be accomplished, for example, using two sampling patterns, each concentrated around significantly different viewing directions, and each bounded away from the true translational direction. The only additional requirement for stability is that $\vec{r}$ can be accurately computed given noise in the optical flow data. That is, the "signal" components of the optical flow observations should not be dominated by noise. A rich depth structure and a small rotational velocity are sufficient to ensure reasonable signal to noise properties of the computed translation constraint vectors (for more details see [8]).

6 Implementation and Results.

Here we consider the convolution algorithm described in section 2 and show examples of its performance. The algorithm is clearly simple enough to run quickly with currently available hardware. In order for this to be a practical approach it is critical that it is robust in the presence of noisy optical flow measurements. In this section we therefore concentrate on the results of a preliminary study of the algorithm's behaviour in the presence of noisy input data.

Due to the ease in which we can obtain well controlled test data, we consider only computer generated data in this paper. Clearly for the results on
such sequences to generalize to real sequences we need to model the noise properties of optical flow measurement techniques. Fleet and Jepson [2], for example, report optical flow measurements with roughly isotropic, Gaussianly distributed, errors having a magnitude about 5% of the length of the optical flow vector. These error results were also obtained from simulated scenes, and therefore should be taken as a lower bound on the sorts of errors we can expect in practice.

The optical flow data we use is generated from the depth map for the computer generated scene shown in Figure 1. The range of projected distances along the optical axis (i.e. the range of values in the "Z-buffer") is roughly a factor of two for this scene. Since we expect the accuracy of the results to depend on the visual angle, we allow the focal distance to vary and keep the Z-buffer fixed. In Figure 2 we show a motion field generated using this Z-buffer, a 60 degree field of view, and a translational velocity of \(\begin{pmatrix} 1, 0, 1 \end{pmatrix}\) (towards and up), with fixation point in center of image.

Figure 2: Flow field for translational velocity \(\begin{pmatrix} 1, 0, 1 \end{pmatrix}\) (towards and up), with fixation point in center of image.

The convolution mask was constructed by modifying a DOG (Difference of Gaussians) mask so that it satisfied (15). The DOG parameters were a center standard deviation of 3 pixels, and a surround standard deviation of 6 pixels. A 31 \(\times\) 31 set of filter taps was used. The DOG coefficients were modified by adding a particular quadratic polynomial which was windowed by the surround Gaussian (i.e. a Hermite polynomial). The particular Hermite polynomial was chosen to ensure (15). The original DOG required only a small modification.

Figure 4 shows the norm of \(\mathcal{F}\) generated by the algorithm. This norm is essentially the absolute value of the response of the DOG mask when applied to the inverse depth data with a general modulation by the length of \(\mathbf{z} \times \hat{T}\) (cf. equation (18)). As we discussed in the previous section, the translation constraint vector \(\mathcal{F}(\mathbf{z})\) has a large amplitude only where there is significant variation in the depth structure. The amplitude response could clearly be useful for other operations such as image segmentation.

For the 60 degree field of view (measured along the horizontal center line), and 10% input error, the recov-
Figure 4: Only the translational component of the flow field shown in Figure 2.

The recovered translational direction was \((0.65, -0.8e^{-5}, 0.76)\). This represents an error of about 8% in the recovered translational direction, or about 5 degrees. Twenty separate runs were done (each taking less than a minute on a Silicon Graphics 4D/340VGX), and the standard deviation of the error measure in degrees was found to be less than a degree. In particular, we found that the method has a small bias (roughly 5 degrees) in the recovered translational direction towards the orientation of the optical axis. For a 90 degree field of view, but the same motion parameters and Z-buffer, we found this bias was reduced to 1.4 degrees (with a similar standard deviation across runs).

For a 20 degree field of view the recovered translational direction was strongly skewed towards the optical axis, with the average recovered translational direction being \((0.39, 0.2e^{-3}, 0.92)\). This represents a 22 degree error in the recovered translational direction, which was again fairly consistent across several runs with different noise samples. The variation in the error was again only about a degree.

These biases probably result from the result, noted in section 5, that the \(v\) vectors are all nearly perpendicular to the mean sampling direction for their image patch. As the field of view becomes smaller, the range of mean sampling directions also decreases, and therefore motion along the optical axis is more weakly constrained. Despite this bias along the optical axis, the projection of the translational velocity perpendicular to the optical axis was always recovered to less than a degree.

In conclusion, the algorithm shows considerable promise. It appears stable with respect to random errors in the optical flow, especially for larger fields of view. The component of the translational direction perpendicular to the optical axis appears to be stable even for narrow fields of view.

Acknowledgements

Thanks to NSERC Canada, IRIS and NASA for financial support. Also, thanks to David Fleet for helpful suggestions on the manuscript.

References