A General Motion Model and Spatio-Temporal Filters for Computing Optical Flow

Hongche Liu[†‡], Tsai-Hong Hong[‡], Martin Herman[‡], and Rama Chellappa[†]

[‡]Intelligent Systems Division, National Institute of Standards and Technology (NIST)
Blg. 220, Rm B124, Gaithersburg, MD 20899

[†]Center for Automation Research/Department of Electrical Engineering,
University of Maryland, College Park 20742

Abstract

Traditional optical flow algorithms assume local image translational motion and apply simple image filtering. Recent studies have taken two separate approaches toward improving the accuracy of computed flow: the application of spatio-temporal filtering schemes and the use of generalized motion models such as the affine model. Each has achieved some improvement over traditional algorithms in specialized situations. In this paper, we analyze the interdependency between them and propose a unified approach. The general motion model we adopt characterizes arbitrary 3-D steady motion. Under perspective projection, we derive an image motion equation that describes the spatio-temporal relation of gray-scale intensity in an image sequence, thus making the utilization of 3-D filtering possible. However, to accommodate this complex motion, we need to extend the filter design to derive additional motion constraint equations. Using Hermite polynomials, we design differentiation filters, whose orthogonality and Gaussian derivative properties insure numerical stability. The resulting algorithm produces accurate optical flow and other useful motion parameters. It is evaluated quantitatively using the scheme established by Barron, et al.[4] and qualitatively with real images.

1. Introduction

This paper describes an algorithm and supporting experimental results for accurate optical flow and motion estimation.

Research in the field of optical flow, starting from Gibson[12] , has spawned many algorithms in the past two decades, and at the same time has led to numerous applications. To name a few, optical flow can be used to compute three-dimensional motion and structure[2] [43] ; to locate the focus of expansion [15] or a moving observer's direction of heading; to segment independently moving objects[2] [28] ; to detect motion[9] ; to stabilize images[6] ; to perform obstacle detection and avoidance[33] [45] [46] [47] ; and to analyze medical video (2D echocardiographic images) to assist in diagnosis[8] . All of these applications use optical flow data in a quantitative way. Although it is true that the optical flow field is not necessarily equal to the motion field[38] , relative accuracy in optical flow is very important in obtaining qualitative properties of motion. For example, discontinuities in optical flow are useful qualitative properties that can be used to locate motion boundaries more precisely if the flow is more accurate.

Evidently, the importance of accurate optical flow can not be overemphasized. In view of this, Barron, Fleet, and Beauchemin[4] developed a scheme for evaluating optical flow algorithms, highlighting the current endeavor to achieve greater accuracy.

However, attempts to obtain accurate motion estimates are impeded by three sources of error: sensor noise, brightness change over time, and quantization error. Sensor noise is caused by optical or electronic irregularities. Brightness change can occur in many situations, including changing light sources, shadowing, camera aperture adjustment, or shading of a Lambertian or specular surface. Quantization error is inherent in digital images. These factors represent physical challenges that cannot be overcome by image processing alone but can be mitigated by filtering techniques. In addition to dealing with these physical errors, are there other ways of improving the current best optical flow algorithm? To answer this, we need to review other systematic difficulties that have been facing us.

The first difficulty is the aperture problem or the ill-posed nature of the flow computation problem. Traditional optical flow algorithms have worked on finding reasonable constraints to solve this problem [3,20,27,32,34,37]. Although many ideas were proposed, the desired accuracy was not achieved due to two factors: lack of attention to better filtering schemes and the use of simple assumption of uniform translational motion. A good filtering scheme is essential in dealing with the aforementioned sources of error, and uniform translation is insufficient for describing general 3-D motion.

Recent studies have taken two separate approaches to improving accuracy. The first is the application of spatio-temporal filtering schemes [11,19,25,35,41]. The second is the use of generalized motion models [5,7,8,17,31,35,39,42] such as the affine model. The fact that these two approaches are actually complementary to each other will become clear as we analyze their individual advantages and disadvantages.

An intuitive idea for achieving better accuracy when applying a filter is to increase its support. A large support alleviates the aperture problem, smooths out more noise, avoids aliasing, and reduces quantization error and truncation error of the filter. For example, to estimate temporal derivatives, recent research has used multiple frames instead of simple successive temporal differences. In fact, more sophisticated spatio-temporal filters, i.e., 3-D filters (Fig 1.2), have been developed to estimate image properties.

However, careless increase in filter support is not adequate due to the second difficulty commonly experienced: lack of a good motion model. Since traditional algorithms use simple filtering schemes and small filter supports, they can safely assume uniform translational motion as described in the image motion equation

$$I(x, y, t) = F(x-ut, y-vt) \quad . \tag{1}$$

Once the spatio-temporal filters are applied and the support increases, the motion within becomes more complicated. Moreover, if we consider perspective projection of the 3-D motion

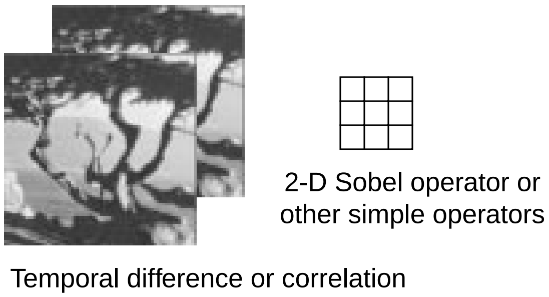


Fig 1. 1 Traditional filtering approach

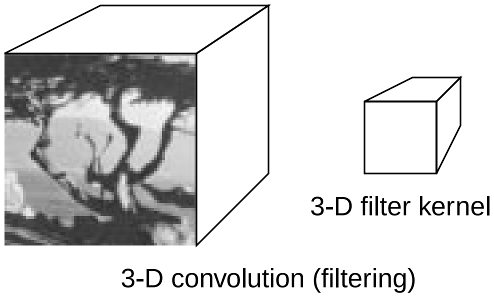


Fig 1.2 Spatio-temporal filtering approach

onto the 2-D image plane, the problem gets worse. For example, a forward moving observer sees a diverging scene in which a patch can undergo both translation and expansion* (Fig 2). Generally, divergence, curl, and deformation as well as translation exist in 2-D image motion. Unless the motion model accommodates all these motion parameters, there is a limit to the useful filter support. Recent research has proposed the affine motion model to cope with this difficulty. However, once the general motion model is derived, one realizes that it may not necessarily improve accuracy because of the new demand of obtaining additional parameters. Even more sophisticated filtering techniques are required to compute additional image properties, for example, one may use higher order derivatives to compute divergence, curl, and deformation [24] .

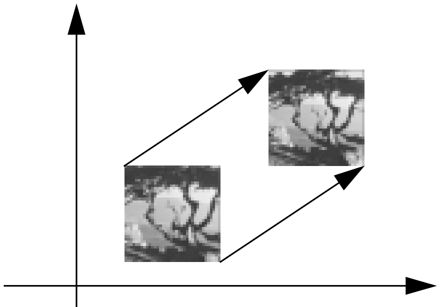


Fig 2. 1 Translation only

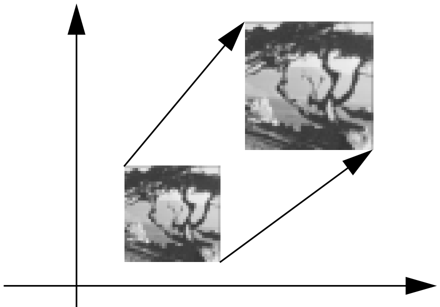


Fig 2. 2 Translation plus expansion

The above two approaches (spatio-temporal filtering and generalized motion model) have achieved a certain degree of improvement over traditional algorithms. Interested readers may refer to Section 5 for more details about these approaches and for comparisons. Nonetheless, the interdependencies between them still set a limit to their accuracy. To answer the question posed earlier: Yes, we can improve on the current optical flow algorithms by unifying a general motion model and a spatio-temporal filtering scheme.

A general image motion model based on 3-D relative motion has been studied [26] . However, we need to extend the *instantaneous* motion model so as to describe *continuous* motion because of the intrinsic requirement of spatio-temporal filtering. The continuous motion model is actually a 4-D model that involves X, Y, Z, t. An image motion equation based on this model is not tractable, especially considering the non-linearity imposed by perspective projection, unless we make a small motion approximation. It is then clear that we need a potent spatio-temporal filter design to solve the problem.

* A patch centered at the focus of expansion has expansion only.

The spatio-temporal filtering scheme we use is based on 3-D Hermite polynomial differentiation filters [18] [25] , which possess several advantages: the orthogonality and Gaussian derivative properties of the filters insure numerical stability; the approach is generalizable to the higher order derivatives we desire; these two properties make possible the coherent application of multiple filters. Numerous physiological models[14,48] also support the theory that the visual receptive field can be modeled by Gaussian derivatives of various widths.

The pursuit of higher accuracy is not complete until we overcome the third difficulty: occlusion or motion boundary effects. That is where accretion or deletion occurs[29] , and the information available to solve the problem is reduced. This difficulty is also common to other vision problems such as stereo matching. However, this issue is beyond the scope of this paper and will be investigated in a future study.

The ultimate goal of this research is to develop a flexible set of algorithms that deals with arbitrary 3-D relative motion and computes accurate optical flow for such applications as obstacle detection or motion segmentation. Our method is not only capable of unifying the two approaches attempted by recent research but also results in algorithms whose output is adequate for many motion applications. Its competitive performance is demonstrated using the evaluation framework established by Barron, et al.[4] .

The remainder of the paper is organized as follows: In Section 2, we present a general motion model for arbitrary 3-D motion and derive an image motion equation. Section 3 introduces a spatio-temporal filtering scheme using 3-D Hermite polynomial differentiation filters, and applies them to the motion estimation problem based on the image motion equation derived previously. Section 4 provides implementation details with attention to numerical stability and algorithm flexibility. Comparisons to previous work and specific contributions of this paper are summarized in Section 5. Section 6 details the results of quantitative evaluation of our algorithm in comparison with existing algorithms cited in Barron, et al. [4] . It also includes noise sensitivity analysis, which was not addressed by Barron, et al. In Section 7 we present a qualitative evaluation of the results of our algorithm using real images in a real motion application. Section 8 concludes the paper.

2. The General Motion Model

In this section, we describe how the local optical flow pattern reflects arbitrary 3-D motion and use this knowledge to derive a general motion model and an image motion equation. Rather than considering *instantaneous* velocity[26] , we consider velocity over time for the sake of spatio-temporal filtering. Let a 3-D point $\vec{P} = (X, Y, Z)^T$ * undergo steady small rotation $(\Omega_X, \Omega_Y, \Omega_Z)$ and translation (T_X, T_Y, T_Z) per unit time. Previous research that deals with generalizing the motion model has used a two-frame strategy as in [10] , which may be formulated as

$$\begin{bmatrix} X' \\ Y' \\ Z' \end{bmatrix} = R \begin{bmatrix} X \\ Y \\ Z \end{bmatrix} + T, \text{ where } R = \begin{bmatrix} 1 & -\Omega_Z & \Omega_Y \\ \Omega_Z & 1 & -\Omega_X \\ -\Omega_Y & \Omega_X & 1 \end{bmatrix} \text{ and } T = \begin{bmatrix} T_X \\ T_Y \\ T_Z \end{bmatrix}. \quad (2)$$

* In an observer-centered coordinate frame; Z is the axis along the line of sight.

Equivalently, we write

$$\begin{bmatrix} X' \\ Y' \\ Z' \\ 1 \end{bmatrix} = M \begin{bmatrix} X \\ Y \\ Z \\ 1 \end{bmatrix}, \text{ where } M = \begin{bmatrix} 1 & -\Omega_Z & \Omega_Y & T_X \\ \Omega_Z & 1 & -\Omega_X & T_Y \\ -\Omega_Y & \Omega_X & 1 & T_Z \\ 0 & 0 & 0 & 1 \end{bmatrix} \text{ is the 3-D motion transformation matrix.} \quad (3)$$

The locus of a 3-D point $\vec{P}(t) = (X(t), Y(t), Z(t))^T$ can then be described by

$$\begin{bmatrix} \vec{P}(t) \\ 1 \end{bmatrix} = MM\ldots M \begin{bmatrix} \vec{P}(0) \\ 1 \end{bmatrix} = M^t \begin{bmatrix} \vec{P}(0) \\ 1 \end{bmatrix} . \quad (4)$$

M^t is a polynomial of matrices. If M were diagonalizable, M^t would be easily computed as $S\Lambda^t S^{-1}$ [21], where Λ is the diagonal matrix composed of the eigenvalues of M and S is the matrix of column eigenvectors. However, M has two identical eigenvalues and is not diagonalizable. Fortunately, M has a *Jordan Canonical Form* SJS^{-1} [36] from which M^t can be computed as SJ^tS^{-1}, where J^t has the analytical form [21]

$$J = \begin{bmatrix} 1 & 1 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1-\Omega i & 0 \\ 0 & 0 & 0 & 1+\Omega i \end{bmatrix} \text{ where } \Omega = \sqrt{\Omega_X^2 + \Omega_Y^2 + \Omega_Z^2} \text{ and } i = \sqrt{-1} . \quad (5)$$

$$\text{Hence, } J^t = \begin{bmatrix} 1 & t & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & (1-\Omega i)^t & 0 \\ 0 & 0 & 0 & (1+\Omega i)^t \end{bmatrix} \approx \begin{bmatrix} 1 & t & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1-t\Omega i & 0 \\ 0 & 0 & 0 & 1+t\Omega i \end{bmatrix} \text{ when } \Omega \ll 1 . \quad (6)$$

The assumption of small rotation, $\Omega \ll 1$, is also used in [10] and most other later studies. Then,

$$M^t = SJ^tS^{-1} \approx \begin{bmatrix} 1 & -t\Omega_Z & t\Omega_Y & ta_X + b_X \\ t\Omega_Z & 1 & -t\Omega_X & ta_Y + b_Y \\ -t\Omega_Y & t\Omega_X & 1 & ta_Z + b_Z \\ 0 & 0 & 0 & 1 \end{bmatrix} = \begin{bmatrix} 1 & -t\Omega_Z & t\Omega_Y & ta_X \\ t\Omega_Z & 1 & -t\Omega_X & ta_Y \\ -t\Omega_Y & t\Omega_X & 1 & ta_Z \\ 0 & 0 & 0 & 1 \end{bmatrix}. \quad (7)$$

where each of a_X, a_Y, a_Z is a function of all of $(\Omega_X, \Omega_Y, \Omega_Z, T_X, T_Y, T_Z)$. The last equality comes from the intuition that $M^0 = I$, therefore b_X, b_Y, b_Z must be the residual error from the approximation in (6) and should be eliminated here. We may regard a_X, a_Y, a_Z as translations in the presence of rotation per unit time.

The locus $\vec{P}(t) = (X(t), Y(t), Z(t))^T$ projects to a point in the 2-D image plane, $(x(t), y(t))$, where

$$\begin{cases} x(t) = fX(t)/Z(t) \\ y(t) = fY(t)/Z(t) \end{cases} \text{, where } f \text{ is the focal length. Let } \begin{cases} x_0 = fX/Z \\ y_0 = fY/Z \end{cases} \text{. Then} \tag{8}$$

$$\begin{aligned} x(t) &= \frac{f(ta_X + X - t\Omega_Z Y + t\Omega_Y Z)}{ta_Z - t\Omega_Y X + t\Omega_X Y + Z} = \frac{f(ta_X f/Z + x_0 - t\Omega_Z y_0 + t\Omega_Y f}{ta_Z f/Z - t\Omega_Y x_0 + t\Omega_X y_0 + f} \\ y(t) &= \frac{f(ta_Y + t\Omega_Z X + Y - t\Omega_X Z)}{ta_Z - t\Omega_Y X + t\Omega_X Y + Z} = \frac{f(ta_Y f/Z + t\Omega_Z x_0 + y_0 - t\Omega_X f}{ta_Z f/Z - t\Omega_Y x_0 + t\Omega_X y_0 + f} \end{aligned}. \tag{9}$$

Note that an instantaneous velocity derived in [26] is a special case of our formulation, namely, the velocity (u, v) at $t = 0$:

$$\begin{aligned} u &= \frac{\partial}{\partial t}x(t)\Big|_{t=0} = f\left(\frac{a_X}{Z} + \Omega_Y\right) - \left(f\frac{a_Z}{Z}x_0 - \Omega_Z y_0\right) + x_0\left(\frac{\Omega_Y}{f}x_0 - \frac{\Omega_X}{f}y_0\right) \\ v &= \frac{\partial}{\partial t}y(t)\Big|_{t=0} = f\left(\frac{a_Y}{Z} - \Omega_X\right) - \left(\Omega_Z x_0 - f\frac{a_Z}{Z}y_0\right) + y_0\left(\frac{\Omega_Y}{f}x_0 - \frac{\Omega_X}{f}y_0\right) \end{aligned}. \tag{10}$$

Note that the flow is generally quadratic. Computing optical flow based on the uniform translation model is far from adequate, and the affine motion model is only valid when there is no rotation in the X and Y directions.

To derive an image motion equation in the form of (1), we start with the brightness constancy equation:

$$I(x(t), y(t), t) = I(x_0, y_0, 0) \quad . \tag{11}$$

Without loss of generality, let $F(x_0, y_0) = I(x_0, y_0, 0)$. It suffices to find x_0, y_0 in terms of $x(t), y(t)$, which will be denoted by x, y for simplicity. The resulting solution is extremely complicated, but assuming small rotation and small 3-D translation relative to distance, namely, $a_X, a_Y, a_Z \ll Z$, we have

$$x_0 \approx \frac{x + t\left(\frac{a_Z}{Z}x + \Omega_Z y - f\left(\frac{a_X}{Z} + \Omega_Y\right)\right)}{1 + \frac{t}{f}(\Omega_Y x - \Omega_X y)} \qquad y_0 \approx \frac{y + t\left(-\Omega_Z x + \frac{a_Z}{Z}y - f\left(\frac{a_Y}{Z} - \Omega_X\right)\right)}{1 + \frac{t}{f}(\Omega_Y x - \Omega_X y)}. \tag{12}$$

Equation (12) can be further simplified by using the approximation $\frac{t}{f}\Omega_Y x, \frac{t}{f}\Omega_X y \ll 1$, as follows:

$$x_0 \approx \left(x + t\left(\frac{a_Z}{Z}x + \Omega_Z y - f\left(\frac{a_X}{Z} + \Omega_Y\right)\right)\right)\left(1 - \frac{t}{f}(\Omega_Y x - \Omega_X y)\right)$$
$$y_0 \approx \left(y + t\left(-\Omega_Z x + \frac{a_Z}{Z}y - f\left(\frac{a_Y}{Z} - \Omega_X\right)\right)\right)\left(1 - \frac{t}{f}(\Omega_Y x - \Omega_X y)\right), \qquad (13)$$

$$x_0 \approx x + t\left(\frac{a_Z}{Z}x + \Omega_Z y - f\left(\frac{a_X}{Z} + \Omega_Y\right)\right) - \frac{t}{f}(\Omega_Y x^2 - \Omega_X xy)$$
$$y_0 \approx y + t\left(-\Omega_Z x + \frac{a_Z}{Z}y - f\left(\frac{a_Y}{Z} - \Omega_X\right)\right) - \frac{t}{f}(\Omega_Y xy - \Omega_X y^2) \qquad (14)$$

The above approximation is justified by the following facts. First, the value of f in pixel units is usually large. For example, for a 256x256 image with a field of view of 45 degrees, f is 309 pixels. Second, since we are concerned with motion in a relatively small image local neighborhood, so-called pointwise analysis, x, y, t are small. Third, a small rotation in the X and Y direction in 3-D space can be approximated in the 2-D image plane as a simple translation. Since we are interested in finding optical flow rather than 3-D motion and structure, we do not lose much accuracy here. The error from this approximation will be absorbed by the translation parameters a_X, a_Y, thus offsetting the optical flow error. Inherent 3-D motion ambiguities related to this were described in [1] [44] . We will also use the above arguments for further simplification in our algorithm development (Section 4.2).

Now the image motion equation is, from (11) and (14),

$$I(x, y, t) = F(x + t(\alpha + \gamma x + \rho y + \delta x^2 + \varepsilon xy), y + t(\beta - \rho x + \gamma y + \delta xy + \varepsilon y^2)) \quad , \qquad (15)$$

where $\alpha = -f\left(\frac{a_X}{Z} + \Omega_Y\right), \beta = -f\left(\frac{a_Y}{Z} - \Omega_X\right), \gamma = \frac{a_Z}{Z}, \rho = \Omega_Z, \delta = -\frac{1}{f}\Omega_Y, \varepsilon = \frac{1}{f}\Omega_X$. (16)

We need to develop a filtering scheme to relate all the above motion parameters to the 3-D filter output and then solve them in order to estimate the optical flow, which is $(-\alpha, -\beta)$ at $x = y = t = 0$ from (10) and (16). Note that these motion parameters are closely related to 3-D motion.

To demonstrate the behavior of the image motion equation, consider the following basic motion patterns:

1. When there is no rotation, and no translation in the Z direction, then $\gamma = \rho = \delta = \varepsilon = 0$ (16), and there is uniform image translational motion, as assumed by traditional algorithms. (Fig 2.1)

1. When there is no rotation, $\rho = \delta = \varepsilon = 0$, hence the image motion is affine without

rotation, i.e. with only expansion and translation. (Fig 2.2)

1. When there is no translation in the Z direction, $\gamma = 0$ and no rotation in the X and Y direction, $\delta = \varepsilon = 0$, the image undergoes translation and rotation. (Fig 3.1)

1. When there is no rotation in the X and Y direction, $\delta = \varepsilon = 0$, the image undergoes affine motion without deformation, i.e. only with translation, expansion and rotation. (Fig 3.2)

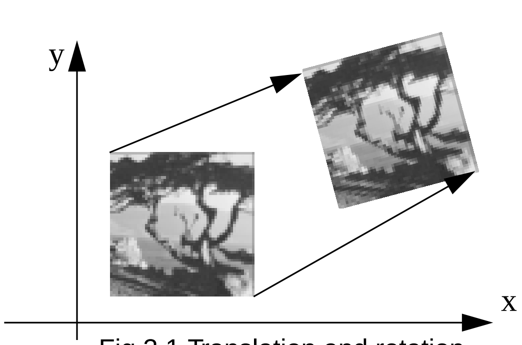


Fig 3.1 Translation and rotation

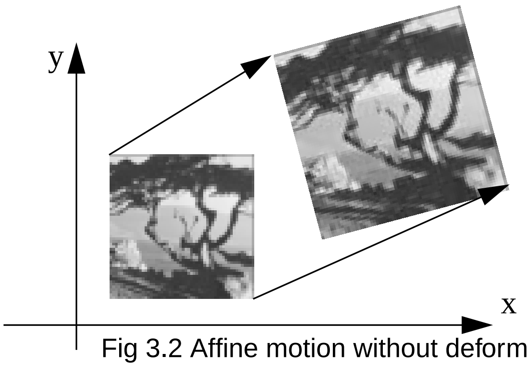


Fig 3.2 Affine motion without deformation

1. An arbitrary 3-D motion generates an image motion like Fig. 4.

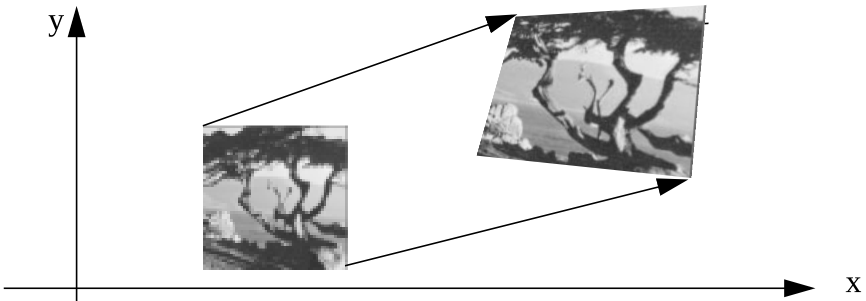


Fig 4. General image motion for arbitrary 3-D motion

Note that the images shown are merely enlarged local neighborhoods to reveal their particular behaviors under motion.

In summary, the image motion equation is based on expedient and reasonable approximations. It is applicable not only to the algorithm developed here, but also to other motion algorithms, although the extent of improvement depends on the particular algorithm. For the gradient-based method, the filtering process is the decisive factor as far as performance is concerned. We formulate the theory of Hermite polynomial differentiation filters in the next section.

3. Hermite Polynomial Filters

3.1 Hermite polynomials

The *n*th Hermite polynomial $H_n(x)$ is a solution of

$$\frac{d^2H_n}{dx^2} - 2x\frac{dH_n}{dx} + 2nH_n = 0. \tag{17}$$

The $H_n(x)$ are derived by Rodrigues' formula [18]

$$H_n(x) = (-1)^n e^{x^2}\frac{d^n}{dx^n}e^{-x^2} . \tag{18}$$

The computation of $H_n(x)$ is especially easy using the following recursive relations:

$$\begin{aligned} H_{n+1}(x) &= 2xH_n(x) - 2nH_{n-1}(x) \\ H_0(x) &= 1 \\ H_1(x) &= 2x \end{aligned} \tag{19}$$

By substituting $G(x)$ (with variance σ^2) for e^{-x^2} in (18), we generalize to Hermite polynomials with respect to the Gaussian function. Let these Hermite polynomials be denoted by $\overline{H}_n(x)$

$$\overline{H}_n(x) = (-1)^n G^{-1}(x)\frac{d^n}{dx^n}(G(x)) \tag{20}$$

Note that $\overline{H}_n(x)$ differs from $H_n(x)$ by a scaling product:

$$\overline{H}_n(x) = \left(\frac{1}{2^{1/2}\sigma}\right)^n H_n\left(\frac{x}{2^{1/2}\sigma}\right) \tag{21}$$

The scalar product of two functions and the L$_2$-norm of a function with $G(x)$ as a weight function are defined as follows:

$$\langle a, b\rangle \equiv \int_{-\infty}^{\infty} G(x)a(x)b(x)dx \quad \text{and} \quad \|a\| \equiv \langle a, a\rangle^{1/2}$$

The orthogonality of { $\overline{H}_n(x)$ } can be expressed in the following way[18] :

$$\langle \overline{H}_m, \overline{H}_n\rangle = \sigma^{-2n} n!\delta_{mn}, \tag{22}$$

The 3D case of Hermite polynomials is especially simple because they are separable. Thus the polynomial with order $n = i + j + k$ is

$$\overline{H}_{ijk}(x, y, t) = \overline{H}_i(x) \cdot \overline{H}_j(y) \cdot \overline{H}_k(t) \tag{23}$$

3.2 Derivation of gradient constraint equations

One of the most important properties of Hermite polynomials is the property of Gaussian derivatives. It is with the aid of this property that we are able to establish gradient constraint equations. This property is manifested in the following theorem.

Theorem 1: A one dimensional signal $I(x)$ can be expanded in terms of Hermite polynomials as

$$I(x) = \sum_{k=0}^{\infty} I_k \frac{\overline{H}_k(x)}{\|\overline{H}_k\|^2}$$

$$\text{Then } I_k = \langle I, \overline{H}_k \rangle = \langle I^{(k)}, \overline{H}_0 \rangle \text{ where } \overline{H}_0(x) = 1 \text{ and } I^{(k)} = \frac{d^k I}{dx^k}. \qquad (24)$$

The proof is given in Appendix A. This theorem states that the kth order Gaussian derivative of the image is the inner product of the image and the kth order Hermite polynomial $\overline{H}_k(x)$. Note that the theorem is true only for unnormalized Hermite polynomials. This fact is used when we assign weights to motion constraint equations of different orders in equation (35).

Recall our image motion equation (15),

$$I(x, y, t) = F(x + t(\alpha + \gamma x + \rho y + \delta x^2 + \varepsilon xy), y + t(\beta - \rho x + \gamma y + \delta xy + \varepsilon y^2)) \quad .$$

Expand both sides with Hermite polynomials,

$$\sum_{i=0}^{\infty} \sum_{j=0}^{\infty} \sum_{k=0}^{\infty} I_{ijk} \frac{\overline{H}_{ijk}}{\|\overline{H}_{ijk}\|^2} = \sum_{i=0}^{\infty} \sum_{j=0}^{\infty} \sum_{k=0}^{\infty} F_{ijk} \frac{\overline{H}_{ijk}}{\|\overline{H}_{ijk}\|^2} \text{ then } I_{ijk} = \langle I, \overline{H}_{ijk} \rangle = F_{ijk} = \langle F, \overline{H}_{ijk} \rangle \qquad (25)$$

Equating I_{ij1} to F_{ij1} and using Theorem 1, we derive

$$\begin{aligned} I_{ij1} &= F_{ij1} = \langle F, \overline{H}_{ij1} \rangle = \langle \frac{\partial F}{\partial t}, \overline{H}_{ij0} \rangle \\ &= \langle (\alpha + \gamma x + \rho y + \delta x^2 + \varepsilon xy) \frac{\partial F}{\partial x_0} + (\beta - \rho x + \gamma y + \delta xy + \varepsilon y^2) \frac{\partial F}{\partial y_0}, \overline{H}_{ij0} \rangle \\ &\approx \langle (\alpha + \gamma x + \lambda \rho + \delta x^2 + \varepsilon xy) \frac{\partial I}{\partial x}, \overline{H}_{ij0} \rangle + \langle (\beta - \rho x + \gamma y + \delta xy + \varepsilon y^2) \frac{\partial I}{\partial y}, \overline{H}_{ij0} \rangle \qquad (26) \end{aligned}$$

We make the above approximation because equations (9) and (16) allow us to derive

$$\frac{\partial F}{\partial x_0} = \frac{\partial I}{\partial x}\frac{\partial x}{\partial x_0} + \frac{\partial I}{\partial y}\frac{\partial y}{\partial x_0} = \frac{\partial I}{\partial x}(1 - \delta f x t) + \frac{\partial I}{\partial y}(\rho t - \delta f y t) \quad . \qquad (27)$$

Since $\left.\frac{\partial F}{\partial x_0}\right|_{x=0,\, y=0,\, t=0} = \left.\frac{\partial I}{\partial x}\right|_{x=0,\, y=0,\, t=0}$ and the inner product is Gaussian weighted, the error is not significant. Besides, without the approximation, the eventual constraint equation would be non-linear and very difficult to solve. The analysis is similar for $\frac{\partial F}{\partial y_0}$. Therefore, we arrive at

$$I_{ij1} \approx \langle(\alpha + \gamma x + \rho y + \delta x^2 + \varepsilon xy)\frac{\partial I}{\partial x}, \overline{H}_{ij0}\rangle + ((\beta - \rho x + \gamma y + \delta xy + \varepsilon y^2)\frac{\partial I}{\partial y}, \overline{H}_{ij0}) \ . \tag{28}$$

$$\begin{aligned} I_{ij1} \approx {} & \alpha\langle\frac{\partial I}{\partial x}, \overline{H}_{ij0}\rangle + \gamma\langle\frac{\partial I}{\partial x}, x\overline{H}_{ij0}\rangle + \rho\langle\frac{\partial I}{\partial x}, y\overline{H}_{ij0}\rangle + \delta\langle\frac{\partial I}{\partial x}, x^2\overline{H}_{ij0}\rangle + \varepsilon\langle\frac{\partial I}{\partial x}, xy\overline{H}_{ij0}\rangle \\ & + \beta\langle\frac{\partial I}{\partial y}, \overline{H}_{ij0}\rangle - \rho\langle\frac{\partial I}{\partial y}, x\overline{H}_{ij0}\rangle + \gamma\langle\frac{\partial I}{\partial y}, y\overline{H}_{ij0}\rangle + \delta\langle\frac{\partial I}{\partial y}, xy\overline{H}_{ij0}\rangle + \varepsilon\langle\frac{\partial I}{\partial y}, y^2\overline{H}_{ij0}\rangle \end{aligned} \tag{29}$$

To simplify (29), we derive from (19) and (21) the following equation

$$x\overline{H}_n(x) \ = \ \sigma^2\overline{H}_{n+1}(x) + n\overline{H}_{n-1}(x) \ \text{. Hence} \tag{30}$$

$$\begin{aligned} I_{ij1} \approx {} & \alpha\langle\frac{\partial I}{\partial x}, \overline{H}_{ij0}\rangle + \gamma\langle\frac{\partial I}{\partial x}, \sigma^2\overline{H}_{i+1j0} + i\overline{H}_{i-1j0}\rangle + \rho\langle\frac{\partial I}{\partial x}, \sigma^2\overline{H}_{ij+1,0} + j\overline{H}_{ij-1,0}\rangle \\ & + \delta\langle\frac{\partial I}{\partial x}, \sigma^4\overline{H}_{i+2j0} + (2i+1)\sigma^2\overline{H}_{ij0} + i(i+1)\overline{H}_{i-2j0}\rangle \\ & + \varepsilon\langle\frac{\partial I}{\partial x}, \sigma^4\overline{H}_{i+1j+1,0} + i\sigma^2\overline{H}_{i-1j+1,0} + j\sigma^2\overline{H}_{i+1j-1,0} + ij\overline{H}_{i-1j-1,0}\rangle \\ & + \beta\langle\frac{\partial I}{\partial y}, \overline{H}_{ij0}\rangle - \rho\langle\frac{\partial I}{\partial y}, \sigma^2\overline{H}_{i+1j0} + i\overline{H}_{i-1j0}\rangle + \gamma\langle\frac{\partial I}{\partial y}, \sigma^2\overline{H}_{ij+1,0} + j\overline{H}_{ij-1,0}\rangle \\ & + \delta\langle\frac{\partial I}{\partial y}, \sigma^4\overline{H}_{i+1j+1,0} + i\sigma^2\overline{H}_{i-1j+1,0} + j\sigma^2\overline{H}_{i+1j-1,0} + ij\overline{H}_{i-1j-1,0}\rangle \\ & + \varepsilon\langle\frac{\partial I}{\partial y}, \sigma^4\overline{H}_{ij+2,0} + (2j+1)\sigma^2\overline{H}_{ij0} + j(j+1)\overline{H}_{ij-2,0}\rangle \end{aligned} \tag{31}$$

Using Theorem 1, we simplify (31) to

$$\begin{aligned} I_{ij1} \approx {} & \alpha I_{i+1j0} + \gamma[\sigma^2 I_{i+2j0} + iI_{ij0}] \\ & + \rho[\sigma^2 I_{i+1j+1,0} + jI_{i+1j-1,0}] \\ & + \delta[\sigma^4 I_{i+3j0} + (2i+1)\sigma^2 I_{i+1j0} + i(i+1)I_{i-1j0}] \\ & + \varepsilon[\sigma^4 I_{i+2j+1,0} + i\sigma^2 I_{ij+1,0} + j\sigma^2 I_{i+2j-1,0} + ijI_{ij-1,0}] \\ & + \beta I_{ij+1,0} - \rho[\sigma^2 I_{i+1j+1,0} + iI_{i-1j+1,0}] + \gamma[\sigma^2 I_{ij+2,0} + jI_{ij0}] \\ & + \delta[\sigma^4 I_{i+1j+2,0} + i\sigma^2 I_{i-1j+2,0} + j\sigma^2 I_{i+1j0} + ijI_{i-1j0}] \\ & [\sigma^4 I \quad (2j+1)\sigma^2 I \quad j(j+1)I \quad] \end{aligned} \tag{32}$$

As stated in the introduction, this is a fundamental element resulting in a coherent spatio-temporal filtering scheme to compute optical flow. This capability stems from two nice properties of (32). The first is the linearity of the equation in terms of the motion parameters as defined in (16); the second is its extensibility to higher orders, i.e., the values of i, j can be as large as required by the number of parameters. Thus to solve for the motion parameters and then for optical flow, we derive a system of linear equations with coefficients computed from spatio-temporal filters I_{ijk}. These result in excellent numerical stability due to the orthogonality of the Hermite polynomial differentiation filters and their inherent Gaussian smoothing. Although (32) appears to be complicated, it in fact suggests a simple, local, and parallel algorithm, which involves only convolutions and solving a linear system, as presented in the next section.

4. Algorithms for Computing Optical Flow

Equation (32) gives rise to not a specific algorithm but a set of algorithms, due to its extensibility. We can derive the same number of equations as unknowns and solve a linear system, or we may incorporate more equations of higher order and solve a linear least square problem. On the other hand, if we possess knowledge about the input image sequence, for example, that there are no rotations around certain directions, the number of motion unknowns can be reduced. We also make other expedient choices based on numerical considerations and experimental findings. All these options are explored in the following subsections.

4.1 The general algorithm

According to (32), we derive six equations up to the third order. Within a 3-D local window, we estimate $\{I_{ijk}\}$ with the discrete approximation $\{\hat{I}_{ijk}(x, y, t)\}$, that is, the 3-D convolution of the *sampled* Hermite polynomial filters with the image sequence, and write the equations in matrix vector form:

$$M_6 s = c\text{, where } M_6 = (M_4 \oplus M_2) \text{ where } \oplus \text{ means concatenation, and} \qquad (33)$$

$$s = \begin{bmatrix} \alpha \\ \beta \\ \gamma \\ \rho \\ \delta \\ \varepsilon \end{bmatrix}, c = \begin{bmatrix} \hat{I}_{001} \\ \hat{I}_{101} \\ \hat{I}_{011} \\ \hat{I}_{201} \\ \hat{I}_{111} \\ \hat{I}_{021} \end{bmatrix}, M_4 = \begin{bmatrix} \hat{I}_{100} & \hat{I}_{010} & \sigma^2(\hat{I}_{200} + \hat{I}_{020}) & 0 \\ \hat{I}_{200} & \hat{I}_{110} & \sigma^2(\hat{I}_{300} + \hat{I}_{120}) + \hat{I}_{100} & -\hat{I}_{010} \\ \hat{I}_{110} & \hat{I}_{020} & \sigma^2(\hat{I}_{210} + \hat{I}_{030}) + \hat{I}_{010} & \hat{I}_{100} \\ \hat{I}_{300} & \hat{I}_{210} & \sigma^2(\hat{I}_{400} + \hat{I}_{220}) + 2\hat{I}_{200} & -2\hat{I}_{110} \\ \hat{I}_{210} & \hat{I}_{120} & \sigma^2(\hat{I}_{310} + \hat{I}_{130}) + 2\hat{I}_{110} & \hat{I}_{200} - \hat{I}_{020} \\ \hat{I}_{120} & \hat{I}_{030} & \sigma^2(\hat{I}_{220} + \hat{I}_{040}) + 2\hat{I}_{020} & 2\hat{I}_{110} \end{bmatrix} \qquad (34)$$

$$M_2 = \begin{bmatrix} \sigma^4(\hat{I}_{300} + \hat{I}_{120}) + \sigma^2\hat{I}_{100} & \sigma^4(\hat{I}_{210} + \hat{I}_{030}) + \sigma^2\hat{I}_{010} \\ \sigma^4(\hat{I}_{400} + \hat{I}_{220}) + \sigma^2(3\hat{I}_{200} + \hat{I}_{020}) + 2\hat{I}_{000} & \sigma^4(\hat{I}_{310} + \hat{I}_{130}) + \sigma^2 2\hat{I}_{110} \\ \sigma^4(\hat{I}_{310} + \hat{I}_{130}) + \sigma^2 2\hat{I}_{110} & \sigma^4(\hat{I}_{220} + \hat{I}_{040}) + \sigma^2(3\hat{I}_{020} + \hat{I}_{200}) + 2\hat{I}_{000} \\ \sigma^4(\hat{I}_{500} + \hat{I}_{320}) + \sigma^2(5\hat{I}_{300} + 2\hat{I}_{120}) + 6\hat{I}_{100} & \sigma^4(\hat{I}_{410} + \hat{I}_{230}) + \sigma^2 3\hat{I}_{210} \\ \sigma^4(\hat{I}_{410} + \hat{I}_{230}) + \sigma^2(4\hat{I}_{210} + \hat{I}_{030}) + 3\hat{I}_{010} & \sigma^4(\hat{I}_{320} + \hat{I}_{140}) + \sigma^2(4\hat{I}_{120} + \hat{I}_{300}) + 3\hat{I}_{100} \\ \sigma^4(\hat{I}_{320} + \hat{I}_{050}) + \sigma^2 3\hat{I}_{120} & \sigma^4(\hat{I}_{230} + \hat{I}_{050}) + \sigma^2(5\hat{I}_{030} + 2\hat{I}_{210}) + 6\hat{I}_{010} \end{bmatrix}$$

Note that filter outputs of up to fifth order are used. s can be solved exactly by (33) for the center pixel of the 3-D window and optical flow at the center of the local window is $(-\alpha, -\beta)$.

4.2 Specialized algorithms

The algorithm presented in the previous subsection, i.e., solving a full-rank linear system, is often an overkill for many image sequences. There are several disadvantages in using it for simple motion: First, it requires the use of higher order filters, whose orthogonality is always distorted in a limited local support; and the use of a large local support is undermined by its susceptibility to motion boundary effects. Second, the linear system is often highly ill-conditioned, since the columns of the matrix are of different orders of magnitude. Third is the higher demand in computation.

Let the focal length be large enough and/or the rotation in the X, Y direction small enough for δ and ε to be negligible. Then $M_2 = 0$ and we have a linear least square problem:

$E = min\|A_4 s_4 - b\|$ where $A_4 = WM_4$, $b = Wc$, and $W = Diag[w_1, w_2, w_3, w_4, w_5, w_6]$.(35)

W is the diagonal weight matrix for the motion constraint equations. According to (25), we use

$$w_1 = \|\bar{H}_{001}\|^{-2}, w_2 = \|\bar{H}_{101}\|^{-2}, w_3 = \|\bar{H}_{011}\|^{-2}, w_4 = \|\bar{H}_{201}\|^{-2}, w_5 = \|\bar{H}_{111}\|^{-2}, w_6 = \|\bar{H}_{021}\|^{-2}$$

Note that this formulation reduces the highest order filter to fourth order. In addition, a least square solution is more stable than a full-rank linear system. The applicability of the weight matrix is another nice feature. We suggest solving (35) by QR decomposition:

$$A_4 = QR\text{, and } E = min\|QRs + b\| = min\|Rs + Q^H b\|\text{, where } Q \text{ is unitary.} \qquad (36)$$

R can be denoted by $\begin{bmatrix} R_s \\ 0 \end{bmatrix}$, where R_s is an upper triangular matrix; and $Q^H b$ is $\begin{bmatrix} b_s \\ b_r \end{bmatrix}$, correspondingly. Equation (36) then becomes

$$\begin{aligned} E &= min(\|R_s s + b_s\| + \|b_r\|) \\ &= \|b_r\| = r \qquad \text{if } R_s \text{ is not singular.} \end{aligned} \qquad (37)$$

$$\text{The solution } s \text{ is computed from } R_s s + b_s = 0 \qquad (38)$$

For all practical purposes, the above algorithm is adequate for computing optical flow. Nonetheless, for many synthetic images or synthesized real images [4] , there is still room to simplify the algorithms and improve the stability of the linear system. For example, in image sequences where $\Omega_Z = 0$, we get $\rho = 0$ and (35) reduces to

$$E = min\|A_3 s_3 - b\|, \qquad (39)$$

where A_3 and s_3 are the first three columns and elements of A_4 and s_4, respectively.

Furthermore, the third column in A_3 involves higher order terms plus a lower order term. Experiments suggest that the lower order term is always dominant and more accurate. Neglecting the higher order terms does not necessarily degrade the accuracy but does save a great deal of computing time. This finding is very crucial especially for real-time implementations.

In terms of computing efficiency, our algorithm is excellent due to the separability of the 3-D Hermite polynomial filter design. Let the 3-D filter size be $W_x \times W_y \times W_t$ and image size be S. The complexity of the computation is $O((W_x + W_y + W_t + C)S)$, instead of $O((W_x W_y W_t + C)S)$, where C is the constant factor associated with solving the linear system. In addition, the above process can run on all image pixels in a parallel fashion. It can achieve maximum speedup running on a CREW (Concurrent Read Exclusive Write) parallel machine [22] .

The advantages of the QR decomposition are the availability of the matrix R_s and the residual. The behavior of R_s and the residual reveal plenty of information about the underlying images and motions. There are certain situations where the optical flow cannot be reliably computed from local information due to, for example, the aperture problem. Therefore, R_s and the residual should be investigated for its possible link to the reliability of the computed optical flow. We devote the next subsection to the discussion of the optic flow errors and confidence measures.

4.3 Confidence measures

Our algorithm provides ample information about the behavior of the system equations. It includes the residual r, the condition number and the determinant of R_s. They can be shown [25] to signify certain image phenomena, e.g., occlusion, the aperture problem, etc., which present difficulties for optical flow computation. Therefore, they can be utilized to locate high error areas and suggest subsequent improvement methods. For the sake of the evaluation in Section 6, we simply use them as confidence measures or threshold values to extract more accurate data, notwithstanding the fact that they can be used for qualitative image analysis.

4.3.1 Residual

The residual of our algorithm is $\|A_n s_n + b\|$ or r (=E) (37). The residual of an overdetermined linear system indicates the degree to which the equations disagree with one another. The reason for the existence of the residual lies in the approximation error of $\{ \hat{I}_{ijk}(x, y, t) \}$. A high approximation error may indicate one of three problems:

1. The assumption of the motion model is violated in the 3-D window V. It is possible that the window covers more than one moving object. Occlusion or multiple independently moving objects in a window can cause this problem.

1. The assumption of constant image brightness is violated. It is not unusual for the brightness of an object to change when the viewing angle changes due to relative

motion. In addition, the observing camera may adjust the brightness gain for different scenes, resulting in a change of object brightness. Similar effects can be caused by the shadow of another object.

1. Quantization and truncation errors. Quantization errors result from sampling Hermite polynomial filters. Truncation errors are introduced when we use limited spatial support to compute $\{ \hat{I}_{ijk}(x, y, t) \}$. Within a small window, the Hermite polynomials are no longer orthogonal and the derivatives computed are not accurate. This situation is worse for higher order differentiation filters. For example, $\langle \overline{H}_n, \overline{H}_n \rangle / \sigma^{-2n} n!$ (22) is 0.93 when $n = 5$ and 0.999945 when $n = 1$ for a window size of 21.

We can model the above errors as perturbations to the linear system [25] :

$$\tilde{E} = min\|(A_n + N)\tilde{s} + (b + \Delta b)\| \text{ , where } N \text{ and } \Delta b \text{ denote errors and } n < 6 . \qquad (40)$$

We prove in Appendix B the following analytical results:

$$\Delta s = \tilde{s} - s \approx -(A_n{}^T A_n)^{-1} A_n^T (Ns + \Delta b) \qquad (41)$$

$$\tilde{r} = \tilde{E} \approx \left\|(I - A_n(A_n{}^T A_n)^{-1} A_n{}^T)(Ns + \Delta b)\right\| \qquad (42)$$

Note that the expressions for both optical flow error Δs (41) and residual $\tilde{r}$ (42) are proportional to the size of the noise vector $(Ns + \Delta b)$ *. It is evident that locations with high residual reflect large errors and inaccurate optical flow values.

Note that the three problems mentioned above suggest contradictory choices for the window size. With larger windows, problems 1 and 2 may be aggravated; with smaller windows, problem 3 becomes worse.

4.3.2 Condition Number and Determinant

The condition number of R_s, denoted by $\kappa(R_s)$, is defined as $\|R_s\| \|R_s^{-1}\|$ and can be shown to be $\frac{|\lambda|_{max}}{|\lambda|_{min}}$, where the λ 's are eigenvalues or diagonal elements of R_s.

A condition number measures the extent to which a linear system maps an input error into an output error, or in brief, the numerical instability of the system. If s contains errors magnified by an ill-conditioned A_n from errors in b, it is not reliable. Since matrix A_n is concerned with the image texture only and not with motion, we find correspondences between a high condition number and the following two image neighborhood situations:

1. when there is a steep edge in the *x*(*y*) direction (Fig 5.1), so that the derivatives are

* It may be deceptive to claim that the residual is proportional to the optical flow error because the error vector is mapped by different matrices ($(A_n{}^T A_n)^{-1} A_n{}^T$, $I - A_n(A_n{}^T A_n)^{-1} A_n{}^T$), so the error also depends on the orientation of the noise with respect to the matrices.

very large for $x(y)$ and small for $y(x)$;

1. when there is a lack of texture in a direction (the $x + y$ direction in Fig 5.2), so that the derivatives in the x direction are approximately proportional to the derivatives in the y direction, i.e. $I(x, y) \approx I(kx + y)$.

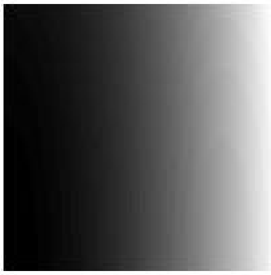


Fig 5.1 Smoothed steep edge.

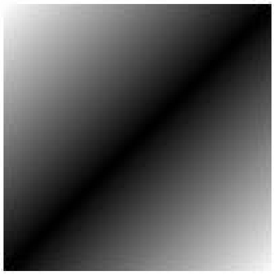


Fig 5.2 Lack of texture in $x+y$ direction.

The above two situations can easily be confirmed by inspecting the QR decomposition process.

The determinant of R_s is the product of all its eigenvalues. In solving (38) or $s = -R_s^{-1} \cdot b_s$, the determinant plays an important role in the matrix inverse. Since we use the QR decomposition method, Q is unitary (orthonormal projection) so the behavior of R_s is similar to the original A_n. A small determinant of R_s indicates one of the following two situations:

1. The three columns of A_n are close to being linearly dependent. This is the same as the second situation in the above discussion of condition number. In fact, a small determinant due to linear dependency also causes a high condition number.

1. All the elements of A_n are very small. This corresponds to a uniform brightness area, e.g. a cloudless sky.

If there is motion in the area where one of these two situations dominates, then it corresponds to what is known as the aperture problem. As noted before, the above situation corresponds to the general case of the aperture problem. It is interesting to note that Barron, Fleet, and Beauchemin [4] recognize empirically that the determinant is a better confidence measure in the application of the Uras, et al. [37] optical flow algorithm than the condition number used in the original paper. Our analysis agrees with their empirical finding.

4.3.3 Integration of confidence measures

Based on the above analysis, we choose a combination of confidence measures according to the nature of a given image sequence.

If the image sequence contains numerous moving objects or the brightness changes significantly, residuals should be used as the confidence measure, as they capture the three problems listed in Section 4.3.1. No other confidence measure is effective for these cases.

The condition number and determinant have something in common although they may capture different situations. Together they signify the relationship between numerical instability and the aperture problem. Empirical findings suggest that they be used as a multiplicative combination, or similarly, in the form of $|\lambda|_{min}$ [25] . This has been proposed by Girosi et al.[13] in a similar context and was used in Barron's implementation [4] of Lucas and Kanade's optical flow algorithm. In our algorithm, $|\lambda|_{min}$ simply indicates the existence of the necessary texture. We shall use it to capture the aperture problem and to avoid numerical instability.

All the above mentioned problems are not unique to our algorithm; they are common to other optical flow algorithms as well.

5. Previous Work and Our Contributions

Recent research in the field of optical flow seems to converge on two ideas to be discussed in this section. They are spatio-temporal filtering and generalized motion models.

An earlier method based on these two ideas has appeared in [35] by Srinivasan. In this approach to a generalized gradient method for optical flow, the author concentrated on generalizing spatio-temporal filtering. He demonstrated his algorithms on various types of motion. However, the algorithms did not deal with motion that simultaneously contained translation, expansion, and rotation. In fact, it is stated that "erroneous results can occur if a translatory motion is superimposed upon the rotation or expansion". Nonetheless, [35] is one of the first efforts in generalizing the optical flow algorithms.

Later, Workhoven and Koenderink [42] introduced the idea of the *affine* flow field (43) to estimate optical flow:

$$\mathbf{V}(\mathbf{x}) = \mathbf{T} + \mathbf{A}\mathbf{x} \quad \text{where } \mathbf{T} = \begin{bmatrix} u_0 \\ v_0 \end{bmatrix} \text{ and } \mathbf{A} = \begin{bmatrix} u_x & u_y \\ v_x & v_y \end{bmatrix}. \tag{43}$$

A series of algorithms [5,7,8,31,39] using this more comprehensive flow field followed.

Based on an infinitesimal affine flow field, both Workhoven and Koenderink [42] and Nagel [31] used Taylor series expansion and 2-D Gaussian derivative filters to derive motion constraint equations. These equations are organized in a linear system in a similar way in both studies. Their work can be regarded as an extension of Horn and Schunck's work [20] . Our work is inspired by Workhoven and Koenderink's algorithm because an extensible motion constraint equation similar to (32) was developed in [42] , though only in 2-D. However, the affine model is not based on the pointwise 3-D motion analysis so their motion equation fails to recognize the dependencies of the first order motion parameters, i.e., $\gamma = u_x = v_y$ and $\rho = u_y = v_x$. This causes numerical instability. Hence their approach does not offer an algorithm with competitive experimental results. In fact, our implementation of their algorithm shows that it is not reliable.

Campani and Verri [7] , Bergen et al. [5] , and Wang and Adelson [39] used local flow field coherence rather than the Taylor expansion to compute flow. Their work can be regarded as an extension of Lucas and Kanade's work [27] . They do not demand high-order gradients but need to perform patchwise computation. Patchwise computation is accurate when the motion has been segmented but inaccurate otherwise due to its strong susceptibility to the aperture problem.

Chou [8] modeled the error in the affine flow field as independent Gaussian noise and used Maximum a Priori (MAP) estimation to minimize the error and compute optical flow. We have shown in (10) that the error modeled in [8] consists of exactly the quadratic terms. It is actually systematic and dependent on motion. Therefore, this noise model is not adequate.

Prior to the affine flow model, Hartley [17] had proposed a quadratic flow field model and used pyramid linking to estimate and segment flow simultaneously. This is the first use of the "correct" motion model in an algorithm. The integration of estimation and segmentation is a valuable lesson for future research, but the lack of temporal or even spatial filtering to deal with noise is its weakness.

We realize that modeling a flow field is essentially a 2-D process, whereas modeling motion is a 3-D process, which is relatively difficult. However we can impose temporal smoothing in an integrated theoretical framework based on Hermite polynomials.

Heeger [19] , Fleet and Jepson [11] , and Weber and Malik [41] also achieved success using spatio-temporal filters. However, they all used a uniform translational motion model and their improvements are limited by this assumption. Among them, Fleet and Jepson attempted to cope with non-translational motion in [11] . They showed that the phase response, instead of the amplitude response, of the velocity-tuned filters is robust to image affine transformations and photometric deformation. Their algorithm is based on constant phase contours and tends to produce more accurate but sparse flow fields.

If the above methods could take advantage of the image motion equation (15), which deals with arbitrary 3-D motion, greater improvements might be achieved. However, these methods might have difficulties with the spatial nonlinearity (specifically, quadratic) and the number of parameters involved. Hermite polynomial filters, on the other hand, have proved to be capable of overcoming these difficulties.

From a theoretical point of view, the image motion corresponding to arbitrary 3-D motion has been studied by Longuet-Higgins and Prazdny[26] and Fang and Huang [10] . We have pushed the effort forward not only by integrating temporally continuous analysis but also by exploring numerical implementations. The figure below summarizes the thread of work leading to our algorithm. An arrow in the figure represents an idea extracted, extended or used similarly.

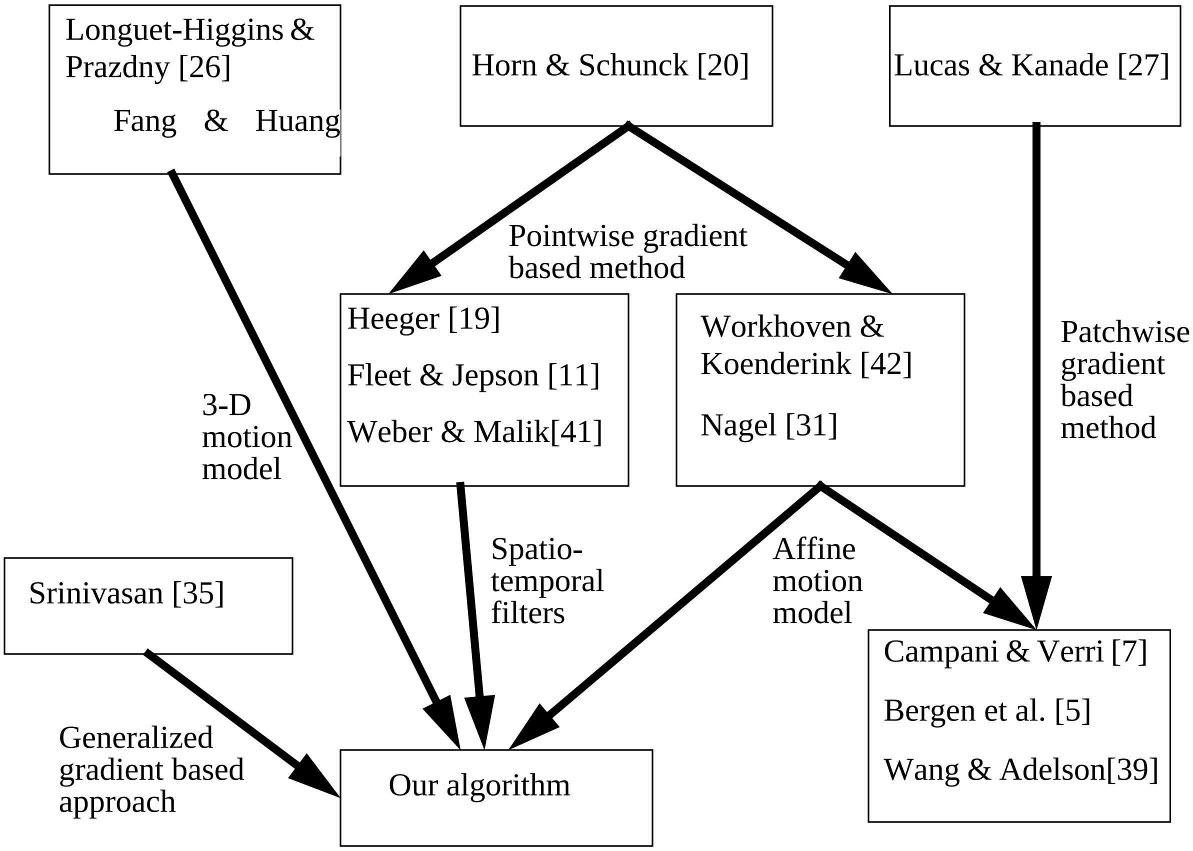
At the application level, our algorithm generates a set of confidence measures that we prove reflect physical phenomena about the image and motion. These measures can then be used for subsequent qualitative processing. In experiments, our algorithm generates accurate and *dense* results, which are very useful for such tasks as motion segmentation and obstacle detection.

In summary, the contributions of this work are a general motion model that lends itself to use with any good spatio-temporal filtering methods for estimating accurate optical flow, and a potent Hermite polynomial theory for motion analysis.

6. Experiments

Based on the work of Barron, Fleet, and Beauchemin[4] , we conducted extensive comparisons between our algorithm and traditional optical flow algorithms, including those by Horn and Schunck[20] , Lucas and Kanade[27] , Uras et al. [37] , Nagel[32] , Anandan[3] , Singh[34] , Heeger[19] , Waxman et al. [40] , and Fleet and Jepson[11] . The synthetic image sequences we used for comparison are Sinusoid, Translating tree, Diverging tree, Yosemite fly-by (provided by Barron), and Moon landing.

If the image sequences used contain only translational (Sinusoid) and diverging motion



(Translating tree, Diverging tree, and Yosemite fly-by), we use the algorithm in (39); if the image motion also contains rotation (Moon landing), we use the algorithm in (35).

The error statistic utilized is the angle error between the computed optical flow time-space direction $(u_e, v_e, 1)$ and the ground truth flow time-space direction $(u_c, v_c, 1)$ averaged over the whole image. Refer to [4] for more details. In order to make extensive comparisons, we implemented our algorithm in such a way that a certain density of output flow can be extracted by thresholding on a chosen confidence measure. Error statistics in the following subsections are displayed in tables. For a single technique with multiple entities in these tables, different threshold values are used in the algorithm to produce multiple densities of output. For the actual threshold values of the comparison algorithms, refer to Barron et al.[4] . The error statistics and associated density for the comparison algorithms were obtained directly from [4] .

6.1 Sinusoid

This is a synthetic image sequence (Fig 6) of a spatial sinusoidal wave traversing toward the upper right side. For our method we chose a window size large enough (17x17x7 for *x*,*y*,*t*) to prevent aliasing. $|1/r|$ was used as the confidence measure. Fig 7.1 shows the true optical flow, while Fig 7.2 shows the flow computed with our method. Our algorithm performs better than all of the other algorithms except Fleet and Jepson’s (Table 2).

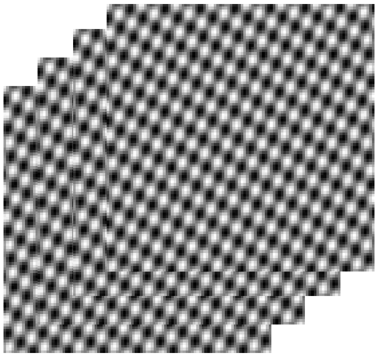


Fig 6. Traversing sinusoid

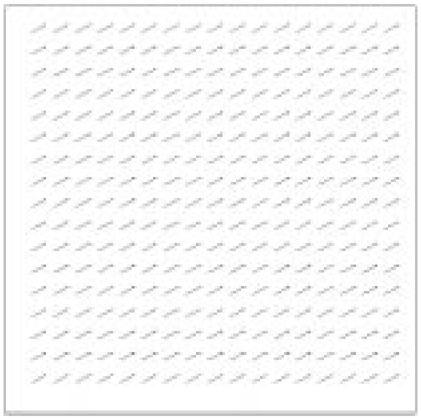


Fig 7.1True optical flow for sinusoid

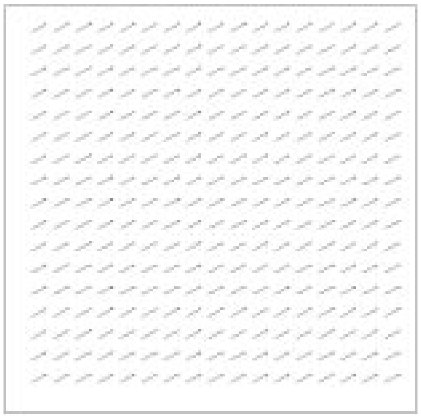


Fig 7.2 Computed optical flow (100%)

Table 2: Summary of Sinusoid error statistics

Density	Our Algorithm		Other Algorithms		
	Average Error	**Standard Deviation**	**Average Error**	**Standard Deviation**	**Technique by**
100%	0.63°	0.06°	4.19°	0.50°	Horn & Schunck (original unthresholded)
			2.55°	0.59°	Horn & Schunck (modified unthresholded)
			2.47°	0.16°	Lucas and Kanade (unthresholded)
			2.59°	0.71°	Uras et al. (unthresholded)
			2.55°	0.93°	Nagel
			30.80°	5.45°	Anandan
			2.24°	0.02°	Singh (step 1 unthresholded)
			0.03°	0.01°	Fleet and Jepson
12.8%	0.63°	0.06°	64.26°	26.14°	Waxman et al.

6.2 Translating tree and Diverging tree

The translating and diverging tree sequences are two realistic synthetic sequences simulating the motion of simple translation (Fig 8.1) and expansion (Fig 8.2), respectively, of a poster. The window size used in our method is 19x19x11 for the translating tree and 17x17x9 for the diverging tree. Due to the lack of texture in some background areas, we used $|\lambda|_{min}$ as the confidence measure. Fig 9 and Fig 10 show the results. Only Uras' and Fleet and Jepson's algorithms perform better than ours for the translating tree sequence (Table 3). For the diverging tree sequence, our results are second only to Fleet and Jepson's (Table 4).

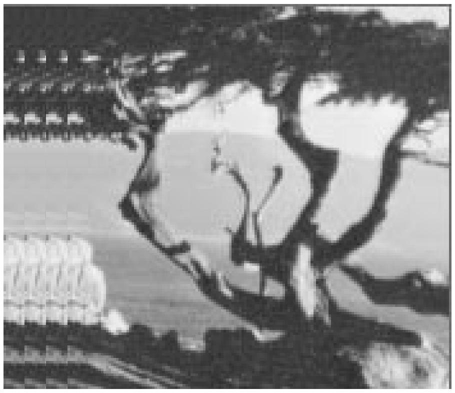


Fig 8.1 Translating tree

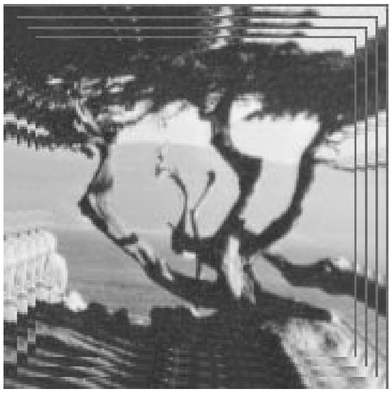


Fig 8.2 Diverging tree

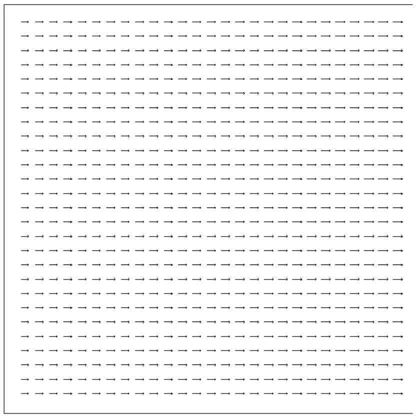


Fig 9.1 True flow for Translating tree

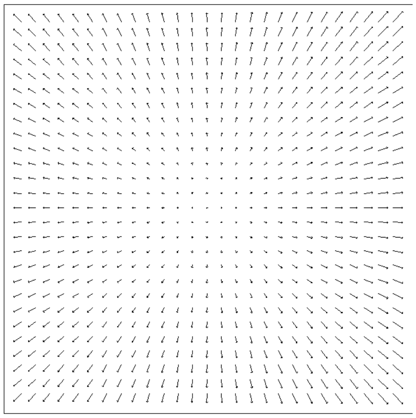


Fig 9.2 True flow for Diverging tree

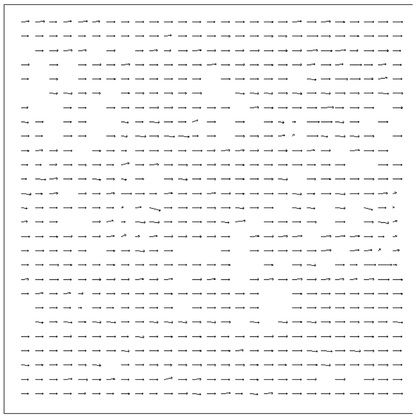


Fig 10.1 Computed flow for Translating tree

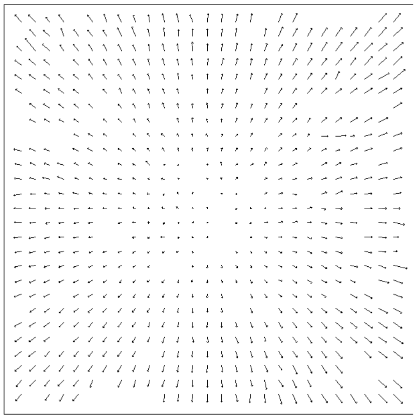


Fig 10.2 Computed flow for Diverging tree

Table 3: Summary of Translating tree error statistics

Density	Our Algorithm		Other Algorithms		
	Average Error	Standard Deviation	Average Error	Standard Deviation	Technique by
100%	0.92°	0.94°	38.72°	27.67°	Horn & Schunck (original unthresholded)
			2.02°	2.27°	Horn & Schunck (modified unthresholded)
			0.62°	0.52°	Uras et al. (unthresholded)
			2.44°	3.06°	Nagel
			4.54°	3.10°	Anandan
			1.64°	2.44°	Singh (step 1 unthresholded)
			1.25°	3.29°	Singh (step 2 unthresholded)
99.6%	0.91°	0.92°	1.11°	0.89°	Singh (step 2)
74.5%	0.69°	0.51°	0.32°	0.38°	Fleet and Jepson
53-57%	0.59°	0.39°	32.66°	24.50°	Horn & Schunck (original)
			5.63°	2.78°	Heeger (level 1)
			1.89°	2.40°	Horn & Schunck (modified)
49.7%	0.57°	0.37°	0.23°	0.19°	Fleet and Jepson
44.2%	0.55°	0.34°	8.50°	13.50°	Heeger (level 0)
40-42%	0.53°	0.33°	0.46°	0.35°	Uras et al.
			0.72°	0.75°	Singh (step 1)
			0.66°	0.67°	Lucas and Kanade
26.8%	0.48°	0.28°	0.25°	0.21°	Fleet and Jepson
13.1%	0.42°	0.24°	0.56°	0.58°	Lucas and Kanade
1.9%	0.35°	0.19°	6.66°	10.72°	Waxman et al.

Table 4: Summary of Diverging tree error statistics

Density	Our Algorithm		Other Algorithm		
	Average Error	Standard Deviation	Average Error	Standard Deviation	Technique by
100%	1.84°	1.33°	12.02°	11.72°	Horn & Schunck (original unthresholded)
			2.55°	3.67°	Horn & Schunck (modified unthresholded)
			4.64°	3.48°	Uras et al. (unthresholded)
			2.94°	3.23°	Nagel
			7.64°	4.96°	Anandan
			17.66°	14.25°	Singh (step 1 unthresholded)
			8.60°	4.78°	Singh (step 2 unthresholded)
99%	1.82°	1.28°	8.40°	4.78°	Singh (step 2)
73.8%	1.59°	1.12°	4.95°	3.09°	Heeger (combined)
60-61%	1.49°	1.02°	0.99°	0.78°	Fleet and Jepson
			8.93°	7.79°	Horn & Schunck (original)
			3.83°	2.19°	Uras et al.
46-48%	1.40°	0.92°	2.50°	3.89°	Horn & Schunck (modified)
			0.80°	0.73°	Fleet and Jepson
			1.94°	2.06°	Lucas and Kanade
28.2%	1.28°	0.79°	0.73°	0.46°	Fleet and Jepson

Table 4: Summary of Diverging tree error statistics

Density	Our Algorithm		Other Algorithm		
	Average Error	Standard Deviation	Average Error	Standard Deviation	Technique by
24.3%	1.24°	0.77°	1.65°	1.48°	Lucas and Kanade
3.9-4.9%	1.09°	0.66°	13.69°	11.83°	Waxman et al.
			5.62°	6.16°	Singh (step 1)

6.3 Yosemite fly-by

The Yosemite Fly-by sequence is a realistic synthetic image sequence (Fig 11). The flight scene is simulated using actual aerial photos and digital terrain maps, with artificial sky and clouds. Since the clouds in the sky change brightness over time, it poses difficulties for all algorithms. Based on our previous analysis, we used $|1/r|$ as the confidence measure to eliminate points that lie in a large blank area in the sky and on motion boundaries in. Fig 12.2. shows the results. Since the motion is rather fast in some areas, we used a larger window (21x21x7). Error statistics are shown in Table 5. Again, the clouds account for the large magnitude error. Our algorithm performs better than all others.

Table 5: Summary of Yosemite fly-by error statistics

Density	Our Algorithm		Other Algorithms		
	Average Error	Standard Deviation	Average Error	Standard Deviation	Technique by
100%	7.13°	13.19°	31.69°	31.18°	Horn & Schunck (original unthresholded)
			9.78°	16.19°	Horn & Schunck (modified unthresholded)
			8.94°	15.61°	Uras et al. (unthresholded)
			10.22°	16.51°	Nagel
			13.36°	15.64°	Anandan
			15.28°	19.61°	Singh (step 1 unthresholded)
			10.44°	13.94°	Singh (step 2 unthresholded)
97.7%	6.39°	6.39°	10.03°	13.13°	Singh (step 2)
64.2%	2.99°	4.54°	22.82°	35.28°	Heeger (level 0)
59.6%	2.85°	4.15°	25.33°	28.51°	Horn & Schunck (original)
44.8%	2.57°	3.50°	15.93°	23.16°	Heeger (combined)
33-35%	2.41°	3.32°	4.28°	11.41°	Lucas and Kanade
			4.63°	13.42°	Fleet and Jepson
			5.59°	11.52°	Horn & Schunck (modified)
			6.06°	12.02°	Nagel
30.6%	2.38°	3.24°	5.28°	14.34°	Fleet and Jepson
15%	2.21°	3.06°	9.87°	14.74°	Heeger (level 1)
			7.55°	19.64°	Uras et al.
8.7%	2.16°	3.05°	3.22°	8.92°	Lucas and Kanade
7.4%	2.14°	3.03°	20.05°	23.23°	Waxman et al.
2.4%	1.91°	2.12°	12.93°	15.36°	Heeger (level 2)

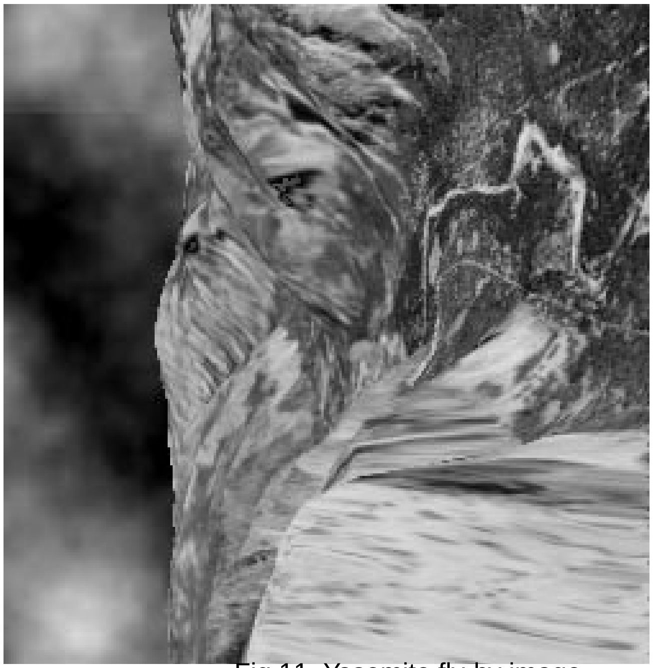


Fig 11. Yosemite fly-by image

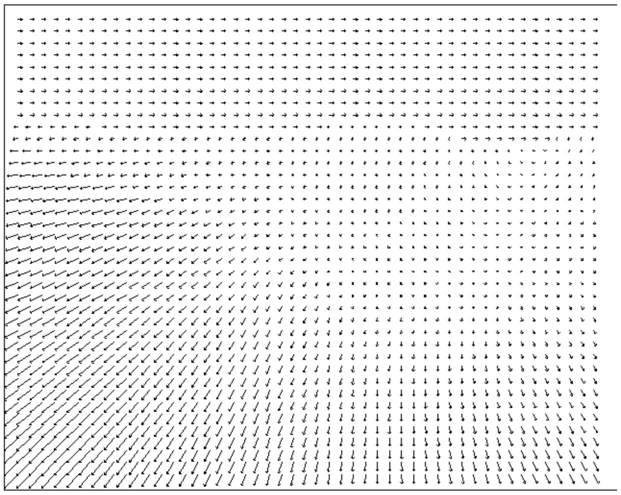


Fig 12.1 True optical flow field for Yosemite fly-by

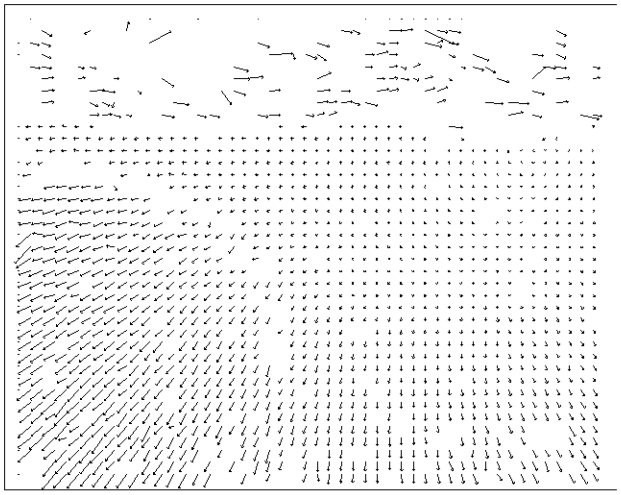


Fig 12.2 Computed optical flow for Yosemite fly-by

6.4 Moon landing

The Moon landing sequence (Fig 13) is generated by gradually rotating and expanding* a picture of the surface of the moon. Visually, it is a bird's-eye view of the moon from a spiral landing spaceship. The purpose of this sequence is to demonstrate our algorithm's capability to handle complex motion, specifically, expansion plus rotation. Our algorithm used a 21x21x7 window and $|\lambda|_{min}$ as the confidence measure since there is no motion boundary. Table 6 shows that our results are better than Fleet and Jepson's and Lucas & Kanade's. It also reveals the amount of

* Rotation and expansion are done using Khoros 1.5 vrotate and vresize functions, respectively.

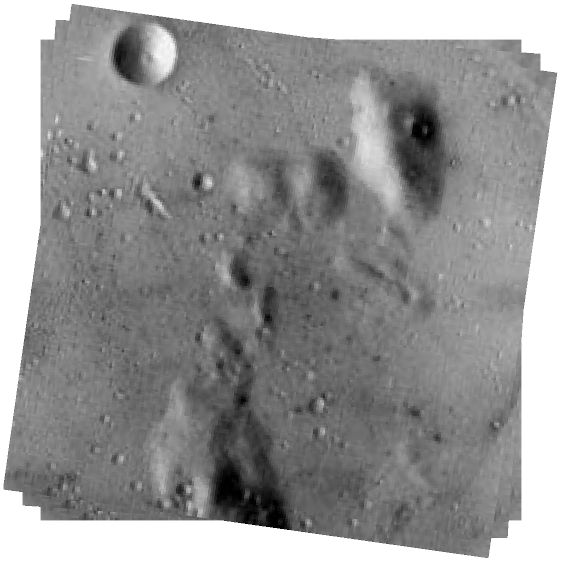


Fig 13. Moon landing sequence

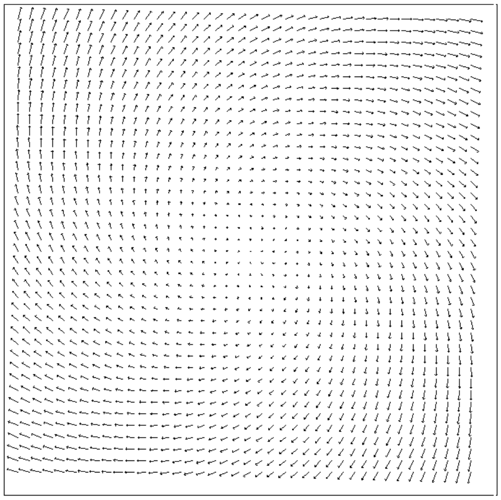


Fig 14.1 Moon landing ground truth flow

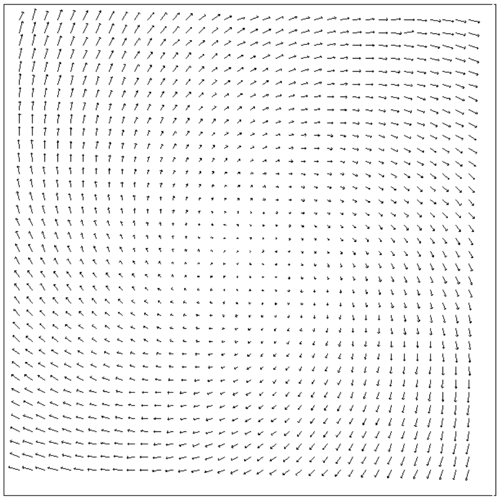


Fig 14.2 Our algorithm's flow field (100%)

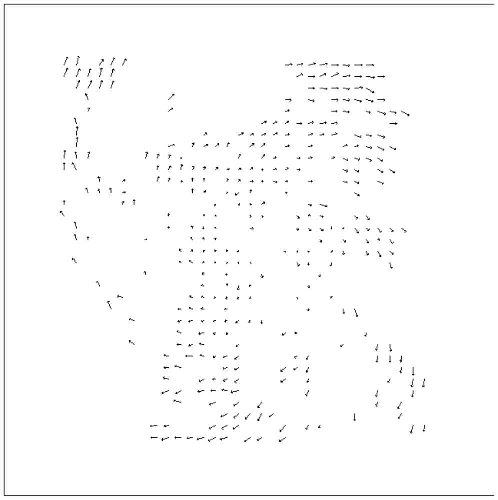


Fig 14.3 Lucas & Kanade's flow field (33.3%)

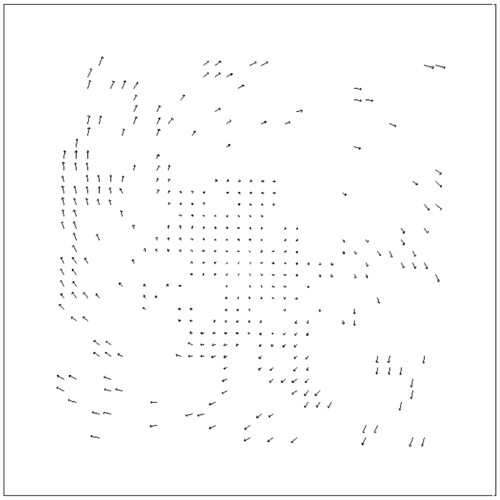


Fig 14.4 Fleet & Jepson's flow field (31.1%)

Table 6: Summary of Moon landing error statistics

Density	Algorithms		
	Average Error	Standard Deviation	Technique by
100%	1.73°	0.87°	Our algorithm (Translation + Rotation +Expansion model)
	1.91°	0.89°	Our algorithm (Translation model)
33.3%	3.91°	3.80°	Lucas and Kanade
	1.37°	0.71°	Our algorithm (Translation + Rotation +Expansion model)
	1.69°	0.83°	Our algorithm (Translation model)
31.1%	2.47°	1.71°	Fleet and Jepson
	1.36°	0.70°	Our algorithm (Translation + Rotation +Expansion model)
	1.68°	0.82°	Our algorithm (Translation model)

improvement (10%-16%) of a generalized motion model over a uniform translation motion model in our algorithm.

6.5 Noise sensitivity

We created noisy images from the synthetic sequences used above and tested the sensitivity of the algorithms to such noise.

The sensitivity analysis is motivated by simple experiments such as the following: On a real-time image processing machine, run a temporal differencing algorithm at video rate on successive frames while keeping the camera and the scene stationary. Instead of getting a uniform output of zero, the actual output always contains random spots of non-zero values. This kind of sensor noise violates brightness constancy and degrades the accuracy of any optical flow algorithm.

In the following tables, we used additive Gaussian noise of zero mean and increasing variance. In order to conduct a fair comparison, the threshold on the confidence measure is fine-tuned in every single run so that the output density is always 50%. We chose two of the best algorithms in [4] , Lucas & Kanade and Fleet & Jepson, for comparison. For the noisy Diverging tree sequence, the noise sensitivity is summarized in Table 5.

Table 7: Summary of Diverging tree noise sensitivity statistics

Noise Standard Deviation	Our Algorithm		Fleet & Jepson		Lucas & Kanade	
	Average Error	Standard Deviation	Average Error	Standard Deviation	Average Error	Standard Deviation
0	1.41°	0.94°	1.09°	0.52°	3.04°	2.53°
3	1.64°	1.08°	1.18°	0.61°	3.28°	2.77°
6	2.03°	1.37°	1.51°	0.93°	3.62°	3.06°
9	2.53°	2.17°	2.15°	1.78°	4.32°	3.79°
12	3.28°	2.78°	3.83°	5.48°	5.17°	4.69°
15	3.87°	3.15°	9.23°	12.04°	5.93°	5.41°

Our algorithm, as well as Lucas & Kanade's has an approximately linear error increase with noise while Fleet & Jepson's has quadratic or even exponential error increase (Fig 15). Despite its

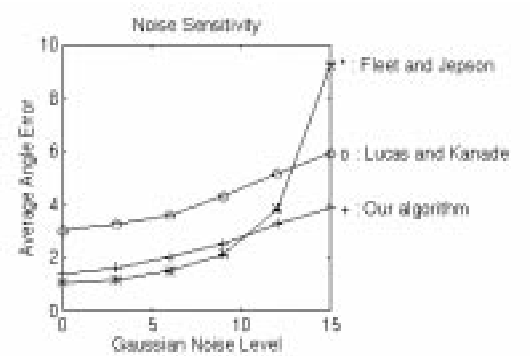


Fig 15. Noise sensitivity plot for Diverging tree

excellent accuracy for noise-free data, Fleet & Jepson's algorithm is outperformed by the other two when the image sequence becomes noisy. We also conducted a similar experiment with the Yosemite fly-by sequence. Unfortunately, Fleet & Jepson's algorithm does not generate high enough density data when the sequence becomes noisy. However, we can once again confirm the linear error with respect to noise for Lucas & Kanade's and our algorithm (Fig 16).

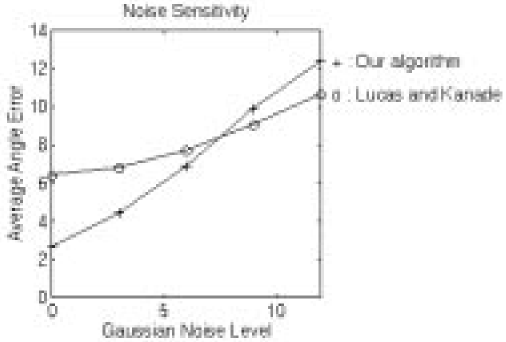


Fig 16. Noisy Yosemite fly-by

Robustness to noise is a very crucial quality for a good optical flow algorithm. We hope this experiment prompts more research in this area. Our algorithm achieved robustness by integrating spatio-temporal smoothing in the 3-D Hermite polynomial differentiation filter theory.

7. Real Images Demonstration

Current optical flow algorithms often have difficulty with real image sequences. Our algorithm performs best on the Yosemite and Moon landing sequences because these sequences model real 3-D motion and are complicated enough to reveal the virtues of our algorithm. We therefore expect it to perform well on real images. Here we demonstrate our algorithm with two real image sequences, HMMWV and NASA. The HMMWV sequence was taken in an outdoor environment with a camera mounted on a forward moving HMMWV (High Mobility Multipurpose Wheeled Vehicle). It was later stabilized to eliminate unsteady motion. The NASA sequence is an indoor diverging scene obtained from Barron [4] . The flow outputs for our algorithm as well as Lucas & Kanade's and Fleet & Jepson's are displayed in Fig 17 and Fig 18. For our algorithm, the output has undergone thresholding based on two confidence measures, $|1/r|$ and $|\lambda|_{min}$.

In the HMMWV sequence*, visual inspection shows that Lucas & Kanade's flow output (Fig 17.3) is very noisy because of random velocity change in a small neighborhood. Fleet & Jepson's flow output (Fig 17.4) shows no indication of the flow field divergence. It is probable that the particular implementations (provided by Barron) of these two algorithms are not tuned to the relatively large flow existing in this sequence. Our algorithm, on the other hand, produces coherently diverging flow field (Fig 17.2) except in the area of the sky.

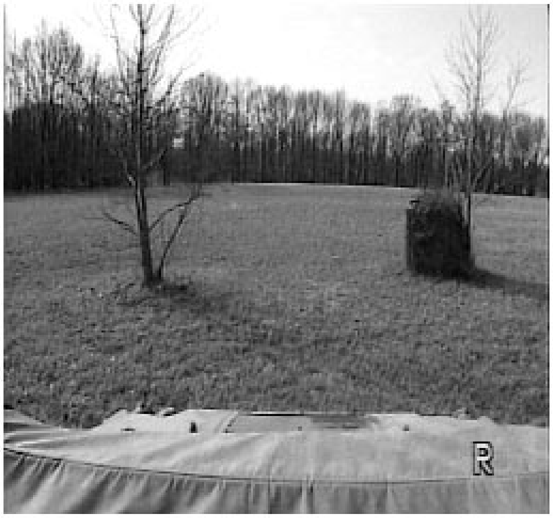


Fig 17.1 HMMWV sequence

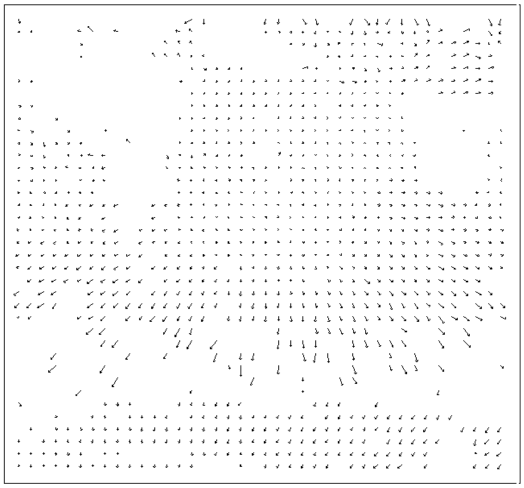


Fig 17.2 Our algorithm's flow field (64% density)

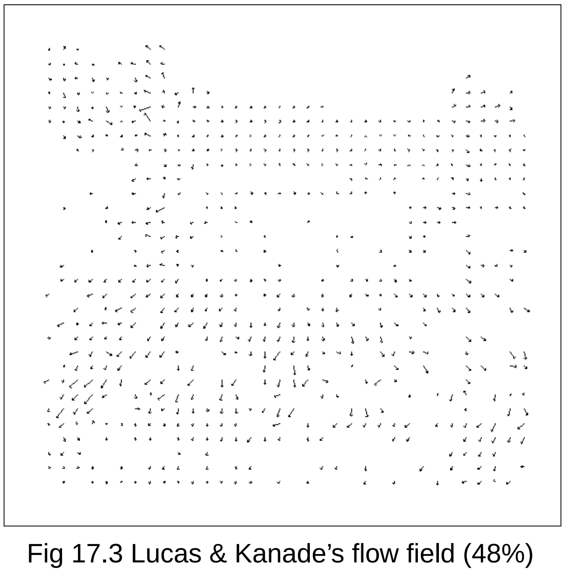


Fig 17.3 Lucas & Kanade's flow field (48%)

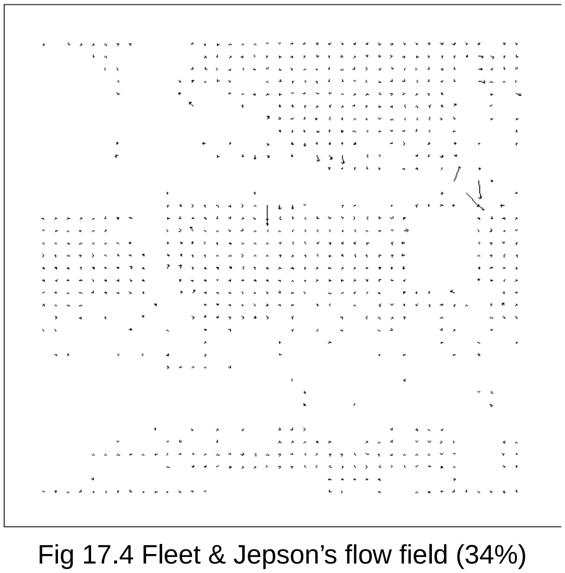


Fig 17.4 Fleet & Jepson's flow field (34%)

In the NASA sequence†, both our flow (Fig 18.2) and Fleet & Jepson's flow outputs (Fig 18.4) are very good, while Lucas & Kanade's algorithm produces a noisy flow field (Fig 18.3). Note that our output density is twice that of Fleet & Jepson's but it achieves approximately the same accuracy visually. If Fleet & Jepson were to generate the same density, it might not look as accurate.

* $\sigma = 0.8$ is used for both algorithms because only 10 image frames are available. The filter size of our algorithm is 21x21x7.

† $\sigma = 2.0$ is used for both algorithms. The filter size of our algorithm is 21x21x7.

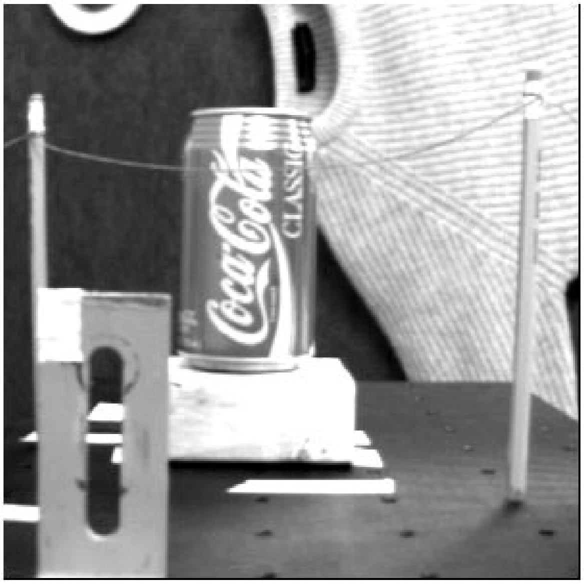


Fig 18.1 NASA sequence

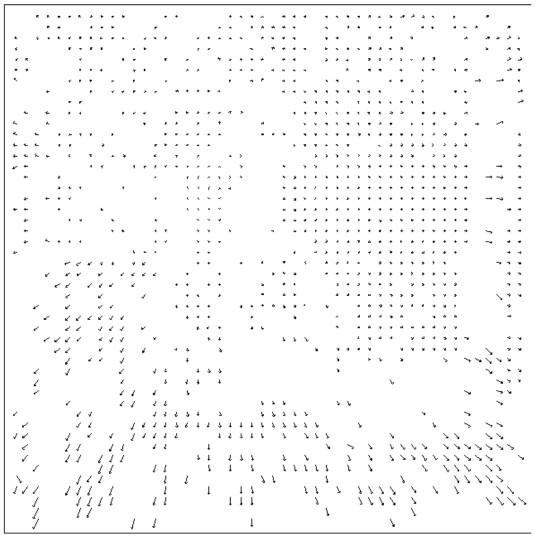


Fig 18.2 Our algorithm’s flow field (75% density)

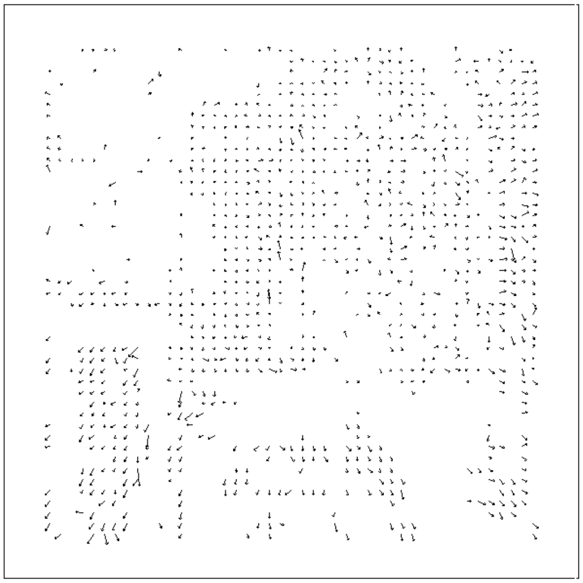


Fig 18.3 Lucas & Kanade’s flow field (48% density)

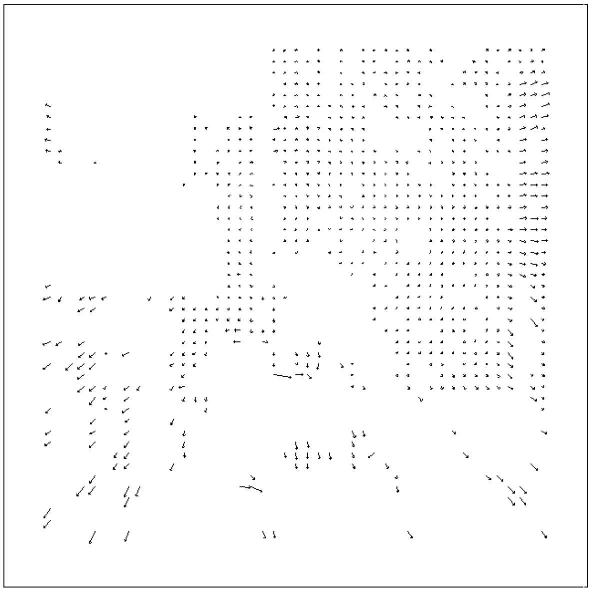


Fig 18.4 Fleet & Jepson’s flow field (37% density)

Finally we apply the NASA flow field outputs from these three algorithms to an obstacle detection algorithm developed by Young, et al.[45] [46] [47] . This algorithm discriminates between obstacle and non-obstacle regions in the image using only the perpendicular component of flow to arbitrarily chosen image lines. In the following figures, a protrusion or a depression represents an obstacle detected by the algorithm.

For the first set of data, we select the horizontal lines from 220 to 260 (Fig 19.2). Over these lines, there is a vertical long metal plate with a hole in the left end of the image strip. We should expect two depressions at the locations of the plate. The detection results from all lines are averaged and then displayed in Fig 20. In Fig 20.1, Lucas & Kanade’s flow does not detect the metal

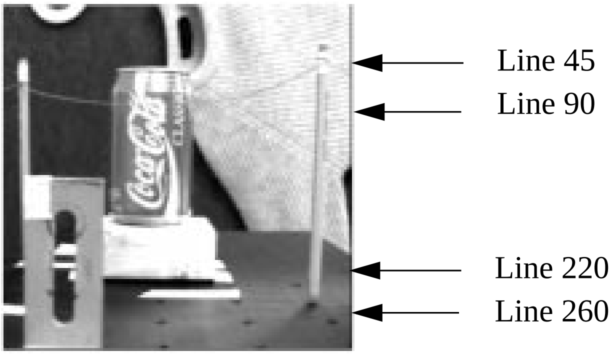


Fig 19.1 NASA image lines 45 to 90

Fig 19.2 NASA image lines 220 to 260

plate clearly; in Fig 20.2, Fleet & Jepson’s flow detects the metal plate but its shape is hardly recognizable; in Fig 20.3, our algorithm not only detects the plate but also shows its shape as should be expected.

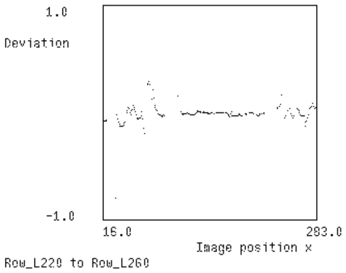


Fig 20.1 Lucas & Kanade’s results

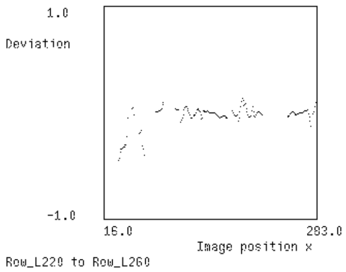


Fig 20.2 Fleet & Jepson’s results

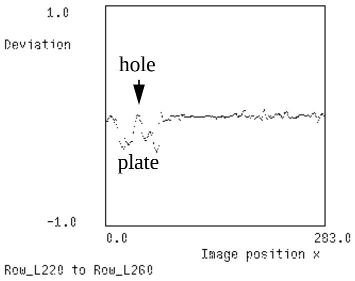


Fig 20.3 Our algorithm’s results

For the second set of data, we select horizontal lines 45 to 90 (Fig 19.1). Over these lines, there is a pole on each end of the image strip and a coke can at the center. In Fig 21.1, Lucas & Kanade’s flow does not make out meaningful objects. In Fig 21.2, Fleet & Jepson’s flow barely detects the coke can and the right pole, and does not detect the left pole. In Fig 21.3, our flow clearly detects all three objects. Note that for this particular image strip we used dense (100%) flow for our algorithm.

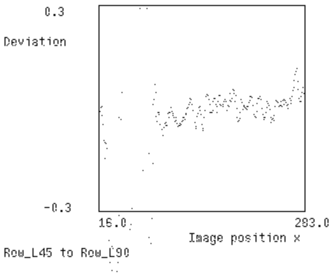


Fig 21.1 Lucas & Kanade’s results

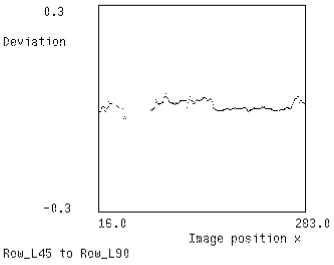


Fig 21.2 Fleet & Jepson’s results

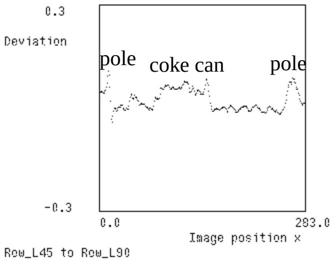


Fig 21.3 Our algorithm’s results

8. Conclusion

Motion estimation is difficult and ill-posed. The past two decades of research have led to spatio-temporal filtering techniques to overcome sensor noise, brightness change over time, and quantization error. The aperture problem is mitigated by increased filter support or other global techniques, while other approaches attempt to use an affine motion model to pursue better accuracy in the optical flow. We have learned from these results and have developed an integrated approach that combines a general motion model and 3-D Hermite polynomial differentiation filters. The general model for arbitrary 3-D motion is useful for all motion algorithms, but better numerical techniques are required to make good use of the model. We have found that Hermite polynomial theory provides necessary advantages for this purpose. It possesses many elegant properties, including orthogonality, extensibility, Gaussian smoothing, etc. Contrary to general belief, the behaviors of these high order differentiation filters are quite insensitive to noise. This observation is supported by the good results in our noise sensitivity analysis. Simplicity adds yet another dimension to the strength of this algorithm, making real-time implementation feasible. Although we are focusing on presenting accurate optical flow results, our algorithm also computes all the motion parameters, including 3-D translation and rotation, along with the flow. These motion parameters can be directly utilized for other motion applications, for example, computing time-to-contact with γ, or performing derotation (stabilization) with ρ (16). These applications, however, are limited by inherent ambiguities due to image noise[1] ,[44] ; on the other hand, a scheme of integrating or propagating motion information globally is restricted by occlusion and motion boundary effects. Our future studies will investigate this problem; hopefully, the results can be integrated into the current work. In summary, this work has generalized and unified several previous successful theoretical approaches and has resulted in a versatile and flexible algorithm.

Acknowledgment

We would like to thank J.L. Barron for providing the images and comparison results upon which our experiments are based. The HMMWV images were provided by Mr. Gombash of the Army Research Lab, and stabilized by Dr. Q. Zheng at the Center for Automation Research, University of Maryland. Their help is greatly appreciated. We thank Dr. Gin-Shu Young for his help on the obstacle detection algorithm. The authors would also like to thank Dr. Azriel Rosenfeld and Marilyn Nashman for their valuable comments.

Appendix. A

We prove Theorem 1 as follows:

Proof: The first equality comes from the orthogonality of $\{ \bar{H}_n(x) \}$. We now prove the second equality, which claims that the scalar product of a function and the *k*th Hermite polynomial is equal to the scalar product of the *k*th derivative of the function and 1.

$$
\begin{aligned}
\langle I, \overline{H}_k \rangle &= \int_{-\infty}^{\infty} G(x) I(x) \overline{H}_k(x) dx \\
&= \int_{-\infty}^{\infty} G(x) I(x) (-1)^k G^{-1}(x) \frac{d^k G(x)}{dx^k} dx \\
&= (-1)^k \int_{-\infty}^{\infty} I(x) \frac{d^k G(x)}{dx^k} dx \\
&= (-1)^k I(x) \frac{d^{k-1} G(x)}{dx^{k-1}} \bigg|_{-\infty}^{\infty} + (-1)^{k-1} \int_{-\infty}^{\infty} \frac{dI(x)}{dx} \frac{d^{k-1} G(x)}{dx^{k-1}} dx \\
&= \int_{-\infty}^{\infty} I^{(1)}(x) (-1)^{k-1} \frac{d^{k-1} G(x)}{dx^{k-1}} dx \\
&= \langle I^{(1)}, \overline{H}_{k-1} \rangle \\
&\vdots \\
&= \langle I^{(k)}, \overline{H}_0 \rangle
\end{aligned}
$$

❑

Appendix. B

Let A_n and b, defined in (34), contain no noise and let the noise be modelled as in (40). Then

$$E = A_n s + b = 0 \text{ and } s = -(A_n^T A_n)^{-1} A_n^T b \,. \qquad (44)$$

Let the new optical flow be $\tilde{s}$ and the new residual be $\tilde{E}$, and assume that $N \ll A_n$ and $\Delta b \ll b$ elementwise. Then

$$\tilde{s} = -[(A_n + N)^T (A_n + N)]^{-1} (A_n + N)^T (b + \Delta b) \quad \text{, and} \qquad (45)$$

$$
\begin{aligned}
[(A_n + N)^T (A_n + N)]^{-1} &\approx (A_n^T A_n [I + (A_n^T A_n)^{-1} (A_n^T N + N^T A_n)])^{-1} \\
&\approx [I - (A_n^T A_n)^{-1} (A_n^T N + N^T A_n)] (A_n^T A_n)^{-1}
\end{aligned}
\quad \text{, so} \qquad (46)
$$

$$\tilde{s} \approx -(A_n^T A_n)^{-1} A_n^T b + (A_n^T A)^{-1} (A_n^T N + N^T A_n)(A_n^T A_n)^{-1} A_n^T b + (A_n^T A_n)^{-1} A_n^T b - (A_n^T A_n)^{-1} A_n^T \Delta b$$

Using (44), this can be simplified as follows:

$$\tilde{s} \approx s - (A_n^T A_n)^{-1} A_n^T N f - (A_n^T A_n)^{-1} A_n^T \Delta b \text{ and } \Delta s \approx (A_n^T A_n)^{-1} A_n^T N s - (A_n^T A_n)^{-1} A_n^T \Delta b \,.$$

For the residual, substituting $\tilde{s}$ into (40), and using (44), we have

$$\tilde{E} \approx \left\|(A_n + N)s - A_n(A_n^T A_n)^{-1} A_n^T N s - A_n(A_n^T A_n)^{-1} A_n^T \Delta b + b + \Delta b\right\|$$

$$\approx \left\|(I - A_n(A_n^T A_n)^{-1} A_n^T)(Ns + \Delta b)\right\| \qquad \text{as in (42).}$$

To understand $\tilde{E}$ better, we analyze the matrix $I - A_n(A_n^T A_n)^{-1} A_n^T$, denoted by T.

It is easy to verify that the only nontrivial eigenvalues of matrix T is/are 1, which means that it maps any vector $(Ns + \Delta b)$ to only the directions specified by the eigenvectors corresponding to the nontrivial eigenvalues.

References

[1] Adiv, G., “Inherent Ambiguities in Recovering 3-D Motion and Structure from a Noisy Flow Field”, IEEE Transactions on Pattern Analysis and Machine Intelligence, vol. 11, no. 5, pp. 477-489, 1989.

[2] Adiv, G. “Determining Three-Dimensional Motion and Structure from Optical Flow Generated by Several Moving Objects”, IEEE Transactions on Pattern Analysis and Machine Intelligence, vol. 7, no. 4, pp. 384-401, 1985.

[3] Anandan, P., “Measuring Visual Motion from Image Sequences”, Ph.D. Thesis, COINS TR 87-21, University of Massachusetts, Amherst MA, 1987.

[4] Barron, J. L., Fleet, D. J., and Beauchemin, S. S., “Performance of Optical Flow Techniques”, International Journal of Computer Vision, vol. 12, no. 1, pp. 43-77, 1994.

[5] Bergen, J.R., Burt, P.J., Hingorani, R, Peleg, S., “A Three-Frame Algorithm for Estimating Two-Component Image Motion”, IEEE Transactions on Pattern Analysis and Machine Intelligence, vol. 14, no. 9, pp. 886-896, 1992.

[6] Burt, P., Hingorani, R., Kolczynski, R., “Mechanisms for Isolating Component Patterns in the Sequential Analysis of Multiple Motion”, Proceedings of the IEEE Workshop on Visual Motion, Princeton, NJ, pp.187-195, 1991.

[7] Campani, M. and Verri, A., “Motion Analysis from First-Order Properties of Optical Flow”, CVGIP: Image Understanding, vol. 50, no. 1, pp. 90-107, July 1992.

[8] Chou, W-S., Chen, Y-C., “Estimation of the Velocity Field of Two-Dimensional Deformable Motion”, Pattern Recognition, vol. 26, no. 2, pp. 351-364, 1993.

[9] Duncan, J., Chou, T., “On the Detection of Motion and the Computation of Optical Flow”, IEEE Transactions on Pattern Analysis and Machine Intelligence, vol. 14, no. 3, pp. 346-352, 1989.

[10] Fang, J.Q., Huang, T.S., “Solving Three Dimensional Small-Rotation Motion Equation”, Proceedings of IEEE Conference on Computer Vision and Pattern Recognition, Washington, DC., pp. 253-258, 1983.

[11] Fleet, D.J. and Jepson, A.L., “Computation of Component Image Velocity from Local Phase Information”, International Journal of Computer Vision, vol. 5, no.1, pp. 77-104, 1990.

[12] Gibson, J., “The Senses Considered as Perceptual Systems”, Houghton-Mifflin, Boston, MA, 1966.

[13] Girosi, F., Verri, A., Torre, V., “Constraints for the Computation of Optical Flow”, Proceedings of the IEEE Workshop on Visual Motion, Irvine, CA, pp. 116-124, 1989.

[14] Grzywacz, N.M. and Yuille A.L., “A Model for Estimate of Local Image Velocity by Cells in The Visual Cortex”, Proceedings of the Royal Society of London, vol. A 239, pp. 129-161, 1990.

[15] Guissin, R., Ullman, S., “Direct Computation of Focus of Expansion from Velocity Field Measurements”, Proceedings of the IEEE Workshop on Visual Motion, Princeton, NJ, pp. 146-155, 1991.

[16] Haralick, R. M. and Lee, J. S., “The Facet Approach to Optic Flow”, Proceedings of the DARPA Image Understanding Workshop, Arlington VA, pp. 84-94, 1983.

[17] Hartley, R., “Segmentation of Optical Flow Fields by Pyramid Linking”, Pattern Recognition Letters, vol.3, no. 5, pp.253-262, 1985.

[18] Hashimoto, M., and Sklansky, J., “Multiple-Order Derivatives for Detecting Local Image Characteristics”, Computer Vision, Graphics, and Image Processing, vol. 39, no. 1, pp. 28-55, 1987.

[19] Heeger, D. J., “Optical Flow Using Spatiotemporal Filters”, International Journal of Computer Vision, vol. 1, no. 4, pp. 279-302, 1988.

[20] Horn, B. K. P. and Schunck, B. G., “Determining Optical Flow”, Artificial Intelligence, vol. 17, pp. 185-204, 1981.

[21] Horn, R.A., Johnson, C.R., “Topics in Matrix Analysis”, Cambridge University Press, Cambridge, UK, Section 6.1, pp. 382-397, 1991.

[22] Ja’Ja’, J., “An Introduction to Parallel Algorithms”, Addison-Wesley, Reading, MA, 1991.

[23] Kearney, J. K., Thompson, W. B. and Boley,D. L., “Optical Flow Estimation: An Error Analysis of Gradient Based Methods With Local Optimization”, IEEE Transactions on Pattern Analysis and Machine Intelligence, vol. 9, no. 2, pp. 229-244, 1987.

[24] Koenderink, J.J. and Van Doorn, A.J., “Representation of Local Geometry in the Visual System”, Biological Cybernetics, vol. 55, pp. 367-375, 1987.

[25] Liu, H., Hong, T., Herman, M., Chellappa, R., “A Reliable Optical Flow Algorithm Using 3-D Hermite Polynomials”, NIST-IR 5333, December, 1993.

[26] Longuet-Higgins, H.C., Prazdny, K., “The Interpretation of a Moving Retinal Image”, Proceedings of the Royal Society of London, vol. B 208, pp. 385-397, 1980.

[27] Lucas, B. D. and Kanade, T., “An Iterative Image Registration Technique with an Application to Stereo Vision”, Proceedings of the DARPA Image Understanding Workshop, pp.121-130, 1981.

[28] Murray, D., Buxton, B., “Scene Segmentation from Visual Motion Using Global Optimization”, IEEE Transactions on Pattern Analysis and Machine Intelligence, vol. 9, no. 2, pp. 220-228, 1987.

[29] Mutch, K., Thompson, W., “Analysis of Accretion and Deletion at Boundaries in Dynamic Scenes”, IEEE Transactions on Pattern Recognition and Machine Intelligence, vol. 7, no.2, pp. 133-138, 1985.

[30] Nagel, H. H., “Constraints for the Estimation of Displacement Vector Fields from Image Sequences”, Proceedings of International Joint Conference on Artificial Intelligence, pp. 945-951, 1983.

[31] Nagel, H. H., “Direct Estimation of Optical Flow and of its Derivatives”, Artificial and Biological Vision Systems, Ed. Orban, G.A., Nagel, H. H., pp. 193-224, 1992.

[32] Nagel, H. H., “Displacement Vectors Derived from Second-order Intensity Variations in Image Sequences”, Computer Vision, Graphics and Image Processing, vol. 21, no. 1, pp. 85-117, 1983.

[33] Nelson, R., Aloimonos, J., “Obstacle Avoidance Using Flow Field Divergence”, IEEE Transactions on Pattern Analysis and Machine Intelligence, vol. 11, no. 10, pp. 1102-1106, 1989.

[34] Singh, A., “An Estimation-Theoretic Framework for Image-Flow Computation”, Proceedings of the International Conference on Computer Vision, pp. 169-177, Osaka, Japan, 1990.

[35] Srinivasan, M.V., “Generalized Gradient Schemes for the Measurement of Two-Dimensional Image Motion”, Biological Cybernetics, vol. 63, pp. 421-431, 1990.

[36] Stewart, G.W., “Introduction to Matrix Computation”, Academic Press, New York, 1973.

[37] Uras, S., Girosi, F., Verri, A., Torre, V. “A Computational Approach to Motion Perception”, Biological Cybernetics, vol. 60, pp. 79-97, 1988.

[38] Verri, A., Poggio, T. “Motion Field and Optical Flow: Qualitative Properties”, IEEE Transactions on Pattern Analysis and Machine Intelligence, vol. 11, no. 5, pp. 490-498, 1989.

[39] Wang, J., Adelson, E., “Layered Representation for Motion Analysis”, Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, New York, NY, pp. 361-366, 1993.

[40] Waxman, A.M., Wu J., Bergholm F. “Convected Activation Profiles and Receptive Fields for Real Time Measurement of Short Range Visual Motion”, Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, Ann Arbor, MI, pp. 717-723, 1988.

[41] Weber, J., Malik, J., “Robust Computation of Optical Flow in a Multi-Scale Differential Framework”, Proceedings of the Fourth International Conference on Computer Vision, Berlin, Germany, 1993.

[42] Werkhoven, P., Koenderink, J.J., “Extraction of Motion Parallax Structure in the Visual System I”, Biological Cybernetics, vol. 63, pp. 185-191, 1990.

[43] Young, G.J., Chellappa, R., “3-D Motion Estimation Using a Sequence of Noisy Stereo Images: Models, Estimation, and Uniqueness Results”, IEEE Transactions on Pattern Analysis and Machine Intelligence, vol. 12, no. 8, pp. 735-759, 1990.

[44] Young, G.J., Chellappa, R., “Statistical Analysis of Inherent Ambiguities in Recovering 3-D Motion from a Noisy Flow Field”, IEEE Transactions on Pattern Analysis and Machine Intelligence, vol. 14, no. 10, pp. 995-1013, 1992.

[45] Young, G-S., “Safe Navigation and Active Vision for Autonomous Vehicles: A Purposive and Direct Solution” Ph.D. Dissertation, University of Maryland, May 1993.

[46] Young, G-S., Hong, T., Herman, M., Yang, J., “New Visual Invariant for Obstacle Detection Using Optical Flow Induced from General Motion”, Proceedings of the IEEE Workshop on Applications of Computer Vision, Palm Springs, CA, pp. 100-109, 1992.

[47] Young, G-S., Hong, T., Herman, M., Yang, J., “Safe Navigation and Active Vision for Autonomous Vehicles: A Purposive and Direct Solution”, Proceedings of the SPIE, vol. 2056: Active Vision and 3D Methods, Boston, MA, pp. 31-42, 1993.

[48] Young, R. A.,“Simulation of Human Retinal Function with the Gaussian Derivative Model”, Proceedings of IEEE Conference on Computer Vision and Pattern Recognition, Miami Beach, FL, pp. 564-569, 1986.