Notes on Motion Estimation

Psych 267/CS 348D/EE 365

Prof. David J. Heeger

October 30, 1996

There are a great variety of applications that depend on analyzing the motion in image sequences. These include motion detection for surveillance, image sequence data compression (MPEG), image understanding (motion-based segmentation, depth/structure from motion), obstacle avoidance, image registration and compositing. The first step in processing image sequences is typically image velocity estimation. The result is called the optical flow field, a collection of two-dimensional velocity vectors, one for each small region (potentially, one for each pixel) of the image.

Image velocities can be measured using correlation or block-matching (for example, see Anandan, 1989) in which each small patch of the image at one time is compared with nearby patches in the next frame. Feature extraction and matching is another way to measure the flow field (for reviews of feature tracking methods see Barron, 1984 or Aggarwal and Nandhakumar, 1988). Gradient-based algorithms are a third approach to measuring flow fields (for example, Horn and Schunk, 1981; Lucas and Kanade, 1981; Nagel, 1987). A fourth approach using spatiotemporal filtering has also been proposed (for example, Watson and Ahumada, 1985; Heeger, 1987; Grzywacz and Yuille, 1990; Fleet and Jepson, 1990). This handout concentrates on the filter-based and gradient-based methods. Emphasis is placed on the importance of multiscale, coarse-to-fine, refinement of the velocity estimates.

1 3D Velocity and Image Velocity

Motion occurs in many applications. Here we concentrate on natural image sequences of 3d scenes in which objects and the camera may be moving. In particular we are interested in measuring the optical flow that arises due to the relative motion between the object and the camera.

To begin, consider a point on the surface of an object. We will represent 3d surface points as position vectors $\vec{X} = (X, Y, Z)^T$ relative to a viewer-centered coordinate frame as depicted in Fig. 1. When the camera moves, or when the object moves, this point moves along a 3d path $\vec{X}(t) = (X(t), Y(t), Z(t))^T$, relative to our viewer-centered coordinate frame. The instantaneous 3d velocity of the point is the derivative of this path with respect to time:

$$\vec{V} = \frac{d\vec{X}(t)}{dt} = \left(\frac{dX}{dt}, \frac{dY}{dt}, \frac{dZ}{dt}\right)^T \tag{1}$$

We are interested in the image locations to which the 3d point projects as a function of time. Under perspective projection, the point $\vec{X}$ projects to the image point $(x, y)^T$ given by

$$\begin{aligned} x &= fX/Z \\ y &= fY/Z, \end{aligned} \tag{2}$$

where f is the "focal length" of the projection. As the 3d point moves through time, its corresponding 2d image point traces out a 2d path $(x(t), y(t))^T$, the derivative of which is the image velocity

$$\vec{\theta} = \left(\frac{dx(t)}{dt}, \frac{dy(t)}{dt} \right)^T \tag{3}$$

We can combine Eqs. 2 and 3 and thereby write image velocity in terms of the 3d position and velocity of the surface point:

$$\vec{\theta} = \frac{1}{Z} \left(\frac{dX}{dt}, \frac{dY}{dt} \right)^T + \frac{1}{Z^2} \frac{dZ}{dt} (X(t), Y(t))^T \tag{4}$$

Each visible surface point traces out a 3d path, which projects onto the image plane to form a 2d path. If one considers the paths of all visible 3d surface points, and their projections onto the image plane, then one obtains a dense set of 2d paths, the temporal derivative of which is the vector field of 2d velocities, commonly known as the *optical flow field*.

To understand the structure of the optical flow field in greater detail it is often helpful to make further assumptions about the structure of the 3d surface or about the form of the 3d motion. We begin by considering the special case in which the objects in the scene move rigidly with respect to the camera, as though the camera was moving through a stationary scene (Longuet-Higgins and Prazdny, 1980; Bruss and Horn, 1983; Waxman and Ullman, 1985; Heeger and Jepson, 1992). This is a special case because all points on a rigid body share the same six motion parameters relative to the camera-centered coordinate frame. In particular, the instantaneous velocity of the camera through a stationary scene can be expressed in terms of the camera's 3d translation $\vec{T} = (T_x, T_y, T_z)^T$, and it instantaneous 3d rotation $\vec{\Omega} = (\Omega_x, \Omega_y, \Omega_z)^T$. Here, the direction of $\vec{\Omega}$ gives the axis of rotation while $|\vec{\Omega}|$ is the magnitude of rotation per unit time. Given this motion of the camera, the instantaneous 3d velocity of a surface point in camera-centered coordinates is

$$\left(\frac{dX}{dt}, \frac{dY}{dt}, \frac{dZ}{dt} \right)^T = - \left(\vec{\Omega} \times \vec{X} + \vec{T} \right) . \tag{5}$$

The 3d velocities of all surface points in a stationary scene depend on the same rigid-body motion parameters, and are given by Eq. 5. When we substitute these 3d velocities for $d\vec{X}/dt$ in Eq. 4 we obtain an expression for the form of the optical flow field for a rigid scene:

$$\vec{\theta}(x, y) = p(x, y)\mathbf{A}(x, y)\vec{T} + \mathbf{B}(x, y)\vec{\Omega} \tag{6}$$

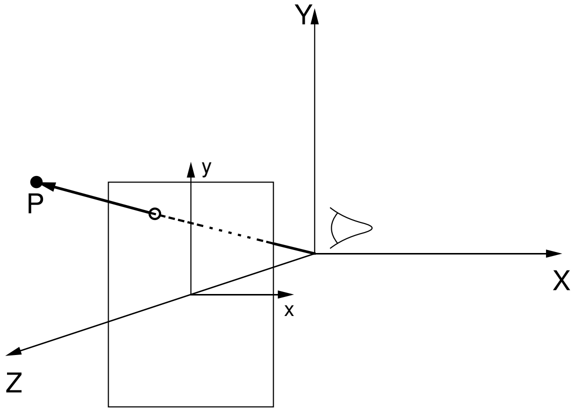
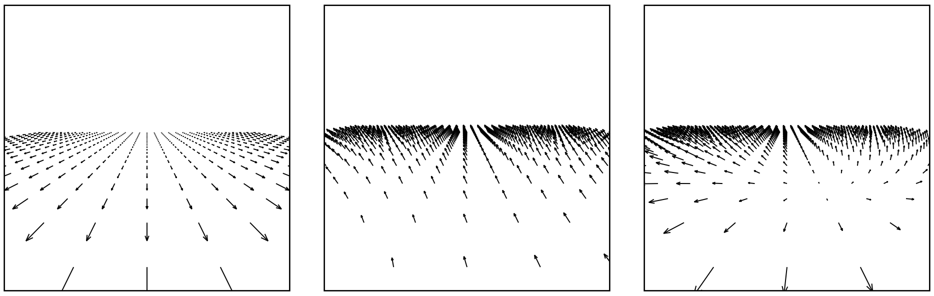


Figure 1: Camera centered coordinate frame and perspective projection.



A B C

Figure 2: Example flow fields. **A:** Camera translation. **B:** Camera rotation. **C:** Translation plus rotation. Each flow vector in C is the vector sum of the two corresponding flow vectors in A and B.

where $p(x, y) = 1/Z(x, y)$ is inverse depth at each image location, and

$$\begin{aligned} \mathbf{A}(x,y) &= \begin{bmatrix} -f & 0 & x \\ 0 & -f & y \end{bmatrix} \\ \mathbf{B}(x,y) &= \begin{bmatrix} (xy)/f & -(f + x^2/f) & y \\ f + y^2/f & -(xy)/f & -x \end{bmatrix}. \end{aligned}$$

The matrices $\mathbf{A}(x, y)$ and $\mathbf{B}(x, y)$ depend only on the image position and the focal length.

Equation 6 describes the flow field as a function of 3D motion and depth. It has two terms. The first term is referred to as the translational component of the flow field since it depends on 3d translation and 3d depth. The second term is referred to as the rotational component since it depends only on 3d rotation. Since $p(x, y)$ (the inverse depth) and $\vec{T}$ (the translation) are multiplied together in Eq. 6, a larger distance (smaller p) or a slower 3D translation (smaller $|\vec{T}|$) both yield a slower image velocity.

Figure 2 shows some example flow fields. Each vector represents the speed and direction of motion for each local image region. Figure 2A is a flow field resulting from camera translation above a planar surface. Figure 2B is a flow field resulting from camera rotation, and Fig. 2C is a flow field resulting from simultaneous translation and rotation. Each flow vector in Fig. 2C is the vector sum of the two corresponding flow vectors is Figs. 2A and B.

Pure Translation. When a camera is undergoing a pure translational motion, features in the image move toward or away from a single point in the image, called the focus of expansion (FOE). The FOE in Fig. 2A is centered just above the horizon.

When the rotation is zero ($\vec{\Omega} = \vec{0}$),

$$\vec{\theta}(x,y) = p(x,y)\mathbf{A}(x,y)\vec{T}$$

i.e.,

$$\begin{aligned} \theta_1(x,y) &= p(x - x_0)T_z \qquad & x_0 &= fT_x/T_z \\ \theta_2(x,y) &= p(y - y_0)T_z \qquad & y_0 &= fT_y/T_z. \end{aligned} \tag{7}$$

The image velocity is zero at image position $(x_0, y_0)^T$; this is the focus of expansion. At any other image position, the flow vector is proportional to $(x,y)^T - (x_0,y_0)^T$, that is, the flow vector points either toward or away from the focus of expansion.

Pure Rotation. When the camera is undergoing a pure rotational motion, the resulting flow field depends only on the axis and speed of rotation, independent of the scene depth/structure (see Fig. 2B for an example). Specifically, rotation results in a quadratic flow field, that is, each component θ_1 and θ_2 of the flow field is a quadratic function of image position. This is evident from the form of $\mathbf{B}(x,y)$ which contains quadratic terms in x and y (e.g., x^2, xy, etc.).

Motion With Respect to a Planar Surface. When the camera is rotating as well as translating, the flow field can be quite complex. Unlike the pure translation case, the singularity in the flow field no longer corresponds to the translation direction. But for planar surfaces, the flow field simplifies to again be a quadratic function of image position (see Fig. 2C for an example).

A planar surface can be expressed in the camera-centered coordinate frame as:

$$Z = m_x X + m_y Y + Z_0.$$

The depth map is obtained by substituting from Eq. 2 to give:

$$Z(x,y) = m_x(x/f)Z(x,y) + m_y(y/f)Z(x,y) + Z_0.$$

Solving for $Z(x,y)$:

$$Z(x,y) = \frac{f\,z_0}{f - xm_x - ym_y},$$

and the inverse depth map is:

$$p(x,y) = \left(\frac{f}{fZ_0}\right) - \left(\frac{m_x}{fZ_0}\right)x - \left(\frac{m_y}{fZ_0}\right)y \tag{8}$$

That is, for a planar surface, the inverse depth map is also planar.

Combining Eqs. 6 and 8 gives:

$$\vec{\theta}(x,y) = \left(\frac{1-m_x x - m_y y}{fZ_0}\right)\begin{pmatrix} -f & 0 & x \\ 0 & -f & y \end{pmatrix}\vec{T} + \mathbf{B}\,\vec{\Omega}. \tag{9}$$

We already know (see above) that the rotational component of the flow field (the second term, $\mathbf{B}\vec{\Omega}$) is quadratic in x and y. It is clear that the translational component (the first term) in the above equation is also quadratic in x and y.

For curved 3d surfaces the translational component, and hence the optical flow field, will be cubic or even higher order. The rotational component is always quadratic.

Occlusions. The optical flow field is often more complex than is implied by the above equations. For example, the boundaries of objects gives rise to occlusions in an image sequence, which cause discontinuities in the optical flow field. A detailed discussion of these issues is beyond the scope of this presentation.

2 Gradient-Based Motion Estimation

The optical flow field described above is a geometric entity. We now turn to the problem of estimating it from the spatiotemporal variations in image intensity. Towards this end, a number of people (Horn and Schunk, 1981; Lucas and Kanade, 1981; Nagel, 1987; and others) have proposed algorithms that compute optical flow from spatial and temporal derivatives of image intensity. This is where we begin.

Intensity Conservation and Block Matching. The usual starting point for velocity estimation is to assume that the intensities are shifted (locally translated) from one frame to the next, and that the shifted intensity values are *conserved*, i.e.,

$$f(x,y,t) = f(x+\theta_1, y+\theta_2, t+1), \tag{10}$$

where $f(x,y,t)$ is image intensity as a function of space and time, and $\vec{\theta} = (\theta_1, \theta_2)$ is velocity. Note that this *intensity conservation* assumption is only approximately true in practice. For example, it ignores possible changes in intensity due to lighting changes.

Then a typical block matching algorithm would proceed, as illustrated in Fig. 3, by choosing a region (e.g., 8x8 pixels) from one frame and shifting it over the next frame to find the best match. One might search for a peak in the cross-correlation

$$\sum f(x,y,t)\, f(x+\theta_1, y+\theta_2, t+1),$$

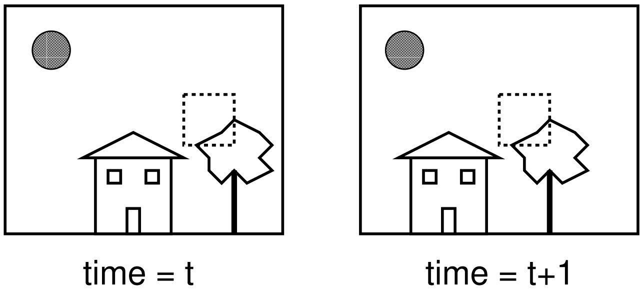


Figure 3: Intensity conservation assumption: the image intensities are shifted (locally translated) from one frame to the next.

or (equivalently) a minimum in the sum-of-squared-differences

$$\sum [f(x,y,t) - f(x+\theta_1, y+\theta_2, t+1)]^2.$$

In both cases, the summation is taken over equal sized blocks in the two frames.

The Gradient Constraint. The principal relationship between intensity derivatives and optical flow is called the gradient constraint. One way that the gradient constraint is often derived is to assume that the time-varying image intensity is well approximated by a first-order Taylor series expansion, i.e.

$$f(x+\theta_1, y+\theta_2, t+1) \approx f(x,y,t) + \theta_1 f_x(x,y,t) + \theta_2 f_y(x,y,t) + f_t(x,y,t),$$

where f_x, f_y, and f_t are the spatial and temporal partial derivatives of image intensity. Substituting this approximation into Eq. 10 gives

$$\theta_1 f_x(x,y,t) + \theta_2 f_y(x,y,t) + f_t(x,y,t) = 0. \tag{11}$$

This equation relates the velocity at one location in the image to the spatial and temporal derivatives of image intensity at that same location. Eq. 11 is called the gradient constraint, and it is illustrated in Fig. 4.

Intensity Conservation. A different way of deriving the gradient constraint can be found if we consider the 2d paths in the image plane, $(x(t), y(t))^T$, along which the intensity is constant. In other words, imagine that we are tracking points of constant intensity from frame to frame. It is the derivative of their paths that give us the velocity.

More formally, a path $(x(t), y(t))^T$ along which intensity is equal to a constant c must satisfy the equation

$$f(x(t), y(t), t) = c\ . \tag{12}$$

As explained above, assuming that the path is smooth, the temporal derivative of the path $(x(t), y(t))^T$ gives us the optical flow. To obtain a constraint on optical flow we therefore differentiate equation 12:

$$\frac{d\, f(x(t), y(t), t)}{d\, t} = 0 \tag{13}$$

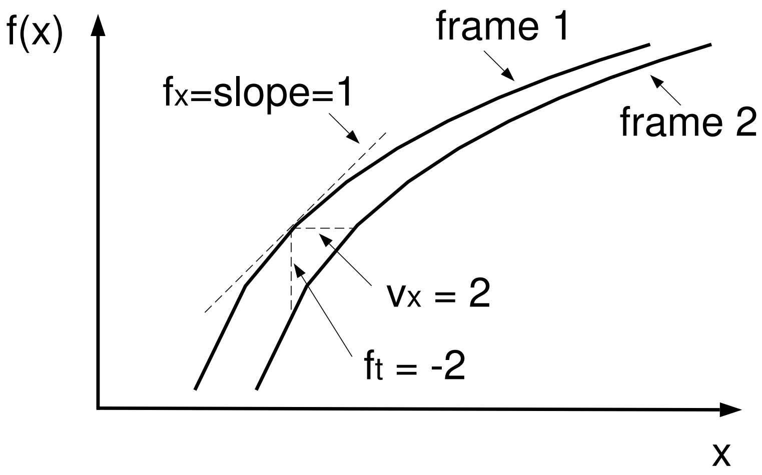


Figure 4: The gradient constraint relates the velocity at one point in the image to the spatial and temporal derivatives of image intensity. In one-dimension: $\theta_1 \, f_x + f_t = 0$.

The right hand side is zero because c is a constant. Because the left-hand-side is a function of 3 variables which all depend on time t, we take the total derivative and use the chain rule to obtain:

$$\begin{aligned} \frac{d\,f(x(t), y(t), t)}{d\,t} &= \frac{\partial f}{\partial x}\frac{d\,x}{d\,t} + \frac{\partial f}{\partial y}\frac{d\,y}{d\,t} + \frac{\partial f}{\partial t}\frac{d\,t}{d\,t} \\ &= f_x \theta_1 + f_y \theta_2 + f_t \end{aligned} \tag{14}$$

In this last step the partial derivatives were rewritten as f_x, f_y, and f_t, and the velocities were substituted for the path derivatives (i.e. $\theta_1 = dx/dt$ and $\theta_2 = dy/dt$). Finally, if we combine Eqs. 13 and 14 we obtain the gradient constraint.

Equation 13 is often referred to as an *intensity conservation* constraint. In terms of the 3d scene, this conservation assumption implies that the image intensity associated with the projection of a 3d point does not change as the object or the camera move. Hence, this assumption is often violated in real image sequences.

Combining Constraints. It is impossible to recover velocity, given the gradient constraint at only a single position, since Eq. 11 offers only one linear constraint to solve for the two unknown components of velocity. Gradient-based methods solve for velocity by combining information over a spatial region. The different gradient-based methods use different combination rules. A particularly simple rule for combining constraints from two nearby spatial positions is:

$$\begin{bmatrix} f_x(x_1, y_1, t) & f_y(x_1, y_1, t) \\ f_x(x_2, y_2, t) & f_y(x_2, y_2, t) \end{bmatrix} \begin{bmatrix} \theta_1 \\ \theta_2 \end{bmatrix} + \begin{bmatrix} f_t(x_1, y_1, t) \\ f_t(x_2, y_2, t) \end{bmatrix} = \vec{0}, \tag{15}$$

where the two coordinate pairs (x_i, y_i) correspond to the two spatial positions. Each row of Eq. 15 is the gradient constraint for one spatial position. Solving this equation simultaneously for both spatial positions gives the velocity that is consistent with both constraints.

A better option is to combine constraints from more than just two spatial positions. When we do this we will not be able to exactly satisfy all of the gradient constraints simultaneously because there will be more constraints than unknowns. To measure the extent to which the gradient constraints are not satisfied we square each constraint and sum them:

$$E(\theta_1, \theta_2) = \sum_{x,y} [\theta_1 f_x(x,y,t) + \theta_2 f_y(x,y,t) + f_t(x,y,t)]^2. \tag{16}$$

Each squared term in the summation is a constraint on the flow from a different (nearby) position. The summation is taken over a local spatial region, e.g., in a Gaussian weighted window. Since there are now more constraints than unknowns, there may not be a solution that satisfies all of the constraints simultaneously. In other words, $E(\theta_1, \theta_2)$ will typically be non-zero for all (θ_1, θ_2). The choice of (θ_1, θ_2) that minimizes $E(\theta_1, \theta_2)$ is the least squares estimate of velocity.

Least Squares Estimate. Since Eq. 16 is a quadratic expression, there is a simple analytical expression for the velocity estimate. The solution is derived by taking derivatives of Eq. 16 with respect to θ_1 and θ_2, and setting them equal to zero:

$$\begin{aligned} \frac{\partial E(\theta_1, \theta_2)}{\partial \theta_1} = \sum_{xy} [\theta_1 (f_x)^2 + \theta_2 (f_x f_y) + (f_x f_t)] &= 0 \\ \frac{\partial E(\theta_1, \theta_2)}{\partial \theta_2} = \sum_{xy} [\theta_2 (f_y)^2 + \theta_1 (f_x f_y) + (f_y f_t)] &= 0 \end{aligned}$$

These equations may be rewritten as a single equation in matrix notation:

$$\mathbf{M} \cdot \vec{\theta} + \vec{b} = \vec{0},$$

where the elements of $\mathbf{M}$ and $\vec{b}$ are:

$$\mathbf{M} = \begin{pmatrix} \sum (f_x)^2 & \sum (f_x f_y) \\ \sum (f_x f_y) & \sum (f_y)^2 \end{pmatrix}$$

$$\vec{b} = \begin{pmatrix} \sum (f_x f_t) \\ \sum (f_y f_t) \end{pmatrix}.$$

The least-squares solution is then given by

$$\hat{\theta} = -\mathbf{M}^{-1} \vec{b}, \tag{17}$$

presuming that $\mathbf{M}$ is invertible.

Implementation with Image Operations. Remember that we want to compute optical flow at every spatial location. Therefore we will need the matrix $\mathbf{M}$ and the vector $\vec{b}$ at each location. In particular, we should rewrite our matrix equation as

$$\mathbf{M}(x,y) \cdot \vec{\theta}(x,y) + \vec{b}(x,y) = \vec{0}\ .$$

As above in Eq. 17, the elements in $\mathbf{M}(x,y)$ and $\vec{b}(x,y)$ at position $(x,y)^T$ are local weighted summations of the intensity derivatives. What we intend to show here is that the elements of $\mathbf{M}(x,y)$ and $\vec{b}(x,y)$ can be computed in a straightforward way using convolutions and image point-operations.

To begin, let $g(x,y)$ be a Gaussian window sampled on a square grid of width w. Then at a specific location $(x,y)^T$ the elements of the matrix $\mathbf{M}(x,y)$ and the vector $\vec{b}(x,y)$ are given by:

$$\mathbf{M} = \begin{pmatrix} \sum_a \sum_b g(a,b) f_x^2(x-a,y-b) & \sum_a \sum_b g(a,b) f_x(x-a,y-b) f_y(x-a,y-b) \\ \sum_a \sum_b g(a,b) f_x(x-a,y-b) f_y(x-a,y-b) & \sum_a \sum_b g(a,b) f_y^2(x-a,y-b) \end{pmatrix}$$

$$\vec{b} = \begin{pmatrix} \sum_a \sum_b g(a,b) f_x(x-a,y-b) f_t(x-a,y-b) \\ \sum_a \sum_b g(a,b) f_y(x-a,y-b) f_t(x-a,y-b) \end{pmatrix}.$$

where each summation is over the width of the Gaussian, from $-w$ to w. Although this notation is somewhat cumbersome, it serves to show explicitly that each element of the matrix (as a function of spatial position) is actually equal to the convolution of $g(x,y)$ with products of the intensity derivatives. To compute the elements of $\mathbf{M}(x,y)$ and $\vec{b}(x,y)$ one simply applies derivative filters to the image, takes products of their outputs, and then blurs the results with a Gaussian. For example, the upper-left component of $\mathbf{M}(x,y)$, ie. $m_{11}(x,y)$, is computed by: (1) applying an x-derivative filter, (2) squaring the result at each pixel, (3) blurring with a Gaussian lowpass filter.

Aperture Problem. When the matrix $\mathbf{M}$ in Eq. 17 is singular (or ill-conditioned), there are not enough constraints to solve for both unknowns. This situation corresponds to what has been called the "aperture" problem. For some patterns (e.g., a gradual curve) there is not enough information in a local region (small aperture) to disambiguate the true direction of motion. For other patterns (e.g., an extended grating or edge) the information is insufficient regardless of the aperture size. Of course, the problem is not really due to the aperture, but to the one-dimensional structure of the stimulus.

The latter case is illustrated in Fig. 5A. The diagonal line in velocity space indicates the locus of velocities compatible with the motion of the grating. At best, we may extract only one of the two velocity components. Figure 5B shows how the motion is disambiguated when there is more spatial structure. The plaid pattern illustrated in Fig. 5B is composed of two moving gratings. The lines give the possible motion of each grating alone. Their intersection is the only shared motion.

Combining the gradient constraints according to the summation in Eq. 15 is related to the intersection of constraints rule depicted in Fig. 5. The gradient constraint, Eq. 11, is linear in both θ_1 and θ_2. Given measurements of the derivatives, (f_x, f_y, f_t), there is a line of possible solutions for (θ_1, θ_2), analogous to the constraint line illustrated in Fig. 5A. For each different position, there will generally be a different constraint line. Equation 15 gives the intersection of these constraint lines, analogous to Fig. 5B.

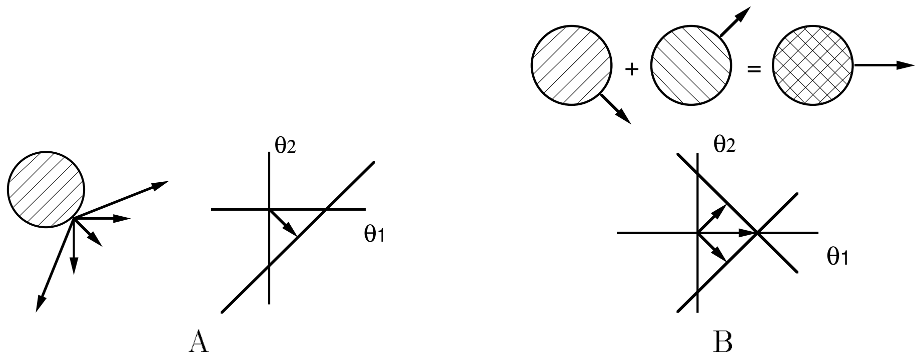


Figure 5: Aperture problem. **A:** Single moving grating viewed through a circular aperture. The diagonal line indicates the locus of velocities compatible with the motion of the grating. **B:** Plaid composed of two moving gratings. The lines give the possible motion of each grating alone. Their intersection is the only shared motion.

3 Space-Time Filters and the Frequency Domain

A number of models of biological motion sensing have been based on direction selective, spatiotemporal linear operators (e.g., Watson and Ahumada, 1985; Adelson and Bergen, 1985; Heeger, 1987; Grzywacz and Yuille, 1990). The concept underlying these models is that visual motion is like orientation in space-time, and that spatiotemporally-oriented, linear operators can be used to detect and measure it.

Figure 6 shows a simple example. Figure 6A depicts a vertical bar moving to the right. Imagine that we stack the consecutive frames of this image sequence one after the next. We end up with a three-dimensional volume (space-time cube) of intensity data like that shown in figure 6B. Figure 6C shows an x-t slice through this space-time cube. The slope of the edges in the x-t slice equals the horizontal component of the bar's velocity (change in position over time). Different speeds correspond to different slopes.

Following Adelson and Bergen (1986), and Simoncelli (1993) we now show that the gradient-based solution can be expressed in terms of the outputs of a set of space-time oriented linear filters. To this end, note that the derivative operations in the gradient constraint may be written as convolutions. Furthermore, we can prefilter the stimuli to extract some spatiotemporal subband, and perform the analysis on that subband. Consider, for example, prefiltering with a space-time Gaussian function. Abusing the notation somewhat, we define:

$$f_x(x,y,t) \;\equiv\; \tfrac{\partial}{\partial x}[g(x,y,t) * f(x,y,t)] \;=\; g_x(x,y,t) * f(x,y,t),$$

where $*$ is convolution and g_x is the x-derivative of a Gaussian. In words, we compute f_x by convolving with $g_x = \partial g/\partial x$, a spatiotemporal linear filter. We compute f_y and f_t similarly.

Then the gradient constraint can be rewritten as:

$$(\theta_1 g_x + \theta_2 g_y + g_t) * f = 0, \tag{18}$$

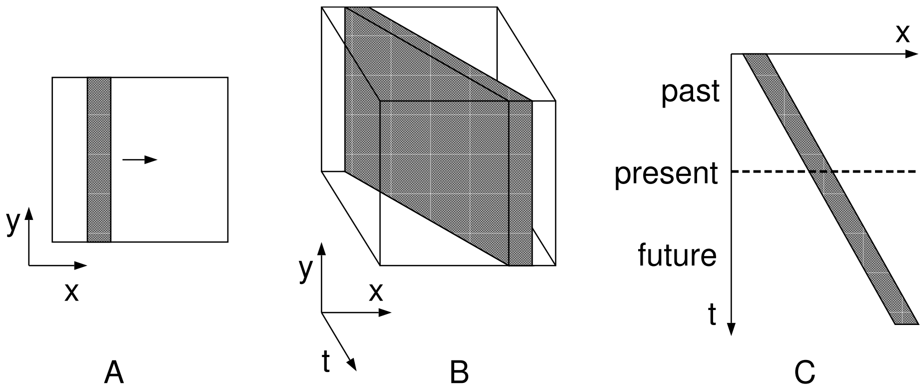


Figure 6: Orientation in space-time (based on an illustration by Adelson and Bergen, 1985). **A:** A vertical bar translating to the right. **B:** The space-time cube of image intensities corresponding to motion of the vertical bar. **C:** An x-t slice through the space-time cube. Orientation in the x-t slice is the horizontal component of velocity. Motion is like orientation in space-time, and spatiotemporally oriented filters can be used to detect and measure it.

where g_x, g_y, and g_t are the space-time derivative filters. For a fixed $\vec{\theta}$, Eq. 18 is the convolution of a space-time filter, namely $(\theta_1 g_x + \theta_2 g_y + g_t)$, with the image sequence, $f(x, y, t)$. This filter is a weighted sum of the three "basis" derivative filters (g_x, g_y, and g_t), and hence, is itself a directional derivative in some space-time direction. The gradient constraint states that there is some space-time orientation for which the directional derivative is zero. This filter-based interpretation of the gradient constraint is depicted (in one spatial dimension) in Fig. 7.

Frequency Domain Analysis of the Gradient Constraint. The gradient constraint for image velocity estimation also has a simple interpretation as regression in the spatiotemporal Frequency domain.

The power spectrum of a translating two-dimensional pattern lies on a plane in the Fourier domain (Watson and Ahumada, 1983, 1985), and the tilt of this plane depends on velocity. An intuition for this result can be obtained by first considering each spatiotemporal frequency component separately. The spatial frequency of a moving sine grating is expressed in cycles per pixel and its temporal frequency is expressed in cycles per frame. Velocity, which is distance over time (or pixels per frame), equals the temporal frequency divided by the spatial frequency. More formally, a drifting 1d sinusoidal grating can be expressed in terms of spatial frequency and velocity as

$$f(x, t) = \sin(w_x(x - \theta_1 t)) \tag{19}$$

where w_x is the spatial frequency of the grating and θ_1 is the velocity. It can also be expressed in terms of spatial frequency w_x and temporal frequency w_t as

$$f(x, t) = \sin(w_x x + w_t t) \tag{20}$$

The relationship between frequency and velocity can be derived algebraically by setting

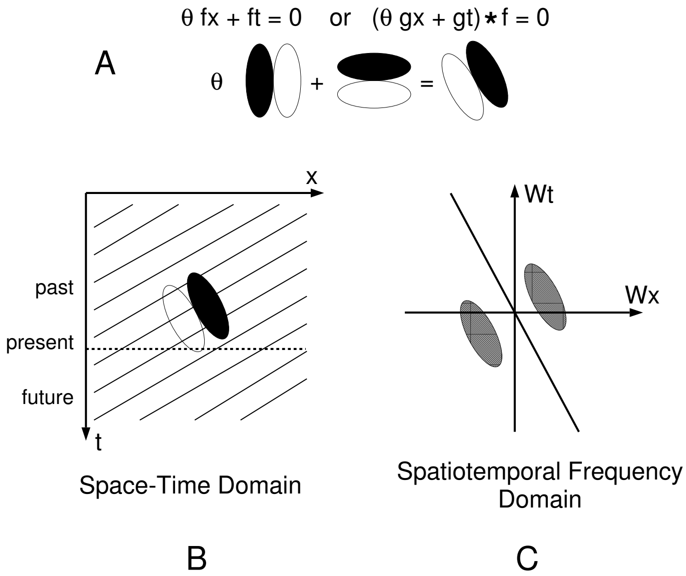


Figure 7: Space-time filters and the gradient constraint. **A:** The gradient constraint states that there is some space-time orientation for which the directional derivative is zero. **B:** Pattern moving from right to left and the space-time derivative filter with the orthogonal space-time orientation so that convolving the two gives a zero response. **C:** Spatiotemporal Fourier transform of bf B. The power spectrum of the translating pattern (the line passing through the origin) and the frequency response of the derivative filter (the pair of symmetrically positioned blobs); the product of the two is zero.

$w_x(x - \theta_1 t) = w_x x + w_t t$. From this one can show that

$$\theta_1 = -w_t/w_x \tag{21}$$

Now consider a one-dimensional spatial pattern that has many spatial-frequency components (e.g., made up of parallel bars, edges, or random stripes), translating with speed θ_1. Each frequency component has a spatial frequency w_x and a temporal frequency w_t but all of the Fourier components share the same velocity so each component satisfies

$$\theta_1\, w_x + w_t = 0.$$

Analogously, two dimensional patterns (arbitrary textures) translating in the image plane occupy a plane in the spatiotemporal frequency domain:

$$\theta_1\, w_x + \theta_2\, w_y + w_t = 0.$$

For example, the expected value of the power spectrum of a translating white noise texture is a constant within this plane and zero outside of it. The plane is perpendicular to the vector $(\theta_1, \theta_2, 1)^T$.

Formally, consider a space-time image sequence $f(x, y, t)$ that we construct by translating a 2d image $f_0(x, y)$ with a constant velocity $\vec{\theta}$. The image intensity over time may be written as

$$f(x, y, t) = f_0(x - \theta_1 t,\ y - \theta_2 t).$$

At time $t = 0$ the spatiotemporal image is equal to $f_0(x, y)$. The three-dimensional Fourier transform of $f(x, y, t)$ is:

$$F(w_x, w_y, w_t) = \int \int \int_{-\infty}^{\infty} f(x, y, t) e^{-j2\pi(xw_x + yw_y + tw_t)} dx dy dt. \tag{22}$$

To simplify this expression, we substitute $f_0(x - \theta_1 t, y - \theta_2 t)$ for $f(x, y, t)$ and arrange the order of integration so that time is integrated last:

$$F(w_x, w_y, w_t) = \int \left[\int \int_{-\infty}^{\infty} f_0(x - \theta_1 t, y - \theta_2 t) e^{-j2\pi(xw_x + yw_y)} dx dy \right] e^{-j2\pi t w_t} dt. \tag{23}$$

The term inside the square brackets is the 2d Fourier transform of $f_0(x - \theta_1 t,\ y - \theta_2 t)$ which can be simplified using the Fourier shift property to yield

$$\begin{aligned} F(w_x, w_y, w_t) &= \int_{-\infty}^{\infty} F_0(w_x, w_y) e^{-j2\pi(\theta_1 t w_x + \theta_2 t w_y)} e^{-j2\pi t w_t} dt \\ &= F_0(w_x, w_y) \int_{-\infty}^{\infty} e^{-j2\pi t(\theta_1 w_x + \theta_2 w_y)} e^{-j2\pi t w_t} dt \end{aligned} \tag{24}$$

where $F_0(w_x, w_y)$ is the 2d Fourier transform of $f_0(x, y)$. The Fourier transform of a complex exponential is a Dirac delta function, and therefore this expression simplifies further to

$$F(w_x, w_y, w_t) = F_0(w_x, w_y)\, \delta(\theta_1 w_x + \theta_2 w_y + w_t)$$

There are many important reasons to understand motion in the frequency domain. For example, if the motion of an image region may be approximated by translation in the image plane then the velocity of the region may be computed by finding the plane (in the Fourier domain) in which all the power resides. Figure 7 illustrates how a derivative filter can be used to determine which plane contains the power. The frequency response of a spatiotemporal derivative filter is a symmetrically positioned pair of "blobs". The gradient constraint states that the blobs can be positioned (by rotating them around the origin) so that multiplying the Fourier transform of the image sequence (the plane) times the frequency response of the filter (blobs) gives zero.

Formally, we rely on Parseval's theorem and the derivative theorem of the Fourier transform to write the gradient method in the frequency domain. Parseval's theorem states that the sum of the squared values in space-time equals the sum of the power in the frequency domain:

$$\sum_{x,y,t} |f(x, y, t)|^2 = \sum_{w_x, w_y, w_t} |F(w_x, w_y, w_t)|^2. \tag{25}$$

The derivative theorem states that the Fourier transform of the derivative of f equals the Fourier transform of f itself multiplied by an imaginary ramp function of the form $-jw$. For example,

$$\mathcal{F}[f_x(x, y, t)] = -jw_x \mathcal{F}[f(x, y, t)] = -jw_x F(w_x, w_y, w_t) \tag{26}$$

Combining Eqs. 16, 25, and 26, gives:

$$E(\theta_1, \theta_2) = \sum_{x,y,t} |\theta_1 f_x(x, y, t) + \theta_2 f_y(x, y, t) + f_t(x, y, t)|^2 \tag{27}$$

$$
\begin{aligned}
&= \sum_{w_x,w_y,w_t} |\theta_1 w_x F(w_x,w_y,w_t) + \theta_2 w_y F(w_x,w_y,w_t) + w_t F(w_x,w_y,w_t)|^2 \\
&= \sum_{w_x,w_y,w_t} [(\theta_1,\theta_2,1)\cdot\vec{w}]^2\,|F(\vec{w})|^2.
\end{aligned}
$$

If all of the assumptions and approximations hold precisely (notably, that the image is uniformly translating for all space and time), then this expression can be precisely minimized by choosing $\vec{\theta}$ to satisfy: $\theta_1 w_x + \theta_2 w_y + w_t = 0$. If any of the approximations are violated then there will not in general be any $\vec{\theta}$ that will satisfy the constraint perfectly. Minimizing Eq. 27 is a weighted, least-squares regression.

To be concrete, let $\vec{w}_i = (w_{xi}, w_{yi}, w_{ti})$ for $1 < i < N$ be the spatiotemporals frequencies for which we have samples of $|F(\vec{w})|$. Rewriting Eq. 27 in matrix notation gives:

$$
E(\theta_1,\theta_2) = \|\mathbf{A}\vec{u}\|^2, \tag{28}
$$

where $\vec{u} = (\theta_1,\theta_2,1)^T$ and

$$
\mathbf{A} = \begin{pmatrix} F(\vec{w}_1)w_{x1} & F(\vec{w}_1)w_{y1} & F(\vec{w}_1)w_{t1} \\ F(\vec{w}_2)w_{x2} & F(\vec{w}_2)w_{y2} & F(\vec{w}_2)w_{t2} \\ \vdots & \vdots & \vdots \\ F(\vec{w}_N)w_{xN} & F(\vec{w}_N)w_{yN} & F(\vec{w}_N)w_{tN} \end{pmatrix}
$$

Because of the weights, $|F(\vec{w})|$, the regression depends most on those frequency components where the power is concentrated. Equation 28 is minimized by a vector $\vec{v}$ in the direction of the eigenvector corresponding to the smallest eigenvalue of $\mathbf{A}^T\mathbf{A}$. The image velocity estimate $\hat{\vec{u}}$ is obtained by normalizing $\vec{v}$ so that its third element is 1.

For real image sequences, the least-squares estimate of image velocity (Eq. 17) is definitely preferred over the frequency domain regression estimate (Eq. 28) because the frequency domain estimate requires computing a global Fourier transform. But for an image sequence containing a uniformly translating pattern, the two estimates are the same.

4 Gradient Constraint and Image Preprocessing

Some degree of preprocessing before applying the gradient constraint is generally necessary to produce reliable estimates of optical flow. It is common for example to smooth the image sequence with a lowpass filter before estimating derivatives, as in Eq. 3. Here we briefly consider more elaborate forms of preprocessing.

Using Higher Order Derivatives. One simple extension involves the use of bandpass prefiltering. The original image sequence might be filtered to enhance image structure that is of particular interest, or to suppress structure that one wishes to ignore. One may also do this to obtain further constraints on the optical flow at a single pixel.

For example, one could prefilter the image with first derivative filters g_x and g_y. Following Nagel et al (1987) and Verri et al (1990), the gradient constraint could then be applied to f_x and f_y (instead of f in as was done above in Eq. 11). This yields two constraints on the optical flow at each location:

$$\begin{aligned} \theta_1 f_{xx} + \theta_2 f_{xy} + f_{xt} &= 0 \\ \theta_1 f_{xy} + \theta_2 f_{yy} + f_{yt} &= 0, \end{aligned} \tag{29}$$

Alternatively, you might use second-order Gaussian derivatives g_{xx}, g_{xy} and g_{yy} as pre-filters. In this case you obtain three gradient constraints at each pixel:

$$\begin{aligned} \theta_1 f_{xxx} + \theta_2 f_{xxy} + f_{xxt} &= 0 \\ \theta_1 f_{xxy} + \theta_2 f_{xyy} + f_{xyt} &= 0 \\ \theta_1 f_{xyy} + \theta_2 f_{yyy} + f_{yyt} &= 0, \end{aligned} \tag{30}$$

where f_{xxx} should be interpreted as $f * g_{xxx}$, and likewise for the other derivatives. Equation 30, written in terms of third derivatives, gives three constraints on velocity.

An advantage of using higher order derivatives is that, in principle, there are more constraints at a single spatial position. Even so, it is typically worthwhile combining constraints over a local spatial region and computing a least-squares estimate of velocity by minimizing:

$$\begin{aligned} E(\theta_1, \theta_2) &= \sum_{x,y} [\theta_1 f_{xxx} + \theta_2 f_{xxy} + f_{xxt}]^2 \\ &+ \sum_{x,y} [\theta_1 f_{xxy} + \theta_2 f_{xyy} + f_{xyt}]^2 \\ &+ \sum_{x,y} [\theta_1 f_{xyy} + \theta_2 f_{yyy} + f_{yyt}]^2 \end{aligned}$$

The solutions can be written in the same form as Eq. 17. The only differences are: (1) that the third derivative formulation uses a greater number of linear filters, and (2) that the third derivative filters are more narrowly tuned for spatiotemporal orientation.

The intensity gradient constraint in Eq. 11 is based on the intensity conservation assumption; it assumes that intensity is conserved (only shifting from one location to another location) over time. The third derivative constraints, Eq. 30, are based on conservation of the second spatial derivatives of intensity, i.e., that f_{xx}, f_{xy}, and f_{yy} are conserved over time.

Nonlinear Pre-Processing. The gradient constraint need not be applied directly to the image intensities or linearly filter versions of the image. One could apply various forms of nonlinear preprocessing instead. There are two approximations underlying the gradient constraint (see above): (1) the image intensity conservation assumption, and (2) the numerical approximation of first-order partial derivatives in space and time. With raw image intensities the intensity conservation assumption is often violated, and derivatives can be difficult to estimate reliably. However, with appropriate preprocessing of the image sequence one may satisfy the conservation approximation more closely and also improve the accuracy of numerical differentiation. Clever preprocessing (for example, local automatic gain control to compensate for changes in illumination) can potentially make a significant difference.

One successful example of nonlinear preprocessing, proposed by Fleet and Jepson (1990, 1991), is to use the phase output of quadrature-pair linear filters. The phase signal is nearly linear in local space-time neighborhoods of an image sequence. The extent of the linearity can be shown to depend on the bandwidth of the filters; narrow bandwidth filters produce linear phase behavior over larger regions. Phase has also been shown to be stable with respect to small affine deformations of the image (e.g., scaling the image intensities and/or adding a constant to the image intensities), so that local phase is more often conserved over time. Fleet and Jepson (1993) also showed that phase is occasionally very unstable, and provided a simple method for detecting places where the conservation assumption is likely to be violated.

5 Multi-Scale Motion Estimation

Temporal Aliasing. The image sequences we are given as input often have temporal sampling rates that are slower than that required by the sampling theorem to uniquely represent the high temporal frequencies in the signal. As a consequence, temporal aliasing is a common problem in computing image velocity in real image sequences. The classic example of temporal aliasing is the wagon wheel effect in old Western movies; when the wheel rotates at the right speed relative to the frame rate, its direction of rotation appears to reverse. This effect can most easily be understood by considering the closest match for a given spoke as the wheel rotates from one frame to the next. Let the angular spacing between spokes be α. If the wheel rotates by more than $\alpha/2$ but less than α from one frame to the next, then the wheel will appear to rotate backwards. In fact, any integer multiple of this range will also do so that rotations between $k\,\alpha/2$ and $k\,\alpha$ will appear to rotate backwards (for any integer k).

Temporal aliasing can be a problem for any type of motion algorithm. For feature-based matching algorithms, temporal aliasing gives rise to false matches. For filtering (and gradient-based) methods, temporal aliasing becomes a problem whenever the motion is large compared to the spatial extent of the filters. With phase-based methods the problems are also evident because phase is only defined uniquely with a single wavelength of a bandpass signal. If the signal shifts by more than one wavelength then the nearest location in the next image with the same phase will not yield the correct match.

The effect of temporal aliasing may also be characterized in the spatiotemporal frequency domain, as illustrated in Fig. 8. Consider a one-dimensional signal moving at a constant velocity. As mentioned above, the power spectrum of this signal lies on a line through the origin. Temporal sampling causes a replication of the signal spectrum at temporal frequency intervals of $2\pi/T$ radians, where T is the time between frames. It is easy to see that these replicas could confuse a gradient-based motion estimation algorithm because the derivative filters may respond as much (or more) to the replicated spectra as they do to the original

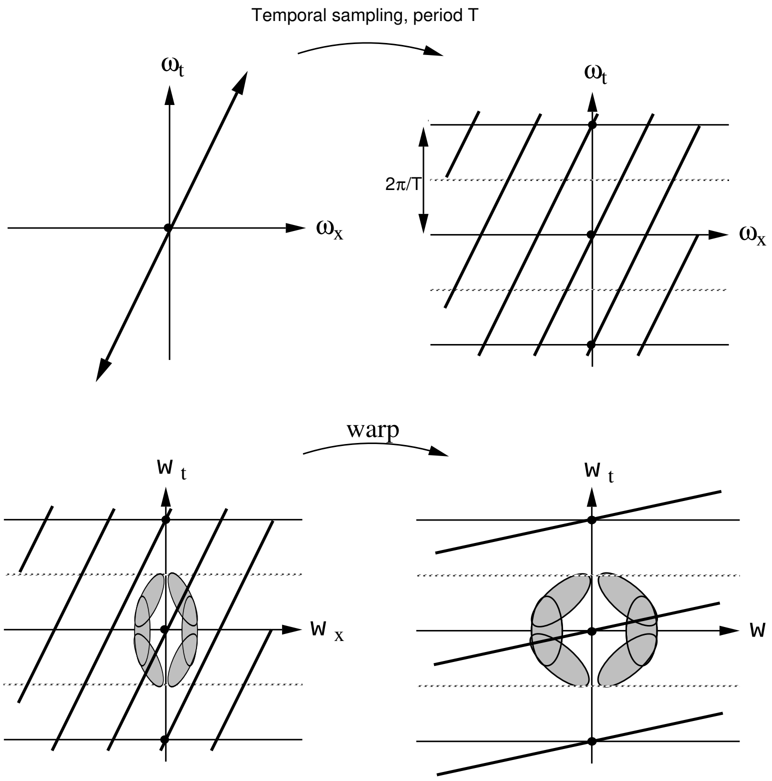


Figure 8: Top-left: The power spectrum of a translating signal occupies a line in the spatiotemporal frequency domain. **Top-right:** Sampling in time (to get the discrete frames of an image sequence) gives rise to replicas of the spectra, that may result in temporal aliasing. These replicas can cause severe problems for a gradient-based motion algorithm because the derivative filters may respond as much (or more) to the replicated spectra as they do to the original. **Bottom-left:** The problem can be alleviated by using coarse-scale derivative filters tuned for lower frequencies, i.e., by blurring the images a lot before computing derivatives. The frequency responses of these coarse-scale filters avoid the replicas. **Bottom-right:** The velocity estimates from the coarse scale are used to warp the images, thereby undoing most of the motion. The finer scale filters can now be used to estimate the residual motion; since the velocities (after warping) are slower, the frequency responses of the fine scale filters may now avoid the replicas.

Eliminating Temporal Aliasing by Coarse-to-Fine Warping. An important observation concerning this type of aliasing is that it only affects the higher spatial frequencies of an image. In particular, for a fixed velocity, those spatial frequencies moving more than half of their period per frame will be aliased, but lower spatial frequencies will not.

This suggests a simple but effective approach for avoiding the problem of temporal aliasing. Estimate the velocity using coarse-scale filters (i.e., spatially blur the individual frames of the sequence before applying the derivative filters) to avoid the potentially aliased high spatial frequencies. These velocity estimates will correspond to the average velocity over a large region of the image because the coarse-scale filters are large in spatial extent.

Given a uniformly translating image sequence, we could stop here. But in typical image sequences, the velocity varies from one point to the next. The coarse-scale filters miss much of this spatial variation in the flow field. To get better estimates of local velocity, we need finer scale (higher frequency) filters with smaller spatial extents.

Following Bergen *et al.* (1992), the coarse motion estimates are used to warp the images and "undo" the motion, roughly *stabilizing* the objects and features in the images. Then finer scale filters are used to estimate the residual motion; since the velocities (after warping) are slower, the frequency responses of the fine scale filters may now avoid the temporally aliased replicas (as illustrated in Fig. 8C). Typically, this is all done with an image pyramid data structure.

Iterative Refinement of Velocity Estimates. This sort of incremental estimation of optical flow with a coarse-to-fine strategy is one example of iteratively refining and updating velocity measurements. The process involves warping the images according to the current velocity estimate and then re-estimating the velocity from the warped sequence. The warping is an attempt to remove the motion and therefore *stabilize* the image sequence. This is a useful idea within a coarse-to-fine strategy to avoid the adverse effects of temporal aliasing as mentioned above. It can also be used to refine the flow estimates without necessarily changing scales. Remember that derivative estimates are often noisy, and the assumptions upon which the velocity estimates were based do not hold exactly. Therefore the velocity estimates can often be improved by iterative refinement.

Here we address the general problem: Given an estimate of the optical flow field $\vec{\theta}^0 = (\theta_1^0, \theta_2^0)^T$ (e.g., it might have been estimated from the coarse scale), we create a warped image sequence $h(x, y, t)$:

$$h(x, y, \Delta t) = \mathcal{W}[f(x, y, t + \Delta t), \vec{\theta}^0(x, y)] = f(x - \theta_1^0 \Delta t, y - \theta_2^0 \Delta t, t + \Delta t),$$

where Δt is the time between two consecutive frames, and $\mathcal{W}[f, \vec{\theta}^0]$ denotes the warping operation that warps image f by the vector field $\vec{\theta}^0$. Note that warping need only be done only over a range of Δt that are required for the temporal differentiation filter.

The warped image sequence $h(x, y, t)$ is then used to estimate the *residual flow field*:

$$\Delta\vec{\theta} = \mathbf{M}^{-1}\,\vec{b}$$

where $\mathbf{M}$ and $\vec{b}$ are computed by taking derivatives (via convolution as above) of $h(x, y, t)$. Given the residual flow estimates $\Delta\vec{\theta}$, the refined velocity estimates become

$$\vec{\theta}^1(x, y) = \vec{\theta}^0(x, y) + \Delta\vec{\theta}(x, y).$$

This new flow estimate is then used to rewarp the original sequence, and the residual flow may be estimated again. In the coarse-to-fine process this new estimate of the flow field is used to warp the images at the next finer scale. This process continues until the finest scale is reached and the residual flow is sufficiently small.

6 Model-Based Motion Estimation

The multiscale gradient/derivative methods described above compute velocity estimates separately for each pixel. They compute the best estimate of translational optical flow in each local neighborhood of the image. Alternatively, one may wish to estimate more complex parametric models over much larger image neighborhoods. For example, camera motion with respect to a planar surface results in a quadratic flow field. If the entire image happens to correspond to a planar (or nearly planar) surface then you can take advantage of this information. Even if the image does not correspond to a planar surface, a model of the flow field may be a reasonable approximation in local neighborhoods. Here we consider an affine flow model, but quadratic flows can be handled similarly.

Following Bergen *et al.* (1992), an affine flow field can be expressed as:

$$\begin{aligned} \theta_1(x, y) &= p_1 + p_2 x + p_3 y \\ \theta_2(x, y) &= p_4 + p_5 x + p_6 y \end{aligned}$$

where $\vec{p} = (p_1, p_2, p_3, p_4, p_5, p_6)^T$ are the parameters to be estimated. This can be written in matrix notation as:

$$\vec{\theta}(x, y) = \mathbf{A}(x, y)\,\vec{p} \tag{31}$$

where

$$\mathbf{A}(x, y) = \begin{pmatrix} 1 & x & y & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & x & y \end{pmatrix}$$

In this instance, the affine model describes the flow field about an origin at location (0,0). if one wished to consider an affine model about the point $(x_0, y_0)^T$ then one might use

$$\mathbf{A}(x, y) = \begin{pmatrix} 1 & x - x_0 & y - y_0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & x_0 & y - y_0 \end{pmatrix}$$

The gradient constraint may be written:

$$\vec{f_s} \cdot \vec{\theta} + f_t = 0, \tag{32}$$

where $\vec{f_s} = (f_x, f_y)^T$ are the spatial derivatives. Combining Eqs. 31 and 32 gives:

$$\vec{f_s}^T \mathbf{A} \vec{p} + f_t = 0,$$

so we want to minimize:

$$E(\vec{p}) = \sum_{x,y} (\vec{f}_s^T \mathbf{A}\vec{p} + f_t)^2, \tag{33}$$

where the summation may be taken over the entire image (or over some reasonably large image region). Equation 33 is quadratic in $\vec{p}$ so the estimate $\hat{\vec{p}}$ is derived in the usual (linear least-squares) way by taking derivatives of E with respect to $\vec{p}$. This yields the solution

$$\hat{\vec{p}} = \mathbf{R}^{-1}\,\vec{s}, \tag{34}$$

where

$$\mathbf{R} = \sum \mathbf{A}^T \vec{f}_s \vec{f}_s^T \mathbf{A},$$

$$\vec{s} = -\sum \mathbf{A}^T \vec{f}_s f_t,$$

and where $\sum$ here means that each element of $\mathbf{R}$ and $\vec{s}$ is obtained by summing (element-by-element) over the entire image region. When $\mathbf{R}$ is singular (or ill-conditioned) there is not sufficient image structure to estimate all six of the flow field parameters; this has sometimes been called the "generalized aperture problem" (but note as before that the problem is not really due to any aperture).

Again, multiscale refinement of the parameter estimates is needed to overcome temporal aliasing when the motion is large. We use the following notation:

$$\vec{\theta} = \mathbf{A}\vec{p} \quad \vec{\theta}^0 = \mathbf{A}\vec{p}^0 \quad \Delta\vec{\theta} = \mathbf{A}\Delta\vec{p},$$

so that

$$\Delta\vec{\theta} = \mathbf{A}(\vec{p} - \vec{p}^0). \tag{35}$$

As before, we let h denote the warped image sequence and we use its derivatives in the gradient constraint:

$$\vec{h}_s \cdot \Delta\vec{\theta} + g_t = 0.$$

Combining with Eq. 35 gives:

$$\vec{h}_s^T \mathbf{A}\vec{p} + (h_t - \vec{h}_s^T \mathbf{A}\vec{p}^0) = 0,$$

so we want to minimize:

$$E(\vec{p}) = \sum [\vec{h}_s^T \mathbf{A}\vec{p} + (h_t - \vec{h}_s^T \mathbf{A}\vec{p}^0)]^2$$

The refined least-squares estimate of the flow field model parameters is therefore:

$$\begin{aligned} \vec{p}^1 &= \mathbf{R}^{-1}(\vec{s} + \mathbf{R}\vec{p}^0) \\ &= \vec{p}^0 + \mathbf{R}^{-1}\vec{s} \end{aligned} \tag{36}$$

where $\mathbf{R}$ and $\vec{s}$ are (as above, except with h instead of f):

$$\mathbf{R} = \sum \mathbf{A}^T \vec{h}_s \vec{h}_s^T \mathbf{A},$$

$$\vec{s} = -\sum \mathbf{A}^T \vec{h}_s h_t,$$

In other words, the refinement of the model parameters has the same form as the refinement of the flow fields estimates above. The residual model parameters $\Delta\vec{p}$ are given by

$$\Delta\vec{p} = \mathbf{R}^{-1}\vec{s}$$

When the model is a good approximation to the image motion, this method is very accurate since one only has to estimate a small number of parameters (6 for the affine flow model) given hundreds or thousands of constraints (one constraint for each pixel throughout a large image region). The problem with the model-based method is that large image regions are often not well modeled by a single parametric motion due to the complexity of the motion or the presence of multiple moving objects. Smaller regions on the other hand may not provide sufficient constraints for estimating the parameters (the generalized aperture problem discussed above). Recent research has extended model-based flow estimation approach by using a piecewise affine model for the flow field, and simultaneously estimating the boundaries between each region and the affine parameters within each region (Wang and Adelson, 1994; Black and Jepson, 1996, Ju *et al.*, 1996).

References

[1] E H Adelson and J R Bergen. Spatiotemporal energy models for the perception of motion. *Journal of the Optical Society of America A*, 2:284–299, 1985.

[2] E H Adelson and J R Bergen. The extraction of spatio-temporal energy in human and machine vision. In *Proceedings of IEEE Workshop on Motion: Representation and Analysis*, pages 151–156, Charleston, S Carolina, 1986.

[3] J K Aggarwal and N Nandhakumar. On the computation of motion from sequences of images — a review. *Proceedings of the IEEE*, 76:917–935, 1988.

[4] P Anandan. A computational framework and an algorithm for the measurement of visual motion. *International Journal of Computer Vision*, 2:283–310, 1989.

[5] J Barron. A survey of approaches for determining optic flow, environmental layout and egomotion. Technical Report RBCV-TR-84-5, Department of Computer Science, University of Toronto, 1984.

[6] J R Bergen, P Anandan, K Hanna, and R Hingorani. Hierarchical model-based motion estimation. In *Proceedings of Second European Conf. on Comp. Vis.*, pages 237–252. Springer-Verlag, 1992.

[7] M J Black and P Anandan. The robust estimation of multiple motions: parametric and piecewise-smooth flow fields. *Computer Vision and Image Understanding*, 63:75–104, 1996.

[8] A R Bruss and B K P Horn. Passive navigation. *Computer Vision, Graphics, and Image Processing*, 21:3–20, 1983.

[9] D J Fleet and A D Jepson. Computation of component image velocity from local phase information. *International Journal of Computer Vision*, 5:77–104, 1990.

[10] D J Fleet and A D Jepson. Stability of phase information. *IEEE Pattern Analysis and Machine Intelligence*, 15:1253–1268, 1993.

[11] D J Fleet, A D Jepson, and M. Jenkin. Phase-based disparity measurement. *Computer Vision, Graphics, and Image Processing*, 53:198–210, 1991.

[12] N M Grzywacz and A L Yuille. A model for the estimate of local image velocity by cells in the visual cortex. *Proceedings of the Royal Society of London A*, 239:129–161, 1990.

[13] D J Heeger. Model for the extraction of image flow. *Journal of the Optical Society of America A*, 4:1455–1471, 1987.

[14] D J Heeger and A D Jepson. Subspace methods for recovering rigid motion I: Algorithm and implementation. *International Journal of Computer Vision*, 7:95–117, 1992.

[15] B K P Horn and B G Schunk. Determining optical flow. *Artificial Intelligence*, 17:185–203, 1981.

[16] S X Ju, M J Black, and A D Jepson. Skin and bones: multi-layer, locally affine, optical flow and regularization with transparency. In *Computer Vision and Pattern Recognition*, pages 307–314, 1996.

[17] H C Longuet-Higgins and K Prazdny. The interpretation of a moving retinal image. *Proceedings of the Royal Society of London B*, 208:385–397, 1980.

[18] B D Lucas and T Kanade. An iterative image registration technique with an application to stereo vision. In *Proceedings of the 7th International Joint Conference on Artificial Intelligence*, pages 674–679, Vancouver, 1981.

[19] H H Nagel. On the estimation of optical flow: relations between different approaches and some new results. *Artificial Intelligence*, 33:299–324, 1987.

[20] E P Simoncelli. *Distributed representation and analysis of visual motion.* PhD thesis, Electrical Engineering Department, MIT, 1993.

[21] A Verri, F Girosi, and V Torre. Differential techniques for optical flow. *Journal of the Optical Society of America A*, 7:912–922, 1990.

[22] J Y A Wang and E H Adelson. Representing moving images with layers. *IEEE Trans. on Image Processing*, 3:625–638, 1994.

[23] A B Watson and A J Ahumada. A look at motion in the frequency domain. In J K Tsotsos, editor, *Motion: Perception and representation*, pages 1–10. Association for Computing Machinery, New York, 1983.

[24] A B Watson and A J Ahumada. Model of human visual-motion sensing. *Journal of the Optical Society of America A*, 2:322–342, 1985.

[25] A M Waxman and S Ullman. Surface structure and three-dimensional motion from image flow kinematics. *International Journal of Robotics Research*, 4:72–94, 1985.