Notes on Motion Estimation

Psych 267/CS 348D/EE 365
Prof. David J. Heeger
November 16, 1998

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. For a good overview and comparison of different flow methods, see (Barron et al, 1994).

1 Geometry: 3D Velocity and 2D Image Velocity

Motion occurs in many applications, and there are many tasks for which motion information might be very useful. Here we concentrate on natural image sequences of 3d scenes in which objects and the camera may be moving. Typical issues for which we want computational solutions include inferring the relative 3d motion between the camera and objects in the scene, inferring the depth and surface structure of the scene, and segmenting the scene using the motion information Towards this end, we are interested in knowing the geometric relationships between 3d motion, surface structure, and 2d image velocity.

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

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

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

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

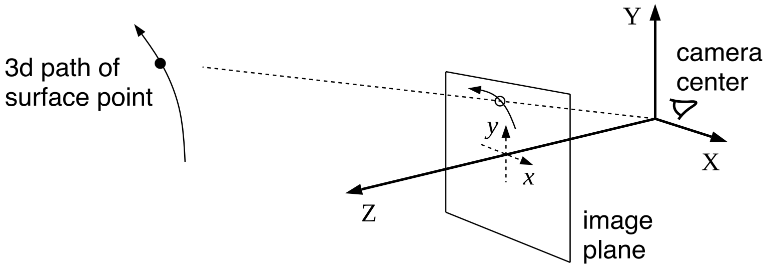

Figure 1: Camera centered coordinate frame and perspective projection. Owing to motion between the camera and the scene, a 3D surface point traverses a path in 3D. Under perspective projection, this path projects onto a 2D path in the image plane, the temporal derivative of which is called 2D velocity. The 2d velocities associated with all visible points defines a dense 2d vector field called the 2d motion field.

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

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

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

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

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 $\mathbf{T} = (T_x, T_y, T_z)^T$, and its instantaneous 3d rotation $\mathbf{\Omega} = (\Omega_x, \Omega_y, \Omega_z)^T$. Here, the direction of $\mathbf{\Omega}$ gives the axis of rotation while $|\mathbf{\Omega}|$ 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

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

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 $d\mathbf{X}/dt$ in

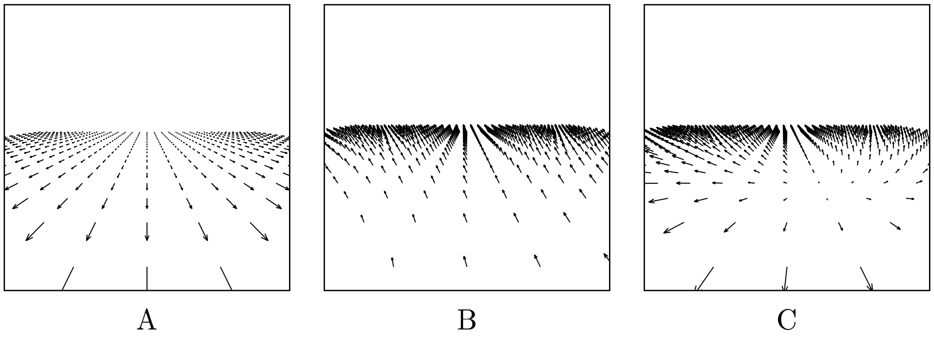

Figure 2: Example flow fields. **A:** Camera translation. **B:** Camera rotation. **C:** Translation plus rotation. Each flow vector in C is the vector sum of the two corresponding vectors in A and B.

Eq. 4 we obtain an expression for the form of the optical flow field for a rigid scene:

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

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 $\mathbf{T}$ (the translation) are multiplied together in Eq. 6, a larger distance (smaller p) or a slower 3D translation (smaller $|\mathbf{T}|$) 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 ($\mathbf{\Omega} = \mathbf{0}$),

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

i.e.,

$$\begin{aligned} u_1(x,y) &= p(x-x_0)T_z \qquad & x_0 &= fT_x/T_z \\ u_2(x,y) &= p(y-y_0)T_z \qquad & y_0 &= fT_y/T_z. \end{aligned} \tag{7}$$

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 u_1 and u_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:

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

We already know (see above) that the rotational component of the flow field (the second term, $\mathbf{B\Omega}$) 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.

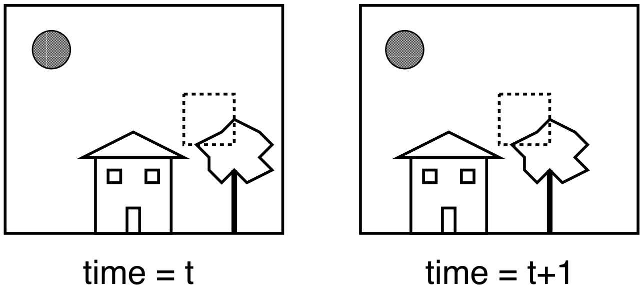

Figure 3: Intensity conservation assumption: the image intensities are shifted (locally translated) from one frame to the next.

2 Gradient-Based 2D 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.,

$$f(x,y,t) = f(x+u_1, y+u_2, t+1), \tag{10}$$

where $f(x,y,t)$ is image intensity as a function of space and time, and $\mathbf{u} = (u_1, u_2)$ is velocity. Of course, this *intensity conservation* assumption is only approximately true in practice. The effective assumption is that the surface radiance remains fixed from one frame to the next in an image sequence. One can fabricate scene constraints for which this holds, but they are somewhat artificial: For example, the scene might be constrainted to contain only Lambertian surfaces (no specularities), with a distant point source (so that changing the distance to the light source has no effect), with no secondary illumination effects (shadows or inter-surface reflection). Althought unrealistic, it is remarkable that this intensity conservation constraint works as well as it does!

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

$$\sum f(x,y,t)\, f(x+u_1, y+u_2, t+1),$$

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

$$\sum [f(x,y,t) - f(x+u_1, y+u_2, t+1)]^2.$$

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

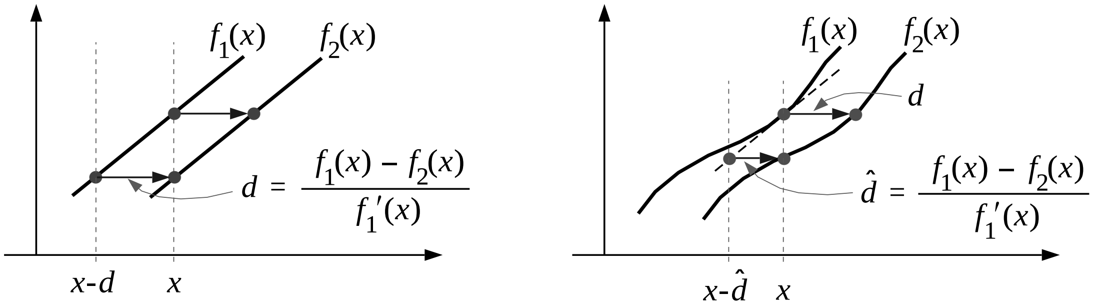
$f_1(x)$
$f_2(x)$
$d = \frac{f_1(x) - f_2(x)}{f_1'(x)}$
x-d
x
$f_1(x)$
$f_2(x)$
d
$\hat{d} = \frac{f_1(x) - f_2(x)}{f_1'(x)}$
x-$\hat{d}$
x

Figure 4: The gradient constraint relates the displacement at one point of the signal to the temporal difference and the spatial derivative (slope) of the signal. For a displacement of a linear signal (left), the difference in signal values at a point divided by the slope gives the displacement. In the case of nonlinear signals (right), the difference divided by the slope gives an approximation to the displacement.

Gradient Constraints. To derive a simple, linear method for estimating the velocity u, let's first digress to consider a 1-d case. Let our signals at two time instants be $f_1(x)$ and $f_2(x)$, and suppose that $f_2(x)$ is a translated version of $f_1(x)$; i.e., $f_2(x) = f_1(x - d)$ for displacement d. This situation is illustrated in Fig. 4. A Taylor series expansion of $f_1(x - d)$ about x is given by

$$f_1(x - d) \;=\; f_1(x) - d\, f_1'(x) + O(d^2 f_1'') \tag{11}$$

where f' denotes the derivative of f with respect to x. Given this expansion, we can now write the difference between the two signals at location x as

$$f_1(x) - f_2(x) \;=\; d\, f_1'(x) + O(d^2 f_1'')$$

By neglecting higher-order terms, we arrive at an approximation to the displacement d, that is,

$$\hat{d} \;=\; \frac{f_1(x) - f_2(x)}{f_1'(x)}\ . \tag{12}$$

The Motion Gradient Constraint. This approach generalizes straightforwardly to two spatial dimensions, to produce the *motion gradient constraint equation.* As above, one assumes that the time-varying image intensity is well approximated by a first-order Taylor series expansion,

$$f(x + u_1, y + u_2, t + 1) \approx f(x, y, t) + u_1 f_x(x, y, t) + u_2 f_y(x, y, t) + f_t(x, y, t),$$

where f_x, f_y, and f_t are the spatial and temporal partial derivatives of image intensity. If we ignore second- and higher-order terms in the Taylor series, and then substitute the resulting approximation into Eq. 10 we obtain

$$u_1 f_x(x, y, t) + u_2 f_y(x, y, t) + f_t(x, y, t) = 0\ . \tag{13}$$

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. 13 is called the motion gradient constraint.

If the time between frames is relatively large, so that the temporal derivative $f_t(x,y,t)$ is unavailable, then a similar constraint can be derived by taking only the first-order spatial terms of the Taylor expansion. In this case we obtain

$$u_1 f_x(x,y,t) + u_2 f_y(x,y,t) + \Delta f(x,y,t) = 0 \ , \tag{14}$$

where $\Delta f(x,y,t) = f(x,y,t+1) - f(x,y,t)$.

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. Image velocity is simply the temporal derivative of such paths.

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

$$f(x(t),y(t),t) = c \ . \tag{15}$$

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 15:

$$\frac{d}{dt} f(x(t),y(t),t) = 0 \tag{16}$$

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:

$$\begin{aligned} \frac{d}{dt} f(x(t),y(t),t) &= \frac{\partial f}{\partial x}\frac{dx}{dt} + \frac{\partial f}{\partial y}\frac{dy}{dt} + \frac{\partial f}{\partial t}\frac{dt}{dt} \\ &= f_x u_1 + f_y u_2 + f_t \end{aligned} \tag{17}$$

In this last step the velocities were substituted for the path derivatives (i.e., $u_1 = dx/dt$ and $u_2 = dy/dt$). Finally, if we combine Eqs. 16 and 17 we obtain the motion gradient constraint.

Equation 16 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. This assumption is often violated to some degree in real image sequences.

Combining Constraints. It is impossible to recover velocity, given just the gradient constraint at a single position, since Eq. 13 offers only one linear constraint to solve for the two unknown components of velocity. Only the component of velocity in the gradient direction is determined. In effect, as shown in Fig 5, the gradient constraint equation constrains the velocity to lie somewhere along a line in velocity space. Further assumptions or measurements are required to constrain velocity to a point in velocity space.

Many 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:

$$\begin{bmatrix} f_x(x_1,y_1,t) & f_y(x_1,y_1,t) \\ f_x(x_2,y_2,t) & f_y(x_2,y_2,t) \end{bmatrix} \begin{bmatrix} u_1 \\ u_2 \end{bmatrix} + \begin{bmatrix} f_t(x_1,y_1,t) \\ f_t(x_2,y_2,t) \end{bmatrix} = \mathbf{0}, \tag{18}$$

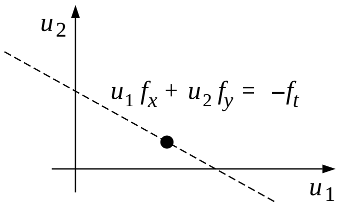

Figure 5: The gradient constraint equation constrains velocity to lie somewhere on a line in 2d velocity space. That is, $u_1 f_x + u_2 f_y = -f_t$ defines a line perpendicular (f_x, f_y). If one had a set of measurements from nearby points, all with different gradient directions but consistent with the same velocity, then we expect the constraint lines to intersect at the true velocity (the black dot).

where the two coordinate pairs (x_i, y_i) correspond to the two spatial positions. Each row of Eq. 18 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:

$$E(u_1, u_2) = \sum_{x,y} g(x,y)\,[u_1 f_x(x,y,t) + u_2 f_y(x,y,t) + f_t(x,y,t)]^2 \ , \tag{19}$$

where $g(x,y)$ is a window function that is zero outside the neighborhood within which we wish to use the constraints. Each squared term in the summation is a constraint on the flow from a different (nearby) position. Usually, we want to weight the terms in the center of the neighborhood higher to give them more influence. Accordingly, it is common to let $g(x,y)$ be a decaying window function such as a Gaussian.

Since there are now more constraints than unknowns, there may not be a solution that satisfies all of the constraints simultaneously. In other words, $E(u_1, u_2)$ will typically be non-zero for all (u_1, u_2). The choice of (u_1, u_2) that minimizes $E(u_1, u_2)$ is the least squares estimate of velocity.

Least Squares Estimate. Since Eq. 19 is a quadratic expression, there is a simple analytical expression for the velocity estimate. The solution is derived by taking derivatives of Eq. 19 with respect to u_1 and u_2, and setting them equal to zero:

$$\begin{aligned}
\frac{\partial E(u_1,u_2)}{\partial u_1} = \sum_{xy} g(x,y)\,[u_1 (f_x)^2 + u_2 (f_x f_y) + (f_x f_t)] &= 0 \\
\frac{\partial E(u_1,u_2)}{\partial u_2} = \sum_{xy} g(x,y)\,[u_2 (f_y)^2 + u_1 (f_x f_y) + (f_y f_t)] &= 0
\end{aligned}$$

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

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

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

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

$$\mathbf{b} = \begin{pmatrix} \sum g\, f_x f_t \\ \sum g\, f_y f_t \end{pmatrix} .$$

The least-squares solution is then given by

$$\hat{\mathbf{u}} = -\mathbf{M}^{-1}\mathbf{b}, \tag{20}$$

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 $\mathbf{b}$ at each location. In particular, we should rewrite our matrix equation as

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

As above in Eq. 20, the elements in $\mathbf{M}(x,y)$ and $\mathbf{b}(x,y)$ 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 $\mathbf{b}(x,y)$ 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 $\mathbf{b}(x,y)$ are given by:

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

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

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 $\mathbf{b}(x,y)$ 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.

Gaussian PreSmoothing. Another practicality worth mentioning here is that some preprocessing before applying the gradient constraint is generally necessary to produce reliable estimates of optical flow. The reason is primarily due to the use of a first-order Taylor series expansion in deriving the motion constraint equation. By smoothing the signal, it is hoped that we are reducing the relative amplitudes of higher-order terms in the image. Therefore, it is common to smooth the image sequence with a lowpass filter before estimating derivatives, as in Eq. 3. Often, this presmoothing can be incorporated into the derivative filters used to computer the spatial and temporal derivatives of the image intensity. This also helps avoid problems caused by small amounts of aliasing, something we'll say more about later.

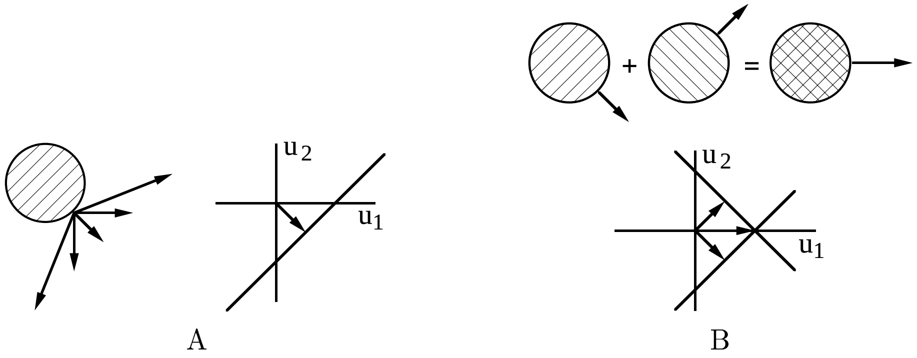

Figure 6: Aperture problem. **A:** Single moving grating viewed through a circular aperture. The diagonal line indicates the locus of velocities compatible with the motion of the grating. **B:** Plaid composed of two moving gratings. The lines give the possible motion of each grating alone. Their intersection is the only shared motion.

Aperture Problem. When the matrix $\mathbf{M}$ in Eq. 20 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, and to ensure the reliability of the velocity estimates it is common to check the condition number of $\mathbf{M}$ (or better yet, the magnitude of its smallest singular value).

The latter case is illustrated in Fig. 6A. 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 6B shows how the motion is disambiguated when there is more spatial structure. The plaid pattern illustrated in Fig. 6B 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. 18 is related to the intersection of constraints rule depicted in Fig. 6. The gradient constraint, Eq. 13, is linear in both u_1 and u_2. Given measurements of the derivatives, (f_x, f_y, f_t), there is a line of possible solutions for (u_1, u_2), analogous to the constraint line illustrated in Fig. 6A. For each different position, there will generally be a different constraint line. Equation 18 gives the intersection of these constraint lines, analogous to Fig. 6B.

Regularization Solution. Another way to add further constraints to the gradient constraint equation is to invoke a smoothness assumption. In other words, one can constrain (regularize) the estimation problem by assuming that resultant flow field should be the smoothest flow field that is also consistent with the gradient constraints. The central assumption here, like that made with the area-based regression approach above, is that the flow in most image regions is expected to be smooth because smooth surfaces generally give rise to smooth optical flow fields.

For example, let's assume that we want the optical flow to minimize any violations of the

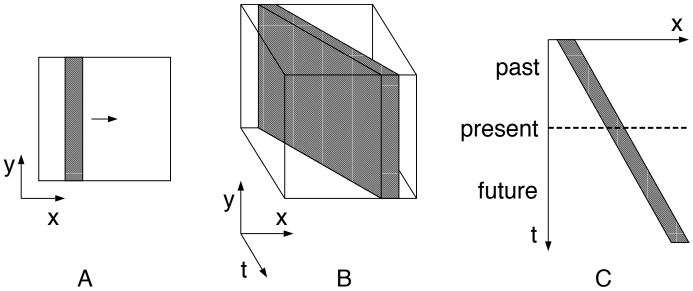

Figure 7: Orientation in space-time (based on an illustration by Adelson and Bergen, 1985). **A:** A vertical bar translating to the right. **B:** The space-time cube of image intensities corresponding to motion of the vertical bar. **C:** An x-t slice through the space-time cube. Orientation in the 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.

gradient-constraint equation, and at the same time, to minimize the magnitude of velocity changes between neighboring pixels. For example, we may wish to minimize:

$$E \;=\; \sum_{x,y} (f_x u_1 + f_y u_2 + f_t)^2 \;+\; \lambda \sum_{x,y} \left(\left(\frac{\partial u_1}{\partial x}\right)^2 + \left(\frac{\partial u_1}{\partial y}\right)^2 + \left(\frac{\partial u_2}{\partial x}\right)^2 + \left(\frac{\partial u_2}{\partial y}\right)^2 \right)$$

In practice, the derivatives are approximated by numerical derivative operators, and the resulting minimization problem has as its solution a large system of linear equations. Because of the large dimension of the linear system, it is commonly solved iteratively using *Jacobi*, *Gauss-Seidel*, or *conjugate-gradient* iteration.

One of the main disadvantages of this approach is the extensive amount of smoothing often imposed, even when the actual optical flow field is not smooth. Some researchers have advocated the use of *line processes* that allow for violations of smoothness at certain locations where the flow field constraints indicate discontinuous flow (due to occlusion for example). This issue is involved and beyond the scope of the handout for now.

3 Space-Time Filters and the Frequency Domain

One can also view image motion and these approaches to estimating optical flow in the frequency domain. This will help to understand aliasing, and it also serves as the foundation for filter-based methods for optical flow estimation. In addition, 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 7 shows a simple example. Figure 7A 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 7B. Figure 7C 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.

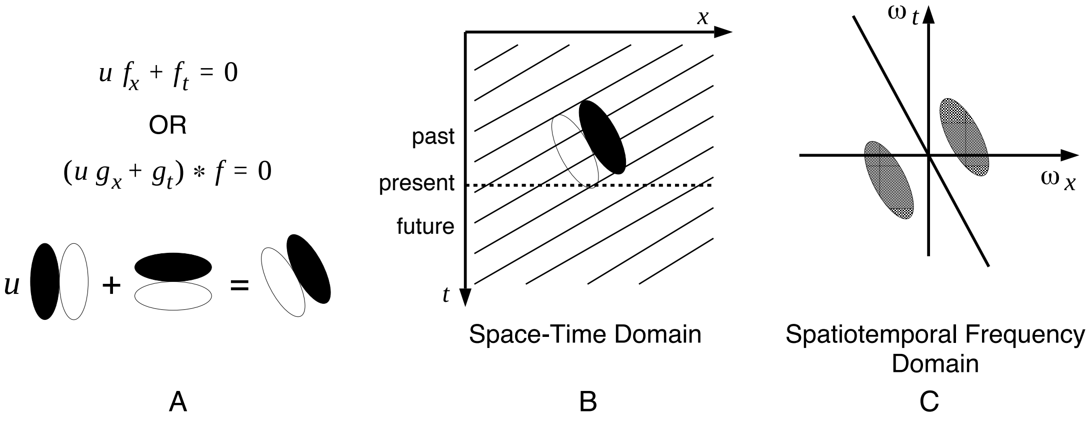

Figure 8: Space-time filters and the gradient constraint. **A:** The gradient constraint states that there is some space-time orientation for which the directional derivative is zero. **B:** Pattern moving from right to left and the space-time derivative filter with the orthogonal space-time orientation so that convolving the two gives a zero response. **C:** Spatiotemporal Fourier transform of **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.

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:

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

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:

$$(u_1 g_x + u_2 g_y + g_t) * f = 0, \tag{21}$$

where g_x, g_y, and g_t are the space-time derivative filters. For a fixed $\mathbf{u}$, Eq. 21 is the convolution of a space-time filter, namely $(u_1 g_x + u_2 g_y + g_t)$, 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. 8.

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; Fleet, 1992), 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

$$f(x,t) = \sin(\omega_x (x - u_1 t)) \tag{22}$$

where ω_x is the spatial frequency of the grating and u_1 is the velocity. It can also be expressed in terms of spatial frequency ω_x and temporal frequency ω_t as

$$f(x,t) = \sin(\omega_x x + \omega_t t) \tag{23}$$

The relation between frequency and velocity can be derived algebraically by setting $\omega_x(x - u_1 t) = \omega_x x + \omega_t t$. From this one can show that

$$u_1 = -\omega_t / \omega_x \tag{24}$$

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 u_1. Each frequency component has a spatial frequency ω_x and a temporal frequency ω_t but all of the Fourier components share the same velocity so each component satisfies

$$u_1\,\omega_x + \omega_t = 0.$$

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

$$u_1\,\omega_x + u_2\,\omega_y + \omega_t = 0.$$

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 $(u_1, u_2, 1)^T$.

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 $\mathbf{u}$. The image intensity over time may be written as

$$f(x,y,t) = f_0(x - u_1 t,\, y - u_2 t).$$

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:

$$F(\omega_x, \omega_y, \omega_t) = \int\int\int_{-\infty}^{\infty} f(x,y,t) e^{-j2\pi(x\omega_x + y\omega_y + t\omega_t)} dx dy dt. \tag{25}$$

To simplify this expression, we substitute $f_0(x - u_1 t, y - u_2 t)$ for $f(x,y,t)$ and arrange the order of integration so that time is integrated last:

$$F(\omega_x, \omega_y, \omega_t) = \int \left[\int\int_{-\infty}^{\infty} f_0(x - u_1 t, y - u_2 t) e^{-j2\pi(x\omega_x + y\omega_y)} dx dy\right] e^{-j2\pi t\omega_t} dt. \tag{26}$$

The term inside the square brackets is the 2d Fourier transform of $f_0(x - u_1 t,\, y - u_2 t)$ which can be simplified using the Fourier shift property to yield

$$\begin{aligned} F(\omega_x, \omega_y, \omega_t) &= \int_{-\infty}^{\infty} F_0(\omega_x, \omega_y) e^{-j2\pi(u_1 t\omega_x + u_2 t\omega_y)} e^{-j2\pi t\omega_t} dt \\ &= F_0(\omega_x, \omega_y) \int_{-\infty}^{\infty} e^{-j2\pi t(u_1\omega_x + u_2\omega_y)} e^{-j2\pi t\omega_t} dt \end{aligned} \tag{27}$$

where $F_0(\omega_x, \omega_y)$ 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

$$F(\omega_x, \omega_y, \omega_t) = F_0(\omega_x, \omega_y)\, \delta(u_1\omega_x + u_2\omega_y + \omega_t)$$

This equation shows that the Fourier transform is only nonzero on a plane, because the delta function is only nonzero when $u_1\omega_x + u_2\omega_y + \omega_t = 0$.

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 8 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:

$$\sum_{x,y,t} |f(x, y, t)|^2 = \sum_{\omega_x, \omega_y, \omega_t} |F(\omega_x, \omega_y, \omega_t)|^2. \tag{28}$$

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 $j\omega$. For example,

$$\mathcal{F}[f_x(x, y, t)] = j\omega_x \mathcal{F}[f(x, y, t)] = j\omega_x F(\omega_x, \omega_y, \omega_t) \tag{29}$$

Combining Eqs. 19, 28, and 29, gives:

$$\begin{aligned} E(u_1, u_2) &= \sum_{x,y,t} |u_1 f_x(x, y, t) + u_2 f_y(x, y, t) + f_t(x, y, t)|^2 \qquad (30) \\ &= \sum_{\omega_x, \omega_y, \omega_t} |u_1\omega_x F(\omega_x, \omega_y, \omega_t) + u_2\omega_y F(\omega_x, \omega_y, \omega_t) + \omega_t F(\omega_x, \omega_y, \omega_t)|^2 \\ &= \sum_{\omega_x, \omega_y, \omega_t} [(u_1, u_2, 1) \cdot \omega]^2\, |F(\omega)|^2. \end{aligned}$$

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 $\mathbf{u}$ to satisfy: $u_1\omega_x + u_2\omega_y + \omega_t = 0$. If any of the approximations are violated then there will not in general be any $\mathbf{u}$ that will satisfy the constraint perfectly. Minimizing Eq. 30 is a weighted, least-squares regression.

To be concrete, let $\omega_n = (\omega_{xn}, \omega_{yn}, \omega_{tn})$ for $1 < n < N$ be the spatiotemporals frequencies for which we have samples of $|F(\omega)|$. Rewriting Eq. 30 in matrix notation gives:

$$E(u_1, u_2) = \|\mathbf{Au}\|^2, \tag{31}$$

where $\mathbf{u} = (u_1, u_2, 1)^T$ and

$$\mathbf{A} = \begin{pmatrix} F(\omega_1)\omega_{x1} & F(\omega_1)\omega_{y1} & F(\omega_1)\omega_{t1} \\ F(\omega_2)\omega_{x2} & F(\omega_2)\omega_{y2} & F(\omega_2)\omega_{t2} \\ \vdots & \vdots & \vdots \\ F(\omega_N)\omega_{xN} & F(\omega_N)\omega_{yN} & F(\omega_N)\omega_{tN} \end{pmatrix}$$

Because of the weights, $|F(\omega)|$, the regression depends most on those frequency components where the power is concentrated. Equation 31 is minimized by a vector $\mathbf{v}$ in the direction of the eigenvector corresponding to the smallest eigenvalue of $\mathbf{A}^T\mathbf{A}$. The image velocity estimate $\hat{\mathbf{u}}$ is obtained by normalizing $\mathbf{v}$ so that its third element is 1.

For real image sequences, the least-squares estimate of image velocity (Eq. 20) is definitely preferred over the frequency domain regression estimate (Eq. 31) 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 Multi-Scale, Iterative 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. 9. 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

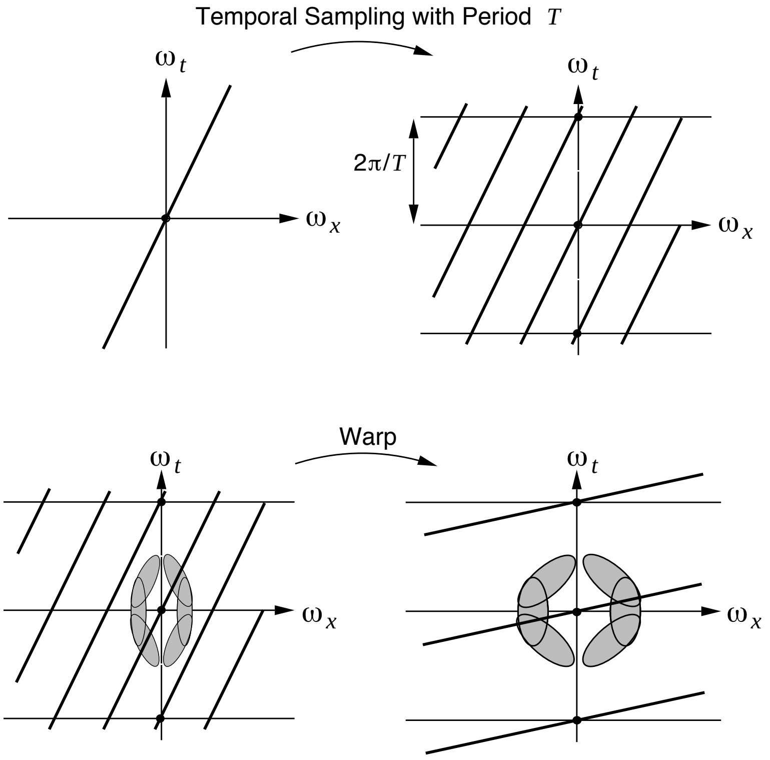

Figure 9: **Top-left:** The power spectrum of a translating signal occupies a line in the spatiotemporal frequency domain. **Top-right:** Sampling in time (to get the discrete frames of an image sequence) gives rise to replicas of the spectra, that may result in temporal aliasing. These replicas can 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 usually affects only 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.

Towards this, following Bergen *et al.* (1992) for example, 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. 9C). Typically, this is all done with an image pyramid data structure.

Coarse-to-fine methods are widely used to measure motion in video image sequences because of the aliasing that regularly occurs. Despite this, it does have its drawbacks, stemming from the fact that the fine-scale estimates can only be as reliable as their coarse-scale precursors. For example, if there is little or no signal energy at coarse-scales, then coarse-scale estimates cannot be trusted and we have no initial guess to offer finer-scales. A related problem occurs if the coarse-scale estimate is very poor, perhaps due to the broad spatial extent of the estimation, then the warping will not significantly decrease the effective velocity as desired, in which case the finer-scale estimation will produce an equally poor result.

Iterative Refinement of Velocity Estimates. Such coarse-to-fine estimation strategies are an 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. In particular, let's return to the derivation of the gradient constraint in the 1d case, shown in Fig. 4, where we wanted to find the displacement d between $f_2(x)$ and $f_1(x)$. We linearized the signals to derive an estimate $\hat{d} = (f_1(x) - f_2(x))/f_1'(x)$. As illustrated in Fig. 4, when f_1 is a linear signal, then $d = \hat{d}$. Otherwise, to leading order, the accuracy of the estimate is bounded by the magnitude of the dispacement and the second derivative of f_1:

$$|\hat{d} - d| \;\leq\; \frac{d^2\,|f_1''(x)|}{2\,|f_1'(x)|} \;. \tag{32}$$

For a sufficiently small displacement, and bounded f_1''/f_1', we expect reasonably accurate estimates.

In any case, so long as the estimation error is smaller than $|d|$, then with Newton's method we can iterate the estimation to acheive our goal of having the estimate $\hat{d}$ satisfy $f_1(x - \hat{d}) - f_2(x) = 0$.

Toward this end, assume we're given an estimate $\hat{d}_0$, and let the error be denoted $u = d - \hat{d}_0$. Then, because $f_2(x)$ is a displaced version of $f_1(x)$ we know that $f_2(x) = f_1(x - \hat{d}_0 - u)$. Accordingly, using gradient-based estimation, Eq. (12), we form a new estimate of d given by $\hat{d}_1 = \hat{d}_0 + \hat{u}$, where

$$\hat{u} \;=\; \frac{f_1(x-\hat{d}) - f_2(x)}{f_1'(x-\hat{d})}\;. \tag{33}$$

Here, $f_1(x - \hat{d})$ is simply a warped version of $f_1(x)$. This process can be iterated to find a succession of estimates $\hat{d}_k$ that converge to d if the initial estimate is in the correct direction.

In the general 2D problem, we are given an estimate of the optical flow field $\mathbf{u}^0 = (u_1^0, u_2^0)^T$ (e.g., it might have been estimated from the coarse scale), and we create a warped image sequence $h(x, y, t)$:

$$h(x, y, \Delta t) = \mathcal{W}[f(x, y, t + \Delta t), \mathbf{u}^0(x, y)] = f(x - u_1^0 \Delta t, y - u_2^0 \Delta t, t + \Delta t),$$

where Δt is the time between two consecutive frames, and $\mathcal{W}[f, \mathbf{u}^0]$ denotes the warping operation that warps image f by the vector field $\mathbf{u}^0$. 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*:

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

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

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

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.

5 Higher-Order Gradient Constraints and Motion Models

There are two relatively straightforward ways in which the gradient-based approach introduced above can be extended. The first is to introduce other forms of image preprocessing before applying the gradient constraint. If such processing is a derivative filter, then this amounts to deriving high-order differential constraints on the optical flow.

The second way we can generalize the method above is by introducing rich parametric models of the optical flow within the neighborhood over which contraints are combined. In the method introduced above, we essentially compute a single estimate of the optical flow in the neighborhood. This amounts to an assumption that the flow is well modelled with a constant flow model within the region. It is a relatively straightforward task to generalize this approach to include higher-order models such as an affine model that allows us to estimate rotation and scaling along with translation in the neighborhood.

5.1 Using Higher Order Derivatives.

In the gradient-based approach discussed above, we tracked points of consistant intensity (or smoothed version of image intensity) using only first-order derivatives. One can, however use various forms of prefiltering before applying the gradient constraint. For example, 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. 13). This yields two constraints on the optical flow at each location:

$$\begin{aligned} u_1 f_{xx} + u_2 f_{xy} + f_{xt} &= 0 \\ u_1 f_{xy} + u_2 f_{yy} + f_{yt} &= 0, \end{aligned} \tag{34}$$

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

$$\begin{aligned} u_1 f_{xxx} + u_2 f_{xxy} + f_{xxt} &= 0 \\ u_1 f_{xxy} + u_2 f_{xyy} + f_{xyt} &= 0 \\ u_1 f_{xyy} + u_2 f_{yyy} + f_{yyt} &= 0, \end{aligned} \tag{35}$$

where f_{xxx} should be interpreted as $f * g_{xxx}$, and likewise for the other derivatives. Equation 35, 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:

$$\begin{aligned} E(u_1, u_2) &= \sum_{x,y} [u_1 f_{xxx} + u_2 f_{xxy} + f_{xxt}]^2 \\ &+ \sum_{x,y} [u_1 f_{xxy} + u_2 f_{xyy} + f_{xyt}]^2 \\ &+ \sum_{x,y} [u_1 f_{xyy} + u_2 f_{yyy} + f_{yyt}]^2 \end{aligned}$$

The solutions can be written in the same form as Eq. 20. 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. 13 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. 35, 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.

It is worth noting that one can also view these higher-order constraints in terms of the conservation of some image property. For example, following Eqs. (15) and (17), second-order constraints arise from an assumption that the spatial gradient $\nabla f(x, y, t)$ is conserved. In effect, this assumption is inconsistent with dilating and rotating deformations. By comparison, the assumption of intensity conservation in Eq. (16) implies only that image must smoothly deform from one frame to the next, but the form of the flow field is otherwise unrestricted.

5.2 Higher-Order Motion Models

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.

Affine Models. Many researchers have found that affine models generally produce much better velocity estimates than constant models (e.g., Fleet and Jepson, 1990; Bergen et al., 1992). An affine motion model is expressed component-wise as

$$\begin{aligned} u_1(x,y) &= p_1 + p_2 x + p_3 y \\ u_2(x,y) &= p_4 + p_5 x + p_6 y \end{aligned}$$

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

$$\mathbf{u}(x,y) = \mathbf{A}(x,y)\,\mathbf{p} \tag{36}$$

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:

$$\mathbf{f}_s \cdot \mathbf{u} + f_t = 0, \tag{37}$$

where $\mathbf{f}_s = (f_x, f_y)^T$ are the spatial derivatives. Combining Eqs. 36 and 37 gives:

$$\mathbf{f}_s{}^T \mathbf{A}\mathbf{p} + f_t = 0,$$

so we want to minimize:

$$E(\mathbf{p}) = \sum_{x,y} (\mathbf{f}_s{}^T \mathbf{A}\mathbf{p} + f_t)^2, \tag{38}$$

where the summation may be taken over the entire image (or over some reasonably large image region). Equation 38 is quadratic in $\mathbf{p}$ so the estimate $\hat{\mathbf{p}}$ is derived in the usual (linear least-squares) way by taking derivatives of E with respect to $\mathbf{p}$. This yields the solution

$$\hat{\mathbf{p}} = \mathbf{R}^{-1}\,\mathbf{s}, \tag{39}$$

where

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

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

and where $\sum$ here means that each element of $\mathbf{R}$ and $\mathbf{s}$ 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:

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

so that

$$\Delta\mathbf{u} = \mathbf{A}(\mathbf{p} - \mathbf{p}^0). \tag{40}$$

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

$$\mathbf{h}_s \cdot \Delta\mathbf{u} + g_t = 0.$$

Combining with Eq. 40 gives:

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

so we want to minimize:

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

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

$$\begin{aligned} \mathbf{p}^1 &= \mathbf{R}^{-1}(\mathbf{s} + \mathbf{R}\mathbf{p}^0) \\ &= \mathbf{p}^0 + \mathbf{R}^{-1}\mathbf{s} \end{aligned} \tag{41}$$

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

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

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

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 $\Delta\mathbf{p}$ are given by

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

Plane Perspective Models.??? Another model that is common is the 8-parameter plane perspective model that models the deformation of the image of a plane that occurs with any rigid transformation of the camera under perspective projection.

6 Phase-Based Methods

The gradient constraint is useful for tracking any differentiable image property. We can apply it other properties other than 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. For example, local automatic gain control to compensate for changes in illumination can make a significant difference.

One successful example of nonlinear preprocessing, proposed by Fleet and Jepson (1990, 1993), is to use the phase output of quadrature-pair linear filters. Phase-based methods are a combination of frequency-based and gradient-based approaches; they use band-pass prefilters along with gradient constraints and a local affine model to estimate optical flow.

Phase Correlation. To explain phase-based approaches it is useful to begin with a method called phase-correlation (Kuglin and Hines, 1975). The method is often derived by assuming pure translation between two signals (here in the 1d case for simplicity):

$$f_2(x) = f_1(x - d) \ . \tag{42}$$

The Fourier shift theorem tells us that their Fourier transforms satisfy

$$F_2(\omega) = F_1(\omega) e^{-i\,d\omega} \ . \tag{43}$$

Their amplitude spectra are identical, $A_1(\omega) = A_2(\omega)$, and their phase spectra differ by $d\omega$, i.e., $\psi_2(\omega) = \psi_1(\omega) - d\omega$.

Taking the product of $F_1(\omega)$ and the complex conjugate of $F_2(\omega)$, and then dividing by the product of their amplitude spectra, we obtain

$$\frac{F_1(\omega)\, F_2^*(\omega)}{A_1(\omega) A_2(\omega)} \;=\; \frac{A_1(\omega) A_2(\omega) e^{i(\psi_1(\omega) - \psi_2(\omega))}}{A_1(\omega) A_2(\omega)} \;=\; e^{i\omega d} \tag{44}$$

Here, $e^{i\omega d}$ is a sinusoidal function in the frequency domain, and therefore its inverse Fourier transform is an impulse, located at the disparity d, that is, $\delta(x + d)$. Thus, in short, phase-correlation methods measure disparity by finding peaks in

$$\mathcal{F}^{-1}\left[\frac{F_1(\omega)\, F_2^*(\omega)}{A_1(\omega) A_2(\omega)}\right] \tag{45}$$

Phase correlation, by using a global Fourier basis, measures a single displacement for the entire image. But for many applications of interest, we really want to measure image velocity locally. So, a natural alternative to the Fourier basis is to use a wavelet basis. This gives us a local, self-similar representation of the image from which we can still use phase information (Fleet, 1994).

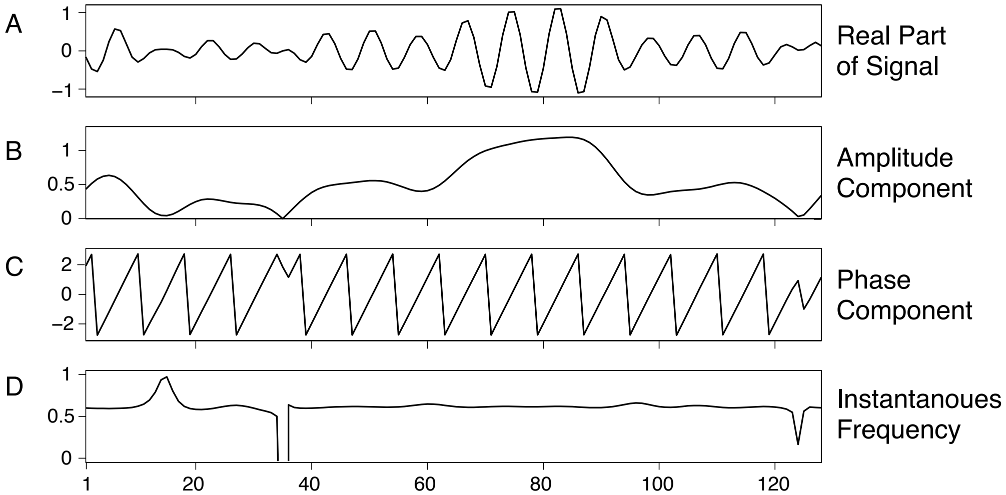

Figure 10: something here.

Local Phase Differences. To develop local phase-based methods, we first need to introduce some notation and terminology associated with band-pass signals. First, with 1d signals and filters, let's assume that we have a band-pass filter $h(x)$ and it's response $r(x)$, perhaps from a wavelet transform. Let's also assume that $r(x)$ is over-sampled to avoid aliasing in individual subbands.

Because $r(x)$ is band-pass, we expect it to modulate at frequencies similar to those in the passband. A good model is usually $r(x) = a(x)\cos(\phi(x))$, where $a(x)$ and $\phi(x)$ are referred to as the amplitude and phase components of the response. Another useful quantity is the instantaneous frequency of $r(x)$ at a point x, defined as $k(x) = d\,\phi(x)/dx$. To see that this definition makes sense intuitively, remember that a sinusoidal signal with constant amplitude is given by $\sin(\omega x)$, in which case the phase component is $\phi(x) = \omega\, x$, and the instantaneous frequency is $k(x) = \omega$. Here the frequency is constant, but with more general classes of band-pass signals, the instantaneous frequency is a function of spatial position.

Figure 10 shows a band-pass signal, along with its amplitude and phase signals, and its instantaneous frequency. Figure 10A shows the real part of the signal, as a function of spatial position, while Figures 10B–C show the amplitude and phase components of the signal. Unlike Fourier spectra, these two signals are functions of spatial position and not frequency. The amplitude represents the magnitude of the signal modulation. The phase component represents the local structure of the signal (e.g., whether the local structure is a peak, or a zero-crossing). Figure 10D shows the instantaneous frequency.

Normally, as can be seen in Figure 10, amplitude is slowly varying, the phase is close to linear, and the instantaneous frequency is close to the center of the pass-band of the signal's Fourier spectrum. This characterization suggests a crude but effective model for a band-pass signal about a point x_0, i.e.,

$$r(x) \;\approx\; a(x_0)\,\cos(\phi(x_0) \,+\, (x - x_0)\phi'(x_0))\;. \tag{46}$$

This approximation involves a constant approximation to the amplitude signal and a first-order approximation to the phase component.

With respect to estimating displacements, we can now point to some of the interesting properties of phase. First, if the signal translates from one frame to the next, then, as long as the filter is shift-invariant, both the amplitude and phase components will also translate. Second, phase almost never has a zero gradient, and it is linear over relatively large spatial extents compared to the signal itself. These properties are useful since gradient-based techniques require division by the gradient, and the accuracy of gradient-based measurements is a function of the linearity of the signal. Together, all this is promising for phase-gradient methods.

Therefore, given two phase signals, $\phi_1(x)$ and $\phi_2(x)$, we can estimate the displacement required to match phase values at a position x using gradient constraints. Fleet et al. (1991) proposed a constraint of the form

$$\hat{d} \;=\; \frac{\lfloor \phi_1(x) - \phi_2(x) \rfloor}{0.5(\phi_1'(x) + \phi_2'(x))} \; . \tag{47}$$

Here, the phase difference is only unique within a range of 2π, and therefore $\lfloor\cdot\rfloor$ denotes a modulo 2π operator. Also, rather than divide by the gradient of one signal, we divide by the average of the two gradients. As with intensity gradient-based methods, this estimator can be iterated using a form of Newton's method to obtain a final estimate. However, in many cases the estimator is accurate enough that only one or two iterations are necessary.

2D Phase-Based Flow and Practical Issues. This method generalizes straightforwardly to 2d, yielding one phase-based gradient constraint at every pixel for every filter. A natural class of filters would be a steerable pyramid, the filters of which are orientation- and scale-tuned.

With a steerable pyramid, the easiest way to extract phase and instantaneous frequency is to use quadrature pair filters. These a complex-valued filters the real and imaginary parts of which are Hilbert transforms of one another. In the Fourier domain, they occupy space in only one half-plane. And because the filter are complex-valued (a pair of real-valued filters), the output $r(x, y, t)$ is also complex-valued. Accordingly, the amplitude components of the output is just the square-root of the sum of the squares of the real and imaginary outputs, i.e. $|r(x)|$. The phase component is the phase angle of $r(x)$, often written as $\arg(r(x))$, and computed using `atan2`.

In order to compute the phase difference, it is convenient to use

$$\lfloor \phi_1(x) - \phi_2(x) \rfloor \;=\; \arg(r_1(x)\, r_2^*(x)) \; . \tag{48}$$

And to compute the phase gradient it is convenient to use the identity

$$\frac{d\phi(x)}{dx} \;=\; \frac{f r'(x) r^*(x)}{|r(x)|^2} \; . \tag{49}$$

This is convenient since, because $r(x)$ is band-pass we know how to interpolate and differentiate it. Conversely, because phase is only unique in the interval $[-\pi,\, \pi)$, there are phase wrap-arounds from π to $-\pi$ which makes explicit differentiation of phase awkward.

Finally, because the responses $r(x, y, t)$ have two spatial dimensions, we take partical derivatives of phase with respect to both x and y, yielding gradient constraint equations of the form

$$u_1 \phi_x(x, y, t) + u_2 \phi_y(x, y, t) + \Delta\phi(x, y, t) = 0 \; , \tag{50}$$

where $\Delta\phi(x, y, t) = \phi(x, y, t + 1) - \phi(x, y, t)$. Since we have many filters in the steerable pyramid, we get one such constraint from each filter at each pixels. We may combine such constraints over all filters within a local neighbourhood and then compute the best least-squares fit to a motion model (e.g. an affine model). And of course this can be iterated to acheive convergence as above.

Also, if it is known that aliasing does not exist in the spatiotemporal signal, then we can also differentiate the response in time, and hence compute the temporal phase derivative. This yields constraints of the form

$$u_1\phi_x(x,y,t) + u_2\phi_y(x,y,t) + \phi_t(x,y,t) = 0 \ , \tag{51}$$

And we may use phase constraints at all scales simultaneously to estimate a local model of the optical flow.

Multiscale Phase Differences Of course, as is often the case with conventional image sequences, like those taken with a video camera, there may be aliasing so that one can not differentiate the signal with respect to time. With respect to phase-based techniques, any signal that is displaced more than half a wavelength is aliased, and therefore phase-based estimates of the displacement will be incorrect. Thus, estimates from responses from filters tuned to higher frequencies will have problems at smaller displacements than responses from filters tuned to lower frequency.

For example, Figure 11 shows the consequences of aliasing. The bandwidth of the 4 filters was approximately 2 octaves, and the passbands were centered at frequencies with wavelengths $\lambda = 4, 8, 16$, and 32 pixels. Figure 11A shows the distributions of instantaneous frequency (weighted by amplitude) that one obtains from these filters with the Einstein image. For relatively large amplitudes, the instantaneous frequencies are concentrated in the respective passbands.

So instantaneous frequencies in the response to the filter with $\lambda = 4$ are generally close to $2\pi/4$. According, we expect to see aliasing appear when the displacements approach 2 pixels (half a wavelength). Indeed, Figure 11B shows that the distribution of displacement estimates obtained from the four filters is very accurate when the displacement is very small. By comparison, when the displacement is -2.5 pixels, as in Figure 11C, one can see that the distribution of estimates from the high frequency filter broadens. As expected, given wavelengths near 4 and a displacement of -2.5, the aliased distribution of estimates occurs near a disparity of 1.5 pixels. When the displacement is 4, in Figure 11D, the high frequency filter produces a broad distribution of aliased estimates near 0. The filter tuned to $\lambda = 8$ also begins to show signs of aliasing. Finally, in Figure 11E, with a displacement of -7.5, aliasing causes erroneous estimates is all but the lowest frequency filter.

Clearly, some form of control structure is required so that errors due to aliasing can be detected and ignored. When talking about gradient based methods we introduced a coarse-to-fine control strategy that could be used here as well. The phase correlation methods suggests another approach. Another approach, somewhat similar to phase correlation is to consider evidence through scale at a wide variety of shifts, and choose that displacement at which the evidence at all channels is consistent. This approach is beyond the scope of this handout.

Phase Stability and Singularities There are several reasons for the success of phase-based techniques for measuring displacements and optical flow. One is that phase signals from band-pass filters are pseudo-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 (Fleet and Jepson, 1993).

Another interesting property of phase is it stability with respect to deformations other than translation between views of a 3d scene. Fleet and Jepson (1991,1993) showed that phase is stable

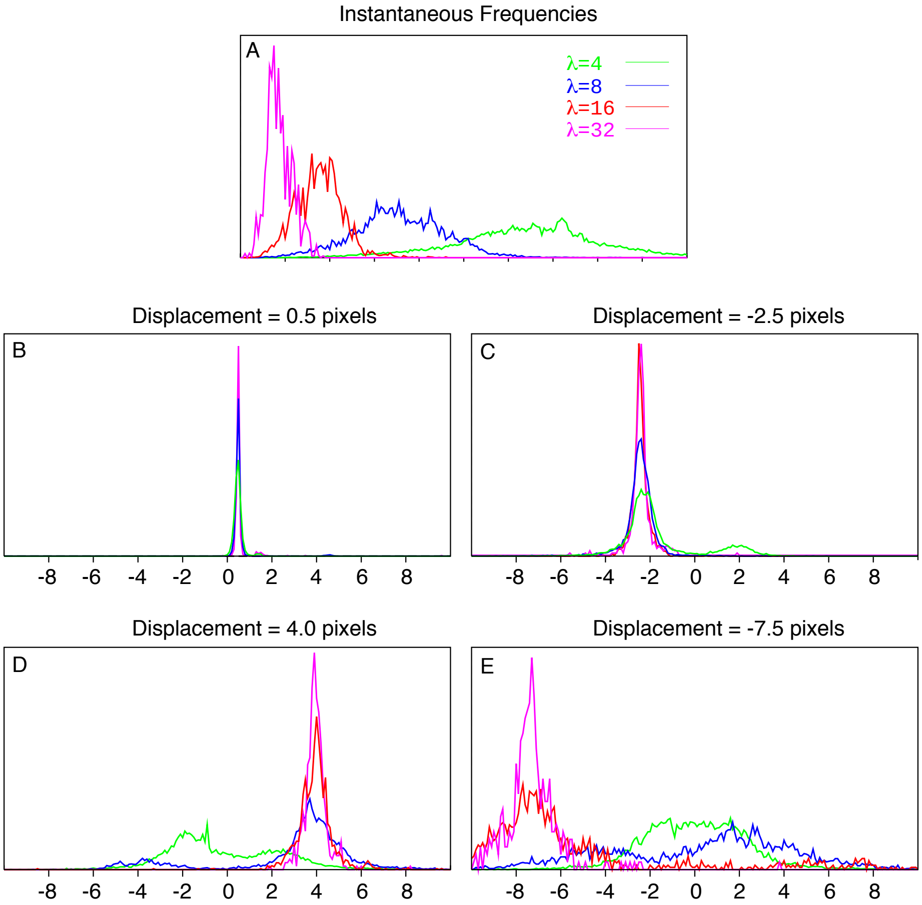

Figure 11: something here.

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. Again, it turns out that the expected stability of phase through scale depends on the bandwidth of the filters. The broader the bandwidth the more stable phase becomes.

For example, Figure 12B,C show the amplitude and phase components of a scale-space expansion $S(x, \lambda)$ of the Gaussian noise signal in Figure 12A. This is the response of a complex-valued Gabor filter with a half-amplitude bandwidth of 0.8 octaves, as a function of spatial position x and wavelength λ, where $16 \leq \lambda \leq 64$ pixels. Level contours of amplitude and phase are shown below in Figure 12D,E. In the context of the scale-space expansion, an image property is said to be *stable* for image matching where its level contours are vertical. In this figure, notice that the amplitude contours are not generally vertical. By contrast, the phase contours are vertical, except for several isolated regions.

These isolated regions of instability are called singularities neighborhoods. In such regions, it can be shown that phase is remarkably unstable, even to small perturbations of scale or of spatial position (Fleet et al, 1991; Fleet and Jepson, 1993). Accordingly, phase constraints on motion estimation should not be trusted in singularity neighborhoods. To detect pixels in singularity

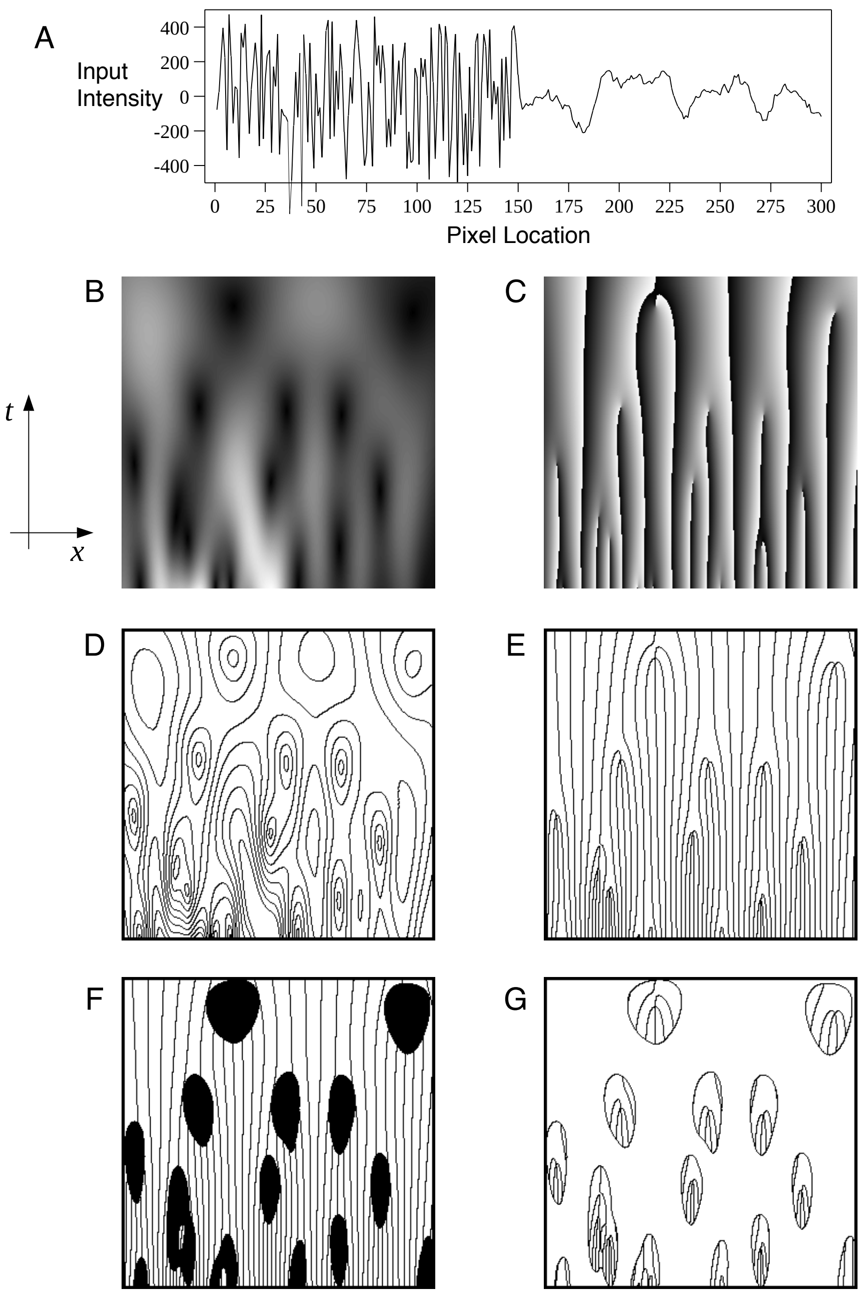

Figure 12: something here.

neighborhoods, Fleet and Jepson (1993) derive the following reliability condition:

$$\sigma(\lambda) \left| \frac{\partial}{\partial x} \log S(x,\, \lambda) \;-\; i\, k(\lambda) \right| \quad \leq \quad \tau \;, \tag{52}$$

where $k(\lambda) = 2\pi/\lambda$, and $\sigma(\lambda)$ is the standard deviation of the Gaussian envelope of the Gabor filter, which increases with wavelength. As τ decreases, this constraint detects larger neighbourhoods about the singular points. In effect, the left-hand side of Eqn (52) is the inverse scale-space distance to a singular point. For example, Figure 12F shows phase contours that survive the stability constraint, with $\tau = 1.25$, while Figure 12G shows the phase contours in regions removed by the constraint, which amounts to about 20% of the entire scale-space area.

7 Multiple Motions

When an affine or constant model is a good approximation to the image motion, the area-based regression methods using gradient-based constraints are very effective. One of the main reasons is that, with these methods one estimates only 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). Another reason is the simplicity of the linear estimation method that follows from the gradient-based constraints and the linear parametric motion models.

The problem with such model-based methods 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). Another exciting development is the use of mixture models where multiple parameterized models are used simultaneously in a given image neighorhood to model and estimate the optical flow.

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 Barron, D J Fleet, and S S Beauchemin. Performance of optical flow techniques. *International Journal of Computer Vision*, 12:43–77, 1994.

[7] 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.

[8] 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.

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

[10] D J Fleet. *Measurement of Image Velocity.* Kluwer Academic Press, Boston, 1992.

[11] D J Fleet. Disparity from local, weighted phase correlation. In *IEEE Internat. Conf. on SMC*, pages 48–56, 1994.

[12] 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.

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

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

[15] 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.

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

[17] 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.

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

[19] A D Jepson and M J Black. Mixture models for optical flow computation. In *IEEE Conference on Computer Vision and Pattern Recognition*, pages 307–314, 1993.

[20] 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.

[21] C Kuglin and D Hines. The phase correlation image alignment method. In *IEEE Int. Conf. Cybern. Soc.*, pages 163–165, 1975.

[22] 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.

[23] 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.

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

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

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

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

[28] 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.

[29] 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.

[30] 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.

[31] Y Weiss. Smoothness in layers: Motion segmentation using nonparametric mixture estimation. In *IEEE Conference on Computer Vision and Pattern Recognition*, pages 520–527, 1997.