```
Published in:   Handbook of Computer Vision and Applications,
                eds.  B Jähne, H Haussecker, and P Geissler,
                volume 2, chapter 14, pages 297-422,
                Academic Press, Spring 1999.
Last modified:  28 Dec 1998.
```

14 Bayesian Multi-Scale Differential Optical Flow

Eero P. Simoncelli

Center for Neural Science, and
Courant Institute of Mathematical Sciences
New York University

14.1 Introduction

Images are formed as projections of the three-dimensional world onto a two-dimensional light-sensing surface. The brightness of the image at each point indicates how much light was absorbed by the surface at that spatial position at a particular time (or over some interval of time). When an object in the world moves relative to the sensor surface, the two-dimensional projection of that object moves within the image. The movement of the projection of each point in the world is referred to as the image velocity or the *motion field*.

The estimation of the image motion field is generally assumed to be the first goal of motion processing in machine vision systems. Motion estimation is also crucial for compression of image sequences (e.g., the MPEG video compression standard uses motion-compensated prediction). There is also evidence that this sort of computation is performed by biological systems. As an approximation to this, computer vision techniques typically compute an estimate of the motion field known as the *optical flow*. The idea is to measure the apparent motion of local regions of the *image brightness pattern* from one frame to the next. In doing this, one is assuming that these intensity patterns are preserved from frame to frame. As many authors have pointed out, the optical flow $\boldsymbol{f}$ is *not* always the same as the motion field $\boldsymbol{v}$ (eg, [1; 2]).

There are many methods of computing optical flow. The most common are correlation, gradient, spatiotemporal filtering, and Fourier phase or energy approaches. Although based on somewhat different assumptions, these approaches are closely related, and can be made identical with proper formulation and choice of parameters [3; 4]. Correlation (usually over a local window) is by far the most prevalent technique. This is presumably due to a combination of intuitive directness and ease of hardware implementation. But a recent study by Barron et. al. suggests that gradient-based implementations have been the most accurate [5]. In addition, the gradient solution is efficient (because the solution can be computed analytically rather than via optimization), and produces sub-pixel displacement estimates. A drawback of the gradient approach is that it may only be used for small displacements. But this difficulty can be alleviated using a multi-scale coarse-to-fine algorithm. This chapter provides a practical description of a Bayesian multi-scale gradient-based optical flow estimation algorithm, based on work previously published in [6; 4; 7].

14.2 Differential formulation

Gradient formulations of optical flow begin with the differential *brightness constancy constraint equation* [8]:

$$\boldsymbol{\nabla}^T g\ \boldsymbol{f} + g_t = 0, \tag{14.1}$$

where $\boldsymbol{\nabla} g$ and g_t are the spatial image gradient and temporal derivative, respectively, of the image at a given spatial location and time (for notational simplicity, these parameters are omitted). The equation places a single linear constraint on the two-dimensional velocity vector $\boldsymbol{f}$ (at each point in space and time). As such, one cannot solve for velocity without imposing some additional constraint. This inherent indeterminacy is commonly known as the *aperture problem*.[1] In locations where the spatial gradient

[1]The expression refers to the fact that the motion of a moving one-dimensional pattern viewed through a circular aperture is ambiguous. Actually, the problem is not really due to the aperture, but to the one-dimensionality of the spatial image structure.

vanishes, the equation provides no constraint on the velocity vector. This is sometimes called the *blank wall problem*.

Typically, the aperture problem is overcome by imposing some form of smoothness on the field of velocity vectors. Many formulations use global smoothness constraints [8], which require global optimization[2]. Alternatively, one may assume locally constant velocity and combine linear constraints over local spatial (or temporal) regions [10]. This is accomplished by writing a weighted sum-of-squares error function based on the constraints from each point within a small region, where the points are indexed by a subscript $i \in \{1, 2, \ldots n\}$:

$$E(\boldsymbol{f}) \quad = \quad \sum_i w_i \left[\boldsymbol{\nabla}^T g(\boldsymbol{x}_i, t) \boldsymbol{f} + g_t(\boldsymbol{x}_i, t) \right]^2 , \tag{14.2}$$

where w_i is a set of *positive* weights.

To compute a linear least-squares estimate (LLSE) of $\boldsymbol{f}$ as a function of measurements $\boldsymbol{\nabla} g$ and g_t, consider the gradient (with respect to $\boldsymbol{f}$) of this quadratic expression:

$$\boldsymbol{\nabla}_{\boldsymbol{f}} E(\boldsymbol{f}) = 2 \sum \boldsymbol{\nabla} g \left[\boldsymbol{\nabla}^T g \; \boldsymbol{f} + g_t \right] = 2 \left[\boldsymbol{M} \boldsymbol{f} + \boldsymbol{b} \right] \tag{14.3}$$

where

$$\boldsymbol{M} = \sum \boldsymbol{\nabla} g \boldsymbol{\nabla}^T g = \begin{bmatrix} \sum g_x^2 & \sum g_x g_y \\ \sum g_x g_y & \sum g_y^2 \end{bmatrix}, \quad \boldsymbol{b} = \begin{bmatrix} \sum g_x g_t \\ \sum g_y g_t \end{bmatrix}. \tag{14.4}$$

and all of the summations are over the patch, weighted by w_i as in (14.2)). The $(\boldsymbol{x}_i, t)$ parameters have been omitted to simplify notation.

Setting the gradient expression equal to the zero vector gives the least-squares velocity estimate:

$$\hat{\boldsymbol{f}} = -\boldsymbol{M}^{-1} \boldsymbol{b}, \tag{14.5}$$

assuming that the matrix $\boldsymbol{M}$ is invertible. Notice that matrix $\boldsymbol{M}$ and the vector $\boldsymbol{b}$ are both composed of blurred quadratic combinations of the spatial and temporal derivatives.

Despite the combination of information over the patch, it is important to recognize that the matrix $\boldsymbol{M}$ can still be singular. In particular, one cannot solve for the velocity in regions of the image where the intensity varies only one-dimensionally (the *extended aperture problem*) or zero-dimensionally (the *extended blank wall problem*). In addition, this basic formulation has difficulties at occlusion boundaries (where two motions can coexist within the same local region), or when image brightness changes are due to photometric effects.

[2]Although Weiss has recently shown that some solutions may be localized through the use of Green's functions (eg, [9]).

Before introducing a model for uncertainty, it should be noted that the basic gradient approach may be extended in a number of ways. One can incorporate higher-order differential measurements (e.g., [11; 12]), or impose stronger constraints on the velocity field (for example, affine motion [13] or rigid-body motion [14]). A total least squares formulation is given in [15], although this solution is difficult to stabilize. The least-squares combination of local constraints may be replaced with a robust combination rule to give improved handling of occlusion boundaries [16]. The most promising recent development are techniques that simultaneously estimate motion and segment the scene into coherently moving regions (e.g., [17; 18; 19; 20; 21; 9]). Finally, gradient-based approaches have been shown to to be closely related to visual motion processing in mammals (e.g., [3; 22; 23; 24; 25; 26]).

14.3 Uncertainty model

Each of the quantities in (14.1)) is an idealization. First, we do not have the actual spatial or temporal derivatives, but *estimates* of these derivatives that are corrupted by image noise, filter inaccuracies, quantization, etc. Second, the equation is a constraint on the optical flow, but we are interested in estimating the *motion field*. As explained earlier, these two quantities often differ because changes in image intensities can be caused by non-motion effects.

These idealizations can be made explicit by introducing a set of additive random variables. Define $\tilde{\boldsymbol{f}}$ as the optical flow, and $\boldsymbol{f}$ as the actual velocity field. The difference between these may be described using a random variable, $\boldsymbol{n}_1$,

$$\tilde{\boldsymbol{f}} = \boldsymbol{f} + \boldsymbol{n}_1$$

Similarly, let $\tilde{g}_t$ be the actual temporal derivative, and g_t the measured derivative. Then

$$g_t = \tilde{g}_t + n_2$$

with n_2 a random variable characterizing the uncertainty in this measurement relative to the true derivative. We assume that the spatial derivatives are measured more accurately than the temporal derivatives, and thus the equation does not include any term for uncertainty in these quantities.

Now the gradient constraint applies to the actual derivatives, and the optical flow vector, and so we may write:

$$\begin{aligned} 0 &= \boldsymbol{\nabla}^T g\ \tilde{\boldsymbol{f}} + \tilde{g}_t \\ &= \boldsymbol{\nabla}^T g(\boldsymbol{f} - \boldsymbol{n}_1) + g_t - n_2 \\ \Rightarrow\ \boldsymbol{\nabla}^T g\ \boldsymbol{f} + g_t &= \boldsymbol{\nabla}^T g\ \boldsymbol{n}_1 + n_2. \end{aligned} \qquad (14.6)$$

This equation gives us a probabilistic relationship between the image *motion field* and the *measurements* of spatiotemporal gradient. It accounts for errors in our derivative measurements, and for deviations of the velocity field from the optical flow. But it still assumes that the underlying optical flow constraint is valid.

In order to make use of this formulation, we must characterize the random variables n_i in our definitions. It is desirable to choose these probability distributions such that $\boldsymbol{f}$ may be estimated analytically (as opposed to numerically). A common choice is to use independent zero-mean Gaussian distributions. The right side of (14.6)) is a zero-mean Gaussian random variable with variance equal to $\boldsymbol{\nabla}^T g \; \boldsymbol{\Lambda}_1 \boldsymbol{\nabla} g + \Lambda_2$, where $\boldsymbol{\Lambda}_1$ and Λ_2 are a covariance matrix and a variance corresponding to $\boldsymbol{n}_1$ and n_2, respectively. We interpret the equation as providing a conditional probability expression:

$$\mathcal{P}(g_t \mid \boldsymbol{f}, \boldsymbol{\nabla} g) \propto \exp\left\{ -\frac{1}{2}(\boldsymbol{\nabla}^T g \; \boldsymbol{f} + g_t)(\boldsymbol{\nabla}^T g \; \boldsymbol{\Lambda}_1 \boldsymbol{\nabla} g + \Lambda_2)^{-1}(\boldsymbol{\nabla}^T g \; \boldsymbol{f} + g_t)\right\}$$

Bayes' rule may be used to write the desired conditional probability:

$$\mathcal{P}(\boldsymbol{f} \mid \boldsymbol{\nabla} g, g_t) = \frac{\mathcal{P}(g_t \mid \boldsymbol{f}, \boldsymbol{\nabla} g)\mathcal{P}(\boldsymbol{f})}{\mathcal{P}(g_t)}.$$

For the prior distribution $\mathcal{P}(\boldsymbol{f})$, we choose a zero-mean Gaussian with covariance $\boldsymbol{\Lambda}_p$. This imposes a preference for slower speeds[3]. The denominator, $\mathcal{P}(g_t)$, is only present for normalization purposes and doesn't affect the relative probabilities. The resulting distribution $\mathcal{P}(\boldsymbol{f}|\boldsymbol{\nabla} g, g_t)$ is Gaussian:

$$\begin{aligned}
&\mathcal{P}(\boldsymbol{f}|\boldsymbol{\nabla} g, g_t) && (14.7)\\
\propto\; &\exp\left\{ -\frac{1}{2}(\boldsymbol{\nabla}^T g \; \boldsymbol{f} + g_t)^T(\boldsymbol{\nabla}^T g \; \boldsymbol{\Lambda}_1 \boldsymbol{\nabla} g + \Lambda_2)^{-1}(\boldsymbol{\nabla}^T g \; \boldsymbol{f} + g_t)\right\} && (14.8)\\
&\cdot \exp\left\{-\frac{1}{2}\boldsymbol{f}^T \boldsymbol{\Lambda}_p^{-1}\boldsymbol{f}\right\}\\
=\; &\exp\left\{-\frac{1}{2}\boldsymbol{f}^T\left[\boldsymbol{\nabla} g(\boldsymbol{\nabla}^T g \; \boldsymbol{\Lambda}_1 \boldsymbol{\nabla} g + \Lambda_2)^{-1}\boldsymbol{\nabla}^T g + \boldsymbol{\Lambda}_p^{-1}\right]\boldsymbol{f}\right.\\
&\quad - g_t(\boldsymbol{\nabla}^T g \; \boldsymbol{\Lambda}_1 \boldsymbol{\nabla} g + \Lambda_2)^{-1}\boldsymbol{\nabla}^T g \; \boldsymbol{f}\\
&\quad \left.-\frac{1}{2} g_t(\boldsymbol{\nabla}^T g \; \boldsymbol{\Lambda}_1 \boldsymbol{\nabla} g + \Lambda_2) g_t\right\}\\
=\; &\exp\left\{-\frac{1}{2}(\boldsymbol{\mu}_f - \boldsymbol{f})^T \boldsymbol{\Lambda}_f^{-1}(\boldsymbol{\mu}_f - \boldsymbol{f})\right\}. && (14.9)
\end{aligned}$$

[3]Such a preference has been suggested to play a role in human perception(e.g., [4; 26]).

The covariance matrix, $\boldsymbol{\Lambda}_f$, and mean vector, $\boldsymbol{\mu}_f$ may be derived by completing the square in the exponent:

$$\begin{aligned}\boldsymbol{\Lambda}_f &= \left[\boldsymbol{\nabla} g(\boldsymbol{\nabla}^T g\ \boldsymbol{\Lambda}_1 \boldsymbol{\nabla} g + \Lambda_2)^{-1} \boldsymbol{\nabla}^T g + \boldsymbol{\Lambda}_p^{-1}\right]^{-1} \\ \boldsymbol{\mu}_f &= -\boldsymbol{\Lambda}_f \boldsymbol{\nabla} g(\boldsymbol{\nabla}^T g\ \boldsymbol{\Lambda}_1 \boldsymbol{\nabla} g + \Lambda_2)^{-1} g_t.\end{aligned}$$

The advantage of the Gaussian form is that it is parameterized by these two quantities that are computed in analytic form from the derivative measurements.

If $\boldsymbol{\Lambda}_1$ is assumed to be a diagonal matrix with diagonal entry λ_1, and the scalar variance of n_2 is rewritten as $\lambda_2 \equiv \Lambda_2$, then the solution becomes:

$$\begin{aligned}\boldsymbol{\Lambda}_f &= \left[\frac{\boldsymbol{M}}{(\lambda_1 \|\boldsymbol{\nabla} g\|^2 + \lambda_2)} + \boldsymbol{\Lambda}_p^{-1}\right]^{-1} \\ \boldsymbol{\mu}_f &= -\boldsymbol{\Lambda}_f \frac{\boldsymbol{b}}{(\lambda_1 \|\boldsymbol{\nabla} g\|^2 + \lambda_2)}.\end{aligned} \tag{14.10}$$

where matrix $\boldsymbol{M}$ and vector $\boldsymbol{b}$ are defined as in (14.4)), but without the summations. Note that multiplying $\boldsymbol{\Lambda}_p$, λ_1 and λ_2 by a common scale factor will not affect the mean, $\boldsymbol{\mu}_f$, of the distribution (although it *will* scale the variance).

The maximum a posteriori (MAP) estimate is simply the mean, $\boldsymbol{\mu}_f$, since the distribution is Gaussian. This solution is very similar to that specified by (14.3)). The differences are that (1) the addition of the prior variance $\boldsymbol{\Lambda}_p$ ensures the invertibility of the matrix $\boldsymbol{M}$, and (2) the quadratic derivative terms in $\boldsymbol{M}$ and $\boldsymbol{b}$ are modified by a compressive nonlinearity. That is, for regions with low contrast (i. e., small $\|\boldsymbol{\nabla} g\|^2$), the λ_2 term dominates the divisor of $\boldsymbol{M}$. For high-contrast regions, the $\lambda_1 \|\boldsymbol{\nabla} g\|^2$ term will normalize the magnitude of the quadratic terms in $\boldsymbol{M}$. This seems intuitively reasonable: When the contrast (SNR) of the signal is low, an increase in contrast should increase our certainty of the velocity estimate. But as the contrast increases above the noise level of the signal, the certainty should asymptotically reach some maximum value rather than continuing to rise quadratically. The noise term n_2 accounts for errors in the derivative measurements. At low signal amplitudes, these will be the dominant source of error. The term $\boldsymbol{n}_1$ accounts for failures of the constraint equation. At high contrasts, these will be the dominant source of error.

The solution described thus far computes velocity for one point in isolation. As described in Sect. 14.2, the constraint at a single location is insufficient to uniquely specify a solution. We may therefore only compute the component of flow that is normal (perpendicular) to the local orientation. In the solution above, the mean will be (approximately) the *normal flow* vector, and the width of these distributions in the direction perpendicular to the normal direction will be determined by $\boldsymbol{\Lambda}_p$. The variance in the normal direction will be determined by both $\mathbf{\Lambda}_p$ and the trace of $\boldsymbol{M}$ (i. e., the sum of the squared magnitudes of the spatial derivatives).

If normal flow (along with variance information) does not provide a satisfactory input for the next stage of processing, then one can combine information in small neighborhoods (as in (14.2)). We now need an uncertainty model for the entire neighborhood of points. The simplest assumption is that the noise at each point in the neighborhood is independent. In practice this will not be correct. Nevertheless, as a first approximation, if we treat the uncertainties as pointwise independent, then the resulting mean and variance are easy to calculate:

$$\mathbf{\Lambda}_f = \left[\sum_i \frac{w_i \boldsymbol{M}_i}{(\lambda_1 \|\boldsymbol{\nabla} g(\boldsymbol{x}_i, t)\|^2 + \lambda_2)} + \mathbf{\Lambda}_p^{-1}\right]^{-1} \tag{14.11}$$

$$\boldsymbol{\mu}_f = -\mathbf{\Lambda}_f \sum_i \frac{w_i \boldsymbol{b}_i}{(\lambda_1 \|\boldsymbol{\nabla} g(\boldsymbol{x}_i, t)\|^2 + \lambda_2)}, \tag{14.12}$$

where, as before, w_i is a weighting function over the patch, with the points in the patch indexed by i. Here, the effect of the nonlinearity on the combination of information over the patch is to provide a type of gain control mechanism. If we ignore λ_2, the solution above normalizes the information, equalizing the contribution from each point in the neighborhood by the magnitude of the spatial gradient. We will refer to this in later sections as the *basic* solution.

In the basic solution, information is combined over fixed size patches, using a fixed weighting function. An adaptive version of this algorithm could proceed by blurring over larger and larger regions (i. e., diffusion) until the magnitude of the variance (determinant of the variance matrix) is below some threshold. Since the variance matrix $\mathbf{\Lambda}_f$ describes a two-dimensional shape, this could be done directionally (i. e., anisotropic diffusion), averaging pixels which lie in the direction of maximal variance until the variance in this direction was below a threshold.

To illustrate the solution given in (14.12), we consider the response to a moving square. We have added a small amount of Gaussian-distributed white noise. Figure 14.1a shows one frame of the input image, along with the resulting distributions near the corner, on a side, and in the center. In the corner, the output is a fairly narrow distribution centered near the correct velocity. The error in the mean is due to the noise in the input. On the side, the ambiguity of the motion along the edge (i. e., the aperture problem) is indicated by the elongated shape of the distribution. In the center, the motion is completely ambiguous and the resulting distribution is essentially the prior. We also show the response for a low-contrast moving square, with the same amount of Gaussian noise, in Fig. 14.1b. Note that the velocity distribution corresponding to the corner is now substantially broader, as is that of the edge.

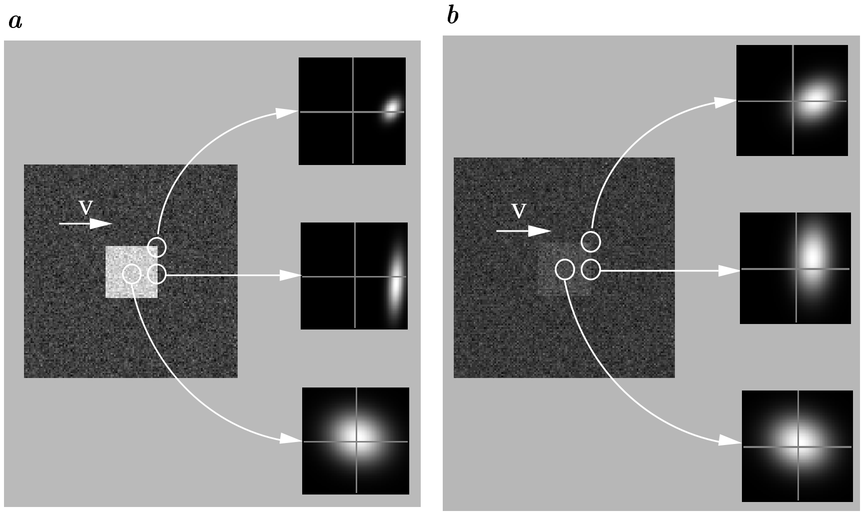


Figure 14.1: Output of the Bayesian algorithm in different regions of a moving square image at two different contrasts. Each plot shows a Gaussian density over the space of image velocities, computed using (14.12). The noise added to both sequences was of the same amplitude.

14.4 Coarse-to-fine estimation

The estimation of gradients from discretely sampled image sequences is prone to error. Some of the errors are due to poor choice of filter kernels: we address the issue of filter kernel design in the next section. For now, we focus on the problem of large translations. If motion from one frame of an image sequence to the next is too large (typically, more than 2 pixels), one cannot estimate the gradient accurately. The problem may be viewed easily in the Fourier domain, where it is evident as temporal aliasing. Consider a one-dimensional signal that is moving at a constant velocity. The power spectrum of this signal lies on a line through the origin [27]. We assume that the spatial sampling is dense enough to avoid aliasing (i. e., the images are spatially band-limited before sampling, at a rate above the Nyquist limit). The temporal sampling of the imagery causes a replication of the signal spectrum at temporal frequency intervals of $2\pi/T$ radians, where T is the time between frames. This is illustrated in Fig. 14.2.

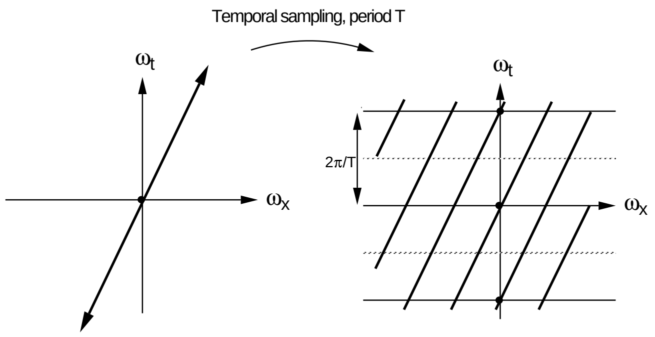


Figure 14.2: *Illustration of the temporal aliasing problem. On the left is an idealized depiction of the power spectrum of a one-dimensional pattern translating at speed v. The power spectrum is distributed along the heavy line, which has a slope of v. Temporally sampling the signal causes a replication of its power spectrum in the Fourier domain at temporal frequency intervals of $2\pi/T$. When the velocity of the signal is high, the replicas of the spectrum will interfere with the filter (gradient) measurements.*

Now consider the gradient-based estimation of optical flow, in the Fourier domain. In particular, the energy function given in (14.2) may be rewritten:

$$\begin{aligned} E(\boldsymbol{f}) &= \sum_{\boldsymbol{x}} \left| \boldsymbol{f}^T \boldsymbol{\nabla} g + g_t \right|^2 \\ &= \sum_{\boldsymbol{k}} \left| \hat{g}(\boldsymbol{k})(\boldsymbol{f}^T \boldsymbol{k}) + \hat{g}(\boldsymbol{k})\omega \right|^2 \\ &= \sum_{\boldsymbol{k}} \left[(\boldsymbol{f}^T \boldsymbol{k}) + \omega \right]^2 |\hat{g}(\boldsymbol{k})|^2 \qquad (14.13) \end{aligned}$$

where the sum on the first line is over all image pixels and the sums on the latter two lines are over all frequencies, $\boldsymbol{k}$. We have used Parseval's rule to switch to the Fourier domain, and the fact that the Fourier transform of the derivative operator in, for example, the $x-$ direction is ik_1. The term in square brackets is the squared ω-distance between the point $\boldsymbol{k}$ and the plane defined by $\boldsymbol{f}^T \boldsymbol{k} = -\omega$. This equation is precisely in the form of a least-squares planar regression error function, weighted by the image power spectrum, $|\hat{g}(\boldsymbol{k})|^2$. Thus, the replicated spectra of Fig. 14.2 can confuse a motion estimation algorithm.

An important observation concerning this type of temporal aliasing is that it affects the higher spatial frequencies of an image. In particular, for a fixed global velocity, those spatial frequencies moving more than half of their period per frame will be aliased, but the lower spatial frequencies will be left intact. This suggests a simple, but effective approach for avoiding the problem of temporal aliasing: Estimate the velocity of a lowpass filtered copy of the image. Note that this "prefilter" must be quite large in spatial extent, in inverse proportion to its small spatial-frequency extent. Given imagery that contains only a single global motion, or a motion field that varies slowly, we could stop our computation at this point. But typical scenes contain more complex motion fields, which will not be captured by these low-frequency estimates.

In order to get better estimates of *local* velocity, higher-frequency bands must be used, with spatially smaller filters. What we would like to do is to use the coarse motion estimate to "undo" the motion, roughly stabilizing the position of the image over time. Then higher frequency filters can be used to extract local perturbations to the large-scale motion. Specifically, we can use higher frequency filters to estimate optical flow on the warped sequence, and this "optical flow correction" may then be composed with the previously computed optical flow to give a new optical flow estimate. This correction process may be repeated at finer and finer scales of a multi-scale pyramid representation.

There are two mechanisms that one could imagine using to stabilize the image. In an interactive setting (i. e., a biological or robotically controlled visual system), the sensors can be moved so as to track a given point or object in the scene. This action reduces the image velocity of the object to zero.

Alternatively, in image-processing situations, where the image-gathering has already occurred, we can warp a spatially and temporally localized region of the image content in a direction opposite to the computed motion. For our purposes, we compute the warped image sequence:

$$\mathcal{W}\{g, \boldsymbol{f}\}(\boldsymbol{x}, t + \Delta t) = g(\boldsymbol{x} - \boldsymbol{f}\Delta t, t + \Delta t),$$

where $\boldsymbol{f}$ is the warp vector field corresponding to the velocity estimated from the coarser scale measurements. Note that the warping only need be done over a range of Δt that covers the temporal extent of the derivative filters that will be applied.

We will concentrate on the warping approach here, although many of the observations apply to the tracking case as well. The warping procedure may be applied recursively to higher and higher frequency subbands. This "coarse-to-fine" estimation process is illustrated in Fig. 14.3. This type of approach has been suggested and used by a number of authors [10; 28; 29; 30; 31].

As described above, in order to generate estimates at different scales, we can apply the differential algorithm to lowpass prefilters of different bandwidth. To illustrate the effectiveness of this technique, consider a simple test pattern containing a disk of high-frequency texture moving at a fairly high velocity. This is illustrated in Fig. 14.4. A local operator attempting to compute motion in the center of the disk would fail. But the multi-scale algorithm is able to lock onto the coarse scale motion of the disc.

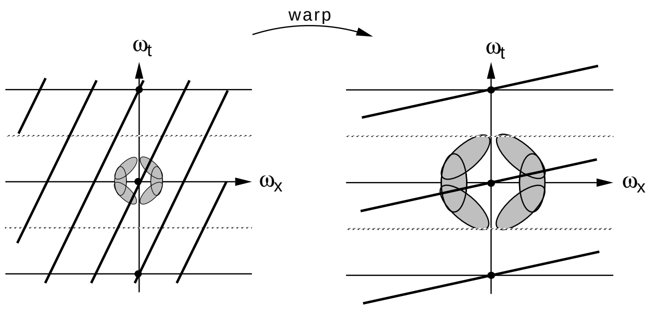


Figure 14.3: *Illustration of the coarse-to-fine approach for eliminating temporal aliasing. On the left is an idealized illustration of the (aliased) power spectrum of the signal. A low frequency subband is used to estimate the motion of this signal. These estimates are then used to "undo" the motion, leaving a smaller, unaliased residual motion (shown on the right – note that the spectrum lies on a line of smaller slope). This motion may then be estimated using higher frequency derivative filters.*

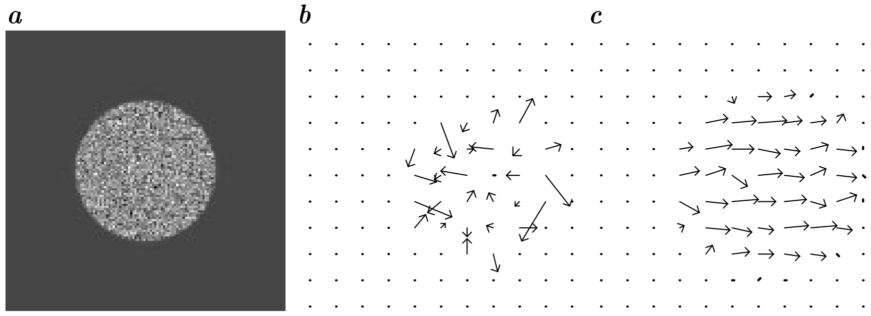


Figure 14.4: ***a*** *The stimulus, a rapidly moving disc containing fine-scale texture.* ***b*** *Optical flow computed, direct gradient algorithm.* ***c*** *Optical flow computed using coarse-to-fine gradient algorithm. The single dots correspond to optical flow vectors of length zero.*

As we have described, a coarse-to-fine algorithm can be used to handle problems of temporal aliasing. It is also a technique for imposing a prior smoothness constraint (see, for example, [32]). This basic technique does, however, have a serious drawback. If the coarse-scale estimates are incorrect, then the fine-scale estimates will have no chance of correcting the errors.

To fix this, we must have knowledge of the error in the coarse-scale estimates. Since we are working in a probabilistic framework, and we have information describing the uncertainty of our measurements, we may use this information to properly combine the information from scale to scale.

We define a *state evolution* equation with respect to scale:

$$\boldsymbol{f}(l+1) = \boldsymbol{E}(l)\boldsymbol{f}(l) + \boldsymbol{n}_0(l), \qquad \boldsymbol{n}_0(l) \sim N(0, \boldsymbol{\Lambda}_0)$$

where l is an index for scale (larger values of l correspond to finer scale), $\boldsymbol{E}(l)$ is the linear interpolation operator used to extend a coarse scale flow field to finer resolution, and $\vec{n_0}$ is a random variable corresponding to the certainty of the prediction of the fine-scale motion from the coarse-scale motion. We assume that the $\vec{n_0}(l)$ are independent, zero-mean, and normally distributed. This type of scale-to-scale Markov relationship has been explored in an estimation context (e. g., [33]).

We also define the *measurement* equation:

$$-g_t(l) = \boldsymbol{\nabla}^T g(l)\boldsymbol{f}(l) + (n_2 + \boldsymbol{\nabla}^T g(l)\boldsymbol{n}_1)$$

as in Sect. 14.3. We will assume, as before, that the random variables are zero-mean, independent and normally distributed. Remember that this equation is initially derived from the total derivative constraint for optical flow. This equation is a bit different than the measurement equation used in most estimation contexts. Here, the linear operator relating the quantity to be estimated to the measurement g_t is *also* a measurement.

Given these two equations, we may write down the optimal estimator for $\boldsymbol{f}(l+1)$, the velocity at the fine scale, given an estimate for the velocity at the previous coarse scale, $\boldsymbol{\mu}_f(l)$, and a set of fine scale (gradient) measurements. The solution is in the form of a standard Kalman filter [34], but with the time variable replaced by the *scale*, l:

$$\begin{aligned}
\boldsymbol{\mu}_f(l+1) &= \boldsymbol{E}(l)\boldsymbol{\mu}_f(l) + \boldsymbol{\kappa}(l+1)\nu(l+1) \\
\boldsymbol{\Lambda}_f(l+1) &= \boldsymbol{\Lambda}'(l+1) - \boldsymbol{\kappa}(l+1)\boldsymbol{\nabla}^T g(l+1)\boldsymbol{\Lambda}'(l+1) \\
\boldsymbol{\kappa}(l+1) &= \frac{\boldsymbol{\Lambda}'(l+1)\boldsymbol{\nabla} g(l+1)}{\boldsymbol{\nabla}^T g(l+1)\left[\boldsymbol{\Lambda}'(l+1) + \boldsymbol{\Lambda}_1\right]\boldsymbol{\nabla} g(l+1) + \Lambda_2} \\
\nu(l+1) &= -g_t(l+1) - \boldsymbol{\nabla}^T g(l+1)\boldsymbol{E}(l)\boldsymbol{\mu}_f(l) \\
\boldsymbol{\Lambda}'(l+1) &= \boldsymbol{E}(l)\boldsymbol{\Lambda}_f(l)\boldsymbol{E}(l)^T + \boldsymbol{\Lambda}_0
\end{aligned}$$

Here, $\boldsymbol{\kappa}(l)$ is the *Kalman gain*, and $\nu(l)$ corresponds to an *innovations* process which represents the new information contributed by the measurements at level l.

The problem with the equations given above is that we cannot compute the derivative measurements at scale l without making use of the velocity estimate at scale $l-1$, due to the temporal aliasing problem. In order to avoid this problem, we must write $\nu(l)$ in terms of derivatives of the warped sequence. We rewrite $\nu(l)$ as follows:

$$\begin{aligned}\nu(l+1) &= -g_t(l+1) - \boldsymbol{\nabla} g(l+1)\cdot \boldsymbol{E}(l)\boldsymbol{\mu}_f(l)\\ &= -\frac{d}{dt} g\left(\boldsymbol{x} + t\boldsymbol{E}(l)\boldsymbol{\mu}_f(l), t\right)\\ &\approx -\frac{\partial}{\partial t}\mathcal{W}\{g(l+1), \boldsymbol{E}(l)\boldsymbol{\mu}_f(l)\}(x,y,t)\end{aligned}$$

Thus, the innovations process is computed as the temporal derivative of the the image at scale $l+1$, after it has been warped with the interpolated flow field from scale l. In order to make the solution computationally feasible, we ignore the off-diagonal elements in $\boldsymbol{\Lambda}'(l+1)$ (i. e., the correlations between adjacent interpolated flow vectors).

Now the Kalman solution may be put into the alternative "update" form by use of the following matrix identity [34]:

$$B^{-1} + C^T A^{-1} C = \left[B - BC^T(CBC^T + A)^{-1}CB\right]^{-1}.$$

The left side corresponds to the inverse of the updated covariance matrix given in the Kalman equations above:

$$\begin{aligned}\boldsymbol{\Lambda}_f(l+1) &= \left[\boldsymbol{\Lambda}'(l+1)^{-1} + \boldsymbol{\nabla} g(\boldsymbol{\nabla}^T g\ \boldsymbol{\Lambda}_1 \boldsymbol{\nabla} g + \Lambda_2)^{-1}\boldsymbol{\nabla}^T g\right]^{-1}\\ &= \left[\boldsymbol{\Lambda}'(l+1)^{-1} + \frac{\boldsymbol{M}}{(\lambda_1\|\boldsymbol{\nabla} g\|^2 + \lambda_2)}\right]^{-1}, \qquad (14.14)\end{aligned}$$

Similarly, we may rewrite the updated mean vector as:

$$\begin{aligned}\boldsymbol{\mu}_f(l+1) &= \boldsymbol{E}(l)\boldsymbol{\mu}_f(l) + \boldsymbol{\Lambda}_f(l+1)\boldsymbol{\nabla} g(\boldsymbol{\nabla}^T g\ \boldsymbol{\Lambda}_1 + \Lambda_2)^{-1}\nu(l+1)\\ &= \boldsymbol{E}(l)\boldsymbol{\mu}_f(l) + \boldsymbol{\Lambda}_f(l+1)\frac{\boldsymbol{b}'}{(\lambda_1\|\boldsymbol{\nabla} g\|^2 + \lambda_2)}, \qquad (14.15)\end{aligned}$$

where the vector $\boldsymbol{b}'$ is defined by

$$\boldsymbol{b}' = \boldsymbol{\nabla} g \nu(l+1).$$

These mean and covariance expressions are the same as those of (14.10) except that: (1) the prior covariance $\boldsymbol{\Lambda}_p$ has been replaced by $\boldsymbol{\Lambda}'(l+1)$, (2) the vector $\boldsymbol{b}$ has been replaced by $\boldsymbol{b}'$, which is computed in the same manner but using the *warped* temporal derivative measurements, and (3) the mean $\boldsymbol{\mu}_f(l+1)$ is augmented by the interpolated estimate from the previous scale.

Figure 14.5 illustrates the effectiveness of this "Kalman filter over scale". The stimulus is a slowly moving textured disk, *with noise added.* The ordinary coarse-to-fine gradient algorithm gives terrible results, because the noise leads to large errors in the coarse-scale velocity estimates that cannot

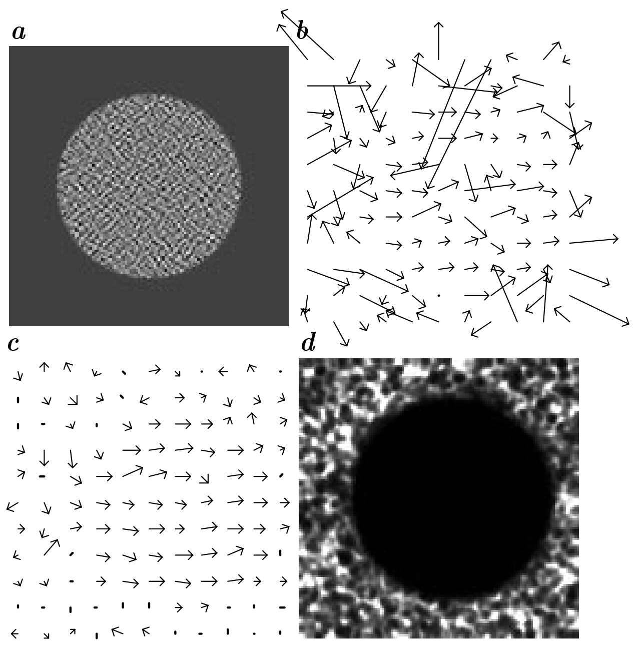


Figure 14.5: *Example using the covariance-propagating coarse-to-fine algorithm.* ***a*** *The stimulus, a slowly moving disc containing fine-scale texture in the presence of additive Gaussian white noise.* ***b*** *Optical flow computed using standard coarse-to-fine gradient algorithm.* ***c*** *Optical flow computed using Kalman-like coarse-to-fine gradient algorithm with covariance propagation.* ***d*** *The determinant of the terminal covariance matrix, indicating uncertainty of the estimate.*

be corrected at finer scales. The covariance-propagating version defined by (14.14) and 14.15) produces better estimates (i. e., the mean vectors are closer to the actual flow), and the covariance information accurately indicates the more uncertain vectors.

Given that the derivative measurements will fail when the image velocity is too high, a more sophisticated version of this algorithm could prune the tree during the coarse-to-fine operation. That is, we can terminate the recursion at a given location (x, y, t) and level l if the interpolated covariance estimate from the previous scale, $\mathbf{\Lambda}'(l)$ is too large.

14.5 Implementation issues

In this section we will discuss some important issues that arise when implementing the algorithms discussed thus far.

Table 14.1: *First-order derivative kernels. Shown are pairs of derivative (D) and interpolator (B) kernels of various sizes.*

${}^3B$	0.223755	0.552490	0.223755			
${}^3D$	−0.453014	0	0.453014			
${}^4B$	0.092645	0.407355	0.407355	0.092645		
${}^4D$	−0.236506	−0.267576	0.267576	0.236506		
${}^5B$	0.036420	0.248972	0.429217	0.248972	0.036420	
${}^5D$	−0.108415	−0.280353	0	0.280353	0.108415	
${}^6B$	0.013846	0.135816	0.350337	0.350337	0.135816	0.013846
${}^6D$	−0.046266	−0.203121	−0.158152	0.158152	0.203121	0.046266

14.5.1 Derivative filter kernels

The choice of convolution kernels used to estimate gradients can have a substantial impact on the accuracy of the estimates [35; 36; 37], and yet many authors do not even describe the filters that they use. The most common choice in the literature is a first-order difference. This type of differentiation arises naturally from the definition of continuous derivatives, and is reasonable when the spacing between samples is well below the Nyquist limit. But simple first order differences are likely to produce poor results for the optical flow problem when applied directly to the input imagery, especially in highly textured regions (i. e., regions with much fine-scale content).

In the digital signal processing community, there has been a fair amount of work on the design of discrete differentiators (see e.g., [38]). This work is usually based on approximating the derivative of a continuous sinc function. The difficulty with this approach is that the resulting kernels typically need to be quite large in order to be accurate. In the computer vision literature, many authors have used sampled Gaussian derivatives which exhibit better differentiation than simple differences, but are less computationally expensive than sinc functions.

We have previously described a simple design procedure for matched pairs of one-dimensional kernels (a lowpass kernel and a differentiator) suitable for gradient estimation [4; 36; 37].[4] Let $\hat{B}(\boldsymbol{k})$ be the Discrete Fourier transform (DFT) of the interpolator (often called the "prefilter"), and $\hat{D}(\boldsymbol{k})$ the DFT of the derivative filter. Then our design method attempts to meet the following requirements:

1. The derivative filters must be good approximations to the derivative of the prefilter. That is, for a derivative along the x-axis, we would like

[4]Fleet and Langley have designed recursive filters for these purposes [39].

$jk_1\hat{B}(k) \approx \hat{D}(k)$, where k_1 is the component of the frequency coordinate in the x direction.

2. The lowpass prefilter should be symmetric, with $\hat{B}(0) = 1$.
3. For computational efficiency and ease of design, the prefilter should be separable. In this case, the derivatives will also be separable, and the design problem will be reduced to one dimension.
4. The design algorithm should include a model for signal and noise statistics (e.g., [40]).

Table 14.1 gives sample values for a set of derivative kernels and their associated prefilters, at three different sizes [37]. These give significantly improved performance in optical flow or orientation estimation tasks.

14.5.2 Averaging filter kernels

The algorithm also requires averaging over a patch, which is equivalent to applying a lowpass filter. We desire this lowpass filter to have only positive weights, since it will be used combine a set of squared constraints and should produce a positive value. There is a tradeoff in choosing the spatial extent of the filter. A large filter will produce better power spectral estimates by combining information over a larger region. But it is also more likely to combine inconsistent motions. The question can only be properly settled given a knowledge of the statistics of the motion of the imagery to be analyzed. We experimented with binomial blurring filters and found that separable application of the kernel $[0.0625, 0.25, 0.375, 0.25, 0.0625]$ produced reliable results without over-blurring.

14.5.3 Multi-scale warping

In Sect. 14.4, we discussed the implementation of coarse-to-fine algorithms to reduce temporal aliasing. Conceptually, this approach operates by using prefilters of varying bandwidths.

A more efficient technique for generating multi-scale representations is to construct an image pyramid [41], by recursively applying lowpass filtering and subsampling operations. In this case, the images at different scales are also represented at different sampling rates. Assuming the lowpass filter prevents aliasing, the effect of the subsampling in the Fourier domain is to stretch the spectrum out. This allows us to use the *same* derivative filters at each scale, rather than designing a whole family of derivative filters at different scales.

The algorithm begins by building a multi-scale pyramid on each frame of the input sequence, and computing the optical flow on the sequence of top level (lowest frequency) images using the computation specified by (14.12). An upsampled and interpolated version of this coarse, low-resolution flow field must then be used to warp the sequence of images in the next pyramid level. We used a simple bilinear interpolator in this case, since the optical flow is somewhat smooth due to the blurring operation. Optical flow is then computed on this warped sequence, and this "optical flow correction" is composed with the previously computed optical flow to give a new optical flow estimate. This correction process is repeated for each level of the pyramid until the flow fields are at the resolution of the original image sequence.

The warping equation is fairly unambiguous in the continuous case. But there are many ways in which one can implement a warping algorithm on discretely sampled image data. Consider the task of warping a frame at time t_1 back to time t_0. The primary issues are:

Indexing Should we use the velocity estimate at t_0 or t_1 (or a velocity estimate *between* the frames) as the warp field? Assuming the velocity vectors are in units of pixels/frame, does the velocity estimate at position $(\boldsymbol{x}, t)$ correspond to the displacement of intensity at $(\boldsymbol{x} - \boldsymbol{f}, t-1)$ to $(\boldsymbol{x}, t)$, or from $(\boldsymbol{x}, t)$ to $(\boldsymbol{x} + \boldsymbol{f}, t-1)$?

Order If our filters are several frames long and thus require warping of several frames, should we use the velocity estimates at each of these frames, or just the velocity estimate of the central frame?

Interpolation Given that velocity vector components are typically not multiples of the pixel spacing, how should we interpolate the intensities of the warped images?

We compared several different variants and chose a simple and efficient warping scheme. We assume an odd-length temporal derivative filter of length $2N_t + 1$, and we use a velocity field estimate associated with the center frame. Since our derivative filters are separable, we apply the spatial portion to N_t frames centered at frame 0. Let $g'(\boldsymbol{x}, t), -N_t \leq t \leq N_t$ be the set of spatially filtered frames. We then combine the temporal differentiation operation with the warping operation as follows:

$$g'_t(\boldsymbol{x}, 0) = \sum_{t=-N_t}^{N_t} d(t) I \left\{ g' \left(\boldsymbol{x} + t\, \hat{\boldsymbol{f}}(\boldsymbol{x}, 0),\ t \right) \right\}$$

where $d(t)$ is the temporal derivative kernel, $\hat{\boldsymbol{f}}$ is the previous estimate of optical flow, and $I\{\cdot\}$ is a bi-cubic spline interpolator used to evaluate g' at fractional-pixel locations.

14.5.4 Boundary handling

Convolution operations are used to compute the derivative filter responses, and to blur the energies. They are also used in coarse-to-fine schemes to construct the multi-resolution image pyramid. Traditionally, convolution boundaries are handled by computing *circular* convolution. That is, the image is treated as one period of a periodic signal. This often produces poor results because it associates the image content near one boundary with that near the opposite boundary.

There are many alternatives for handling edges. Let $h(n)$ be a one-dimensional signal, indexed by the discrete variable n, with $n = 0$ corresponding to the leftmost sample. Then we define an edge-handling mechanism (for the left edge) by assigning a value for the function f at negative values of n. Several example methods that we have experimented with are:

1. Reflect the image about its edge pixel (or just beyond the edge pixel): $h(-n) = h(n)$ (or $h(-n) = h(n-1)$).
2. Imbed the image in a "sea of zeros": $h(-n) = 0$, for each $n > 0$.
3. Repeat the edge pixel: $h(-n) = h(0)$
4. Reflect and invert (so as to preserve zeroth and first-order continuity): $h(-n) = 2h(0) - h(n)$.
5. Return zero for the convolution inner product whenever the filter kernel overhangs an edge of the image.

For the blurring and pyramid filtering operations we have found that reflection (item 1) is preferable. For the derivative operations, we choose to repeat the edge pixel (item 3).

14.6 Examples

We computed velocity field estimates for a set of synthetic and real image sequences in order to examine the behavior of the basic (first derivative) solution of (14.12).

14.6.1 Performance measures

In cases where we the velocity field is known, we can analyze the errors in our estimates. There are a number of ways to do this. The simplest measure is the squared magnitude of the difference between the correct and estimated flow:

$$E_{\mathrm{mag2}} = |\hat{\boldsymbol{f}} - \boldsymbol{f}|^2.$$

where $\boldsymbol{f}$ is the actual velocity, and $\hat{\boldsymbol{f}}$ is the estimate. Viewing an image containing these values at each spatial location often provides useful information about the spatial structure of the errors. Errors in optical flow are sometimes reported as a ratio of the error magnitude to magnitude of the actual flow, but this is problematic when the actual flow vectors are small.

Fleet and Jepson [42] used an error criterion based on the unit vector normal to the velocity plane in spatiotemporal frequency:

$$E_{\mathrm{angular}} = \arccos\left[\bar{\boldsymbol{u}}(\hat{\boldsymbol{f}}) \cdot \bar{\boldsymbol{u}}(\boldsymbol{f})\right],$$

where $\bar{\boldsymbol{u}}(\cdot)$ is a function producing a three-dimensional unit vector:

$$\bar{\boldsymbol{u}}(\boldsymbol{f}) = \frac{1}{\sqrt{|\boldsymbol{f}|^2+1}} \begin{bmatrix} \boldsymbol{f}_1 \\ \boldsymbol{f}_2 \\ 1 \end{bmatrix},$$

and the resulting angular error is reported in units of degrees.

We also define a measure of bias in order to quantify characteristic over- or under- estimation of velocity magnitudes:

$$E_{\text{bias}} = \frac{\boldsymbol{f} \cdot (\boldsymbol{f} - \hat{\boldsymbol{f}})}{|\boldsymbol{f}|}.$$

Positive values of this measure, for example, indicate that the algorithm is overestimating the velocity magnitude.

In situations where we have estimated velocity field covariances, $\boldsymbol{\Lambda}_f$, as well as means, $\boldsymbol{\mu}_f$, we can check that the covariance information adequately describes the errors in the flow estimates. The appropriate technique here is to normalize each of the errors according to the covariance information:

$$E_{\text{normalized}} = \sqrt{(\boldsymbol{f}_{\text{actual}} - \boldsymbol{f}_{\text{est}})^T \boldsymbol{\Lambda}_f^{-1} (\boldsymbol{f}_{\text{actual}} - \boldsymbol{f}_{\text{est}})}.$$

If the flow field errors are exactly modeled by the additive Gaussian noise model, then a histogram of the values of the $E_{\text{normalized}}$ values should be distributed as a two-dimensional univariate Gaussian integrated over its angular coordinate:

$$h(x) \propto x e^{-x^2/2}, \qquad x > 0.$$

That is, a χ statistic.

14.6.2 Synthetic sequences

We generated a series of very simple synthetic test sequences to study the error behavior of the algorithm. These stimuli involve only translation of the image patterns, and therefore fully obey (modulo intensity quantization noise) the total derivative (14.1) for optical flow. Furthermore, since the entire image translates with a single velocity, the combination of information in a neighborhood is fully justified. Thus, these examples are primarily a test of the filters used to measure the derivatives, and the prior probability constraint used to determine a solution when there is an aperture or blank-wall problem. For this section, we used only the single-scale basic gradient algorithm, and we set the noise parameters as $\lambda_1 = 0, \lambda_2 = 1$, and $\lambda_p = 1e^{-5}$.

To illustrate the spatial behavior of the algorithm, we estimated the velocity field of an impulse image moving at one pixel per frame. The flow

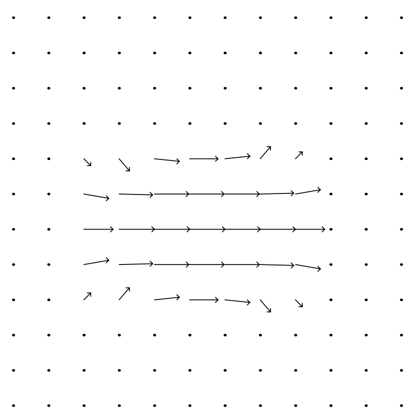


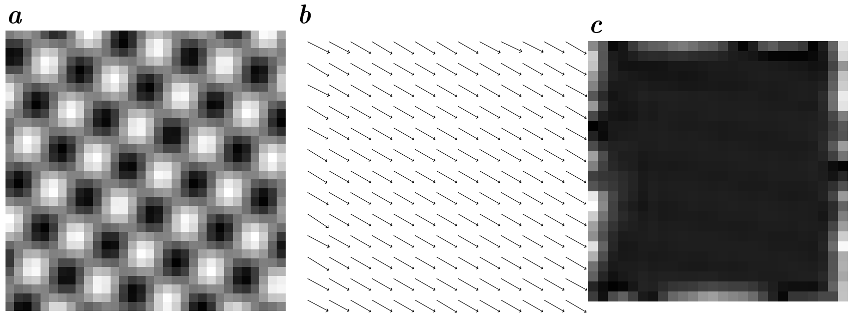
Figure 14.6: *Velocity field estimated for a spatial impulse moving rightward at one pixel/frame. Dots correspond to zero velocity vectors.*

field is shown in Fig. 14.6. The estimated velocity is correct in the center of the image (at the location of the impulse). The finite size of the derivative filters (five-tap kernels were used for this example) and the blurring of the energies leads to the situation shown, in which the impulse drags part of the background along. The velocity surrounding the impulse is consistent with the image intensities: since the image is zero everywhere except at the impulse, the motion is completely indeterminate.

Next, we examined a sinusoidal plaid pattern, taken from Barron et. al. [5]. Two sinusoidal gratings with spatial frequency 6 pixels/cycle are additively combined. Their normal orientations are at 54° and −27° with speeds of 1.63 pixels/frame and 1.02 pixels/frame, respectively. The flow is computed using the multi-scale algorithm. We built a one-level Gaussian pyramid on each frame, using the following five-tap kernel: $[0.0625, 0.25, 0.375, 0.25, 0.0625]$. Derivative filters used are the five-tap first derivative kernels given in Table 14.1. The covariance parameters were set as follows: $\lambda_1 = 0, \lambda_2 = 1, \lambda_p = 1e^{-5}$, and $\lambda_0 = 0.15$. One frame of the sequence, the estimated flow, and the error magnitude image are shown in Fig. 14.7.

Also shown is a table of error statistics. The errors compare quite favorably with the mean angular errors reported by Barron et. al. [5]. Our mean angular error is an order of magnitude less than all the methods examined, except for that of Fleet and Jepson for which the value was 0.03°. But Barron et. al. point out that the Fleet and Jepson results are computed with filters that are tuned for the sinusoids in the stimulus and that for a stimulus composed of different sinusoids, the algorithm would exhibit biases.

We also note also that our algorithm is *significantly* more efficient than most of the algorithms in [5]. For example, the Fleet and Jepson algorithm



Mean E_{mag2}	$1.597 \cdot 10^{-4}$
Mean E_{angular}	0.2746
St. dev. E_{angular}	0.300
Mean E_{bias}	0.004378

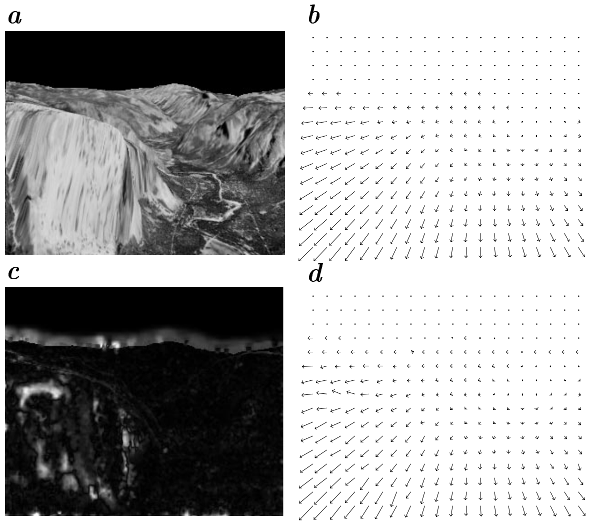
Figure 14.7: *Velocity field estimates for sinusoidal plaid sequence:* ***a*** *example frame from the sequence,* ***b*** *estimated velocity field, and* ***c*** *error magnitude image. Note that the error is concentrated at the boundaries, where derivative measurement is difficult. Error statistics for this computation are given in the table (see text for definitions).*

is implemented with a set of 46 kernels. These are implemented separably as 75 convolutions with one-dimensional 21-tap kernels. Our solution requires 8 convolutions (with one-dimensional kernels) for the first derivative measurements with kernels that are only three or five taps in length.

Moving one step closer to real imagery, we estimated velocity fields for a "texture-mapped" fly-through sequence of the Yosemite valley [5]. Starting with an aerial photograph and a range (height) map of the Yosemite valley, a sequence of images was rendered for a series of camera positions. Photometric effects are *not* included in this rendering process: the image pixel values are interpolated directly from the intensities of the original photograph. Thus, the sequence contains all of the standard problem sources except for lighting effects (i. e., singular regions, temporal aliasing, and multiple motions at occlusions). Note that we have the camera motion and the depth of each point in the image, we can compute the *actual* image motion fields.

Again, we computed velocity fields using the multi-scale solution. This time, we build a three-level Gaussian pyramid. Parameter settings were as follows: $\lambda_1 = 2e^{-5}, \lambda_2 = 0.004, \lambda_p = 0.5$, and $\lambda_0 = 0.15$. The results are illustrated in Fig. 14.8. We show a frame from the original sequence, the correct velocity field, the estimated velocity field, and the error magnitude

[5]This sequence was generated by Lyn Quam at SRI International.



Mean E_{mag2}	0.031747
Mean E_{angular}	3.8105
St. dev. E_{angular}	7.09378
Mean E_{bias}	-0.01083

Figure 14.8: *Results of applying the algorithm to the synthetic "Yosemite" sequence:* ***a*** *example frame from the original sequence,* ***b*** *correct flow field,* ***c*** *estimated velocity field, and* ***d*** *error magnitude image. Note that errors are concentrated near occlusion boundaries.*

image. Also given is a table of statistics. The statistics do not include the points closer than 10 pixels to the border.

The results are quite accurate, with most errors occurring (as expected) at occlusion boundaries, and at the borders of the image (which may be viewed as a type of occlusion boundary). But qualitative comparisons with the results of the Heeger or Fleet and Jepson algorithm indicate that the errors near these boundaries are contained within smaller regions near the boundaries, since the support of the filters is much smaller.

Furthermore, the error statistics compare quite favorably to those reported in [5]. In particular, the best result reported is that of Lucas and Kanade, with a mean angular error of 3.55° and standard deviation of 7.09°. This is almost identical to our result, but the flow vector density is only 8.8%. The best result reported at 100% is that of Uras, which had a mean angular error of 10.44°, and standard deviation of 15.00°. The values given in Fig. 14.8 are significantly lower.

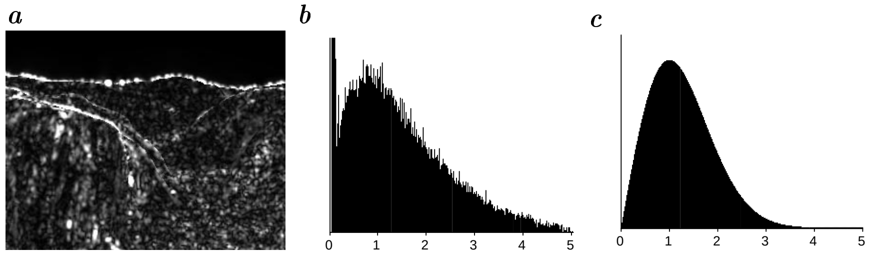


Figure 14.9: ***a*** *Image of $E_{\text{normalized}}$ for the Yosemite sequence velocity estimates.* ***b*** *Histogram of $E_{\text{normalized}}$ values.* ***c*** *Expected distribution $h(x)$ for this histogram (see text).*

To analyze the appropriateness of the noise model, we computed a $E_{\text{normalized}}$ at each point. We show this image in Fig. 14.9, along with the histogram of values. If the flow field errors were exactly modeled by the simple additive gaussian noise terms, then this histogram would be in the form of the χ statistic distribution (also plotted in Fig. 14.9). Qualitatively, the error histogram is seen to match, suggesting that the noise model is not unreasonable.

14.7 Conclusion

In this chapter, we described a Bayesian estimator for motion fields. We combine differential constraints over local spatial regions (thereby assuming a smooth motion field), and we assume a Gaussian prior probability density in which slower speeds are more likely. The output of the algorithm is a Gaussian distribution over the space of image velocities, at each position in the image. The mean of the distribution is a gain-controlled modification of the basic differential optical flow solution. The covariance matrix captures directional uncertainties, allowing proper combination of the output with other sources of information.

We developed a coarse-to-fine estimation algorithm for handling the problem of large displacements (temporal aliasing). Here we are able to take advantage of the uncertainty information provided by the covariance estimates, propagating this information using a Kalman filter over scale. Propagation of motion fields (and their covariances) over time has been described in [43].

We discussed the details of algorithm implementation, and showed several diagnostic examples, designed to demonstrate the various strengths and weaknesses of the algorithm we have developed. We generated accurate velocity field estimates for a number of synthetic sequences.

14.8 References

[1] B K P Horn. *Robot Vision.* MIT Press, Cambridge, MA, 1986.

[2] A Verri and T Poggio. Motion field and optical flow: Qualitative properties. *IEEE Pat. Anal. Mach. Intell.*, pages 490–498, 1989.

[3] E H Adelson and J R Bergen. The extraction of spatiotemporal energy in human and machine vision. In *Proc. IEEE Workshop on Visual Motion*, pages 151–156, 1986.

[4] E P Simoncelli. *Distributed Analysis and Representation of Visual Motion.* PhD thesis, Massachusetts Institute of Technology, Department of Electrical Engineering and Computer Science, Cambridge, MA, January 1993. Also available as MIT Media Laboratory Vision and Modeling Technical Report #209.

[5] J L Barron, D J Fleet, and S S Beauchemin. Performance of optical flow techniques. *Intl. J. Comp. Vis.*, 12(1):43–77, 1994.

[6] E P Simoncelli, E H Adelson, and D J Heeger. Probability distributions of optical flow. In *Proc Conf on Computer Vision and Pattern Recognition*, pages 310–315, Mauii, Hawaii, June 1991. IEEE Computer Society.

[7] E P Simoncelli. Coarse-to-fine estimation of visual motion. In *Proc Eighth Workshop on Image and Multidimensional Signal Processing*, pages 128–129, Cannes, France, September 1993. IEEE Sig Proc Society.

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

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

[10] 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.

[11] H H Nagel. Displacement vectors derived from second order intensity variations in image sequences. *Computer Vision, Pattern Recognition, and Image Processing*, 21:85–117, 1983.

[12] S Uras, F Girosi, A Verri, and V Torre. A computational approach to motion perception. *Biol. Cybern.*, 60:79–87, 1988.

[13] R Eagleson. Measurement of the 2D affine Lie group parameters for visual motion analysis. *J. Spatial Vision*, 1992.

[14] J Mendelsohn, E Simoncelli, and R Bajcsy. Discrete-time rigidity-constrained optical flow. In *Int'l Conf Computer Analysis of Images and Patterns*, pages 255–262, Kiel, Germany, September 1997. Springer-Verlag.

[15] J Weber and J Malik. Robust computation of optic flow in a multiscale differential framework. *Intl. J. Comp. Vis.*, 14(1):67–81, 1996.

[16] M J Black and P P Anandan. The robust estimation of multiple motions: Parametric and piecewise-smooth flow fields. *Computer Vision and Image Understanding*, 63(1):75–104, 1996.

[17] T J Darrell and E Simoncelli. Separation of transparent motion into layers using velocity-tuned mechanisms. In *European Conf on Computer Vision.*

Springer, 1994.

[18] J Y A Wang and E H Adelson. Representing moving images with layers. *IEEE Trans. Im. Proc.*, 3(5):625–638, 1994.

[19] S Hsu, P P Anandan, and S Peleg. Accurate computation of optical flow by using layered motion representations. In *Proc. ICPR*, pages 743–746, Jerusalem, Israel, October 1994.

[20] S Ayer and H S Sawhney. Layered representation of motion video using robust maximum-likelihood estimation of mixture models and mdl encoding. In *Fifth Int'l Conf. Comp. Vis.*, pages 777–784, Cambridge, MA, June 1995.

[21] S X Ju, M J Black, and Y Yacoob. Skin and bones: Multi-layer, locally affine, optical flow and regularization with transparency. In *IEEE Conf. CVPR*, pages 307–314, San Francisco, CA, June 1996.

[22] J. J. Koenderink and A. J. van Doorn. Representation of local geometry in the visual system. *Biol. Cybern.*, 55:367–375, 1987.

[23] D J Heeger and E P Simoncelli. Model of visual motion sensing. In L Harris and M Jenkin, editors, *Spatial Vision in Humans and Robots*, pages 367–392. Cambridge University Press, 1992.

[24] R A Young and R M Lesperance. A physiological model of motion analysis for machine vision. In *Proc SPIE: Human Vision, Visual Processing, and Digital Display IV*, volume 1913, pages 48–123, 1993.

[25] E P Simoncelli and D J Heeger. A model of neuronal responses in visual area MT. *Vision Research*, 38(5):743–761, 1998.

[26] Y Weiss and E H Adelson. Slow and smooth: A Bayesian theory for the combination of local motion signals in human vision. Technical Report 1624, AI Laboratory, Massachusetts Institute of Technology, Cambridge, MA, 1998.

[27] 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. ACM, Baltimore, 1983.

[28] Lyn Quam. Hierarchical warp stereo. In *Proceedings of the DARPA Image Understanding Workshop*, pages 149–155, September 1984.

[29] W Enkelmann and H Nagel. Investigation of multigrid algorithms for estimation of optical flow fields in image sequences. *Computer Vision, Graphics, and Image Processing*, 43:150–177, 1988.

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

[31] R Battiti, E Amaldi, and C Koch. Computing optical flow across multiple scales: an adaptive coarse-to-fine strategy. *Intl. J. Comp. Vis.*, 6(2):133–145, 1991.

[32] Richard Szeliski. Bayesian modeling of uncertainty in low-level vision. *Intl. J. Comp. Vis.*, 5(3):271–301, December 1990.

[33] K C Chou, A S Willsky, A Benveniste, and M Basseville. Recursive and iterative estimation algorithms for multi-resolution stochastic processes. Technical Report LIDS-P-1857, MIT Laboratory for Information and Decision Sciences, March 1989.

[34] A Gelb, editor. *Applied Optimal Estimation.* MIT Press, 1989.

[35] Per-Erik Danielsson. Rotation-invariant linear operators with directional response. In *5th Int'l Conf. Patt. Rec.*, Miami, December 1980.

[36] E P Simoncelli. Design of multi-dimensional derivative filters. In *First Int'l Conf on Image Proc*, volume I, pages 790–793, Austin, Texas, November 1994. IEEE Sig Proc Society.

[37] H Farid and E Simoncelli. Optimally rotation-equivariant directional derivative kernels. In *Int'l Conf Computer Analysis of Images and Patterns*, pages 207–214, Kiel, Germany, September 1997. Springer-Verlag.

[38] Alan V. Oppenheim and Ronald W. Schafer. *Digital Signal Processing*. Prentice-Hall, Inc., Englewood Cliffs, 1975.

[39] D J Fleet and K Langley. Recursive filters for optical flow. *IEEE Pat. Anal. Mach. Intell.*, 17(1):61–67, 1995.

[40] B. Carlsson, A Ahlen, and M. Sternad. Optimal differentiators based on stochastic signal models. *IEEE Trans. Signal Proc.*, 39(2), February 1991.

[41] P J Burt and E H Adelson. The Laplacian pyramid as a compact image code. *IEEE Trans. Comm.*, COM-31(4):532–540, April 1983.

[42] D Fleet and A Jepson. Computation of component image velocity from local phase information. *Intl. J. Comp. Vis.*, 5(1):77–104, 1990.

[43] A Singh. Incremental estimation of image-flow using a kalman filter. In *Proceedings IEEE Workshop on Visual Motion*, Princeton, October 1991.