Optical Flow Estimation

Goal: Introduction to image motion and 2D optical flow estimation.

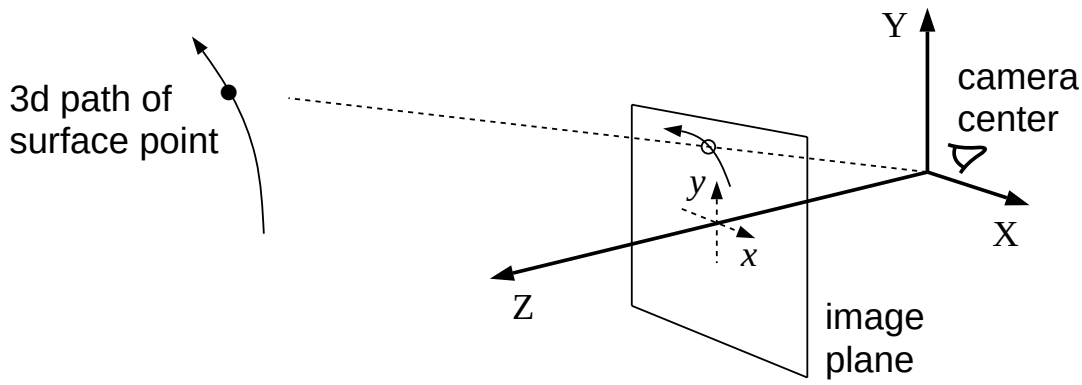
Motivation:

- Motion is a rich source of information about the world:
  - segmentation
  - surface structure from parallax
  - self-motion
  - recognition
  - understanding behavior
  - understanding scene dynamics
- Other correspondence / registration problems:
  - stereo disparity (short and wide baseline)
  - computer-assisted surgery (esp. multi-modal registration)
  - multiview alignment for mosaicing or stop-frame animation

Readings: Fleet and Weiss (2005)

Matlab Tutorials: intromotionTutorial.m, motionTutorial.m

Introduction to Image Motion



A 3D point $\vec{\mathbf{X}}$ follows a space-time path $\vec{\mathbf{X}}(t)$. Its velocity is $\vec{\mathbf{V}}$ is

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

Perspective projection (for nodal length f) of the 3D path onto the image plane produces a 2D path,

$$\vec{\mathbf{x}}(t) \;=\; (x(t),\, y(t)) \;=\; \left(\frac{fX(t)}{Z(t)}, \frac{fY(t)}{Z(t)} \right) ,$$

the instantaneous 2D velocity of which is

$$\begin{aligned} \vec{\mathbf{u}} \;&=\; \left(\frac{dx(t)}{dt}, \frac{dy(t)}{dt} \right)^T \\ &=\; \frac{f}{Z(t)} \left(\frac{dX(t)}{dt}, \frac{dY(t)}{dt} \right)^T \;-\; \frac{f}{Z^2(t)} \frac{dZ(t)}{dt} (X(t),\, Y(t))^T . \end{aligned}$$

Definition: **2D Motion Field** – 2D velocities for all visible points.

Optical Flow Field – Estimate of the 2D motion field.

Optical Flow

Two Key Problems:

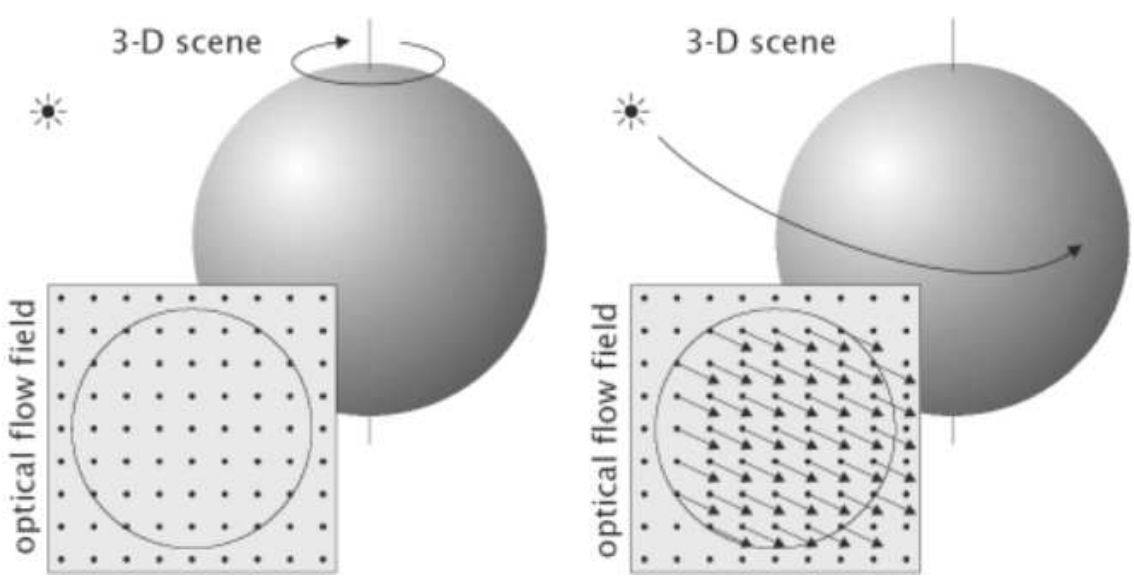
1. Determine what image property to track.
2. Determine how to track it

Brightness constancy: Let's track points of constant brightness, assuming that surface radiance is constant over time:

$$I(x, y, t+1) \;=\; I(x - u_1, y - u_2, t)\;.$$

Brightness constancy is often assumed by researchers, and often violated by Mother Nature; so the optical flow field is sometimes a poor approximation to the 2D motion field.

> E.g., a rotating Lambertian sphere with a static light source produces a static image, but a stationary sphere with a moving light source produces drifting reflection (figure from Jahne et al, 1999).



Gradient-Based Motion Estimation

How do we find the motion of a 1D greylevel function $f(x)$? E.g., let's say that $f(x)$ is displaced by distance d from time 1 to time 2:

$$f_2(x) = f_1(x - d) \ .$$

We can express the shifted signal as a Taylor expansion of f_1 about x:

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

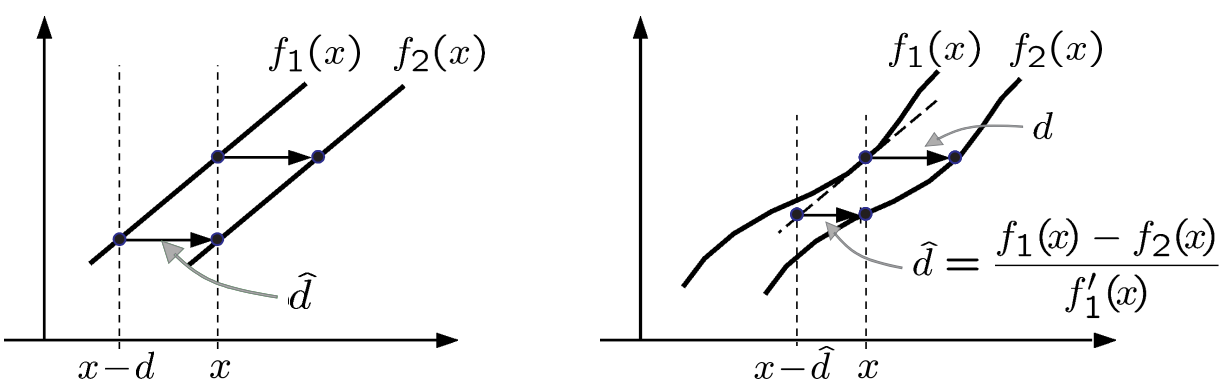
in which case the difference between the two signals is given by

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

Then, a first-order approximation to the displacement is

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

For linear signals the first-order estimate is exact.



For nonlinear signals, the accuracy of the approximation depends on the displacement magnitude and the higher-order signal structure.

Gradient Constraint Equation

In two spatial dimensions we express the brightness constancy constraint as

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

As above, we substitute a first-order approximation,

$$\begin{aligned} & f(x+u_1, y+u_2,\ t+1) \approx \\ & \qquad f(x,y,t) + u_1 f_x(x,y,t) + u_2 f_y(x,y,t) + f_t(x,y,t) \end{aligned} \tag{2}$$

into the left-hand side of (1), and thereby obtain the *gradient constraint equation*:

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

Written in vector form, with $\vec{\nabla} f \equiv (f_x,\ f_y)^T$:

$$\vec{\mathbf{u}}^T \vec{\nabla} f(x,y,t) + f_t(x,y,t) \ = \ 0\ .$$

When the duration between frames is large, it is sometimes more appropriate to use only spatial derivatives in the Taylor series approximation in (2). Then one obtains a different approximation

$$\vec{\mathbf{u}}^T \vec{\nabla} f(x,y,t) + \Delta f(x,y,t) \ = \ 0$$

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

Brightness Conservation

One can also derive the gradient constraint equation directly from brightness conservation.

Let $(x(t),\ y(t))$ denote a space-time path along which the image intensity remains is constant; i.e., the time-varying image f satisfies

$$f(x(t), y(t), t) = c$$

Taking the total derivative of both sides gives

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

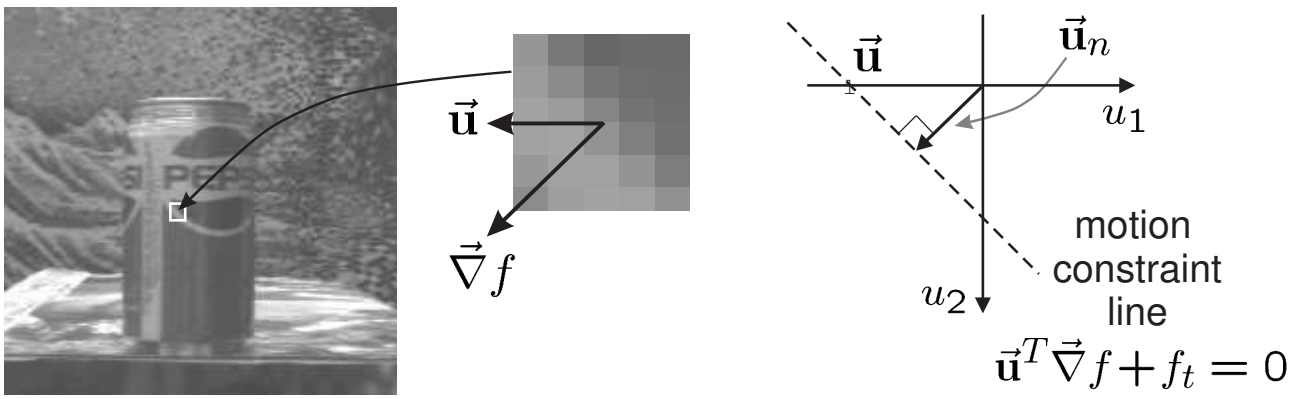
The total derivative is given in terms of its partial derivatives

$$\begin{aligned}
\frac{d}{d\,t}\, f(x(t), y(t), t) &= \frac{\partial f}{\partial x}\frac{dx}{dt} \;+\; \frac{\partial f}{\partial y}\frac{dy}{dt} \;+\; \frac{\partial f}{\partial t}\frac{dt}{dt} \\
&= f_x\, u_1 \;+\; f_y\, u_2 \;+\; f_t \\
&= \vec{\mathbf{u}}^T \vec{\nabla} f + f_t \\
&= 0
\end{aligned}$$

This derivation assumes that there is no aliasing, so that one can in principle reconstruct the continuous underlying signal and its derivatives in space and time. There are many situations in which this is a reasonable assumption. But with common video cameras temporal aliasing is often a problem with many video sequences, as we discuss later in these notes.

Normal Velocity

The gradient constraint provides one constraint in two unknowns. It defines a line in velocity space:



The gradient constrains the velocity in the direction normal to the local image orientation, but does not constrain the tangential velocity. That is, it uniquely determines only the normal velocity:

$$\vec{\mathbf{u}}_n = \frac{-f_t}{||\vec{\nabla} f||} \frac{\vec{\nabla} f}{||\vec{\nabla} f||}$$

When the gradient magnitude is zero, we get no constraint!

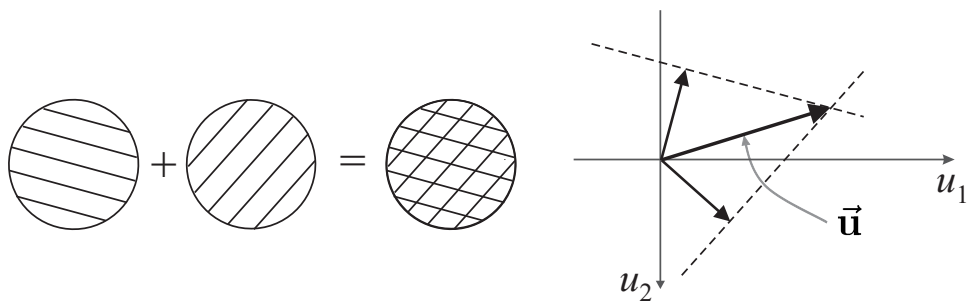
In any case, further constraints are required to estimate both elements of the 2D velocity $\vec{\mathbf{u}} = (u_1,\ u_2)^T$.

Area-Based Regression

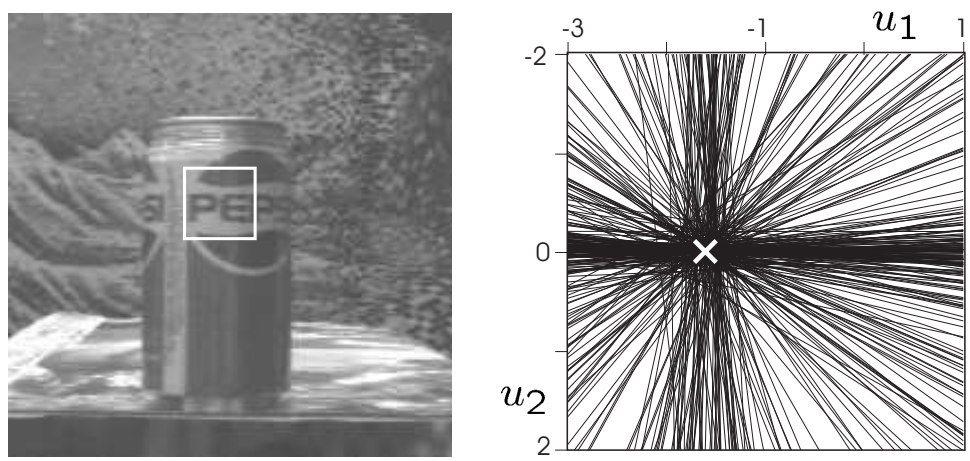
Smoothness Assumption: 2D motion is smooth in the local image neighborhood of the point at which we are estimating optical flow.

E.g., Let's say we have gradient constraints at two adjacent pixels, $\vec{\mathbf{x}}_1$ and $\vec{\mathbf{x}}_2$, which have the same velocity, $\vec{\mathbf{u}} = (u_1,\ u_2)^T$:

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



More generally, we can use many constraints from within a local region about the point at which we wish to estimate the optical flow. Here is an example from the Pepsi sequence:



Area-Based Regression (Linear LS)

We will not satisfy all constraints exactly. Rather we seek the velocity that minimizes the squared error in each constraint (called the least-squares (LS) velocity estimate):

$$E(u_1, u_2) = \sum_{x,y} g(x,y) \left[u_1 f_x(x,y,t) + u_2 f_y(x,y,t) + f_t(x,y,t)\right]^2 \quad (3)$$

where $g(x,y)$ is a low-pass window that helps give more weight to constraints at the center of the region.

Solution: Differentiate E with respect to (u_1, u_2) and set to zero:

$$\frac{\partial E(u_1,u_2)}{\partial u_1} = 2\sum_{x,y} g(x,y) \left[u_1 {f_x}^2 + u_2 f_x f_y + f_x f_t\right] = 0$$

$$\frac{\partial E(u_1,u_2)}{\partial u_2} = 2\sum_{x,y} g(x,y) \left[u_2 {f_y}^2 + u_1 f_x f_y + f_y f_t\right] = 0$$

The constraints thus yield two linear equations for u_1 and u_2.

In matrix notation, these *normal equations* and their solution are

$$\mathbf{M}\,\vec{\mathbf{u}} + \vec{\mathbf{b}} = \vec{0} \quad , \qquad \hat{\mathbf{u}} = -\mathbf{M}^{-1}\vec{\mathbf{b}}$$

(assuming $\mathbf{M}^{-1}$ exists), where

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

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

Area-Based Regression (Implementation)

Since we want the optical flow at each pixel, we can compute the components of the normal equations with a set of image operators. This is much prefered to looping over image pixels.

1. First, compute 3 image gradient images at time t, corresponding to the gradient measurements:

$$f_x(\vec{\mathbf{x}}) \ , \quad f_y(\vec{\mathbf{x}}) \ , \quad f_t(\vec{\mathbf{x}})$$

2. Point-wise, we compute the quadratic functions of the derivative images. This produces five images equal to:

$$f_x^2(\vec{\mathbf{x}}) \ , \quad f_y^2(\vec{\mathbf{x}}) \ , \quad f_x(\vec{\mathbf{x}}) f_y(\vec{\mathbf{x}}) \ , \quad f_t(\vec{\mathbf{x}}) f_y(\vec{\mathbf{x}}) \ , \quad f_t(\vec{\mathbf{x}}) f_x(\vec{\mathbf{x}})$$

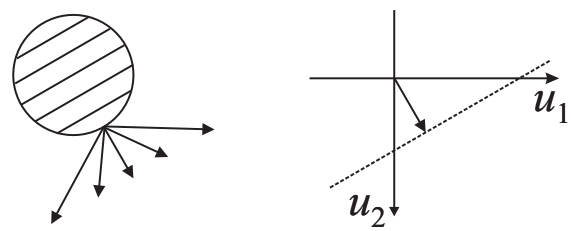
3. Blurring the quadratic images corresponds to the accumulating of local constraints under the spatial (or spatiotemporal) support window g above. This produces five images, each of which contains an element of the normal equations at a specific image locations:

$$g(\vec{\mathbf{x}}) * f_x^2(\vec{\mathbf{x}}) \ , \quad g(\vec{\mathbf{x}}) * f_y^2(\vec{\mathbf{x}}) \ ,$$
$$g(\vec{\mathbf{x}}) * [f_x(\vec{\mathbf{x}})\, f_y(\vec{\mathbf{x}})] \ , \quad g(\vec{\mathbf{x}}) * [f_t(\vec{\mathbf{x}})\, f_y(\vec{\mathbf{x}})] \ , \quad g(\vec{\mathbf{x}}) * [f_t(\vec{\mathbf{x}})\, f_x(\vec{\mathbf{x}})]$$

4. Compute the two images containing the components of optical flow at each pixel. This is given in closed form since the inverse of the normal matrix (i.e., $\mathbf{M}$ above) is easily expressed in closed form.

Aperture Problem

Even with the collection of constraints from within a region, the estimation will be undetermined when the matrix $\mathbf{M}$ is singular:



When all image gradients are parallel, then the normal matrix for the least-squares solution becomes singular (rank 1). E.g., for gradients $m(x,y)\vec{\mathbf{n}}$, where $m(x,y)$ is the gradient magnitude at pixel (x,y)

$$\mathbf{M} \;=\; \left(\sum_{x,y} g(x,y)\, m^2(x,y) \right) \vec{\mathbf{n}}\, \vec{\mathbf{n}}^T$$

Aperture Problem: For sufficiently small spatial apertures the normal matrix, $\mathbf{M}$, will be singular (rank deficient).

Generalized Aperture Problem: For small apertures M becomes singular, but for sufficiently large apertures the 2D motion field may deviate significantly from the assumed motion model.

Heuristic Diagnostics: Use singular values, $s_1 \geq s_2 \geq s_3$, of $\sum_{x,y} g\, \vec{\mathbf{h}}\, \vec{\mathbf{h}}^T$, for $\vec{\mathbf{h}} = (f_x, f_y, f_t)^T$, to assess the rank of the local image structure:

- if s_2 is "too small" then only normal velocity is available
- if s_3 is "large" then the gradient constraints do not lie in a plane so a single velocity will not fit the data well.

Such "bounds" will depend on the LS region size and the noise.

Iterative Estimation

The LS estimate minimized an approximate objective function (since we linearized the image to obtain the gradient constraints):

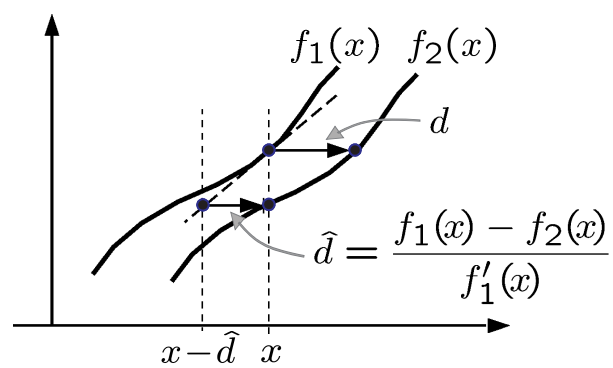
$$\hat{E}(u_1, u_2) \;=\; \sum_{x,y} g(x,y)\;[\,u_1 f_x(x,y,t) + u_2 f_y(x,y,t) + f_t(x,y,t)\,]^2$$

Nevertheless, the real objective function of interest is

$$E(u_1, u_2) \;=\; \sum_{x,y} g(x,y)\;[f_2(x,y,t{+}1) - f_1(x{-}u_1, y{-}u_2, t)\,]^2$$

The estimation errors caused by this approximation are second-order in the magnitude of the displacement:

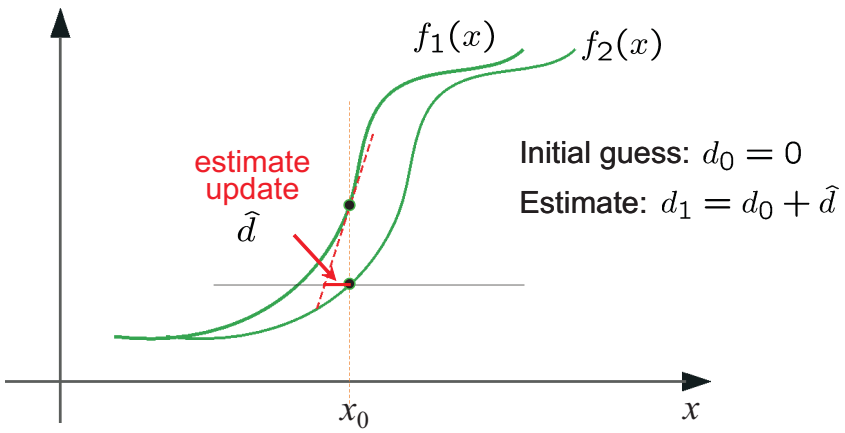
$$|\hat{d} - d| \;\leq\; \frac{d^2\,|f_1''(x)|}{2\,|f_1'(x)|} + O(d^3)$$

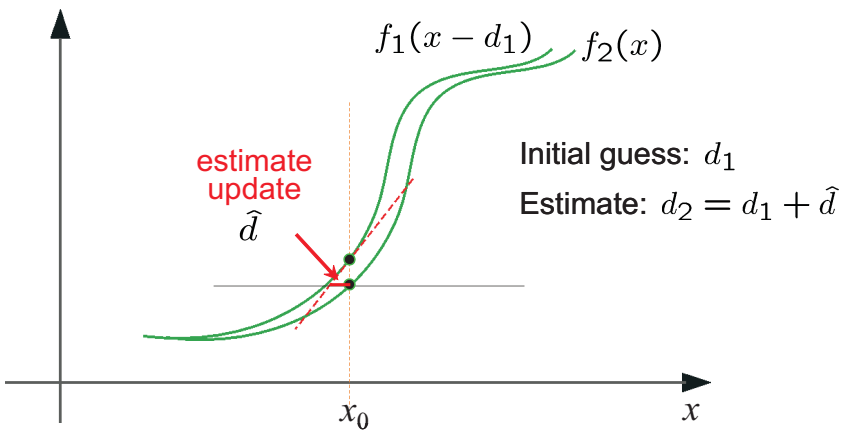


So we might converge toward a solution, with a sufficiently good initial guess, by iterating the estimation, decreasing the displacement error in each iteration. That is, repeat the steps:

1. estimate the shift given the two images,
2. warp one image toward the other

Iterative Estimation (cont)





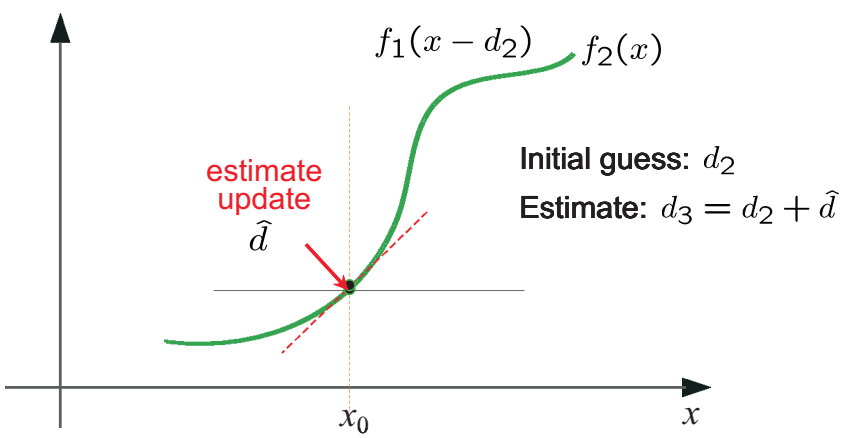
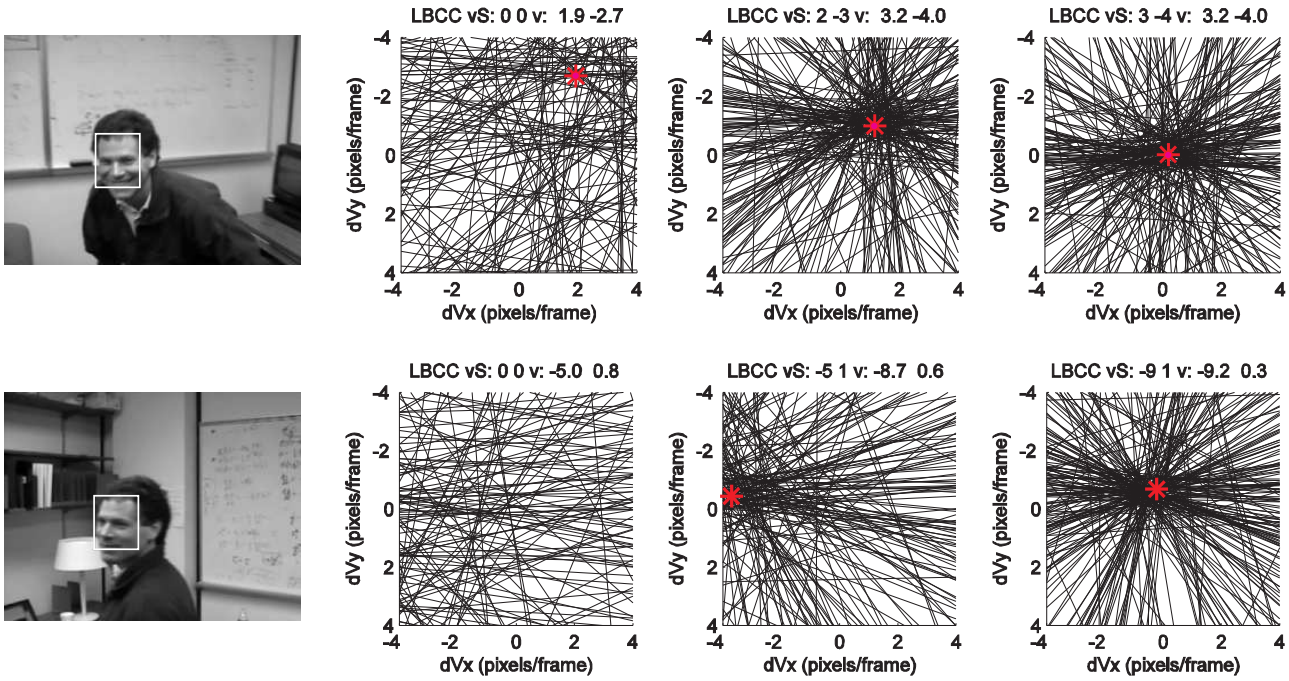


Image Warps

Example: 3 iterations of refinement (from the motion tutorial). Here, **vS** and **v** denote the preshift velocity and the LS estimate. Figure axes represent incremental horizontal and vertical velocities (given **vS**).

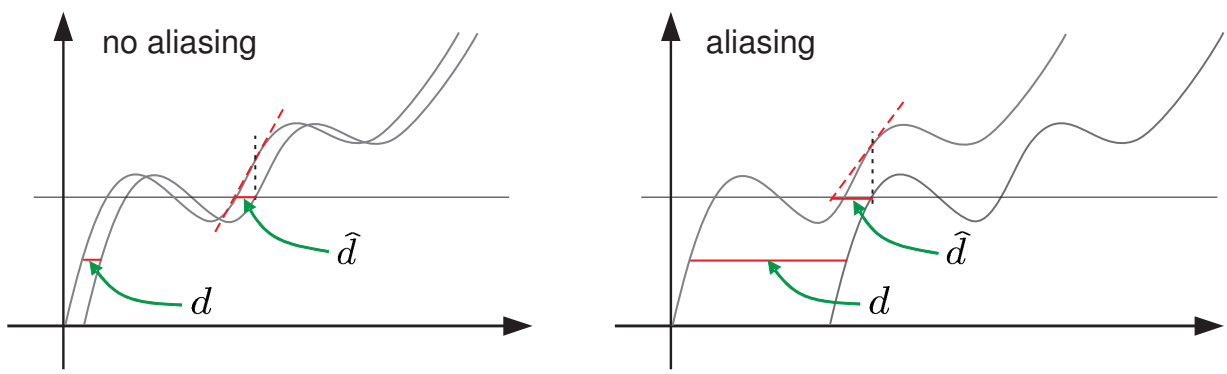


Some practical issues:

- initially, if the displacement is large, many of the constraints might be very noisy
- warping is time-consuming and introduces interpolation noise, so stick to integer warps where possible.
- warp one image, take derivatives of the other so you don't need to re-compute the gradient after each iteration.
- often useful to low-pass filter the images before motion estimation (for better derivative estimation, and somewhat better linear approximations to image intensity)

Aliasing

When a displacement between two frames is too large (i.e., when a sinusoid is displaced a half a wavelength or more), the input is aliased. This is caused by fast shutter speeds in most cameras.



Aliasing often results in convergence to the wrong 2D optical flow.

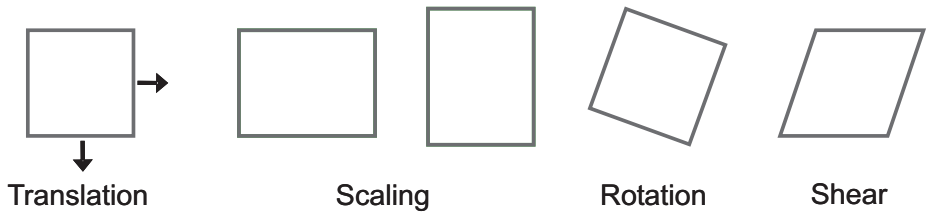
Possible Solutions:

1. prediction from previous estimates to *pre-warp* the next image
2. use feature correspondence to establish rough initial match
3. **coarse-to-fine estimation:** Progressive flow estimation from coarse to fine levels within Gaussian pyramids of the two images.
   - start at coarsest level, $k = L$, warp images with initial guess, then iterate warping & LS estimation until convergence to $\vec{\mathbf{u}}_L$.
   - warp levels $k = L - 1$ using $-\vec{\mathbf{u}}_L$, then apply iterative estimation until convergence $\vec{\mathbf{u}}_{L-1}$.

     ...
   - warp levels $k = 0$ (i.e., the original images) using $\vec{\mathbf{u}}_1$, then apply iterative estimation until convergence.

Higher-Order Motion Models

Constant flow model within an image neighborhood is often a poor model, especially when the regions become larger. Affine models often provide a more suitable model of local image deformation.



Affine flow for an image region centered at location $\vec{\mathbf{x}}_0$ is given by

$$\vec{\mathbf{u}}(\vec{\mathbf{x}}) \;=\; \begin{pmatrix} a_1 & a_2 \\ a_3 & a_4 \end{pmatrix} (\vec{\mathbf{x}} - \vec{\mathbf{x}}_0) + \begin{pmatrix} a_5 \\ a_6 \end{pmatrix} \;=\; \mathbf{A}(\vec{\mathbf{x}}; \vec{\mathbf{x}}_0)\, \vec{\mathbf{a}}$$

where $\vec{\mathbf{a}} = (a_1,\, a_2,\, \cdots\,,\, a_6)^T$ and

$$\mathbf{A}(\vec{\mathbf{x}}; \vec{\mathbf{x}}_0) \;=\; \begin{bmatrix} x-x_0 & y-y_0 & 0 & 0 & 1 & 0 \\ 0 & 0 & x-x_0 & y-y_0 & 0 & 1 \end{bmatrix}$$

Then the gradient constraint becomes:

$$\begin{aligned} 0 \;&=\; \vec{\mathbf{u}}(x,y)^T \vec{\nabla} f(x,y,t) + f_t(x,y,t) \\ &=\; \vec{\mathbf{a}}^T \mathbf{A}(x,y)^T \vec{\nabla} f(x,y,t) + f_t(x,y,t)\ , \end{aligned}$$

so, as above, the weighted least-squares solution for $\vec{\mathbf{a}}$ is given by:

$$\hat{\mathbf{a}} \;=\; \mathbf{M}^{-1} \vec{\mathbf{b}}$$

where, for weighting with the spatial window g,

$$\mathbf{M} = \sum_{x,y} g(\vec{\mathbf{x}})\, \mathbf{A}^T\, \vec{\nabla} f\, \vec{\nabla} f^T \mathbf{A}\ , \qquad \vec{\mathbf{b}} \;=\; -\sum_{x,y} g(\vec{\mathbf{x}})\, \mathbf{A}^T\, \vec{\nabla} f\; f_t$$

Similarity Transforms

There are many classes of useful motion models. Simple translation and affine deformations are the commonly used motion models.

Another common motion model is a special case of affine deformation called a similarity deformation. It comprises a uniform scale change, along with a rotation and translation:

$$\vec{\mathbf{u}}(\vec{\mathbf{x}}) \;=\; \alpha \begin{pmatrix} \cos\theta & -\sin\theta \\ \sin\theta & \cos\theta \end{pmatrix} (\vec{\mathbf{x}} - \vec{\mathbf{x}}_0) + \begin{pmatrix} d_1 \\ d_2 \end{pmatrix}$$

We can also express this transform as a linear function of a somewhat different set of motion model parameters. In particular, in matrix form we can write a similarity transform as

$$\vec{\mathbf{u}}(x, y) \;=\; A(x, y)\, \vec{\mathbf{a}}$$

where $\vec{\mathbf{a}} \;=\; (\alpha\, \cos\theta,\, \alpha\, \sin\theta,\, d_1,\, d_2)^T$, and

$$A \;=\; \begin{pmatrix} x - x_0 & -y + y_0 & 1 & 0 \\ y - y_0 & x - x_0 & 0 & 1 \end{pmatrix}$$

With this form of the similarity transform, we can formulate a LS estimator for similarity transforms in the same way we did for the affine motion model.

There are other classes of motion model that are also very important. One such class is the 2D homography (we'll discuss this later).

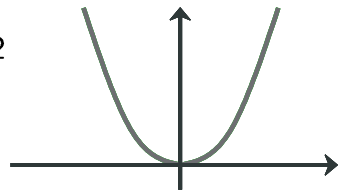
Robust Motion Estimation

Sources of heavy-tailed noise (outliers): specularities & highlights, jpeg artifacts, interlacing, motion blur, large displacements, and multiple motions (occlusion boundaries, transparency, etc.)

Problem: Least-squares estimators are extremely sensitive to outliers

error penalty function

$$\rho(\epsilon) = \epsilon^2$$



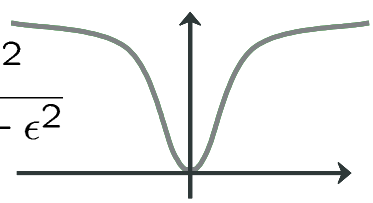
influence function

$$\psi(\epsilon) = \frac{\partial \rho(\epsilon)}{\partial \epsilon} = 2\,\epsilon$$

Solution: Redescending error functions (e.g., Geman-McClure) help to reduce influence of outliers:

error penalty function

$$\rho(\epsilon;\, s) = \frac{\epsilon^2}{s + \epsilon^2}$$



influence function

$$\psi(\epsilon;\, s) = \frac{2\,\epsilon\, s}{(s + \epsilon^2)^2}$$

Previous objective function:

$$E(\vec{\mathbf{u}}) = \sum_{x,y} g(\vec{\mathbf{x}})\, [f(\vec{\mathbf{x}}, t{+}1) - f(\vec{\mathbf{x}}{-}\vec{\mathbf{u}},\, t)]^2$$

Robust objective function:

$$E(\vec{\mathbf{u}}) = \sum_{x,y} g(\vec{\mathbf{x}})\, \rho(f(\vec{\mathbf{x}}, t{+}1) - f(\vec{\mathbf{x}}{-}\vec{\mathbf{u}},\, t))$$

Probabilistic Motion Estimation

Probabilistic formulations allow us to address issues of noisy measurements, confidence measures, and prior beliefs about the motion.

Formulation: Measurements of image derivatives, especially f_t, are noisy. Let's assume that spatial derivatives are accurate, but temporal derivative measurements have additive Gaussian noise:

$$\tilde{f}_t(x, y, t) \;=\; f_t(x, y, t) + \eta(x, y, t)$$

where $\eta(x, y, t)$ has a mean-zero Gaussian density with variance σ^2.

Then, given the gradient constraint equation, it follows that

$$\vec{\mathbf{u}}^T \vec{\nabla} f + \tilde{f}_t \;\sim\; N(0,\, \sigma^2)$$

So, if we know $\vec{\mathbf{u}}$, then the probability of observing $(\vec{\nabla} f,\ \tilde{f}_t)$ is

$$p(\vec{\nabla} f, \tilde{f}_t \,|\, \vec{\mathbf{u}}) \;=\; \frac{1}{\sqrt{2\pi}\sigma} \exp\left(\frac{-(\vec{\mathbf{u}}^T \vec{\nabla} f + \tilde{f}_t)^2}{2\sigma^2} \right)$$

Assuming constraints at N pixels, each with IID Gaussian noise, the joint likelihood is the product of the individual likelihoods:

$$L(\vec{\mathbf{u}}) \;=\; \prod_{x,y} p(\vec{\nabla} f(\vec{\mathbf{x}}, t), \tilde{f}_t(\vec{\mathbf{x}}, t) \mid \vec{\mathbf{u}})$$

Optimal Estimation

Maximum likelihood (ML): Find the flow $\vec{\mathbf{u}}$ that maximizes the probability of observing the data given $\vec{\mathbf{u}}$. Often we instead minimize the negative log likelihood:

$$\begin{aligned} L(\vec{\mathbf{u}}) &= -\sum_{x,y} \log p(\vec{\nabla} f(\vec{\mathbf{x}},t),\ \tilde{f}_t(\vec{\mathbf{x}},t) \mid \vec{\mathbf{u}}) \\ &= -N \log(\sqrt{2\pi}\sigma) + \sum_{x,y} \frac{(\vec{\mathbf{u}}(\vec{\mathbf{x}},t) \cdot \vec{\nabla} f(\vec{\mathbf{x}},t) + \tilde{f}_t(\vec{\mathbf{x}},t))^2}{2\sigma^2} \end{aligned}$$

Note: L is identical to our previous objective function $\hat{E}$ except for the support $g(x,y)$ and the constant term.

Maximum A Posteriori (MAP): Include prior beliefs, $p(\vec{\mathbf{u}})$, and use Bayes' rule to express the posterior flow distribution:

$$p(\vec{\mathbf{u}} \mid \{\vec{\nabla} f, f_t\}_{\vec{\mathbf{x}}}) = \frac{p(\{\vec{\nabla} f, f_t\}_{\vec{\mathbf{x}}} \mid \vec{\mathbf{u}})\, p(\vec{\mathbf{u}})}{p(\{\vec{\nabla} f, f_t\}_{\vec{\mathbf{x}}})}$$

For example, assume a (Gaussian) slow motion prior:

$$p(\vec{\mathbf{u}}) = \frac{1}{\sqrt{2\pi}\sigma} \exp\left(\frac{-\vec{\mathbf{u}}^T \vec{\mathbf{u}}}{2\sigma_u^2}\right)$$

Then the flow that maximizes the posterior is given by

$$\hat{\mathbf{u}} = \left(\mathbf{M} + \frac{1}{\sigma_u^2}\mathbf{I}\right)^{-1} \vec{\mathbf{b}}$$

where

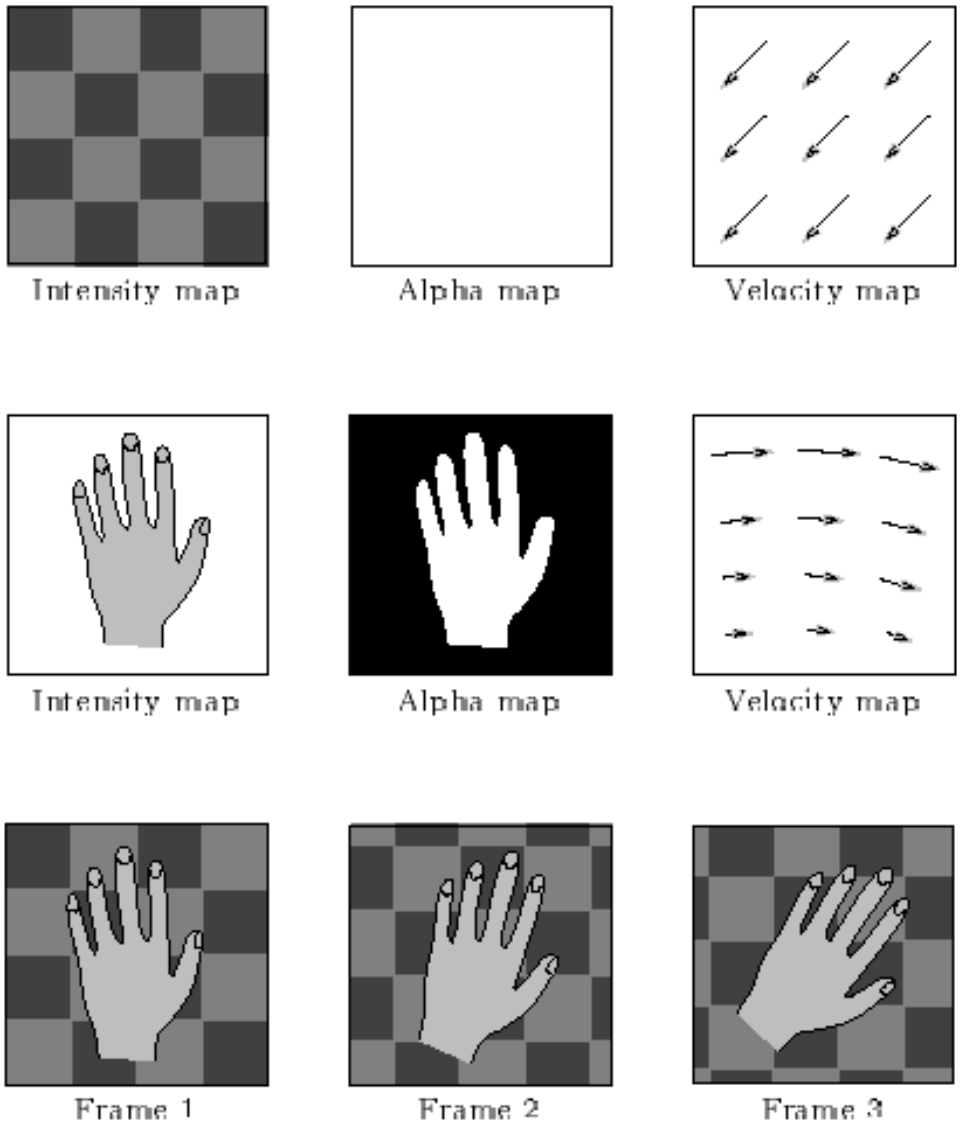
$$\mathbf{M} = \frac{1}{\sigma^2} \sum_{x,y} \vec{\nabla} f\, \vec{\nabla} f^T \quad , \quad \vec{\mathbf{b}} = -\frac{1}{\sigma^2} \sum_{x,y} \vec{\nabla} f\, f_t$$

ML / MAP Estimation (cont)

Remarks:

- The inverse Hessian of the negative log likelihood provides a measure of estimator covariance (easy for Gaussian densities).
- The noise variance, σ^2, could vary as function of $\vec{\mathbf{x}}$. For example, one might expect an affine flow model to fit well near the patch centre, $\vec{\mathbf{x}}_0$, but not as well further from the centre. If $\sigma^2(\vec{\mathbf{x}}) = 1/g(\vec{\mathbf{x}} - \vec{\mathbf{x}}_0)$ for a low-pass window g, then one obtains a weighted LS estimator like that introduced in (3).
- Alternative noise models can be incorporated in spatial and temporal derivatives (e.g., total least squares). One can also assume non-Gaussian noise and outliers (to obtain robust estimators)
- Other properties of probabilistic formulation: Ability to propagate "distributions" to capture belief uncertainty.
- Use of probabilistic calculus for fusing information from other sources

Layered Motion Models



Layered Motion:

The scene is modeled as a cardboard cutout. Each layer has an associated appearance, an alpha map (specifying opacity or occupancy in the simplest case), and a motion model. Layers are warped by the motion model, texture-mapped with the appearance model, and then combined according to their depth-order using the alpha maps.

Further Readings

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.

J. L. Barron, D. J. Fleet, and S. S. Beauchemin. Performance of optical flow techniques. *International Journal of Computer Vision*, 12(1):43–77, 1994.

J. R. Bergen, P. Anandan, K. Hanna, and R. Hingorani. Hierarchical model-based motion estimation. *Proc. Second European Conf. on Comp. Vis.*, pp. 237–252. Springer-Verlag, 1992.

D. J. Fleet and Y. Weiss. Optical flow estimation. In *Mathematical models for Computer Vision: The Handbook*. N. Paragios, Y. Chen, and O. Faugeras (eds.), Springer, 2005.

A. Jepson and M. J. Black. Mixture models for optical flow computation. *Proc. IEEE Conf. Computer Vision and Pattern Recognition*, pp. 760–761, New York, June 1993.

E P Simoncelli, E H Adelson, and D J Heeger. Probability distributions of optical flow. *Proc. Conf. Computer Vision and Pattern Recognition*, pp. 310–315, Mauii, HI, June 1991.

J.Y.A. Wang and E.H. Adelson. Representing moving images with layers, *IEEE Trans. on Image Processing*, 3(5):625–638, 1994.

Y. Weiss and E.H. Adelson. A unified mixture framework for motion segmentation: Incorporating spatial coherence and estimating the number of models. *IEEE Proc. Computer Vision and Pattern Recognition*, San Francisco, pp. 321–326, 1996.