We now turn to a problem related to corner detection, which comes up in video processing and in binocular stereo vision. Suppose we have two images $I(x, y)$ and $J(x, y)$ that are almost the same. We may have two neighboring image frames in a video, or we may have two images taken by two cameras (binocular stereo). We would like to find correspondences between points in the two images. In particular, we will look at the case that the images are related by local shifts. The problem of finding these shifts is sometimes known as "image registration", since we want to register the images (or put the points into correspondence). In the context of motion and video, it is known as "optical flow". In the context of binocular vision, it is called "disparity" estimation. Today we will look at a classical method for solving this problem, which is known as the Lucas-Kanade method.[1] We will see that the method involves similar concepts and algorithms to the corner detection problem. We will also briefly relate this problem to the "tracking" problem, where one is concerned with registering specific points or regions (rather than all points in the image) and one performs the registration over many image frames, not just two frames.

1D image registration

Let's first go to the 1D problem to make sure we understand the basic idea, and work with images $I(x)$ and $J(x)$. For any fixed x_0 in image J, we would like to find a corresponding point $x_0 + h$ in image I such that $I(x + h) \approx J(x)$ in a neighborhood of x_0. This means that I is shifted by $h > 0$, and so we would need to shift I to the left to make it align with J.

We assume that the shift h between I and J is roughly constant in local neighborhoods in the image. Then, to find h at each x_0 we could try to minimize the following:

$$\sum_{x \in Ngd(x_0)} W(x - x_0)\{I(x + h) - J(x)\}^2. \tag{1}$$

We take a weighted sum over the local neighborhood here. We also implicitly assume that the I and J images have been smoothed to reduce the effects of noise.

We could minimize the above by just doing a brute force search over h. But this would force us to choose particular h values e.g. integer pixel increments and there is no reason why the true shift is a pixel increment. Instead, we take a first order Taylor series approximation of I at each point x in the neighborhood of x_0:

$$I(x + h) \approx I(x) + h\frac{dI}{dx}(x)$$

and then minimize the following:

$$\sum_{x \in Ngd(x_0)} W(x - x_0)(I(x) - J(x) + h\frac{dI(x)}{dx}\ )^2. \tag{2}$$

The only variable in the above expression is h. The rest are given intensity values and their derivatives which are estimated by taking local differences. This is a familiar least squares problem. Specifically, the above expression is a quadratic $ah^2 + bh + c$ and by inspection (let $h \to \pm\infty$) this quadratic has a minimum rather than maximum. So we would like to find the value of h that

[1] B.D. Lucas and T. Kanade, "An iterative image registration technique with an application to stereo vision", IJCAI 1981.

minimizes this expression. That is easy. We just take the derivative with respect to h and set it to 0,

$$\sum_{x \in Ngd(x_0)} W(x - x_0)(I(x) - J(x) + h\frac{dI(x)}{dx}\ )\frac{dI(x)}{dx}\ = 0$$

which can be rewritten:

$$h = \frac{\sum_{x \in Ngd(x_0)} W(x - x_0)(J(x) - I(x))\frac{dI(x)}{dx}}{\sum_{x \in Ngd(x_0)} W(x - x_0)(\frac{dI(x)}{dx}\ )^2}.$$

Note again that this solution is not restricted to integer values of h here.

Recall we have made a first order approximation of $I(x + h)$ around x, namely $I(x)$ should be linear over small distances h. This assumption is reasonably because we have assumed that we have smoothed the image with a Gaussian which has reduced the noise and blurred the edges. We can only assume that the first order model (Eq. 2) holds for small values of h. If the real displacement h happens to be very different from 0, then higher order terms, $\frac{1}{k!}\frac{d^k I}{d^k x}(x)h^k$, in the Taylor series expansion of $I(x + h)$ would contribute and the actual error function (Eq. 1)) would be more complicated than the quadratic error function of (Eq. 2)).

Iterative method

We do not expect the estimate of h to be exactly correct since the Taylor expansion of $I(x)$ will generally have *some* second and higher order term contribution. What to do? Suppose we have estimated h and we wish to refine our estimate. Or more generally, suppose after k iterations of refining our estimate h, we have an estimate h_k and we want further refine this estimate to be h_{k+1}. How do we do it?

Given an estimate h_k for the shift between I and J at some location x_0, we want to find an h such that $I(x + h_k + h) \approx J(x)$ in a neighborhood of $x = x_0$. We proceed exactly as before, but now $x + h_k$ replaces x as the parameter for $I()$. So,

$$I(x + h_k + h) \approx I(x + h_k) + h\frac{dI(x + h_k)}{dx}$$

and now we want to minimize

$$\sum_{x \in Ngd(x_0)} W(x - x_0)(I(x + h_k) - J(x) + h\frac{dI(x + h_k)}{dx}\ )^2$$

and so

$$h = -\frac{\sum_{x \in Ngd(x_0)} W(x - x_0)(I(x + h_k) - J(x))\frac{dI(x+h_k)}{dx}}{\sum_{x \in Ngd(x_0)} W(x - x_0)(\frac{dI(x+h_k)}{dx}\ )^2}.$$

Notice here that if h_k is not an integer, then we need to interpolate the intensities to estimate what $I(x + h_k)$ and $\frac{dI(x+h_k)}{dx}$ mean. (Interpolation can be done in several ways, but for simplicity just assume linear interpolation.)

After solving for h at the kth step, we update:

$$h_{k+1} \leftarrow h_k + h\ .$$

We can repeat this until the increment h is small, in which case the estimate has converged. Or if it doesn't converge, then we give up.

Why might this method not work? One reason is that we need the image intensity derivatives to be non-zero, since otherwise computing h will give us a division by 0. Intuitively, if the image intensity I derivative is zero, then the image is locally constant and so there is no basis for matching different x positions in image $I(x)$ values to any particular $J(x)$ value.

But the iterative algorithm might not converge even if the intensity derivatives are non-zero, since the error function of (Eq. 1) actually might have multiple local minima as a function of h and the iterative algorithm might get stuck in the wrong minimum. When the true displacement h is small, one typically does not get stuck in local minima. However, when the h values are large, one typical *does* get stuck, and another approach is needed. We will present this approach in lecture when we discuss scale space.

2D Image registration

Let's next consider 2D images $I(x, y)$ and $J(x, y)$. We would like to know what shift (h_x, h_y) minimizes

$$\sum_{(x,y)\in Ngd(x_0,y_0)} \{I(x + h_x, y + h_y) - J(x, y)\}^2$$

We could use brute force with integer values of (h_x, h_y) but then this would give us an integer valued solution only. Instead we use a Taylor series expansion as before. Assume that the images have been smoothed by convolving with a 2D Gaussian. Then,

$$I(x + h_x, y + h_y) \;\approx\; I(x,\ y) + \frac{\partial I}{\partial x} h_x + \frac{\partial I}{\partial y} h_y$$

where h_x, h_y are small, and the partial derivatives are evaluated at (x, y). We then minimize the sum of squares:

$$\sum_{(x,y)\in Ngd(x_0,y_0)} W(x - x_0, y - y_0)\{I(x, y) - J(x, y) + \frac{\partial I}{\partial x} h_x + \frac{\partial I}{\partial y} h_y\}^2$$

where we have weighted the terms with a Gaussian window. This is a quadratic expression in two variables h_x and h_y and it has a single minimum. To see it is a minimum, note that if we fix either h_x or h_y and let that other go to infinity, then the expression goes to positive infinity.

We can write the above expression within the curly brackets by defining an $N \times 1$ vector as follows:

$$\begin{bmatrix} \frac{\partial I}{\partial x} & \frac{\partial I}{\partial y} \\ \vdots & \vdots \end{bmatrix}_{N\times 2} \begin{bmatrix} h_x \\ h_y \end{bmatrix}_{2\times 1} - \begin{bmatrix} J(x, y) - I(x, y) \\ \cdots \end{bmatrix}_{N\times 1}$$

where the rows correspond to the different (x, y) in the neighborhood of (x_0, y_0). We are trying to find the (h_x, h_y) vector that minimizes the length of this N vector. You may recognize this as the regression problem. Specifically, we are now minimizing $(\mathbf{Au} - \mathbf{b})^\mathbf{T}\mathbf{W}(\mathbf{Au} - \mathbf{b})$ where $\mathbf{u} = (h_x, h_y)^T$ and $\mathbf{W}$ is a diagonal matrix where W_{ii} is the weight of the corresponding pixel x in the neighborhood. Similar to what we saw in the least squares lecture, we now want to solve

$$\mathbf{A^T W A u} = \mathbf{A^T W b}.$$

For the present problem, this gives:

$$\begin{bmatrix} \sum_{x,y} W(\cdot)(\frac{\partial I}{\partial x})^2 & \sum_{x,y} W(\cdot)(\frac{\partial I}{\partial x})(\frac{\partial I}{\partial y}) \\ \sum_{x,y} W(\cdot)(\frac{\partial I}{\partial x})(\frac{\partial I}{\partial y}) & \sum_{x,y} W(\cdot)(\frac{\partial I}{\partial y})^2 \end{bmatrix} \begin{bmatrix} h_x \\ h_y \end{bmatrix} = \begin{bmatrix} \sum_{x,y} W(\cdot)(J(x,y) - I(x,y))\frac{\partial I}{\partial x} \\ \sum_{x,y} W(\cdot)(J(x,y) - I(x,y))\frac{\partial I}{\partial y} \end{bmatrix}$$

and we need to solve for the (h_x, h_y).

You recognize the matrix on the left. It is the second moment matrix $\mathbf{M}$ we discussed last lecture. We can solve for (h_x, h_y) as long as the second moment matrix is invertible, or equivalently, as long as both eigenvalues are non-zero. We get a zero eigenvalue only if the image gradients in the neighborhood are all parallel, or if the gradients are zero.

It is no accident that the second moment matrix arises in today's problem and in the corner detection problem last lecture. In corner detection, we are looking for points (x, y) which are locally distinctive. This meant that if we were to compare intensities in the neighborhood of (x_0, y_0) with intensities in the neighborhood of a nearby point (x_0+h_x, y_0+h_y) then we would like these two local intensity patterns to be different. In the case of finding corresponding points between two image patches in images I and J, we have the same issues. If the intensities are not locally distinctive, then there is no unique way to find a correspondence. For example, if the gradients of I are all in the same direction near (x_0, y_0), then we could slide I in a direction that is perpendicular to the gradient and the local intensities wouldn't change. But this just means that we cannot decide uniquely where to slide I locally so it matches J.

Just like in the 1D case, we can define an iterative method. We have taken a Taylor series expansion of $I(x, y)$ near (x_0, y_0) and this means we cannot expect a perfect estimate of (h_x, h_y). To improve accuracy, we can try to iterate. Suppose we have the k^{th} estimate (h_x^k, h_y^k) and we wish to estimate (h_x^{k+1}, h_y^{k+1}). Then, we update our estimates

$$h_x^{k+1} \leftarrow h_x^k + h_x$$

$$h_y^{k+1} \leftarrow h_y^k + h_y.$$

We repeat until we converge (or if we don't converge then we give up).

for each pixel (x_0, y_0) {
    $k = 0$
    $(h_x^0, h_y^0) = (0, 0)$
    repeat {
        Solve for (h_x, h_y) using $I(x + h_x^k + h_x, y_0 + h_y^k + h_y)$ and $J(x, y)$
            using linear (matrix) approximation
        $(h_x^{k+1}, h_y^{k+1}) \leftarrow (h_x^k + h_x, h_y^k + h_y)$
        $k \leftarrow k + 1$
    } until the sum of squared error is sufficiently small, or give up

Optical flow (Lucas-Kanade 1981)

In the case where we have a video, the notation changes slightly. We now have a sequence of images $I(x, y, t)$ where t is frame number. In this case, we are comparing two neighboring image frames $I(x, y, t)$ and $I(x, y, t+1)$, so we can compare $I(x, y, t)$ with a shifted $I(x + h_x, y + h_y, t + 1)$ and write the latter as

$$I(x + h_x, y + h_y, t + 1) \;\approx\; I(x,\; y, t) + \frac{\partial I}{\partial x} h_x + \frac{\partial I}{\partial y} h_y + \frac{\partial I}{\partial t}$$

and we will minimize the (weighted) sum of squares of terms:

$$\frac{\partial I}{\partial x} h_x + \frac{\partial I}{\partial y} h_y + \frac{\partial I}{\partial t}$$

where

$$\frac{\partial I}{\partial t} \equiv I(x, y, t + 1) - I(x, y)$$

Verify for yourself that the problem now is to solve:

$$\begin{bmatrix} \sum_{x,y} W(\cdot)(\frac{\partial I}{\partial x})^2 & \sum_{x,y} W(\cdot)(\frac{\partial I}{\partial x})(\frac{\partial I}{\partial y}) \\ \sum_{x,y} W(\cdot)(\frac{\partial I}{\partial x})(\frac{\partial I}{\partial y}) & \sum_{x,y} W(\cdot)(\frac{\partial I}{\partial y})^2 \end{bmatrix} \begin{bmatrix} h_x \\ h_y \end{bmatrix} = - \begin{bmatrix} \sum_{x,y} W(\cdot)(\frac{\partial I}{\partial t})\frac{\partial I}{\partial x} \\ \sum_{x,y} W(\cdot)(\frac{\partial I}{\partial t})\frac{\partial I}{\partial y} \end{bmatrix}$$

For points (x_0, y_0) at which the (second momemt) matrix on the left is invertible – in particular, if both eigenvalues are strictly greater than zero – one can solve for (h_x, h_y). One can also iterate the solution to get a more accurate estimate. We will return to this problem next lecture when we discuss *scale space.*

KLT tracking (Tomasi and Kanade 1991)

The Lucas-Kanade method allows you to estimate the image deformation from one frame to another, and in particular it allows for a deformation estimate whereever the second moment matrix has two positive eigenvalues. (To get a reliable estimate, the eigenvalues need to both be sufficiently large – otherwise, the estimate will be too sensitive to noise and errors in the translation model – see below).

A similar method is used for tracking locally distinctive points. This is called KLT tracking. The main idea is described in paper by Tomasi and Kanade in 1991 and the method is based on the Lucas and Kanade method described above, but restricted to points for which both eigenvalues are sufficiently large.

Tracking with scale, shear, rotation (Shi and Tomasi 1994)

The above method allows for local translations only, but images can differ from each other in more general ways. A more general model is to allow local scaling, shearing, and rotation. This more general model is often used when we are *tracking* locally distinctive points.

Let's suppose we have two nearby image patches in frames I and J and we have a locally distinctive point at $I(x_0, y_0)$. For any point $\mathbf{x} = (x, y)$ in a neighborhood of $\mathbf{x_0} = (x_0, y_0)$, we have (to first order)

$$I(\mathbf{x} + \mathbf{h} + \mathbf{D}(\mathbf{x} - \mathbf{x_0})) = J(\mathbf{x})$$

where $\mathbf{h} = (h_x h_y)$ is a local translation as above, and $\mathbf{D}$ is a general 2×2 matrix that models "deformations" which include shear, rotation, and scaling,

$$\begin{bmatrix} D_{11} & D_{12} \\ D_{21} & D_{22} \end{bmatrix}.$$

To see how various transformations $\mathbf{D}$ could be realized, consider a few basic examples. More complicated examples can be created by multiplying the following matrices together.

- For an identity transformation (no rotation, scaling or shear),

$$\begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix}.$$

- For a scaling by a scale factor s_x or s_y,

$$\begin{bmatrix} s_x & 0 \\ 0 & 1 \end{bmatrix} \quad \begin{bmatrix} 1 & 0 \\ 0 & s_y \end{bmatrix}.$$

- For a shear in the x direction

$$\begin{bmatrix} 1 & a \\ 0 & 1 \end{bmatrix}.$$

- For a rotation by angle θ,

$$\begin{bmatrix} \cos\theta & -\sin\theta \\ \sin\theta & \cos\theta \end{bmatrix}.$$

We treat these deformations together in the constant matrix $\mathbf{D}$, whose entries we would like to estimate along with those of the vector $\mathbf{h}$, by minimizing the following:

$$\sum_{\mathbf{x} \in Ngd(\mathbf{x_0})} W(\mathbf{x} - \mathbf{x_0})(I(\mathbf{x} + \mathbf{h} + \mathbf{D}(\mathbf{x} - \mathbf{x_0})) - J(\mathbf{x}))^2$$

where $\mathbf{x} = (x, y)$ and $\mathbf{x_0} = (x_0, y_0)$.

One can then set up a least squares problem similarly as before by taking the local Taylor series expansion of $I()$ around each $\mathbf{x} = (x, y)$. This is a higher dimensional problem, namely that one solves for six variables rather than two: the four elements of the $\mathbf{D}$ matrix and the h_x and h_y. I will post an exercise on how to do this. (If you wish to see the details on how to do this, have a look at the original paper by Shi and Tomasi (1994).)

[See the slides for some examples of Lucas-Kanade optical flow and an example of tracking.]