

Harris corners

Consider any unit vector $(\Delta x, \Delta y) = (\cos \theta, \sin \theta)$. Notice that

$$(\cos \theta, \sin \theta) \mathbf{M} (\cos \theta, \sin \theta)^T = \sum_{(x,y) \in N_{gd}(x_0,y_0)} (\cos \theta, \sin \theta) \begin{bmatrix} \frac{\partial I}{\partial x} \\ \frac{\partial I}{\partial y} \end{bmatrix} \begin{bmatrix} \frac{\partial I}{\partial x} & \frac{\partial I}{\partial y} \end{bmatrix} \begin{bmatrix} \cos \theta \\ \sin \theta \end{bmatrix}$$

which is the sum of the square of the directional derivatives in direction θ , over the neighborhood of (x_0, y_0) .

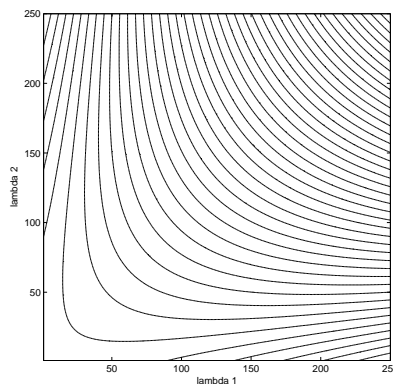
For the intensities at a point (x, y) to be locally distinctive, the above quantity should be large. How can we quickly examine this condition for each (x_0, y_0) ? Since \mathbf{M} is symmetric, it has two real eigenvalues and the two eigenvectors are orthogonal. By inspection, $(\Delta x, \Delta y) \mathbf{M} (\Delta x, \Delta y)^T \geq 0$ for any $(\Delta x, \Delta y)$. Thus, the eigenvalues of \mathbf{M} must be positive i.e. letting $(\Delta x, \Delta y)$ be an eigenvector with eigenvalue λ , we see $\lambda |(\Delta x, \Delta y)| \geq 0$. For $(\Delta x, \Delta y) \mathbf{M} (\Delta x, \Delta y)^T$ to be much larger than 0 for any unit length $(\Delta x, \Delta y)$, we require that *both* eigenvalues must be much greater than zero.

Note that computing the eigenvalues requires solving a quadratic equation, $a\lambda^2 + b\lambda + c = 0$, which in turn requires computing a square root. We will check every pixel (x_0, y_0) in an image to see if it is locally distinctive, and so it seems we need to compute a square root at each pixel, which can be slow.¹ One trick to avoid computing λ_1, λ_2 explicitly was proposed by Harris and Stevens² is to take advantage of a basic fact from matrix algebra that the determinant of a matrix with eigenvalues λ_1 and λ_2 is the product $\lambda_1 \lambda_2$ and the trace is $\lambda_1 + \lambda_2$. The trick is to convert the conditions “both eigenvalues are large” into a condition on the determinant and trace of \mathbf{M} , both of which can easily be computed from \mathbf{M} i.e. no need to compute the λ 's explicitly.

Harris and Stevens noticed that if k is small constant (say $k \approx 0.1$) then

$$\lambda_1 \lambda_2 - k(\lambda_1 + \lambda_2)^2$$

is negative if one of the eigenvalues is 0, and this quantity is small if both of the eigenvalues are small. Otherwise this quantity is large. (The figure below shows the iso-values of this expression when $k = 0.1$.) Thus the *Harris corner detector* or *Harris interest point detector*, as its now known,



¹Twenty years ago, this was more of an issue.

²“A combined corner and edge detector” by C. Harris and M. Stevens, Proceedings of the 4th Alvey Vision Conference, 1988. Cited nearly 4000 times according to Google Scholar.

looks for points in the image where the above expression takes a value above some given threshold.

Notice that if the determinant is near 0 but the trace is much different from zero, then one of the eigenvalues must be large and the other must be near zero. In this case, there must be strong image intensity gradients in the neighborhood of the point, but the gradients would be in only one direction. Thus, Harris and Stevens argued that using the trace and determinant of the second moment matrix could be used to detect “edges” as well as corners, where by “edges” they mean points whose neighborhood contains strong gradients that are all roughly in the same direction. (This now is a more general notion of an edge that we discussed earlier, just as Harris’s “corner” is a more general notion of a corner.)

Image registration (Lucas-Kanade)

We now turn to a related problem which comes up in video processing and in 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 (stereo). We would like to find correspondences between points in the two images. This is sometimes known as “image registration”. Today we will look at a classical method for solving this problem, known as the Lucas-Kanade method.³ We will see that the method involves similar concepts and algorithms to the edge and corner detection problem.

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 . To find h , we could try to minimize the following:

$$\sum_{x \in \text{Nbd}(x_0)} \{I(x + h) - J(x)\}^2$$

We use a summation here because we don’t just want to find a matching point. We want a small neighborhood of matching points.

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. Instead, one typically assumes $I(x)$ is smooth (because we have blurred it with a Gaussian beforehand), takes a first order Taylor series approximation:

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

and then minimizes the following:

$$\sum_{x \in \text{Nbd}(x_0)} \left(I(x) - J(x) + h \frac{dI(x)}{dx} \right)^2.$$

³B.D. Lucas and T. Kanade, “An iterative image registration technique with an application to stereo vision”, IJCAI 1981.

The only variable in the above expression is h . The rest are given intensity values and their derivatives. So, the above expression is a quadratic $ah^2 + bh + c$. We would like to find the value of h that minimizes this expression. That is easy. We just take the derivative with respect to h and set it to 0,

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

and so

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

Note that we are not restricted to integer values of h here.

Iterative method

The above method assumes that the function $I(x)$ is linear in the neighborhood of x_0 . While this approximation may be a good enough to get an estimate of h , we do not expect it to give us h exactly since $I(x)$ will generally have second and higher order terms as well.

Suppose we have estimated h and we wish to refine our estimate. Or after k iterations of estimating h , we have an h_k and we want an estimate h_{k+1} . How do we do it? We want to find an h such that $I(x + h_k + h) = J(x)$ near $x = x_0$. 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

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

and so

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

Notice here that if h_k is not an integer, then we need to interpolate to say what $I(x + h_k)$ and $\frac{dI(x + h_k)}{dx}$ are.

After solving for h at the k th step, we update:

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

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

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

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

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. Assume that the images have been blurred with a 2D Gaussian so that they are smooth,

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

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

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

This is a quadratic expression in two variables 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.

Take the derivative of the above summation with respect to h_x and h_y and set each to zero. This gives

$$\begin{bmatrix} \sum \left(\frac{\partial I}{\partial x}\right)^2 & \sum \left(\frac{\partial I}{\partial x}\right)\left(\frac{\partial I}{\partial y}\right) \\ \sum \left(\frac{\partial I}{\partial x}\right)\left(\frac{\partial I}{\partial y}\right) & \sum \left(\frac{\partial I}{\partial y}\right)^2 \end{bmatrix} \begin{bmatrix} h_x \\ h_y \end{bmatrix} = - \begin{bmatrix} \sum (I(x, y) - J(x, y)) \frac{\partial I}{\partial x} \\ \sum (I(x, y) - J(x, y)) \frac{\partial I}{\partial y} \end{bmatrix}$$

You recognize the matrix on the left. It is the second moment matrix \mathbf{M} we saw in the corner detection method. We can solve for (h_x, h_y) as long as the second moment matrix is invertible, or equivalently, as long as the two eigenvalues are non-zero.

We get a zero eigenvalue only if the directional derivatives at *all points* in the neighborhood vanish. That can happen in two ways. One way is that the direction of $\nabla I(x, y)$ is constant in the neighborhood (and so the directional derivative in the perpendicular direction would be zero). The other is that $\nabla I(x, y)$ is zero in the entire neighborhood i.e. the image intensity is constant. In the latter case, both of the eigenvalues 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. Our condition for this being true was that the eigenvalues of \mathbf{M} were non-zero and we now see what this means in terms of the intensity gradients (namely that they cannot all be zero, and they cannot all be in the same direction). The same idea holds when we are trying to match patches *between images* I and J . 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 locally 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 (locally).

A few more points to mention: First, another way of writing the second moment matrix is in terms of an $n \times 2$ matrix \mathbf{A} whose rows are $(\frac{\partial I}{\partial x}, \frac{\partial I}{\partial y})$ for the different $(x, y) \in \text{Ngd}(x_0, y_0)$.

$$\mathbf{M} = \mathbf{A}^T \mathbf{A} = \begin{bmatrix} \sum (\frac{\partial I}{\partial x})^2 & \sum (\frac{\partial I}{\partial x})(\frac{\partial I}{\partial y}) \\ \sum (\frac{\partial I}{\partial x})(\frac{\partial I}{\partial y}) & \sum (\frac{\partial I}{\partial y})^2 \end{bmatrix}$$

This can clean up the notation a bit. For example, if we write the $I(x, y)$ and $J(x, y)$ values as vectors \vec{I} and \vec{J} then see that we want to solve:

$$\mathbf{A}^T \mathbf{A} (h_x, h_y)^T = \mathbf{A}^T (\vec{J} - \vec{I}).$$

Second, 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}) . We define a new function $I^k(x, y) = I(x + h_x^k, y + h_y^k)$ and we solve for (h_x, h_y) . Then, we update our estimate

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

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

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