Machine vision
Lecture Summary # 4

Fig. 1. Edge detection using the Sobel edge detector. (a) Edges by applying the horizontal Sobel mask. (b) Edges by applying the vertical Sobel mask, and (c) Original image.

A. Second derivative edge detection

The gradient can be defined as the derivative of a multidimensional function, it can be used to measure change rates of the function in the x- and y-directions. Edge detection algorithms that we discussed previously are based on different approximations of the gradient. The application of the horizontal and vertical Sobel edge detectors is shown in figure 1. The original image is shown at the bottom of figure 1.

Another solution is to use the second derivative given by:

$$\nabla^2 f = \frac{\partial^2 f}{\partial x^2} + \frac{\partial^2 f}{\partial y^2}$$

This operator is called the Laplacian of function $f$. Since the Laplacian is a second order derivative, edges correspond to zero crossing. In a similar way to the gradient approximation, Laplacian operator can be approximated by difference equations such as

$$\nabla^2_x f = f(i,j+1) - 2f(i,j) + f(i,j-1)$$
$$\nabla^2_y f = f(i+1,j) - 2f(i,j) + f(i-1,j)$$

from which we obtain

$$\nabla^2 f = f(i+1,j) - 2f(i,j) + f(i-1,j) + f(i,j+1) - 2f(i,j) + f(i,j-1)$$

The mask for Laplacian is given

$$L = \begin{bmatrix} 0 & 1 & 0 \\ 1 & -4 & 1 \\ 0 & 1 & 0 \end{bmatrix}$$

Another Laplacian mask that gives more importance to the center element is given by

$$L = \begin{bmatrix} 1 & 1 & 1 \\ 1 & -8 & 1 \\ 1 & 1 & 1 \end{bmatrix}$$

Note that the sum of the elements in the masks is zero in both cases.

B. Example

For an image given by

$$A = \begin{bmatrix} c_1 & c_2 & c_3 \\ c_4 & c_5 & c_6 \\ c_7 & c_8 & c_9 \end{bmatrix}$$

The Laplacian is given by

$$\nabla^2 f = -4c_5 + c_2 + c_4 + c_6 + c_8$$

For the image of last time:

$$A = \begin{bmatrix} 30 & 66 & 65 \\ 14 & 30 & 70 \\ 12 & 15 & 40 \end{bmatrix}$$

the Laplacian is $\nabla^2 f = 45$ based on mask (6) and 72 based on mask (7).

C. Properties of Laplacian

- It is less expensive computationally: one mask only.
- It is scalar, and thus it does not provide information about edge direction.
- It is more sensitive to noise, even an isolated point results is zero crossing.

D. Directional derivative

The directional derivative along a vector $v$ represents the rate of change in a given direction. The directional derivative along vector $v = (v_1, v_2)$ is defined by the limit

$$\nabla_v f = \lim_{h \to 0} \frac{f(x + hv) - f(x)}{h}$$

The directional derivative is given by

$$\nabla_v f = \nabla f \cdot v$$

The derivative in direction $\theta$ is given by

$$\nabla_{\theta} f = \nabla_x f \cos \theta + \nabla_y f \sin \theta$$
Fig. 2. Application of the masks given by system (14)

θ = 0 corresponds to the x-component of the gradient and θ = 90 corresponds to the y-component of the gradient. Two simple masks that can be used to perform inclined edge detection are

\[
M_1 = \begin{bmatrix} -1 & -1 & 2 \\ -1 & 2 & -1 \\ 2 & -1 & -1 \end{bmatrix} \quad M_2 = \begin{bmatrix} 2 & -1 & -1 \\ -1 & 2 & -1 \\ -1 & -1 & 2 \end{bmatrix}
\] (14)

The application of these matrices to the image of figure 2-top is shown in figure 2-bottom. These masks are not effective in detecting horizontal and vertical edges.

E. Laplacian of Gaussian (LoG)

The second derivative of the image is very sensitive to noise. Filtering (smoothing) before applying Laplacian mask could help solve the problem. One algorithm introduced by Marr and Hildreth uses the Laplacian of Gaussian (LoG), which combines Gaussian filtering with Laplacian for edge detection. The output of the LoG operator is obtained using convolution as follows

\[
g(x, y) = \nabla^2 \left[ h(x, y) \ast f(x, y) \right]
\] (15)

or

\[
g(x, y) = \left[ \nabla^2 h(x, y) \right] \ast f(x, y)
\] (16)

where \( h(x, y) \) is the mask of the Gaussian filter and \( \nabla^2 h(x, y) \) is the Laplacian of the Gaussian filter.

Based on equations (15) and (16), there are two different ways to implement the LoG filter:

- Perform convolution of the image with the Gaussian filter
- Then compute the Laplacian of the result.

F. Canny edge detector

The canny edge detector is among the most popular edge detection algorithms. It consists of five different steps

- Smoothing
- Finding gradients (edge detection, any of the edge detection masks can be used)
- Non-maxima suppression
- Double thresholding
- Edge tracking (linking)

The steps are illustrated in figure 3 and can be summarized as follows

- Step 1 Smoothing with Gaussian filter to reduce noise and unwanted details in the image.
- Step 2 Use gradient to detect edges, we can use any of the classical masks: Sobel, Prewitt, Roberts
- Step 3 Non maxima suppression to convert blurred edges to sharp edges
- Step 4 Thresholding to keep strong edges and remove noise.
- Step 5 Edge linking to connect edge points.

G. Some useful commands

- gradient(F) calculates the gradient of a matrix. Does not take uint8, you need to convert to double
- quiver: plots the orientation (angle of the gradient) and its magnitude

\[
G=\text{im2double}(G)
\]
\[
F=\text{gradient}(G)
\]
\[
[L1,L2]=\text{gradient}(G)
\]
\[
\text{quiver}(L1,L2)
\]

Function quiver takes the origin at the bottom left of the figure, therefore, necessary adjustment need to be made.
H. Harris corner detector

If we consider a small window in the image and we shift the window by small amount in various directions, we will have three different scenarios:

- No change: flat area
- Change in 1D: edge
- Change in 2D: corner

Change in intensity for a shift \((u,v)\) is given by

\[
E(u,v) = \sum_{i,j \in W} w(i,j) [f(i + u, j + v) - f(i, j)]^2
\]

(17)

where

- \(f(i,j)\) is the intensity
- \(f(i + u, j + v)\) is the shifted intensity
- \(w(i,j)\) is the window function.

The window function can be a rectangle or a 2D Gaussian. The difference

\[
f(i + u, j + v) - f(i, j)
\]

(18)

plays an important role in the algorithm, flat area corresponds to very small values of the difference and thus values of \(E(u,v)\) near zero. Corner or edge corresponds to higher values of \(E(u,v)\).

Using Taylor series for the shifted intensity, we get

\[
E(u,v) = \sum_{i,j \in W} w(i,j) [f(i,j) + u \frac{\partial f}{\partial x} + v \frac{\partial f}{\partial y} - f(i,j)]^2
\]

Using Taylor series for the shifted intensity, we get

\[
E(u,v) = \sum_{i,j \in W} w(i,j) \left[ u^2 (\nabla_x f)^2 + v^2 (\nabla_y f)^2 + 2uv \nabla_x f \nabla_y f \right] (19)
\]

Using Taylor series for the shifted intensity, we get

\[
E = \begin{bmatrix} u & v \end{bmatrix} \sum_{i,j \in W} \left[ \begin{bmatrix} (\nabla_x f)^2 & \nabla_x f \nabla_y f \\ \nabla_x f \nabla_y f & (\nabla_y f)^2 \end{bmatrix} \right] \begin{bmatrix} u \\ v \end{bmatrix}
\]

(21)

The entries of matrix \(M\) are just the components of the gradient. Harris suggested that it is possible to study the cornerness of an image using the eigenvalues of matrix \(M\). \(M\) is positive definite, thus \(\lambda_1, \lambda_2 > 0\). The determinant and the trace of \(M\) are given by

\[
det(M) = \lambda_1 \lambda_2
\]

(24)

\[
Trace(M) = \lambda_1 + \lambda_2
\]

(25)

The three cases are:

1) Both eigenvalues are small: Flat area, no edge, no corner. This corresponds to small values of the trace.
2) One eigenvalue is large and the other one is small, this corresponds to an edge.
3) Both eigenvalue are large; this corresponds a large value for the determinant and therefore, a corner.

Harris cornerness is given by the following equation

\[
H = \lambda_1 \lambda_2 - K(\lambda_1 + \lambda_2)
\]

(26)

For example, \(K = 0.04\). An illustration is shown in figure 5.
outside the circle. The filter in the frequency domain is given by

\[ H(u, v) = \begin{cases} 
1 & \text{when } L(u, v) \leq R_0 \\
0 & \text{when } L(u, v) > R_0 
\end{cases} \]  

where \( L \) represents a distance between point \((u, v)\) and the center of the FFT of the filter:

\[ L(u, v) = \sqrt{\left(u - \frac{m}{2}\right)^2 + \left(v - \frac{n}{2}\right)^2} \]  

The physical realization of the ideal low pass filter is not possible because of the sharp cut off frequency.

- **Low pass Butterworth.** The transfer function of the low pass Butterworth filter is given by

\[ H(u, v) = \frac{1}{1 + \left[ \frac{L(u, v)}{R_0} \right]^{2N}} \]  

where \( N \) is an integer typically with small values. This filter does not have a sharp cutoff frequency between passed and blocked frequencies.

- **Gaussian low pass filter.** The FFT of a Gaussian filter is Gaussian too, that is:

\[ H(u, v) = e^{-\frac{L(u, v)^2}{2R_0^2}} \]  

where the standard deviation is replaced by the radius, the standard deviation measures the spread about the center and \( R_0 \) does the same thing.

### B. Image sharpening in the frequency domain

High pass filters are used for this purpose. The general relationship between high pass and low pass filters is given by

\[ H_{HP} = 1 - H_{LP} \]  

The following high pass filters are related to the low pass filters discussed above.

- **Ideal high pass filter**

\[ H(u, v) = \begin{cases} 
1 & \text{when } L(u, v) \geq R_0 \\
0 & \text{when } L(u, v) < R_0 
\end{cases} \]  

- **High pass Butterworth filter**

\[ H(u, v) = \frac{1}{1 + \left[ \frac{R_0}{L(u, v)} \right]^{2N}} \]  

- **High pass filter derived from Gaussian filter**

\[ H(u, v) = 1 - e^{-\frac{L(u, v)^2}{2R_0^2}} \]  

The filters’ FFTs (representation in the frequency domain) are shown in figure 6.

Another filter that can be used for edge detection is the Laplacian filter. The transfer function of Laplacian filter is given by

\[ H(u, v) = -4\pi^2L^2(u, v) \]
Fig. 6. FFTs of low pass and high pass filters. (1) Ideal low pass, (2) Gaussian low pass, (3) Butterworth low pass, (4) Ideal high pass, (5) High pass derived from Gaussian low pass, (6) Butterworth high pass
Fig. 7. Frequency response of Gaussian low pass filter

Fig. 8. Frequency response of high pass filter
Fig. 9. Illustration of thresholding using histogram. Top: original image and histogram of the original image, (c) binary image with threshold 75, (d) binary image with interval [75,150]; (e) binary image with threshold 150.