EN.520.665 HW1

Qihua Gong

1. For all the three edge detector, the Gaussian Filter is the only separable circular symmetric fliter, so most edge detection algorithms use it. Edges generally appears in areas with different colors, brightness or textures. On way is to define the edge as an area where the brightness changes sharply. Mathematically, the slope and direction of a surface are defined by gradients and the derivative of the imagine will emphasize the high frequence part and amplify the noise. Therefore, it's generally need to perform low-pass filtering before calculating the gradient. To make the response of the edge decdor independent of direction, a circularly symmetric smoothing filter is needed. ~~The function~~ After using the Guassian fliter, it can both separate the operator and speed up the ~~ope~~ operation.

Marr-Hildreth: The improvement of the Marr-Hildreth edge detection algorithm is to first use a Gaussian filter to smooth the image, and then calculate the Laplacian of the result. The Gaussian part of the operator blurs the imagine, thereby reducing the gray scale and noise of the structure to a much smaller extent in terms of size. Compared with the mean filter smoothing, the Gaussian function smooths the image in the two domains of space and frequency, so the possibility of introducing non-existent artificial interference in the original image is small.

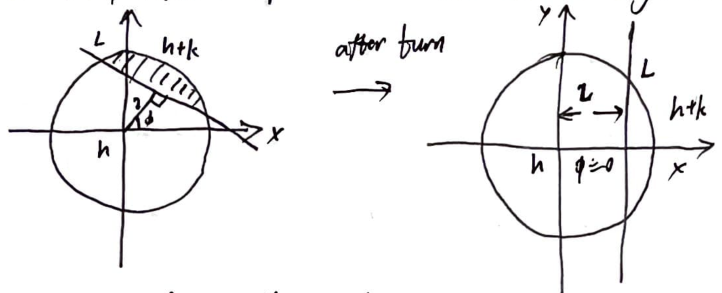
Canny edge detector: In order to minimize the influence of noise on the edge detection result, It's necessary to filter out the noise to prevent false detection cause by noise. In order to smooth the imagine, a Gaussian filter is used to convolve the image. This step will smooth the image to reduce the obvious noise influence on the detector. The choice of size of the Gaussian convolution kernel will affect the performance of the Canny dettor. The large the size, the lower sensitivity of the detector to noise, but the positioning error of edge detection will also increase.

SIFT: Use Gaussian kernel to smooth the image, because Gaussian convolution kernel is the only linear kernel that realizes scale transformation. The SIFT processing method has a splitting process of Gaussion pyramid. After the image Gaussian pyramid is created, the adjacent layers in each group can be obtained by subtraction DOG, the difference of Gaussian, which is the premise of detecting the extreme points of the image in later stage.

2. Sub-pixel extraction of edges:

There's a kind of sub-pixel extraction algorithm call Zerike moment. it's a integral operator.

For the ideal model of zernile moments in edge detection.



L enclosed by the unit circle represent the ideal edge

$$Z'_{n,m} = Z_{n,m} \exp(-jm\phi)$$

$Z'_{n,m}$ is the zernike moment after the image rotates clockwise around the origin. Use the rotation invariana of zernike moments, can calculate the three important parameters k, l, ϕ, then to achieve the precise positioning of the edge

$$\phi = \tan^{-1}\left(\frac{Im(Z_{1,1})}{Re(Z_{1,1})}\right)$$

$$l = \frac{Z'_{2,0}}{Z'_{1,1}} = \frac{Z_{2,0}}{Z_{1,1}\exp(-j\phi)}$$

$$k = \frac{3Z'_{1,1}}{2(1-l^2)^{\frac{3}{2}}} = \frac{3Z_{1,1}\exp(-j\phi)}{2(1-l^2)^{\frac{3}{2}}}$$

to $M_{n,m}(i,j) = \iint_{S_{i,j}\cap C} V^*_{n,m} dx dy$

for keypoints: first, take the Sub-pixel Localization

$$D(x) = D + \frac{\partial D^T}{\partial x} x + \frac{1}{2} x^T \frac{\partial^2 D}{\partial x^2} x$$

$$\hat{x} = -\frac{\partial^2 D^{-1}}{\partial x^2}\frac{\partial D}{\partial x}$$ to get location in terms of $(x, y, 6)$

the Filter low contrast point

$$D(\hat{x}) = D + \frac{1}{2}\frac{\partial D^T}{\partial x}\hat{x}$$

compute Gradient for each blurred image

$$m(x,y) = \sqrt{(L(x+1,y)-L(x-1,y))^2 + (L(x,y+1)-L(x,y-1))^2}$$

$$\theta(x,y) = \tan^{-1}((L(x,y+1)-L(x,y-1))/(L(x+1,y)-L(x-1,y)))$$

Then get key points by create Histogram with 36 bins for orientation. Weight each points with Gaussian window of 1.56. Create keypoint for all peaks with value >=0.8 max bin

3. The essence of the SIFT algorithm can be ~~to~~ conclude to the problem of finding ~~feature points~~ key points on different scale spaces. The main steps of SIFT algorithm can seperate to:

1. Extract key points
2. Add detailed local features to key points
3. Through the pairwise comparison of the feature points of both parties — the key points of the feature vector, find art several pairs of future points that match each other. Then establish the correspondence between the object scenes.

The key points that need to searched in SIFT are some very prominent points that will not disappear due to changes in care conditions. For example, corner points, edge points, bright spot in dark areas and dark spots in bright areas. Since there are the same scenes in the two images, a certain method is used to extract their respective stable points. It finds k nearest neighbours of each feature and individually compares each feature of the new things with other feature and ~~sea~~ forms full set of matches and keypoints that agree to the object location, scale and illumination.

4. The rotation invariance ~~of~~ extension of the LBP is known as LBP-HF.

P is the number of sampling points and R is the radius

$$(x_p, y_p) = \left(x + R\cos\left(\frac{2\pi p}{P}\right), y - R\sin\left(\frac{2\pi p}{P}\right)\right)$$

$$LBP_{P,R}(x,y) = \sum_{p=0}^{P-1} s(f(x,y) - f(x_p, y_p))2^p$$

Let $U_P(n,r)$ be a uniform LBP pattern, with n being the number of 1's in pattern and r the rotation

$U_P(n,r)$, due to rotational symmetry, $P+1$ steps will yield same pattern, so replace r with $k+r \bmod P$

$$h_\theta(U_P(n, r+k)) = U_P(n,r)$$

$$H(n,u) = \sum_{r=0}^{P-1} h(U_P(n,r))e^{-i2\pi ur/P}$$

$\because$ DFT, $\therefore h'(U_P(n,r)) = h(U_P)(n, r-k)$

$$H'(n,u) = H(n,u)e^{-i2\pi uk/P}$$

$$\overline{H'(n,u)} = \overline{H(n,u)e^{-i2\pi ur/P}} = \overline{H(n,u)}\, e^{i2\pi ur/P}.$$

$$H'(n_1,u)\overline{H'(n_2,u)} = H(n_1,u)e^{-i2\pi ur/P}\,\overline{H(n_2,u)}\,e^{i2\pi ur/P} = H(n_1,u)\overline{H_2(n_2,u)}$$

$$\therefore LBP^{u2}\text{-}HF(n_1, n_2, u) = H(n_1,u)\overline{H(n_2,u)}$$

9	3	1
8	5	2
7	4	6

$\xrightarrow{\text{threshold}}$

1	0	0
1		0
1	0	1

$$LBP^{ri}_{P,R} = \min\{ROR(LBP_{P,R}, i) \mid i = 0, 1, \cdots, P-1\}$$

$\downarrow$ Rotate 90° clock wise

min: 00010111

7	8	9
4	5	3
6	2	1

$\longrightarrow$

1	1	1
0	●	0
1	0	0