Lecture Node for Edge detection

Yujie Song
Sun Yat-Sen University
No. 132, Huan East Road, Xuekouwai City, Panyu District, Guangzhou City
1638846513@qq.com

Abstract

In the field of Digital Image Processing, the various features of images are often the targets we pay attention to. The edge of the image, as the curve of some pixel points with the largest image gradient, naturally contains a lot of effective information of the image. In this report, I investigated some technologies of image edge detection, from the most basic algorithm such as Roberts more than 50 years ago, to the most common Canny method now, and then to the most cutting-edge neural network algorithm. Finally, I have carried out some experiments on other algorithms except neural network algorithm, and analyzed some advantages and disadvantages of various algorithms.

1. Introduction

Image features usually represent important properties of images, and image edge, one of the most basic features of images, has been widely concerned by researchers since the end of the 20th century. People try to extract the edge from the image while ignoring the color and other features. This research field is called "Edge detection", and the image extracted through edge detection is called the edge image.

Edge detection is the first step in Digital Image Processing usually. For example, in image classification, we can obtain edge images through edge detection first, and then using other technology (such as using MLP classifier) to classify these edge images basing the information contained in them. Therefore, the edge detection results will directly affect the subsequent image processing process and the final result.

2. Pre-knowledge

Digital image: using one or a group of data (such as gray value, RGB, etc.) to represent a pixel of an image, and the whole image is represented as a two-dimensional matrix composed of these data.

Grayscale image: the image represented by the grayscale value, whose range is [0, 255], represents the black-and-white degree of the pixel (0 means completely black, 255 means completely white). Since edge extraction is generally operated on grayscale images, it is necessary to convert RGB trichromatic images into grayscale images:

$$Gray_{x,y} = Red_{x,y} \times 0.299 + Green_{x,y} \times 0.587 + Blue_{x,y} \times 0.114$$

While Gray, Red, Green, Blue represent the Gray value and RGB value of a pixel respectively.

Filtering: an operation equivalent to convolution nearly. A filtering kernel (convolution kernel) is used to convolve the operation image (two-dimensional matrix) to get the new two-dimensional matrix after convolution:

$$NewGray_{x,y} = \sum_{i=x-r}^{x+r} \sum_{j=y-r}^{y+r} K_{i,j} \times Gray_{i,j}$$

While NewGray, Gray, K respectively represent the Gray value after filtering, the Gray value before filtering and the filter kernel, and r represents the radius of the filter kernel. In the process of filtering, 0 is usually added to the outer ring of the image to achieve the matrix size unchanged after filtering.

Edge: refers to the mutability and discontinuity of the local area of the image, which is embodied in the great change of the value of the adjacent position of the gray image.

Gradient: in the field of Digital Image Processing, gradient represents the change intensity of a pixel in the image along the X and Y directions (i.e. the partial derivative value) :

$$\nabla f_{x,y} = [g_x, g_y] = \left[\frac{\partial f_{x,y}}{\partial x}, \frac{\partial f_{x,y}}{\partial y}\right]$$

For the discrete grayscale image, the partial derivative of X and Y can be simplified to the difference of adjacent positions:

$$\left[\frac{\partial f_{x,y}}{\partial x}, \frac{\partial f_{x,y}}{\partial y}\right] = \left[\frac{f_{x+\Delta x,y} - f_{x,y}}{\Delta x}, \frac{f_{x,y+\Delta y} - f_{x,y}}{\Delta y}\right] \approx \left[f_{x+1,y} - f_{x,y}, f_{x,y+1} - f_{x,y}\right]$$

Therefore, the gradient of the image can be simplified as the difference between the gray values of adjacent pixels in the X and Y directions. In practical operation, simple addition and subtraction can also be used to replace the complex derivative operation.

3. Edge detection operator

3.1. The basic idea

The edge features are reflected in a local area and the gray values of adjacent locations vary greatly. Therefore, by detecting the gradient value of each pixel in the x and y directions, and judging whether the difference in gray value of its neighboring pixels is too large, it can be explained whether this pixel is an edge point. The commonly used method in the industry is to filter the image by constructing the corresponding filter kernel, and finally obtain the gradient value of each pixel of the image in the X and Y directions.

For example, the filter kernels (also called filter operators) that calculate the gradient values of the pixels in the x and y directions are as follows:

$$K_x = \begin{bmatrix} -1 & 1 \end{bmatrix}, \quad K_y = \begin{bmatrix} -1 \\ 1 \end{bmatrix}$$

After filtering the original image in the X and Y directions, two edge images are obtained, and then the two edge images are merged to obtain the final edge image:

Figure 1: filter operations on the image in the X and Y directions, and finally merge them.

For this basic filter operator, we can simply judge whether the pixel is an edge point based on the characteristics of the adjacent pixels in the X and Y directions of the pixel. A more comprehensive way is judging the characteristics of a pixel in a neighborhood centered on the pixel, which derives other filter operators.

3.2. Derivative filter operators

Roberts operator, proposed in 1963, is a relatively simple edge filter operator, using the local difference to find the edge of the image. The specific method is: using the difference between two adjacent pixels in the diagonal direction to approximate the gradient magnitude.

$$K_x = \begin{bmatrix} 1 & 0 \\ 0 & -1 \end{bmatrix}, \quad K_y = \begin{bmatrix} 0 & -1 \\ 1 & 0 \end{bmatrix}$$

It can be seen from the filter operator that this algorithm focuses on the characteristics of pixels in the diagonal direction when judging edge points, so it can well judge the edges in the diagonal direction.

Prewitt operator uses a 3*3 edge filter operator to perform operations on the pixels in the eight neighborhoods of the pixel. The recognition neighborhood is larger, and the features which can be recognized will be more:

$$K_x = \begin{bmatrix} -1 & 0 & 1 \\ -1 & 0 & 1 \\ -1 & 0 & 1 \end{bmatrix}, \quad K_y = \begin{bmatrix} -1 & -1 & -1 \\ 0 & 0 & 0 \\ 1 & 1 & 1 \end{bmatrix}$$

It calculates both horizontal/vertical adjacent vectors and diagonal vectors, so its feature extraction and recognition capabilities will be stronger.

Sobel operator, on the basis of Prewitt operator, adds the concept of weight. It believes that the distance between adjacent points has a different impact on the current pixel. The closer the pixel point corresponds to the current pixel, the greater the impact. Thus, the effect of prominent edge is better realized.

$$K_x = \begin{bmatrix} -1 & 0 & 1 \\ -2 & 0 & 2 \\ -1 & 0 & 1 \end{bmatrix}, \quad K_y = \begin{bmatrix} -1 & -2 & -1 \\ 0 & 0 & 0 \\ 1 & 2 & 1 \end{bmatrix}$$

Laplacian operator, unlike the above operators, do not break them down into X and Y operators, but combine them into one operator that can compute all four or eight directions at once. There are two forms of Laplacian operator, which are four neighborhood operator and eight neighborhood operator. The four neighborhood operator is the gradient of the four directions of the center pixel in the neighborhood, while the eight neighborhood operator is the gradient of the eight directions.

$$K_x = \begin{bmatrix} 0 & -1 & 0 \\ -1 & 4 & -1 \\ 0 & -1 & 0 \end{bmatrix}, \quad K_y = \begin{bmatrix} -1 & -1 & -1 \\ -1 & 8 & -1 \\ -1 & -1 & -1 \end{bmatrix}$$

3.3. Characteristics of various operators

Roberts Operator: Because the filter kernel of 2*2 is adopted, it is particularly dependent on the information of local pixels. Although it has the advantage of accurate edge positioning, it is particularly easy to be interfered by nearby noises. Therefore, it is only suitable for processing images of high quality, or denoising images in advance.

Prewitt Operator: The biggest difference between it and Roberts is the use of a 3*3 filtering kernel, which can recognize gradients in both vertical and horizontal directions as well as diagonals. As the size of filtering kernel increases, the probability of noise interference increases correspondently, but the degree of single noise interference is lower than that of Roberts algorithm.

Sobel Operator: Similar to the Prewitt algorithm, but with the addition of the concept of position weight, the influence of the four adjacent pixels on the horizontal and vertical lines on the pixel is increased, while the influence of the four adjacent pixels on the diagonal is correspondingly lower. A slight modification of the operator reduces the influence of the diagonal noise, but the influence of the noise in the horizontal and vertical directions becomes correspondingly larger. In actual applications, the effect is generally better than the Prewitt algorithm.

Laplacian Operator: An operator similar to Gaussian smoothing. The greatest advantage is that it combines gradient recognition in the X and Y directions and requires only one filtering operation, nearly doubling the speed of edge detection. However, since the grayscale value of the pixel itself is considered for calculation, its own pixel value can affect the recognition of the edge, and the probability of being interfered by noise increases accordingly.

4. Canny method

Canny edge detection method, proposed by John F. Canny in 1986, is the most widely used edge detection algorithm at present. Compared with the above algorithm, the accuracy of this method is much higher, and the extracted edge image is also more vivid.

The specific steps of Canny method are as follows:

4.1. Gaussian filter

Due to the inevitable occurrence of some noise in the image, the noise points will be easily identified as edge points if the image is not denoised in advance. Therefore, the Canny algorithm first uses the most commonly used Gaussian filter to denoise the image.

The basic principle of Gaussian filtering is as follows: according to the gray value of the pixel point to be filtered and its neighborhood points according to the parameter rule generated by the "Gaussian distribution formula" to carry out the weighted average, which can effectively filter the high-frequency noise superimposed in the ideal image (namely the pixel point with large gradient).

Two-dimensional Gaussian formula:

$$G_{x,y} = \frac{1}{2\pi\sigma^2} e^{-\frac{x^2+y^2}{2\sigma^2}}$$

Common Gaussian filter kernel has 3*3 and 5*5 two kinds:

$$K_3 = \frac{1}{16} \times \begin{bmatrix} 1 & 2 & 1 \\ 2 & 4 & 2 \\ 1 & 2 & 1 \end{bmatrix}, \quad K_5 = \frac{1}{273} \begin{bmatrix} 1 & 4 & 7 & 4 & 1 \\ 4 & 16 & 26 & 16 & 4 \\ 7 & 26 & 41 & 26 & 7 \\ 4 & 16 & 26 & 16 & 4 \\ 1 & 4 & 7 & 4 & 1 \end{bmatrix}$$

4.2. Calculate Gradient image and Angle image

After the Gaussian filter is carried out in the first step of Canny method, the edge filter operator is also used to calculate the preliminary edge image. Specifically, the edge filtering operator of Canny algorithm is Sobel operator.

$$G_x = image * K_x, \quad G_y = image * K_y$$

Where image, G_x, G_y are the original image, the preliminary edge image in X and Y directions respectively, and * is the filtering (convolution) operation.

Then calculate the gradient amplitude and gradient angle for the pixels at the same position of G_x, G_y. The calculation formula is as follows:

$$G = \sqrt{G_x^2 + G_y^2}, \quad \theta = arctan\left(\frac{G_y}{G_x}\right)$$

It can be seen from the formula that the gradient image and angle image actually describe the modulus and direction angle of each pixel gradient of the image.

4.3. Non-Maximum Suppression

Roberts and other algorithms introduced in 3.2 all have the same problems as edge width, weak edge interference and so on. The same is true for the preliminary edge image calculated in 4.2. Therefore, the further operation of Canny method is to use the non-maximum suppression method to process the previously generated gradient image and angle image. The basic algorithm is described as follows:

1. Compare the gradient amplitude of the current pixel with the two nearest pixels along the positive and negative gradient directions.

2. If the gradient amplitude of the current pixel is greater than the other two pixels, the pixel point will be retained as the edge point, otherwise will be suppressed (that is, it will not be considered as the edge point).

In the process of selecting the two nearest pixels in the gradient direction, if the gradient angle is an integer multiple of 45 degrees, the two nearest pixels are two of the pixels in the eight neighborhood (Figure 2 left). And if the gradient angle is not an integer multiple of 45 degrees, then create a straight line passing through the pixel with the gradient angle (Figure 2 right), it will have two intersections with the square boundary of the eight neighborhood (p1, p2), at this time, the gradient amplitudes of the "closest two pixels" use the gradient amplitudes formed by linear interpolation of two adjacent points (p3-p4, p5-p6):

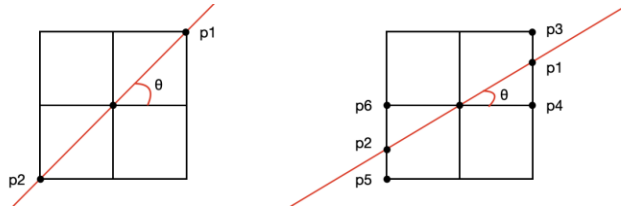

Figure 2: A sample diagram of non-maximum suppression.

The calculation formula of linear interpolation is as follows:

$$G_{p1} = \tan\theta * G_{p3} + (1 - \tan\theta) * G_{p4}$$
$$G_{p2} = \tan\theta * G_{p5} + (1 - \tan\theta) * G_{p6}$$

4.4. Dual threshold method

After the above three-step operation, there will still be some false edges in the image, that is, some pixels with relatively small gradient amplitude. In the last step of the Canny method, the dual threshold method is used to determine the final edge. The basic algorithm is described as:

1. Select two thresholds, pixels with gradient amplitude less than the low threshold are considered false edges, and pixels with gradient amplitude greater than the high threshold are considered strong edges.

2. For a pixel with a gradient amplitude in the middle, if it is connected to a strong edge pixel (specifically, there are strong edge pixels in the eight neighborhood), it is also regarded as a strong edge point, otherwise it is regarded as a false edge.

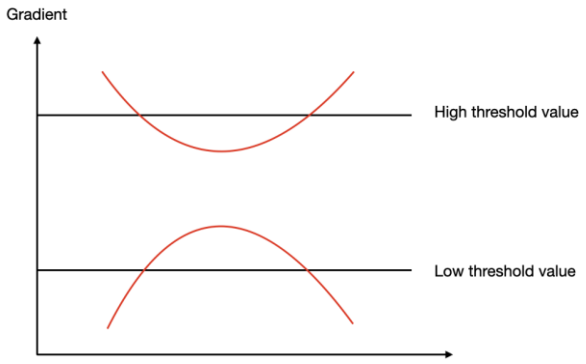

Figure 3: A sample diagram of the dual threshold method. The pixels in the upper red line with intermediate gradient amplitudes are also regarded as strong edge points, and the pixels in the lower red line with intermediate gradient amplitudes are regarded as false edge points.

5. Experiment

This experiment is based on the Python cv2 library, which encapsulates the filtering operation and the complete Canny method, simplifying the operation process of the experiment.

5.1. Python correlation function

Filtering: The cv2 library provides the function cv2.filter2D to implement filtering operation. The prototype of the function is as follows:

dst = cv2.filter2D(src, ddepth, kernel[, dst[, anchor[, delta[, borderType]]]])

Commonly used parameters and the result are as follows:

- dst: image filtering results (a two-dimensional integer list)
- src: The original image (the processing in this experiment is a grayscale image, a two-dimensional integer list)
- ddepth: Image depth, containing information about the type of data stored in the image (`CV2.CV_16s` means grayscale diagram)
- kernel: filter operator, i.e., Roberts and other operators mentioned above (a two-dimensional integer list)

After the filtering operation is finished, we need to use cv2.convertScaleabs function to turn it back to the original `uint8` form, otherwise the image will not be displayed. The prototype of the function is as follows:

dst = cv2.convertScaleAbs(src[, dst[, alpha[, beta]]])

Commonly used parameters and the result are as follows:

- dst: Image data type conversion results (a two-dimensional integer list)
- src: image filtering results (a two-dimensional integer list)

As Roberts and other algorithms filter X and Y directions respectively, they need to be merged at last. In this case, the function cv2.addweighted is adopted. The prototype of the function is as follows:

dst = cv2.addWeighted(src1, alpha, src2, beta, gamma[, dst[, dtype]])

Commonly used parameters and the result are as follows:

- dst: Image merge results (a two-dimensional integer list)
- src1: X direction filtering result image (a two-dimensional integer list)
- alpha: The weight of the X direction filtering result image
- src2: Y direction filtering result image (a two-dimensional integer list)
- beta: The weight of the Y direction filtering result image
- gamma: The offset value of the final image

Canny method: In Python cv2 library, cv2.GaussianBlur function encapsulates the steps of Gaussian filtering alone, while cv2.Canny function encapsulates other steps in Canny algorithm except Gaussian filtering. The function prototypes are as follows:

dst = cv2.GaussianBlur(src, ksize, sigmaX[, dst[, sigmaY[, borderType]]])

Commonly used parameters and the result are as follows:

- dst: image Gaussian filtering results (a two-dimensional integer list)
- src: the original image (the processing in this experiment is a grayscale image, a two-dimensional integer list)
- ksize: size of Gaussian filter kernel (a tuple containing two integers)
- sigmaX: sigma in Gaussian filter equation

edge = cv2.Canny(image, threshold1, threshold2[, edges[, apertureSize[, L2gradient]]])

Commonly used parameters and the result are as follows:

- edge: Canny edges result, returns a binary graph (a two-dimensional integer list)
- image: image Gaussian filtering results (a two-dimensional integer list)
- threshold1: the low threshold in the dual threshold method
- threshold2: the high threshold in the dual threshold method

5.2. Experimental results of edge detection algorithms

We use Roberts, Prewitt, Sobel, Laplacian and Canny method at the same time to extract the edges of the same image (where Laplacian method uses eight neighborhood operator, and the high and low thresholds of Canny method are 150 and 50 respectively). The results are shown in Figure 4:

Figure 4: Experimental results of various edge detection algorithms.

It can be seen that the edge image extracted by Roberts algorithm is dark, and only the outer contour of the rhino is relatively bright. The edge image extracted by Sobel algorithm is brighter, and the markings and wrinkles of the rhino body can be extracted clearly. The results of Prewitt algorithm and Laplacian algorithm are similar, while Prewitt algorithm can extract the outline of rhino more obviously. However, the edge image extracted by Canny method only contains strong edges and all edges have the same intensity, ignoring the feature of edge intensity, so that from a visual point of view, the edge image extracted by Canny method is not as beautiful as Sobel's edge image. But it shows the markings of rhino and the texture of trees more clearly. Generally speaking, each edge image has its own advantages.

Then we do the same for another image, and the results are shown in Figure 5. It can be seen that the result of

Roberts algorithm is still the fuzziest; The result of Sobel algorithm is the best, the extracted edge is clear, and the white waves in the center of the image can be extracted. The result of Prewitt algorithm is the second, because the white wave edge extracted is more fuzzy than that of Sobel algorithm. However, Laplacian algorithm can hardly extract white waves (due to the fuzzy nature of its filter operator). For Canny method, its processing results for island stone texture and mangrove texture are almost the same, which will actually lead to the disappearance of its semantic features. Moreover, the appearance and feeling of the processed edge image is not as good as Sobel algorithm.

What is worth mentioning is that for the bottom edge of the island in the red box of the original image(Figure 6), none of these algorithms can extract its edge.

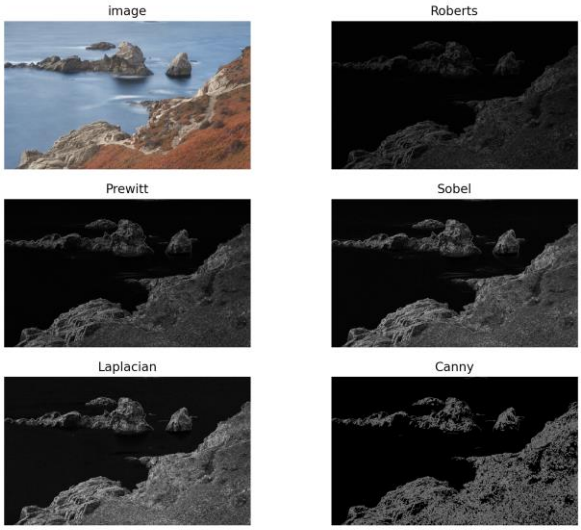

Figure 5: Another experimental results of various edge detection algorithms.

Figure 6: The hard-to-extract island area (red box area) and the white wave area at the center (yellow box area)

5.3. Experimental results of edge detection algorithms

We test the Canny algorithm separately. The high threshold was maintained at 150, and the low threshold was set to 0, 50, and 100. The results of the tests are shown in Figure 7:

Figure 7: Experimental results of Canny method at different thresholds.

It can be seen that a unique advantage of Canny method is that it combines the dual threshold method, so it can obtain different edge images by adjusting the high and low thresholds. Therefore, in solving practical problems, specific high and low thresholds can be set according to the edge density required by practical problems, so as to obtain more ideal edge images.

The same experimental test is carried out on the landscape map above, and the results are shown in Figure 8. By the experimental results it can be seen that when the high and low threshold value is set to 50 and 0 respectively, Canny method can extract the bottom edge of the island in the red box(Figure 6), and the center of the figure white sea area, but correspondingly the island stone and mangrove's texture are very closely, the results almost see no difference. When the high threshold is raised to 100, the bottom edge of the island in the red box and the white wave area in the center of the image are almost completely undetectable. When the high and low thresholds were set to 200 and 100 respectively, the extracted results of island stone and mangrove's texture were significantly different. The island stone texture showed a strip fold shape, while the mangrove texture was similar to the scattered point shape. When the high and low thresholds are set to 300 and 200 respectively, Canny method can almost only extract the outer edge of the island. Therefore, it can also reflect the diversity of results of Canny method, which will be more conducive to solving practical problems.

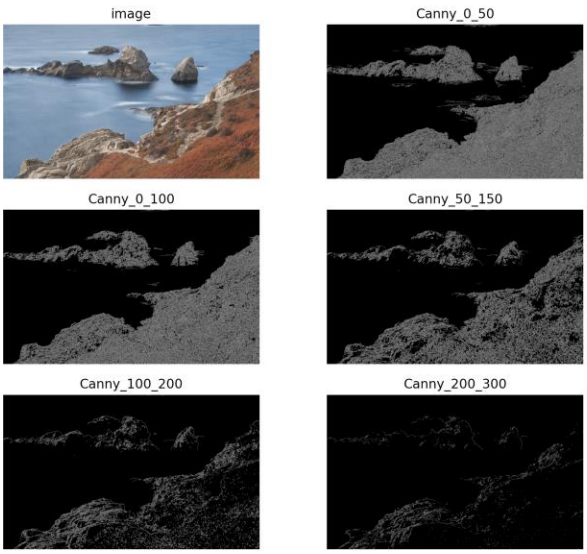

Figure 8: Another experimental results of Canny algorithm at different thresholds

5.4. Time test of various algorithms

We test the running time of various edge extraction algorithms, processing the same image at the same time. The results are shown in Figure 9:

```
RobertsTime:      0.0207 s
PrewittTime:      0.0245 s
SobelTime:        0.0225 s
LaplacianTime:    0.0108 s
CannyTime:        0.0121 s
```

Figure 9: The running time of various algorithms

It can be seen that the running time of Roberts, Prewitt and Sobel algorithms is above 0.02s, while the running time of Laplacian algorithm and Canny algorithm is about 0.01s. The results are also the same after many experiments. Roberts, Prewitt and Sobel algorithms have almost the same time (Roberts is slightly shorter). In principle, the main reason is that they all filter twice, while Roberts' filter kernel is slightly smaller (2*2, Prewitt and Sobel are both 3*3). The Laplacian algorithm takes about half the time of the previous three algorithms, mainly because it only performs one filtering operation; The Canny algorithm also took less time to run than the previous three algorithms, which surprised me. Because in principle, Canny algorithm first carries out a Gaussian filtering operation, then carries out two Sobel filtering operations, and then has non-maximum suppression and dual threshold operation, and in the online reference of various introductions of Canny method, it is also considered that Canny method is more complex and consumes a lot of time. However, it can be seen from the experiment that the total time of Canny method is shorter than that of Sobel algorithm. For the time being, I can only think that the time optimization of Canny algorithm encapsulated by Python's CV2 library is better.

6. The frontier technology of image edge detection

Research on the field of image edge detection is mainly concentrated in the late last century. After the Canny method was proposed, the industry generally believes that it has been able to meet most of the requirements of edge detection. Therefore, in a period of time, the field of edge detection basically has not changed much. After deep learning technology, especially neural network technology, has been widely used, new research directions have emerged in the field of image edge detection, which can be classified into two categories:

1. Using the edge detection technology mentioned above to preprocess the image dataset and assists the subsequent algorithm to achieve better experimental results.
2. Using deep learning technology to realize image edge detection.

Among them, the first kind is simply preprocessed by using the original technology, while the second kind often presents new requirements and new technologies. For example:

- We only want to identify the outline of the rhino, trees and pillars in the image above, not the markings of the rhino and trees.
- We only want to identify the outline of the rhino's tail, not anything else.

To put it simply, we only extract some edges worthy of our attention, and ignore some edges that we think are of little information value despite the large gradient. At this point, all the edge extraction techniques mentioned above almost completely fail, and the idea needed is to add the semantic information of the image for processing. At this time, perhaps only machine learning technology represented by CNN can meet this requirement. For example, [1] Richer Convolutional Features for Edge Detection, a paper of CVPR 2017, it designed the RCF network based on VGG16 convolutional neural network for edge detection, which achieved a good effect:

Figure 10: Some examples of RCF. From top to bottom: BSDS500, NYUD, Multicue-Boundary, and Multicue-Edge (datasets). From left to right: origin image, ground truth, RCF edge map, origin image, ground truth, and RCF edge map.

7. Conclusion

This lecture node mainly introduces a variety of edge detection methods based on image gradient, from the most basic gray difference to Sobel and other operators, and then a complete introduction of Canny edge detection method, use these algorithms to do several experiments, observe the edge images generated by them, analyze the differences between them. Finally, we investigate the current advanced edge detection method: deep neural network. Through the whole learning process, from understanding the principle, experimental implementation, to research on the recent papers of top conferences, I also have a broad understanding of the field of image edge detection, and here I would like to summarize the advantages and disadvantages of the above algorithm:

- Roberts and other algorithms. Advantages: only applied to the filtering operation, simple operation, fast detection speed, detection results can also meet most of the needs. Disadvantages: detection results are often inferior to Canny method and deep neural network algorithm.
- Canny method. Advantages: edge detection results are fine, can set a threshold to filter the weak edge. Disadvantages: the detection procedure is complicated and the speed is relatively slow, so it is difficult to be applied to real-time monitoring.
- Deep neural network algorithm. Advantages: can combine the semantic information of the image to carry out specific edge detection, can customize the detection and undetection feature edge. Disadvantages: need a large number of artificially annotated datasets, network training time is very long, currently only applicable to the academic field, temporarily difficult to be popularized in daily life.

References

[1] Liu Yun,Cheng Ming-Ming,Hu Xiaowei,Bian Jia-Wang,Zhang Le,Bai Xiang,Tang Jinhui. Richer Convolutional Features for Edge Detection.[J]. IEEE transactions on pattern analysis and machine intelligence,2019,41(8).