Canny edge detection based on Open CV

Zhao Xu[1], Xu Baojie, Wu Guoxin[1],
1.Key Laboratory of Modern Measurement & Control Technology Ministry of Education,Beijing Information Science & Technology University,Beijing 100192, China
Email: *qingsezhuifeng@163.com*

Abstract–*As a traditional edge detection algorithm, Canny operator is widely used and improved. Canny adopts hysteresis threshold, the different values of threshold have great influence on the detection result, but the number can not reflect the detection result intuitively. Four of algorithms are introduced through the use of Open CV programming, the programming can visually display the detection results under different thresholds, which is convenient for image edge detection.*
Keywords–*Edge detection; Canny operator; hysteresis threshold; Open CV.*

I. INTRODUCTION

The edge is the most basic feature of an image, which refers to the set of pixels that have a sudden change in the gray level. Edge detection is a basic method to recognize and segment the edges of images based on gray discontinuous points. The Canny operator is a multiply-scale edge detection algorithm proposed by John F.Canny in 1986, the goal is to find an optimal edge detection algorithm, which is widely used in the field of image processing, and are constantly improved and innovated. The Canny operator chooses the edge points by threshold approach, so the threshold value has an important influence on the results of image edge detection [1-3]. How to obtain an accurate threshold and determine the optimal threshold is a key problem, however the influence of different thresholds can not be visualized by numbers on the detection results[4].

Open CV is a great tool for image processing with strong function, it has a sufficient computer vision library to deal with graphic problems. Programming with Open CV algorithm library can display different results under different thresholds, which is convenient for finding the optimal threshold, and is propitious to image edge detection[5-6].

II. CANNY EDGE DETECTION

Using Canny edge detection, there are usually several steps[7]:

Step 1: Denoise image before detecting edge of the image, and usually use the Gauss smoothing filter to reduce noise, according to (1)

$$G(\mathrm{x}, y)=\frac{1}{2\pi\sigma^2}e^{-\frac{x^2+y^2}{2\sigma^2}} \tag{1}$$

Step 2: Calculate the gradient amplitude and direction, usually the gradient direction takes the 4 angles - 0, 45, 90, and 135 degrees,according to (2), (3)

$$G=\sqrt{G_{\mathrm{x}}^2+G_y^2} \tag{2}$$

$$\theta=\arctan\left(\frac{G_y}{G_x}\right) \tag{3}$$

G_{x} , G_{y} as a pair of convolution array.

Step 3: Non maximum suppression， which eliminates non edge pixels, leaving only a few fine lines.

Step 4: Select the hysteresis threshold, hysteresis threshold needs two thresholds which retain or exclude pixels to select the edge.

If the amplitude of the pixel position is higher than the high threshold, the pixel is reserved as an edge pixel.

If the amplitude of the pixel position is less than the high threshold, the pixel is excluded.

If the amplitude of the pixel position is between the two thresholds, the pixel is reserved only when connected to a pixel higher than the high threshold.

III. OPEN CV IMAGE PROCESSING PROGRAMMING

A. Introduction of Open CV

Open CV is a cross platform computer vision library with an open source which began as a research project at Intel in 1998.5 and has been available since 2000.Open CV is aimed at providing the tool needed to solve computer-vision problems, it can be prepared by C and C++ language, contains a mix of low-level image-processing functions and high-level algorithms [8-9].

B. Canny () function

void Canny(InputArray image, OutputArray edges,double threshold1, double threshold2,int apertureSize = 3, bool L2gradient = false);

The canny function including six parameters, the first parameter for the input image, second parameters for the output image, third parameters for the first hysteresis threshold, fourth parameters for the second hysteresis threshold, the fifth parameter is Sobel

978-1-5090-5035-2/17/$31.00 ©2017 IEEE

operator which is applied to the pore size, the sixth parameter is the identification of the image gradient magnitude calculation.

C. CreateTrackbar() function

createTrackbar(const String& trackbarname, const String& winname , int* value, int count,TrackbarCallback onChange = 0,void* userdata = 0);

The createTrackbar function creates a slider which can adjust numbers.

D. GetTickCount()/getTickFrequency() function

```
double t = (double)getTickCount();
do something ...
t = ((double)getTickCount()-t)/getTickFrequency();
```

The getTickCount/getTickFrequency function is two timing functions, and timing is important in image processing applications[10].

E. Open CV programming

Programming uses two createTrackbar functions to control the value of high and low threshold respectively, and obtains the result of image edge detection via high and low threshold callback canny function. The kernel code is as follows:

```
createTrackbar("lowthreshold", "dstImage", &lowthreshold, 120, on_canny);
createTrackbar("hignthreshold", "dstImage", &hignthreshold, 120, on_canny);
...
void on_canny(int, void*)
{
blur(grayimage, cannyimage, Size(3, 3));
Canny(cannyimage, cannyimage, lowthreshold, hignthreshold, 3);
dstimage = Scalar::all(0);
srcimage.copyTo(dstimage, cannyimage);
imshow("dstImage", dstimage);
}
```

IV. GPU ACCELERATION WITH OPEN CV

Many computer-vision projects must be executed in real time, which implies that the processing if a single frame should complete within 30-40milliseconds, which is a very challenging requirement[11-12].

The first GPUs were fixed-function pipelines specialized for accelerated drawing of shapes on a computer display, however the programmers make the fixed-function GPUs partially programmers by adding shaders, which was a big step forward.The GPU became a useful tool for image processing and computer-vision tasks[13-15].

GPU accelerate can be used with Open CV , before programming the GPU program,we should install CUDA and check the presence of the GPU device in the computer.The code reads as follows:

```
int num_devices = gpu::getCudaEnabledDeviceCount();
if (num_devices <= 0) {
    std::cout << "there is no device." << std::endl;
    return -1;
}
int enable_device_id = -1;
for (int i = 0; i < num_devices; i++) {
    gpu::DeviceInfo dev_info(i);
    if (dev_info.isCompatible()) {
        enable_device_id = i;
    }
}
if (enable_device_id < 0) {
    std::cout << "GPU module is not built for GPU" << std::endl;
    return -1;
}
```

If the program check that the computer have the GPU devices,the GPU accelerator can be used by Open CV.The code is as follows:

```
gpu::GpuMat image_gpu, edge_image;
image_gpu.upload(gray);
gpu::blur(image_gpu, edge_image, Size(3, 3));
gpu::Canny(image_gpu, edge_image, 61, 95);
edge_image.download(edge);
dst = Scalar::all(256);
image1.copyTo(dst, edge);
    imshow("result", dst);
```

V.EXPERIMENTAL RESULTS AND ANALYSIS

A. Experimental results

To verify that the program is valid, enter an original picture called "orange" for testing. The original drawing and test results are shown in Fig. 1. 2.and 3.

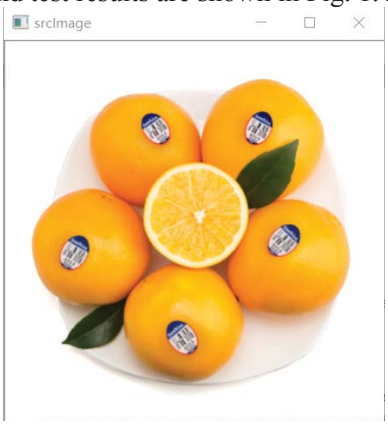

Fig. 1 Primitive diagram

Fig. 2 The running time of the algorithm

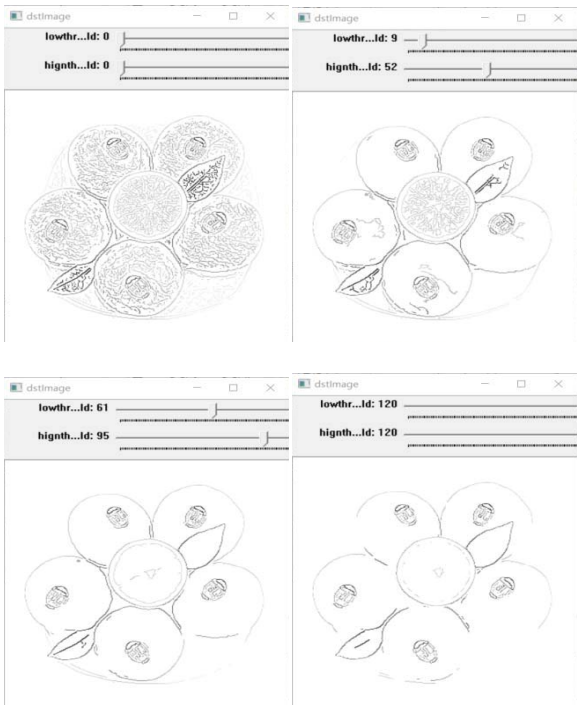

Fig. 3 Detection results under different thresholds

According to fig.3, we can found that there is a good result when low threshold is about 61 and high threshold is about 95. so, these two values will be applied in the program without GPU and program with GPU acceleration individually to view the effect of GPU acceleration. the test results are shown in Figure 4,5 and 6 below.

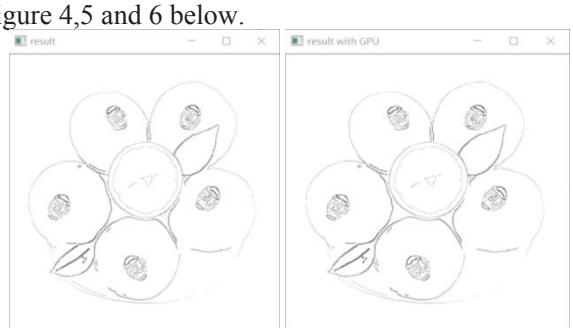

Fig. 4 results with GPU and without GPU

Fig. 5 the use time of result without GPU

Fig. 6 the use time of result with GPU acceleration

B.Analysis

According to Fig.2 and Fig.3 analysis: the operation speed is fast, only sixteen ms will be detected through Open CV programming. From the test results, different thresholds have great influence on the test results. when the high and low threshold is double zero, the detection results have many false edges, as the threshold increases, we can see that the pseudo edge decreases, but when the threshold is the maximum value, some effective edge has also been removed. Therefore, we can intuitively see the influence of different thresholds on the detection results via using Canny edge detection based on Open CV, so as to find and verify the optimal threshold.

According to Fig.4, Fig.5 and Fig.6, two images processed using GPU acceleration and without GPU are the same, but they take different time, the program which apply the GPU acceleration uses the time less than the program without the use of the GPU , but the accelerating effect is not obvious, the main reason is that the consumption of time which the program spent on displaying picture much more than the edge detection of the time consumption, so the optimization algorithm did not bring better acceleration effect.When the algorithm consumes much more time

than the picture display and memory transfer, the acceleration effect becomes apparent.

VI.CONCLUSION

With the development of computer vision technology, Open CV offers a number of algorithms for image processing and can use GPU chips to speed up programs. Open CV is a good image processing tool, it is worth using and popularizing. This paper proposes a canny edge detection based on Open CV and introduces four algorithms including canny function, GPU acceleration and so on. The immediacy and convenience can be seen from the test results which is provided by image edge detection based on Open CV, and is propitious to image edge detection.

REFERENCE

[1] J.Canny. A computational approach to edge detection. IEEE Transactions on Pattern Analysis and Machine Intelligence, 1986: 679-698

[2] An N, Liu H. Research and application of image processing methods [J]. Chinese Journal of Scientific Instrument, 2006, (S1): 792-793.

[3] Xia JF, Ying J. An improved Canny edge detection algorithm [J]. Electronic World, 2017, 10: 22-23.

[4] Xu CJ. Canny edge detection based on Open CV in Ubuntu[J]. Digital Technology and Applications, 2011, 1: 53-54,

[5] Dai L, Wu D, Zhang J. Research on ESPI image enhancement technology based on Open CV vision library[J]. Journal of Electronic Measurement and Instrumentation, 2013, 27(10): 975-979.

[6] Chen H, Li R. Application of Canny edge detection algorithm in robot vision[J]. Modern Computer, 2011, 11: 17-20.

[7] Mao X, Len X. Introduction to OPEN CV3 programming[M]. Bei Jing: Publishing House of Electronics Industry, 2015.

[8] Qin X, Wen Z, Vivian J. Image processing based on Open CV[J]. Electronic Test, 2011, 07: 39-41.

[9] Wang F, Li Y, Liu J. Implementation of machine vision image processing technology based on Open CV[J]. Mechanical and Electronic, 2010, (06): 54-57.

[10] Chen L. Application of computer vision class library Open CV in VC[J]. Microcomputer Information, 2007, (12): 209-210+171.

[11] Chen D. Development and application of computer vision technology[J]. Computer Knowledge and Technology, 2008,4 (35): 2449-2450+2452

[12] Bay H, Ess A. Speeded-up robust features(SURF)[J].Computer Vision and Image Understanding, 2008, 110(3): 346-359.

[13] Yan J, Hang Z, Yi Q. Fast implementation of high resolution digital video image registration based on CUDA[J]. Chinese Journal of Scientific Instrument, 2014,35(02): 380-386.

[14] Chen X, Cheng X, Ye Q. Research on multi-channel parallel frequency algorithm based on GPU[J]. Chinese Journal of Scientific Instrument, 2010, 31(07): 1674-1680.

[15] Li Y, Zhao K, Chu X. Speeding up K-Means Algorithm by GPUs. IEEE International Conference on Computer and Information Technology, 2010: 115-122.

AUTHOR BIOGRAPHY

Zhao Xu was born in He Bei, China, in 1992. he received bachelor's from Beijing Information Science & Technology University(BISTU), Beijing, China, in 2014 and 2017, respectively. Now he is a master with the Department of Mechanical and Electrical(BISTU), Bei Jing. His research interests include fault diagnosis and image processing.