Research of Image Pre-processing Algorithm Based on FPGA

Yang Yongjin Zhou Xinmei Xiang Zhongfan*

School of Mechanical Engineering and

Automation Xi Hua University,

ChengDu, China

Emails: yangyongjin007@163.com

Submitted: Feb. 03, 2013 ***Accepted: June 30, 2013*** ***Published: Sep. 05, 2013***

Abstract- How to design a low-cost , reliable and real-time target recognition system with large amount of data has become a hot topic in the area of image processing .However, Edge detection has played an important role in target recognition system. The threshold of traditional canny edge detection algorithm must be setting by human, and has a large number of calculations. In order to overcome the shortcomings of the traditional Canny algorithm, proposing an adaptive threshold edge detection algorithm, and realizing it by hardware. This paper will introduce the implementation of the common low-level image processing algorithm in the FPGA, including color space convert module , edge extraction algorithms module , Hough transform module .The results of the experiment indicate that to realize the large amount of calculation of image processing by FPGA hardware logic, not only improves the effect of image processing, but also has high real-time!

Index terms*:* FGPA, candy operator, Edge detection, Gaussian transform

I. INTRODUCTION

Edge detection is a basic computer vision technology in image processing and analysis. It has been widely studied and many related algorithms such as Robert, Prewitt, Kirsch, Gauss-Laplace, Canny etc. have been proposed. The canny operator with good signal-to-noise ratio and detection precision has been widely applied in the fields of image processing [1][2].

When using canny operator to detect edge, we need to confirm the parameter of the high and low threshold. Different thresholds have a great influence on the result of edge detection. In order to overcome the shortcomings of the Canny operator, this paper introduces an adaptive threshold Canny operator. the threshold calculation method can be obtained according to the own information of image and the analysis of gradient magnitude histogram of image[3]. The adaptive generating threshold overcomes the deficiency of the traditional algorithm which needs to set the threshold artificially.

And, in the light of the low real-time of the traditional visual, this paper introduces the application of thresholds edge detection algorithm on FPGA. Field-programmable gate array (FPGA) in the application of digital signal processing, will gradually become the mainstream of the front-end signal processing. The filter algorithm plays an important role in signal processing, signal detection and communication field. In the real-time information processing system, the requirement of the performance and processing speed of the filter are strict. Especially under the condition of satisfying the system performance, the processing speed is essential. The experimental results show that, the realization of the adaptive threshold edge detection algorithm on FPGA can greatly shorten the time consumed by the algorithm, has the advantages of high real-time, strong adaptability.

This paper discusses the basic principles and specific method of real - time image pre-processing algorithm by using EP2C20Q240 devices of Cyclone II series of ALTERA[4].

II. REALIZATION OF COLOR SPACE CONVERSION

The color image collected by capture card is YUV image. YUV image needs to be converted into RGB image in many applications. This section introduces the FPGA realization method of converting YUV images into RGB image.

YUV image and RGB image conversion relation is as follows:

$$R=Y+1.4075*(V-128) \quad (1)$$

$$G=Y+0.3455*(U-128)-0.7169*(V-128) \quad (2)$$

$$B=Y+1.779*(U-128) \quad (3)$$

It's hard to realize in the FPGA because of involving floating point arithmetic, but it can be achieved through the look-up table or shaping approximation. The method of Look-up table is to establish four one dimensional arrays and each array stores the result of a product term in the above formulate. YUV values in the range 0-255, so each array is 256 units and occupies 1K units of storage space. The calculation speed of Look-up table is fast and the calculation accuracy is high, but the RAM resources in FPGA are limited. In order to save valuable RAM resources in FPGA and with the low computational accuracy demanding, shaping approximation is a good way. Approximation calculation formula is as follows:

$$u=YUVdata[UPOS]-128 \quad (4)$$

$$V=YUVdata[VPOS]-128 \quad (5)$$

$$Rdif=v+((v*103)>>8) \quad (6)$$

$$Invgdif=((u*88)>>8)+((v*138)>>8) \quad (7)$$

$$Bdif=u+((u*198)>>8) \quad (8)$$

$$r=YUVdata[YPOS]+rdif \quad (9)$$

$$g=YUVdata[YPOS]-invgdif \quad (10)$$

$$b=YUVdata[YPOS]+bdif \quad (11)$$

Among them, YUVdata [YPOS], YUVdata [UPOS], YUVdata [VPOS] were input Y component, U component and V component, r, g, b , respectively, for the output of the r component , G component and the b component .

In order to prevent overflow and determining whether the calculation result is in the range 0-255, the calculation result of the more than 255 is set to 255 and the calculation result of less than 0 is set to 0.

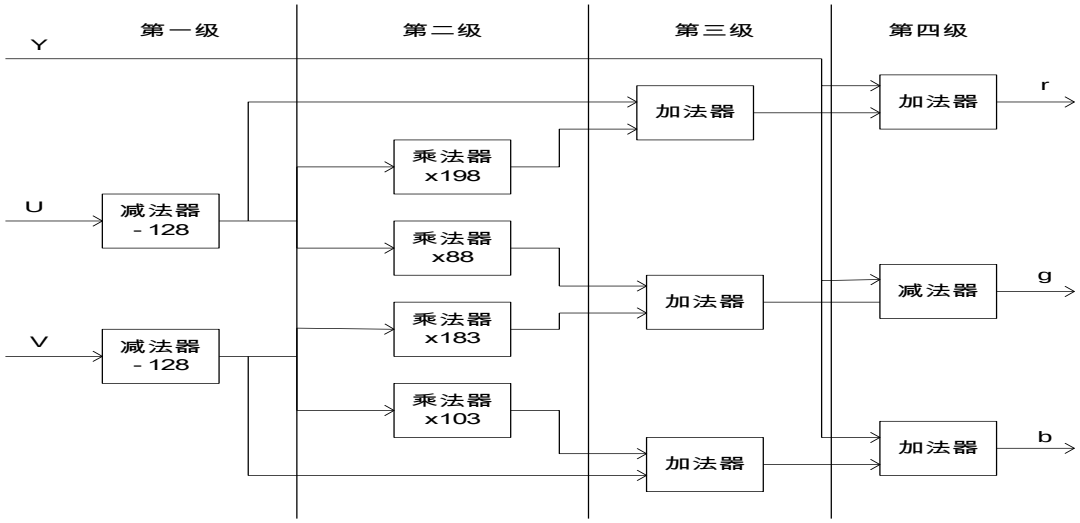

Figure 1. Color model conversion of the block diagram of FPGA realization

The whole conversion process is divided into 4 lines. The first line consists of two subtracters to operate UV component minus 128[5]. The second pipeline has four multipliers of 8x8 to operate the four multiplications in above formula. since each multiplication result shift to right eight bit as input of the next line, four multipliers high eight bit are kept as multiplication results. The third pipeline constituted by the three adders to achieve the operation of rdif, invgdif and bdif. The last line consists of two adders and a subtracter with the final results r, g and b.

III. REALIZATION OF EDGE EXTRACTION ALGORITHM

Edge detection is a basic computer vision processing technology in image processing and analysis and is researched extensively. Many related algorithms are put forward, such as Robert, Prewitt, Kirsch, Gauss - Laplace, Canny, etc. The Canny operator with good signal-to-noise ratio and precision has been widely used in the fields of image processing. This paper describes the implementation of the adaptive threshold Canny on FPGA[6].

First introduced adaptive threshold algorithm before introduce FPGA implementation of adaptive threshold Candy operator. Assuming the image collected via 7113 and filtered is I(i, j), after a non-maxima suppression , all pixels whose edge marks are not zero can be described by gradient histogram, in which x axis stands for gradient amplitude while y coordinate stands for the number

of pixels. Investigation of a plurality of the histogram of gradient, we can find that the following features:

(1) The image of most of the proportion of background area pixel gradient amplitude is lesser, in gradient amplitude histogram near zero exists a corresponding background area pixel set peak;

(2) The image edge location of the pixels of the gradient value is bigger, the edge pixels in the gradient amplitude histogram zero right will form a series of peak, every peak on behalf of a kind of edge pixels intensity set;

(3) In a gradient magnitude in the histogram, between the background peak and a first edge spikes have a smooth transition region, the threshold should be selected in this area that division error minimal[7].

From the characteristics of the gradient histogram, the selection of threshold value is actually to find the smooth region between background spike and the first edge peak. The selection of threshold value is the gradient magnitude of a random pixel in this region. In order to identify this region easily, making difference between two adjacent points of gradient magnitude[8], it can get the difference histogram.

$$diff(i) = \begin{cases} diff & diff > th \\ 0 & else \end{cases} \qquad (12)$$

diff=abs(NMS(i+1)-NMS(I)) , NMS (i) is gradient histogram after non-maxima suppression.

After the conversion of the above formula, the smooth region between background peak and the first edge peak of the gradient histogram is transformed into crossing zero collection of difference histogram. In this paper it selects the gradient amplitude of the first crossing zero as the threshold value. Therefore, the high and low threshold values of the Canny operator can be determined by the following formula:

$$Th_h = Arg(diff(i) = 0) \qquad (13)$$

$$Th_l = Th_h / 2 \qquad (14)$$

Th_h represents high threshold value and Th_l represents low threshold value. Arg represents gradient magnitude which is the first i pixel corresponding with diff (i) = 0[9].Using FPGA to achieve the improvement of Candy algorithm block diagram of the system are as follows:

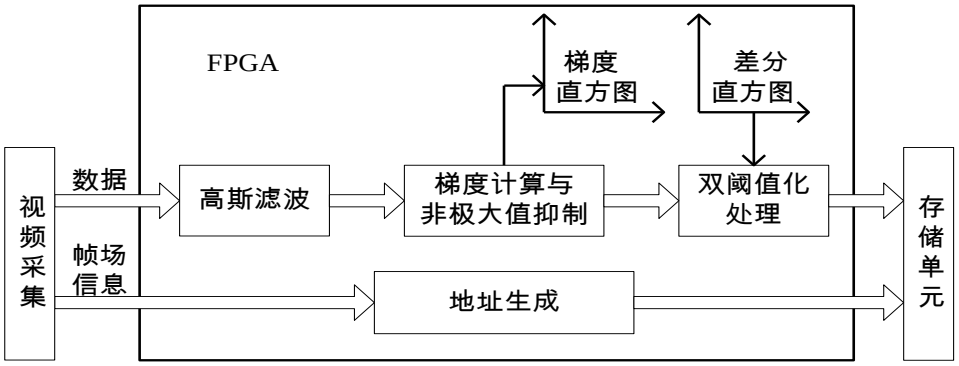

Figure2.The Improved Canny algorithm implemented on the FPGA

Video capture module transforms the analog signal of camera into the image data which together with frame and field information is then transmitted to FPGA. In the FPGA, the image data first go through a Gaussian filter, and then calculate the gradient value of each pixel of and suppress the non- maxima of the gradient amplitude. The results of Non maxima suppression are figured out according to difference histogram and calculate the high, low threshold. Finally, the edge image is obtained through dual-threshold processing of the results of the non - maxima suppression. Hardware implementation of each module is elaborated in the following parts.

a .Realization of Gaussian filter

Gauss filter is essentially a signal filter. Its purpose is to process signal smoothing. It is usually used to deal with the fuzzy and reduce the noise signal. Its fuzzy processing is often used for preprocessing and eliminates some unnecessary components when obtaining the target, which benefits the processing of the Gaussian filter. Gaussian filter formula is as follows：

$$g(x,y) = G(x,y) * f(x,y) = \frac{1}{2\pi\partial^2}\exp\left[-\frac{x^2+y^2}{2\partial^2}\right] * f(x,y) \qquad (15)$$

g(x,y) is the image going through Gaussian filter , f(x,y) is the original image.

The Gaussian filter is a normal distribution function. When it is used as a weighting function, its coefficients are symmetric. The intermediate pixel values are larger than other surrounding pixel values, so the intermediate pixels are of great important. Considering the design purpose, the 5X 5 Gauss template is chosen. Through dispersing, integerization and normalization of the 5X 5 Gauss template, the following formula is obtained.

$$\begin{pmatrix} 1 & 2 & 3 & 2 & 1 \\ 2 & 4 & 6 & 4 & 2 \\ 3 & 6 & 7 & 6 & 3 \\ 2 & 4 & 6 & 4 & 2 \\ 1 & 2 & 3 & 2 & 1 \end{pmatrix} \tag{16}$$

The image will be calculated in 5 x5 templates, so firstly a 5 x5 window should be opened for the input image. As soon as clock arrives, every row of the window shifts to the left and the new pixel will be added to the right end. It needs window operation with five rows of pixels, so only in the fifth rows of reading image can make window operation. The moving diagram is shown below:

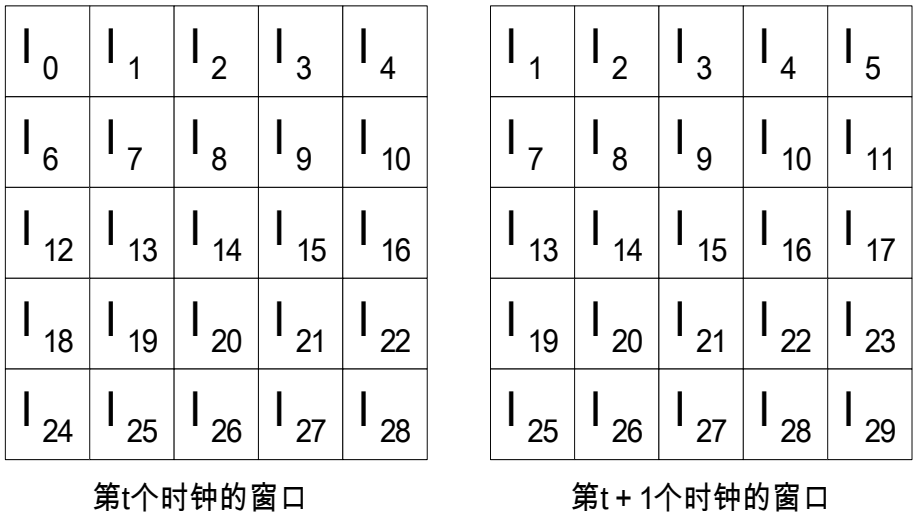

Figure 3. Moving diagram of Gauss filter

b .Gradient calculation and realization of non-maxima suppression module

After Image going through the Gaussian filter and using open circuit 1 to open a 5x5 window, the 5x5 neighborhood window of each pixel is obtained. Template arithmetic circuit gets EH, EV, EDR and EDL. The comparison circuit compares the absolute value of EH, EV, EDR and EDL and the maximum value is regarded as the gradient value and the direction corresponding to the maximum value is gradient direction. Encoding circuit is to encode four gradient directions by using 2-bit registers. The values 00, 01, 11, 10 of the 2-bit registers respectively represent the four directions of horizontal, vertical, left diagonal and right diagonal. The center pixel's Gradient direction and gradient magnitude of the current window is obtained by comparing encoding circuit. The open circuit 2 Opens a 3x3 window for the calculated gradient magnitude data, which together with data of gradient direction are sent to a non- maxima suppression module. Non-maxima suppression circuit compares whether the gradient magnitude of the center pixel is largerr than the gradient magnitude of two adjacent pixels along the gradient direction in the current 3x 3 neighborhood windows[10]. The comparison result is stored in the corresponding marked register and finally the gradient image going through the non-maxima suppression is outputted. Module circuit diagram is shown below:

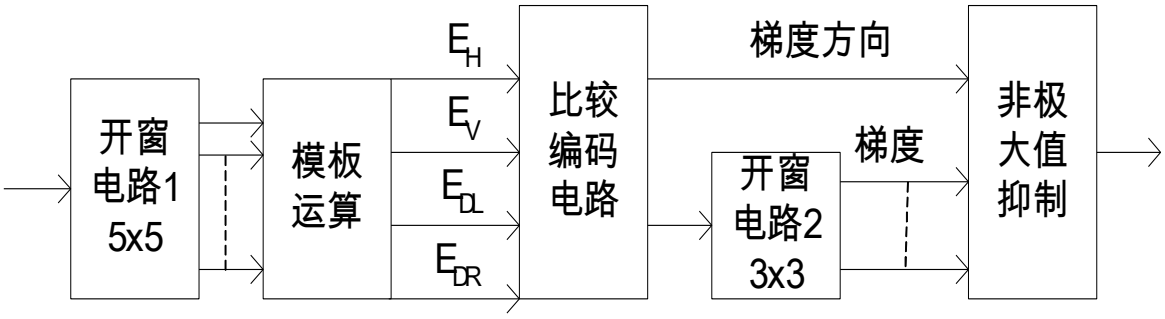

Figure 4. Module realization circuit

c .Realization of the adaptive dual-threshold

The selection of adaptive threshold is based on gradient histogram, therefore we need to make histogram statistics for the image after going through non-maxima suppression[11]. Take 360 x280x8bit gray image for example, after calculation, gradient values centralize in 0-100, so it needs 100register groups (not including the point whose gradient value is zero, for the point will not affect dividing the threshold). Each register's width is 12bit. These registers are used to store

pixel numbers of different gradient value and the gradient value is the address of register groups. At the beginning of each image, these registers are cleared. After inputting gradient value and through Address selector, the content of the register with corresponding address istaken out and sent to the accumulator 1. The content is rewritten in the register after adding 1 until all the pixels points of the whole image are counted[12][13].

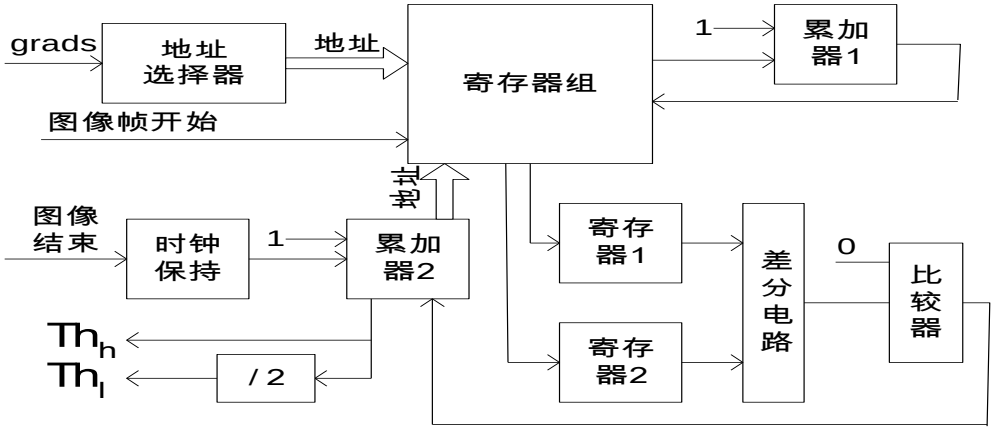

Figure 5. The selection circuit of adaptive dual-threshold

When the image end signal comes, the clock hold circuit keeps the 100 system clock. Within the valid period of the clock, each clock coming, the content of accumulator 2 will be added 1 automatically. The accumulated result is regarded as the address of register group. The contents of the corresponding address register and the contents of the next address register are respectively sent to the register 1 and register 2. The comparator compares whether the result of the contents' difference of register 1 and register 2 is zero. If the result is zero, it will send the signal to stop accumulating and stop the accumulation of accumulator2. At this time, the value of accumulator 2 is the high threshold Thh. The low threshold Thl. is obtained by shifting one bit of Thh to the right

d .Realization of threshold processing module based on dual-threshold

The image through non-maxima suppression is marked as f. each point (i,j) going through conparator1 and conparator2 respectively compare with high and low threshold. All the points that are larger than high threshold are set to 1 and the other pixel points are set to 0. The

obtaining image is marked as f_1 that represents a strong edge image[14]. All the pixel points that are smaller than the high threshold and larger than low threshold are set to 1 and the other pixel points are set to 0. The obtaining image is marked as f2 that represents weak edge image. If f1(i,j) represents 1, then the current pixel points are certainly edge points and the output is 1. if f1(i,j) represents 0, while f2(i,j) represents 1, it needs to view 8 neighborhood of f1(i,j). If the 8 neighborhood exists strong edge points, the current pixel point is edge point and the output is 1. All the non-edge points' output is 0.Proceeding like this until the entire image scan is completed, the output is binarized edge image, the realization of the circuit diagram is shown below:

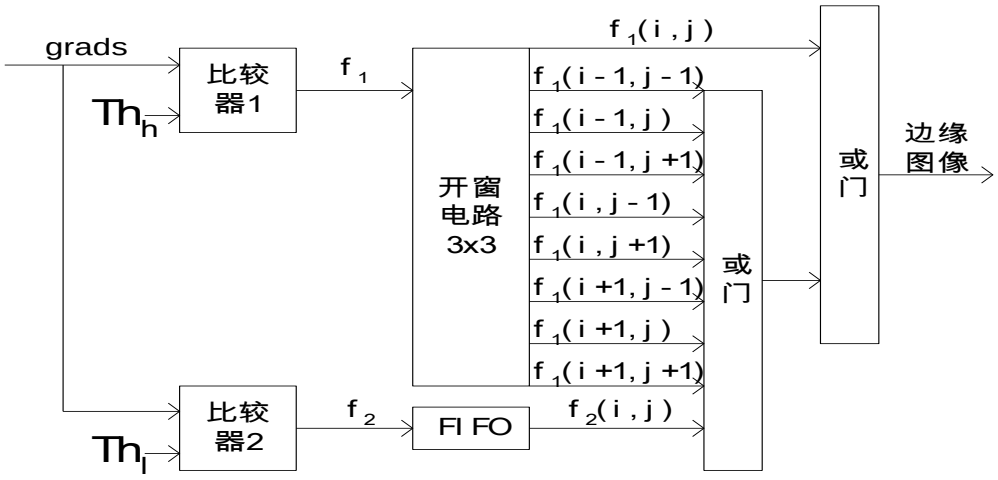

Figure6. Realization of threshold processing module based on dual-threshold

A 3x3 window is opened for the strong edge image, so (i,j) is actually put off one line and one column comparing the input point. The image f_2 needs to add FIFO to put off each point, so it can correspond with the point in F1[15][16].

IV. REALIZATION OF THE HOUGH TRANSFORM

Assuming that the size of the image is NxN, taking the center of the image as the coordinate origin, the variation range of ρ is $(-N\sqrt{2}, N\sqrt{2})$ and the variation range of θ is (0, π). Considering the trigonometric conversion relation[10], Calculation formula of Hough transforms is equivalent to the following two equations:

$$\rho_{0\sim\pi/2} = x\cos\theta + y\sin\theta \qquad 0<\theta<\pi/2 \qquad (17)$$

$$\rho_{\pi/2\sim\pi} = x\cos(\theta+\pi/2) + y\sin(\theta+\pi/2)$$
$$= -x\sin\theta + y\cos\theta \qquad 0<\theta<\pi/2 \qquad (18)$$

Viewing from formula (17), (18), when θ changes in the range of (0, π / 2), $\rho_{0\sim\pi/2}$ and $\rho_{\pi/2\sim\pi}$ can be obtained simultaneously. Through the conversion of formula (17), (18), the change range of θ is cut to half, while the calculation of ρ is replaced by parallel calculation of the $\rho_{0\sim\pi/2}$ and $\rho_{\pi/2\sim\pi}$. Thus parallel calculation can realized.

Because θ is in the discrete digital space, the values of θ in the range of(0, π /2) are a series of discrete values: θ, Δ θ, 2 Δ θ, 3 Δ θ,....., π /2. Assuming that θ gets value at the i-th time, θ =i Δ θ = θ 0, so $\rho_{\pi/2\sim\theta} = \rho x_i$, $\rho_{\pi/2\sim\theta} = \rho y_i$. When θ = (i+1) Δ θ, ρx_{i+1} and ρy_{i+1} meet:

$$\rho x_{i+1} = x\cos(\theta_0+\Delta\theta) + y\sin(\theta_0+\Delta\theta)$$
$$= (\rho x_i + \tan\Delta\theta * \rho y_i)/\cos\Delta\theta \qquad (19)$$

$$\rho y_{i+1} = -x\sin(\theta_0+\Delta\theta) + y\cos(\theta_0+\Delta\theta)$$
$$= (\rho y_i - \tan\Delta\theta * \rho x_i)/\cos\Delta\theta \qquad (20)$$

In formula (19), (20), assuming $\tan\Delta\theta = 2^{-5}$, without consideration of the influence of $\cos\Delta\theta$, then:

$$\rho x_{i+1} = \rho x_i + 2^{-5} * \rho y_i \qquad (21)$$

$$\rho y_{i+1} = \rho y_i - 2^{-5} * \rho x_i \qquad (22)$$

wherein, $\rho x_0 = x$, $\rho y_0 = y$, i=0,1,2,3……。System diagram of improved Hough transform is shown below:

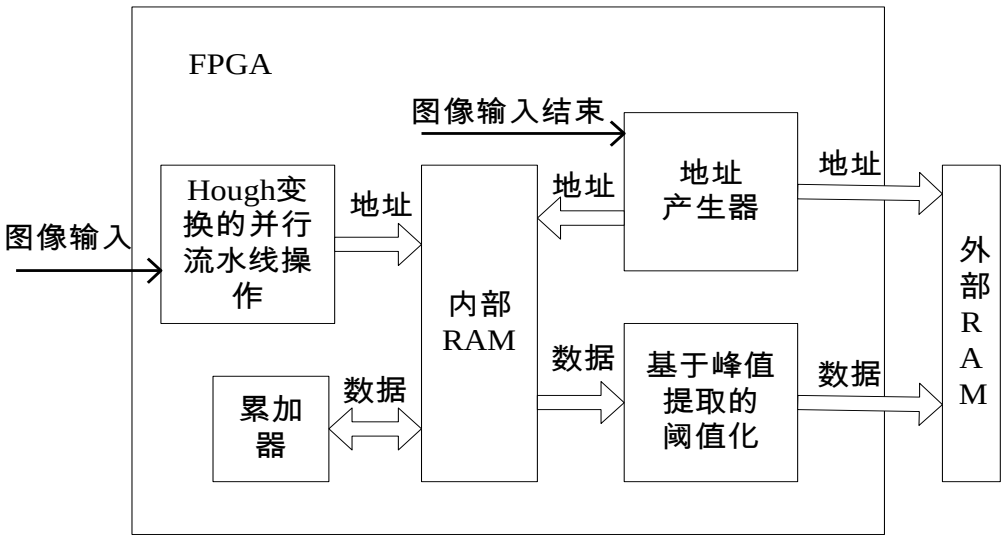

Figure 7. Gauss Transform Realize in FGPA

After inputting edge image and through pipeline operation modules of Hough transform, the value of ρ with different θ is calculated. Taking the value of p as the inner address of RAM, cumulative unit of the address is accumulated through the accumulator and then the cumulative result is stored into the original cumulative unit. When the end signal of edge image input arrives, address generator simultaneously generates internal and external RAM address based on row and column order of the two-dimensional accumulation matrix. Reading the original two-dimensional accumulation matrix from the internal RAM, thresholding accumulation matrix is obtained by extracting thresholding module from peak value and then the new accumulation matrix is stored into the corresponding external RAM address. The directions of the arrows in the figure only represent the directions of the data stream. Each module under the drive of system clock is operated synchronously. The realization of each module's hardware will be introduced in the following parts[17] .

a .Hardware realization of parallel lines

Considering θ increasing in the range of (0, π / 2) by step of Δθ, assuming $\Delta\theta = 1°$, each pixel needs to calculate the value of ρ when $90/1 \approx 50$ times. Therefore, the Hough transform can be designed as a 50 - stage pipelined operation. Each stage pipeline realizes the calculation (Shift, plus) of the formula (21) and (22) to get the results of ρ_{x_i} and ρ_{y_i}. Taking

ρx_i and ρy_i as the address of RAM2i and RAM2i +1 respectively, the cumulative unit of the address is accumulated and then the cumulative result is stored into the cumulative unit of the original address. Taking output of each pipeline as the input of next pipeline when the clock comes, 51 pairs of ρx_i and ρy_i can be obtained simultaneously and at the same time, 102 cumulative units with the addresses of ρx_i and ρy_i in RAM are accumulated. All the calculations of the Hough transform can be completed in a clock cycle. The parallel pipelined hardware diagram of Hough transform is shown below:

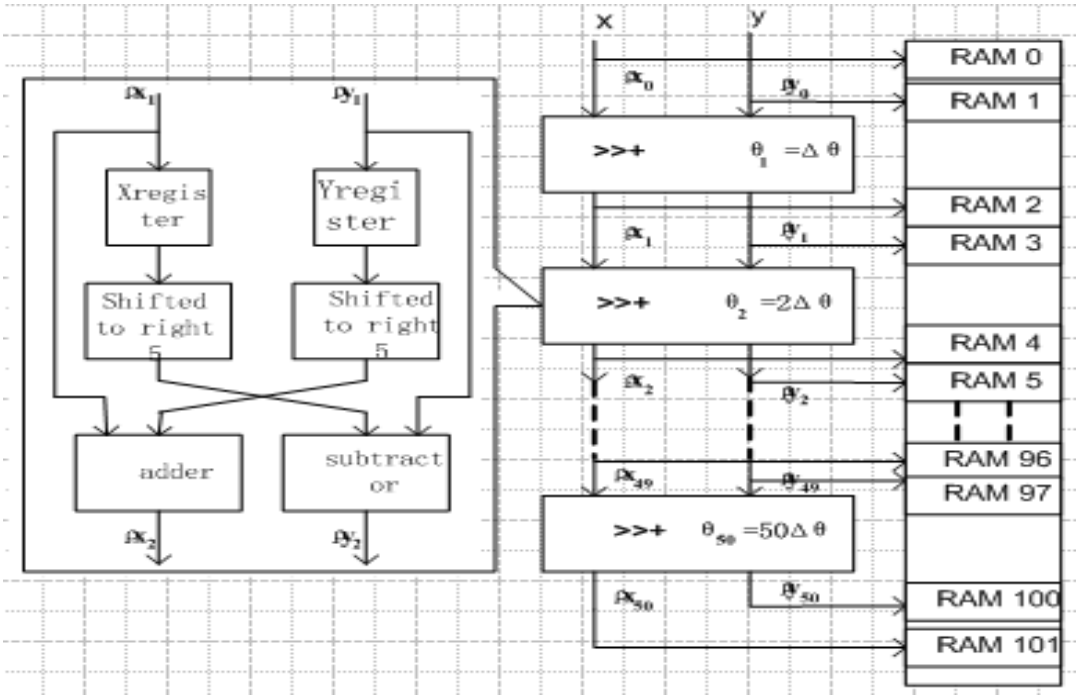

Figure 8. Realization diagram ofparalled pipelined hardware of Hough transform

b .hardware realization of the Peak extraction threshold

When the end signal of edge image input arrives, the address generator generates internal RAM address according to the p value from small to large. in every clock cycle, Each RAM outputs an cumulative unit data, so the internal RAM totally output cumulative unit data. Open circuit consists of 102x3 registers and obtains 3 rows and 102 columns window data. Window data overall move down one line in each clock cycle and data of the first line is filled by the 102 data that are outputted by RAM.The comparator module compares each data of the middle row in the

102x3 window with the surrounding 3x3 neighborhood data. If the current center data are partial maximum values and larger than the given threshold, the cumulative unit corresponding to the center data is set to 1, otherwise it is set to 0.

In Each clock cycle, the comparator module outputs 102 thresholding cumulative unit data which are stored in the external memory. If the new cumulative unit value is 1, it represents a straight line. With the value of 0, it represents there is no straight line. Thresholding hardware diagram based on Peak extraction is shown below:

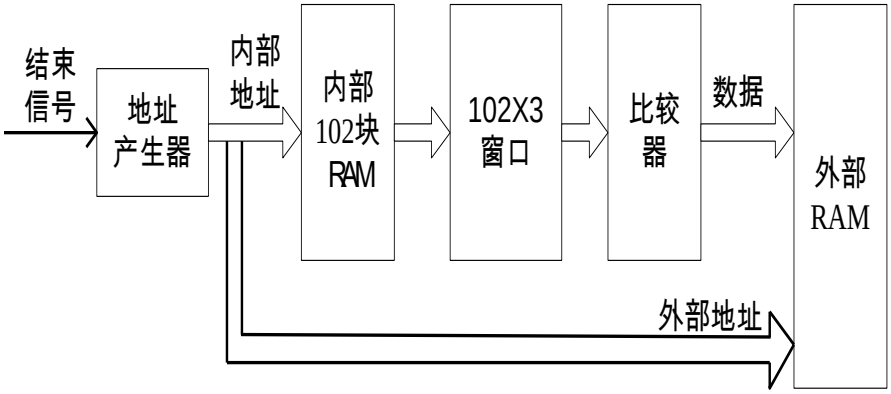

Figure 9. Peak extraction threshold hardware implementation

V. THE EXPERIMENT RESULT AND ANALYSIS

The logic diagram of this design is realized by using Verilog language[18][19]. Verilog is a standard hardware description language and is supported almost by all commercial development platform and simulation tool, which can not only shorten the design cycle, but also reduce the investment risk. The target chip adopts EP2C20Q240 devices of Cyclone II series of ALTERA. The whole design processes such as input, debugging, synthesis, simulation, the final implementation and device programming are performed on the platform of ALTERA's Quartus ii 9.0. Figure 10 is the canny algorithm using resource in FPGA,Figure 11 is the RTL view of the Canny algorithm in FPGA,Figure 12 is the result of the Canny algorithm for edge detection. The experimental results indicate the realization of edge detection on FPGA.

Flow Status	Successful - Mon Apr 29 16:55:25 2013
Quartus II Version	9.1 Build 222 10/21/2009 SJ Web Edition
Revision Name	imageboard
Top-level Entity Name	imageboard
Family	Cyclone II
Device	EP2C20Q240C8
Timing Models	Final
Met timing requirements	Yes
Total logic elements	1,952 / 18,752 (10 %)
Total combinational functions	1,571 / 18,752 (8 %)
Dedicated logic registers	1,333 / 18,752 (7 %)
Total registers	1349
Total pins	117 / 142 (82 %)
Total virtual pins	0
Total memory bits	21,868 / 239,616 (9 %)
Embedded Multiplier 9-bit elements	0 / 52 (0 %)
Total PLLs	1 / 4 (25 %)

Figure 10. The resource consumption result in the cyclone ii device

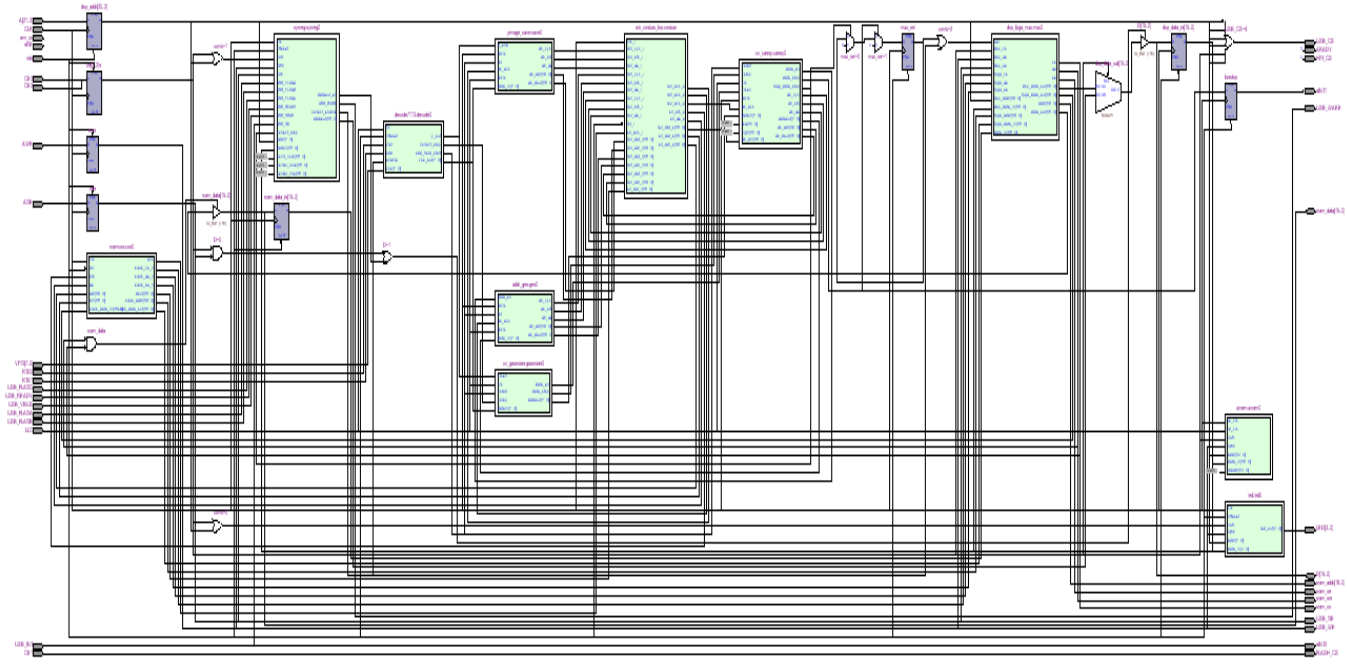

Figure 11. The RTL view of the canny algorithm implemented on FPGA

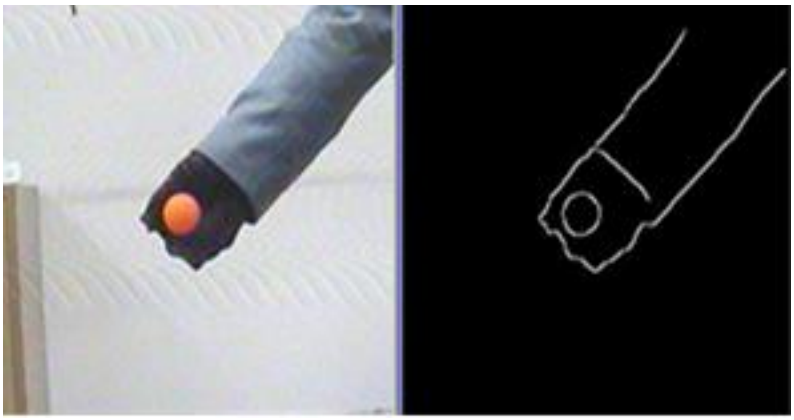

Figure 12.Schematic diagram of experimental results

VI. CONCLUSIONS

This paper introduces an Image pre-processing system based on FPGA to solve the problem of image data calculation in the practice of engineering. By using adaptive Canny edge detection algorithm, can take advantage of the image itself, adaptive extraction threshold to avoid errors caused by human factors.Through the actual data it can conclude that it is feasible to make fast algorithm by using FPGA. It can also achieve the established goals and meet the requirement of real-time.

Acknowledgment

We would like to thank the financial supported by the innovation fund of postgraduate, Xi Hua University. we would also like to acknowledge the support from the Sichuan Science And Technology Innovation Seeding Project for the first author and the support from the key laboratory of manufacturing and automation of Sichuan province at Xi Hua University for the second author.

References

[1] Mei Yuesong,Yang Shuxing,Mo Bo.Adaptive edge detection of Gray image[J].Computer engineering and Applications,2007,43(5):63-66.

[2] Chen Ren,Xu Kaiyu.Research on figure edge detection technology of infrared image[J].Laser and infrared,2005(9):703-705

[3] John Canny.A computational approach to edge detection[J].IEEE transactions on pattern analysis and machine intelligence,1986(6):679-698.

[4] Altera INC.Cyclone II device handbook[H],2003:17-21.

[5] Milan Sonka, Vaclav Hlavac,Roger Boyle. Image processing,analysis,and machine vision[M].USA:Thomson Learning and PT Press,1999:60-62.

[6] Chu Zhenyong,Wen Muyun.FPGA design and application,Xi'an,Electronic and Science University Press ,2002.

[7] Scott Tattersall,Kenneth Dawson-Howe.Adaptive shadow identification through automatic parameter estimation in video sequences[C]//Irish Machine Vision and Image Processing Conferenence,2003:57-64.

[8] Dierichx B,Meynants G. Missing Pixel Correction Algorithm for Image Sensors.SPIE,1998,34(10):200-203.
[9] Armstrong James R,F Gail Gray.VHDL design representation and synthesis[M].USA:Prentice Hall PTR,2002:403-405
[10] Altera Inc.Introduction to the quartus II software[H].2006:152-174.
[11] H Rabah, H Mathlas. Linear array processors with multiple access modes memory for real-time image processing. Proceeding of IEEE, 2002.
[12] J. Canny, A computational approach to edge detection, IEEE Transactions on Pattern analysis and Machine Intelligence 8 (6) (1986) 679–698.
[13] James R Armstrong, F Gail Gray. Li Zongbo, Wang Ronghui translation. VHDL design express and synthesis. Beijing: Mechanical Industry Press, 2002.
[14] Xue MingXing. Sobel algorithm based on FPGA devices. Electronic components application. 2008 (10).
[15] M Y Niamat, Prabhu. Logic BIST Architecture for FPGAs. IEEE Midwest Symposium on Circuits and Systems, 2001, 442~445.
[16] Li Guogang,Yu Jun. realization method based on FPGA image VGA graphics controller [J]. information technology, 2006 (7).
[17] Michael D. Ciletti, Modelling, Synthesis and Rapid Prototyping, Prentice Hall, 1999
[18] Zheng Youquan. IC technology lectures [J]. world electronic components ,2005 -9.
[19] Zhao Shuguang,Guo Li. Programmable logic devices principle, development and application [M]. Xi'an Electronic and Science University press, 2000.