FPGA implementation of eight-direction Sobel edge detection algorithm based on adaptive threshold

Kaiqiang Zhang¹,a, Qiang Liao¹*

¹ School of Mechanical Engineering, Chongqing University, Chongqing, Shapingba District, Chongqing 400030, China
² zhangkaiqiang@cqu.edu.cn
³ Corresponding author's e-mail: lq922@163.com

Abstract. As the basic content of digital image processing, edge detection is increasingly applied in machine vision, face recognition, target tracking and other fields. In particular, edge detection based on Sobel operator is generally applied in industrial practices. However, most current edge detections based on PC-side serial processing are extremely arduous to satisfy real-time processing. This paper proposes FPGA implementation of eight-direction Sobel edge detection algorithm based on adaptive threshold. Specifically, an eight-direction Sobel edge detection algorithm based on adaptive threshold is presented. Subsequently, the specific process of implementing the algorithm on FPGA using the design philosophy of parallel pipeline is described. Finally, a comparison of the performance between traditional Sobel and improved algorithm reveals that the improved algorithm offers distinct advantages in accuracy and detection effectiveness, and the parallel implementation of FPGA hardware can obviously enhance the detection speed and fulfil the real-time requirement.

1. Introduction

Edge as the significant feature of digital image contains abundant important information of image. Accordingly, edge detection regarded as an indispensable basic content of digital image processing is a precondition for image segmentation, feature extraction and image understanding. Due to the convenient implement and the efficient edge detection, the traditional Sobel[1] edge detection algorithm, a first-order differential operator to perform edge detection on the image, is usually the preferred option in practical applications. However, the traditional Sobel algorithm that only sensitive to the horizontal and vertical edges of the image is difficult to detect edge features in other directions. Many small edges are generally ignored, which makes it difficult to adapt to complex edges and high precision requirements. To solve this problem, this paper uses the adaptive threshold multi-directional Sobel algorithm to detect the edge of the image. Through the selection of the adaptive threshold, the errors caused by the subjective setting of the threshold can be greatly reduced. Simultaneously, using the edge detection in eight directions enables edge features in other directions of the image and finer edge points to be detected, which improves the accuracy of edge detection.

The majority of the edge detection is currently based on PC-based MATLAB or VS development tools. In spite of the upper main-frequency of PC, the efficiency of image processing by software
serial processing method is poor. Especially the running speed of real-time and online image processing through PC is far from enough [2,3]. As a high-density programmable logic device, FPGA offers the characteristics of high integration, easy use, and short development and market cycles. In addition, FPGA devices can greatly reduce processing delays, improve real-time performance and reliability through parallel pipeline design philosophy, and are now widely used in medical imaging, finance, deep learning, image recognition and natural language processing. Therefore, this paper implements the adaptive threshold multi-directional edge detection algorithm on FPGA, which makes this algorithm have great advantages and application prospects in high-precision real-time image processing.

2. Sobel edge detection operator

2.1. Traditional Sobel edge detection algorithm

Traditional Sobel operator, a discrete first-order difference operator used to calculate the approximate value of the one-step amplitude value of the digital image function, uses the horizontal and vertical two convolution kernels as the direction template. It performs convolution for each pixel in the image to detect edges from different directions, and conducts convolution weighting operations on the $3 \times 3$ neighborhood of the original image through the $3 \times 3$ size window. The calculation process is expressed respectively as follows:

$$G_x = \begin{bmatrix} -1 & 0 & 1 \\ -2 & 2 & 0 \\ -1 & 0 & 1 \end{bmatrix} * f \quad (1)$$

$$G_y = \begin{bmatrix} -1 & -2 & -1 \\ 0 & 0 & 0 \\ 1 & 2 & 1 \end{bmatrix} * f \quad (2)$$

It is assumed that the original image is $f$, formula (1) is the convolution of the convolution kernel in the horizontal direction and the neighborhood of the original image, and formula (2) is the convolution of the convolution kernel in the vertical direction and the neighborhood of the original image. Supposing that the mathematical model of the original image is defined as $f(i,j)$, the specific calculation expressions of the gradient in the horizontal and vertical directions could be described as follows:

$$|G_x| = |f(i-1,j+1)+2f(i,j+1)+f(i+1,j+1)-f(i-1,j-1)-2f(i,j-1)-f(i+1,j-1)| \quad (3)$$

$$|G_y| = |f(i+1,j-1)+2f(i+1,j)+f(i+1,j+1)-f(i-1,j-1)-2f(i,j-1)-f(i-1,j+1)| \quad (4)$$

The calculation results of formulas (3) and (4) take the norm to measure the magnitude of the gradient, and the maximum value of the two convolution results is used as the output value of the gradient at this point. The expression is given by formula (5).

$$G = \max \{|G_x|, |G_y|\} \quad (5)$$

Traditional Sobel operator calculates the gray weighted value of the pixel neighborhood through the horizontal and vertical operator templates, and finally takes the extreme value to judge the edge of the image. However, the edges of the image are multi-directional. The traditional Sobel operator has only two direction templates, hence it is only sensitive to the edges in the vertical and horizontal directions, and is less sensitive to the edges in other directions. Therefore, for edge images with high complexity and high-precision requirements, traditional Sobel edge detection generally ignores the edge features in other directions of the image, making the edge detection result incomplete.

2.2. Improved eight-direction Sobel edge detection algorithm
The traditional Sobel algorithm performs convolution calculations on discrete pixel chromaticity values, and then calculates the approximate value of the maximum gradient in the image direction through differential weighted average. Edge information is generated in this algorithm by calculating the vector gradient corresponding to each pixel in the image. However, the actual image has many edge directions, only two direction templates are used that makes other gradient directions cannot be detected, and massive edge information will be missed. Hence, to offset the shortcomings of the traditional Sobel algorithm for edge detection and increase edge information in other directions, the traditional Sobel algorithm templates are expanded, which uses the extended multi-directional template operator for edge detection.

![Figure 1. Eight directions of expansion](image)

As shown by the solid arrow in Figure 1, the algorithm uses the edge detection templates of $0^\circ$, $22.5^\circ$, $45^\circ$, $67.5^\circ$, $90^\circ$, $112.5^\circ$, $135^\circ$, $157.5^\circ$ in eight directions [6] for detection, the dashed arrow indicates the direction of symmetry. Taking the absolute value of the convolution result can realize edge detection in eight symmetric directions. The extended eight-direction template operator uses the $5 \times 5$ convolution template [7], and the convolution is calculated respectively as follows:

\[
G_{0^\circ} = \begin{bmatrix}
0 & 0 & 0 & 0 & 0 \\
-1 & -2 & -4 & -2 & -2 \\
0 & 0 & 0 & 0 & 0 \\
1 & 2 & 4 & 2 & 1 \\
0 & 0 & 0 & 0 & 0 
\end{bmatrix} * f
\]

\[
G_{22.5^\circ} = \begin{bmatrix}
0 & 0 & 0 & 0 & 0 \\
0 & -2 & -4 & -2 & 0 \\
0 & 2 & 4 & 2 & 0 \\
0 & 0 & 0 & 0 & 0 
\end{bmatrix}
\]

\[
G_{45^\circ} = \begin{bmatrix}
0 & 0 & 0 & -1 & 0 \\
0 & -2 & -4 & 0 & 1 \\
0 & 0 & 0 & 0 & 0 \\
-1 & 0 & 2 & 1 & 0 \\
0 & 0 & 0 & 0 & 0 
\end{bmatrix} * f
\]

\[
G_{67.5^\circ} = \begin{bmatrix}
0 & 0 & 0 & 0 & 0 \\
0 & -2 & -4 & -2 & 0 \\
0 & 0 & 0 & 0 & 0 \\
-1 & 0 & 2 & 1 & 0 \\
0 & 0 & 0 & 0 & 0 
\end{bmatrix}
\]

\[
G_{90^\circ} = \begin{bmatrix}
0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 
\end{bmatrix}
\]

\[
G_{112.5^\circ} = \begin{bmatrix}
0 & 0 & 0 & 0 & 0 \\
0 & -2 & -4 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 
\end{bmatrix}
\]

\[
G_{135^\circ} = \begin{bmatrix}
0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 
\end{bmatrix}
\]

\[
G_{157.5^\circ} = \begin{bmatrix}
0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 
\end{bmatrix}
\]
Through eight convolution templates, carry out the convolution weighting calculation of the image in the eight directions, and finally take the maximum value of the absolute value of the gradient in the eight convolution results as the output value of the current pixel. The expression is given by formula (6).

\[
G = \max \left\{ |G_{0^\circ}|, |G_{22.5^\circ}|, |G_{45^\circ}|, |G_{67.5^\circ}|, |G_{90^\circ}|, |G_{112.5^\circ}|, |G_{135^\circ}|, |G_{157.5^\circ}| \right\} \tag{6}
\]

### 2.3. Selection of adaptive threshold

The complete edge information can be detected by extending the Sobel operator template in multiple directions, and the edge positioning accuracy is higher, but similar to the classic Sobel algorithm, for images with noise superimposed, the edge detection effect of the extended multi-directional Sobel operator template is still unsatisfactory and the ability to resist noise is also poor. It is usually necessary to change the threshold several times to achieve an accurate edge detection result. Hence, the selection of the threshold is very important. Assuming that the threshold is selected too low, easy to judge the noise points as edge points, resulting in many false edges. Supposing that the threshold is selected too high, some edges with small gray values in the image will be ignored, resulting in many discontinuous edges.

This paper adopts a simple and efficient threshold adaptive algorithm to divide an image into many $3 \times 3$ matrix pixel windows to select a reasonable threshold, and quickly calculates the median gray level of the pixels in the $3 \times 3$ window as the adaptive threshold. It is assumed that the pixel data of the window neighborhood of the image is $P_{11}, P_{12}, P_{13}, P_{21}, P_{22}, P_{23}, P_{31}, P_{32}, P_{33}$, the specific calculation process as follows:

Figure 2. Implementation process of adaptive threshold selection

In Figure 2, for the $3 \times 3$ neighborhood pixel window of the image: first, sort the rows to obtain the maximum value (MAX), median value (MID), and minimum value (MIN) of each row. And then compare the values to get the minimum of the 3 maximum values (MAX_MIN), simultaneously, compare the 3 medians and 3 minimums in the 3 rows to get the median of the 3 medians (MID_MID) and the maximum value of the three minimum values (MIN_MAX). Finally, the three values (MAX_MIN, MID_MID, and MIN_MAX) obtained previously are compared, and the middle value (MID) is selected as the adaptive threshold $T$. Via this processing, each pixel of the image will obtain a corresponding threshold in real-time, and the edge detection gradient amplitude of each pixel is
compared with the threshold of that point to implement an edge detection algorithm with adaptive threshold.

3. FPGA implementation of improved edge detection algorithm

Hardware platform of this paper selects the EP4CE6F17C8 chip of the Altera Cyclone IV series as the system core chip to implement the adaptive threshold multi-directional edge detection algorithm mentioned above, and the other parts mainly include the CMOS-OV5640 camera, SDRAM chip, VGA display, etc. Software platform uses Intel (Altera) FPGA integrated development software Quartus 17.1 to finish the integrated FPGA design flow from design input to hardware configuration.

3.1. Hardware system architecture

Compared with the traditional Sobel edge detection algorithm, the extended multi-directional edge detection algorithm mentioned in this paper uses the $5 \times 5$ convolution kernel to detect multi-directional edges, and select the median value of the $3 \times 3$ neighborhood window to obtain the adaptive threshold of the pixel. The complexity of the algorithm increases, and the use of FPGA parallel design philosophy can greatly improve the processing speed and decrease computing time. Hardware system mainly includes the following parts: I2C bus configuration camera module, image data format conversion module, edge detection algorithm module, SDRAM controller module, VGA display controller module and PLL module.

![Figure 3. Block diagram of the hardware system structure](image)

Hardware structure block diagram of the system is shown in figure 3. In this system, first, configure the register of the CMOS-OV5640 camera through the I2C bus, and deploy the OV5640 into the RGB565 image output mode to output a 24 bit resolution color image. The image data format conversion module converts the image data format, converting RGB color images into grayscale images, and the PLL module is used to achieve frequency division of the main clock frequency. Subsequently, in the edge detection algorithm module, adaptive threshold algorithm and edge detection algorithm process the gray image, and then output the edge image. The specific implementation process of the edge detection algorithm module as follows: the gray image first passes adaptive threshold algorithm for image denoising and image adaptive threshold acquisition, and then eight-direction extended Sobel operator for convolution operation is performed and calculated for gradient amplitude value which is compared with the adaptive threshold, the edge of the detected image is output. The SDRAM chip accesses the extracted edge image, and finally displays the edge image through the VGA display.
3.2. The gradient calculation module of improved Sobel operator

In this module, in order to implement the improved gradient calculation of multi-direction Sobel edge detection, first the $5 \times 5$ pixel window template need to be acquired. This paper performs line delay via 4 shift registers (Line_Shift_RAM) in the IP core of the Quartus software. The image resolution in CMOS-OV5640 is $640 \times 480$, therefore, the depth of the row buffer is set to 640. Column data are delayed by 5 D flip-flops to achieve the simultaneous output of 25 data in the window, and the $5 \times 5$ sliding pixel window data can be output in parallel. The specific process diagram is shown in Figure 4.

![Figure 4. Schematic diagram of window data parallel output](image_url)

The 25 pixels passing through the $5 \times 5$ sliding window are convolved with the data in the extended Sobel operator template in 8 directions after the 25 data are output in parallel, and finally maximum value of the gradient amplitude is obtained through the comparator. The expression of convolution operation can be performed by Verilog HDL code. Since the data in the extended Sobel operator template are 0, 1, 2, 4, the multiplication operation can be replaced by shift operation, which can greatly save FPGA logic resources, simplify the design and increase the running speed. Taking the calculation of the gradient ($G_{45^\circ}$) in the $45^\circ$ direction as an example, the convolution operation code as follows:

```verilog
always @ (posedge clk or negedge rst_n) begin
    if(!rst_n) begin
        G45_temp1 <= 0;
        G45_temp2 <= 0;
        G45_data <= 0;
    end
    else begin
        G45_temp1 <= matrix_p25 + (matrix_p34<<2) + (matrix_p43<<2) + (matrix_p44<<1) + matrix_p52;
        G45_temp2 <= matrix_p14 + (matrix_p22<<1) + (matrix_p23<<2) + (matrix_p32<<2) + matrix_p41;
        G45_data <= (G45_temp1 >= G45_temp2) ? (G45_temp1 - G45_temp2) : (G45_temp2 - G45_temp1);
    end
end
```
end

3.3. Adaptive threshold calculation module

In this work, similarly, first the $3 \times 3$ pixel window template needs to be obtained to achieve an improved selection of adaptive threshold calculation. This work performs line delay by calling the two shift registers (Line_Shift_RAM) in the IP core of the Quartus software. Three D flip-flops can be used to delay the column data to implement the simultaneous output of 9 data in the $3 \times 3$ window in parallel, which will achieve the parallel output of sliding pixel window data. The specific process is shown in Figure 5.

![Figure 5. Schematic diagram of window data parallel output](image)

Median value of 9 pixels needs to be quickly selected as the adaptive threshold of the edge detection algorithm after the 9 pixels are output in parallel. In FPGA, using the parallel pipeline design philosophy, through the three-stage comparator, it can quickly compare in parallel to get the median value. The specific implementation process is shown in the following figure:

![Figure 6. Process diagram of adaptive threshold selection](image)

In Figure 6, the 9 data of 3 rows enter the comparators A1, A2, and A3 respectively by row. Subsequently, the maximum value of the data obtained by each row through the comparator A is sent to B1, and the median value of the obtained data by each row through the comparator A is sent to B2, and the minimum value of the data obtained by each row through the comparator A is sent to B3. Next, the 3 values obtained by the comparator B (the minimum value obtained by the comparator B1, the median value obtained by the comparator B2, and the maximum value obtained by the comparator B3) are sent to the comparator C in parallel. Finally, the median value obtained by the comparison C is output as the adaptive threshold T. Through the threshold adaptive algorithm, there is no need to set a fixed threshold, which can greatly reduce the errors caused by artificially setting the threshold based on subjective experience.
4. Experiment results and discussions
The RTL codes of each module synthesize, simulate, and compile on the Quartus software after the design of each module has been completed. Subsequently, the VGA display will display the edge image of the captured video image by OV5640 camera after the generated .sof file is downloaded to the FPGA board by the JTAG interface mode. We intercept a frame of the original gray video image and the edge image processed by several different edge detection algorithms to verify that the video image edge detection system designed in this paper can display the edge image normally on the VGA monitor, the experimental results as in Figure 7.

![Grayscale Image](image1.png) ![Traditional Sobel](image2.png) ![Eight-directions given threshold](image3.png) ![Eight-direction adaptive threshold](image4.png)

Figure 7. Comparison of detection algorithm results

Figure 7 (a) is a screenshot of the gray-scale video image via the image format conversion, (b) is the image effect after processing by the traditional Sobel edge detection algorithm, (c) is the image effect after processing by the extended multi-directional Sobel but threshold value given, (d) is the multi-directional Sobel edge detection effect map processed by the adaptive threshold proposed in this paper. The experimental results show that the improved algorithm presents further details, significantly continuous edge, and the edge contour is obvious and clear. Particularly, compared with the detection effect of a given threshold, the adaptive threshold detection method can detect more precise edges and small edges, which further enhances the edge detection effect.

The adaptive threshold multi-directional Sobel edge detection is implemented by FPGA, and the actual hardware system resource occupation as follows:

| LE  | Registers | Multiplier | PLL | Memory |
|-----|-----------|------------|-----|--------|
| 2036| 1059      | 0          | 2   | 45056  |

Figure 8. FPGA chip hardware resource occupation
5. Conclusions
As the basis of image recognition, image understanding, target tracking and other technologies, edge detection is widely used in artificial intelligence, deep learning and other fields. In this paper, the edge detection direction is increased to eight by extending traditional Sobel operator, and through adaptive threshold algorithm which makes each pixel get a corresponding threshold, and finally this eight-direction Sobel edge detection algorithm based on adaptive threshold is implemented on FPGA. It is found that the improved adaptive threshold edge detection algorithm is distinctly superior to the traditional Sobel edge detection algorithm by comparison and analysis of the results, and the accuracy and the effect of the detection results are greatly improved, making this algorithm have great advantages and ample application prospects in complex edges and high accuracy requirements situation.

The adaptive threshold multi-directional Sobel edge detection algorithm proposed in this paper has offered satisfactory edge detection results on FPGA, but there is still numerous space of optimization and improvement for image edge detection in different scenes. In the future, the combination of FPGA hardware and various embedded hard cores and software to achieve the joint design of software and hardware will be a promising research filed to adapt to various application scenarios and fulfil the stability and real-time requirements of each occasion.

Acknowledgments
The authors would like to thank the faculties and colleagues in our laboratory for their encouragement and the significant support in experimental equipment of our studies.

References
[1] Cherri AK, Karim MA. (1989) Optical symbolic substitution: edge detection using Prewitt, Sobel, and Roberts operators. Applied Optics, 28(21): 4644-4648.
[2] Chen Lei, Ye Yan. (2019) Multi-direction adaptive threshold edge detection algorithm and FPGA parallel implementation. Wireless Communication Technology, 28(04): 21-26.
[3] Zhang Shuaiyan. (2019) Design of image edge detection algorithm based on FPGA [D]. Hainan University. pp. 1-20.
[4] Li Tianci. (2016) A threshold adaptive edge detection algorithm and its FPGA implementation [D]. South China University of Technology. pp. 25-33.
[5] Zheng Yingjuan. (2014) Research on edge detection algorithm based on eight-direction Sobel operator[D]. Hebei Normal University. pp. 17-29.
[6] Zheng Yingjuan, Zhang Youhui, Wang Zhiwei, Zhang Jing, (2013) Fan Shengjuan. Edge detection algorithm based on eight-direction Sobel operator. Computer Science, 40(S2): 354-356.
[7] Zou Xiangxi, Zhang Yonghui, Zhang Shuaiyan, Zhang Jian. (2018) FPGA implementation of edge detection for Sobel operator in eight directions. In: IEEE Asia Pacific Conference on Circuits and Systems. Chengdu. pp. 520-523.