Research and Implementation of the HD Video Real-Time Edge Detection System Based on FPGA

Mingmin Ou, Jun Wang, Yi Wu and Jin Yi
College of Electrical Engineering and Automation, Fuzhou University, Fuzhou 350108, China.
Email: oumingmin0703@163.com

Abstract. In this paper, we propose a high-definition (HD) video real-time edge detection system, also presenting its hardware design and implementation results. Traditional median filtering blurs image details while removing noise, therefore we propose the improved median filtering algorithm which not only effectively removes impulse noise, but also retains more image details. For the problem that traditional sobel edge detection algorithm easily loses graphic details. This paper proposes the improved sobel edge detection algorithm which extends the gradient direction to eight, introducing local adaptive threshold and non-maximum suppression. The proposed algorithms were designed with Verilog-HDL, and the real-time operation was verified and evaluated using an field programmable gate array (FPGA) based test system at an operating frequency of 200MHz for 1280×720 HD video.

1. Introduction
As an important part of smart cities, the construction of intelligent transportation will promote the development of smart cities. The monitoring system [1] is a key component of intelligent transportation. With the widespread application of high-definition (HD) video surveillance, the amount of data has increased exponentially [2]. Obtaining favorable information from massive data is helpful for the system to make decisions. How to reduce the amount of data processed by the system without affecting the system decision has become the focus of current research. The edge is the boundary between the target and background, which not only greatly reduces the image data, but also retains important attributes of image [3]. At present, most edge detection systems are implemented based on general purpose computer. The efficiency of image processing by software serial method is quite low, which can not meet the real-time image processing of HD video.

In this paper, we propose the improved median filtering algorithm and improved sobel edge detection algorithm, and design a real-time edge detection system for HD video based on FPGA. The system uses the pipeline structure and makes full use of the FPGA's parallel processing capabilities. Experimental results show that the system can meet the real-time and accuracy of HD video processing. The remainder of this paper is organized as follows: Section 2 explains the improved algorithms. Section 3 describes the hardware architecture of the improved algorithms. Section 4 describes the results obtained implementing the system on an FPGA. Finally, Section 5 concludes this paper.

2. Algorithms Analysis
In this section, we introduce the improved median filtering algorithm and improved sobel edge detection algorithm in detail. Firstly, the improved median filtering algorithm is used to reduce the impact of noise, such as salt and pepper noise. Then, by increasing the gradient direction of the sobel
edge detection to eight, the gradient values in eight directions are calculated and compared with the adaptive threshold to distinguish the image edges.

In the process of image generation and transmission, the image is inevitably interfered by external factors, which results in the superposition of noise and the degradation of image quality [4]. Without noise pollution, the value of any pixel in the image is highly correlated with adjacent pixels. Salt and pepper noise, also known as impulse noise, is the black and white point noise generated by image sensor, transmission channel and decoding processing [5]. It randomly changes some pixel values so that the value of the target pixel is significantly different from the value of the nearby pixel, resulting in the degradation of image quality. The probability density function of the impulsive noise in the image can be expressed as follows:

\[
P(z) = \begin{cases} 
P_a, & z = a \\
P_b, & z = b \\
0, & \text{otherwise}
\end{cases}
\]  

(1)

In formula (1), if \( P_a \) or \( P_b \) is equal to zero, \( P(z) \) is a unipolar pulse; if both are non-zero and very close, then \( P(z) \) is a bipolar pulse.

2.1. Improved Median Filtering Algorithm

Aiming at the problem that traditional median filtering algorithm blurs image details while removing noise [6], this paper proposes an improved median filtering algorithm that not only effectively removes impulse noise, but also retains more image details. Bipolar impulsive noise [7] is characterized by black and white point in grayscale images, and their grayscale ranges are \([0, \alpha]\) and \([255 - \alpha, 255]\) respectively. The numerous experimental results demonstrate that when \( \alpha \) equal to 20, characteristics of black and white can be represented more accurately. The implementation process of the median filtering algorithm proposed in this paper is as follows:

Step one: The 3x3 filtering window is generated with target point as the center as shown in Figure 1, and all pixel values in window are sorted at the same time, and then the median value is calculated.

\[
\begin{pmatrix}
P_{11} & P_{12} & P_{13} \\
P_{21} & P_{22} & P_{23} \\
P_{31} & P_{32} & P_{33}
\end{pmatrix}
\]

Figure 1. 3x3 filtering window.

Step two: Determine the range of the pixel value of the target point. If it is located at \([\alpha, 255 - \alpha]\), keep it unchanged and go to Step four, otherwise go to Step three.

\[
M(i, j) = x(i, j) \\
\text{s.t } \alpha \leq x(i, j) \leq 255 - \alpha
\]  

(2)

Step three: Calculate the absolute value of the difference between the pixel value of the target point and the median value. If the absolute value of the difference is not greater than the threshold, it will remain unchanged; otherwise, replace the pixel value of the target point with the median value.

\[
e(i, j) = |x(i, j) - m(i, j)|
\]  

(3)

\[
M(i, j) = \begin{cases} 
x(i, j), & \text{s.t } e(i, j) \leq T \\
m(i, j), & \text{otherwise}
\end{cases}
\]  

(4)

Step four: End of the algorithm.
According to the above steps, the flow chart of the improved median filtering algorithm proposed in this paper is shown in Figure 2.

![Flow chart of the improved median filtering algorithm](image)

**Figure 2.** Flow chart of the improved median filtering algorithm.

The Matlab software is used to evaluate the performance of the improved median filtering algorithm. Figure 3 illustrates that the result of applying the traditional median filtering algorithm and improved algorithm to the image respectively. When the traditional median filtering algorithm is applied to image, it can be seen that most of the salt and pepper noise is removed and the image details become blurred. When the improved algorithm is applied to image, compared with the traditional algorithm, not only the denoising effect is obvious but more details of the image are retained.

![Performance of the improved median filtering algorithm](image)

**Figure 3.** Performance of the improved median filtering algorithm for image (a) original image (b) noisy image (c) traditional algorithm (d) improved algorithm.
2.2. Improved Sobel Edge Detection Algorithm

Edge is the essential feature of an image. It refers to a collection of pixels whose local pixel values change dramatically, usually existing between the target and background [8]. Sobel operator is a discrete difference operator that contains two sets of template matrices in the horizontal and vertical directions [9]. Through the convolution operation of template and image, the gradients in horizontal and vertical directions are obtained. The above two results are combined to obtain the approximate gradient of the target point, and then compared with threshold to determine the edge of image.

The traditional sobel edge detection algorithm only considers the gradients in the horizontal and vertical directions, and the threshold is fixed [10]. It is easy to lose image details when doing edge detection. Aiming at the above problems, this paper proposes the improved sobel edge detection algorithm: extending the gradient direction to eight as shown in Figure 4, introducing local adaptive threshold and non-maximum suppression.

$$\begin{pmatrix}
-1 & 0 & 1 \\
-2 & 0 & 2 \\
-1 & 0 & 1
\end{pmatrix} \begin{pmatrix}
0 & 1 & 2 \\
-1 & 0 & 1 \\
-2 & -1 & 0
\end{pmatrix} \begin{pmatrix}
1 & 2 & 1 \\
0 & 0 & 0 \\
-1 & -2 & -1
\end{pmatrix} \begin{pmatrix}
2 & 1 & 0 \\
1 & 0 & -1 \\
0 & -1 & -2
\end{pmatrix}$$

$$\begin{pmatrix}
1 & 0 & -1 \\
0 & -1 & -2 \\
-1 & -2 & -1 \\
-2 & -1 & 0
\end{pmatrix} \begin{pmatrix}
2 & 0 & -2 \\
1 & 0 & -1 \\
0 & 0 & 0 \\
-1 & 0 & 1
\end{pmatrix} \begin{pmatrix}
2 & 1 & 0 \\
1 & 2 & 1 \\
0 & 1 & 2
\end{pmatrix}$$

Figure 4. Gradient templates in eight directions.

The gradient calculation formula for the eight directions and the gradient synthesis formula of the target position \((i, j)\) are as follows:

$$G_0 = (P_{13} + 2P_{23} + P_{33}) - (P_{11} + 2P_{21} + P_{31})$$

$$G_{45} = (P_{12} + 2P_{32} + P_{22}) - (P_{21} + 2P_{31} + P_{32})$$

$$G_{90} = (P_{11} + 2P_{13} + P_{13}) - (P_{31} + 2P_{32} + P_{33})$$

$$G_{135} = (P_{12} + 2P_{31} + P_{21}) - (P_{23} + 2P_{33} + P_{32})$$

$$G_{180} = (P_{11} + 2P_{21} + P_{31}) - (P_{13} + 2P_{23} + P_{33})$$

$$G_{225} = (P_{21} + 2P_{31} + P_{32}) - (P_{12} + 2P_{13} + P_{23})$$

$$G_{270} = (P_{31} + 2P_{21} + P_{31}) - (P_{11} + 2P_{21} + P_{31})$$

$$G_{315} = (P_{23} + 2P_{33} + P_{32}) - (P_{21} + 2P_{11} + P_{21})$$

$$G(i, j) = \sum |G_k| \quad k = 0, 45, 90, 135, 180, 225, 270, 315$$

The adaptive local threshold for sobel edge detection at the target position \((i, j)\) proposed in this paper is defined by:

$$C(i, j) = \frac{1}{8} \times parm \times \sum_{i=1}^{3} \sum_{j=1}^{3} P_{ij}$$

$$s.t \quad \{ j \neq 2 \mid i = 2 \}, \quad parm = 4$$

$$T(i, j) = \sqrt{C(i, j)}$$

Where \(parm\) is variable parameter, initially set to 4. With the continuous movement of the 3×3 window, the target point and the local threshold are also constantly changing, thereby achieving different regions using different thresholds for edge detection.
The final edge binary image \( B(i, j) \) is defined as follows:

\[
B(i, j) = \begin{cases} 
255, & \text{if } G(i, j) > T(i, j) \\
0, & \text{otherwise}
\end{cases}
\]  

In an edge binary image, areas with large gradient changes usually have wider edges, which makes it difficult to distinguish the edges of the image clearly. In order to overcome this problem, non-maximum suppression algorithm (NMS) is used to suppress those redundant edges, so that the edges of the image are clearer. The NMS algorithm calculation formula is as follows:

\[
B'(i, j) = \begin{cases} 
255, & \text{if } B(i, j) > B(i, j-1) \& \& B(i, j) > B(i, j+1) \& \& (k = 0 \mid k = 180) \\
255, & \text{if } B(i, j) > B(i+1, j-1) \& \& B(i, j) > B(i-1, j+1) \& \& (k = 45 \mid k = 225) \\
255, & \text{if } B(i, j) > B(i-1, j) \& \& B(i, j) > B(i+1, j) \& \& (k = 90 \mid k = 270) \\
255, & \text{if } B(i, j) > B(i-1, j-1) \& \& B(i, j) > B(i+1, j+1) \& \& (k = 135 \mid k = 315) \\
0, & \text{otherwise}
\end{cases}
\]  

Firstly, gradients of eight directions in formula (5) are sorted to determine direction \( k \) of maximum gradient. Then, the target point is compared with two adjacent pixels in this direction. If grayscale value of the target point is greater than two adjacent pixels, grayscale value of target point remains unchanged; otherwise it is set to 0. In the above formula, \( B'(i, j) \) is the edge binary image processed by the NMS algorithm. The NMS algorithm is used to locate the edge of image in a width of one or two pixels, and the purpose of distinguishing the edge of image is achieved visually.

Figure 5 shows that the result of applying the traditional sobel edge detection algorithm and improved algorithm to the image respectively. Compared with traditional algorithm, the improved algorithm in this paper can detect more edge details of image and the edges are more exquisite.
3. Hardware Architecture Design
A real-time edge detection for HD video is implemented in the proposed system based on FPGA. The overall system design is shown in Figure 6, including video data acquisition, FIFO data buffering, DDR3 data storage, image preprocessing, control algorithm implementation, and HDMI video signal display, etc. The ov5640 camera captures real-time video data and output RGB format data stream after encoding. The received video data is decoded in the FPGA and buffered into the FIFO. The vdma controller will start to store the video data into external DDR3 memory when the rising edge of the vertical synchronizing signal. Driven by hdmi timing, the video data in DDR3 is read out and processed by improved median filtering algorithm, improved sobel edge detection algorithm and morphological operations to obtain high-quality real-time edge images and display them on the HDMI display.

![Figure 6. Overall scheme of the proposed HD video real-time edge detection system.](image)

3.1. Fast Median Filtering Architecture
Figure 7 shows that two shift registers are used to buffer data for generating the 3×3 matrix window. The width of the shift register is set to 8 and the depth is set to 1024. The two shift registers are connected in series with the current input data to form the three lines of the image. Then generate the 3×3 matrix window centered on the target point through the ordinary register buffer inside the FPGA.

![Figure 7. Block diagram of the 3×3 matrix window generation.](image)

In the stage of median value calculation, parallel and pipeline structures are applied to improve the operating efficiency. The processes of data sorting in each row and column of the 3×3 matrix window are performed in parallel as shown in Figure 8. It only takes three clock cycles to complete the median calculation of the 3×3 matrix window. As the video data stream continues to flow in, the 3×3 matrix window will move backwards and generate the local median of the window at the same time.

![Figure 8. Block diagram of the median value calculation in the 3×3 matrix window.](image)
3.2. Edge Detection Architecture

Figure 9 shows the architecture of improved sobel edge detection algorithm, which consists of $3 \times 3$ matrix window generation, template convolution, gradient synthesis, adaptive threshold calculation, and non-maximum suppression. Initially, the $3 \times 3$ matrix window is generated with target point as the center, and the gradient of target point is calculated according to formulas (5) and (6). The average value of pixels near target point is calculated at the same time, and the local threshold is determined according to formula (7) and (8). Afterwards, gradient value of target point is compared with a threshold to generate the edge binary image. Finally, the edge binary image is used for non-maximum suppression.

$\begin{align*}
&x(i, j) \\
&3 \times 3 \text{ window generation} \\
&P_0 \rightarrow \text{template convolution} \rightarrow G_k \rightarrow \text{gradient synthesis} \rightarrow G_0 \rightarrow \text{binary image} \rightarrow B_0 \rightarrow \text{non-maximum suppression} \\
&\rightarrow B'(i, j) \\
&\rightarrow \text{average calculation} \rightarrow C_0 \rightarrow \text{threshold calculation} \rightarrow T_0 \\
&\end{align*}$

**Figure 9.** Block diagram of the improved sobel edge detection algorithm.

As shown in Figure 10, each element $P_0$ of the $3 \times 3$ matrix window performs the product operation simultaneously with the corresponding gradient operator $G_0$, and the operation results are added to obtain the gradient of target point. In the template convolution and gradient calculation process, the pipeline structure is used, and the data is processed in parallel, which greatly improves the real-time performance of the entire system.

**Figure 10.** Block diagram of the template convolution and gradient synthesis.

4. FPGA Implementation Results

The improved median filtering algorithm and improved sobel edge detection algorithm for 1280×720 HD video are proposed in this paper, and they are synthesized and implemented on a Xilinx Artix 7 (XC7A200T) FPGA. Table 1 summarizes the results of the implementation based on target hardware. In the FPGA test system with the clock frequency of 200 MHz, the proposed improved algorithms were implemented with 11045 LUTs, 9456 FFs, 4 DSPs, and 15 BRAMs. The results of our proposed HD video real-time edge detection system running on FPGA are shown in Figure 11. It can be seen from the experimental results that the system has completed the real-time video acquisition, encoding and decoding, data buffering, algorithm processing, and video display. Compared with the traditional algorithm, the edge detected by the improved algorithm is more clear. At the same time, we also have verified that real-time processing is possible at the 1280×720 HD image resolution.
Figure 11. Diagrams of the experimental result (a) real-time image (b) grayscale image (c) traditional algorithm (d) improved algorithm.

Table 1. Performance of the proposed HD video real-time edge detection system implemented on Artix 7 (XC7A200T) FPGA.

| Resources | Utilization | Utilization % |
|-----------|-------------|---------------|
| LUTs      | 11045       | 8.25          |
| FFs       | 9456        | 3.53          |
| DSPs      | 4           | 0.54          |
| BRAMs     | 15          | 4.11          |

5. Conclusion
The HD video real-time edge detection system was proposed, and the results of its implementation were presented. Compared with the traditional sobel edge detection algorithm, the improved algorithm proposed in this paper has higher detection accuracy and higher processing speed in real-time HD video. The proposed HD video real-time edge detection system was implemented on the FPGA, and its 30fps real-time processing for 1280×720 HD video was verified at an operating frequency of 200MHz.

6. References
[1] C. Zhang, T. Sun, J. Chen and P. Li, "A visibility monitoring system utilizing roadside video camera facilities for highway systems," 2017 3rd International Conference on Information Management (ICIM), Chengdu, 2017, pp. 486-490.
[2] J. Wu, R. Tan and M. Wang, "Streaming High-Definition Real-Time Video to Mobile Devices with Partially Reliable Transfer," in *IEEE Transactions on Mobile Computing*, vol. 18, no. 2, pp. 458-472, 1 Feb. 2019.

[3] E. Dong, K. Li and J. Tong, "FPGA Based Design and Implementation of Improved Edge Detection Algorithm using LOG Operator," 2018 *IEEE International Conference on Mechatronics and Automation (ICMA)*, Changchun, 2018, pp. 2092-2096.

[4] I. Landa, M. M. Vélez, A. Arrinda, R. Torre and M. Fernández, "Impulsive noise characterization and its effect on digital audio quality," 2015 *IEEE International Symposium on Broadband Multimedia Systems and Broadcasting*, Ghent, 2015, pp. 1-3.

[5] Y. Wang, J. Wang, X. Song and L. Han, "An Efficient Adaptive Fuzzy Switching Weighted Mean Filter for Salt-and-Pepper Noise Removal," in *IEEE Signal Processing Letters*, vol. 23, no. 11, pp. 1582-1586, Nov. 2016.

[6] K. Han, Z. Wang and Z. Chen, "Fingerprint Image Enhancement Method based on Adaptive Median Filter," 2018 24th Asia-Pacific Conference on Communications (APCC), Ningbo, China, 2018, pp. 40-44.

[7] M. Rakhshanfar and M. A. Amer, "Low-Frequency Image Noise Removal Using White Noise Filter," 2018 25th *IEEE International Conference on Image Processing (ICIP)*, Athens, 2018, pp. 3948-3952.

[8] M. Mittal *et al.*, "An Efficient Edge Detection Approach to Provide Better Edge Connectivity for Image Analysis," in *IEEE Access*, vol. 7, pp. 33240-33255, 2019.

[9] X. Wang, H. Ma, X. Chen and S. You, "Edge Preserving and Multi-Scale Contextual Neural Network for Salient Object Detection," in *IEEE Transactions on Image Processing*, vol. 27, no. 1, pp. 121-134, Jan. 2018.

[10] E. Onat, "FPGA implementation of real time video signal processing using Sobel, Robert, Prewitt and Laplacian filters," 2017 25th *Signal Processing and Communications Applications Conference (SIU)*, Antalya, 2017, pp. 1-4.