IJISET - International Journal of Innovative Science, Engineering & Technology, Vol. 2 Issue 6, June 2015.
www.ijiset.com
ISSN 2348 – 7968

Implementation of Canny Edge Detector Algorithm using FPGA

Poonam S. Deokar
M.E Student,MCEORC, Nashik, India,
poonamdeokar19@gmail.com

Abstract: Edge can be defined as discontinuities in image intensity from one pixel to another.Edge detection is one of the most fundamental algorithms in digital image processing .The Canny algorithm computes the higher and lower thresholds for edge detection based on the entire image statistics, which prevents the processing of blocks independent of each other. Canny algorithm is used as a standard from many years. However, this is a time consuming algorithm. Here a new approach is presented to calculate gradient using approximate method which reduces the latency and resource utilization. The proposed algorithm is implemented using FPGA. Synthesis and image analysis results are presented in this paper.
.

Keywords: FPGA, Canny Edge Detector, Latency, Threshold.

1. INTRODUCTION

The purpose of edge detection in general is to significantly reduce the amount of data in an image, while preserving the structural properties to be used for further image processing. Edge detection has been widely applied in various different computer vision systems. It is used to identify changes in luminosity of the image, changes in the intensity due to changes in scene structure. Using software, Edge detection algorithms are implemented, and their hardware implementation is possible with Very Large Scale Integration (VLSI) technology, for real-time applications [1].
The Canny edge detector is used from last few years for edge detection and has better performance than previous algorithms. Canny edge detection algorithm is also known as the optimal edge detector. Canny's intentions were to enhance the many edge detectors in the image. Implementation of Canny algorithm with hardware has high latency and cannot be employed in real-time applications [2]. Another shortcoming of the commonly used Canny algorithm is that the computational cost of the algorithm is very high and it cannot be implemented in real time to meet the needs of mobile robot vision system.
To overcome these shortcomings of canny algorithm, a new approach is proposed in this paper, which computes the approximate value of gradient. Approximate calculation reduces the cost of design as well as resource utilization is also optimized to great extent. Proposed algorithm is tested on the various images to analyze the performance of the algorithm.

2. CANNY EDGE DETECTION

The block diagram of the canny edge detection algorithm is shown in Fig. The Canny algorithm [3] consists of the following steps:

1. Smoothing the input image.
2. Calculating the horizontal gradient Gx and vertical gradient Gy at each pixel location.
3. Computing the gradient magnitude G and direction θG at each pixel location.
4. Applying Non-Maximal Suppression (NMS) to thin edges.
5. Computing high and low thresholds based on the histogram of the gradient magnitude for the entire image.
6. Performing hysteresis Thresholding.

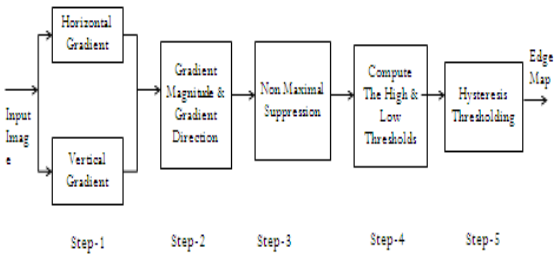

Fig. 1: *Block Diagram of the Canny Edge Detection Algorithm*

1. Smoothing – Smoothing of the image is achieved by median filter to remove noise.

ISSN 2348 – 7968

2 .Calculation of Gx and Gy - It is performed using the Sobel Operator. The Sobel operator uses a pair of 3x3 convolution masks, one estimating the gradient in the x-direction (columns) and the other estimating the gradient in the y-direction (rows).

3. Gradients and direction calculation -

The approximate absolute gradient magnitude (edge strength) at each point can be found by the formula which is simpler to calculate compared to the equation of exact gradient magnitude.

Exact formula is given by,

$$G(magnitude) = \sqrt{Gx^2 + Gy^2}$$

Approximate calculation can be done by,

G(approximate)= 15\16 * max(Gx,Gy) + 0.4* min(Gx,Gy).

The formula for finding the edge direction is given below:

theta = invtan (Gy / Gx)

4.Non-Maximal Suppression - Once the direction of the gradient is known, the pixel that has no local maximum gradient magnitude is eliminated. If the pixel's gradient direction is one of 8 possible main directions the gradient magnitude of this pixel is compared with two of its immediate neighbors along the gradient direction and the gradient magnitude is set to zero if it does not correspond to a local maximum. ng gradients.

5. *Threshold Calculation* - The high threshold is computed such that a percentage p1 of total pixel in the image would be classified as Strong edge. The high threshold corresponds to the point at which value of gradient magnitude is Cumulative distributive function (CDF) equals to 1- p1. The low threshold is calculated as percentage p2 of high threshold.

6. *Hysteresis Threshold* - If the gradient magnitude of pixel is greater than high threshold then this pixel is considered as strong edge. If the gradient magnitude of pixel is between high and low threshold then this pixel is considered as weak edge. Hysteresis used to determine the edge map.

3. CANNY EDGE DETECTION ALGORITHM USING FPGA

The Embedded system for implementing the Canny edge detection algorithm based on an FPGA platform.

Our FPGA Architecture design follows the exact procedure of the Canny algorithm, therefore our implementation has 5 blocks[5]:

- Smoothing
- Sobel Gradient calculation
- Non Maximum Suppression
- Thresholding
- Hysteresis

3.1 Smoothing

The digital image is first smoothened using median filter to remove noise. In this implementation 3x3 image matrix is used to achieve parallel filtering. Median filtering with FPGA needs 3 line FIFO ,6 D-type Flip-flops with controller.

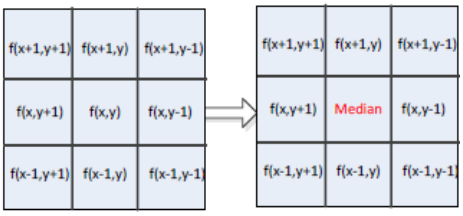

Fig. 2: Median Filtering

3.2 Sobel Gradient Calculations

To calculate horizontal and vertical gradient 3x3 image matrix is multiplied with the Sobel operator.

I1	I2	I3
I4	I5	I6
I7	I8	I9

m1	m2	m3
m4	m5	m6
m7	m8	m9

Fig. 3: Image matrix and mask

$$y = i1m1 + i2m2 + i3m3 + \cdots \ldots + i9m9$$

$$y = \sum_{a=1}^{9} iama$$

To implement horizontal and vertical gradient blocks in FPGA[5], we require Adder, Subtractor, Multiplier, Comparator. Input to this blocks are image pixels. Output of this comparator is used to calculate the Gradient magnitude. It calculates approximate root value. FPGA hardware requires multiplier, comparator to calculate the gradient value.

Exact formula is given by,

$$G(magnitude) = \sqrt{Gx^2 + Gy^2}$$

Approximate calculation can be done by,

G(approximate)= 15\16 * max(Gx,Gy) + 0.4* min(Gx,Gy).

ISSN 2348 – 7968

3.3 Non-Maximal Suppression

The formula for finding the edge direction is given below:
theta = invtan (Gy / Gx)
Horizontal and vertical gradient magnitudes are fetched from memory and used as input to NMS unit. It computes gradient direction at each pixel. Gradient magnitudes of gradient magnitudes of four nearest neighbors along the direction are selected to compute two intermediate gradients. Gradient magnitudes of four nearest neighbors along the direction are selected to compute two intermediate gradients. The final gradient magnitude after directional NMS (marked as Mag_NMS(x, y) is stored back into local memory and used as the input for the hysteresis thresholding unit.

3.5 Calculation of Thresholds

The Canny edge detection algorithm uses double thresholding. Edge pixels stronger than the high threshold are marked as strong; edge pixels weaker than the low threshold are suppressed and edge pixels between the two thresholds are marked as weak. Thresholding is executed by a double comparator. The high threshold is ThH, low threshold is computed as 40% of the high threshold.

3.6 Thresholding with Hysteresis

Strong edges are interpreted as "certain edges", and can immediately be included in the final edge image. Weak edges are included if and only if they are connected to strong edges. Hysteresis is basically a procedure of comparing the value of the pixel at hand with the values of three neighboring pixels above it and the one directly on the left. If the pixel is a possible edge and one of the aforementioned neighboring pixels is a definite edge, then the pixel becomes a definite edge. Otherwise it is left as is.

4. EXPERIMENTAL RESULTS

Experimental results include the testing on images on MATLAB to select the proper operator for gradient calculation. Proposed algorithm is tested on FPGA Platform Virtes-4 is used.

4.1 MATLAB Results

Different operators like Prewitt, Robert, Sobel, and Laplacian are applied on the Lena and Cameraman images to check the performance. Simulation Results shows the performance of Sobel operator is better, hence it is selected for gradient calculation.

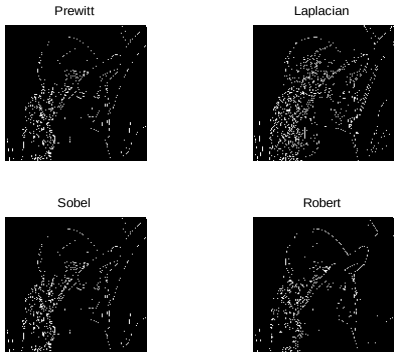

Fig.4: Lena image output with different operator

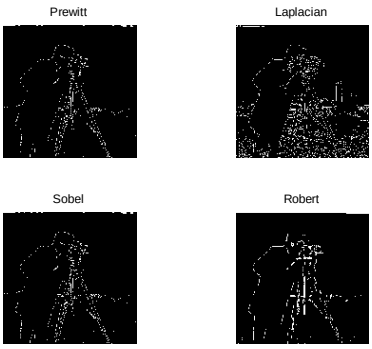

Fig.5: Cameraman image output with different operator

4.2 FPGA Implementation Results

Table shows the resource utilization summary for gradient calculation. Present Canny algorithm uses FIR Filter IP for the calculation of gradient, proposed canny algorithm implementation uses the approximate gradient calculation. So, it is observed that resource utilization is reduced by almost 30%, ultimately cost of the design is also reduced.

TABLE 1: Resource Utilization Summary

Logic Utilization	Using IP Core	Discrete Implementation	Optimization	% Optimization
Number of Slice Flip Flops	317	230	87	27
Number of occupied Slices	268	183	85	32
4 input LUTs	217	147	70	32

ISSN 2348 – 7968

Image results

a) *b)*

Fig.5: a) Original Image b) Output Image

CONCLUSION

In this paper, Canny algorithm with the approximate calculation of the gradient is presented. This proposed method is implemented and tested on FPGA platform. Taste images are taken from standard database. It is observed that algorithm has better edge detection performance with reduced latency. As system involves approximate calculation it increases speed of operation, also hardware required for the calculation is also optimized. It reduces computation cost as compared to original canny edge detection algorithm.

REFERENCES

[1] Wenhao He and KuiYuan ,“ An Improved Canny Edge Detector and its Realization on FPGA” Proceedings of the 7th World Congress on Intelligent Control and Automation June 25 - 27, 2008, Chongqing, China.

[2] Christos Gentsos, Calliope-LouisaSotiropoulou and Spiridon Nikolaidis Nikolaos Vassiliadis “Real-Time Canny Edge Detection Parallel Implementation for FPGAs” 978- 1-4244-8 157 -6/ 1 0 ©20 10 IEEEICECS 20 10 499-502.

[3] “Canny Edge Detection”, 09gr820, March ,2009.

[4] Caixia Deng, Weifeng Ma, Yin Yin ,“ An Edge Detection Approach of Image Fusion Based on Improved Sobel Operator”, 2011 4th International Congress on Image and Signal Processing.

[5] Li song, Kang He, “Parallel Hough Transform-Based Straight Line Detection and Its FPGA Implementation in Embedded Vision”,Sensor 2013,13,9223-9247.

[6] Shashidhar Ram Joshi,Roshan Koju,”Study and Comparison of Edge Detection Algorithm”, 978-1-4673-2590-5/12©2012 IEEE.