Hardware Implementation of Edge Detection Algorithms

[1]Vaishnav Tej Akhil, [2]Prof.Amit Kumar, [3]Prof.Ekta Chotai

Electronics and Communication Department, Marwadi College of Engineering, GTU, Rajkot, India

[1]Vaishnavtej9@gmail.com, [2]Amit.kumar@marwadieducation.edu.in, [3]Ekta.chotai@gmail.com

***Abstract*—An Edge detection algorithm is used in Image processing in order to reduce the data to be processed. It is widely used in various real time applications such as traffic signaling, number plate detection, tumor detection etc. So to use it in real time we require hardware implementation of it. In this paper we have presented details about how to implement an Edge detection algorithm by loading an image into FPGA using VHDL.An image is loaded into FPGA kit through few ways which is mentioned in this paper and the most effective way is utilized in the end through which there is computational ease and less memory requirement.**

I. INTRODUCTION

In today's era edge detection is widely used in much application. Many types of edge detection techniques are available. Edge detection is a type of image segmentation techniques which determines the presence of an edge or line in an image and outlines them in an appropriate way. The main purpose of edge detection is to simplify the image data in order to minimize the amount of data to be processed.

Different types of edge detection algorithms are

1. ROBERTS
2. PREWITT
3. SOBEL
4. CANNY

The principle of edge detection is: Edge widely exists between objects and backgrounds, objects and objects. Therefore, the general method of edge detection is to study the changes of a single image pixel in a gray area, use the variation of the edge neighboring first order or second-order to detect the edge.

II. VARIOUS EDGE DETECTION ALGORITHMS

A. *Roberts and Prewitt Operator*

The Roberts Cross operator performs a simple, quick to compute, 2-D spatial gradient measurement on an image. Pixel values at each point in the output represent the estimated absolute magnitude of the spatial gradient of the input image at that point.The operator consists of a pair of 2×2 convolution kernels as shown in Figure. One kernel is simply the other rotated by 90 degree.

$$H_1 = \begin{bmatrix} 1 & 0 \\ 0 & -1 \end{bmatrix} \quad H_2 = \begin{bmatrix} 0 & 1 \\ -1 & 0 \end{bmatrix}$$

Figure 1: Robert mask

These kernels are designed to respond maximally to edges running at 45° to the pixel grid, one kernel for each of the two perpendicular orientations. The kernels can be applied separately to the input image, to produce separate measurements of the gradient component in each orientation (call these *H1* and *H2*). These can then be combined together to find the absolute magnitude of the gradient at each point and the orientation of that gradient.

Prewitt operator is given as

$$H_1 = \begin{bmatrix} -1 & 0 & 1 \\ -1 & 0 & 1 \\ -1 & 0 & 1 \end{bmatrix} \text{ and } H_2 = \begin{bmatrix} -1 & -1 & -1 \\ 0 & 0 & 0 \\ 1 & 1 & 1 \end{bmatrix}.$$

Figure 2 :Prewitt mask

Robert's operator have a big disadvantage that it has no fix centre as it 2 by 2 mask. It is more prone to noise. Prewitt's operator have good noise reduction capacity but it does not have any special effect around center pixel. It is same for all the pixels.

B. *Canny Operator*

The Canny algorithm uses an optimal edge detector based on a set of criteria which include finding the discontinuities by identifying strong edges, and preserving the relevant weak edges, in addition to maintaining some level of noise suppression. While the results are desirable, the hysteresis stage slows the overall algorithm down considerably. The performance of the Canny algorithm depends heavily on the adjustable parameters, σ, which is the standard deviation for the Gaussian filter, and the threshold values, th and tl. σ also controls the size of the Gaussian filter. The bigger the value for σ, the larger the size of the Gaussian filter becomes. This implies more blurring, necessary for noisy images, as well as detecting larger edges. As expected, however, the larger the scale of the Gaussian, the less accurate is the localization of the edge. Smaller values of σ imply a smaller Gaussian filter which limits the amount of blurring, maintaining finer edges in the image.

The Canny Edge Detection Algorithm has the following Steps: Step 1: Smooth the image with a Gaussian filter.

Step 2: Compute the gradient magnitude and orientation using finite- difference approximations for the partial derivatives.

C. *Sobel Operator*

The Sobel operator is a classic first order edge detection operator computing an approximation of the gradient of the image intensity function. At each point in the image the result of the Sobel operator is the corresponding norm of this gradient vector. The Sobel operator only considers the two orientations which are 0 and 90 degrees convolution kernels.

-1	0	1
-2	0	2
-1	0	1

(a)

-1	-2	-1
0	0	0
1	2	1

(b)

Figure 3 : Sobel mask

The Sobel operator performs a 2-D spatial gradient measurement on an image and emphasizes regions of high spatial gradient that correspond to edges. Typically it is used to find the approximate absolute gradient magnitude at each point in an input grayscale image.
Compared to other edge operator, Sobel has two main advantages**:**

1. Since the introduction of the average factor, it has some smoothing effect to the random noise of the image.
2. Because it is the differential of two rows or two columns, so the elements of the edge on both sides has been enhanced, so that the edge seems thick and bright.

III. HARDWARE IMPLEMENTATION

Any edge detection operator or algorithm is a software so to work on it in real time we require a hardware implementation of it through some processors. FPGA or VHDL have parallel processor architecture so their speed is more and area occupied is less as they are application specific compared to other multipurpose processors.

Application specific hardware designs enjoy exponential gains in computation speed, compared to equivalent software designs, since they can take advantage of the highly parallel nature of their architecture. The main thing is to interface the matlab with xilinx for a real time image for its application in real time which requires a hardware description language for which we are using VHDL[4] [5] [6].

System Generator can be useful when implementing the design flow as described by With Simulink, with its blocksets, and Matlab code we can explore an algorithm in a functional domain. Using System Generator blocks (Xilinx blockset) we can also design in the functional domain, but towards an FPGA implementation (physical domain). System Generator provides interfaces to CAD tools (ISE, Model Sim). System Generator automatically compiles designs into low-level representations. Experiments using hardware generation can suggest the hardware speeds that are possible and, through the resource estimation, give a rough idea of the cost of the design in hardware. If a promising approach is identified, system generator can create the bit stream (physical level) to the FPGA.

System Generator allows refinements (lower levels) to be done in steps. Some portions of the design can be made ready for implementation in hardware, while others remain described by Simulink blocks, System generator blocks or were designed outside and are inside a wrapper to be wired in the System Generator tool. The mixed descriptions can be simulated in a multiple environment.
System Generator offers mechanisms to:

1) To import HDL code into a design. A configuration wizard can be used to associate the HDL module to a Black Box block. The wizard creates an M-function that defines the interface, the implementation and the simulation behavior of the black box block it is associated with.
2) To automatically generate an HDL test bench, including test vectors. Upon requested, System Generator generates a test bench that produces files to allow comparisons of simulation results between Simulink and Model Sim(HDL Simulator). The test bench is a wrapper that feeds the stimuli to the HDL for the design and compares HDL results against expected ones.
3) To perform hardware co-simulations, hardware run under the control of Simulink, bringing the power of MATLAB and Simulink to bear for data analysis and visualization. For hardware Co-Simulation, a bit stream is created and associated to a block. When the design is simulated in Simulink, results for the compiled portion are calculated in hardware.

The most efficient and convenient way is to load an image directly into FPGA kit. We can use Altera or Spartan kit according to our requirement and use VHDL code to implement edge detection. So in the end it generates a bit stream file which can be used through hardware in real time applications.

The image loaded is stored in FPGA RAM which is temporary. For any image with high resolution or size we can use MATLAB for interfacing through simulink otherwise direct loading is a feasible way to interface. It depends on the image how much variables are required and of Boolean data type in VHDL.

IV. CONCLUSION

We conclude that we have studied a lot about edge detection algorithms and their applications .Out of them we conclude that Sobel algorithm gives the best result. We have implemented this software based algorithms on hardware through Spartan/Altera kit using VHDL as a hardware description language. Through this we can have a real time application of these algorithms and improvisation can be done in future.

REFERENCES

[1] G.T. Shivrakshan "A Comparison of various Edge Detection Techniques used in Image Processing" IJCSI International Journal of Computer Science issues volume 9.
[2] Wenshuo Gao, Lei Yang, Xiaoguang Zhang, Huizhong Liu "An improved Sobel algorithm",IEEE.
[3] Nick Kanopoulos," Design of an Image Edge Detection Filter Using the Sobel Operator",IEEE journal of solid-state circuits, vol. 23.
[4] Fuming Sun, Jing Liang, Xiaoling Li, Qin Wang,"Texture Simulation and Implementation Based on Matlab & Simulink.
[5] G. Anusha, Dr.T. JayaChandra Prasad, Dr.D. Satya Narayana ,"Implementation of SOBEL Edge Detection on FPGA", International Journal of Computer Trends and Technology-volume3.
[6] Craig Moore, Harald Devos and Dirk Stroobandt" Optimizing the FPGA Memory Design for a Sobel Edge Detector".