Edge Detection on Field Programmable Gate Array (FPGA)

Pallavi Sahay[1], Avinash Kumar Singh[2], Diksha Sahay[3]

[1,2]Assistant Professor, Department of Electronics & Communication Engineering, Dr. C.V. Raman University, Vaishali, Bihar

[3]Department of MCA, Indira Gandhi National Open University (IGNOU)

Abstract

Edges are significant local changes in the image and are very important features for analysing images. Edges typically occur on the boundary between two different regions in an image. An edge in an image is a significant local change in the image intensity, usually associated with a discontinuity in either the image intensity or the first derivative of the image intensity**.** Edge detection is one of the tools that can be used in many image processing applications for withdrawing information from image. Sobel edge detection is one of the operators that operate on gradient based edge selection methods to find edge pixels in the image. This report proposed an implementation of Sobel edge Detection algorithm to find the pixels in grayscale images. We present a video streaming architecture and IP implementation using high level synthesis. A video with 720p resolution streamed from the HDMI source and real time edge detected video captured on another monitor. For implementation, ZYBO board or kit is used with Vivado software with 2018.3 V, which has provided adequate peripherals for implementation.

Keywords: Sobel, Edge, FPGA, High Level Language, Gradient, Register Transfer Level, IP

1. Introduction:

This report presents an architecture for Sobel Edge detection on the Field Programmable Gate Array (FPGA) boardor ZYBO board with Zynq 7000 series with the help of Vivado 2018.3 version of software, which is inexpensive in terms of computation. Hardware implementation of the Sobel Edge detection Algorithm is chosen because hardware presentation is a good scope of parallelism over software. On the other hand, Sobel Edge detection can work with less deterioration in high levels of noise. Here we describe the development of image processing hardware and implementation of the same on FPGA board. Here the stages of Sobel Edge Detection Algorithm is developed in Vivado 2018.3V HLS tool and then exported to use it as an IP. The implementation of this generated IP is synthesized and the generated Bitstream is verified and implemented on ZYBO Z7000 Board.

Digital image processing is an ever expanding and dynamic area with applications reaching out into our everyday life such as medicine, space exploration, surveillance , authentication, automated industry inspection and many more areas like intelligent transport systems, autonomous vehicles and self-guided armaments. Application such as these involves different processes like image enhancement, object detection etc. Basically, Edge detection is a mathematical method to detect points in the image where the brightness of the image changes. Edge detection refers to the process of identifying and locating sharp discontinuities in an image. The discontinuities are abrupt changes in pixel intensity which characterizes boundaries of an object in a scene. Classical methods of Edge detection involve convolving an image with an operator the 2-D filter, which is constructed to be sensitive to large gradients in an image while returning values of 0 in uniform regions. Typical Edge detection techniques are Sobel, Prewitt, Fuzzy logic, Roberts and Canny Edge detection.

2. Steps in Edge Detection:

There are many steps in Edge detection i.e categories in following four types as:

1. **Filtering:** Since gradient computation based on intensity values of only two points are susceptible to noise and other vagaries in discrete computations, filtering is commonly used to improve the performance of an edge detector with respect to that of the noise. However, there is a trade-off between edge strength and noise reduction. More filtering is done to reduce noise results in a loss of edge strength.
2. **Edge enchancement:** In order to facilitate the detection of edges, it is essential to determine changes in intensity in the neighbourhood of a point. Enhancement emphasizes pixels where there is a significant change in local intensity values and is usually performed by computing the gradient magnitude.
3. **Edge detection:** We want only points with strong edge content. However, many points in an image have a nonzero value for the gradient, and not all of these points are edges for a particular application. Therefore, some method should be used to determine which points are edge points. Frequently, Thresholding provides the criterion used for detection of a particular type of edges with predefined considered values.
4. **Edge localization:** The location of the edge can be estimated with sub pixel resolution if required for the application. The edge orientation can also be estimated.

3. Sobel Edge Detection:

Sobel operator is a discrete differentiation operator used to calculate an approximation of the gradient of an image intensity function for Edge detection. At each pixel of an image, it gives either the corresponding gradient vector or normal to the vector. This convolves the input image with the kernel and computes the gradient magnitude and direction.

Sobel Edge detection can be implemented by filtering an image with a left mask or kernel. Filter the image again with the other mask. After this square of the pixels values of each filtered image. Now add the two results and compute their root. The 3x3 convolution masks for the Sobel based operator as shown.

$$Gx = \begin{array}{|c|c|c|}\hline +1 & 0 & -1 \\ \hline +2 & 0 & -2 \\ \hline +1 & 0 & -1 \\ \hline \end{array} \Leftrightarrow \text{Image}$$

$$Gy = \begin{array}{|c|c|c|}\hline +1 & +2 & -1 \\ \hline 0 & 0 & 0 \\ \hline -1 & -2 & -1 \\ \hline \end{array} \Leftrightarrow \text{Image}$$

Fig.1: Convolution masks for Sobel Operator

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

$\theta = a \tan \left(\frac{Gy}{Gx}\right)$ …………………..(ii)

Using this above information, we are able to find the gradients' direction.

These values of images are convolved with the Sobel X and Sobel Y operations and then the gradient is calculated to have the final output values of the edge detected image. In the Sobel operator block, we have generated a combinational logic for performing arithmetic operation, which provides gradient of the image as a result i.e. output. In other hand we can say that Sobel edge detector is a window- based operator which requires pixel neighborhood information for computing the edge map of a particular pixel.

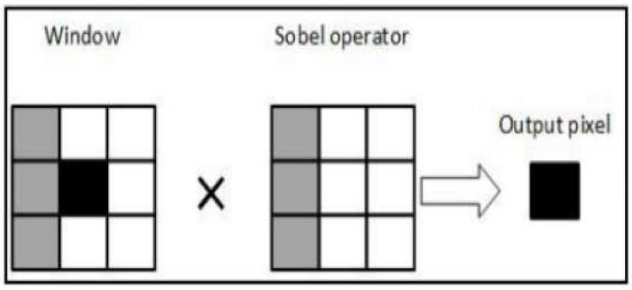

Fig.2: Implemetation of Sobel opreator

After applying the above Edge detection technique we are able to see the original and Edge detected image which are given as:

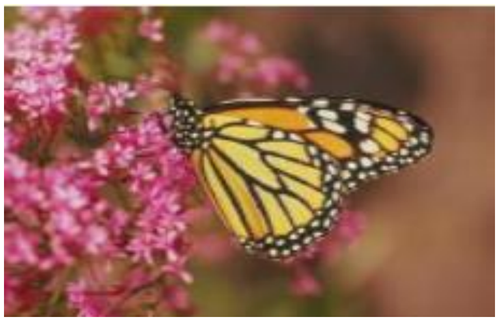

Fig.3: Original Image

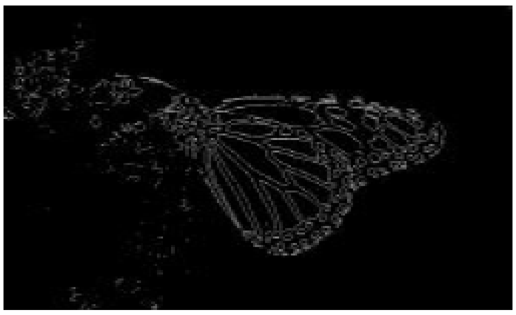

Fig.4: Edge detected Image

4. Block diagram:

For Sobel Edge detector, there are three main modules: Sobel buffer memory, gradient computation module,and edge map module. Sobel edge detector is a window based operator which requires pixel neighborhood information for computing the edge map of a particular pixel. Therefore, 8-bit gray pixeldata coming from the camera interface module cannot be processed directly. It must be stored in FPGA memory before processing. The gradient computation module uses eight neighborhood pixels coming from buffer memory for computing the approximate gradient value which is the sum of absolute values of horizontal and vertical gradients. Edge map module is a simple comparator which compares the gradient value (GRD) with user defined threshold (TH). The block diagram of above modules is given as.

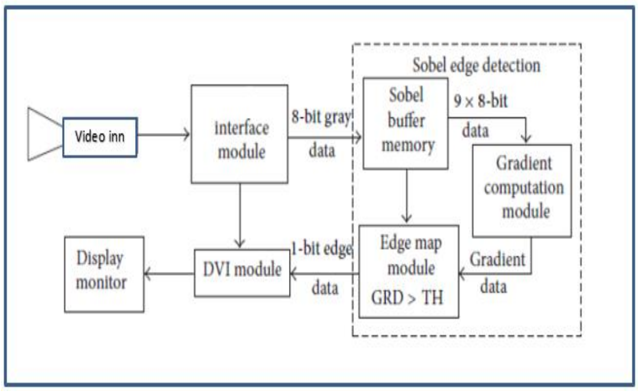

Fig.5: Block diagram of different modules

5. ZYBO-SoC Trainer Board

The ZYBO or Zynq board is a feature-rich, ready to use, entry level embedded software and digital circuit development platform built around the smallest member of the Xilinx Zynq 7000 family, the Z-7010. The Z-7010 is based on the Xilinx all programmable system on chip (AP Soc) architecture, which tightlyintegrates a dual- core ARM Cortex-A9 processor with Xilinx 7 series field programmable gate array (FPGA) logic.

When coupled with the rich set of multimedia and connectivity peripherals available on the ZYBO, the Zynq Z-7010 can host a whole system design. The on-board memories, video and audio I/O, dual- role USB, Ethernet and SD slot will have your design up and ready with no additional hardware needed. Additionally, six Pmod connectors are available to put any design on an easy growth path. The ZYBO Z-7010 trainer board is given as:

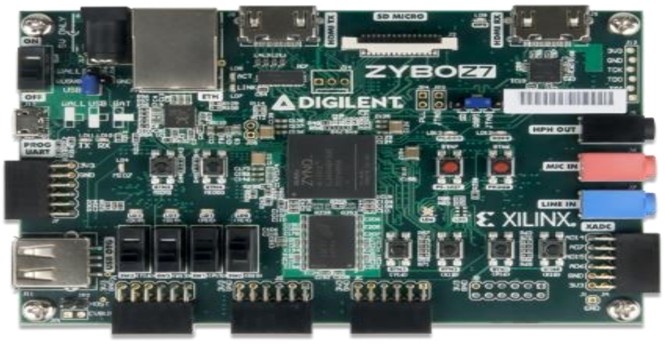

Fig.6: Development Board of FPGA

6. Sobel IP block:

The Sobel block is designed with four inputs lines and one output line. Out of these four input lines one is, 8-bit data line is used to provide image pixel data as input to the Sobel operator via line buffer. The other three are the control lines i.e. clk (clock input), rst(reset), rdy(ready). The clk is used for the clock for the clock input for the Sobel filter, rst is used to reset the module and the rdy indicates that all the required data for the Sobel operation are ready and the Sobel operation can now be performed on that data. The ready signal used and the Sobel operation can now be performed on that data. The ready signal used here is active high i.e. when the signal is high, it indicates that the image pixel data is ready. The IP block of Sobel is given below.

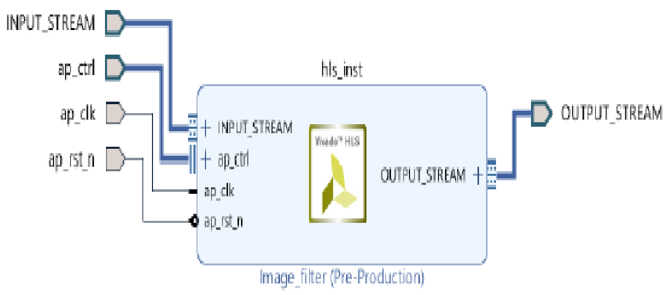

Fig.7: IP of Sobel Operator

7. High Level Synthesis(HLS):

High-Level Synthesis is a robotized configuration process in which the desired algorithm behaviour is interpreted and the digital hardware is created which implements that behaviour. The implementation and synthesization of Sobel Edge detection of FPGA through Xilinx in Register Transfer Level (RTL) and High Level Synthesis (HLS) tool transform C specification into a Register Transfer Level implementation. Actually, High level synthesis creates an optimized implementation based on default behaviour, constraints and any optimization directives you specify. You can use optimization directives to modify and control the default behaviour of the internal logic and I/O ports. This allows you to generate variations of the hardware implementation form the same C code. The Xilinx Vivado HLS tool synthesizes a C function into an IP block that you can integrate into a hardware system.. The main advantage is that use of HLS increases the level of abstraction, while developing an FPGA application which saves time. Apart from this, HLS also gives better control over optimizing the design architecture by efficiently building and verifying the hardware.

Hardware acceleration is the technique of utilizing computer hardware to perform certain functions more efficiently than software-based implementation. The hardware can be defined either at Register Transfer Level (RTL) or at algorithmic level. In the RTL level the hardware is defined using hardware description languages like Verilog or VHDL. Detailed microarchitecture if the design needs to be implemented at this level. RTL can be directly input to a logic synthesis tool followed by further steps in FPGA implementation at this level. RTL can be directly input to a logic synthesis tool followed by further steps in FPGAimplementation. In case of High Level synthesis, we only define the algorithm the system has to implement and the interconnect protocol. The HLS tool handles the rest of the micro architecture design by transforming any utilized or partially timed code into a fully synthesizable RTL code which gives a detailed cycle by cycle implementation on the hardware.

In Xilinx FPGA design suite, HLS is performed using Xilinx Vivado HLS tool and its libraries such as:

1. HLS_OpenCV which enables us to leverage the OpenCV framework
2. HLS_Video library allows us to utilize the necessary image processing functions.

3. Phases of HLS:

The high-level synthesis used to describe the design of the IP is untimed unlike the Verilog/VHDL designs. Thus, when the HLS tool translates high level code into Verilog/VHDL code, it goes through a number of stages before creating the RTL. Scheduling, this stage determines and Schedules execution of instructions that are to be implemented. Binding- this stage is for assigning the resources available on the device once the scheduling phase is completed. Control logic extraction- the control logic extraction phase is used to extract the control signals and create control like creating state machines to exert control over the behaviour of the model.

4. Implementation Window:

The processing result of Sobel extraction is verified by a real-time video system. The original image or video is captured by different methods, USB camera or another monitor, after processing the image or video shown on the display device through the HDMI interface. The entire project is implemented on ZYBO board, which is one of the Zynq family FPGA. The Z-7010 is based on the Xilinx All Programmable system on chip (APSoC) architecture, which tightly integrates a dual-core ARM Cortex A9 processor with Xilinx 7 series field programmable gate array (FPGA) logic. The implementation windwo for this overall are given below.

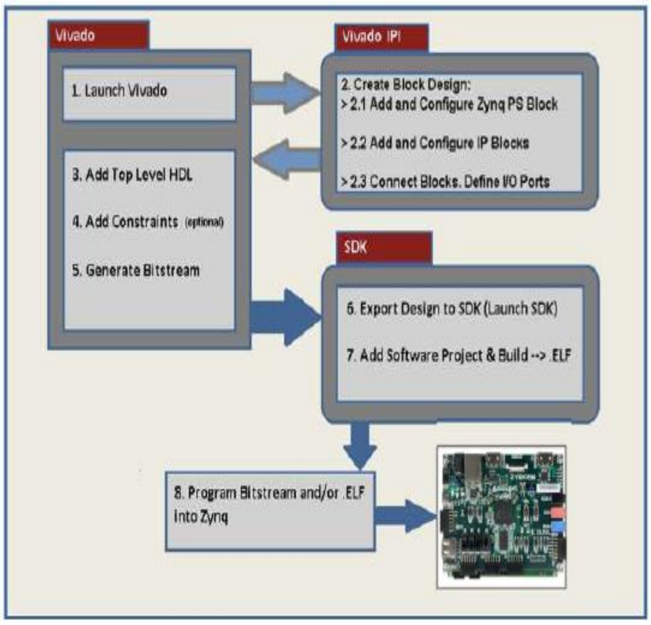
Vivado
1. Launch Vivado
3. Add Top Level HDL
4. Add Constraints (optional)
5. Generate Bitstream
Vivado IPI
2. Create Block Design:
>2.1 Add and Configure Zynq PS Block
>2.2 Add and Configure IP Blocks
>2.3 Connect Blocks, Define I/O Ports
SDK
6. Export Design to SDK (Launch SDK)
7. Add Software Project & Build --> .ELF
8. Program Bitstream and/or .ELF into Zynq

Fig.7: Implementation window

5. Results:

In this project "Edge detection on FPGA" using Sobel Edge detector is reviewed and focus has been made in detecting the edges of the digital images. For implementation of the whole project there are several stages of result that should be incorporated here.

In the first stage of implementation, different colour patterns as well as colour blends is generated with different resolution as well as different frame buffers and output can be seen by another monitor through HDMI with very low delay of time that is done successfully.In the second stage of implementation, start or stop video streaming as video in through HDMI processed and video out through HDMI that can be seen in another monitor and this is done successfully.In the last stage of implementation, Sobel edge detection is implemented in the second stage of implementation and edge detected video out can be seen on another monitor through HDMI and this is the final result which is done successfully.

6. Conclusion:

Edge detection on Field Programmable Gate Array is successfully design and according to the requirements and specifications but design of IP block is very lenghty and time consuming, also the software used in this project takes very long time for installation and implementation. As we all know that nowadays, this modern era, these types of technique are used in many place and will be more effective in upcoming decades.

References:

1. https://en.wikipedia.org/wiki/Edge_detection
2. D Sundraranjan"Digital Image Processing: A Signal and Algorithm Approach"Springer; 1st ed. edition 2017
3. S. Singh, A.K. Saini and R saini"Real time FPGA based implementation of colour image: Edge Detection international journal of image, graphics and Signal processing"
4. Sheetal D. Bhoyar , TarnnumPathan, Amit D. Landge"Design and Implementation of Sobel Edge Detection technique using VHDL"
5. Poonam Pawar "FPGA Implementation of Image Detection Algorithm"
6. Xilinx AXI reference guide VG761 March 7, 2011
7. Vivado Design Suite user guide HLS UG902 May 30, 2014
8. DigilentZybo FPGA Board reference manual
9. The Zynq Book 1st Edition
10. Mohamed A. El-Sayed"Edge Detection of Images: Algorithms of Edges Detection for Digital Image"
11. NeolSolanki and Neel Tailor "Sobel Edge detection on Zynq based architecture with Vivado"
12. Naeem Akbar Channar"A Comparative Study of Edge Detection Techniuqe in Digital Images"