ISSN: 2277-9655
Scientific Journal Impact Factor: 3.449
(ISRA), Impact Factor: 2.114

IJESRT

INTERNATIONAL JOURNAL OF ENGINEERING SCIENCES & RESEARCH TECHNOLOGY

Design and Simulation of Canny Edge Detection

D Narayana Reddy*, Vaijanath V.Yerigeri, Harish Sanu

* Assistant professor, Dept. of ITE, M.B.E.S College of Engineering, Ambajogai, Maharashtra, India
HOD, Dept. of ITE, M.B.E.S College of Engineering, Ambajogai, Maharashtra, India

Abstract

Edge is one of the prominent features in the image processing applications. The edge detection algorithm is carried out by using different methods. Canny edge algorithm is one of the well known edge detection algorithm. In proposed design, the canny edge detection algorithm is designed in verilog and simulated using the MATLAB and Modelsim. The input image is converted to text/pixel values using MATLAB and is stored in a new text file. Verilog with test bench program is used to assign inputs, outputs and memory locations in the form of text file. In Verilog the neighbouring pixels are compared and the edge of the image is detected, edged text is stored in new text file. The MATLAB is used to convert the edged text/pixel values to edged image.

Keywords: Canny edge detection.

Introduction

Edges characterize boundaries and are therefore a problem of fundamental importance in image processing. Edges in images are areas with strong intensity contrasts, a jump in intensity from one pixel to the next. Edge detection of an image significantly reduces the amount of data and filters out useless information, while preserving the important structural properties in an image.

Strong intensity contrast in an image shows the edge of an image, and the edges will identify the boundary of an image. Edge detection is a very important first step in many algorithms used for segmentation, tracking and image/video coding. The Canny edge detection is predominantly used due to its ability to extract significant edges. Edge detection, as a basic operation in image processing, has been researched extensively. A lot of edge detection algorithms, such as Robert detection, Prewitt detection, Kirsch detection, Gauss-Laplace detection and Canny detection have been proposed. Among these algorithms, Canny algorithm has been used widely in the field of image processing because of its good performance. The Canny edge detection is predominantly used in many real-world applications due to its ability to extract significant edges with good detection and good localization performance.

The proposed Canny edge detection algorithm is designed using Verilog HDL and simulated using MATLAB and Modelsim. The input to the Verilog is in the form of text/pixel values. The edges of an image are determined by comparing the neighboring text pixels. The edge detected output is in the form of text/pixel values. MATLAB is used for the conversion of the input image to text/pixel values and output edge detected text/pixel output to the edged output. The proposed design can be implemented for any format of images like jpg, gif, bmp etc.

Literature survey

R. Ponneela Vignesh et.al in [1] implemented the Canny edge detection algorithm in FPGA device, and it is applicable for image segmentation, image tracking, image coding etc. In this paper Canny edge algorithm reduces memory requirements, decreased latency, increased through output with no loss in edge detection performance and Canny edge detection algorithm use probability for finding error rate localization and response in various images. Chandrashekar N.S et.al in [2] explains the distributed Canny edge detection algorithm that results in significantly reduced memory requirements, decreased latency and increased throughput with no loss in edge detection performance as compared to the original Canny algorithm. Tejaswini H.R et.al in [3] explains that the edge detection is an eloquent step in image processing and in object recognition. The Canny edge detection is called as optimal detection due to its

ISSN: 2277-9655
Scientific Journal Impact Factor: 3.449
(ISRA), Impact Factor: 2.114

good performance. Samina Jafar et.al in [4] explains the edge detection is one of the key stages in image processing and objects reorganization. The Canny Edge Detection is one of the most widely used edge detection algorithm due to its good performance. T. Rupalatha et.al in [5] explains edge detection is one of the basic operation carried out in image processing and object identification. In this paper, the distributed Canny edge detection algorithm that results in significantly reduced memory requirements, decreased latency and increased throughput with no loss in edge detection performance as compared to the original Canny algorithm.

Proposed methodology

The proposed Canny Edge Detection flow

The Canny edge detection consists of following steps:

- The input image is converted into text/pixel values using MATLAB program.
- Those values are stored in files and these values are passed to Canny edge detection design through test bench.
- The pixel values are stored in external text file, and are captured by Verilog HDL with help of commands like “sreadmemh” or “sreadmemb”.
- Edges are finding in the image using Verilog HDL with Modelsim software and final image value is stored in another file.

By using the MATLAB program the edge detected images values are converted into image.

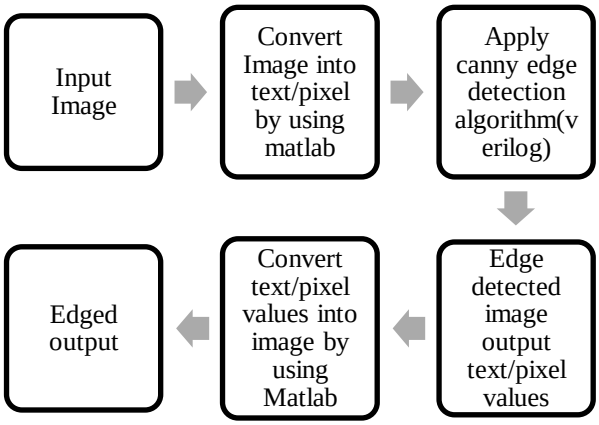


Fig1: Over all block diagram of canny edge detection.

Flow chart

Apply the input, is in form of RGB image is converted into gray scale image. rgb2gray(RGB) converts the true color image RGB to the grayscale intensity image I. rgb2gray converts RGB images to grayscale by eliminating the hue and saturation information while retaining the luminance. Then it is resized in the range of 256x256 and creating cell, it consists 9 values. The middle value of a cell makes it as a text/pixel value. These text/pixel vales are stored into one external text file. After getting image in the form of text/pixel values, link the MATLAB and Modelsim. MATLAB is powerful software package for mathematical simulation, especially when signals are represented in an abstract i.e. mathematical form. To test a hardware system in terms of the abstract inputs and outputs, linking Modelsim VHDL or Verilog simulator with MATLAB is very helpful. The most beneficial use could be to write a test bench for the HDL model in MATLAB. The test cases can be

ISSN: 2277-9655
Scientific Journal Impact Factor: 3.449
(ISRA), Impact Factor: 2.114

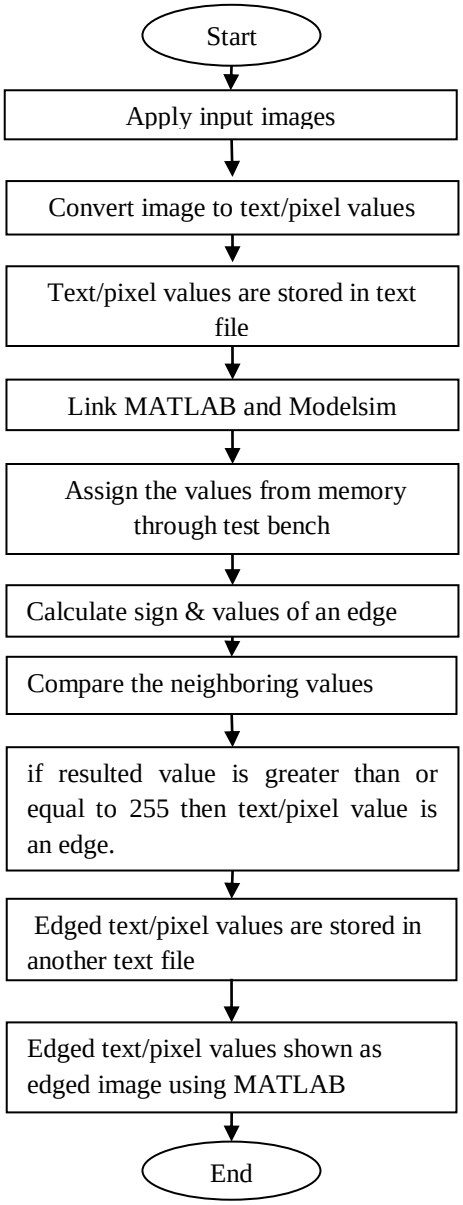


Fig2: Overall flow chart of a proposed methodology.

generated in MATLAB at a higher level of abstraction, making it suitable for regression testing with random inputs. Given the ease of interfacing, this provides a very attractive alternative to Verilog PLI or VHDL FLI programming. For the link to work, Modelsim has to be invoked from the command prompt of MATLAB. For this purpose, MATLAB needs to know the location of MODELSIM. Assign the values from memory through test bench and values are temporally damped into registers, by performing operations and compute the output values and compare the neighboring values, if resulted value is greater than or equal to 255 then text/pixel values is an edge and the edged text/pixel values are stored in another text file. The text file contains only two values 255 it represents the edge of an image and 0 it represents the non edge of an image. The edged text/pixel values are as shown edged image using MATLAB.

Conversion of input image to text/pixel values

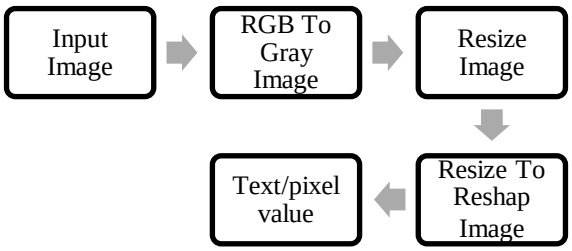


Fig3: Input Image converted to text/pixel by using MATLAB.

First consider input image, it is given as input to MATLAB; image is converted in to RGB to GRAY scale image then the image is resized. The identical pixel values are removed from the resized image, which means that the useless information is removed from the resized image which is called as Reshaped image; the reshapes image is then digitalizes in which the input image is converted to the text/pixel values. The process flow of conversion of input image to the Text/pixel values in MATLAB is as shown in fig3.

ISSN: 2277-9655
Scientific Journal Impact Factor: 3.449
(ISRA), Impact Factor: 2.114

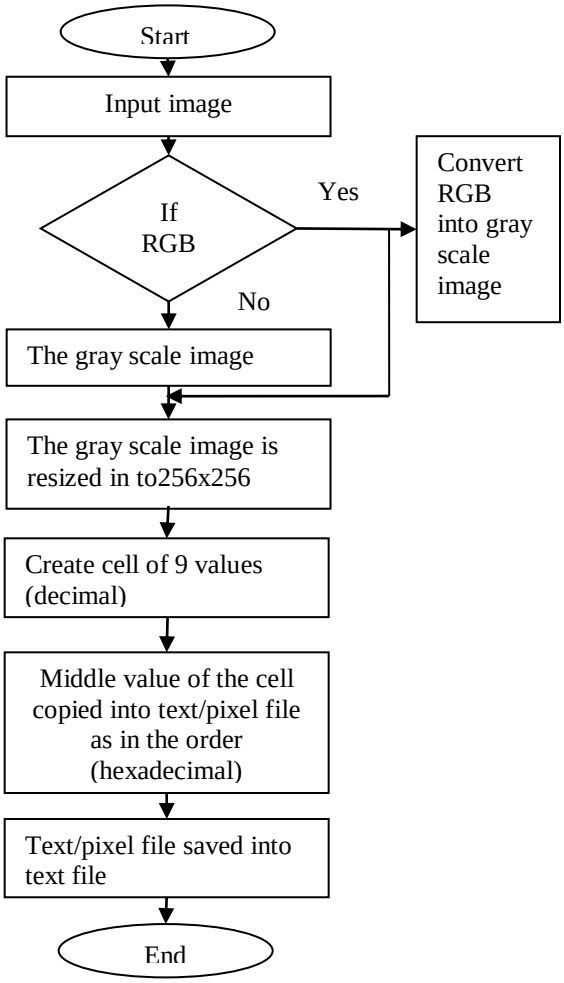


Fig4: Flow chart of an image is converted into text/pixel values.

Apply the input, is in form of RGB image is converted into gray scale image. rgb2gray(RGB) converts the true color image RGB to the grayscale intensity image I. rgb2gray converts RGB images to grayscale by eliminating the hue and saturation information while retaining the luminance. Then it is resized in the range of 256x256 and creating cell, it consists 9 values. The middle value of a cell makes it as a text/pixel value. These text/pixel vales are stored into one external text file. After getting image in the form of text/pixel values, link the MATLAB and Modelsim. MATLAB is powerful software package for mathematical simulation, especially when signals are represented in an abstract i.e. mathematical form. To test a Hardware system in terms of the abstract inputs and outputs, linking Modelsim VHDL or Verilog simulator with MATLAB is very helpful. The most beneficial use could be to write a test bench for the HDL model in MATLAB. The test cases can be generated in MATLAB at a higher level of abstraction, making it suitable for regression testing with random inputs. Given the ease of interfacing, this provides a very attractive alternative to Verilog PLI or VHDL FLI programming. For the link to work, Modelsim has to be invoked from the command prompt of MATLAB. For this purpose, MATLAB needs to know the location of MODELSIM. Assign the values from memory through test bench and values are temporally damped into registers, by performing operations and compute the output values and compare the neighboring values, if resulted value is greater than or equal to 255 then text/pixel values is an edge and the edged text/pixel values are stored in another text file. The text file contains only two values 255 it represents the edge of an image and 0 it represents the non edge of an image. The edged text/pixel values are as shown edged image using MATLAB.

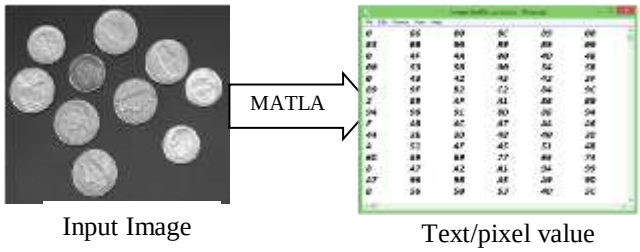


Fig5: Coin image converted into text/pixel value.

Verilog Implementation

Link the MATLAB to Modelsim, MATLAB is powerful software package for mathematical simulation, especially when signals are represented in an abstract i.e. mathematical form.

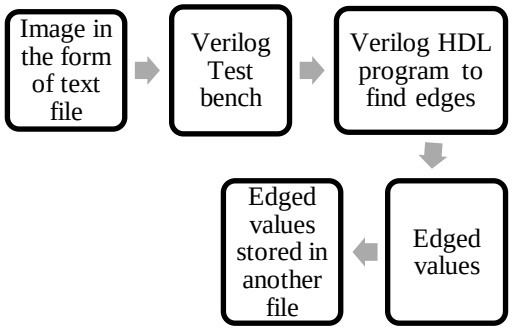


Fig6: Canny edge detection flow in Verilog.

ISSN: 2277-9655
Scientific Journal Impact Factor: 3.449
(ISRA), Impact Factor: 2.114

Flow chart

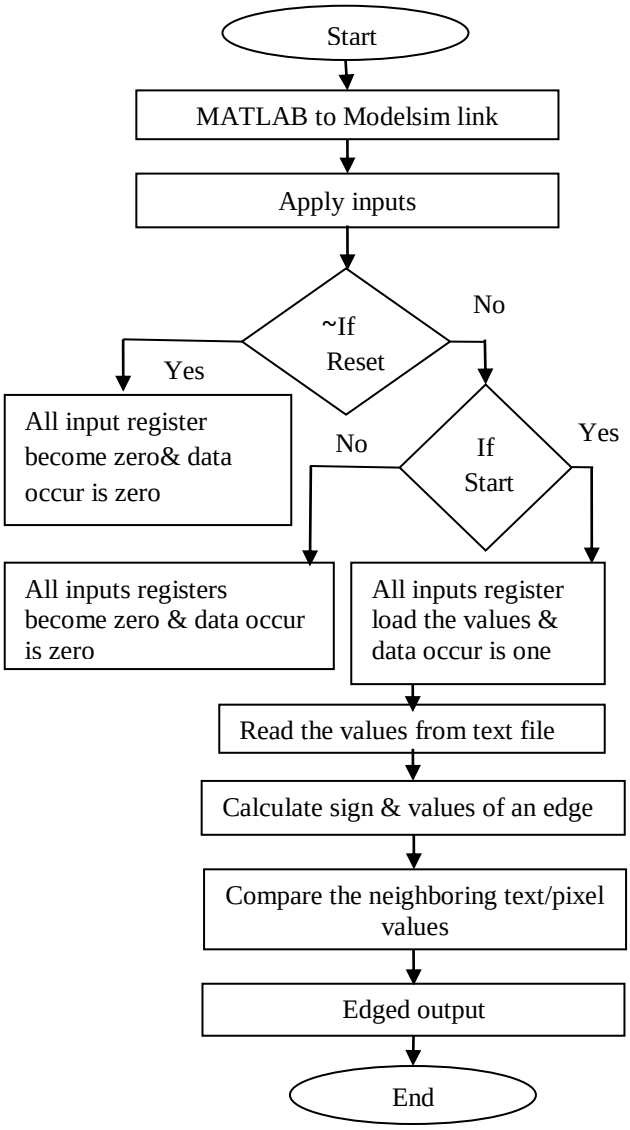


Fig6: Flow chart for verilog implementation.

To test a hardware system in terms of the abstract inputs and outputs, linking Modelsim VHDL or Verilog simulator with MATLAB is very helpful. The most beneficial use could be to write a test bench for the HDL model in MATLAB. The test cases can be generated in MATLAB at a higher level of abstraction, making it suitable for regression testing with random inputs. Given the ease of interfacing, this provides a very attractive alternative to Verilog PLI or VHDL FLI programming. For the link to work, Modelsim has to be invoked from the command prompt of MATLAB. For this purpose, MATLAB needs to know the location of MODELSIM. MATLAB and Modelsim communicate with each other either through shared memory server or through TCP/IP links using socket calls. If MATLAB and Modelsim are running on different machines then TCP/IP link is mandatory. For same machine, shared memory is preferable.

Apply the inputs, to create registers and wires for each and outputs & calculate the sign & values of an edge. The registers are used to store inputs for temporally from a memory and wires are used to interconnect to both inputs and outputs. If inverse of reset then the all input register become zero& data occur is zero. if inverse of reset is no then check start bit is one then the all inputs register load the values & data occur is one , if start bit is zero then all inputs registers become zero & data occur is zero .if start bit is one then all inputs are register are loaded from the number of inputs. The computing the sign values as 0 and 1 and edged values & if damped text/pixel file. Text/pixel file stored in the memory & all data access from the memory, but here register are used because the registers already present in the processor & need not go anywhere else (memory) to fetch the data. All outputs and registers values are updated & process continuous.

As shown in fig.4.7. The Verilog code and test bench code are interconnected to each other. Process start the inputs text/pixel values are loaded through test bench, assign the memory and read the values from memory, these values are damped in to register. Then open one new file after to store the edge detected text/pixel values by some integers.

If the data occur is one then find the gradients in both directions. That by using registers operation to add, by comparing neighboring text/pixel values and shifting the text/pixel values to find an edge detected text/pixel value. The resulted value is in the form of hexadecimal it is converted into decimal. The resulted value is greater than or equal to 255 it is consider as edge and the edge is denoted as 255. The resulted value is less than 255 or equal to 0, calls it as non edge and is denoted as 0. These values are displayed in the Modelsim window, this window consists of only two values. The Modelsim window consists of resulted values are in the form of decimal values. The pixel value is 255 consider as edge and 0 is consider as a non edge.
Example:

ISSN: 2277-9655
Scientific Journal Impact Factor: 3.449
(ISRA), Impact Factor: 2.114

Conversion of edged text/pixel values to edged output image

As shown in below diagram it explains how to convert the text/pixel values in to image by using MATLAB. The conversion of edged text/pixel values to edged output image as shown in fig.4.10.

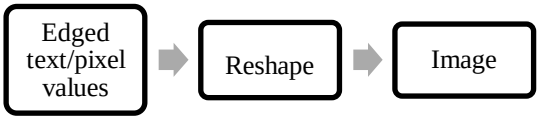
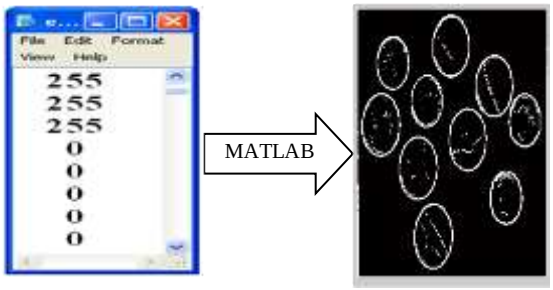


Fig.4.10: Text/pixel value converted into image.

The edged text/pixel vales are present in Modelsim window those edged text/pixel values are stored in another new text file call it as edged text file. This text file is given to an input to an MATLAB. First read the text file by using suitable MATLAB command and then text/pixel values are reshaped in the order of 256x256. Then reshaped text/pixel values are converted in to image by using MATLAB. The resulted image is edged output image. As shown in below example. The MATLAB is not directly fetching the input from the Modelsim window; it fetching input from the text file.

Consider one example of an edge detected text/pixel values in the form on single column that will be converted into an image. These edged values are taken from the coin input image. After Modelsim process will get these values. The text/pixel values consist only two values, those are 255 it represent edge of an image and another value is 0 it represents the non edge of an image. The edged output image is shown in fig.4.11. The edged output image shows the white line, this white line indicates that the edge of an image. The left side of fig shows the edge detected text/pixel values present in the text file and right side shows the edged output image. The text file contains number edged text/pixel values and non edge text/pixel values.

Example:



Text / pixel value

Edged Image

Fig.4.11: Edged text/pixel value is converted into edged coin image.

Result

The canny edge algorithms are designed in Verilog. And the simulated wave form using Modelsim simulator is shown in Fig 6. The matlab is interred connected with the Verilog coding; hence the image is converted to edged image as shown in Fig 7.

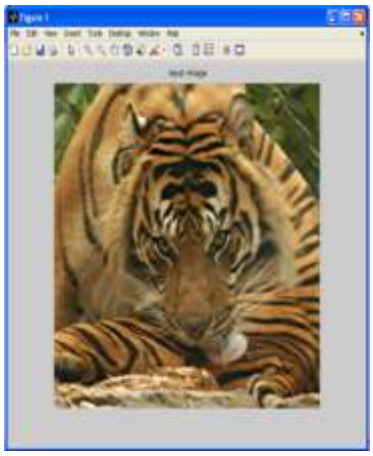


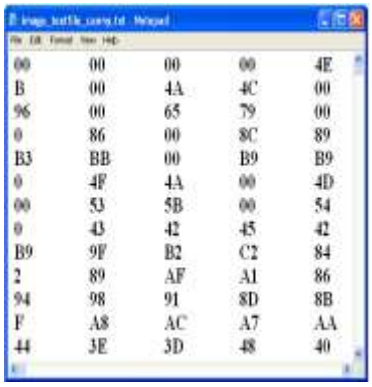
Fig.5.1: Input tiger.jpg image.

Fig.5.2: Gray scale image.

ISSN: 2277-9655
Scientific Journal Impact Factor: 3.449
(ISRA), Impact Factor: 2.114

Fig.5.3: Resized image.

Fig.5.4: Reshaped image.



00	00	00	00	4E
B	00	4A	4C	00
96	00	65	79	00
0	86	00	8C	89
B3	BB	00	B9	B9
0	4F	4A	00	4D
00	53	5B	00	54
0	43	42	45	42
B9	9F	B2	C2	84
2	89	AF	A1	86
94	98	91	8D	8B
F	A8	AC	A7	AA
44	3E	3D	48	40

Fig.5.5: Input text/pixel values.

Fig.5.6: Modelsim shows the edge and non edge.

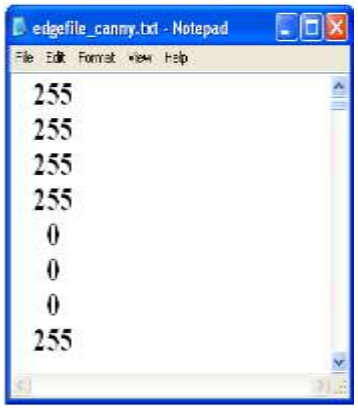


Fig.5.7: Edged text/pixel values.

Fig.5.8: Edged output image.

Reference

1. R. Ponneela Vignesh, R. Rajendran, "Performance and Analysis of Edge Detection Using FPGA Implementation". International Journal of Modern Engineering Research (IJMER) Vol.2, Issue.2, Mar-Apr 2012 pp-552-554 ISSN: 2249-6645.
2. Chandrashekar N.S, Dr. K.R. Nataraj, "A Distributed Canny Edge Detection and Its Implementation on FPGA" International

ISSN: 2277-9655
Scientific Journal Impact Factor: 3.449
(ISRA), Impact Factor: 2.114

Journal of Computational Engineering Research (ijceronline.com) Vol. 2 Issue.7. Issn 2250-3005(online) November 2012.

3. Tejaswini H.R, Vidhya N, Swathi R Varma, Santhosh B, "An Implementation of Real Time Optimal Edge Detection and VLSI Architecture". International conference on electronics and communication engineering, 28th april-2013, bengaluru, isbn: 978-93-83060-04-7.
4. Samina Jafar, Anupsingh Ramprakashsingh Rajput, "Improved Distributed Canny Edge Detection In VHDL". VSRD International Journal of Electrical, Electronics & Communication Engineering, Vol. 3 No. 6 June 2013.
5. T. Rupalatha**,** Mr. C. Leelamohan, Mrs. M. Sreelakshmi, "Implementation of Distributed Canny Edge Detection On FPGA". International Journal of Innovative Research in Science, Engineering and Technology, Vol. 2, Issue7, July 2013.
6. Divya. D, Sushma P. S, "FPGA Implementation of a Distributed Canny Edge Detection". International Journal of Advanced Computational Engineering and Networking, ISSN: 2320-2106 Volume- 1, Issue- 5.
7. Mallavarapu. Ramu, T. V. S. Adinarayana, "A Hardware/Software Co-Design Architecture Implementation of Canny Edge Detection Using FPGA and MATLAB". International Journal of software& hardware research in engineering.