CANNY EDGE DETECTION: A COMPREHENSIVE REVIEW

Himangi Agrawal, Krish Desai

E-Mail Id: 21bec046@nirmauni.ac.in, 21bec026@nirmauni.ac.in

Department of Electronics and Communication Engineering Institute of Technology, Nirma University, Ahmedabad, Gujarat, India

Abstract- Canny edge detection is a widely employed technique in image processing known for its effectiveness in identifying and highlighting edges within digital images. Because of its excellent performance, the Canny Edge Detector is one of the most used edge detection algorithms. For several image processing techniques, including image enhancement, image segmentation, tracking, and image/video coding, edge detection serves as a preliminary step. Compared to the Sobel algorithm, Canny's edge detection approach yields much lower memory requirements, reduced latency, and enhanced throughput without sacrificing edge detection performance. This paper provides a comprehensive review of canny edge algorithm, elucidating each step. In this paper, the canny edge algorithm is implemented on an image as well as in real time using MATLAB and its Simulink model. We have also performed high level synthesis of the proposed algorithm using HDL coder.

Keywords: Canny edge, Gradient, Non-maximum suppression, Sobel kernels, Thresholding.

1. INTRODUCTION

In the realm of image processing and computer vision, the accurate identification of edges within images is a fundamental task with widespread applications. As edges define boundaries, they represent a fundamentally important problem in image processing. Strong intensity contrasts, or a change in intensity from one pixel to the next, are what defines edges in images. By removing unnecessary information and drastically lowering the amount of data, edge detection preserves [1] an image's crucial structural elements. The development of the canny edge detection algorithm was motivated by a number of significant issues with the previous edge detection techniques, including their vulnerability to noise and the requirement for precise edge localization. The goal of Canny was to develop an algorithm that could precisely localize edges even in the presence of noise, minimize false positives, and reliably identify edges.

Here is a list of standards to enhance the edge detection techniques used today. First and foremost, there is a low error rate. It is crucial that edges in pictures are not overlooked and that non-edges elicit no reaction at all. A well-localized set of edge points is the second requirement. Stated differently, there should be as little space as possible between the edge pixels that the detector finds and the real edge. Having a single reaction to a single edge constitutes the third requirement. This was put into place because the previous two were insufficiently significant to totally rule out the potential for more than one response to an edge. Lastly, compared to other edge detection techniques [3-4], the suggested methodology yields results with less time consumption. So canny edge detection has be standard for years with best performance as compared to other edge detection algorithms.

2. CANNY EDGE DETECTION

The Canny edge detection algorithm is a multi-stage process designed to identify edges in an image while minimizing false positives and accurately localizing the detected edges. The first stage involves Gaussian smoothing. In order to lower noise, the image must first be smoothed using a Gaussian filter. By removing unnecessary information from the image, this step helps to highlight the key details. The input image is convolved with Gaussian kernel. The Gaussian kernel is defined as

$$G(x, y)= 1/2\pi\sigma 2e-(x^2 +y^2)/(2\sigma^2) \quad (1)$$

where σ is the standard deviation and x and y are spatial coordinates. The next step is the gradient calculation. Convolution with Sobel kernels is used to compute the image's gradient. The gradient shows how quickly intensity changes in both the x and y directions. This process [2] aids in locating the areas of the picture that exhibit the biggest variations in intensity, which frequently line up with edges. On an image, the operator measures the spatial gradient in two dimensions. In this step, a 3x3 operator is used to convolve the blurry image that was produced during the image smoothing stage. The operator creates a gradient image; it is a discrete differential operator. The operators applied to the gradients, both vertical and horizontal. A first stage smoothed image is convolved with the horizontal and vertical operators to produce the gradient image. The horizontal and vertical gradients' magnitudes are represented by the pixel values of the images Gx and Gy, respectively.

Next, the gradient's magnitude, or edge strength, is roughly calculated using the following formula [15, 17]: The gradient in the x and y directions are used to calculate the edge's direction. The direction of the tangent to the contour that the edge defines in two dimensions is known as the edge direction. The arctangent is used to determine the edge direction of each pixel in an edge direction image. Upon determining the edge directions, non-maximum suppression is implemented. Pixels are traced along the gradient in the edge direction using non-maximum

suppression, and values perpendicular to the gradient are compared. The value in the edge direction is compared with two values of perpendicular pixels. They are suppressed, or have their pixel value changed to 0, if their value is less than the pixel on the edge. If not, the higher pixel value is set as the edge, and the remaining two are suppressed with a pixel value of 0.

The final step in canny edge detection [16] is thresholding with hysteresis, which is used to remove non-edge pixels and spurious points from the outcomes of non-maximum suppression. To achieve thin edges in the input image for thresholding with hysteresis, it underwent image smoothing, edge strength and pixel calculation, and the non-maximum suppression stage. This stage's results, which are obtained by applying the two threshold values, should only provide us with the legitimate edges in the provided image. To overcome the issue of breaks in the edges, the algorithm employs edge tracking by hysteresis. This involves setting two thresholds: a high threshold T1(high) and a low threshold T2(low). Pixels with gradient magnitudes above T1 (high) are considered strong edges, while those between T2 (low) and T1 (high) are considered weak edges. The algorithm [8,9]]starts with the strong edges and traces along the gradient direction. If a weak edge pixel is connected to a strong edge pixel, it is also classified as a strong edge. This helps to connect and form continuous contours along the edges.

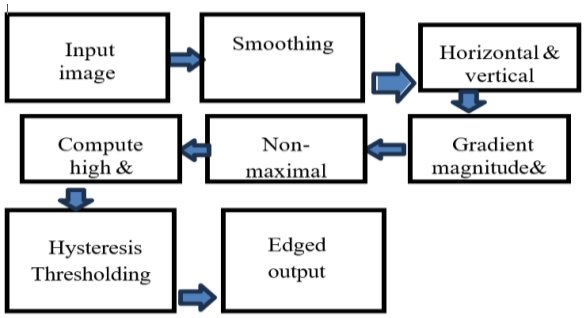

Fig. 2.1 Block Diagram of Canny Edge Detection

Gx =

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

Gy =

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

Fig. 2.2 Sobel Operators

Step 1: A set of procedures needs to be followed in order to put the canny edge detector algorithm into practice. Prior to attempting to find and detect any edges, any noise in the original image must be filtered out. Furthermore, the Gaussian filter is only utilized in the Canny algorithm [13] since it can be computed with a straightforward mask. Standard convolution techniques can be used to carry out the Gaussian smoothing after an appropriate mask has been computed. Typically, a convolution mask is substantially smaller than the original image. Consequently, the mask is moved across the picture, adjusting a square of pixels at a time. The detector's sensitivity to noise decreases as the Gaussian mask width increases. As the Gaussian width is increased, there is a slight increase in the localization error in the detected edges as well.

The first step in the uncanny edge detection process is image smoothing. To produce an intermediate image, the pixel values of the input image are convolved using predefined operators. This method is applied to an image to lessen noise in it or to make it appear less pixilated. Using a Gaussian filter to convolve the input image is how image smoothing is done with the help of equation (1) characterizing how narrow the peaked function is.

Step 2: Finding the edge strength requires taking the image's gradient after the image has been smoothed and the noise has been removed. An image's 2-D spatial gradient[16]is measured by the Sobel operator. Next, it is possible to determine the approximate absolute gradient magnitude (edge strength) at every point.

In order to estimate the gradient in the x-direction (columns) and the gradient in the y-direction (rows), the Sobel operator employs a pair of 3x3 convolution masks. In this step, a 3x3 Sobel operator is convolved with the blurry image that was obtained during the image smoothing stage. A gradient image [11] is produced by the discrete differential operator known as the Sobel operator. The gradients in the horizontal and vertical directions were computed using Sobel operators. In Figure 2, the sobel operators are displayed. To get a gradient image, apply the Sobel operators' fig (2). Equations (2) and (3) demonstrate how to convolve a smoothed image from the first stage with the horizontal and vertical Sobel operators, respectively, to obtain the gradient image.

Gx= (I * gx) (2)

Gy= (I * gy) (3)

Equation (4) uses the images Gx and Gy from equations (2) and (3) to determine the "edge strength" of a pixel in

an image. The strength of the edge G is

$$|G| = \sqrt{Gx\ 2 + Gy\ 2} \quad (4)$$

Step 3: The gradient in the x and y directions are used to calculate the edge's direction. However, if the sum equals zero, an error will be produced. Therefore, a restriction needs to be set in the code whenever this occurs. The edge direction must equal 90 degrees or 0 degrees, depending on the value of the gradient in the y-direction, whenever the gradient in the x-direction equals zero. The edge direction will equal zero degrees if Gy is zero. In the absence of that, the edge direction will be 90 degrees. "Edge direction is defined as the direction of the tangent to the contour that the edge defines in two dimensions" is the simple formula for determining edge direction. The arctangent is used to determine each pixel's edge direction in an edge direction image.

$$\theta = \arctan(Gy/Gx) \quad (5)$$

Step 4: In the non-maximum suppression stage, each pixel's edge strength in an image derived from equation (4) is utilized. Before being used in non-maximum suppression, the edge directions from equation (5) are rounded off to one of four angles: 0 degrees, 45 degrees, 90 degrees, or 135 degrees.

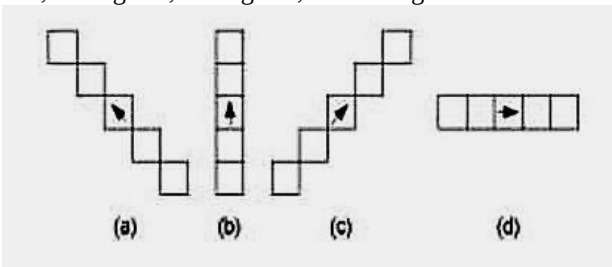

Fig. 2.3 Linear Window of Angle (a) 135° (b) 90° (c) 45° (d) 0°

Step 5: It is now necessary to apply non-maximum suppression once the edge directions are known. When using non-maximum suppression, any pixel value that isn't thought of as an edge is suppressed (set to 0). This allows the edge to be traced along in the edge direction. The resultant image will have a thin line. Most edge detection algorithms use non-maximum suppression (NMS). In a given local neighborhood[7], all pixels with edge strengths less than or equal to the maximal value are designated as zero through this process. This local neighborhood may consist of a linear window with a length of five pixels in various directions. The edge direction of the pixel under consideration for a block in an image determines the linear window that is taken into consideration.

Step 6: The final step in clever edge detection is thresholding with hysteresis, which is used to remove non-edge pixels and suspicious points from the outcomes of non-maximum suppression. To achieve thin edges in the input image for thresholding with hysteresis, it underwent image smoothing, edge strength and pixel calculation, and non-maximum suppression stage. The output of this step should provide us with only the legitimate edges in the provided image. This is achieved by utilizing the two threshold values for the edge strength of each image pixel, T1 (high) and T2 (low). An edge is deemed to be definitive if its strength exceeds T1. Zero is set for edge strength if it is less than T2. If there is a path connecting a pixel with edge strength below T1 to a pixel with edge strength above T1, then the pixel with edge strength between T1 and T2 thresholds is taken into consideration. All of the path's pixels must have edge strengths of at least T2. This procedure lowers the likelihood of streaking. Thresholds T1 and T2 are also reliant on the intensity of the image's pixels, just as edge strength is dependent on pixel intensity. Hence, the clever edge detectors use adaptive algorithms to calculate the thresholds T1 and T2. Consequently, the clever edge detection method detects every edge in a picture [15].

3. PROPOSED METHODOLOGY IN SIMULINK

The methodology is for detecting edges in a video stream using a Simulink model. The vision toolbox is used for the model and it is hardware compatible. The behavioral of the pixel stream edge detector block is verified by full-frame block from computer vision toolbox.

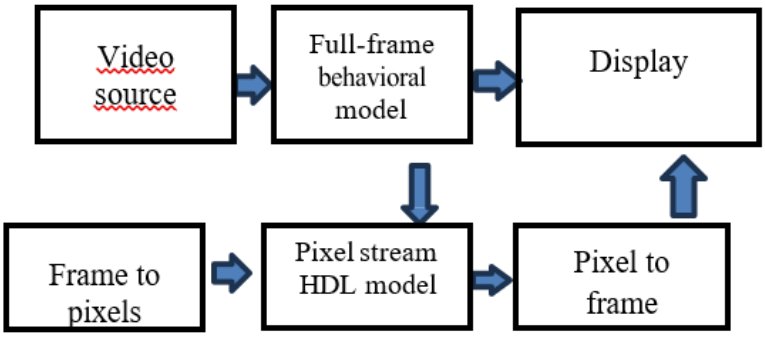

Fig. 3.1 Proposed Canny Edge Algorithm

The full-frame behavioral model is a subsystem, which employs the frame-based Edge Detection block. The

following fig. 3.2 is the subsystem implemented in simulink for full-frame model:

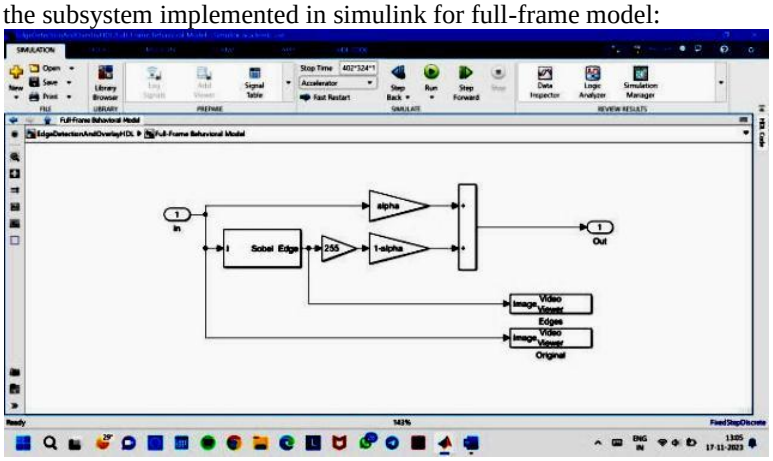

Fig. 3.2 Full-Frame Model

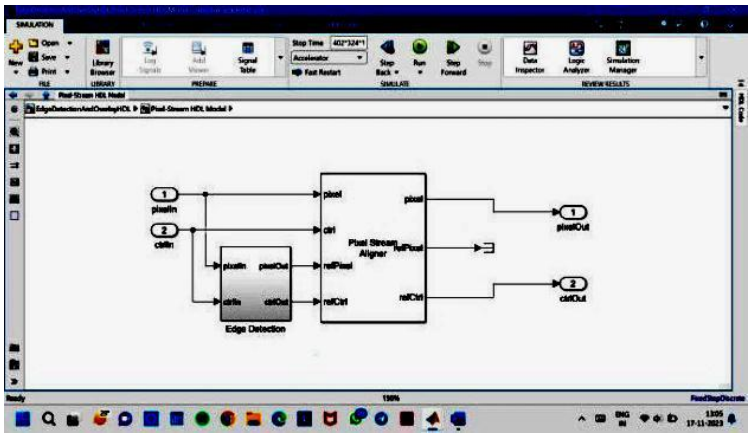

Fig. 3.3 Pixel-Stream Model

Converting an entire frame image to a pixel stream is the aim of the Frame-To-Pixels tool. The active image is enhanced with non-image data to replicate the effects of the horizontal and vertical blanking periods present in actual hardware video systems. The pixel stream subsystem is shown in fig.6 and 7.The Edge Detector block from the Vision HDL Toolbox will introduce latency due to the nature of pixel-stream processing[18], which is not the case with the Edge Detection block in the Full-Frame Behavioral Model. We are unable to directly weight and combine two images to create the overlaid image due to the latency. The two pixel streams are synchronized before the sum using the Pixel Stream Aligner block in order to resolve this problem. The Pixels- To-Frame block is a companion to the Frame-To-Pixels block, which translates an entire image frame into a pixel stream. It uses the synchronization signals to reverse- convert the pixel stream back into the full frame. The Pixels-To- Frame block outputs a 2-D matrix of the entire image, so there's no need to continue with the bus that contains the five synchronization signals.

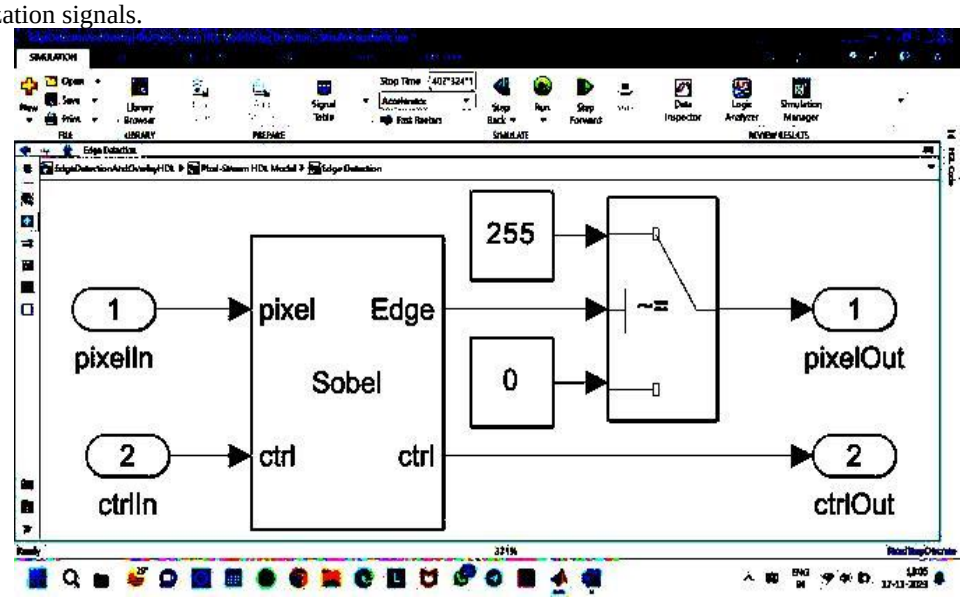

Fig. 3.4 Subsystem for Edge Detection Block

4. HIGH LEVEL SYNTHESIS

The model is hardware compatible, so it is converted into RTL using HDL coder. It is then implemented in Quartus Altera II and simulated in Model-sim.

Fig. 4.1 High Level Synthesis of Simulink Model

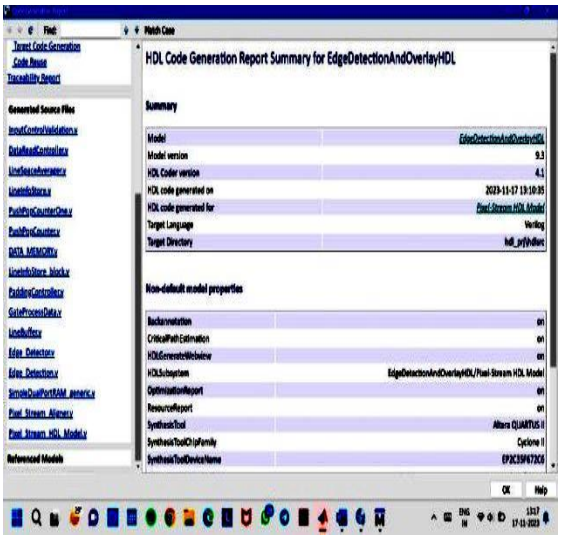

Fig. 4.2 HDL Report of Simulink Model

5. RESULTS

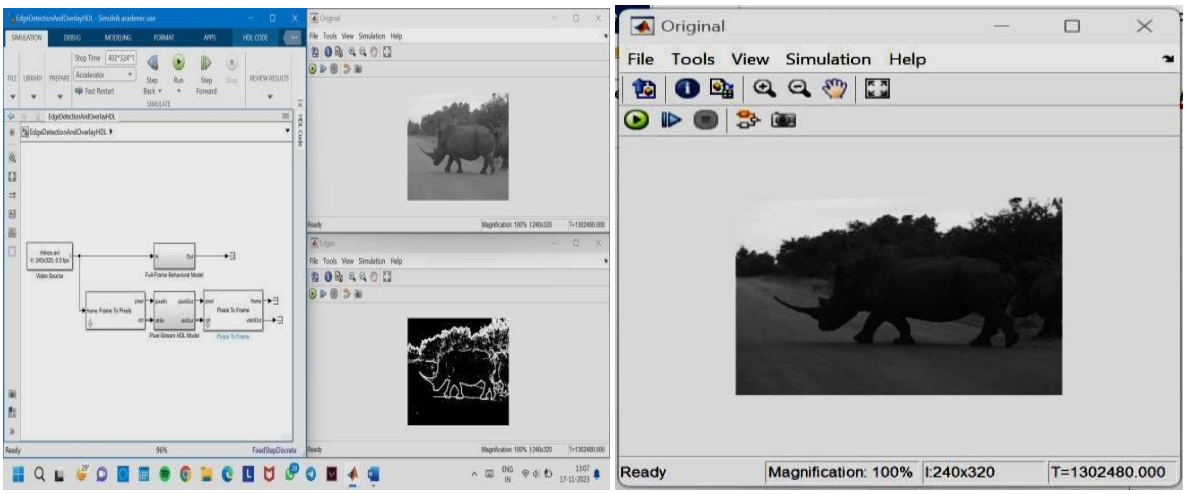

Fig. 5.1 Canny Edge Detection Through Simulink Model **Fig. 5.2 Input Video Stream to Simulink Model**

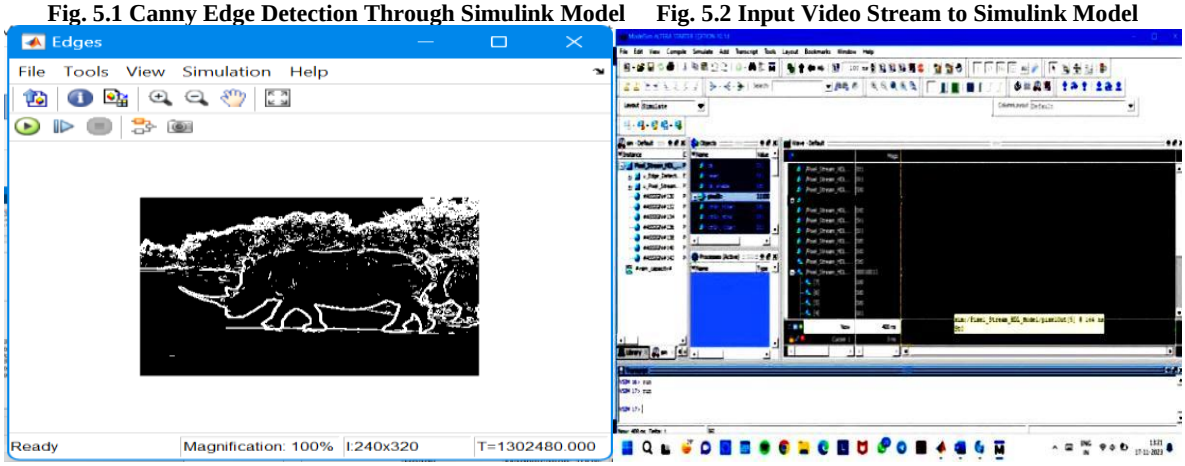

Fig. 5.3 Results Of Canny Edge Algorithm in Simulink **Fig. 5.4 Simulation Result Using Model-Sim**

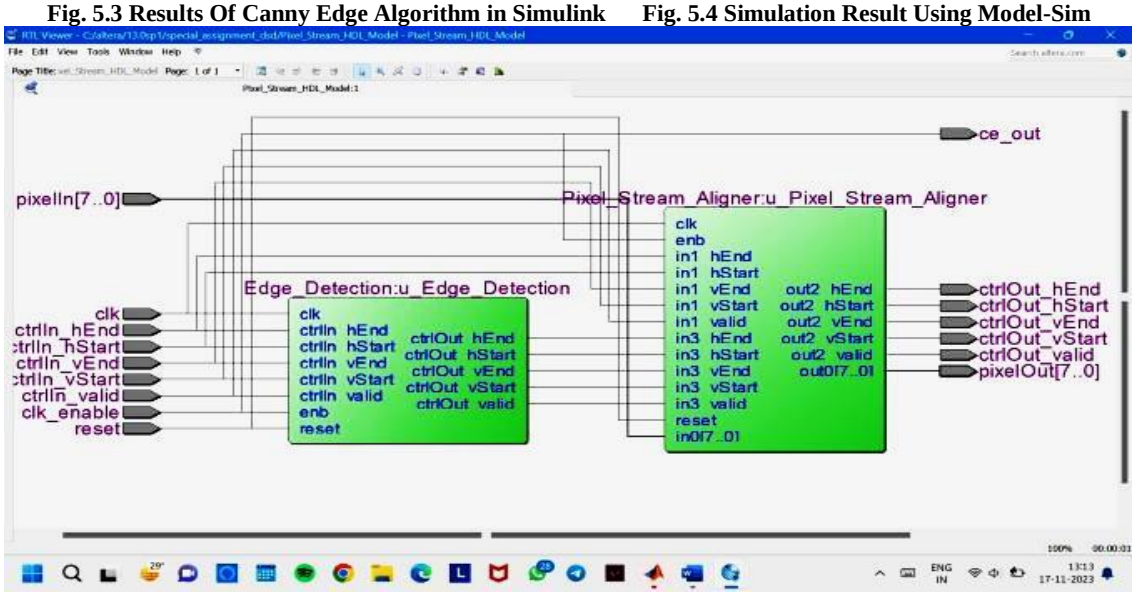

Fig. 5.6 RTL View in Quartus

CONCLUSION

In this paper we have successfully demonstrated the implementation of canny edge algorithm using both MATLAB and Simulink. The analysis and simulation results offer insightful information about the algorithm's functionality and suitability for real-world situations. The input images' edges were successfully found and highlighted by the Canny edge detection algorithm. The simulation results demonstrated the algorithm's ability to precisely localize the edges and suppress the noise.

The canny edge algorithm has been applied to an image as well as in real-time and their results have been analyzed. For the Simulink model, video stream is given as input. The model is successfully implemented in Verilog and the results have been analyzed and simulated. Real-time canny edge detection can be used for various applications like lane detection in autonomous vehicles. The vehicle can comprehend its position on the road and aid in navigation with the help of real-time lane boundary detection. It was discovered that the Canny algorithm's performance is depended on the values of certain parameters, specifically the high and low thresholds for edge detection and the Gaussian filter's (σ) standard deviation. For best outcomes, these parameters must be adjusted based on the properties of the input images.

ACKNOWLEDGMENT

We would like to express our sincere gratitude to all our batch mates who have contributed to the successful implementation of this paper on, canny edge detection using MATLAB. This endeavor would not have been possible without the unwavering support and guidance of Prof. Hardik Joshi of Nirma University, whose expertise and encouragement played a pivotal role in shaping the paper.

REFERENCES

[1] Laigong Guo L and Wu, S (2023) 'FPGA Implementation of a Real-Time Edge Detection System Based on an Improved Canny Algorithm' Journal of Applied Science, 13, 870.

[2] Mostafa, N.N., Ahmed, K., El-Henawy, I. (2021)"Hybridization between deep learning algorithms and

neutrosophic theory in medical image processing: A survey" Neutrosophic Sets and Systems, Volume 45, 2 Pages 378-401.

[3] Ehsan Akbari et al. "Implementing canny edge detection algorithm for noisy image" (August 2020) Bulletin of Electrical Engineering and Informatics 9(4):1404-1410, DOI: 10.11591/eei.v9i4.1837.

[4] Ramnarayan et al., (2019) A Review on Edge detection Technique "Canny Edge Detection" International Journal of Computer Applications (0975–8887), Volume 178 – No. 10.

[5] Malathy H Lohithaswa, 'Canny Edge Detection Algorithm on FPGA'(2015) , IOSR Journal of Electronics and Communication Engineering (IOSR-JECE) e-ISSN: 2278- 2834,p- ISSN: 2278-8735.Volume 10, Issue 1, Ver. 1 PP 15-19.

[6] R. Jeyakumar et al. (2015), "FPGA implementation of edge detection using Canny algorithm," 2015 Online International Conference on Green Engineering and Technologies (IC-GET), Coimbatore, India, 2015, pp. 1-4, doi: 10.1109/GET.2015.7453798.

[7] D Narayana Reddy et al. (2014), 'Canny Edge Detection using Verilog', 21International Journal of Engineering Sciences & Research Technology vol 3 no.6.

[8] R Mehta and N Agrawal, (2014) "Comparative Analysis of Median Filter and Adaptive Filter for Impulse Noise - A Review" International Journal of Computer Applications, Proceedings of RAWCAI, pp 29-34.

[9] Xu, Qian; et al. (2014) "A distributed canny edge detector Algorithm and FPGA implementation". In: IEEE Transactions on Image Processing, Vol. 23, No. 7, p. 2944-2960.

[10] Poonam Dhankhar and Neha Sahu" (2013) A Review and Research of Edge Detection, Techniques for Image Segmentation" IJCSMC, Vol. 2, Issue. 7, pg.86 – 92.

[11] Ranita Biswas and Jaya Sil, (2012) "An Improved Canny Edge Detection Algorithm Based on Type-2 Fuzzy Sets" Journal of Procedia Technology, Volume 4, Pages 820-824

[12] G.T Srivanshan and Dr. C. Chandrasekar (2012) "A Comparison of Various Edge Detection Techniques used in Image Processing" International Journal of Computer Science & I.T (IJCSIT), Vol 9, No 1.

[13] Jamil A. et al. (2012) "Image Segmentation using Edge Detection and Threshold" The 13th International Arab Conference on Information Technology (ACIT),2012 Dec 10-13 ISSN:1812-0857.

[14] N Agrawal, K Venugopalan, (2011) "Speckle reduction in remote sensing images" Proceedings of International Conference on Emerging Trends in Networks and Computer.

[15] Vyas, Megha & Yadav, Vinod & Vyas, Shripati & Joshi, R. (2021). Voltage Sag Mitigation Using Distribution Static Compensator. 10.1007/978-981-15-8586-9_24.

[16] Tirole, R., Joshi, R.R., Yadav, V.K., Maherchandani, J.K. and Vyas, S. (2022). Intelligent Control Technique for Reduction of Converter Generated EMI in DG Environment. In Intelligent Renewable Energy Systems (eds N. Priyadarshi, A.K. Bhoi, S. Padmanaban, S. Balamurugan and J.B. Holm-Nielsen

[17] Vyas, M., Yadav, V.K., Vyas, S., Joshi, R.R. and Tirole, R. (2022). A Review of Algorithms for Control and Optimization for Energy Management of Hybrid Renewable Energy Systems. In Intelligent Renewable Energy Systems (eds N. Priyadarshi, A.K. Bhoi, S. Padmanaban, S. Balamurugan and J.B. Holm-Nielsen). https://doi.org/10.1002/9781119786306.ch5.

[18] Sasanka Sekhor Sharma, RR Joshi, Raunak Jangid, Shripati Vyas, Bheru Das Vairagi, Megha Vyas., 2020, MITIGATION OF TRANSIENT OVER-VOLTAGES AND VFTO EFFECTS ON GAS INSULATED SUBSTATION. Solid State Technology, Volume 63, Issue 5/.

[19] Sujit Kumar et al 2021. Strategies to Enhance Solar Energy Utility in Agricultural Area of Rajasthan State, India. J. Phys.: Conf. Ser. 1854 012013. DOI 10.1088/1742-6596/1854/1/012013.

[20] S. V. . . et. al., "Life Extension Of Transformer Mineral Oil Using AI-Based Strategy For Reduction Of Oxidative Products", TURCOMAT, vol. 12, no. 11, pp. 264–271, May 2021.

[21] H. Kumawat and R. Jangid, "Using AI Techniques to Improve the Power Quality of Standalone Hybrid Renewable Energy Systems", Crafting a Sustainable Future Through Education and Sustainable Development, IGI Global, Pages 219-228, 2023.

[22] Vyas, S., Joshi, R.R., Kumar, V. (2022). An Intelligent Technique to Mitigate the Transient Effect on Circuit Breaker Due to the Occurrence of Various Types of Faults. In: Bansal, R.C., Zemmari, A., Sharma, K.G., Gajrani, J. (eds) Proceedings of International Conference on Computational Intelligence and Emerging Power System. Algorithms for Intelligent Systems. Springer, Singapore. https://doi.org/10.1007/978-981-16-4103-9_21.

[23] Vyas, M., Kumar, V., Vyas, S., Swami, R.K. (2023). Grid-Connected DFIG-Based Wind Energy Conversion System with ANFIS Neuro-Fuzzy Controller. In: Namrata, K., Priyadarshi, N., Bansal, R.C., Kumar, J. (eds) Smart Energy and Advancement in Power Technologies. Lecture Notes in Electrical Engineering, vol 927. Springer, Singapore. https://doi.org/10.1007/978-981-19-4975-3_48.

[24] Muthukrishnan. R and M.Radha (2011) "Edge Detection Techniques for image Segmentation" International Journal of Computer Science & Information Technology (IJCSIT) Vol3, No 6.

[25] R. Jangid et. al., "Smart Household Demand Response Scheduling with Renewable Energy Resources", IEEE Third International Conference on Intelligent Computing and Control System, Organized by Vaigai College of Engineering during May 15-17, 2019 at Madurai, India.

[26] S. Kumar; R. Jangid and K. Parikh "Comparative Performance Analysis of Adaptive Neuro-Fuzzy Inference System (ANFIS) & ANN Algorithms Based MPPT Energy Harvesting in Solar PV System." International Journal of Technical Research and Science, vol. 8, Issue 3, March 2023.
[27] S. Sharma; R. Jangid and K. Parikh "Development of Intelligent Control Strategy for Power Quality Improvement of Hybrid RES Using AI Technique" International Journal of Technical Research and Science, vol. VIII, Issue II, Feb. 2023.
[28] L. Jhala et al., "Development of Control Strategy for Power Management in Hybrid Renewable Energy System" International Journal of Technical Research and Science, vol. VI, Issue XII, Dec. 2021.
[29] P. S. Rajpurohit, et al., "Design of DE Optimized PI and PID Controller for Speed Control of DC Drives" International Journal of Research in Engineering, Science and Management, Volume-2, Issue-6, June-2019.
[30] N. Dhakre, et al., "Optimal Synchronization of PSS and Statcom Based Controller Using De Algorithm" International Journal for Research in Applied Science & Engineering Technology, Volume-5, Issue-XI, Nov.-2017.
[31] P. Megha, et al., "Flow Analysis of Transmission System Incorporating STATCOM" International Journal of Inventive Engineering and Sciences, Volume-3, Issue-1, Dec.-2014.
[32] D. Trivedi, et al., "Optimization of Voltage Stability of Transmission line using UPQC" International Journal of Engineering Research & Technology, Volume-4, Issue-2, Feb.-2015.
[33] S. Lakshmi and Dr. V. Sankaranarayanan (2010) "A study of Edge Detection Techniques for Segmentation Computing Approaches" IJCA Special Issue on Imaging and Biomedical Applications" CASCT, 2010.
[34] Lijun Ding and Ardeshir Goshtasby (2001)"On the Canny edge Journal of Pattern Recognition. Volume 34, Issue 3, March 2001, Pages 721-725.
[35] Canny J. A Computational Approach to Edge Detection (1986). IEEE Trans. Pattern Anal. Mach. Intell. 1986; vol. 8, no. 6, p. 679-698.