Webology (ISSN: 1735-188X)
Volume 18, Number 6, 2021

An Enhancing Canny Edge Detection Using SUSAN Filter Based On Deep Learning Algorithm For Improved Image Edge Detection

[*]A. Bhuvaneshwari [1] , Dr. S. Britto Ramesh Kumar [2]

[1]Research Scholar, Department of Computer Science, St. Joseph's College (Autonomous), Affiliated to Bharathidasan University, Tiruchirappalli, Tamil Nadu, India.

[2]Assistant Professor, Department of Computer Science, St. Joseph's College (Autonomous), Affiliated to Bharathidasan University, Tiruchirappalli, Tamil Nadu, India.

Abstract

In digital images, edges can be detected by applying mathematical techniques in order to identify the locations in which the brightness of an image unexpectedly changes. Edges are generally curved lines segmented into sharp adjustments in picture brightness. This paper proposes a technique for finding sharp and precise edges. Additionally, a deep learning method combined with a Gaussian Canny filter is used to replace the SUSAN filter of detection. This paper aims to evaluate how efficient deep learning is by combining the Canny method and the Smallest Uni value Segment Assimilating filter. Additionally, adding these SUSAN filters will eliminate image noise, and help users detect the corner of pictures for the improved canny algorithm. SUSAN performs better than traditional edge detectors, and produces more accurate edges.

Keywords: Canny, SUSAN Filter, Edge Detection, CNN

1. Introduction:

Computers that mimic and demonstrate "human" thinking capacity connected with human mind, such as "having to learn" and "problem-solving," were historically referred to as "Artificial Intelligence (AI). Machine learning (ML) is an Artificial Intelligence technology that creates systems that can learn and develop without it being programmed. ML, like AI, focuses on creating computer systems that acquire data and utilise it to learn on their own. Deep Learning (DL) is being used to create completely automated systems like self-driving vehicles. These vehicles can reorganise challenges and improve situational awareness due to their sensors and computer analytics. Image processing is the process of enhancing the quality of an image or extracting useful information from it. To process massive amounts of images fast and effectively, AI Image Processing combine powerful computational technology with Machine

Learning and computer vision. A CNN is a Deep Neural Network that is used primarily for analyzing visual imagery. It belongs to the class of Deep Neural Networks called Convolutional Neural Networks (CNN) or Conv Nets. Multilayer perceptrons are the basis of CNNs, which require very little pre-processing.

2. Related Works:

The recognition of object boundaries with noise using the standard Canny algorithm is described in this study, which offers poor feature extraction. ML Classification Methods are used to improve the Canny operator and the morphological filter boundary edge detection algorithms in order to overcome this challenge. The Morphology filter smoothness, on the other hand, blurs the Canny edge recognition technique, making it more difficult to recognise a corner edge. [1] In CNN, pre-defined kernels are used as filters or masks for image processing. This work contains 41 separate blurrings, edge detection, sharpening, discrete cosine modification, and other general-purpose of these kernels. The Convolution Neural Network is an initial layer. The proposed CNN avoids repeated calculation and speeds up training by 30% without sacrificing accuracy. This work, however, is time-intensive and computationally difficult [2] In this paper, the Sobel operator is used to detect edges with a combined median filter and calculate PSNR for a better image. In addition, using the combined median filter, the Sobel operator identifies the edges of an image and eliminates irrelevent noise. In this paper, compare and analyze several types of edge detection approaches, including Prewitt, Robert, and Canny, using Matlab [3] To solve this problem, a combination of median filters is used to recover edges. This includes those detected by Roberts, Sobel, Prewitt, and Canny, and then processed by a median filter. A number of digital images are used in simulations. In comparison to the Sobel operator, Prewitt provides higher PSNR. Similar results indicate that Canny is better at detecting edges than other algorithms [4]

The edge detection thresholds are automatically calculated based on the type of block by modifying FPGA Canny's algorithm. Moreover, the block-based algorithm significantly decreased the area and increased the frequency of the image. Additionally, the edge detection performances of the proposed algorithm are better than the existing algorithm. As the newly developed algorithm preserved more useful edge information, it gives better results than the current algorithm. [5] This paper is to present a method of segmenting an image using a CNN model based on a GUI application to detect the fractured area in bone images. This method uses the CNN algorithm to overcome both of these problems. Simulations have shown that the proposed method enables edge detection at aggregate scales to be performed more efficiently [6]. The paper contains many masks including Prewitt and Canny that are used to extract edge information from a wide variety of images. Canny has consistently given the sharpest pictures and finest edge continuity in comparison to other edge detectors. Compared to Prewitt, the canny produces the sharpest and clearest edge. An image is cleaned by applying a Gaussian filter to the Canny Edge Detector. One of the advantages of that is an improvement in the signal-to-noise ratio. The method of non-maxima suppression yields a one-pixel-wide set of ridges [7]. For efficient edge identification, propose the Pixel Difference Network, a simple, lightweight architecture. PiDi Net uses unique pixel differential convolutions to improve task

performance by combining basic edge detection operators with common convolutional operations in modern CNNs. Several experiments on the BSDS500, and Multi cue are provided to show its usefulness as well as its high learning and inference efficiency [8]. This study presents a parallel design for FPGA-based Sobel edge detection. This strategy reduces the design's complexity while simultaneously shortening the processing time [9].

3. Edge detection:

Edge detection steps:

- ✓ Smoothing
- ✓ Enhancement
- ✓ Detection
- ✓ Localization
- ✓ Edge thinning

4. Proposed work:

It detects edges using a Gaussian-based algorithm. Noise has no effect on this operator. It detects picture features without changing or impacting them. It performs well in terms of localization because it extracts picture features without modifying them and Noise sensitivity is reduced. The Canny edge detection algorithm is composed of 5 steps:

1. Image noise reduction
2. Image gradient calculation
3. Non-maximum suppression
4. Applied double threshold
5. Tracking edges by Hysteresis.

4.1 Noise Reduction:

To remove the noise, use a Gaussian filter to smooth the image. Canny processing reduces noise and unnecessary features, using a Gaussian filter to process images. Although the canny procedure is great for detecting edges and reducing noise, it does not work well when detecting joint connections and corner edges. The canny method struggles with finding junction edges and junction connections. By smoothing with a Gaussian filter, the Canny method is blurred, making edges more difficult to discern. There's no way of finding the neighboring pixels at the corner edge, so there are no open final edges and no connections.

The SUSAN approach can now be applied to locate this complex corner edge. Replacing the canny gaussian filter and SUSAN filter with this one gives better connecting edges and produces good connections. The edges are detected more accurately. This nonlinear SUSAN filter removes noise from images, finds the image edge, and finds the corner pixels of the image. By SUSAN scanning the rectangular order, nearby pixels are captured and applied to determine whether the current pixel value is an edge location or not. In this case, edge noise is reduced in the region. The SUSAN filter was created in place of the Gaussian filter since it was difficult to identify corner pixels on the image boundaries using that filter. It extracts probable corners with the SUSAN filter and then uses the 5x5 template to identify exactly where corners are situated.

4.2 Gradient Calculation:

The Gradient computation stage calculates the gradient magnitude of the image using applying edge detection approaches to determine the edge strength and direction. Edges represent a change in the picture intensity of dots. The simplest technique to detect it is to use masks that highlight the picture intensity variation in both horizontal (x) and vertical (y) directions (y). The gradient vector of an image represents modify in picture brightness (intensity) of an image in a two-dimensional space. It can be determined how strong the gradient is when determining edge positions.

The (image) gradient vector representation is calculated by convolving the original picture with a derivative filter to obtain the first-order partial derivatives. A possible edge is simply identified by the values with the highest rates of change, hence possible edges possibilities are the derived values with the largest magnitude.

4.3 Non-maximum suppression:

To smooth out the edges, use non-maximum suppression. Unnecessary pixels that aren't part of an edge should be removed. If the picture intensity gradient magnitude amount is greater than a certain threshold, pixels are accepted as edges.

The non-max-suppression phases are dependent on these inputs:

- Create a 0 sized matrix with the same dimensions as the original picture gradient intensity array.
- Determine the edge direction using the angle matrix's to angle value.
- Check to see if the pixel dots in the same orientation have a greater density than the one being processed right now.
- Retrieve the image after the non-max suppression technique has been applied.

4.4 Double threshold:

Apply two threshold stages, to determine the fine edges. A dual threshold step tried to differentiate between three types of pixel resolution: strong, weak, and irrelevant: Strong pixels are pixels that have intensity so high that are sure they contribute to the final edge.

- Use a two-threshold technique to locate likely edges. The double threshold step seeks to distinguish between three types of pixels: strong, weak, and irrelevant:
- Powerful strong pixels are those with such a high intensity that they are certain to add to the final edge.
- Weak pixels are those with a pixel intensity that isn't high enough to be called strong, but not low enough to be regarded irrelevant for edge detection.
- Remaining pixels are ignored since they are irrelevant to the edge.

Dual thresholds now apply to:

- To find the strong pixels, a strong threshold is used (pixel intensity greater than the high threshold)
- The irrelevant pixels are identified using a low threshold (intensity lower than the poor threshold.

4.5 Edge Tracking by Hysteresis

Large and powerful edges are considered as "sure edges" and can be placed into the finalized edge pixels right away. Poor edges are only mentioned if they are linked to strong edges. In addition, noise or other minor differences are unlikely to result in huge edges. As a result, strong edges in the original image will nearly entirely be due to true edges. Real edge boundaries or noise/color changes can also cause weak edges. This last category will most likely be dispersed across the entire image individually of edges, with only a tiny portion next to strong edges. Weak edges caused by real edges are far more likely to be linked to strong edges directly. Identify the gradient in an image that has been cleaned. This proposed technique is used to analyze the maximum edges. When the thresholding is finished and the sharp edge lines are obtained, the output picture is thinned further. After training and testing, non-max suppression in a picture results in a more accurate measure of true edges.

4.6 Convolutional Neural Network:

The proposed edge detection approach is evaluated using Convolutional Neural Network architecture in this study. An input layer, a hidden layer, and an output layer make up a Convolutional Neural Network. This activation function and last convolutional layer are referred to as hidden layers because they encapsulate the inputs and outputs of hidden units in any feed-forward Neural Network. A Convolutional Neural Network's hidden layers are composed of several layers that perform convolutions. This usually consists of a layer that multiplies the convolution kernal filter's grid with the layer's input matrix. In Re LU, this grid

is commonly employed as the activation function. The convolution technique creates a feature map by moving the convolution kernel along the layer's input matrix. This feature map is again added to the input of the down-sampling and up-sampling layers that follow. When two layers are "completely linked," so each node in the very first layer is connected to each and every node in the second. In the last fully connected layer, a soft - max function is often identified, which provides a probability value between 0 and 1 at each of classified labels in the model. Finally, CNN employs decision-making in predicting edges with more accuracy.

6. Experimental Result:

This work is indented to measure the edge detection efficiency of different algorithms in terms of Accuracy, Precision, Recall, Specificity and F-Score. Standard edge detection algorithms are implemented by fetching corresponding codes from Github [13]. New and proposed method algorithms are converted from document files to codes with the help of Code Master Library (CML) [14] which is invoked through VC++ programming Language. A dedicated User Interface (UI) is designed using Visual Studio [15] to encapsulate the functionalities of the discussed methods. BSDS500 dataset [16] is used to train the proposed method and to test the performance of the compared methods. The user interface is given in Figure [4]. The entire dataset is divided into 10 equal data chunks and the performance metrics are measured for each data chunk individually for detailed analysis report generation. Parameters such as True Positive, True Negative, False Positive and False Negative are measured for the methods. Performance evaluation metrics such as Accuracy, Precision, Recall, Specificity and F-Score are calculated accordingly for the measured data. The UI is capable of generating a report files and presenting the result in plotted graph format based of the observed data.

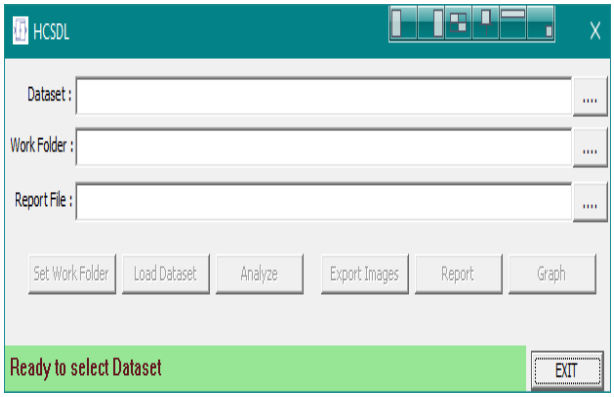

Figure [1]: User interface

7. Results and Discussion:

The tests are conducted by dividing the entire dataset into distinct pieces to comparing the overall dynamic performance of existing techniques with the proposed model. All procedures at every time chunk are evaluated for Accuracy, Precision, Recall, Specificity, and F-Score to produce the comprehensive picture.

7.1. Accuracy:

The proposed HCSDL ranked in first with the best prediction of 99.07 percentages, DLED is in second place.

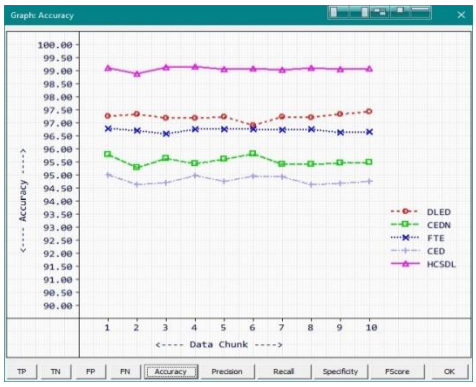

Figure [2] accuracy comparison graph

7.2. Specificity:

The HCSDL classification technique, according the conclusions, has the highest Specificity Level of 99.08 percentages.

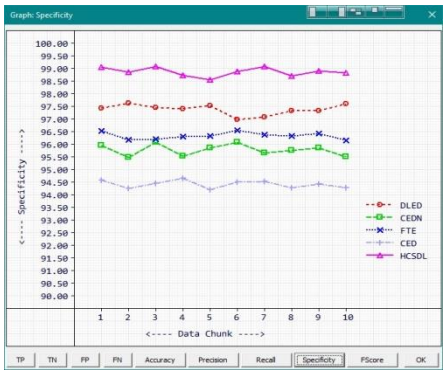

Figure [3] specificity comparison graph

7.3. Precision:

HCSDL has a precision value of 99.09%, making it the most accurate method.

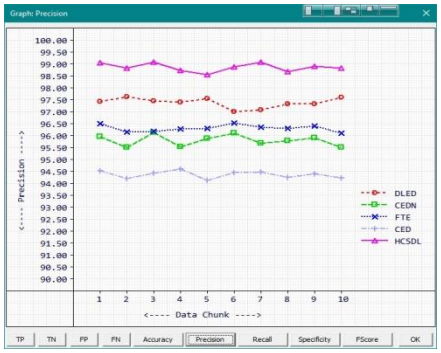

Figure [4] precision comparison graph

7.4. Recall:

The calculated findings indicate that HCSDL has the highest Recall Value of 99.58 %. The F1-Score for the approaches is calculated, HCSDL has the best amount of 99.15%.

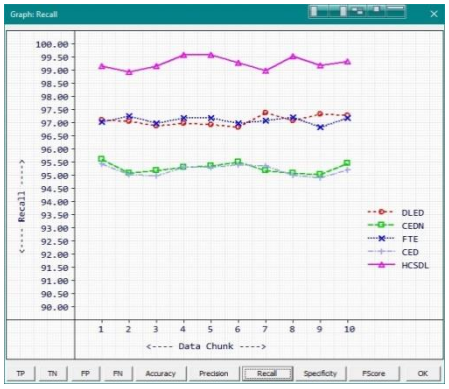

Figure [5] Recall comparison graph

7.5 F1 Score:

The F1-Score for the methods is determined, HCSDL comes out on top with a score of 99.15 percentage.

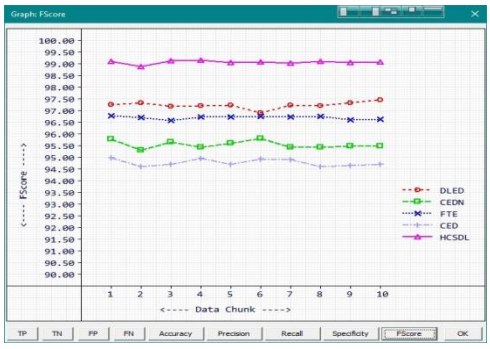

Figure [6] F1Score comparison graph

8. Conclusion:

The SUSAN filter implementation in VC++ is used to replace the Gaussian Filter canny technique to increase the accuracy of the edge detection problems. The pre-trained neural network is used to identify more edge pixel values using a proposed method. An edge is predicted directly from a picture patch using a CNN. As a result, the canny method and SUSAN filter become more efficient without compromising image quality when applying such systems. Detecting more edges enhances the edge fidelity and efficiency.

References:

[1] Shigang Wang, Kai Ma, Guoqiang Wu, " Edge Detection of Noisy Images Based on Improved Canny and Morphology", ECICE (IEEE), 2021

[2] Yousun Shin, Young Min Kwon, “Extension of Convolutional Neural Network with General Image Processing Kernels”, IEEE, 2018
[3] Yatendra Kashyap , Anirudh Vyas , Rahul Raghuwanshi , Raju Sharma, “Edge Detection Using Sobel Method With Median Filter”, International Journal of Modern Trends in Engineering and Research (IJMTER) Volume 02, Issue 05, May – 2015
[4] Preeti Topno , Govind Murmu , “An Improved Edge Detection Method based on Median Filter”, IEEE 2019 Devices for Integrated Circuit (Dev IC).
[5] B Pradeep Kumar, B hari Kumar, ”FPGA Implementation of Modified Canny Edge Detection Algorithm”, International Journal of Management Technology And Engineering, volume IX, Issue XII, DECEMBER/2019
[6] Siva S Sinthura; Y. Prathyusha; K. Harini; Y. Pranusha; B. Poojitha, “Bone Fracture Detection System using CNN Algorithm” 2019 International Conference on Intelligent Computing and Control Systems (ICCS), IEEE, 2020.
[7] Priyam , Diganta Dey , Shreya, Dipanjan Polley, “Edge Detection by Using Canny and Prewitt”, International Journal of Scientific & Engineering Research, Volume 7, Issue 4, April-2016
[8] Zhuo Su, Wenzhe Liu, Zitong Yu, Dewen Hu Qing Liao, “Pixel Difference Networks for Efficient Edge Detection”, Computer Vision and Pattern Recognition IEEE, 2021
[9] Varun Sanduja, Rajeev Patial ,“Sobel Edge Detection using Parallel Architecture based on FPGA” International Journal of Applied Information Systems (IJAIS), ISSN- 2249-0868, 2012
[10] https://github.com/
[11] http://codershunt.weebly.com/projects/algorithm-to-code-converter
[12] https://visualstudio.microsoft.com/
[13] https://paperswithcode.com/dataset/bsds500#:~:text=Berkeley%20Segmentation%20Data%20Set%20500%20(BSDS500)%20is%20a%20standard%20benchmark,interior%20boundaries%20and%20background%20boundaries.