Proceedings of the Third International Conference on Artificial Intelligence and Smart Energy (ICAIS 2023)
IEEE Xplore Part Number: CFP23OAB-ART; ISBN: 978-1-6654-6216-7

Improved Performance of Canny Edge Detection in Low-Light Conditions

Pradeep surya Dadi
Department of Mechatronics Engineering
Chennai Institute of Technology
Chennai, India

Saranya G
Department of Computer Science and Engineering,
Amrita School of Computing,
Amrita Vishwa Vidyaapeetham
Chennai campus,
Chennai, India

Tamilvizhi T
Department of Computer Science and Engineering
Panimalar Engineering College
Chennai, India
tamilvizhi.phd.it@gmail.com

Leema Nelson
Department of Computer Science and Engineering
Chitkara University Institute of Engineering & technology,
Chitkara University,
Punjab, India
leema.nelson@chitkara.edu.in

Surendran R
Department of Computer Science and Engineering
Saveetha School of Engineering
Saveetha Institute of Medical and Technical Sciences,
Chennai, India
surendran.phd.it@gmail.com

2023 Third International Conference on Artificial Intelligence and Smart Energy (ICAIS) | 978-1-6654-6216-7/23/$31.00 ©2023 IEEE | DOI: 10.1109/ICAIS56108.2023.10073915

***Abstract*- Visual simultaneous localization and mapping (SLAM), which has numerous potential uses, is swiftly emerging as a significant development in embedded vision. In our analysis, there remain several unsolved dilemmas about SLAM that are visual, making it less favored over Lidar SLAM. Because it offers an affordable solution for mapping and navigation in mobile robots. These mobile robots can be defined as robots whose unique base is not fixed, so as technology establishes and advances, we. In this report, a real-time light, that is, dim or image or video in low light problem enhancement algorithm is proposed that significantly improves the overall performance of the Canny Edge part detection algorithm, which usually may be useful for benefit recognition in low light issues in cellular robots for several mapping and navigation tasks.**

Keywords—Edge detection, Canny edge algorithm, Facial Expressions, Active Vision, Surveillance, Mobile Robots, Slam.

I. INTRODUCTION

Edges tend to be one of the most essential attributes of an object or a certain location in relation to feature recognition, classification of an object, location, and mapping [1]. This is, without doubt, a pair of affixed pixels forming the boundary between two disjointed regions. Such discontinuities usually fit:

- Level discontinuities
- Area path discontinuities
- Changes in material properties
- Variations in scene illumination

These sides tend to be further classified as: horizontal side, right edge, and diagonal edges. Therefore, edge detection is a way of segmenting an image into discontinuous parts. It's a method that is certainly preferred for images that are digital, such as: design recognition, image morphology, and function reduction. Advantage recognition allows people to look at photo features. For a start, this is certainly a considerable level. That is gray. This surface denotes the finish of morphology and features. Only the final features. Within an image while keeping its functions, which are often architectural. Detection providers. They are this surface denotes the finish of morphology and features.

Although this feature detection algorithm carries out well in great lighting problems, it doesn't identify edges or simply any feature in dim-light-to-dark dilemmas. An algorithm has been proposed which will be used to enhance the image or video before applying the medial side. This is certainly Canny Edge, which in turn optimizes the image/video frame-by-frame and detects so many more edges/features, therefore improving the overall performance by approx 60%. So the software is of great advantage to the OPENCV Python collection and Canny Edge advantage detection collection, which are more beneficial than mathematical digital tools [2]. This study deduces a MATLAB algorithm that produce far better outputs. Python and MATLAB are utilized in this research study. Python is one of the most widely used resource program coding languages [3].

II. METHODS USED

The proposed work has employed the medial side. This is certainly the method that Canny Edge is explaining, as the advanced detection technique is just a multistep algorithm to identify the edge of any input photo. The below mentioned to be followed while finding the edges of an image in Fig. 1.

$$S = I * g(x,y) = g(x,y) * I \quad (1)$$

$$g(x,y) = \frac{1}{\sqrt{2\pi\sigma}} e^{\frac{x^2+y^2}{2\sigma^2}} \quad (2)$$

Gausian Distribution = g(x,y).

Where I = input image

Calculate the derivative of the filter with respect to X and Y dimensions and convolve with the gradient magnitude along the dimensions [4]. Moreover, the direction of the image can be calculated using the tangent of the angle between the two dimensions.

$$\nabla S = \nabla(g * I) = (\nabla g) * I \quad (3)$$

Authorized licensed use limited to: VIT University- Chennai Campus. Downloaded on February 25,2024 at 15:58:31 UTC from IEEE Xplore. Restrictions apply.

$$\nabla S = \begin{bmatrix} gx \\ gy \end{bmatrix} * I = \begin{bmatrix} gx * 1 \\ gy * 1 \end{bmatrix} \quad (4)$$

$$\nabla g = \begin{bmatrix} \frac{\partial g}{\partial x} \\ \frac{\partial g}{\partial y} \end{bmatrix} = \begin{bmatrix} gx \\ gy \end{bmatrix} \quad (5)$$

The above convolution results in a gradient vector that has magnitude and direction.

$$(S_x, S_y) Gradient\ Vector$$

$$magnitude = \sqrt{{S_x}^2 + {S_y}^2} \quad (6)$$

$$direction = \theta = tan^{-1} \frac{S_y}{S_x} \quad (7)$$

Fig. 1. The greyscale image

In Fig. 2, the Gaussian Derivatives contribute to the edges with Gradient Magnitude of left and right images.

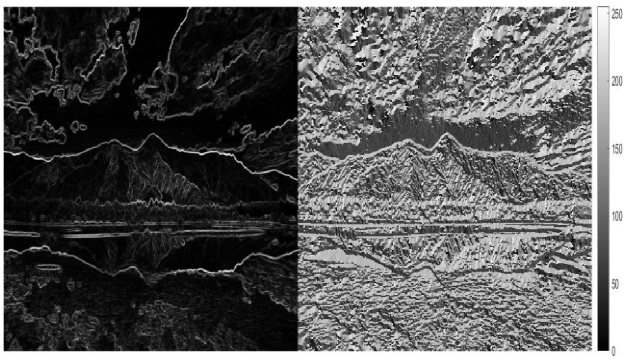

Fig. 2. X-Derivative gaussian and Y-derivative gaussiasn

Along with a benefit, it's generally seen that few things make the presence regarding the advantage better. So we can neglect those side things which don't contribute more towards feature visibility [5]. To ultimately achieve the same, the non is used by us Maximum Suppression method. Here we mark the real things from the bend regarding the edge where in fact the magnitude is largest. This is got by buying a maximum along with a piece regular to the curve [6].

Consider the edge of the figure, this is certainly below, which has three advantage points. Believe the point (x, y) the point obtains the gradient, that is, the largest of sides. Search for the side points of the course perpendicular to the advantage and verify if their particular gradient is less than (x,y). Then those non-maximal things can be controlled as shown in Fig. 3

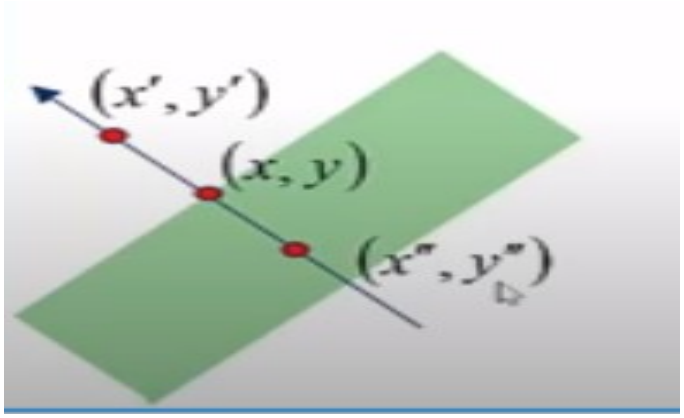

Fig. 3. Across the bend, the values are lower than the (x, y) gradient.

The image/video frame has been optimized by employing a set of operations meant to modify the image before applying the Canny Edge advantage detection method. Transform the feedback image or each video frame to grayscale: an image that is grayscale one in which the worth of each pixel is a single sample indicating simply a quantity of light; this is certainly, conveys just intensity information in portrait digital photography, computer-generated imaging, and colorimetric [7]. Grayscale photographs, often known as black-and-white or monochrome, which are certainly grey, are entirely made up of grayscale hues.

The contrast goes from black at the level that is certainly the least expensive to white at the greatest. The grayscale images need less information for each pixel compared to full-color photos. The Gray color is one where the purple, green, and blue elements all have equal intensity in the RGB room, and thus it's just required to specify an intensity that is solitary for every pixel. Hence, further computations become a good deal easier [8]. Opencv is a massive computer system that is certainly an open-source, machine learning, and image processing library. It may analyze photos and films to discover products, people, and sometimes even the handwriting of a human. In this guide, we are going to view just how to convert a color movie to grayscale [9].

A. Method:

• Import the cv2 module.
• The video clip file is changed making use of the cv2.VideoCapture() function.
• Creating a cycle is certainly countless.

Authorized licensed use limited to: VIT University- Chennai Campus. Downloaded on February 25,2024 at 15:58:31 UTC from IEEE Xplore. Restrictions apply.

• With the read() purpose, retrieve the video clip frames within the cycle.
• To transform the frame to grayscale, use the cv2.cvtColor() function aided by the argument cv2. Color BGR2GRA Y.

The cv2.imshow() purpose is employed to show the framework.

```
# importing the module
import cv2

# reading the video
source = cv2.VideoCapture('# Add input file file here')

# running the loop
while True:

    # extracting the frames
    ret, img = source.read()

    # converting to gray-scale
    gray = cv2.cvtColor(img, cv2.COLOR_BGR2GRAY)

    # displaying the video
    cv2.imshow("Live", gray)

    # exiting the loop
    key = cv2.waitKey(1)
    if key == ord("q"):
        break

# closing the window
cv2.destroyAllWindows()
source.release()
```

Fig. 4. Python Code for cv2 module

Calculate the histogram regarding the image, this is certainly grayscale: The histogram of an image offers details about the brightness comparison and power circulation in an image [10]. cv2.calcHist() has been applied to calculate. The histogram of the image in Fig. 4. The histogram depends on the regularity of the luminance in the image. Therefore, the colors which are purchased are graphed on the x-axis in addition to the number of pixels for every shade regarding the y-axis, giving an overview associated with the circulation of colors in the image. Whenever an image is converted to grayscale, the colors tend to be limited to 100 or 255 levels of gray [11].

A histogram is just a visual depiction associated with regularity with which a color that is different can be found in an image. It is a plot or graph that depicts an image's intensity of circulation. This is a story with pixel values (usually ranging from 0 to 255) within the X-axis; additionally, the true range of pixels within the photo is within the Y-axis [12].

The cv2.calcHist() strategy in Python opencv determines the histogram of 1 or even more arrays. This enables you to calculate the histogram of single and photos which are multichannel photos. We only used a channel that is solitary in this article in Fig. 5.

calcHist() syntax cv2.calcHist(images, channels, mask, histSize; this is a float32 or uint8 image resource. networksNetworks: It denotes the index regarding the station, enclosed in square brackets. For an image that is grayscale, its value is [0].

Maskmask: it's an image of a mask. The histogram associated with the image that is supplied as "None."

tSize: This parameter reflects the actual range of bins provided as a record.

Range: the power is represented by an appreciative range.

Needed Segments:
• opencv (cv2)

• Numpy

• Matplotlib

The input image is to be converted to histogram.

Fig. 5. Input image

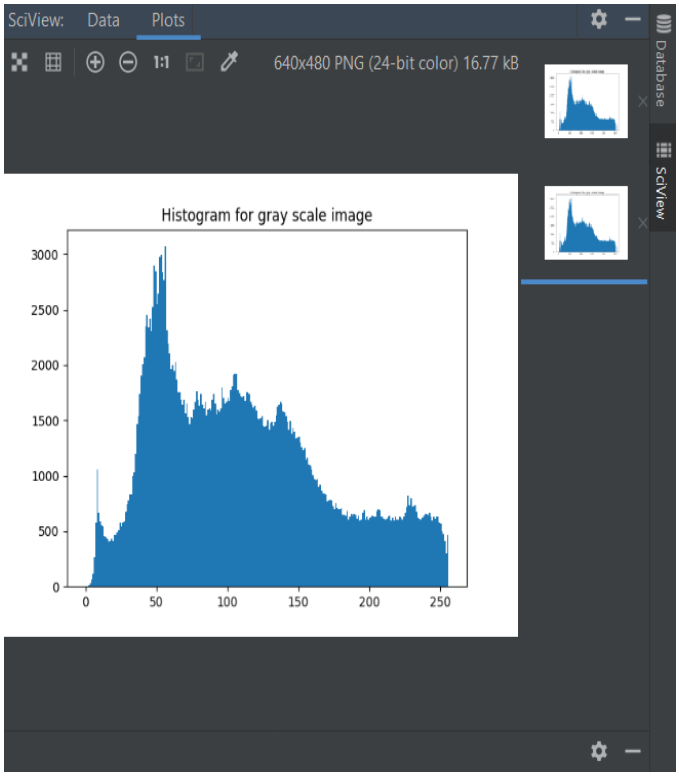

Fig. 6. A histogram of grayscale images without a mask

The signal for processing the histogram of a total image is certainly the grayscale supplied under. Hist is really a (256,1) array in this instance. Each range product presents the number

Authorized licensed use limited to: VIT University- Chennai Campus. Downloaded on February 25,2024 at 15:58:31 UTC from IEEE Xplore. Restrictions apply.

of pixels with the tone value that is connected. In this case, the histograms are plotted using the matplotlib package. A graphing hist() method. This is an example of a picture histogram in Fig. 6 [13].

The mask is made up of a black image with the same proportions as the filled image and some white areas that match the image where you want to determine the histogram.

Calculate the cumulative circulation of the grayscale histogram and use scale transformation to affect the brightness and comparison:

Histogram equalization can be used automatically to change the intensity levels of picture pixels. Histogram equalization entails adjusting the intensity values so that the resulting image closely resembles a specified histogram. Histeq, the histogram equalization function, by default attempts to match a histogram, that is, flat 64 containers; however, you could provide a different histogram instead. Hexagonal histogram equalization is a technique for modifying strength values. This example shows how to use simple techniques to replace the comparison of a grayscale image histogram equalization. The picture is certainly an initial reasonable comparison, with most of the pixel values dropping in the middle regarding the intensity range. Histeq yields an image with pixel values equally distributed within the range.

The histogram is a collective histogram in which the straight axis provides not only the counts for the single bin, but also counts for the container plus all bins for lower values of the reaction variable. After calculating the histogram's collective distribution, locate the items in the histogram for clipping and calculate the value of variables alpha and beta, which determines the factor by which the image power must be altered based on the histogram's collective distribution. The factor alpha will be beta and increased in this product to obtain an illuminated version of the image.

After optimizing the brightness and contrast of the image by modifying the power distribution, we make the following adjustments using the Canny Edge detection method for much better side detection:

The image is convolved through a Gaussian filter, which is really a low-pass filter that removes the outlier pixels or even high frequency elements through the image to cut back sound in Gaussian Blur operation. The edges have power, this is certainly low eliminated. The opencv Gaussian filtering provides a cv2.GaussianBlur() solution to blur a graphic by using Gaussian Kernel. Each pixel in an image is multiplied by a Gaussian Kernel, which is a square array.

Conversion to grayscale before applying Canny Edge detection. The Canny Edge technique works closely with a simpler picture, and the grayscale photos are mostly the ones that are simplified. Grayscale is probably decreasing complexity: coming from a 3D pixel value (Roentgen, G, B) up to a price, that is, 1D. Numerous jobs like advantage detection try not to fare better with 3D pixels. Converting a graphic to grayscale does not break down the image by any means because after all, the Canny Edge technique only produces black colored photos which are white. So grayscale doesn't impact edge detection in any way.

Now, we use the Canny Edge Detection on the edge (this is certainly cv2. Canny strategy in OpenCV. Canny Edge advantage recognition is really an algorithm that multistage detects an array of edges in pictures. When we compare the output, the true difference can be seen in the total results. When a picture is preprocessed using the method described above, many more functions are detected than in the original image.

A Gaussian filter is used to remove noise from the input image. Processing the Gaussian filter byproduct to look for the gradient of image pixels with magnitude along the x and y dimensions. Considering a small grouping of neighbors for almost any curve within a direction perpendicular to the supplied edge, we suppress the benefit that this pixel is undoubtedly non-max. Finally, we use the Hysteresis Thresholding strategy to protect pixels larger than the gradient magnitude and ignore pixels smaller than the paid-off limitation value. Before delving into the steps, here are the three conclusions reached by J.K Canny Edge, the algorithm's creator: Good Detection: The best sensor must remove the possibility of false positives and negatives. Good localisation: The detected edges must be close to the genuine edge response. Single: The detector must reunite only one point from each side. Steps to take the Canny Edge Algorithm: Image Smoothing or Noise Removal: During the noise, the pixel is almost certainly not similar to its neighboring pixels that are neighboring. This may result in poor or recognition that isunacceptable. The noise, preventing the desired edges in outcome images. In the under instance, we are convolving the filter . This is certainly the Gaussian kernel g (x, y) having an image we want to avoid comparable, we apply a Gaussian filter that will evolve due to the photo and be removed. Here we wish to make sure that any provided pixel should really be comparable to its pixels which are neighbouring outputs, so we use the matrix [1 1, 1] to keep the similarity between pixels and take away the sound.

III. RESULTS & ANALYSIS

The above method has been tested on a set of real-time dim-lighted images/videos from an indoor environment. Firstly, the Canny Edge detection was tried on the original dim-lit images/video and then the same has been tried on the processed images. The images and videos were taken through a normal RGB mobile camera. The left image in the above Fig. 7 is an image of a flower taken in dim light. The left image shows the features detected by Canny Edge detection, which is almost nil.

Authorized licensed use limited to: VIT University- Chennai Campus. Downloaded on February 25,2024 at 15:58:31 UTC from IEEE Xplore. Restrictions apply.

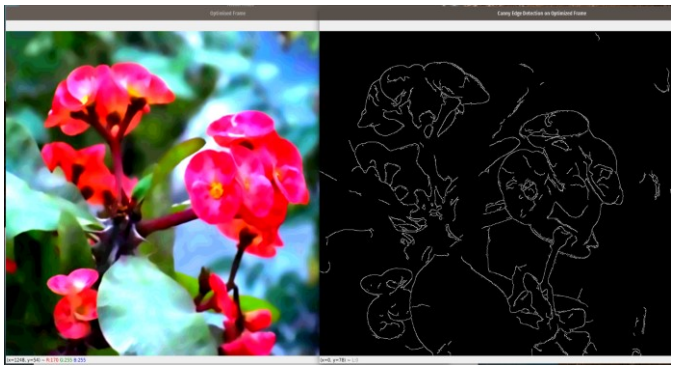

Fig. 7. Canny Edge detection in image 1

The left image in the above figure was obtained after pre-processing the dim-lighted image with the proposed method, and the right one was obtained by using the Canny Edge detection method; the performance appears to have improved significantly. The above algorithm was also tested on video datasets, yielding similar results. Here are a few frames from the video dataset: Canny Edge is a popular edge detection algorithm for computer vision. However, it has several shortcomings when used in indoor environments with insufficient light. In this paper, we present a novel algorithm that uses improvisation to improve edge detection performance in indoor environments in Fig. 8.

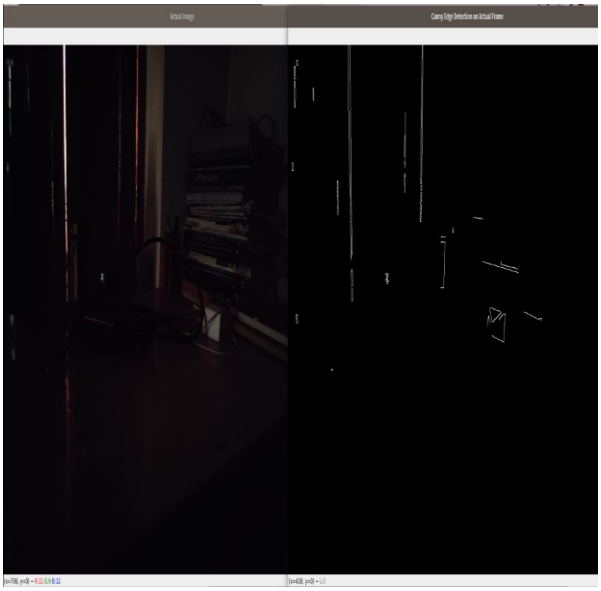

Fig. 8. Canny Edge detection in image 2

The performance of Canny Edge detection on the original image is measured using the area under the ROC curve (AUC), which is a standard metric for assessing the quality of detection in Fig. 9.

.

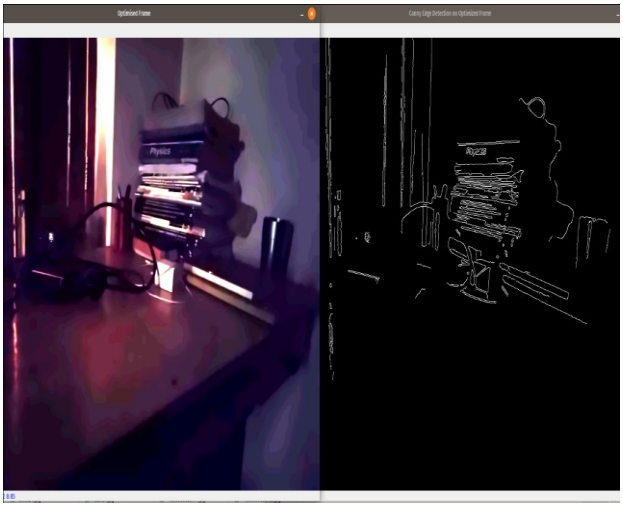

Fig. 9. Canny Edge detection in image 3

The performance of Canny Edges on an optimized image is measured and compared to the performance of other edge detectors.

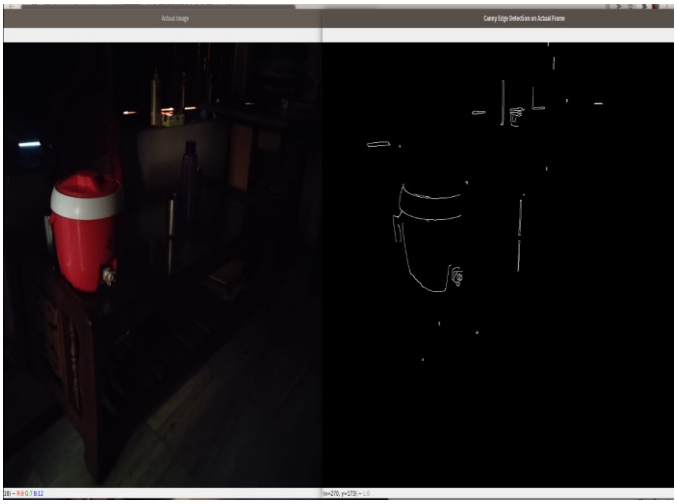

Fig. 10. Canny Edge detection in image 4

The performance of Canny Edge detection on the original image was assessed by comparing the results of the detection with those of the original image itself in Fig. 10.

Authorized licensed use limited to: VIT University- Chennai Campus. Downloaded on February 25,2024 at 15:58:31 UTC from IEEE Xplore. Restrictions apply.

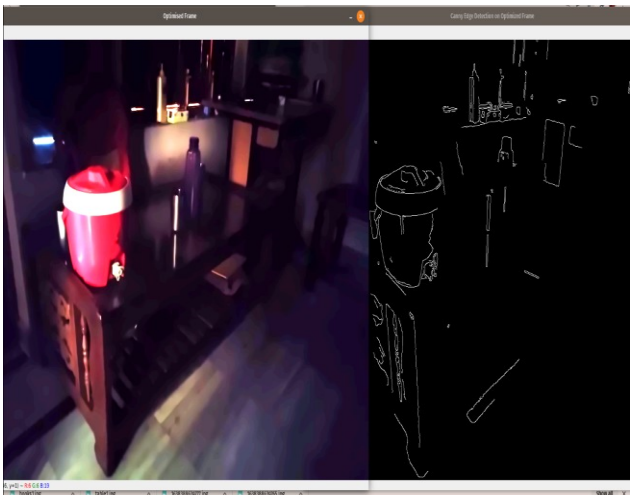

Fig. 11. Canny Edge detection in image 5

Canny Edge detection is a phase of image processing that detects edges in an image. In computer vision, edge detection is a fundamental problem that is widely used in a variety of applications in Fig. 11.

The overall performance of the Canny Edge side detection algorithm has been significantly improved by pre-processing the image before applying the algorithm. Furthermore, an integration analysis for the strategy superior to various other function recognition formulas that detect features based on the power of pixels in an image and will be used in cellular robots can be performed. Slam optimisation of the robot can be improved cheaper because this is really a algorithm and no extra materials are required, and with this mobile robots like robot vacuum cleaners as well as other drones, quadcopters, etc as any robot designed to use slam for navigation may use this algorithm for better navigation, and in the future these slam models are additionally integrated alongside other much better algorithms.

IV. CONCLUSION

The Canny Edge side detection algorithm has been significantly improved by pre-processing the image before applying the algorithm. Furthermore, an integration analysis for the strategy superior to various other function recognition formulas that detect features based on the power of pixels in an image can be performed. Slam optimization of the robot can be improved cheaper because this is really a algorithm and no extra materials are required, and with this the mobile robots like robot-vacuum cleaners as well as other drones, quadcopters, etc as any robots designed to use slam for navigation may use this algorithm for better navigation, and in the future these models are additionally integrated alongside other much better algorithms.

ACKNOWLEDGMENTS

All authors discussed the results and contributed to the final manuscript. The authors would like to thank Chennai Institute of Technology, Amrita Vishwa Vidyaapeetham, Panimalar Engineering College, Chitkara University and Saveetha Institute of Medical and Technical Sciences for providing us with various resources and unconditional support for carrying out this research work.

References

[1] Xu Qian, Chakrabarti Chaitali and Lina J. Karam, "A Distributed Canny Edge Detector and Its Implementation On FPGA", School of Electrical Computer and Energy Engineering Arizona State University IEEE, pp. 500-505, 2011.

[2] G. K. J, G. S, S. Rajendran, J. S. Vimali, J. Jabez and S. Srininvasulu, "Identification of Cyber Threats and Parsing of Data," 2021 5th International Conference on Trends in Electronics and Informatics (ICOEI), 2021, pp. 556-564, doi: 10.1109/ICOEI51242.2021.9452925.

[3] Gao Jie and Liu Ning, "An improved adaptive threshold canny edge detection algorithm", IEEE International Conference on Computer Science and Electronics Engineering, pp. 164-168, 2012.

[4] Altera DE2-115 Manual and Altera Youtube channel, [online] Available: www.altera.com.

[5] B. Muralikrishna, K. Gnana Deepika, B. Raghu Kanth and V. G. Swaroop Vemana, "Image Processing using IP Core Generator through FPGA", International Journal of Computer Applications, vol. 46, no. 23, pp. 48-52, May 2012.

[6] R. Surendran and T. Tamilvizhi, "Friendly Online Technology Development Cloud Service for Bahraini Students based on E-Advisor," 2018 International Conference on Innovation and Intelligence for Informatics, Computing, and Technologies (3ICT), 2018, pp. 1-7, doi: 10.1109/3ICT.2018.8855783

[7] T. Nakano, T. Morie and A. Iwata, "A Face/Object Recognition System Using FPGA Implementation of Coarse Region Segmentation", SICE Annual Conference 2003, pp. 1418-1423, Aug. 4–6, 2003.

[8] K. Heikkinen and P. Vuorimaa, "Computation of Two Texture Features in Hardware", Proceedings of the 10 th International Conference on Image analysis and processing , pp. 125-129, 1999.

[9] S. Sivanantham, N. Nitin Paul and R. Suraj Iyer, "Object Tracking Algorithm Implementation for Security Applications", Far East Journal of Electronics and Communications, vol. 16, no. 1, 2016.

[10] Xu Qian, S. Varadarajan, C. Chakrabarti and L.J. Karam, "A Distributed Canny Edge Detector: Algorithm and FPGA Implementation", Image Processing IEEE Transactions on, vol. 23, no. 7, pp. 2944-2960, July 2014.

[11] Tamilvizhi.T, Surendran.R, Bommi. R. M, "Radio Frequency Identification (RFID) based ubiquitous health care data handling", IOP Conference Series: Materials Science and Engineering, 2020, 994(1), 012021.

[12] P. Bao, L. Zhang and X. Wu, "Canny edge detection enhancement by scale multiplication", IEEE Trans. Pattern Anal. Mach. Intell., vol. 27, no. 9, pp. 1485-1490, Sep. 2005.

[13] Y. Luo and R. Duraiswami, "Canny edge detection on NVIDIA CUDA", Proc. IEEE CVPRW, pp. 1-8, Jun. 2008.

Authorized licensed use limited to: VIT University- Chennai Campus. Downloaded on February 25,2024 at 15:58:31 UTC from IEEE Xplore. Restrictions apply.