Al-Nahrain Journal of Science
ANJS, Vol. 28(2), June, 2025, pp. 178-188

Enhancing Edge Detection with an Optimized Canny Algorithm, Hough Transform, and Machine Learning Integration

Russel K. Lafta[1,*], Zainab N. Sultani[1]

[1] Department of Computer Science, College of Science, Al-Nahrain University, Baghdad, Iraq

Article's Information

Received: 15.12.2024
Accepted: 27.03.2025
Published: 15.06.2025

Keywords:
Geometric Shape Recognition
Noise Reduction
Threshold Optimization
Hough Transform
Machine Learning

Abstract

This paper addresses the challenge of accurate edge detection in complex and noisy image environments by proposing a hybrid approach that integrates classical image processing with machine learning (ML). First, an optimized Canny edge detection algorithm is applied, where adaptive thresholding and a bilateral filter are used to enhance edge precision by reducing noise and preserving edge continuity. Next, the Hough Transform is employed to detect geometric structures, particularly lines, ensuring robustness against distortions and irregularities. To further refine the detected edges, XGBoost is trained on extracted edge features, learning patterns to differentiate between true and false edges. The hybrid approach leverages the strengths of traditional edge detection while enhancing adaptability through ML. Experimental results on 50 images from BSD dataset show that the proposed hybrid method surpasses standalone techniques, achieving an accuracy of 88.47%, a precision of 29.79%, a recall of 38.55%, and an AUC of 0.80 using XGBoost. This combination of classical image processing methods and ML demonstrates significant improvements in edge extraction performance, making it a reliable solution for edge detection in diverse and noisy image datasets.

http://doi.org/10.22401/ANJS.28.2.17
*Corresponding author: russel.kareem11@gmail.com

1. Introduction

Images are consistently a critical resource of information for discovering and perceiving the world and illustrating the world. In practical utilization of image processing, a key essential feature of an image is its edges, whose details are often used in higher-level image processing techniques. As a result, edge detection and extraction techniques have become the principle of many image processing-related technologies and have become a significant research area in the existing digital image processing domain[1]. Classical edge extraction method is simple and appropriate to utilize, and has been commonly implemented in different areas, however, it also has significant flaws in practical implementation. Widely applied edge extraction approaches usually sole focus on the image alone, but dismiss the contextual details of depth image, do not consider the visual characteristics of each layer, and in the implementation of the algorithm, the parameters need to be set constantly, so that the identification results are dissatisfying [2, 3]. Also, a significant difficulty of automated digital image analysis is shape recognition, which is an important part of object detection. Unfortunately, direct searching and finding of a class of even simple geometric patterns like straight lines, circles, and ellipses in an image are computationally heavy. Additionally, obscured curves or missing and spurious data pixels resulting from the noise emanating from either the imperfection of image data or edge detector makes this task non-trivial [4]. To tackle this issue, almost half of century ago, Hough [5] while attempting to identify and plot the tracks of subatomic particles in bubble chamber photographs, devised the Hough Transform (HT), which is a novel approach that convert such global curve detection problem into an efficient peak recognition problem in parameter space. The studied on HT-based recognition affirms to its major achievement in pattern recognition, and is too extensive to be summarized in a few lines;

here we refer for a survey [4] and mention only some works that we consider essential in its growth: the original explanation for hough in [5] for straight line detection, the expansion to circles and ellipses in [6], and finding in feature images without analytical illustration in 2D [7]. Lately, HT has been broadened in [8] to classes of (planar) curves whose algebraic forms were recognized and to low-degree polynomial curves, see [9]. Recent developments have combined the HT with contemporary machine learning methods to boost its performance. For example, combining HT with K-Means clustering has shown improved shape detection accuracy in tasks like iris recognition by improving shape detection accuracy[10]. Other studies, such as those incorporating HT with convolutional neural networks (CNNs) and support vector machines (SVMs)[11], have similarly improved object detection and edge recognition in complex environments. Most of these studies suffer from challenges such as sensitivity to noise and parameter tuning, which affect the accuracy of segmentation and recognition. Edge detection techniques, like the Canny edge detector, remain widely used because of their ability to accurately detect edges with high precision. However, like the HT, the traditional Canny method has limitations, such as sensitivity to noise and the challenges associated with setting the optimal threshold, leading to the emergence of an approach that combines these techniques with machine learning (ML) methods to improve accuracy and stability. The researcher in [12] presented an approach to combine traditional edge detection methods with machine learning algorithms such as KNN, MLP, and SVM, while optimizing the illumination and contrast before extraction. This combination aims to enhance detection accuracy and reduce noise. However, the method suffers from some limitations, such as increased computational complexity, the need for intensive preprocessing, and the need to tune parameters to achieve optimal performance across diverse datasets. Flores-Vidal et al. presented a pixel-level classification algorithm to identify edges, where a feature vector is generated for each pixel instead of using the entire image, applying machine learning techniques such as Random Forest, Neural Network, and Logistic Regression [13]. However, the method relies on the Sobel operator, which is less accurate compared to more advanced algorithms such as Canny. To tackle this problems, recent works have introduced optimization strategies, including adaptive thresholding techniques and noise-resilient preprocessing, enhancing the continuity of the detected edges, as previously discussed in [14]. In this paper, we used the Hough Transform, a robust and efficient technique widely recognized for detecting lines. To improve the edge extraction process before applying the Hough Transform, a refined and optimized version of the Canny algorithm was utilized [14], specifically hyperparameters fine-tuning to enhance precision and minimize noise. Subsequently, machine learning algorithms, including boosting techniques like XGBoost, are integrated to further refine the detection process by leveraging their ability to improve classification performance and handle complex data distributions. The combination of these methods optimized Canny edge detection, Hough Transform, and advanced machine learning techniques results in a more accurate and efficient approach to edge detection. The remainder of this paper is orderly as follows: Section 2 material and methods describes the tools, data, and procedures used in the research. Section 3 theory, Section 4 present result and discussion. Finally, Section 5 concludes the paper, summarizing the contributions and outlining potential future work.

2. Materials and Methods

The first 50 images from the Berkeley Segmentation Dataset (BSD)[15] were used in this study. MATLAB 2024a was employed for image processing on a laptop with 16 GB of RAM. The Canny edge detection method was optimized using the Flower Pollination Algorithm (FPA) for threshold tuning, followed by the application of the Hough Transform for and the results were evaluated using Random Forest (RF), Neural Network (NN) and XGBoost. All images were resized to dimensions of 481x321 or 321x481 pixels during preprocessing. To evaluate performance, k-fold cross-validation (k=5) was conducted, and the results were compared against ground-truth images created through human annotation and dilation. The goal of this study is to improve the accuracy and robustness of edge extraction. A structured approach was implemented in this study to enhance edge detection and shape recognition as shown in figure1. The following sub-sections provide a comprehensive explanation of each method.

2.1. The Optimized Canny method

Classical edge detection techniques, such as the canny edge detector, rely on fixed parameters that can result in limited effectiveness under different conditions. As discussed in [14], the bilateral filter was employed [16] alongside the Flower Pollination Algorithm (FPA) [17]to improve the Canny edge

detection process. Unlike the traditional Gaussian filter, the bilateral filter was used to preserve edge sharpness while effectively reducing noise, ensuring that essential edge details are maintained during preprocessing. By taking into account both spatial proximity and pixel intensity differences, the bilateral filter prevents the blurring of crucial edges, a common limitation of Gaussian filtering. Following this, FPA a nature inspired optimization algorithm was applied to optimize the Canny edge detection thresholds. FPA iteratively adjusts the low and high threshold parameters by exploring various solutions through global and local pollination. Represented by the following equations:

- **Global Pollination**:

$$\chi_i^{t+1} = \chi_i^{(t)} + \epsilon \times \left(\chi_g^{(t)} - \chi_i^{(t)}\right) \quad \dots(1)$$

where: $\chi_i^{(t)}$ is the current position (flower), $\chi_g^{(t)}$is the current global best solution, and ϵ is the step size generated by Gaussian distribution.

- **Local Pollination**:

$$\chi_i^{t+1} = \chi_i^{(t)} + \epsilon \times \left(\chi_j^{(t)} - \chi_i^{(t)}\right) \quad \dots(2)$$

where $\chi_j^{(t)}$ is a randomly selected neighbor from the population.

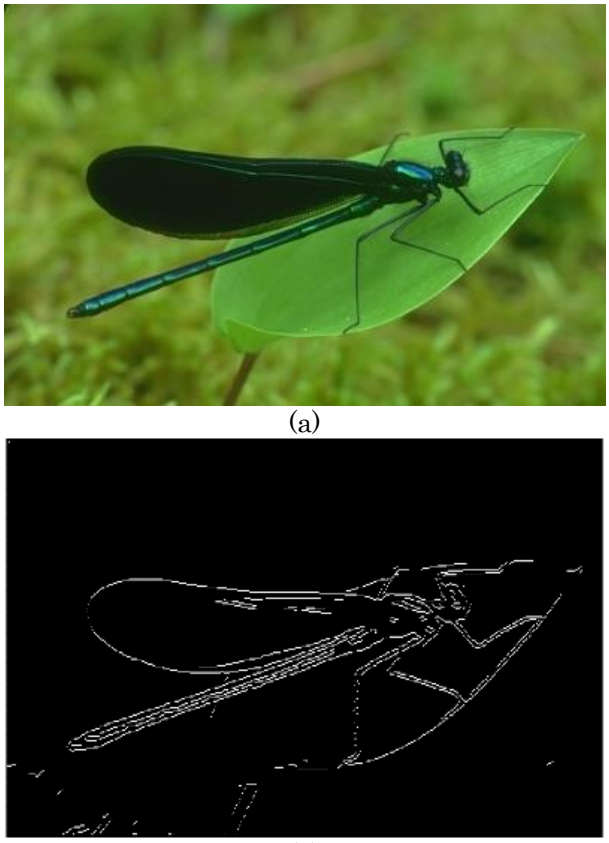
The objective function used in the FPA evaluates the quality of the edge map produced by the Canny edge detection method for a given set of thresholds. It is defined as:

$$\text{obj(i)} = -\sum\left(\text{gradientMagnitude}\left(\text{suppressedImage} > \text{thresholds(i, 2)}\right)\right) \quad \dots(3)$$

The objective function minimizes the gradient magnitudes of strong edges in the suppressed image, ensuring optimal thresholds. This integration of the bilateral filter and FPA, informed by prior studies as noted in [14], significantly improves edge detection precision and robustness. Figure 2 displays the edge maps obtained using the optimized Canny algorithm.

2.2. Shape Recognition Using Hough Transform

The Hough Transform (HT), presented by Paul Hough in 1962 [5], is a widely used technique for identifying geometric shapes in images by transforming points in an image space into a parameter representation. A key challenge in image processing is identifying straight lines or shapes in digitized images, particularly when dealing with incomplete, broken, or noisy edges. Traditional edge detectors often fail to connect disjointed edges or handle cluttered environments effectively, leading to inaccuracies in geometric shape recognition.[6], HT overcomes this limitation by accumulating evidence for a line in a parameter space defined by distance from the origin and angle of inclination. This transformation utilizes accumulator cells, where each edge point votes for potential parameters that represent lines passing through it. Peaks in the accumulator space correspond to the most likely lines in the image. Similarly, the method extends to detect circles and other parametric shapes, addressing challenges in cluttered environments where traditional methods fail[18].



(a)

(b)

Figure 2. (a) Original Image, (b) Image After Applying Optimized Canny.

In the proposed work, we employed the Hough Transform to find lines in edge-detected images processed using an optimized Canny edge detection algorithm. The Hough Transform is based on the equation for a line in polar coordinates:

$$\rho = x\cos(\theta) + y\sin(\theta) \quad \dots(4)$$

where ρ is the distance from the origin, θ is the angle of the line, and (x, y) are the coordinates of a point on the line. The accumulator cells were carefully configured, with parameters such as distance resolution (ρ) set to 1 pixel to provide a fine enough grid to accurately detect lines and angular

resolution (θ) set to 45 degrees (with θ ranging from -90 to 89 degrees) to captures a broad range of line orientations without unnecessarily increasing the computational complexity of the Hough Transform.

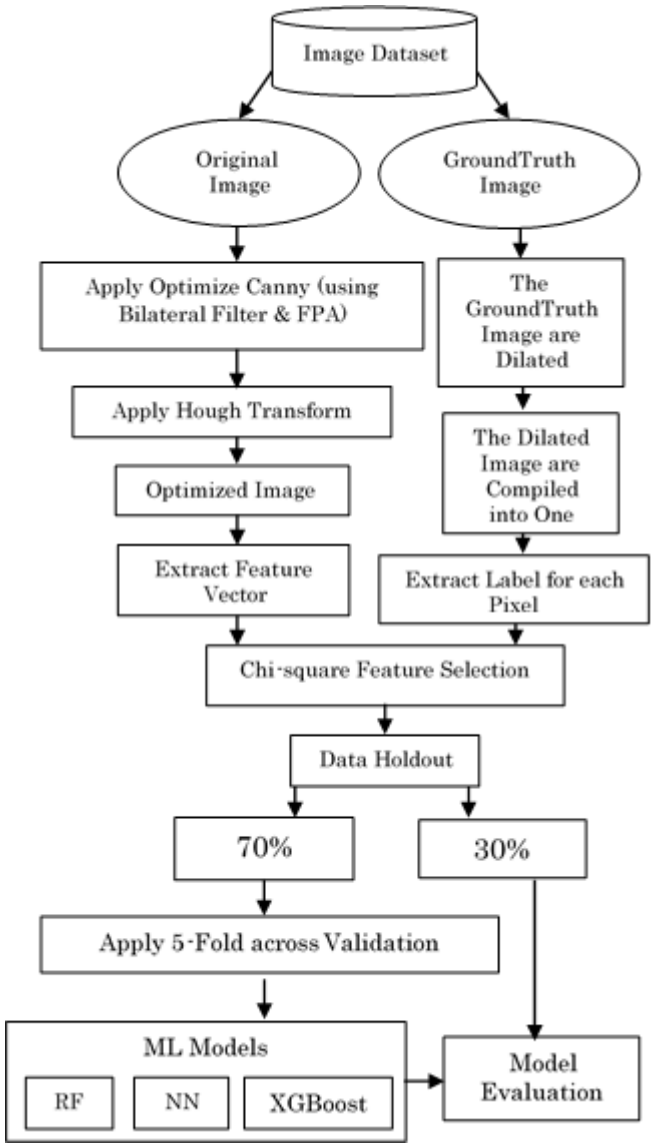


Figure 1. The Block diagram of the proposed work.

To enhance the line detection process, we experimented with different maximum vote (maxvote) values, which allow us to refine the detection by filtering out weaker lines while retaining the most significant ones. Four different maxvote values were tested: 5, 7, 9, and 12. These values are defined as a percentage of the maximum value in the accumulator matrix and help eliminate less-reliable lines. The variation in maxvote significantly impacts the line detection accuracy and helps balance sensitivity and computational efficiency. As shown in Figure 3, changing the maxvote value leads to noticeable differences in the number and quality of the detected lines. The figure demonstrates the effect of maxvote values of 5, 7, 9, and 12 on line detection. The figure highlights how increasing the maxvote value reduces false positives, resulting in fewer, but more accurate, line detections. A maxvote of 7 was found to provide the best balance between detecting relevant lines and minimizing computational load.

Additionally, a threshold of 50% of the maximum value in the accumulator matrix was used to identify significant lines in the parameter space. To further refine the shape detection process, we integrated boosting techniques like XGBoost, which helps improve performance by learning from data and enhancing the accuracy of shape recognition, especially in complex and noisy images. To illustrate the impact of the HT on edge-detected images, we present an image before and after applying the Hough Transform. Figure 4 (b) shows the edge-detected image before the Hough Transform, where edges are identified using the optimized Canny edge detection method. Figure 4 (c) displays the results after applying the Hough Transform, highlighting the detected lines and geometric shapes. The transformation clearly demonstrates the Hough Transform's ability to connect broken edges and accurately detect lines in noisy environments. Experimental results shown the effectiveness of this hybrid method in addressing the challenges of incomplete edge data and cluttered environments, with significant improvements in edge and curve detection accuracy.

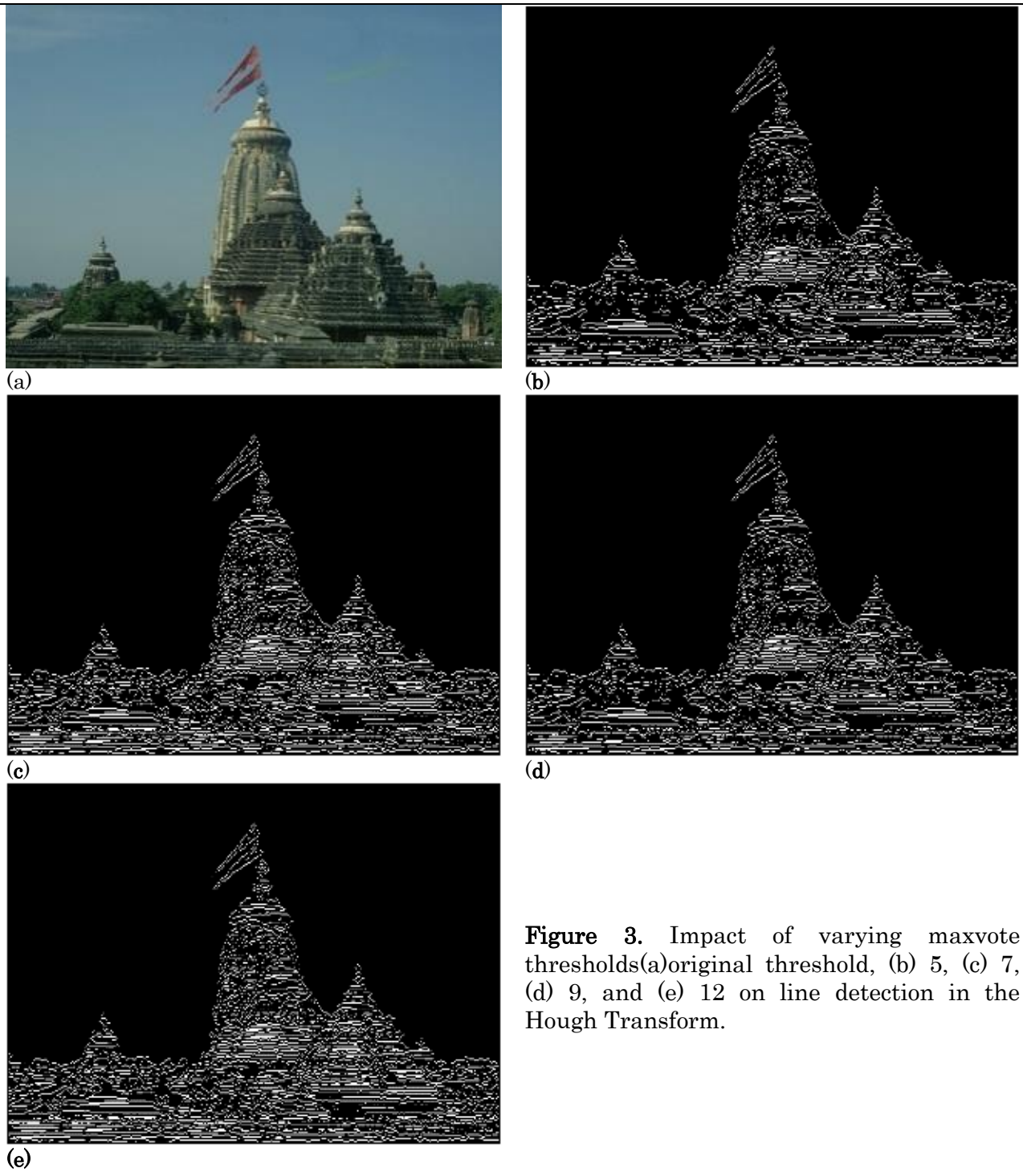


Figure 3. Impact of varying maxvote thresholds(a)original threshold, (b) 5, (c) 7, (d) 9, and (e) 12 on line detection in the Hough Transform.

2.3. Feature Extraction and Label

We utilized the Berkeley Segmentation Dataset (BSD)[15], particularly the first 50 images. The images in the BSD include ground truth segmentations, which are used for measurable assessment of edge detection performance. This dataset is commonly recognized for benchmarking segmentation and edge detection algorithms. In this study, the feature dataset was built following the method discussed in [13] , using multiple human-labeled references to generate a reliable ground truth for supervised classification. Each pixel was marked as an edge if drawn by at least h humans, with labels expanded through mathematical morphology via dilation to account for human inaccuracy.

The final ground-truth image was created by gathering these dilated references. Additionally, predictor variables were extracted from the image after applying the optimized Canny edge detection method with the Hough Transform. This approach successfully captures edge information while minimizing personal biases from individual references. The collected variables are organized into three groups: pixel predictors (such as gray channel intensity, edge angle components, and pixel positions), neighborhood predictors (including values and statistical measures from surrounding pixels), and segment predictors (such as segment length, intensity, and positional relationships). These features provide a complete illustration of the image for classification tasks. Table1 as shown above provides an organized view of these predictor variables.

Table 1. The constructed image feature and their description

Category	Feature	Description
Pixel Predictor (V1 to V6)	V1	Intensity of the gray channel, representing the pixel's "edginess."
	V2	Potential edge angle (θ), broken into components sin(θ) and cos(θ).
	V3, V4	Horizontal (i) and vertical (j) positions of the pixel.
	V5, V6	Minimum distance from the pixel to the horizontal and vertical boundaries.
Neighbors Predictor (V7 to V29)	V7 to V14	Values of the 3×3 neighborhood pixels.
	V15 to V17	Maximum, minimum, and average intensity of the 8 neighboring pixels.
	V18 to V20	Maximum, minimum, and average intensity of the 3 neighboring pixels in the same row.
	V21 to V23	Maximum, minimum, and average intensity of the 3 neighboring pixels in the same column.
	V24 to V29	Maximum, minimum, and average intensity of the 3 neighboring pixels along both diagonals.
Segment Predictor (V30 to V31)	V30	Length of the segment to which the pixel is a member of.
	V31	Mean intensity of the segment.
	V32	Maximum edginess within the segment.
	V33	Standard deviation of the segment intensity.
	V34	Rule of thirds position (central placement of the segment and its Euclidean distance to rule of thirds intersections).
	V35	Area of the minimum rectangle enclosing the segment.
	V36	Belonging of the pixel to a segment.

2.4. Machine Learning Integration

Machine learning (ML) algorithms were applied in the edge identification process. In this study, XGBoost, Neural Network (NN), and Random Forest (RF) classifiers were used following the Hough Transform (HT) and the optimized Canny edge detection algorithm. The classifiers were trained using features extracted from the edge-detected images. To ensure the relevance of the selected features and enhance classification performance, the Chi-Square method was applied, effectively reducing dimensionality by focusing on the most informative features. This statistical technique evaluates the independence of categorical variables and identifies those most relevant to the target variable. The Chi-Square statistic is calculated as:

$$\chi^2 = \sum \frac{(O_i - E_i)^2}{E_i} \quad \ldots(5)$$

where Oi represents the observed frequency, and Ei is the expected frequency under the null hypothesis that the feature and the class are independent. The best achieved optimal performance found using 20 key variables: {v1, v2, v3, v5, v7, v8, v9, v15, v17, v18, v20, v21, v22, v23, v24, v25, v26, v27, v28, v29}. XGBoost (Extreme Gradient Boosting) which is an effective machine learning algorithm built on gradient boosting, designed to optimize both speed and performance. For the XGBoost model, key hyperparameters were fine-tuned as follows: the

learning rate was set to 0.1, the maximum tree depth was configured at 6, and the number of estimators was set to 200. For the Neural Network (NN), the model was structured with a single hidden layer comprising 100 neurons. The training process utilized Gradient Descent with Backpropagation, incorporating forward and backward propagation with a sigmoid activation function for non-linearity and weight updates. The maximum number of iterations for training was set to 200. In the case of the Random Forest (RF) model, the number of trees was configured to 100. The dataset was partitioned into training and test sets, with 70% of the data used for training and 30% reserved for testing. Additionally, k-fold cross-validation with k=5 was applied to the training set to further validate the model's performance. The use of machine learning not only enhances edge detection accuracy but also adapts to various image characteristics, addressing the limitations of classical edge detection methods. The proposed work can be summarized in the algorithm (1.1).

Algorithm 1.1: Optimized Canny Edge Detection with Hough Transform

Input: RGB Image **Output:** Detected Edges
Step 1: Apply Optimized Canny Edge Detection -Smooth the image using Bilateral filtering. -Compute gradient magnitude and direction. -Apply non-maximal suppression. -Optimize high and low threshold values using the Flower Pollination Algorithm (FPA). -Track Edge by hysteresis. **Step 2: Detect lines using Hough Transform** **Step 3: Extract feature vectors (v1-v36) from detected edges** **Step 4: Apply Chi-square for feature selection** **Step 5: Classify detected features using a machine learning models** **Step 6: Display final results (Edge detected image)**

3. Results and Discussion

The effectiveness outputs from the three models: Neural Networks (NN), Random Forest (RF), and XGBoost was evaluated based on multiple metrics, as summarized in Table 2. NN achieved an accuracy of 82.72%, with a low precision of 21.61% but a relatively higher recall of 46.84%, indicating its ability to retrieve positive samples effectively but with a high rate of false positives. The model's F1-score of 29.57% and AUC of 0.76 suggest moderate overall performance. In contrast, RF demonstrated the highest accuracy at 93.04% and a significantly high precision of 87.17%, highlighting its ability to predict positive samples with high confidence. However, its recall was only 9.33%, indicating a failure to retrieve most positive samples, leading to an imbalanced F1-score of 16.85%. XGBoost, applied as an ensemble model, showed a balanced performance, achieving an accuracy of 88.47% with a precision of 29.79% and a recall of 38.55%. While its precision and recall were not individually superior to RF and NN, respectively, the F1-score of 33.61% demonstrates improved balance between these metrics. Additionally, XGBoost attained an AUC of 0.80, indicating strong generalization and robust classification performance across different classes. These results highlight the trade-offs between individual models and the advantage of ensemble methods in balancing key performance metrics. Figure 4 illustrates an example of an image after edge detection using these three classifiers. A comparative analysis with the work in [14], the Random Forest achieved an AUC of 0.81, similar to this work, with slightly lower accuracy (93.17% vs. 94.04%). However, it showed significantly higher precision (95.86%) but very low recall (10.13%), indicating a strong focus on positive predictions at the expense of capturing true positives. Their Logistic Regression model, with an AUC of 0.75 and accuracy of 85.94%, performed similarly to Neural Network in balancing precision (21.94%) and recall (31.82%), but it falls short of the F-measure achieved by XGBoost. These comparisons emphasize the trade-offs between models. In comparison with the results in [13], which used Sobel edge detection followed by ML models, the performance is notably lower. The Neural Network in [13] achieved an AUC of 0.55, significantly lower than our Neural Network (0.76), with higher F1-Score (40.03%) than our approach. Similarly, Random Forest and Logistic Regression yielded even lower results with AUC values of 0.54 and 0.49, respectively. This suggests that although the compared method performs well at a specific threshold, its lower AUC implies

reduced stability across varying decision thresholds. These results suggest that the combination of Sobel edge detection with ML models fails to match the effectiveness of optimized Canny edge detection followed by ML, as employed in this work, in terms of edge feature extraction and overall classification performance. These comparisons underline the advantages of integrating optimized Canny edge detection with machine learning (especially XGBoost) over traditional edge detection methods like Sobel, leading to more robust and balanced edge detection, with improved precision and recall.

Table 2. The performance of the proposed models compared to the related studies.

ML Algorithms	AUC	Accuracy	Precision	Recall	F-measure
Random Forest	0.81	94.04%	87.17%	9.33%	16.85%
Neural Network	0.76	82.72%	21.61%	46.84%	29.57%
XGBoost	0.80	88.47%	29.79%	38.55%	33.61%
Random Forest [14]	0.81	93.17%	95.86%	10.13%	18.32%
Logistic Regression [14]	0.75	85.94%	21.94%	31.82%	25.97%
Neural Network[13]	0.55	-	-	-	40.03%
Random Forest[13]	0.54	-	-	-	40.0%
Logistic Regression[13]	0.49	-	-	-	38.1%

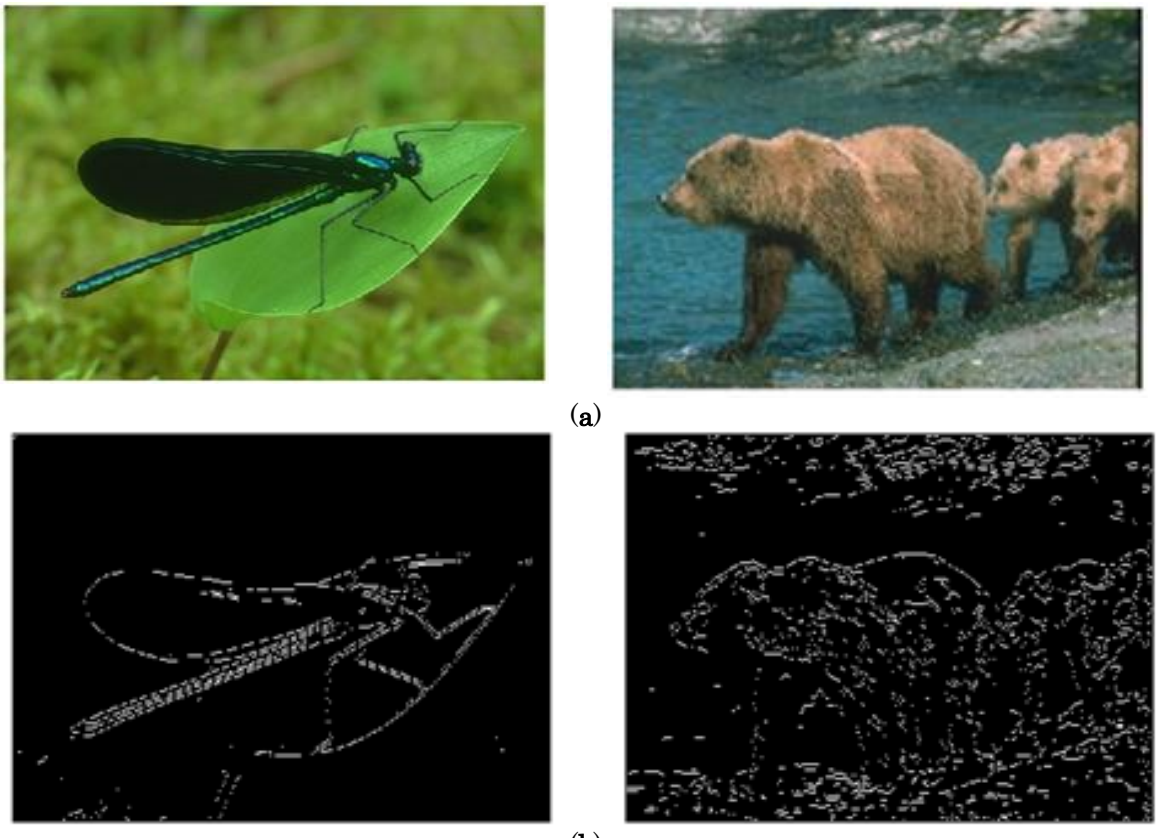


Figure 4. (a) Original image, (b) Edge detected image used optimized Canny.

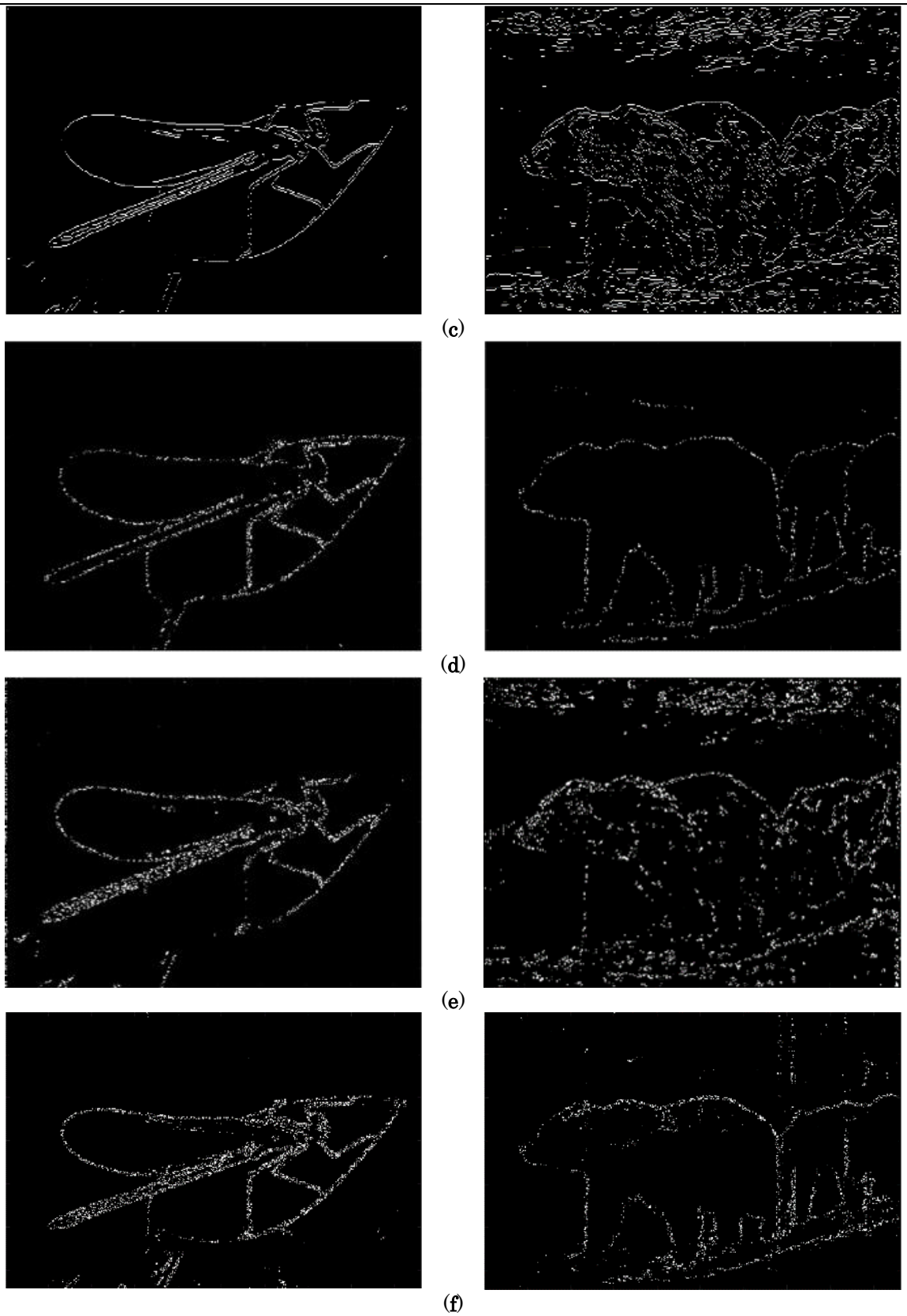


Figure 4. (continued) (c) Edge detected image after apply Hough Transform (d) Edge detected by RF model (e) Edge detected by NN model and (f)Edge detected by XGBoost.

4. Conclusions

In conclusion, this study has demonstrated a significant enhancement in edge extraction by integrating the Hough Transform (HT) with machine learning (ML) and an optimized Canny algorithm. However, several challenges were encountered throughout the process. One of the main difficulties was managing noise in image data, which was addressed through the optimization of the Canny edge detection technique. The integration of machine learning (XGBoost) also posed challenges in selecting the most relevant features and fine-tuning the model for accurate edge classification. Despite these challenges, experimental results show that the proposed method, using XGBoost, achieves 88.47% accuracy, 29.79% precision, 38.55% recall, and an AUC of 0.80, demonstrating its effectiveness in detecting complex geometric patterns and curves. The integration of ML helped fine-tune parameter selection, while the optimized Canny algorithm effectively reduced noise without losing essential details. These findings highlight the advantages of combining traditional and ML approaches for more precise edge detection.

Outlook toward the future, different route can be explored. One of the paths is related to the potential of deep learning models to further enhance the edge detection process. Another future work could investigate the incorporation of 3D edge detection and curve recognition, extending the current 2D framework to more complex three-dimensional datasets.

Conflicts of Interest: The authors declare no conflict of interest.

Founding Statement: This research was self-funded.

References

[1] Sun, R.; Lei, T.; Chen, Q.; Wang, Z.; Du, X.; Zhao, W.; and Nandi, A. K.; "Survey of image edge detection". Front. Signal Process, 2: 826967, 2022.

[2] Gaurav, K.; and Ghanekar, U.; "Image steganography based on Canny edge detection, dilation operator and hybrid coding". J. Inf. Secure. Appl, 41: 41-51, 2018.

[3] Xie, X.; Ge, S.; Xie, M.; Hu, F.; and Jiang, N.; "An improved industrial sub-pixel edge detection algorithm based on coarse and precise location". J. Ambient Intell. Humaniz. Comput, 11: 2061-2070, 2020.

[4] Mukhopadhyay, P.; and Chaudhuri, B. B.; "A survey of Hough Transform". Pattern Recognit., 48(3): 993-1010, 2015.

[5] Hough, P.V. "Method and Means for Recognizing Complex Patterns". U.S. Patent 3,069,654, 1962.

[6] Duda, R. O.; and Hart, P. E.; "Use of the Hough transformation to detect lines and curves in pictures". Commun. ACM, 15(1): 11-15, 1972.

[7] Ballard, D. H.; "Generalizing the Hough transform to detect arbitrary shapes". Pattern Recognit., 13(2): 111-122, 1981.

[8] Beltrametti, M. C.; and Robbiano, L.; "An algebraic approach to Hough transforms". J. Algebra, 371: 669-681, 2012.

[9] Conti, C.; Romani, L.; and Schenone, D.; "Semi-automatic spline fitting of planar curvilinear profiles in digital images using the Hough transform". Pattern Recognit., 74: 64-76, 2018.

[10] Babu, G.; and Pinjari, A. K.; "A new design of iris recognition using hough transform with K-means clustering and enhanced faster R-CNN". Cybern. Syst, 55(2): 551-584, 2024.

[11] Bamogo, M.; and Sere, A.; "An application of the Hough transform with convolutional neural networks to detect lines". In: Proceedings of the International Conference on Innovations and Technological Solutions for Sustainable Development (Intersol 2024), Dakar, Senegal, July; Springer, 2024.

[12] Park, K.; Chae, M.; and Cho, J. H.; "Image pre-processing method of machine learning for edge detection with image signal processor enhancement". Micromachines, 12(1): 73, 2021.

[13] Flores-Vidal, P.; Castro, J.; and Gomez, D.; "Postprocessing of Edge Detection Algorithms with Machine Learning Techniques". Math. Probl. Eng, 2022(1): 1-12, 2022.

[14] Russel, K. L.; and Zainab, N. S.; "An Optimized Canny Edge Detection with Traditional Machine Learning for Edge Detection Enhancement". Eng. Technol. (under review).

[15] Martin, D.; Fowlkes, C.; Tal, D.; and Malik, J.; "A Database of Human Segmented Natural Images and Its Application to Evaluating Segmentation Algorithms and Measuring Ecological Statistics". In: Proceedings of the Eighth IEEE International Conference on Computer Vision (ICCV 2001), Vancouver, Canada, 7–14 July; IEEE, 2001.

[16] Tomasi, C.; and Manduchi, R.; "Bilateral Filtering for Gray and Color Images". In: Proceedings of the Sixth International Conference on Computer Vision (ICCV 1998), Bombay, India, January; IEEE, 1998.

[17] Yang, X.-S.; "Flower Pollination Algorithm for Global Optimization". In: Proceedings of the International Conference on Unconventional Computing and Natural Computation, Orléans, France, 3–7 September; Springer, 2012.

[18] Illingworth, J.; and Kittler, J.; "A survey of the Hough transform". Comput. Gr. Image Process., 44(1): 87-116, 1988.