Journal of Artificial Intelligence and Engineering Applications

Website: https://ioinformatic.org/

June 2026. Vol. 5. No. 3; e-ISSN: 2808-4519

Comparative Analysis of Sobel, Prewitt, and Canny Methods in Detecting Object Edges in Betta Fish Images

Alfin Alfarizi[1]*, Cici El Dirrah Syafitri Simanungkalit[2], Fahmi Nur Alimsyah Purba[3], Lailan Sofinah Harahap[4]

[1,2,3,4]*Universitas Islam Negeri Sumatera Utara*
alfin0701232113@uinsu.ac.id[1*], cicieldirah@gmail.com[2], fahmi0701232114@uinsu.ac.id[3], lailansofinah@uinsu.ac.id[4]

Abstract

Edge detection is a crucial stage in digital image processing for recognizing the shape and structure of an object. The application of edge detection to betta fish images presents a unique challenge due to their layered, intricately textured, and often semi-transparent fin morphology. This study aims to analyze and compare the performance of three edge detection algorithms, namely Sobel, Prewitt, and Canny, in extracting shape features from betta fish images. The research methodology involved converting the dataset images into a grayscale format and subsequently implementing the three algorithms using the OpenCV library in the Python programming language. The evaluation was conducted visually by observing the sharpness of the edge lines, object continuity, and the occurrence of noise. The results indicate that the Canny algorithm provides the most optimal performance, as it is capable of detecting the thin edge lines of the fish fins with greater detail and continuity due to its hysteresis thresholding process. Meanwhile, the Sobel and Prewitt methods produced thicker edge lines but were less sensitive to the details of the transparent fins. This study is expected to serve as a reference in selecting the appropriate segmentation method for biological objects with complex morphologies.

Keywords: *Image Processing, Edge Detection, Sobel, Prewitt, Canny, Betta Fish.*

1. Introduction

The advancement of computer vision technology today relies heavily on the quality of feature extraction from digital images. One of the most fundamental preprocessing stages is edge detection [1]. This process aims to identify the boundary lines of an object by detecting drastic changes in pixel intensity. Accurate edge extraction is crucial in determining the success of subsequent stages, such as pattern recognition, segmentation, and object classification [2].

In practice, the performance of edge detection algorithms is greatly influenced by the morphological characteristics of the object being studied. Images of betta fish (*Betta sp.*) present unique visual challenges in image processing [3]. These creatures possess layered fin structures that are intricately textured, exhibit complex color gradations, and are often semi-transparent, causing the boundary between the fins and the water background to become blurred. The inability of an algorithm to capture these structural details can result in the loss of crucial shape feature information [4].

Various edge detection methods have been developed, including first-derivative-based operators such as Sobel and Prewitt, as well as optimization-based operators such as Canny [5]. Although these methods have been widely tested on common objects, comprehensive evaluations of their performance in responding to the visual challenges posed by betta fish morphology remain very limited. Therefore, this study aims to conduct a comparative performance analysis of the Sobel, Prewitt, and Canny algorithms [6]. This comparison will be evaluated based on the visual quality of edge sharpness, noise level, and computational time efficiency, in order to determine the most optimal method for betta fish image segmentation.

The results of this comparative analysis are expected not only to provide a theoretical evaluation, but also to offer practical contributions to the field of computer vision system development, particularly in the extraction of biologically complex morphological objects [7]. Practically, identifying the most accurate and efficient edge detection algorithm can serve as a baseline for designing advanced intelligent systems [8]. Some potential implementations include automated betta fish species classification systems, image-based animal health monitoring, and precision quality grading systems. By understanding the computational advantages and limitations of each method in real-world scenarios, this study is expected to serve as a literature reference for the development of more optimized image processing software in the future.

2. Research Methods

This study employs a quantitative and comparative experimental approach. The research stages are systematically designed, spanning from data collection, image preprocessing, edge detection algorithm implementation, through to the evaluation stage.

2.1. Image Dataset

The data used in this study consists of secondary digital images of betta fish (*Betta sp.*) against a plain background to minimize visual interference. The images are in .jpg format and processed using the Python programming language, supported by the OpenCV library for image processing operations and Matplotlib for data visualization.

2.2. Preprocessing

Prior to the edge detection stage, the original image in the RGB (Red, Green, Blue) or BGR color space is first converted into Grayscale format [9]. This is done because edge detection algorithms (Sobel, Prewitt, and Canny) operate based on single-pixel intensity change calculations, rendering color information unnecessary and reducing the computational workload [10].

3. Edge Detection Algorithm Implementation

At this stage, the grayscale image is processed using three different methods:

1. Sobel Method: Uses a 2D spatial gradient operator (a combination of 3×3 kernel matrices in the horizontal and vertical directions) to find the approximate absolute gradient of the image [11].
2. Prewitt Method: Operates on a similar principle to Sobel, but uses different (constant) kernel matrix weight values, which are considered more sensitive to vertical and horizontal edges [12].
3. Canny Method: A multi-stage method that not only computes the matrix gradient, but also applies a Gaussian filter to reduce noise, Non-Maximum Suppression to thin the edge lines, and Hysteresis Thresholding to ensure edge lines remain unbroken [13].

The edge extraction process in this study is implemented using the OpenCV library. The following are code snippets for each algorithm function:

4. Performance Evaluation

The evaluation is conducted by comparing the output of the three methods based on two main parameters:

1. Visual Evaluation (Qualitative): Observing how well each algorithm represents the original morphology of the betta fish, encompassing the sharpness of fin boundaries, the continuity of edge lines, and the minimal level of noise detected outside the object.
2. Computational Time (Quantitative): Measuring the execution time difference (in seconds) from the moment the edge detection function is called until a new output matrix is produced, in order to assess the computational efficiency of each algorithm [14].

5. Results and Discussion

The testing was conducted using an image of a white betta fish against a solid black background (zero-noise background) that had undergone a color space conversion to grayscale. This image was then processed using three edge detection algorithms — Sobel, Prewitt, and Canny — to evaluate their qualitative and quantitative performance based on real experimental data.

5.1. Qualitative Visual Evaluation

Based on the visualization of the test results in Figure 1, a striking difference in characteristics among the three methods is observed in their representation of the betta fish's morphological structure. The experimental results yield findings that contradict the general assumption, wherein the classical methods actually outperform the optimization-based method.

Fig. 1 : Original Image of Betta Fish

1. Sobel Method Results: The Sobel algorithm demonstrates the most superior visual performance and information richness in this test. This method not only successfully captures the main contour lines along the outer body of the fish, but remarkably manages to extract the fin bone texture details within the interior as well. The clean black background minimizes Sobel's primary weakness against noise, allowing the operator to focus on detecting subtle color gradients in the fins, producing a highly comprehensive visual representation reminiscent of an X-ray scan.
2.

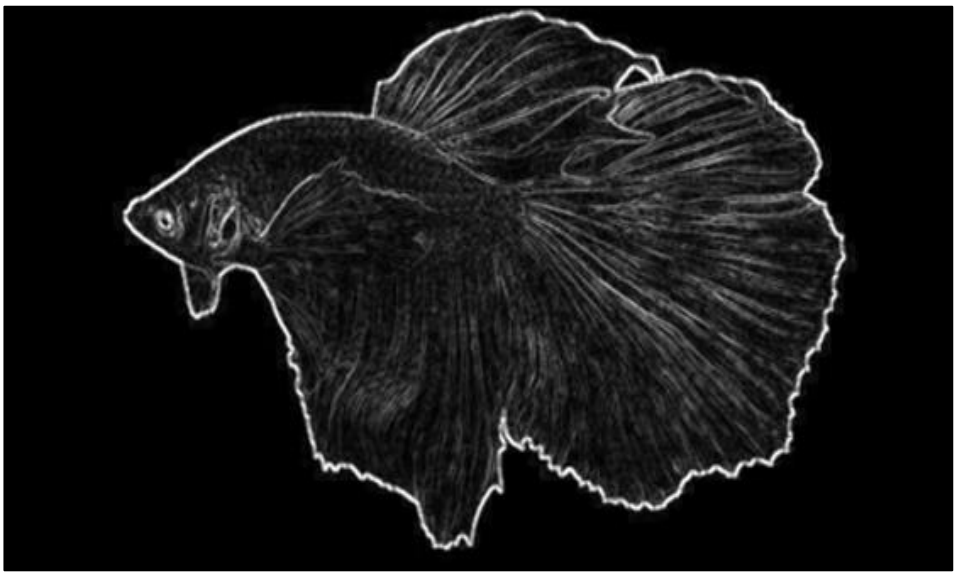

Fig. 2 : Sobel Edge Detection Results

3. Prewitt Method Results: The output of the Prewitt method follows a pattern very similar to that of Sobel, with the internal fin texture successfully captured. However, the resulting visualization exhibits considerably weaker and dimmer pixel intensity. This indicates that the constant weights of the Prewitt operator are less sensitive in responding to subtle color gradient changes in the test image compared to the exponential weights of the Sobel operator.

Fig. 3 : Prewitt Edge Detection Results

4. Canny Method Results: Contrary to theoretical expectations, the Canny algorithm actually experiences significant feature loss. Although Canny successfully produces very thin and sharp outer edge lines (silhouette), the algorithm completely fails to extract the fin bone texture within the interior of the fish's body. This loss of detail is caused by Canny's initial stage the Gaussian Blur which inadvertently blurs and erases the subtle fin texture. Furthermore, the thresholding values applied are too high to capture the color gradations of the inner fin region, causing that area to be classified as non-edge and resulting in an empty black (blank) area.

Fig. 4 : Canny Edge Detection Results

5.2. Computational Time Evaluation

The efficiency of each method is measured based on the computational time when executing the image matrix using the Python programming language. The following are the real processing time recordings from the program terminal:

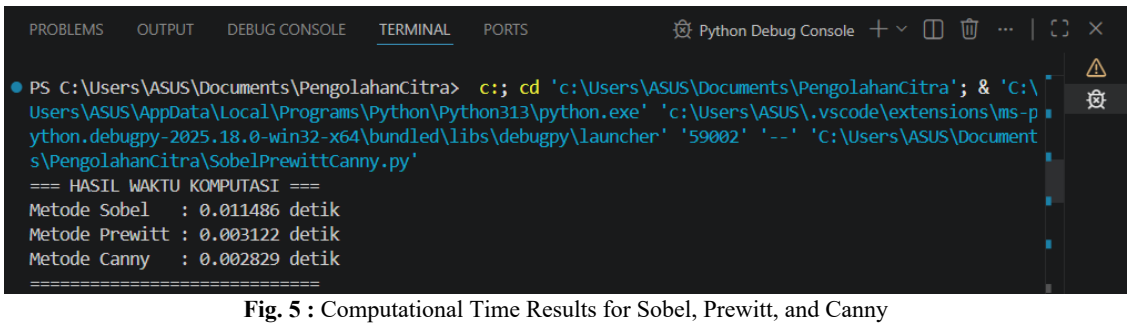

Fig. 5 : Computational Time Results for Sobel, Prewitt, and Canny

The data above reveals highly interesting findings regarding computational time efficiency. In theory, the Canny method is often considered to require more time due to its multi-stage process (noise reduction, Non-Maximum Suppression, and double thresholding). However, in this real-world experiment, the Canny method actually recorded the fastest (most efficient) computational time.

Beyond the image characteristic factor, this computational time efficiency anomaly is heavily influenced by the function call overhead in

the programming script used. In the implementation of the Sobel and Prewitt methods, edge extraction is carried out through a series of separate function stages, including the calculation of the horizontal gradient matrix (X-axis), the vertical gradient matrix (Y-axis), absolute value conversion, and the process of merging both matrices (addWeighted). Each of these separate function calls demands additional memory allocation and execution load on the Python interpreter. In contrast, the Canny method is executed through a single integrated function call (cv2.Canny), where all of its mathematical complexity is processed internally within an optimized library backend, thereby significantly reducing computational time.

Table 1 : Comparison of Sobel, Prewitt, and Canny Methods

No.	Method	Computational Time	Detection Quality
1	Sobel	0.011486 seconds	Very detailed and clear
2	Prewitt	0.003122 seconds	Detailed but dim
3	Canny	0.002829 seconds	Outer silhouette only (loss of detail)

Based on the summary in Table 1 above, a very clear trade-off phenomenon (performance exchange) is observed between computational time and the resulting detection quality. The Sobel algorithm is proven to be absolutely superior in extracting the complete morphological feature details of the fish; however, as a consequence, this method imposes the longest processing time on the system. In contrast, the Canny method offers exceptionally high execution speed, but at the cost of sacrificing the quality of internal object texture extraction.
To illustrate this temporal efficiency disparity more comprehensively, the quantitative computational time data is visualized through a graphical representation. This graph is crucial for highlighting the time complexity overhead that occurs in the execution of the Sobel algorithm compared to its counterparts.

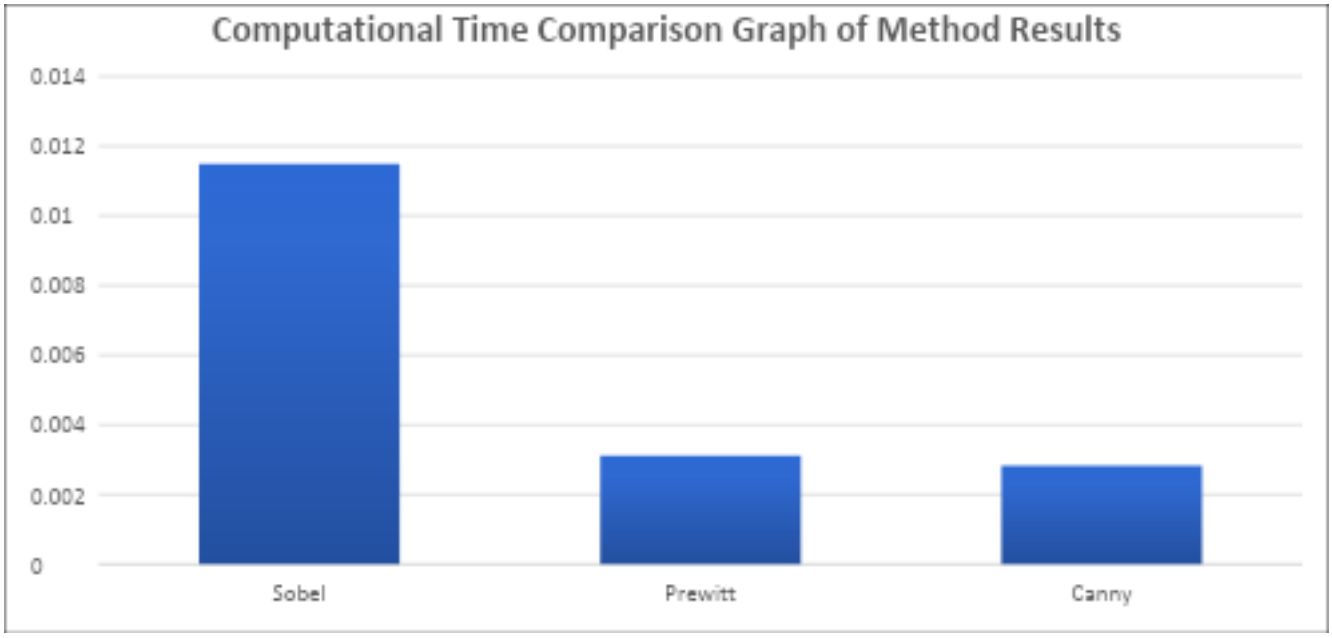

Fig. 6 : Computational Time Comparison Graph of Method Results

Advanced Computational Time Analysis: The speed of the Canny algorithm surpassing Sobel in this test can be deconstructed further through an analysis of backend architecture and image characteristics:

1. Function Call Architecture and C++ Backend Optimization in OpenCV
   The most crucial factor lies in the software architecture bridge. The entire complexity of Canny's five-stage algorithm is encapsulated into the cv2.Canny() function. When called, execution is directly delegated to OpenCV's highly-optimized compiled C/C++ backend module, minimizing inter-process overhead in Python. Meanwhile, manually executing Sobel in Python forces the system to repeatedly read and rewrite the image matrix into memory.
2. Influence of Pixel Distribution Characteristics and Hysteresis Thresholding Efficiency
   The test betta fish image features a very sharp contrast between the white object and the solid black background. This condition provides a computational advantage during the Hysteresis Thresholding stage in the Canny method. Since the color transition from object to background is extremely drastic, the majority of pixels are immediately classified as either strong edge or non-edge. This minimizes the edge tracking iterations for weak edges, drastically cutting down processing time.
3. Conclusion of Time and Quality Analysis (Trade-Off)
   Based on the experimental arguments above, it can be concluded that no single method is superior across all parameters. The choice of algorithm must be tailored to the needs of the system. If the primary goal of the system is to analyze anatomical details or fin texture (such as a fish health identification system), then the Sobel Method is the best choice regardless of its computational load. However, if the system is intended for real-time outer shape detection (shape tracking) such as counting moving fish, then the Canny Method is more recommended due to its superior execution speed.

6. Conclusion

From the results of the comparative performance analysis of the Sobel, Prewitt, and Canny algorithms in detecting object edges in white betta fish images against a dark background, several key conclusions can be drawn as follows:

1. Visual Performance (Extraction Quality): The Sobel method is proven to deliver the most optimal and comprehensive visual feature extraction results. Under zero-noise background conditions, Sobel is capable of extracting the internal fin bone texture details completely and clearly. In contrast, the Prewitt method produces dim edge lines, while the Canny method actually experiences significant feature loss in the interior of the fish's body due to the blurring effect (Gaussian Blur) and threshold values that filter out subtle textures.
2. Computational Time Efficiency: In terms of speed, the experimental data defies the general theoretical assumption. The Canny method turns out to be the fastest and most efficient, recording a time of 0.002829 seconds, owing to OpenCV library backend optimization and the efficiency of Hysteresis Thresholding. On the other hand, the repeated function call executions within the Python interpreter cause the Sobel method to record the longest processing time at 0.011486 seconds.
3. Method Selection Recommendation (Trade-Off): No single method is superior across all parameters. The choice of algorithm must be tailored to the objectives of the system. The Sobel algorithm is highly recommended for developing systems that demand anatomical analysis and texture detail (such as species classification or fish health monitoring). However, for real-time system implementations that only require outer shape silhouette tracking (shape tracking), the Canny algorithm is the more appropriate choice due to its superior computational efficiency.

Based on the limitations identified in this study, several recommendations can be proposed for future research:

1. Performing modifications and dynamic tuning of the threshold parameters (low/high threshold) in the Canny algorithm to test whether Canny is capable of extracting internal object textures as clearly as the Sobel method.
2. Expanding the testing scenarios by using fish image datasets with complex backgrounds or high lighting noise levels to evaluate the robustness of the Sobel algorithm against visual interference.

References

[1] D. Anggraini, P. Hapsari, and W. K. Nofa, “Perbandingan Efektivitas Metode Canny-Robert Untuk Deteksi Tepi Citra Grayscale Terhadap Noise,” *J. Surya Inform.*, vol. 15, no. 1, pp. 20–28, 2025.
[2] L. Khoirani, R. Ariansyah, and S. Supiyandi, “Aplikasi Pengolahan Citra Untuk Peningkatan Deteksi Tepi Melalui Segmentasi Citra,” *Mars J. Tek. Mesin, Ind. Elektro Dan Ilmu Komput.*, vol. 2, no. 3, pp. 196–203, 2024.
[3] M. S. Moelya, “Perbandingan Metode Canny, Sobel, Dan Laplacian Of Gaussian Dalam Mendeteksi Tepi Citra Objek Bergerak,” *J. Sist. Inf. TGD*, vol. 3, no. 4, pp. 450–460, 2024.
[4] A. Herlambang and A. Rahmatulloh, “Peningkatan Akurasi Deteksi Tepi Pada Citra Inversi Menggunakan Sobel-Canny-Prewitt,” *J. Inform. \& Multimed.*, vol. 17, no. 1, pp. 42–49, 2025.
[5] R. S. Firzanah, N. Z. Anugrani, K. Samsidar, and T. A. Ayyubi, “Implementasi Metode Thresholding Untuk Segmentasi Citra Digital Dalam Lingkungan Visual Studio Code,” *MAPLE Mechatronics J. Prof. Entrep.*, vol. 6, no. 1, pp. 1–8, 2024.
[6] M. R. Qisthiano and A. O. Pratiwi, “Deteksi Tepi Pada Citra Objek Benda Menggunakan Algoritma Sobel dan Prewitt dengan Python,” *J. Inform. dan Tek. Elektro Terap.*, vol. 13, no. 2, pp. 45–52, 2025.
[7] P. Widodo and B. Santoso, “Evaluasi Kinerja Operator Prewitt dan Canny untuk Pengolahan Citra Digital Berbasis OpenCV,” *J. Ilm. Teknol. Inf.*, vol. 23, no. 1, pp. 55–62, 2025.
[8] M. P. Sari and A. Nugroho, “Implementasi Teknik Image Enhancement dan Deteksi Tepi Pada Pengenalan Pola Objek,” *J. Teknol. Inf. dan Ilmu Komput.*, vol. 11, no. 3, pp. 405–412, 2024.
[9] D. Pratama and E. Haryanto, “Analisis Komputasional Algoritma Canny dan Prewitt dalam Pemrosesan Matriks Citra Digital,” *J. CoreIT*, vol. 11, no. 2, pp. 150–158, 2025.
[10] T. S. R. Dini and others, “Penerapan Python Dalam Perbandingan Metode Deteksi Tepi (Sobel, Prewitt, Canny) Untuk Analisis Pengenalan Pola Pada Daun,” *J. Publ. Ilmu Komput. dan Multimed.*, vol. 4, no. 2, pp. 95–105, 2025.
[11] T. Hidayat and R. Putra, “Komparasi Akurasi Ekstraksi Fitur Bentuk Pada Objek Biologis Menggunakan Metode Sobel dan Canny,” *J. Komput. Terap.*, vol. 11, no. 1, pp. 22–30, 2025.
[12] K. D. Verado, Y. F. Riti, and N. E. Mongkol, “Perbandingan Algoritma Sobel, Kirsch, Laplacian of Gaussian dan Canny Untuk Deteksi Pada Citra Keretakan Dinding,” *J. Ilm. Komput. dan Inform.*, vol. 14, no. 1, pp. 1–10, 2025.
[13] A. Setiawan and F. Ramadhani, “Analisis Perbandingan Deteksi Tepi Canny dan Sobel Pada Citra Medis Menggunakan Python,” *J. RESTI (Rekayasa Sist. dan Teknol. Informasi)*, vol. 8, no. 2, pp. 112–119, 2024.
[14] Y. F. Riti and others, “Perbandingan Algoritma Sobel, Prewitt, Robert, Canny, Kirsch, dan Laplacian of Gaussian Dalam Deteksi Tepi Pada Citra Rontgen,” *J. TEKTRIKA*, vol. 9, no. 2, pp. 41–50, 2024.