Penerapan Metode Konvolusi Gabor Filter dan Guided Filter pada Algoritma Canny Edge Detection

A. A. Gde Krisna Murti
Teknik Informatika, Primakara University
Denpasar, Bali
gungkrisna@outlook.com

***Abstrak*—Pengolahan citra merupakan ilmu komputer yang telah diterapkan di berbagai industri, mulai dari diagnosis medis hingga pengolahan citra astronomi. Salah satu teknik fundamental dalam pengolahan citra adalah pendeteksian tepi yang ditujukan untuk menentukan batasan antar objek. Canny merupakan salah satu algoritma pendeteksian tepi yang dikenal atas efektivitas dan akurasinya hingga masih terus digunakan dalam pengembangan teknologi terbaru saat ini seperti ControlNet. Namun, algoritma pendeteksian tepi memiliki kelemahan dalam proses *noise suppression* dengan filter Gaussian yang dikenal tidak dapat mempertahankan kualitas tepi pada citra dengan efektif. Untuk mengatasi permasalahan ini, dilakukan konvolusi filter *Gabor* dan *Guided* dalam proses *noise suppression*. Implementasi ini menghasilkan peningkatan kualitas citra tepi mengacu pada hasil pengujian *Mean Squared Error* (MSE) dengan peningkatan 8.62%, *Peak Signal-to-Noise Ratio* (PSNR) dengan peningkatan 7.06%, dan *Structural Similarity Error* (SSIM) dengan peningkatan 38.46%.**

Keywords—Canny edge detection, Gabor filter, Guided filter, Gaussian filter, image processing

I. PENDAHULUAN

Pendeteksian tepi merupakan teknik pengolahan citra yang digunakan untuk menemukan batasan suatu objek dengan mendeteksi perubahan intensitas piksel yang berdekatan pada suatu citra digital [1]. Beberapa metode lainnya juga sering digunakan untuk meningkatkan kualitas deteksi tepi, seperti menghitung perbedaan gradien, penerapan transformasi matematika seperti Hough *transform* [2], hingga menggunakan filter spasial [3], [4]. Teknik ini umumnya digunakan untuk memisahkan suatu objek dan/atau mengekstraksi elemen penting dari suatu gambar. Beberapa algoritma pendeteksian tepi yang umum digunakan diantaranya adalah Sobel, Laplacian, Prewit, Robert, dan Canny.

Salah satu algoritma pendeteksian tepi yang dikenal paling efektif dan akurat dalam pengolahan citra adalah metode Canny [5]. Algoritma ini melibatkan proses reduksi noise menggunakan filter Gaussian, kalkulasi gradien menggunakan filter Sobel, *non-maximum suppression*, *double threshold*, dan *edge tracking by hysteresis*. Meskipun saat ini telah dikembangkan metode pendeteksian tepi yang kompleks Segment Anything Model yang dikembangkan Meta [6], teknologi pendeteksian tepi yang sederhana namun tangguh seperti Sobel, Prewitt, maupun Canny masih dibutuhkan dengan pertimbangan kecepatan dan biaya komputasi yang lebih efisien. Bahkan, algoritma pendeteksian tepi berbasis gradien masih dipercaya dalam perkembangan bidang *computer vision* terkini, seperti ControlNet yang menggunakan Canny *edge detection* untuk mengontrol gambar yang dihasilkan oleh *image diffusion model* seperti Stable Diffusion [7].

Meskipun efektif, algoritma pendeteksian tepi Canny masih memiliki kelemahan. Hal ini terkait dengan proses *noise suppression* yang menggunakan konvolusi Gaussian filter. Filter linier ini umum digunakan untuk melakukan pengaburan citra agar tekstur citra yang akan diproses menjadi lebih halus dengan tujuan reduksi *noise*. Sayangnya, Gaussian bukan merupakan filter yang dapat mempertahankan kualitas tepi pada citra dengan efektif [8]. Hal ini berdampak pada menurunnya kualitas citra yang dihasilkan oleh algoritma Canny *edge detection*.

Studi ini melakukan implementasi konvolusi filter Gabor dan filter Guided pada tahap *noise suppression* dari algoritma Canny. Penerapan ini dilakukan untuk mengatasi kelemahan Canny *edge detection* yang masih kurang efektif dalam mempertahankan kualitas tepi pada output citra yang dihasilkan. Implementasi dilakukan menggunakan bahasa pemrograman Python untuk setiap langkah dalam algoritma Canny yang diusulkan. Sedangkan untuk pengujian algoritma, studi ini memanfaatkan metrik *Mean Squared Error* (MSE), *Peak Signal-to-Noise Ratio* (PSNR), dan *Structural Similarity Index* (SSIM).

II. METODE

Pendeteksian tepi dengan implementasi filter Gabor dan Guided pada Canny Edge Detection meliputi lima tahap utama, yaitu penyiapan citra digital, *noise suppression*, *gradient computation, non-maxima suppression*, dan *hysteresis thresholding*. Proses pengolahan citra dan pengujian dilakukan menggunakan bahasa pemrograman Python melalui Google Colab. Berikut adalah *flowchart* dari proses pendeteksian tepi.

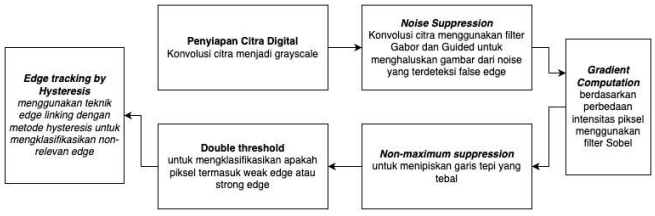

Fig. 1. Alur kerja algoritma pendeteksian tepi yang diusulkan.

A. Penyiapan Citra Digital

Secara teknis, gambar atau citra digital merupakan sekumpulan matriks piksel dalam format baris dan kolom, dengan ukuran piksel yang bergantung pada resolusi dari gambar tersebut. Karena algoritma pengolahan citra ini menggunakan kalkulasi intensitas *grayscale*, konversi gambar ke *grayscale* perlu dilakukan. Metode konversi *grayscale* yang paling sederhana adalah *average method* dengan mengambil nilai rata-rata dari setiap piksel. Namun, studi ini menggunakan *luminosity method* karena lebih akurat

mengingat mata manusia cenderung sensitif terhadap warna hijau dibandingkan spektrum warna lainnya, sehingga bobot G pada persamaannya paling besar. Persamaan matematis dari algoritma ini adalah sebagai berikut:

$$Grayscale = 0.21R + 0.72G + 0.07B \quad (1)$$

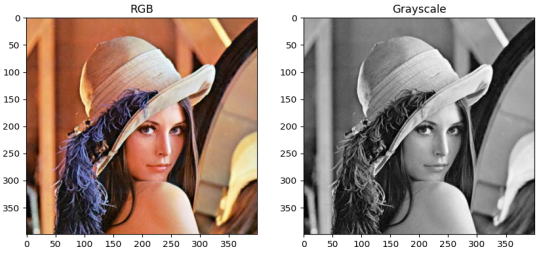

Fig. 2. Hasil konversi citra RGB menjadi Grayscale menggunakan *luminosity method*

B. Noise Suppression

Algoritma Canny menggunakan berbagai metode matematis yang berdasarkan nilai derivatif, contohnya pada tahap *gradient computation*, sehingga hasil pendeteksian tepi menjadi sangat sensitif terhadap keberadaan *noise*. Untuk itu, perlu dilakukan *noise suppression* guna menghaluskan gambar menggunakan filter spasial seperti Gaussian. Namun, mengingat algoritma Gaussian merupakan filter yang melakukan pengaburan kualitas citra digital tanpa mempertahankan kualitas tepi [8], studi ini memilih pendekatan *noise suppression* menggunakan filter spasial Gabor dan Guided. Filter Gabor merupakan filter linier yang digunakan untuk meningkatkan kualitas gambar dengan secara selektif meningkatkan frekuensi tertentu pada komponen citra digital [9]. Filter ini sangat berguna untuk kebutuhan *edge detection, denoising*, dan penajaman kualitas gambar. Berikut adalah persamaan matematis untuk filter Gabor:

$$g(x,y;\lambda,\theta,\psi,\sigma,\gamma) = \frac{1}{2\pi\sigma^2} e^{-\frac{x'^2+\gamma^2y'^2}{2\sigma^2}} e^{i(2\pi\frac{x'}{\lambda}+\psi)} \quad (2)$$

dimana

$$x' = x\cos\theta + y\sin\theta$$
$$y' = -x\sin\theta + y\cos\theta \quad (3)$$

Pada algoritma ini, λ mendefinisikan frekuensi pada tekstur, dan dapat diubah untuk memperoleh tekstur di di suatu arah. Selain itu, terdapat juga θ yang merupakan orientasi dari filter, ψ adalah *phase offset*, σ merupakan deviasi standar dari Gaussian, serta γ yang merupakan aspek rasio spasial. Implementasi filter Gabor pada program membutuhkan parameter input berupa ukuran kernel, orientasi, frekuensi, dan deviasi standar dari Gaussian pada derajat sumbu x dan y.

Sedangkan Guided merupakan *edge-preserving smoothing filter* yang digunakan dalam pengolahan citra [10]. Filter Guided menggunakan struktur linier dengan biaya komputasi yang rendah karena perhitungannya yang sederhana. Algoritma ini mengasumsikan bahwa relasi setiap *guidance* dan *filtering output* adalah linier. Berikut adalah persamaan matematis dari filter Guided:

$$I_p = a_p I_p^g + b_p \quad (4)$$

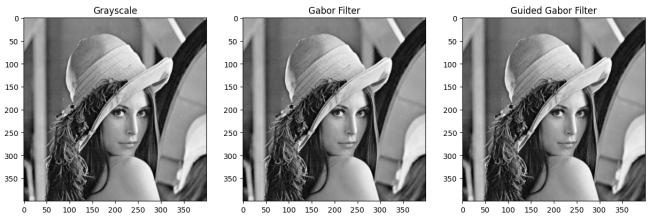

Fig. 3. Hasil konversi citra Grayscale menjadi Gabor yang kemudian di konvolusi menggunakan filter Guided.

C. Gradient Computation

Setelah melakukan langkah *preprocessing image*, selanjutnya adalah melakukan *gradient computation*. Pendeteksian tepi berkaitan dengan perubahan intensitas piksel, sehingga cara termudah untuk mendeteksinya adalah dengan menerapkan filter yang menampilkan perubahan intensitas piksel dalam dua arah, yakni horizontal (***x***) dan vertical (***y***)**.** Metode Canny menggunakan algoritma pendeteksian tepi Sobel guna mengukur intensitas tepi dua arah dengan melakukan konvolusi kernel berukuran 3x3 terhadap gambar [11]. Secara matematis, persamaan konvolusi dari ***I*** dengan kernel Sobel $\boldsymbol{G_x}$ dan $\boldsymbol{G_y}$ adalah sebagai berikut:

$$G_x = \begin{bmatrix} -1 & 0 & 1 \\ -2 & 0 & 2 \\ -1 & 0 & 1 \end{bmatrix} * I \qquad G_y = \begin{bmatrix} -1 & -2 & -1 \\ 0 & 0 & 0 \\ 1 & 2 & 1 \end{bmatrix} * I \quad (5)$$

Kemudian, nilai magnitudo ***G*** dan slope $\boldsymbol{\theta}$ dari gradien dihitung sebagai berikut:

$$G = \sqrt{(G_x)^2 + (G_y)^2} \quad (6)$$

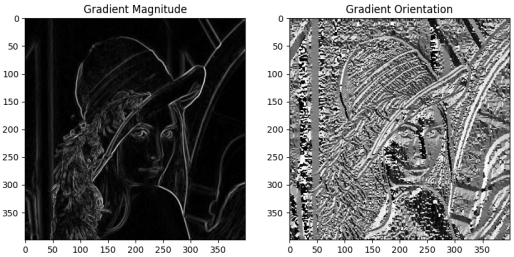

Fig. 4. Hasil konvolusi citra Guided Gabor menggunakan Sobel *edge detection* dengan output *gradient magnitude* dan *gradient orientation*.

D. Non-maximum Suppression

Berdasarkan output yang *gradient magnitude* yang dihasilkan melalui proses konvolusi citra Guided Gabor menggunakan Sobel, terlihat bahwa terdapat inkonsistensi terhadap ketebalan garis tepi. Secara ideal, *edge detection* yang baik menghasilkan tepi yang tipis. Untuk itu, perlu dilakukan langkah berikutnya yakni *non-maximum suppression* untuk menipiskan tepi.

Sebuah gradien dikatakan *non-maximum* apabila gradien tersebut adalah yang terkecil dalam intensitas *grayscale* dibandingkan dengan arah lainnya. Prinsip *non-maximum suppression* secara sederhana adalah dengan mengunjungi setiap titik gradien matriks intensitas dan mencari piksel di segala arah dengan nilai gradien terbesar. *Non-maximum* suppression akan mengubah piksel dengan gradien kecil menjadi nol dan mempertahankan yang terbesar [12].

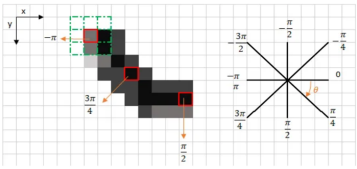

Fig. 5. Ilustrasi algoritma *non-maximum suppression*

Secara teknis, algoritma *non-maximum suppression* diterapkan dengan langkah-langkah sebagai berikut:

- Buat sebuah matriks yang diinisialisasi dengan nilai 0 dengan ukuran yang sama dengan matriks gradien intensitas asli.
- Identifikasi arah tepi berdasarkan nilai sudut pada matriks sudut.
- Periksa apakah piksel dengan arah yang sama memiliki intensitas yang lebih tinggi dari piksel yang sedang diproses.
- Tampilkan gambar yang telah diproses menggunakan algoritma *non-maximum suppression*.

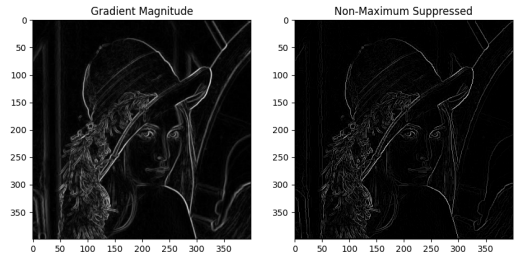

Fig. 6. Hasil pengolahan citra *gradient magnitude* dengan *non-maximum suppression* guna menghasilkan *output* ketebalan tepi citra yang konsisten.

E. Double Threshold

Langkah selanjutnya adalah mengidentifikasi setiap piksel tepi menjadi tiga kategori, yakni *strong*, *weak*, dan *non-relevant*.

- *Strong pixels* merupakan piksel-piksel yang memiliki intensitas tinggi (lebih tinggi dari *high threshold*) sehingga dapat dipastikan merupakan tepi yang valid dalam output *edge detection*.
- *Weak pixels* merupakan piksel-piksel yang memiliki nilai intensitas yang tidak cukup untuk dikategorikan sebagai *strong edge*, namun juga tidak terlalu rendah untuk dikategorikan sebagai *non-relevant edge.*
- Piksel dengan intensitas rendah (lebih rendah dari low threshold) dikategorikan sebagai non-relevant edge.

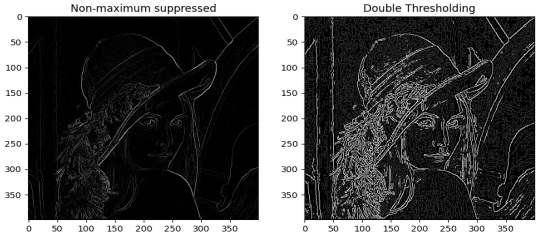

Fig. 7. Hasil *double thresholding* terhadap citra *non-maximum suppressed. Tepi* yang dikategorikan sebagai *weak pixels* berwarna abu-abu dan *strong pixels* berwarna putih.

F. Edge Tracking by Hysteresis

Pada tahap *double thresholding*, citra diidentifikasi ulang dengan kepastian bahwa *strong pixels* ditampilkan pada *output* dan *non-relevant pixels* akan diabaikan. Namun, masih terdapat ketidakpastian terhadap *weak pixels* yang memiliki intensitas diantara *low* dan *high threshold*. Untuk itu, dilakukan proses *edge tracking* melalui mekanisme *Hysteresis* untuk mengidentifikasi *weak pixels* secara lebih lanjut.

Mekanisme Hysteresis melakukan transformasi *weak pixels* menjadi *strong pixels*, hanya apabila di sekitar piksel yang diproses terdapat setidaknya satu *strong pixels*. Mekanisme ini diilustrasikan sebagai berikut:

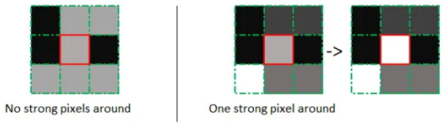

Fig. 8. Ilustrasi mekanisme Hysteresis dalam menentukan kembali apakah *weak pixels* termasuk *non relevant* (kiri) atau dapat ditransformasi menjadi *strong pixels* (kanan).

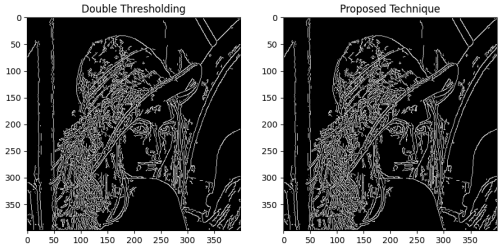

Fig. 9. Citra *double thresholding* setelah melalui mekanisme *Hysteresis* yang juga menjadi hasil *edge detection* akhir dari modifikasi algoritma *Canny* yang diusulkan.

III. HASIL DAN PEMBAHASAN

Algoritma pendeteksian tepi dijalankan menggunakan Python melalui Google Colab dengan sampel gambar Lena. Implementasi konvolusi filter Gabor dan Guided pada algoritma Canny kemudian diuji untuk selanjutnya dibandingkan dengan algoritma Canny asli yang menggunakan filter Gaussian.

A. Mean Squared Error (MSE)

Metrik statistik *mean squared error* (MSE) digunakan untuk membandingkan citra hasil deteksi tepi dengan citra yang sebenarnya. Algoritma pendeteksian tepi disebut memiliki hasil yang baik apabila memiliki nilai MSE yang kecil [13]. Berikut adalah definisi matematis MSE:

$$MSE = \frac{1}{n}\sum_{i=1}^{n}(y_i - \hat{y}_i)^2 \quad (6)$$

TABLE I. NILAI MEAN SQUARED ERROR (MSE)

Algoritma	MSE (10^3)
Yang diusulkan	18.44
Canny detector	20.03

Fig. 10. Nilai *mean squared error* (MSE) berdasarkan hasil pengujian kedua algoritma pendeteksian tepi.

Fig. 10 menunjukkan bahwa algoritma Canny yang menggunakan konvolusi filter Gabor dan Guided pada tahap *noise suppression* memiliki nilai *mean squared error* (MSE) yang lebih rendah dengan selisih 8.62% dibandingkan dengan algoritma Canny yang menggunakan filter Gaussian pada proses *noise reduction*.

B. *Peak Signal-to-Noise Ratio* (PSNR)

PSNR (*Peak Signal-to-Noise Ratio*) merupakan metrik yang digunakan untuk mengevaluasi kualitas citra dengan menghitung perbandingan antara nilai tertinggi sinyal dengan besarnya *noise* yang ada [14]. Semakin besar nilai PSNR maka kualitas deteksi tepi akan semakin baik. Berikut adalah definisi matematis untuk PSNR:

$$PSNR = 20 \cdot \log_{10}(max_i) - 10 \cdot \log_{10}(MSE) \quad (6)$$

TABLE II. NILAI PEAK SIGNAL TO NOISE RATIO (PSNR)

Algoritma	PSNR
Yang diusulkan	5.46
Canny detector	5.10

Fig. 11. Nilai *peak signal to noise ratio* (PSNR) berdasarkan hasil pengujian kedua algoritma pendeteksian tepi.

Fig. 11 menunjukkan bahwa algoritma Canny yang menggunakan konvolusi filter Gabor dan Guided pada tahap *noise suppression* mengalami peningkatan sebesar 7.06% berdasarkan metrik *peak signal to noise ratio* (PSNR) dibandingkan dengan algoritma Canny yang menggunakan filter Gaussian pada proses *noise reduction*.

C. *Structural Similarity Index* (SSIM)

Structural Similarity Index (SSIM) merupakan metode untuk mengevaluasi kemiripan antara dua citra digital. Dalam konteks *edge detection*, SSIM digunakan untuk membandingkan tepi prediksi dengan tepi pada citra asli. Metode SSIM mempertimbangkan tekstur dan struktur citra digital berdasarkan fakta bahwa mata manusia dapat melihat perbedaan dengan akurat pada area tekstur dan tepi dibandingkan pada area halus, dengan melibatkan perhitungan rata-rata, varians, dan kovarian dari luminans, kontras, dan struktur dari dua gambar yang dibandingkan [15]. Nilai SSIM berkisar antara -1 (tidak mirip) dan 1 (kemiripan sempurna).

TABLE III. NILAI STRUCTURAL SIMILARITY INDEX (SSIM)

Algoritma	SSIM
Yang diusulkan	0.0104
Canny detector	0.0064

Fig. 12. Nilai *structural similarity index* (SSIM) berdasarkan hasil pengujian kedua algoritma pendeteksian tepi.

Fig. 12 menunjukkan bahwa algoritma Canny yang menggunakan konvolusi filter Gabor dan Guided pada tahap *noise suppression* mengalami peningkatan sebesar 38.46% berdasarkan metrik *structural similarity index* (SSIM) dibandingkan dengan algoritma Canny yang menggunakan filter Gaussian pada proses *noise reduction*.

IV. KESIMPULAN

Pendeteksian tepi Canny memiliki kelemahan pada proses *noise reduction* karena filter Gaussian tidak memiliki kemampuan untuk mempertahankan tepi dengan baik. Untuk itu, studi ini menerapkan filter Gabor dan Guided yang dikenal sebagai *edge preserving smoothing filter* pada tahap *noise reduction*. Hasil pengujian menunjukkan adanya peningkatan terhadap kualitas deteksi tepi citra, ditunjukkan melalui peningkatan metrik *Mean Squared Error* (MSE) sebesar 8.62%, metrik *Peak Signal-to-Noise Ratio* (PSNR) sebesar 7.06%, dan *Structural Similarity Error* (SSIM) sebesar 38.46%. Pengujian ini mengindikasikan bahwa pendekatan yang digunakan dapat mengatasi kelemahan algoritma Canny dalam mempertahankan kualitas tepi pada citra dengan efektif.

REFERENSI

[1] M. F. V. Ruslau, R. A. Pratama, and E. Meirista, "Edge detection of digital image with different edge types," J Phys Conf Ser, vol. 1569, no. 4, Jul. 2020, doi: 10.1088/1742-6596/1569/4/042069.

[2] L. Cheng, J. Fang, Y. Wu, and K. Kang, "Research on improved image edge detection based on Hough transform," p. 26, Oct. 2021, doi: 10.1117/12.2611452.

[3] Y. Li and B. Liu, "Improved edge detection algorithm for canny operator," IEEE Joint International Information Technology and Artificial Intelligence Conference (ITAIC), vol. 2022-June, pp. 1–5, 2022, doi: 10.1109/ITAIC54216.2022.9836608.

[4] E. H. A. Mansour and F. Bretaudeau, "A novel edge detection method based on efficient gaussian binomial filter," International Journal of Advances in Intelligent Informatics, vol. 7, no. 2, pp. 211–224, Jul. 2021, doi: 10.26555/IJAIN.V7I2.651.

[5] S. R. Joshi and R. Koju, "Study and comparison of edge detection algorithms," 2012 Third Asian Himalayas International Conference on Internet, 2012, doi: 10.1109/AHICI.2012.6408439.

[6] S. Mohapatra, A. Gosai, and G. Schlaug, "Brain Extraction comparing Segment Anything Model (SAM) and FSL Brain Extraction Tool," 2023.

[7] L. Zhang and M. Agrawala, "Adding Conditional Control to Text-to-Image Diffusion Models," Feb. 2023, Accessed: Jul. 01, 2023. [Online]. Available: https://arxiv.org/abs/2302.05543v1

[8] T. Kuang, Q. X. Zhu, and Y. Sun, "Edge detection for highly distorted images suffering Gaussian noise based on improve Canny algorithm," Kybernetes, vol. 40, no. 5, pp. 883–893, Jun. 2011, doi: 10.1108/03684921111142430.

[9] Z. Hussain, "Texture Image Segmentation Using Gabor Filter and Anisotropic Diffusion Filter," 2010.

[10] V. Patil, Dr. P. Malathi, and Dr. M. Sharma, "Edge Preservation using Guided Image Filter Technique," 2015.

[11] D. Gader, "A Comparison of the Sobel Filter in C, OpenCV and CUDA," 2019.

[12] R. Mohan and D. Loganathan, "Edge Preserved and Segmented Image Denoising," 2017.

[13] N. Tariq, R. A. Hamzah, T. F. Ng, S. L. Wang, and H. Ibrahim, "Quality Assessment Methods to Evaluate the Performance of Edge Detection Algorithms for Digital Image: A Systematic Literature Review," IEEE Access, vol. 9, pp. 87763–87776, 2021, doi: 10.1109/ACCESS.2021.3089210.

[14] U. Sara, M. Akter, and M. S. Uddin, "Image Quality Assessment through FSIM, SSIM, MSE and PSNR—A Comparative Study," Journal of Computer and Communications, vol. 07, no. 03, pp. 8–18, 2019, doi: 10.4236/JCC.2019.73002.

[15] K. N. Raju and K. S. P. Reddy, "Comparative study of Structural Similarity Index (SSIM) by using different edge detection approaches on live video frames for different color models," 2017 International Conference on Intelligent Computing, Instrumentation and Control Technologies, ICICICT 2017, vol. 2018-January, pp. 932–937, Apr. 2018, doi: 10.1109/ICICICT1.2017.8342690.