

## PERBANDINGAN ALGORITMA *SOBEL* DAN *CANNY* UNTUK DETEKSI TEPI CITRA DAUN LIDAH BUAYA

Louis Maximillian<sup>1</sup>, Yosefina Finsensia Riti<sup>2</sup>, Mario Anugraha<sup>3</sup>,  
Yohanes Junardi Palis<sup>4</sup>

<sup>1,2,3,4</sup> Teknik Ilmu Informatika Universitas Katolik Darma Cendika  
Jl. Dr. Ir. H. Soekarno No.201, Klampis Ngasem, Kec. Sukolilo, Surabaya, Jawa Timur  
E-mail :louis.maximillian@student.ukdc.ac.id<sup>1</sup>, yosefina.riti@ukdc.ac.id<sup>2</sup>,  
mario.anugraha@student.ukdc.ac.id<sup>3</sup>, yohanes.palis@student.ukdc.ac.id<sup>4</sup>

### Abstrak

Penyakit daun yang umum terjadi pada tanaman lidah buaya, seperti busuk daun, busuk akar, infeksi bakteri, dan serangan virus, dapat menimbulkan kerusakan yang cukup parah. Identifikasi penyakit-penyakit tersebut masih mengandalkan pengalaman petani dan seringkali menimbulkan interpretasi yang salah. Solusi modern telah ditemukan melalui penerapan teknologi informasi, khususnya di bidang pengolahan citra digital. Dengan menggunakan metode ini, diagnosis penyakit pada daun lidah buaya dapat ditingkatkan melalui deteksi tepi objek pada gambar daun. Hasil deteksi tepi ini memungkinkan mengidentifikasi gejala penyakit dengan lebih akurat. Dalam konteks ini, algoritma *Canny* dan *Sobel*, dua algoritma yang umum digunakan untuk deteksi tepi pada gambar, terbukti menjadi pilihan yang efektif. Dengan menggunakan metode tersebut, gambar tepi daun lidah buaya dapat diidentifikasi secara akurat. Ini adalah langkah penting dalam mendukung petani dalam diagnosis dini penyakit dan mengambil tindakan tepat waktu untuk mengatasi masalah ini. Penelitian ini bertujuan untuk mendapatkan algoritma terbaik pendeteksi tepi daun lidah buaya berdasarkan nilai *Mean Squared Error* (MSE) dan *Peak Signal-to-Noise Ratio* (PSNR). Hasil pengujian menunjukkan bahwa algoritma *Sobel* memberikan hasil yang lebih baik dengan rata-rata pengukuran MSE sebesar 2781.88 dan rata-rata PSNR sebesar 14.04, sedangkan algoritma *Canny* memiliki rata-rata MSE sebesar 3542.02 dan rata-rata PSNR sebesar 12.92.

**Kata kunci:** Lidah Buaya, Algoritma *Canny*, Algoritma *Sobel*, MSE, PSNR.

### Abstract

*Leaf diseases that commonly occur in aloe vera plants, such as leaf rot, root rot, bacterial infections, and virus attacks, can cause quite severe damage. Identification of these diseases still relies on farmers' experience and ultimately leads to wrong interpretations. Modern solutions have been found through the application of information technology, especially in the field of digital image processing. By using this method, the diagnosis of diseases on aloe vera leaves can be improved through the detection of object edges in leaf images. The results of this edge detection make it possible to identify disease symptoms more accurately. In this context, the Canny and Sobel algorithms, two algorithms commonly used to detect edges in images, prove to be effective choices. By using this method, the image of the edge of the aloe vera leaf can be identified accurately. This is an important step in supporting farmers in early diagnosis of the disease and taking timely action to address the issue. This research aims to obtain the best algorithm for detecting the edges of aloe vera leaves based on the Mean Squared Error (MSE) and Peak Signal-to-Noise Ratio (PSNR) values. The test results show that the Sobel algorithm provides better results with an average MSE measurement of 2781.88 and an average PSNR of 14.04, while the Canny algorithm has an average MSE of 3542.02 and an average PSNR of 12.92.*

**Keywords:** *Aloe Vera, Canny Algorithm, Sobel Algorithm, MSE, PSNR.*

## 1. PENDAHULUAN

Aloe vera atau lidah buaya berasal dari Afrika Selatan, Madagaskar, dan Arab. Ciri fisik tumbuhan ini adalah daging buah yang tebal, daun panjang, ujung menyempit, daun berwarna hijau dan lengket. Tanaman lidah buaya banyak tumbuh dan berkembang di Indonesia, tanaman ini telah lama dikenal kegunaannya

sebagai tanaman obat terhadap berbagai macam penyakit. Dalam pembudidayaan tanaman lidah buaya perlu diperhatikan kesehatan pada tanaman tersebut, sehingga terhindar dari penyakit, salah satunya adalah penyakit pada daun. Untuk mendeteksi penyakit pada daun, petani dapat melakukan berbagai cara sesuai dengan pengalaman mereka, namun terkadang hasil interpretasi dari petani bisa saja menimbulkan kesalahan. Sehingga diperlukan suatu metode yang dapat membantu para petani melakukan interpretasi pada daun lidah buaya, yaitu dengan bantuan teknologi komputer. Dalam teknologi komputer salah satu bidang ilmu yang dapat membantu dalam diagnosis penyakit daun adalah pengolahan citra digital, dimana dari citra daun dapat dilakukan analisis mengenai penyakit daun dengan lebih mudah dan cepat. Pengolahan citra yang dilakukan adalah melalui deteksi tepi citra daun, dimana dari tepi citra daun dapat diperoleh fitur-fitur penting untuk melakukan analisis lebih lanjut, dan salah satunya adalah mengetahui apakah tanaman *aloe vera* tersebut terindikasi penyakit daun atau tidak.

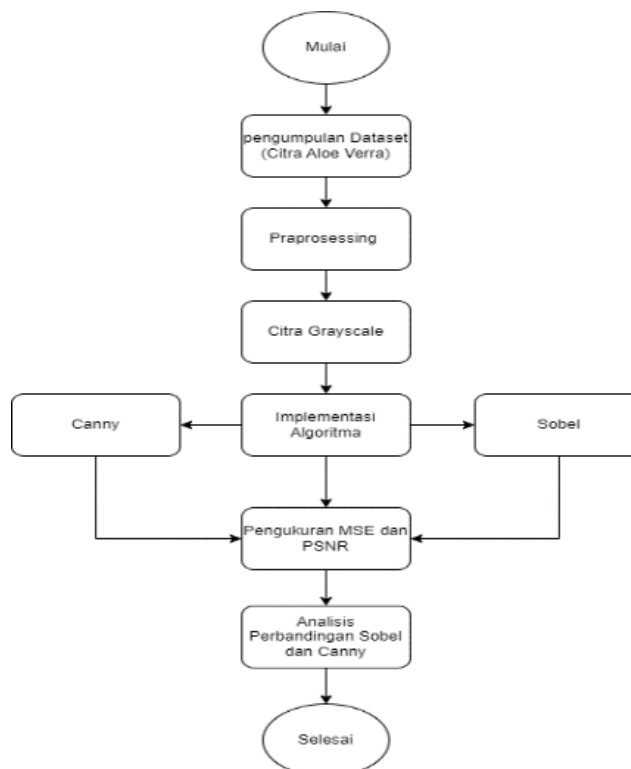
Deteksi tepi adalah langkah awal pada segmentasi gambaran buat menerima informasi mengenai suatu gambaran. Tepi berisi serangkaian titik dalam ketinggian yang berbeda. Hasil deteksi tepi ditampilkan berupa kontur kecerahan banyak sekali objek dalam gambaran. Banyak metode yg dipakai buat mendeteksi tepi dalam gambaran. Pada penelitian ini, memakai metode *Canny & Sobel* buat segmentasi gambaran deteksi tepi & menerapkan proses deteksi tepi memakai kedua metode tersebut[1].

*Sobel*, metode *Sobel* merupakan peningkatan lebih baik dari metode *Robert* dengan HPF (*High Pass Filter*) dengan penggunaan *buffer nol* adalah sebuah teknik yang menggabungkan prinsip dari metode *Sobel* yang memanfaatkan fungsi *Laplacian* dan *Gaussian*, yang dalam hal ini disebut sebagai fungsi HPF (*High Pass Filter*). Metode *Sobel* merupakan *noise* bisa dikurangi sebelum menguji perhitungan deteksi tepi. Gunakan algoritma *Sobel* untuk mendeteksi tepi citra telah dilakukan sebelumnya, dimana *dataset* yang digunakan adalah data citra daun kari sri lanka, hasil deteksi tepi adalah sangat baik dengan PSNR 30.56. Peneliti lainnya dilakukan oleh [2] dengan *dataset* yang digunakan adalah citra huruf A, dimana hasil deteksi tepi menggunakan *Sobel* adalah sangat baik dengan nilai PSNR 30.74. Peneliti lainnya dilakukan oleh [3] dengan *dataset* yang digunakan adalah citra burung cendrawasih, dimana hasil deteksi tepi menggunakan *Sobel* adalah sangat baik dengan nilai MSE 43.56. Metode *Canny* dikenal karena kemampuannya menghasilkan tepi dengan ketebalan 1 piksel yang optimal. Untuk menghilangkan *noise* dalam gambar asli, digunakan kernel turunan *Gaussian* sehingga tepi yang terdeteksi tampak lebih halus. Deteksi tepi yang cerdas memastikan kontrol tepi dengan tingkat kesalahan yang minimal. Dengan menggunakan *operator Canny* dengan cara yang sama, gambar tepi terbaik dapat dihasilkan. Operator berpengalaman biasanya memulai dengan menghaluskan gambar menggunakan *filter Gaussian*. Kelebihan dari metode *Canny* adalah bahwa ia sangat efektif dalam mendeteksi tepi yang lemah dan sangat tahan terhadap *noise*. Penggunaan algoritma *Canny* dalam deteksi tepi citra sudah dilakukan sebelumnya, antara lain penelitian yg dilakukan oleh [2] dimana *dataset* yang digunakan adalah data citra huruf A, hasil deteksi tepi adalah sangat baik dengan PSNR 9.32. Peneliti lainnya dilakukan oleh [4] dengan *dataset* yang digunakan adalah citra uang 100 ribu, dimana hasil deteksi tepi menggunakan *Sobel* adalah sangat baik dengan nilai PSNR 18.34. Peneliti lainnya dilakukan oleh [5] *dataset* yang digunakan adalah citra daun sirih merah, dimana hasil deteksi tepi menggunakan *Sobel* adalah sangat baik dengan nilai MSE.

Dari hasil *literature review* di atas dapat diketahui bahwa algoritma *Sobel* dan *Canny* merupakan algoritma deteksi tepi yang baik dari segi nilai MSE kecil dan nilai PSNR tinggi. Oleh karena itu, pada penelitian ini kami menguji algoritma *Sobel* dan *Canny* untuk deteksi tepi tanaman *aloe vera*. Pada pengujian ini, dilakukan penggunaan parameter pengukuran MSE (*Mean Squared Error*) dan PSNR (*Peak Signal to Noise Ratio*). Tujuan dari pengujian ini adalah untuk menentukan algoritma terbaik dalam mendeteksi tepi tanaman *aloe vera* berdasarkan nilai MSE dan PSNR

## 2. METODOLOGI

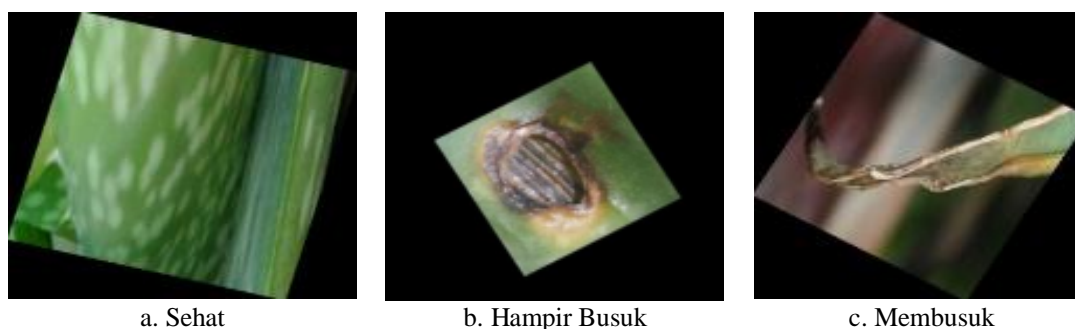
Dalam melaksanakan metode penelitian terdapat beberapa tahapan yaitu sebagai berikut:



Gambar 1. Tahapan Metode Penelitian

### 2.1 Pengumpulan Dataset

Penelitian ini menggunakan 9 gambar *dataset aloe vera* yang diambil dari *kaggle*. *Dataset* dapat diakses melalui *website* [6], *dataset* yang berfungsi untuk membantu mengidentifikasi deteksi tepi pada tanaman lidah buaya, pada Tabel 1 adalah contoh gambar citra *aloe vera* yang digunakan.



Gambar 2. Contoh Citra Lidah Buaya

### 2.2 Praprocessing

Preprocessing atau pra-pemrosesan adalah serangkaian langkah yang dilakukan pada data atau citra sebelum dilakukan analisis atau pemrosesan lebih lanjut. Tujuan dari preprocessing adalah untuk mempersiapkan data agar siap digunakan dalam tahap *grayscale* dengan cara memperbaiki, menghilangkan derau, atau mengubah format data agar sesuai dengan kebutuhan pemrosesan [7].

### 2.3 Citra Grayscale

*Grayscale* adalah daya upaya pengarsipan khayal yang mengubah pandangan hidup piksel pusat suatu khayal menjadi khayal keabuan. Karena khayal *grayscale* adalah khayal yang setiap pikselnya mengandung dunia tambah pandangan hidup semangat bergegar ganggang 0 tampak 255, cerita pandangan hidup piksel

khayal *grayscale* bisa direpresentasikan oleh matriks sehingga mencebikkan daya upaya komputasi khayal *grayscale*. praktik berikutnya. Formula strata abu-abu[8]:

$$\text{Gray} = (R + G + B)/3 \tag{1}$$

Keterangan:

R = Red

G = Green

B = Blue

### 2.3 Implementasi Algoritma

#### A. Algoritma Sobel

*Sobel*, seperti evolusi *operator Prewitt*, merupakan salah satu kemajuan dari metode deteksi tepi sebelumnya (metode *Robert*) yang menggunakan HPF (*high pass filter*) dengan alokasi buffer nol. Algoritma ini berisi algoritma pemrograman yang bertindak sebagai *filter* gambar. *Filter* ini menemukan semua *edge* yang ada. *Filter* ini menggunakan *operator* yang disebut *Sobel*[9]. Ukuran kernel yang digunakan Metode *Sobel* adalah 3x3 piksel untuk menghitung gradien, sehingga perkiraan kemiringan berada di tengah jendela. Matriks seperti itu digunakan untuk mendapatkan piksel pusat, sehingga menjadi pusat matriks. Menggunakan matriks ini seperti menggunakan raster, yaitu mengisi matriks dengan piksel yang diinginkan di sekitarnya (piksel tengah). Misalkan susunan piksel di sekitar piksel (x,y) adalah[10]:

Tabel 1. Tabel Susunan Piksel

$\alpha 0$	$\alpha 1$	$\alpha 2$
$\alpha 7$	(x,y)	$\alpha 3$
$\alpha 6$	$\alpha 5$	$\alpha 4$

Berdasarkan penempatan piksel tetangga, besarnya gradien dihitung menggunakan *operator*. *Sobel* adalah sebagai berikut[10]:

$$G = \sqrt{S_x^2 + S_y^2} \tag{2}$$

Di mana:

G = besarnya gradien *operator Sobel*

Sx = gradien *Sobel* mendatar

Sy= Gradien *Sobel* Vertikal

di mana G adalah besarnya gradien di pusat kernel dan turunan parsial dihitung menggunakan persamaan berikut[10]:

$$S_x = (\alpha 2 \ c \ \alpha 3 \ \alpha 4) - (\alpha 0 \ c \ \alpha 7 \ \alpha 6)$$

$$S_y = (\alpha 0 \ c \ \alpha 1 \ \alpha 2) - (\alpha 6 \ c \ \alpha 5 \ \alpha 4)$$

di mana c adalah konstanta dengan nilai 2. Sx dan Sy diimplementasikan dalam kernel berikut[10]:

Tabel 2. Tabel Kernel

$S_x =$	<table style="border-collapse: collapse; text-align: center;"> <tr><td>-1</td><td>0</td><td>1</td></tr> <tr><td>-2</td><td>0</td><td>2</td></tr> <tr><td>-1</td><td>0</td><td>1</td></tr> </table>	-1	0	1	-2	0	2	-1	0	1	$S_y =$	<table style="border-collapse: collapse; text-align: center;"> <tr><td>1</td><td>2</td><td>1</td></tr> <tr><td>0</td><td>0</td><td>0</td></tr> <tr><td>-1</td><td>-2</td><td>-1</td></tr> </table>	1	2	1	0	0	0	-1	-2	-1
-1	0	1																			
-2	0	2																			
-1	0	1																			
1	2	1																			
0	0	0																			
-1	-2	-1																			

Algoritma metode *Sobel* untuk mendeteksi tepi pada citra digital adalah[10]:

- a. Gambar input adalah gambar skala abu-abu
- b. Konvolusi gambar skala abu-abu dengan kernel *Sobel horizontal* dan vertikal

$$(S_x) = \begin{bmatrix} -1 & 0 & 1 \\ -2 & 0 & 2 \\ -1 & 0 & 1 \end{bmatrix}$$

Gambar 3. Kernel Sobel Horizontal

$$(S_y) = \begin{bmatrix} 1 & 2 & 1 \\ -2 & 0 & 2 \\ -1 & -2 & -1 \end{bmatrix}$$

Gambar 4. Kernel Sobel Vertikal

c. Hitung ukuran gradien menggunakan rumus[10] :

$$G = \sqrt{S_x^2 + S_y^2} \tag{3}$$

d. Gambar yang dicetak merupakan hasil gradasi besar (G)[10].

A. Algoritma Canny

Algoritma Canny adalah algoritma deteksi tepi yang dilakukan dengan menggunakan fungsi matriks gambar dan metode konvolusi operator gaussian[11]. Canny perlu memaksimalkan pendeteksian tepi yang akurat dengan memaksimalkan rasio sinyal ke noise dan tepi yang terdeteksi harus dekat dengan tepi asli. Algoritma Canny bekerja melalui lima tahap yang berbeda: penghalusan, pencarian gradien, redaman non-maksimum, pengambangan ganda, dan pelacakan tepi dengan histeresis. Langkah-langkah ini digunakan untuk menerapkan deteksi tepi yang kompleks. Berikut adalah langkah-langkahnya[12]:

1. Smoothing

Smoothing adalah proses mengaburkan gambar untuk menghilangkan noise. Proses pengaburan ini dilakukan dengan menggunakan filter gaussian. Pemulusan citra dilakukan dengan menerapkan persamaan berikut[13]:

$$G(i, j) = \frac{1}{2\pi\sigma^2} \times e^{-\frac{(i-w)^2+(j-v)^2}{2a^2}} \tag{4}$$

informasi:

$e = 2,71$  (konstanta euler)

$\sigma =$  simpangan baku (sigma)

$\pi = 3,14$  (pi)

2. Finding Gradient

Setelah menghilangkan noise pada proses smoothing, dilakukan proses untuk menentukan resistansi tepi. Tepi harus ditandai pada gambar dengan gradien besar. Operator gradien digunakan dan pencarian dilakukan secara horizontal dan vertikal. Di bawah ini adalah hasil pencarian menggunakan Persamaan[14]

$$G_x \begin{bmatrix} -1 & 0 & 1 \\ -2 & 0 & 2 \\ -1 & 0 & 1 \end{bmatrix} \quad G_y \begin{bmatrix} 1 & 2 & 1 \\ 0 & 0 & 0 \\ -1 & -2 & 1 \end{bmatrix}$$

Gambar 5. Operator Horizontal dan Vertical Algoritma Canny

Hasil dari kedua operator digabungkan untuk memberikan hasil gabungan dari sisi vertikal dan horizontal menggunakan persamaan berikut[14]:

$$G = \sqrt{G_x^2 + G_y^2} \tag{5}$$

Kemudian menentukan arah tepian yang ditemukan dengan menggunakan persamaan[14]

$$\theta = \arctan\left(\frac{G_y}{G_x}\right) \tag{6}$$

3. Non-maximum

Penghapusan penghilangan yang kurang maksimal dilakukan di sepanjang tepi, menghilangkan piksel yang dibuang sebagai tepi gambar. Hanya nilai maksimum yang ditandai sebagai tepi. Ini membuat ujungnya lebih ramping[13].

#### 4. *Double thresholding*

Untuk membuat citra biner, bagi menjadi dua kondisi dimana nilai di bawah T1 diubah menjadi hitam dengan nilai 0, dan nilai di atas T2 diubah menjadi putih dengan nilai 255. Kedua kondisi ini menentukan ambang rendah (T1) dan ambang tinggi (Q2)[15].

#### 5. *Edge Tracking by Hysteresis*

Tepi terakhir ditentukan dengan menekan semua tepi yang tidak terhubung dengan tepi yang kuat. Piksel apa pun yang terhubung ke piksel putih dan memiliki nilai lebih besar dari T1 juga dianggap sebagai tepi[5].

## 2.4 Pengukuran MSE dan PSNR

*Mean Squared Error* (MSE) mengukur sejauh mana perbedaan rata-rata kuadrat. Nilai MSE dihasilkan dari selisih citra yang dihasilkan yang memiliki lokasi piksel yang sama dengan citra aslinya. Semakin tinggi nilai MSE, semakin besar perbedaan antara citra asli dan citra akhir. Selama pemrosesan citra, dekomposisi, rekonstruksi, dan kompresi, nilai MSE menurun. Namun, nilai MSE yang lebih tinggi untuk deteksi tepi gambar berarti lebih banyak tepi gambar yang terdeteksi dan titik tepi gambar yang lebih lemah terdeteksi[16].

Berikut adalah langkah-langkah untuk mengukur MSE (*Mean Squared Error*):

1. Siapkan dua citra yang akan dibandingkan: citra asli (A) dan citra hasil rekonstruksi
2. Pastikan kedua citra memiliki dimensi yang sama. Jika perlu, sesuaikan ukuran citra agar sesuai.
3. Untuk setiap piksel (i, j), hitung selisih antara intensitas piksel pada citra asli (A) dan citra hasil rekonstruksi (B): selisih = A (i, j) - B (i, j)
4. Kuadratkan selisih untuk setiap piksel: selisih kuadrat = selisih<sup>2</sup>
5. Hitung rata-rata dari selisih kuadrat dengan menjumlahkan semua piksel dan membaginya dengan jumlah piksel:  $MSE = \Sigma(\text{selisih\_kuadrat}) / (\text{jumlah piksel})$

Rumus untuk menghitung MSE adalah (Mustafid & 'Uyun, 2017):

$$MSE = (1 / (\text{jumlah piksel})) * \Sigma ((A(i,j) - B(i,j))^2) \quad (7)$$

dimana  $\Sigma$  adalah tanda tambahan, (i,j) adalah lokasi piksel, A(i,j) adalah intensitas piksel pada citra asli, dan B(i,j) adalah intensitas piksel pada citra hasil rekonstruksi. gambar. MSE mewakili perbedaan kuadrat rata-rata antara intensitas piksel dari gambar asli dan yang direkonstruksi. Semakin rendah nilai MSE, semakin tinggi kualitas rekonstruksi citra[16].

Sedangkan PSNR mengukur kualitas gambar yang dimodifikasi dengan membandingkan gambar asli dengan gambar yang dimodifikasi. PSNR dihitung dengan membagi nilai maksimum yang dapat diwakili oleh piksel dengan akar kuadrat dari MSE. Semakin tinggi nilai PSNR maka semakin baik kualitas gambar yang dihasilkan. Kedua metrik ini biasanya digunakan dalam pemrosesan gambar digital untuk membandingkan kualitas gambar yang dibuat menggunakan teknik pemrosesan gambar yang berbeda seperti kompresi gambar dan pengurangan *noise*. PSNR (*Peak Signal-to-Noise Ratio*) adalah metrik yang digunakan untuk mengukur kualitas gambar yang diubah. Formulasinya adalah[16]:

$$PSNR = 20 * \log_{10}(\text{PIXEL\_MAX} / \text{sqrt}(MSE)) \quad (8)$$

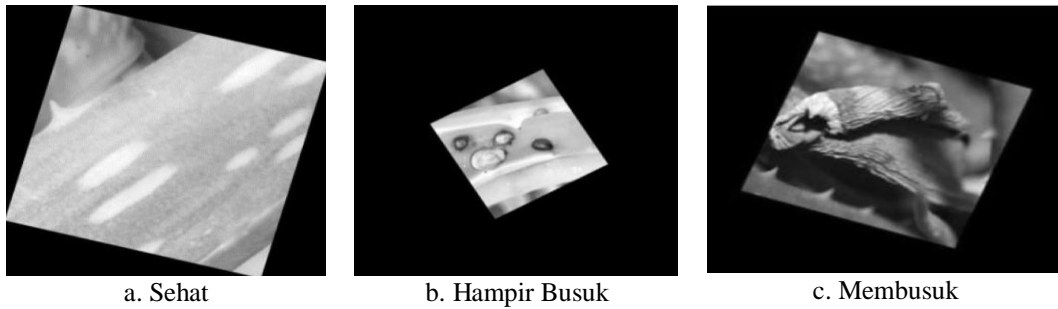
## 3. HASIL DAN PEMBAHASAN

### 3.1 Hasil Preprocessing

Pada Tahap *preprocessing* ini adalah semua ukuran citra *aloe vera* yang awalnya ukuran gambarnya berbeda telah disamakan menjadi ukuran 257x252, ini adalah langkah awal untuk memproses citra *aloe vera* menjadi citra *grayscale*.

### 3.2 Hasil Grayscale

Dibawa ini adalah contoh hasil dari citra asli *aloe vera* yang diubah ke dalam citra *grayscale*.

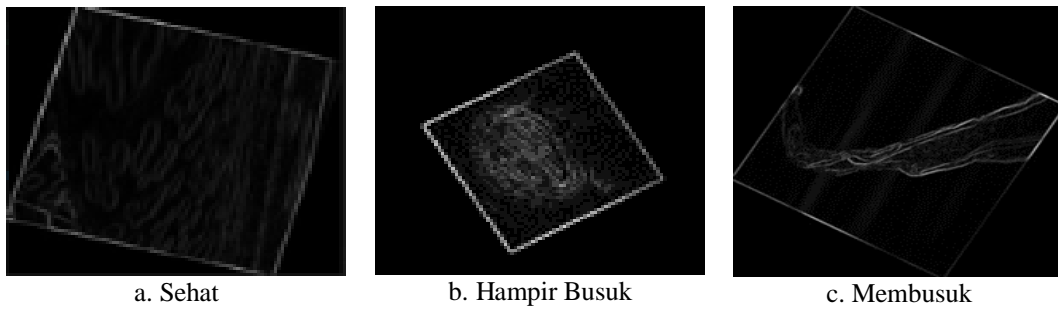


Gambar 6. Contoh Citra *Grayscale* Lidah Buaya

### 3.3 Deteksi Tepi Citra *Aloe Vera*

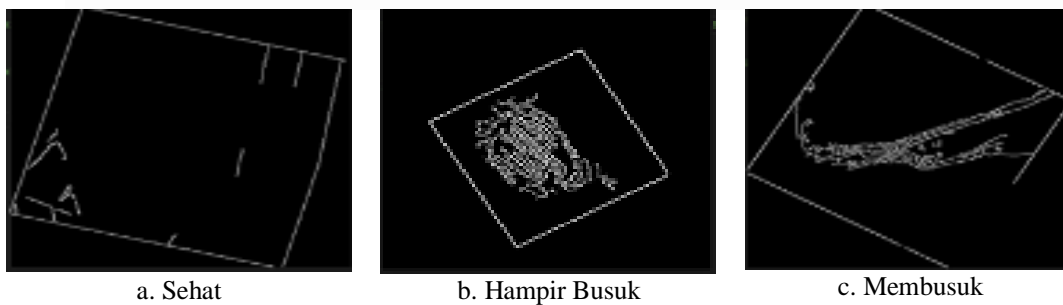
Deteksi tepi adalah teknik pemrosesan gambar yang digunakan untuk mengidentifikasi area dengan perubahan intensitas yang tiba-tiba dalam suatu gambar. Dengan kata lain, deteksi tepi bertujuan untuk menemukan batasan atau batas antara subjek dan latar belakang gambar. Deteksi tepi penting dalam banyak aplikasi, seperti segmentasi objek, pengenalan pola, dan analisis gambar. Penelitian ini membandingkan kinerja dari dua algoritma deteksi tepi yaitu *Sobel* dan *Canny* dalam deteksi tepi pada citra *aloe vera*.

#### A. Hasil deteksi tepi dengan Algoritma *Sobel*



Gambar 7. Contoh Hasil Deteksi Tepi Citra *Aloe Vera*

#### B. Hasil deteksi tepi dengan Algoritma *Canny*






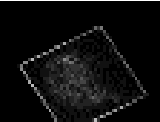
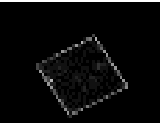

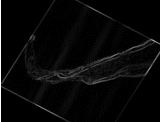

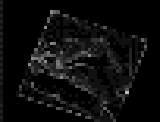
Gambar 8. Contoh Hasil Deteksi Tepi Citra *Aloe Vera*



**3.4 Hasil Pengukuran MSE dan PSNR**

a. Algoritma *Sobel*

**Tabel 3.** Hasil Pengukuran MSE dan PSNR Citra *Aloe Vera*



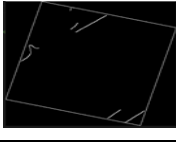

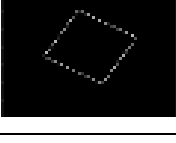
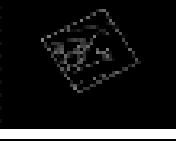



No	Citra	Gambar	MSE	PSNR
1	Sehat 1		2952.65	13.43
2	Sehat 2		2603.33	13.98
3	Sehat 3		1977.98	15.17
4	Hampir Membusuk 1		2730.07	13.43
5	Hampir Membusuk 2		3923.90	12.19
6	Hampir Membusuk 3		2654.64	13.89
7	Membusuk 1		2508.98	14.14
8	Membusuk 2		3507.07	12.68
9	Membusuk 3		2187.47	14.73
	Rata Rata		2782.88	14,04

Hasil pengukuran rata rata MSE dan PSNR pada citra *aloe vera* dengan algoritma *Sobel* dapat dilihat Pada Tabel 7, dengan rata rata MSE 2781.88 dan PSNR 14.04.



b. Algoritma *Canny*

**Tabel 4.** Hasil Pengukuran MSE dan PSNR Citra *Aloe Vera*

No	Citra	Gambar	MSE	PSNR
1	Sehat 1		3528.16	12.66
2	Sehat 2		3010.34	13.34
3	Sehat 3		1686.40	15.86
4	Hampir Membusuk 1		3334.38	12.90
5	Hampir Membusuk 2		6359.03	10.10
6	Hampir Membusuk 3		2599.94	13.98
7	Membusuk 1		2508.98	11.96
8	Membusuk 2		4604.81	11.50
9	Membusuk 3		2618.52	13.95
	Rata Rata		3542.02	12.92

Hasil pengukuran rata rata MSE dan PSNR citra *aloe vera* dengan algoritma *Sobel* dapat di lihat Pada Tabel 8, dengan rata rata MSE 3542.02 dan PSNR 12.92.

### 3.5 Analisis Hasil

Berdasarkan hasil analisis pada PSNR dan MSE pada analisa ini dapat dilihat pada Tabel 3 dan Tabel 4 di citra nomor 9 di masing-masing algoritma, tampilan citra lebih jelas tepinya dan bukan seperti bercak putih di keseluruhan citra *aloe vera* ketika menggunakan algoritma *Sobel*. Sehingga dapat disimpulkan bahwa metode *Sobel* memiliki kinerja yang baik dalam mendeteksi tepi pada citra. Hal ini dikarenakan metode *Sobel* memiliki beberapa kelebihan, yaitu [17] memberikan deteksi tepi yang lebih akurat dan kualitas citra yang lebih baik daripada algoritma *Canny*, memberikan rata-rata skor MSE yang lebih rendah dan skor rata-rata PSNR yang lebih tinggi daripada algoritma *Canny*, dan dapat digunakan untuk mendeteksi tepi pada citra dengan baik.

Selain itu dapat di lihat dari hasil pengukuran MSE dan PSNR citra *aloe vera* dimana algoritma *Sobel* memberikan rata-rata MSE 2782.88 dan PSNR 14.04, sedangkan algoritma *Canny* memberikan rata-rata MSE 3542.02 dan PSNR 12.92. Penelitian ini menunjukkan bahwa algoritma *Sobel* memiliki rata-rata skor MSE yang lebih rendah dan skor rata-rata PSNR yang lebih tinggi daripada algoritma *Canny*.

## 4. PENUTUP

Hasil analisis algoritma *Sobel* dan *Canny* pada gambar *aloe vera* menunjukkan bahwa algoritma *Sobel* memiliki rata-rata MSE sebesar 2782.88 dan PSNR sebesar 14.04, sedangkan algoritma *Canny* memiliki rata-rata MSE sebesar 3542.02 dan PSNR sebesar 12.92. Karena MSE lebih rendah dan PSNR lebih tinggi pada algoritma *Sobel*, maka dapat disimpulkan bahwa algoritma *Sobel* lebih baik dibandingkan algoritma *Canny* dalam mendeteksi tepi dan menghasilkan gambar dengan kualitas lebih tinggi pada tepi. Untuk penelitian selanjutnya dapat menggunakan algoritma yang lainnya seperti algoritma *Robert*, untuk mendapatkan rata-rata PSNR yang lebih besar dan rata-rata MSE yang lebih kecil untuk kasus yang sama atau berbeda. Sehingga dapat membantu efektivitas deteksi tepi terbaik, dan menjadi pembeda di masing-masing algoritma.

## DAFTAR PUSTAKA

- [1] A. Zalukhu, "Implementasi Metode Canny Dan Sobel Untuk Mendeteksi Tepi Citra," *Jurikom*), vol. 3, no. 6, pp. 25–29, 2016.
- [2] A. A. R. Andi Sofyan Anas, "Deteksi Tepi dalam Pengolahan Citra Digital," *Semin. Nas. TIK dan Ilmu Sos.*, pp. 1–6, 2017.
- [3] N. K. A. W. Putu Teguh Krisna Putra, "Pengolahan Citra Digital Deteksi Tepi Untuk Membandingkan Metode Sobel, Robert dan Canny," *MERPATI*, vol. 2, no. 2, pp. 253–261, 2014.
- [4] Wahyu Supriyatn, "Perbandingan Metode Sobel, Prewitt, Robert dan Canny pada Deteksi Tepi Objek Bergerak," *ILKOM*, vol. 12, no. 2, pp. 112–120, 2020.
- [5] Edy Winarno, "Aplikasi Deteksi Tepi pada Realtime Video menggunakan Algoritma Canny Detection," *J. Teknol. Inf. Din.*, vol. 16, no. 1, pp. 44–49, 2011.
- [6] Kaggle, "Aloe Vera," *Kaggle*.
- [7] J. W. Yodha and A. W. Kurniawan, "Pengenalan Motif Batik Menggunakan Deteksi Tepi Canny Dan K-Nearest Neighbor," *Techno.COM*, vol. 13, no. 4, November, pp. 251–262, 2014.
- [8] Teresa, "Pewarnaan Citra Grayscale ke dalam Citra Berwarna dengan Menggunakan Pseudocoloring berbasis Palet Warna," 2019.
- [9] K. Fitriya and M. H. Kom, "Segmentasi Region of Interest ( Roi ) Garis Telapak Tangan," *J. Explor. It!*, vol. 11, no. 1, pp. 29–40, 2019.
- [10] C. Mauludin, "DETEKSI TEPI TINGKAT TRANSPARAN BATU PERMATA," vol. 6, no. 01, pp. 45–50, 2017.
- [11] A. N. Hermans and S. Juerman, "Implementasi Algoritma Canny dan Backpropagation dalam Pengenalan Pola Rumah Adat."
- [12] Y. Ningsih, "Implementasi Metode Canny Edge Detection Untuk Identifikasi Defect Solder," vol. II, pp. 39–46, 2021.
- [13] E. Budianita, D. Muliani, F. Yanto, and Pizaini, "Penerapan Algoritma Canny Dan LVQ 3 Untuk Klasifikasi Jenis Tanaman Mangga," *Ejournal.Uin-Suska.Ac.Id*, vol. 12, no. November, pp. 1–12, 2019.
- [14] N. P., K. Kusriani, and M. P. Kurniawan, "Segmentasi Citra Ikan Arwana Super Red Berdasarkan Deteksi Tepi Menggunakan Algoritma Canny," *J. Teknol. Inf.*, vol. 3, no. 2, p. 200, 2019, doi: 10.36294/jurti.v3i2.1092.

- 
- [15] V. A. Effendy and F. Maspiyanti, "Perbandingan Algoritma Canny Edge Detection Dan Prewitt Pada Deteksi Stadium Diabetik Retinopati," *J. Ilm. Inform.*, vol. 9, no. 02, pp. 87–94, 2021, doi: 10.33884/jif.v9i02.3762.
- [16] A. Mustafid and S. 'Uyun, "Segmentasi Citra Sapi Berbasis Deteksi Tepi Menggunakan Algoritma Canny Edge Detection," *J. Buana Inform.*, vol. 8, no. 1, pp. 27–36, 2017, doi: 10.24002/jbi.v8i1.1074.
- [17] K. Letelay, "Perbandingan Kinerja Metode Deteksi Tepi Pada Citra," *J-Icon*, vol. 7, no. 1, pp. 1–8, 2019.
- [18] B. Sinaga, J. Manurung, M. H. Silalahi, and S. Ramen, "Deteksi Tepi Citra Dengan Metode Laplacian of Gaussian Dan Metode Canny," *J. Sains Komput. Inform.*, vol. 5, no. 2, pp. 1066–1084, 2021, [Online]. Available: <https://tunasbangsa.ac.id/ejurnal/index.php/jsakti>