

Penerapan Hue, Saturation, Value (HSV) dengan Sobel dalam Segmentasi Citra Digital untuk Deteksi Kematangan Tomat.

Herriyance¹, Zaimah Panjaitan², Muhammad Dahria³, Beni Andika⁴

¹S1 Ilmu Komputer, Universitas Sumatera Utara

^{2,3,4} Sistem Informasi, STMIK Triguna Dharma

Email: ¹Herriyance@usu.ac.id, ²zaimahp09@gmail.com, ⁴mdahria1@gmail.com, ⁴beniandika2010@gmail.com

Email Penulis Korespondensi: Herriyance@usu.ac.id

Abstrak

Klasifikasi tingkat kematangan buah tomat secara manual sering menghasilkan penilaian yang tidak konsisten, yang dapat menimbulkan kerugian ekonomi dalam industri pertanian. Maka dari itu diperlukan sistem yang dapat menggabungkan ilmu pengolahan citra digital dalam penilaian kematangan tomat. Metodologi yang digunakan mencakup digitalisasi gambar melalui penangkapan citra tomat lewat kamera secara *real time*. Selanjutnya dilakukan *preprocessing* menggunakan teknik high-pass filtering untuk penajaman citra dan peningkatan kontras. Setelah itu dilakukan transformasi nilai warna dari RGB (red/merah, green/hijau, blue/biru) ke HSV (Hue/warna dasar, saturation/saturasi, value/nilai warna) untuk uji coba analisis warna dan juga deteksi tepi menggunakan operasi sobel. Warna dan tepi tersebut akan menciptakan segmentasi untuk menjadi acuan klasifikasi tomat. Untuk menguji lebih jauh dalam akurasi segmentasi dengan HSV, maka dilakukan perbandingan terhadap model segmentasi dengan RGB dan juga simulasi kualitas kamera rendah menggunakan noise dan blurring. Kombinasi teknik deteksi tepi dan segmentasi berbasis HSV akan memberikan akurasi 54,7% sementara segmentasi berbasis RGB dengan akurasi 72%. Akurasi juga akan menurun ketika jarak antara objek dan kamera semakin panjang. Hasil luaran dari konsep HSV adalah hasil klasifikasi tomat dari tomat mentah, tomat hampir matang dan tomat matang yang ditampilkan pada pemrograman web yang dibangun menggunakan arsitektur HTML (*Hypertext Markup Language*) dan JavaScript.

Kata Kunci : Klasifikasi, Kematangan Tomat, Pengolahan Citra Digital, *High-Pass Filtering*, *Sobel*

Abstract

The manual classification of tomato ripeness levels often results in inconsistent assessments, which can lead to economic losses in the agricultural industry. Therefore, a system that integrates digital image processing is needed to assess tomato ripeness. The methodology involves image digitization by capturing real-time images of tomatoes using a camera. Next, preprocessing is carried out using high-pass filtering techniques to enhance image sharpness and contrast. Then, color values are transformed from RGB (Red, Green, Blue) to HSV (Hue, Saturation, Value) for experimental color analysis, along with edge detection using the Sobel operation. The resulting color and edge data are used to create segmentation, which serves as a reference for tomato classification. To further evaluate segmentation accuracy using HSV, a comparison is made with RGB-based segmentation models and simulations of low camera quality using noise and blurring. The combination of edge detection and HSV-based segmentation yields an accuracy of 54.7%, while RGB-based segmentation achieves 72% accuracy. Accuracy also decreases as the distance between the object and the camera increases. The output of the HSV-based concept is the classification result of tomatoes into unripe, nearly ripe, and ripe categories, which is displayed through a web application built using HTML (*Hypertext Markup Language*) and JavaScript architecture.

Keywords: Classification, Tomato Ripeness, Digital Image Processing, *High-Pass Filtering*, *Sobel*

1. PENDAHULUAN

Dalam industri pertanian modern, penentuan tingkat kematangan buah tomat merupakan faktor penting yang sangat memengaruhi kualitas produk serta nilai jual di pasaran. Proses klasifikasi kematangan yang masih banyak dilakukan secara manual oleh petani atau pekerja lapangan kerap menghasilkan penilaian yang tidak konsisten karena bergantung pada persepsi visual manusia. Hal ini menjadi semakin kompleks ketika berhadapan dengan volume produksi yang tinggi dalam waktu terbatas, yang meningkatkan potensi terjadinya kesalahan klasifikasi dan menyebabkan kerugian ekonomi.

Kondisi lingkungan pengambilan gambar yang tidak ideal seperti pencahayaan buruk, noise dari kamera, serta kualitas gambar yang rendah juga menjadi tantangan besar dalam implementasi sistem klasifikasi otomatis [1]. Untuk menjawab tantangan ini, solusi yang ditawarkan adalah mengembangkan sistem klasifikasi kematangan tomat berbasis aplikasi web real-time yang mengandalkan teknologi pengolahan citra digital.

Salah satu pendekatan efektif dalam sistem ini adalah penggunaan transformasi ruang warna RGB ke HSV. Ruang warna HSV dinilai lebih stabil terhadap perubahan intensitas pencahayaan dan lebih representatif dalam menggambarkan persepsi warna yang dilihat oleh mata manusia. Beberapa penelitian telah menunjukkan efektivitas HSV dalam klasifikasi buah, salah satunya adalah studi oleh Ramadhan dan Nugraha (2021) yang berhasil mengidentifikasi objek berdasarkan warna dan bentuk dengan HSV dan kontur [2]. Penelitian oleh Sari et al. (2021) juga menunjukkan keberhasilan metode K-Nearest Neighbor (K-NN) dalam mengklasifikasi buah berdasarkan warna dalam ruang HSV [3]. Selanjutnya, Humaira et al. (2021) menerapkan metode Jaringan Syaraf Tiruan (JST) dengan fitur HSV untuk menentukan tingkat kualitas dan kematangan buah tomat [4].

Dari hasil referensi, teridentifikasi adanya gap yang dapat dijadikan dasar dalam Penelitian ini. Pertama, belum banyak penelitian yang mengintegrasikan alur preprocessing dan segmentasi warna HSV secara menyeluruh dalam aplikasi web real-time untuk klasifikasi kematangan tomat. Kedua, integrasi metode segmentasi tepi dan transformasi ruang warna untuk meningkatkan ketahanan sistem terhadap kondisi gambar yang buruk masih jarang dijumpai. Ketiga, penerapan sistem klasifikasi tomat yang dapat diakses secara luas melalui platform web masih terbatas.

Tujuan dari penelitian ini adalah menganalisa konsep klasifikasi kematangan buah tomat secara real-time dengan tingkat akurasi tinggi, bahkan dalam kondisi pencahayaan dan kualitas gambar yang tidak ideal di salah satu aplikasi berbasis website. Aplikasi ini akan mengimplementasikan pipeline pengolahan citra digital yang mencakup digitalisasi gambar, preprocessing (deblurring dan contrast enhancement), transformasi warna RGB ke HSV, segmentasi berbasis HSV dan deteksi tepi, serta klasifikasi akhir berdasarkan fitur warna. Dengan sistem ini, diharapkan tercipta alat bantu yang objektif, efisien, dan akurat untuk membantu petani dan pelaku industri pertanian dalam meningkatkan produktivitas serta mengurangi potensi kerugian akibat kesalahan klasifikasi manual. Selain itu, sistem ini juga menjadi bagian dari kontribusi terhadap pengembangan teknologi smart farming yang dapat diakses luas di Indonesia.

2. METODOLOGI PENELITIAN

2.1 Tahapan Penelitian

Tahapan Penelitian ini menjelaskan alur pengerjaan dalam membangun sistem klasifikasi tingkat kematangan tomat secara real-time berbasis web. Tujuan dari Penelitian ini adalah mengklasifikasikan tomat menjadi lima kategori (Busuk, Matang, Hampir matang, Setengah matang, Mentah) dengan menggunakan kamera dan metode pengolahan citra digital secara manual dengan JavaScript.

Tabel 1. Tahapan Pengerjaan Penelitian

No.	Tahapan	Deskripsi
1.	Digitalisasi Citra	Mengakses kamera dengan <code>getUserMedia()</code> dan menangkap frame video.
2.	Simulasi Kualitas Kamera Buruk	Menambahkan noise Gaussian dan efek blur untuk menguji ketahanan sistem.
3.	Deblurring & Kontras	Menggunakan <i>High-Pass Filtering</i> untuk deblurring dan Linear Contrast Enhancement.
4.	Transformasi Warna	Mengubah warna dari RGB ke HSV untuk mempermudah analisis warna.
5.	Deteksi Tepi (Edge Detection)	Menggunakan operator Sobel untuk menyoroti kontur tomat.
6.	Segmentasi	Kombinasi segmentasi HSV dan hasil edge detection untuk memisahkan objek.
7.	Klasifikasi	Aturan berdasarkan nilai Hue dan keberadaan kontur; diuji dengan atau tanpa noise.

2.2 Penerapan Metode dalam Penelitian

2.2.1 Arsitektur Sistem (HTML + JS)

Sistem dikembangkan dengan memanfaatkan:

- HTML** : untuk kerangka halaman dan elemen UI seperti `<video>`, `<canvas>`, dan tombol kontrol.
- JavaScript** : untuk seluruh logika pengolahan citra, mulai dari akses kamera, manipulasi piksel, hingga klasifikasi objek tomat.

2.2.2 Digitalisasi (Akses Kamera)

- Menggunakan Web API (`navigator.mediaDevices.getUserMedia`) untuk menangkap video secara real-time.
- Citra ditampilkan pada elemen `<video>`, kemudian ditransfer ke `<canvas>` untuk analisis data piksel (RGBA).

2.2.3 Simulasi Noise dan Blur

Untuk menguji ketahanan sistem terhadap kondisi kamera yang buruk, dua simulasi diterapkan:

- Noise** : Gaussian Noise ditambahkan ke setiap piksel warna dengan deviasi tertentu.
- Blur** : Menggunakan Gaussian Blur 3x3 seperti pada Gambar 2.

Rumus Noise:

$$N(\mu = 0, \sigma) \quad (1)$$

Keterangan:

- $N(\mu = 0, \sigma)$: bilangan acak dari distribusi Gaussian dengan mean = 0 dan standard deviation = σ
- $\mu = 0$ (rata-rata)
- σ : standar deviasi (nilai konstanta 8)

Rumus Blur:

$$I_{\text{blur}}(x, y) = \frac{1}{16} \sum_{i=-1}^1 \sum_{j=-1}^1 I(x+i, y+j) \cdot K(i, j) \quad (2)$$

Keterangan:

- $K(i, j)$: elemen kernel Gaussian:

Kernel Gaussian Blur

$$\begin{bmatrix} 1 & 2 & 1 \\ 2 & 4 & 2 \\ 1 & 2 & 1 \end{bmatrix}$$

- $I(x+i, y+j)$: nilai intensitas piksel tetangga di posisi $(x+i, y+j)$

Konvolusi digunakan untuk mengurangi noise pada citra. Kernel Gaussian 3x3 diterapkan untuk melakukan *smoothing* terhadap piksel gambar. Nilai setiap piksel baru dihitung dari bobot intensitas tetangga-tetangganya. Proses ini penting untuk mempermudah segmentasi pada tahap selanjutnya [1].

2.2.4 Deblurring dan Kontras

- a. **Deblurring:** High-pass filtering diterapkan dengan kernel seperti di Gambar 3 untuk menekankan detail tepi dan menghilangkan efek blur [16].

$$\text{Output}(i, j) = \sum_m \sum_n \text{Input}(i + m, j + n) \times \text{Kernel}(m, n)$$

Keterangan:

- $\text{Input}(i + m, j + n)$: Nilai piksel dari citra asli pada posisi $(i + m, j + n)$. Ini menunjukkan bahwa piksel pusat bergeser sesuai posisi kernel.
- $\text{Kernel}(m, n)$: Matriks kernel (atau filter) yang berisi bobot-bobot yang akan dikalikan dengan nilai piksel di sekitar titik (i, j) .
- $\text{Output}(i, j)$: Hasil akhir setelah operasi konvolusi di posisi (i, j) . Ini adalah nilai baru piksel setelah filtering.
- $\Sigma\Sigma$ (double summation): Menunjukkan bahwa kita menjumlahkan hasil perkalian antara nilai input dan kernel untuk seluruh elemen dalam jendela kernel.

Kernel Deblurring

$$\begin{bmatrix} -1 & -1 & -1 \\ -1 & 9 & -1 \\ -1 & -1 & -1 \end{bmatrix}$$

- b. **Kontras:** Peningkatan kontras dilakukan dengan rumus linear:

$$\text{newValue} = \text{factor} \times (\text{oldValue} - 128) + 128 \quad (4)$$

$$\text{factor} = \frac{(259 \times (\text{contrast} + 255))}{255 \times (259 - \text{contrast})} \quad (5)$$

2.2.5 Transformasi

Untuk meningkatkan efektivitas dalam analisis warna dan segmentasi objek, dilakukan transformasi ruang warna:

- a. **RGB ke HSV:** Citra yang semula berada dalam format RGB (Red, Green, Blue) diubah menjadi format HSV (Hue, Saturation, Value).
- b. **Rumus Transformasi RGB ke HSV:**

- Normalisasi nilai RGB ke rentang $[0, 1]$:

$$R = \frac{r}{255}, G = \frac{g}{255}, B = \frac{b}{255} \quad (6)$$

- Hitung komponen dasar:

$$C_{\max} = \max(R, G, B) \quad (7)$$

- Hue (H) – dalam derajat $(0-360^\circ)$:

$$H = \begin{cases} 0, & \text{jika } \Delta = 0 \\ 60 \times \left(\frac{G-B}{\Delta} \bmod 6 \right), & \text{jika } C_{\max} = R \\ 60 \times \left(\frac{B-R}{\Delta} + 2 \right), & \text{jika } C_{\max} = G \\ 60 \times \left(\frac{R-G}{\Delta} + 4 \right), & \text{jika } C_{\max} = B \end{cases} \quad (8)$$

- Dinormalisasi ke rentang $[0, 1]$:

$$h = \frac{H}{360} \quad (9)$$

Transformasi ini memisahkan informasi warna (Hue) dari pencahayaan (Value), sehingga proses deteksi dan segmentasi menjadi lebih stabil terhadap perubahan intensitas cahaya. HSV juga lebih mendekati persepsi warna manusia dibandingkan RGB, sehingga mempermudah dalam menetapkan ambang batas warna pada tahap segmentasi [17].

2.2.6 Edge Detection

Untuk menyoroti batas objek dan memisahkan tomat dari latar belakang, dilakukan proses deteksi tepi:

- a. **Edge Detection dengan Kernel Sobel:** Deteksi tepi dilakukan menggunakan operator Sobel yang mengkombinasikan konvolusi dengan dua kernel (horizontal dan vertikal). Metode ini menyoroti perubahan intensitas piksel yang signifikan, yang biasanya terjadi pada kontur objek [18]. Dengan matrik kernel *Kernel Sobel X* $\begin{bmatrix} -1 & 0 & 1, -202, -101 \end{bmatrix}$ dan *Kernel Sobel Y* $\begin{bmatrix} -1 & -2 & -1, 0 & 0 & 0, 1 & 2 & 1 \end{bmatrix}$, serta Gradien total:

$$G = \sqrt{G_x^2 + G_y^2} \quad (10)$$

Dengan demikian, kontur tomat dapat diidentifikasi secara lebih jelas, mempermudah proses segmentasi dan ekstraksi fitur pada tahap selanjutnya.

2.2.7 Segmentasi

Untuk memisahkan objek tomat dari latar belakang secara akurat, dilakukan proses segmentasi berbasis kombinasi informasi warna dan tepi:

Segmentasi (HSV + Edge detection): Segmentasi dilakukan dengan menggabungkan hasil transformasi HSV dan deteksi tepi. Warna (Hue) digunakan untuk menyaring area dengan karakteristik warna khas tomat, sementara informasi tepi dari deteksi Sobel digunakan untuk memperjelas batas objek [19].

Pendekatan gabungan ini meningkatkan akurasi segmentasi, terutama pada citra dengan latar yang kompleks atau pencahayaan tidak merata, karena mempertimbangkan baik informasi warna maupun struktur objek.

2.2.8 Klasifikasi

Setelah proses segmentasi, langkah selanjutnya adalah mengklasifikasikan objek tomat berdasarkan ciri-ciri visualnya:

- Klasifikasi (Hue + Kontur):** Klasifikasi dilakukan dengan memanfaatkan informasi warna (Hue) dan bentuk (kontur) dari objek yang telah disegmentasi. Warna Hue digunakan untuk mengidentifikasi tingkat kematangan tomat, sementara kontur membantu membedakan bentuk atau ukuran yang mungkin relevan untuk klasifikasi kualitas [20].
- Proses klasifikasi diuji dalam dua kondisi:** Citra normal dan citra yang telah mengalami simulasi kualitas buruk (noise dan blur).

Evaluasi ini bertujuan untuk mengukur ketahanan model terhadap citra dari kamera dengan kualitas rendah.

3. HASIL DAN PEMBAHASAN

Pada bagian ini dipaparkan hasil implementasi sistem klasifikasi kematangan tomat berbasis pengolahan citra digital dengan tahapan preprocessing, transformasi warna, segmentasi, dan klasifikasi. Hasil diwakili oleh gambar pada tiap tahap, serta dijelaskan proses dan manfaat dari setiap metode.

3.2 Hasil Proses Pengolahan Citra

3.2.1 Digitalisasi Citra

berikut jenis tomat yang bersumber dari researchgate.net (tomat mentah), halosehat.com (tomat setengah matang), dan id.made-in-china.com (tomat matang).



Gambar 1. tomat mentah Gambar 2. tomat setengah matang Gambar 3. tomat matang

Gambar yang akan dilakukan dalam perhitungan adalah *Gambar 3* yaitu gambar tomat matang. Pengujian citra akan diambil secara langsung dari webcam. Tahapan ini merepresentasikan input mentah tanpa perlakuan. Nilai piksel di ekstrak dari gambar yang telah diambil secara *real-time*.

3.2.2 Proses Noise dan Efek Blur

Proses Penambahan Noise dan Efek Blur dilakukan untuk menguji pengaruh kualitas teknologi kamera yang digunakan saat pengambilan gambar. Hal ini dilakukan dengan mensimulasikan kondisi nyata yang lebih bervariasi dengan cara menambahkan Gaussian Noise dan Gaussian Blur pada citra hasil digitalisasi.

Disini kita mengambil beberapa piksel sampel untuk menunjukkan proses yang akan dilakukan dalam pengolahan citra ini.



Gambar 4. sampling

Gambar 5. Nilai piksel sampling

Setelah kita mendapatkan nilai piksel yang dapat kita gunakan untuk perhitungan kita tambahkan noise Gaussian dengan rumus: $N(\mu = 0, \sigma)$. Dimana mean 0 artinya pusat distribusi ada di nol dan standar deviasi $\sigma = 8$ yang menandakan seberapa menyebar nilai acaknya. Jadi contoh nilai acak yang didapat pada 1 piksel sebagai berikut:

Contoh untuk piksel $P(195,360) = [192, 61, 44]$:

→ noise = [+4, -1, +5],

Maka,

$[192+4 = 196, 61-1=60, 44+5=49]$, Setelah didapat nilai noisenya, nilai noisenya ditambahkan ke pikselnya: hasil = [196, 60, 49]

Noise ditambahkan ke seluruh piksel dengan nilai noise yang berbeda-beda, maka contoh hasilnya sebagai berikut:

(196, 60, 49), (203, 59, 42), (202, 68, 41), (195, 57, 40), (192, 47, 31), (184, 54, 50), (184, 52, 58), (187, 64, 36), (184, 65, 39)

Setelah Noise ditambahkan ke seluruh piksel, kita akan menambahkan Gaussian Blur dengan rumus:

$$I_{\text{blur}}(x,y) = \frac{1}{16} \sum_{i=-1}^1 \sum_{j=-1}^1 I(x+i,y+j) \cdot K(i,j) \text{ dan kernelnya } \begin{bmatrix} 1 & 2 & 1 \\ 1 & 4 & 2 \\ 1 & 2 & 1 \end{bmatrix}$$

$$\text{Blur } R = 1 \div 16 \times (196 \times 1 + 203 \times 2 + 202 \times 1 + 195 \times 2 + 192 \times 4 + 184 \times 2 + 184 \times 1 + 187 \times 2 + 184 \times 1) = 1 \div 16 \times 3065 = 192$$

$$\text{Blur } G = 1 \div 16 \times (60 \times 1 + 59 \times 2 + 68 \times 1 + 57 \times 2 + 47 \times 4 + 54 \times 2 + 52 \times 1 + 64 \times 2 + 65 \times 1) = 1 \div 16 \times 893 = 56$$

$$\text{Blur } B = 1 \div 16 \times (49 \times 1 + 42 \times 2 + 41 \times 1 + 40 \times 2 + 31 \times 4 + 50 \times 2 + 58 \times 1 + 36 \times 2 + 39 \times 1) = 1 \div 16 \times 638 = 40$$

Maka hasil yang didapat adalah $P(196,361) = [192, 56, 40]$. Dimana hasil blur tersebut diletakkan di titik pusat sampelnya. Proses tersebut berlangsung hingga seluruh piksel dijalani dan didapat citra hasil penambahan Noise dan Blur. Hasilnya adalah pada *Gambar 6*.



Gambar 6. setelah diberikan gaussian noise

Dalam sistem ini kita dapat melakukan klasifikasi kematangan tomat dengan melewati proses Noise dan Efek Blur dan tidak melewati Noise dan Efek Blur. Jadi hasil dari proses ini akan dijadikan perbandingan untuk melihat pengaruh antara menggunakan Noise dan Efek Blur dan yang tidak menggunakan Noise dan Efek Blur dengan cara perhitungan akurasi masing - masing kategori.

3.2.3 Proses Deblurring (Penajaman)

Tahap selanjutnya adalah tahap deblurring. Tahap deblurring dilakukan untuk mengurangi efek blur dari citra yang sudah melewati proses Noise dan Efek Blur sebelum-nya maupun tidak melewati proses Noise dan Efek Blur dengan cara menggunakan High Pass Filtering.

Maka disini kita akan menerapkan High Pass Filtering dengan rumus:

$$I_{\text{blur}}(x,y) = \frac{1}{16} \sum_{i=-1}^1 \sum_{j=-1}^1 I(x+i,y+j) \cdot K(i,j) \text{ dan kernelnya } \begin{bmatrix} -1 & -1 & -1 \\ -1 & 9 & -1 \\ -1 & -1 & -1 \end{bmatrix}$$

$$\text{DeBlurring } R = 192 \times (-1) + 191 \times (-1) + 189 \times (-1) + 191 \times (-1) + 190 \times (9) + 188 \times (-1) + 191 \times (-1) + 189 \times (-1) + 188 \times (-1) + (-189) + (-188) = -192 - 191 - 189 - 191 + 1710 - 188 - 191 - 189 - 188 = 191$$

$$\text{DeBlurring } G = 61 \times (-1) + 61 \times (-1) + 62 \times (-1) + 61 \times (-1) + 62 \times (9) + 62 \times (-1) + 63 \times (-1) + 63 \times (-1) + 64 \times (-1) = -60 - 61 - 62 - 61 + 558 - 62 - 63 - 63 - 64 = 61$$

$$\text{DeBlurring } B = 44 \times (-1) + 44 \times (-1) + 45 \times (-1) + 44 \times (-1) + 45 \times (9) + 47 \times (-1) + 46 \times (-1) + 47 \times (-1) + 48 \times (-1) = -44 - 44 - 45 - 44 + 405 - 47 - 46 - 47 - 48 = 40$$

Maka hasil Deblurring adalah: $P(196,361) = [191, 61, 40]$. Kernel Deblurring akan dioperasikan hingga seluruh piksel. Maka didapat citra hasil deblurring pada *Gambar 7*. Deblurring dilakukan agar lebih tajam.



Gambar 7. Hasil Deblurring

3.2.4 Proses Penyesuaian Kontras

Setelah melakukan deblurring gambar diperbaiki kontrasnya agar rentang nilai warna lebih mencolok. Untuk meningkatkan kontrasnya kita menggunakan rumus peningkatan kontras berbasis faktor. Rumusnya sebagai berikut:

$$\text{newValue} = \text{factor} \times (\text{oldValue} - 128) + 128$$

$$\text{factor} = (259 \times (\text{contrast} + 255)) \div (255 \times (259 - \text{contrast}))$$

Nilai contrast yang dipilih adalah 50, karena 50 berada di tengah-tengah range positif agar penguatan kontras tidak terlalu ekstrem.

$$factor = (259 \times (50 + 255)) \div (255 \times (259 - 50)) = \frac{79,045}{53,295} = 1.483$$

Setelah mendapatkan nilai faktor, nilai faktor diaplikasikan ke setiap channel RGB

$$newValue_R = 1.483 \times (191 - 128) + 128 = 222.429 = 221$$

$$newValue_G = 1.483 \times (61 - 128) + 128 = 28.639 = 29$$

$$newValue_B = 1.483 \times (40 - 128) + 128 = -2.504 = 0 \text{ (setelah clipping ke range } 0 - 255)$$

Maka hasil dari proses kontras adalah $P(196,361) = [221, 29, 0]$. Setelah proses kontras dilakukan ke seluruh piksel maka hasilnya adalah pada Gambar 8.



Gambar 8. Hasil Penyesuaian kontras

3.2.5 Proses Edge Enhancement (Peningkatan Tepi)

Setelah proses penyesuaian kontras dilakukan proses Edge Enhancement (Peningkatan Tepi). Proses ini dilakukan untuk memperjelas ujung objek dengan latar belakang objek. Pada tahap ini, metode Sobel digunakan untuk menonjolkan kontur objek tomat pada kanvas, berdasarkan perhitungan gradien X dan Y terhadap intensitas warna grayscale. Adapun rumus kernel untuk Sobel adalah sebagai berikut.

Nilai piksel setelah proses kontras:

$$[220, 28, 0] [219, 28, 0] [218, 29, 1]$$

$$[220, 28, 0] [221, 29, 0] [219, 30, 1]$$

$$[221, 30, 1] [220, 30, 1] [219, 31, 2]$$

Tahap pertama kalikan dan jumlahkan dengan kernel G_x :

$$\begin{bmatrix} -1 & 0 & 1 \\ -2 & 9 & 2 \\ -1 & 0 & 1 \end{bmatrix}$$

$$Gx_R = -220 + 0 + 218 + (-440) + 0 + 438 + (-221) + 0 + 219 = 875 - 881 = -6$$

$$Gx_G = -28 + 0 + 29 + (-56) + 0 + 60 + (-30) + 0 + 31 = 120 - 114 = 6$$

$$Gx_B = 0 + 0 + 1 + 0 + 0 + 2 + (-1) + 0 + 2 = 4$$

Untuk G_y :

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

$$Gy_R = -220 + (-438) + (-218) + 0 + 0 + 0 + 221 + 440 + 219 = 880 - 876 = 4$$

$$Gy_G = -28 + (-56) + (-29) + 0 + 0 + 0 + 30 + 60 + 31 = 121 - 113 = 8$$

$$Gy_B = 0 + 0 + (-1) + 0 + 0 + 0 + 1 + 2 + 2 = 4$$

Lalu tahap kedua adalah mendapatkan nilai besar tepi (magnitude) dengan rumus:

$$G = \sqrt{Gx^2 + Gy^2}$$

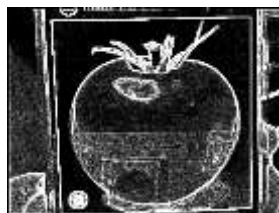
Maka,

$$\begin{aligned} G_R &= \sqrt{Gx_R^2 + Gy_R^2} \\ &= \sqrt{((-6)^2 + (4)^2)} \\ &= 7.21 \approx 7 \end{aligned}$$

$$\begin{aligned} G_B &= \sqrt{Gx_B^2 + Gy_B^2} \\ &= \sqrt{((4)^2 + (4)^2)} \\ &= 5.66 \approx 6 \end{aligned}$$

$$\begin{aligned} G_G &= \sqrt{Gx_G^2 + Gy_G^2} \\ &= \sqrt{((6)^2 + (8)^2)} \\ &= \sqrt{100} = 10 \end{aligned}$$

Maka hasil gradient dari Edge enhancementnya adalah $[7, 10, 6]$. Proses tersebut dijalankan terhadap seluruh piksel yang hasilnya akan seperti Gambar 9



Gambar 9. Hasil Edge Enhancement

3.2.6 Proses Transformasi RGB ke HSV

Setelah proses Edge Enhancement transformasi warna dilakukan dari ruang RGB ke HSV untuk melakukan segmentasi warna tomat. Hanya nilai hue yang divisualisasikan, agar fokus terhadap spektrum warna.

Contoh piksel setelah edge enhancement : $P(196,361) = [227, 37, 5]$

Hal pertama dilakukan adalah menormalisasi nilai RGB dengan nilai desimal 0-1.

Normalisasi:

$$r = 221 \div 255 = 0.8667 \text{ (86.67\% dari intensitas maksimum merah)}$$

$$g = 29 \div 255 = 0.1137 \text{ (11.37\% dari intensitas maksimum hijau)}$$

$$b = 0 \div 255 = 0.0000 \text{ (0\% dari intensitas maksimum biru)}$$

Selanjutnya menentukan nilai Max, Min dan Delta(Δ)

$$\max = \text{Nilai max. dari RGB} = 0.8667 \text{ (nilai } r)$$

$$\min = \text{Nilai min. dari RGB} = 0.0000 \text{ (nilai } b)$$

$$\text{delta}(\Delta) = \max - \min = 0.8667 - 0.0000 = 0.8667$$

V adalah nilai warna yang paling terang. Maka,

$$V = \max = 0.8667$$

S adalah nilai murni warna tersebut. Maka,

$$S = \Delta \div \max$$

$$S = 0.8667 \div 0.8667 = 1.0000$$

H merepresentasikan posisi warna di spektrum (0-360°). Dalam rumusnya menggunakan Modulo 6 dan $\times 60$ karena mengkonversi ke sistem derajat (0-360°). Jadi Hue memiliki kondisi:

Jika $\text{delta} = 0$, maka $H = 0$ (grayscale, tidak ada hue)

Jika $\max = r$, maka $H = ((g - b) \div \text{delta}) \% 6$ [Wilayah Merah – Kuning]

Jika $\max = g$, maka $H = (b - r) \div \text{delta} + 2$ [Wilayah Hijau – Cyan]

Jika $\max = b$, maka $H = (r - g) \div \text{delta} + 4$ [Wilayah Biru – Magenta]

maka,

$$H = ((g - b) \div \text{delta}) \% 6$$

$$H = ((0.1137 - 0.0000) \div 0.8667) \% 6 = 0.1312 \% 6 = 0.1312 \text{ (karena } 0.1312 < 6)$$

Maka H nya adalah 0.1312. Bila dikonversi ke derajat menjadi:

$$H_{\text{derajat}} = H \times 60 = 0.1312 \times 60 = 7.872^\circ$$

Dalam proses ini nilai Hue di normalisasi antara 0-1 agar memudahkan perhitungan pada tahap segmentasi.

$$H_{\text{normalized}} = H_{\text{derajat}} / 360 = 7.872 \div 360 = 0.0219$$

Maka nilai HSV pada piksel tersebut adalah $P(196,361) = [0.0219, 1.0000, 0.8667]$. Tahap ini dilakukan sampai mendapatkan seluruh nilai HSV pada piksel gambar. Maka hasilnya akan seperti Gambar 10.



Gambar 10. Transformasi RGB ke HSV

3.2.7 Proses Segmentasi Berbasis Warna HSV

Setelah gambar di transformasi dari RGB ke HSV proses selanjutnya adalah segmentasi berdasarkan warna HSV. Segmentasi dilakukan berdasarkan rentang warna tomat: merah (matang), jingga/kuning (setengah matang), dan hijau (mentah). Piksel yang tidak termasuk segmentasi akan ditiadakan dalam visualisasinya, sedangkan area yang termasuk akan ditampilkan dan akan dihitung nilai rata-rata RGB-nya sebagai dasar klasifikasi.

HSV hasil sebelumnya:

$$P(196,361) = [0.0219, 1.0000, 0.8667]$$

Piksel di segmentasi dengan syarat:

$$\text{Merah (Matang)} \rightarrow 0 \leq H \leq 0.06 \wedge S > 0.4 \wedge V > 0.2$$

$$\text{Kuning/jingga (Setengah matang)} \rightarrow 0.08 \leq H \leq 0.18 \wedge S > 0.3$$

$$\text{Hijau (Mentah)} \rightarrow 0.22 \leq H \leq 0.45 \wedge S > 0.2 \wedge V > 0.2$$

$$\text{Perbandingan:} \rightarrow h = 0.0240$$

$$- h \geq 0 ? \rightarrow 0.0219 \geq 0 \rightarrow \text{TRUE}$$

$$- h \leq 0.06 ? \rightarrow 0.0219 \leq 0.06 \rightarrow \text{TRUE}$$

$$- s > 0.4 ? \rightarrow 1.0000 > 0.4 \rightarrow \text{TRUE}$$

$$- v > 0.2 ? \rightarrow 0.8667 > 0.2 \rightarrow \text{TRUE}$$

Karena piksel $P(196,361) = [0.0219, 1.0000, 0.8667]$ memenuhi syarat merah matang maka piksel tersebut terdeteksi sebagai tomat bagian merah. Jika tidak memenuhi dari syarat tersebut maka piksel tersebut memiliki nilai 0 yaitu hitam (background). Maka hasil segmentasi keseluruhan akan ditampilkan pada output visual.

Setelah di segmentasi tomat akan diklasifikasi berdasarkan rata-rata piksel tomat yang terdeteksi.

$$1. v < 0.2 \text{ (busuk)} ? \rightarrow 0.8667 < 0.2 \rightarrow \text{FALSE}$$

$$2. h \geq 0 \wedge h \leq 0.06 \wedge v > 0.5 \text{ (matang)} ?$$

$$h \geq 0 \rightarrow 0.0219 \geq 0 \rightarrow \text{TRUE}$$

$$h \leq 0.06 \rightarrow 0.0219 \leq 0.06 \rightarrow \text{TRUE}$$

$$v > 0.5 \rightarrow 0.8667 > 0.5 \rightarrow \text{TRUE} \rightarrow \text{Maka hasilnya: TRUE. Tomat tersebut adalah tomat matang.}$$

Setelah klasifikasi hasil akan ditampilkan dengan gambar yang sudah di segmentasi dan dengan hasil klasifikasi tomat itu sendiri seperti pada *Gambar 11*.



Gambar 11. Hasil segmentasi berdasarkan HSV

3.2.8 Proses Segmentasi Berbasis Warna RGB

Pada proses ini segmentasi dilakukan sama dengan segmentasi berdasarkan HSV. Pada tahap ini kita langsung menggunakan nilai R (Merah), G (Hijau), B (Biru) tanpa transformasi ke HSV. Hal ini dilakukan untuk menjadi perbandingan metode HSV yang digunakan. Tomat akan dianggap "tersegmentasi" jika memenuhi salah satu kondisi ini: Piksel di segmentasi dengan syarat, sama seperti proses segmentasi berbasis HSV:

$$\text{Merah (Matang)} \rightarrow 0 \leq H \leq 0.06 \wedge S > 0.4 \wedge V > 0.2$$

$$\text{Kuning/jingga (Setengah matang)} \rightarrow 0.08 \leq H \leq 0.18 \wedge S > 0.3$$

$$\text{Hijau (Mentah)} \rightarrow 0.22 \leq H \leq 0.45 \wedge S > 0.2 \wedge V > 0.2$$

$$\text{Nilai RGB sebelumnya:} \rightarrow [227, 37, 5]$$

Selanjutnya adalah proses klasifikasinya. Adapun syarat klasifikasi RGB adalah sebagai berikut:

$$\text{Merah (Matang)} \rightarrow R > 100 \wedge G < 80 \wedge B < 80$$

$$\text{Kuning/jingga (Setengah Matang)} \rightarrow R > 130 \wedge G > 110 \wedge B < 60$$

$$\text{Hijau (Mentah)} \rightarrow R < 100 \wedge G > 100 \wedge B < 100$$

Maka,

$$\text{Pengecekan Merah (Matang):} \rightarrow r = 221 > 100 \rightarrow \text{TRUE}, g = 29 < 80 \rightarrow \text{TRUE}, b = 0 < 80 \rightarrow \text{TRUE}, \text{ dengan Hasil: TRUE}$$

$$\text{Pengecekan Kuning/Jingga (Setengah Matang):} \rightarrow r = 221 > 130 \rightarrow \text{TRUE}, g = 29 > 110 \rightarrow \text{FALSE}, b = 0 < 60 \rightarrow \text{TRUE}, \text{ dengan Hasil: FALSE}$$

$$\text{Pengecekan Merah (Matang):} \rightarrow r = 221 < 100 \rightarrow \text{FALSE}, g = 29 > 100 \rightarrow \text{FALSE}, b = 0 < 100 \rightarrow \text{TRUE}, \text{ dengan Hasil: FALSE}$$

Hasil dari klasifikasi tersebut menunjukkan bahwa tomat tersebut berhasil diklasifikasikan. Tomat tersebut diklasifikasikan sebagai tomat matang. Hasilnya dapat dilihat pada *Gambar 12*.



Gambar 12. Hasil segmentasi berdasarkan RGB

3.3 Pembahasan untuk Implementasi dan Klasifikasi

Untuk mengevaluasi performa sistem klasifikasi kematangan tomat, dilakukan penghitungan akurasi berdasarkan hasil pengujian. Pengujian dilakukan dengan melakukan 10 kali pengujian pada setiap kematangan, jarak, dan simulasi noise. Pengujian dilakukan dengan menggunakan dua pendekatan model warna, yaitu HSV dan RGB.

3.3.1 Klasifikasi dengan Simulasi Blur dan Noise

Tabel berikut menampilkan jumlah data uji, jumlah prediksi benar dan salah, serta rata-rata akurasi pada masing-masing kategori kematangan (Matang, Mentah, dan Setengah).

Akurasi merupakan matrik evaluasi yang menunjukkan seberapa banyak prediksi yang benar dibandingkan dengan total keseluruhan data uji. Rumus akurasi yang digunakan adalah sebagai berikut:

$$\text{Akurasi (\%)} = \left(\frac{\text{Jumlah Prediksi Benar}}{\text{Jumlah Total Uji}} \right) \times 100 \quad (11)$$

a. Metode HSV

Pengujian dilakukan sebanyak 50 kali untuk setiap tingkat kematangan tomat, yaitu matang, mentah, dan setengah matang. Pengujian ini dilakukan menggunakan metode HSV dengan proses blur dan noise. Metode HSV dengan

proses blur dan noise ini diuji terhadap total 150 sampel, dan hasil klasifikasi dicatat dalam bentuk jumlah prediksi yang benar, salah, serta rata-rata akurasi per kelas kematangan. Akurasi dihitung berdasarkan perbandingan antara jumlah prediksi yang benar dengan total data uji menggunakan rumus yang ditampilkan sebelumnya.

Tabel 2. Hasil Klasifikasi dengan Blur dan Noise (HSV)

Kematangan	Total Uji	Benar	Salah	Rata-rata Akurasi (%)
Matang	50	10	40	20.0
Mentah	50	15	35	30.0
Setengah	50	50	0	100.0
Total	150	75	75	50.0

b. Metode RGB

Pengujian dilakukan sebanyak 50 kali untuk setiap tingkat kematangan tomat, yaitu matang, mentah, dan setengah matang. Pengujian ini dilakukan menggunakan metode RGB dengan proses blur dan noise. Metode RGB dengan proses blur dan noise ini diuji terhadap total 150 sampel, dan hasil klasifikasi dicatat dalam bentuk jumlah prediksi yang benar, salah, serta rata-rata akurasi per kelas kematangan. Akurasi dihitung berdasarkan perbandingan antara jumlah prediksi yang benar dengan total data uji menggunakan rumus yang ditampilkan sebelumnya.

Tabel 3. Hasil Klasifikasi dengan Blur dan Noise (RGB)

Kematangan	Total Uji	Benar	Salah	Rata-rata Akurasi (%)
Matang	50	50	0	100.0
Mentah	50	10	40	20.0
Setengah	50	50	0	100.0
Total	150	110	40	73.3

$$\text{Akurasi Gabungan} = \left(\frac{185}{300} \right) \times 100 = 61.67 \%$$

Pada simulasi blur dan noise, metode RGB menunjukkan performa klasifikasi yang lebih baik dibandingkan HSV. RGB berhasil mencapai akurasi 100% pada kategori Matang dan Setengah, sedangkan HSV hanya akurat pada kategori Setengah. Akurasi rata-rata untuk semua data uji dengan gangguan visual adalah 61.7%.

c. Hasil Klasifikasi Tanpa Simulasi Noise dan Blur

Pengujian ini dilakukan berdasarkan hasil klasifikasi tanpa adanya simulasi gangguan visual seperti blur dan noise untuk mengetahui sejauh mana sistem mampu mengenali tingkat kematangan buah tomat dalam kondisi citra yang bersih. Pengujian dilakukan dengan menggunakan dua pendekatan model warna, yaitu HSV dan RGB. Tabel berikut menampilkan jumlah data uji, jumlah prediksi benar dan salah, serta rata-rata akurasi pada masing-masing kategori kematangan (Matang, Mentah, dan Setengah). Evaluasi dilakukan untuk membandingkan konsistensi performa model terhadap data yang tidak mengalami gangguan visual.

d. Metode HSV

Pengujian dilakukan sebanyak 50 kali untuk setiap tingkat kematangan tomat, yaitu matang, mentah, dan setengah matang. Pengujian ini dilakukan menggunakan metode HSV tanpa proses blur dan noise. Metode HSV tanpa proses blur dan noise ini diuji terhadap total 150 sampel, dan hasil klasifikasi dicatat dalam bentuk jumlah prediksi yang benar, salah, serta rata-rata akurasi per kelas kematangan. Akurasi dihitung berdasarkan perbandingan antara jumlah prediksi yang benar dengan total data uji menggunakan rumus yang ditampilkan sebelumnya.

Tabel 4. Hasil Klasifikasi Tanpa Gangguan (HSV)

Kematangan	Total Uji	Benar	Salah	Rata-rata Akurasi (%)
Matang	50	15	35	30.0
Mentah	50	17	33	34.0
Setengah	50	50	0	100.0
Total	150	82	68	54.7

e. Metode RGB

Pengujian dilakukan sebanyak 50 kali untuk setiap tingkat kematangan tomat, yaitu matang, mentah, dan setengah matang. Pengujian ini dilakukan menggunakan metode RGB tanpa proses blur dan noise. Metode RGB tanpa proses blur dan noise ini diuji terhadap total 150 sampel, dan hasil klasifikasi dicatat dalam bentuk jumlah prediksi yang benar, salah, serta rata-rata akurasi per kelas kematangan. Akurasi dihitung berdasarkan perbandingan antara jumlah prediksi yang benar dengan total data uji menggunakan rumus yang ditampilkan sebelumnya.

Tabel 5. Hasil Klasifikasi Tanpa Gangguan (RGB)

Kematangan	Total Uji	Benar	Salah	Rata-rata Akurasi (%)
Matang	50	48	2	96.0
Mentah	50	20	30	40.0

Setengah	50	40	10	80.0
Total	150	110	40	72.0

$$\text{Akurasi Gabungan} = \left(\frac{192}{300} \right) \times 100 = 64.0 \%$$

Tanpa adanya gangguan seperti blur atau noise, klasifikasi meningkat sedikit terutama pada metode RGB. RGB menunjukkan performa konsisten tinggi terutama pada kategori Matang dan Setengah. Namun, klasifikasi kategori Mentah masih menjadi tantangan. Rata-rata akurasi secara keseluruhan adalah 64.0%.

f. Pengaruh Jarak Terhadap Hasil Klasifikasi

Selanjutnya dilakukan pengujian pengaruh jarak pengambilan gambar terhadap hasil klasifikasi kematangan buah tomat. Uji dilakukan pada lima variasi jarak yaitu 5 cm, 10 cm, 20 cm, 30 cm, dan 50 cm, baik dengan kondisi citra yang mengandung blur maupun tanpa blur. Pengamatan dilakukan terhadap performa klasifikasi dengan menggunakan metode HSV dan RGB. Tabel berikut menunjukkan hasil akurasi klasifikasi berdasarkan masing-masing metode dan kondisi blur untuk tiap jarak, serta dihitung rata-rata akurasi keseluruhan. Pengujian ini bertujuan untuk mengetahui seberapa sensitif sistem klasifikasi terhadap perubahan jarak kamera dalam pengambilan citra objek.

Tabel 6. Pengaruh Jarak terhadap Akurasi Klasifikasi

Jarak (cm)	HSV Blur (%)	RGB Blur (%)	HSV No Blur (%)	RGB No Blur (%)	Rata-rata Semua (%)
5	100.0	100.0	100.0	100.0	100.0
10	50.0	66.7	60.0	83.3	65.0
20	33.3	66.7	46.7	76.7	55.8
30	33.3	66.7	33.3	66.7	50.0
50	33.3	66.7	33.3	33.3	41.7

Jumlah seluruh nilai akurasi

Baris 1: $100 + 100 + 100 + 100 = 400.0$

Baris 2: $50 + 66.7 + 60 + 83.3 = 260.0$

Baris 3: $33.3 + 66.7 + 46.7 + 76.7 = 223.4$

Baris 4: $33.3 + 66.7 + 33.3 + 66.7 = 200.0$

Baris 5: $33.3 + 66.7 + 33.3 + 33.3 = 166.6$

Total Akurasi = $400.0 + 260.0 + 223.4 + 200.0 + 166.6 = 1250.0$

Rata - rata Akurasi Semua Data = $\left(\frac{1250.0}{20} \right) = 62.5 \%$

Berdasarkan hasil pengujian pada berbagai jarak pengambilan gambar, dapat disimpulkan bahwa jarak memiliki pengaruh signifikan terhadap akurasi klasifikasi tingkat kematangan buah. Jarak terdekat, yaitu 5 cm, menunjukkan hasil terbaik dengan akurasi sempurna (100%) untuk seluruh metode dan kondisi, baik HSV maupun RGB, dengan atau tanpa blur. Namun, seiring bertambahnya jarak akurasi cenderung menurun secara bertahap. Penurunan paling tajam terjadi pada jarak 30 cm dan 50 cm, terutama pada metode HSV yang sangat sensitif terhadap detail visual. Metode RGB cenderung lebih stabil pada jarak menengah, tetapi tetap mengalami penurunan performa pada jarak yang lebih jauh.

Akurasi rata-rata dari seluruh kombinasi metode dan jarak adalah 62.5%, yang mengindikasikan bahwa untuk mendapatkan hasil klasifikasi yang optimal, pengambilan citra sebaiknya dilakukan pada jarak dekat. Hal ini disebabkan oleh penurunan tingkat warna objek yang dituju, terutama jika disertai gangguan seperti blur.

4. KESIMPULAN

Adapun kesimpulan yang dihasilkan dari penelitian yang dilakukan dengan metode yang berikan adalah Klasifikasi tomat memiliki perubahan akurasi berdasarkan jarak, Jarak akan mempengaruhi kualitas dari metode deblurring, deteksi tepi, dan segmentasi. Semakin besar jarak antara kamera dengan tomat maka warna akan lebih sulit dideteksi. Analisis kematangan tomat lebih akurat ketika dianalisis menggunakan RGB dibandingkan HSV dengan Hasil tes tanpa menggunakan noise dan blur HSV memiliki akurasi 54.7% dan RGB memiliki akurasi 72%. Keterbatasan dalam klasifikasi warna dengan kondisi lingkungan yang berbeda sehingga diperlukan metode pengolahan citra digital lebih lanjut dalam penyesuaian warna pada lingkungan sekitar. Serta segmentasi terhadap tomat belum mencapai bentuk tomat yang sempurna sehingga dapat dikembangkan lebih lanjut dalam penerapan teknologi yang lebih kompleks.

UCAPAN TERIMA KASIH

Terimakasih di ucapkan kepada kedua orang tua serta keluarga yang telah memberikan motivasi, do'a, dan dukungan moral maupun materil. Serta kepada mahasiswa stambuk 2022 prodi S1 ilmu Komputer diantaranya, Fathya Azra Nazmira (221401039), Alvaro Hutabarat (221401084), Khaisa Shafa Dwinanda S (221401001), Zihan Syakilla (221401133) dan Raffael Remco Marchelino Sumbayak (221401121) yang sudah membantu dalam penelitian ini, serta seluruh pihak yang tidak dapat disebutkan satu persatu. Kiranya jurnal ini bisa memberi manfaat bagi pembaca dan dapat meningkatkan kualitas jurnal selanjutnya

DAFTAR PUSTAKA

- [1] A. W. Damar, D. P. Almeyda, I. M. M. Putra, and L. Maliahatuningrum, "Analisis Perbandingan Metode Pra Pemrosesan Citra untuk Deteksi Tepi Canny pada Citra Berbagai Kondisi Jalan menggunakan Bahasa Pemrograman Python," *Jurnal Teknologi dan Ilmu Komputer Prima (JUTIKOM)*, vol. 7, no. 1, pp. 24–33, Apr. 2024.
- [2] S. Aras, P. Tanra, dan M. Bazhar, "Detection of Tomato Fruit Ripeness Level Using YOLOv5," *MALCOM: Indonesian Journal of Machine Learning and Computer Science*, vol. 4, no. 2, pp. 623–628, Apr. 2024. <https://doi.org/10.57152/malcom.v4i2.1270>.
- [3] H. Firdaus, F. Farizi, R. A. Syakur, A. T. Ramadhan, dan P. Rosyani, "Deteksi Tingkat Kematangan Buah Tomat Menggunakan Pengolahan Citra Dengan OpenCV dan Python," *BIN: Bulletin of Informatics*, vol. 2, no. 1, pp. 130–136, Jul. 2024. <https://ojs.jurnalmahasiswa.com/ojs/index.php/bin/article/view/342/105>.
- [4] N. I. Humaira, R. B. Andriani, and D. D. Pranowo, "Klasifikasi Tingkat Kualitas dan Kematangan Buah Tomat Menggunakan Metode Jaringan Syaraf Tiruan (JST)," *Jurnal Teknik Informatika dan Sistem Informasi*, vol. 7, no. 1, pp. 45–53, 2021.
- [5] M. Harim, H. A. T. Murni, dan R. A. Saputra, "Segmentasi Citra Telapak Tangan Menggunakan Deteksi Tepi Prewitt, Sobel, Roberts, dan Canny," *Jurnal Informatika Merdeka Pasuruan (JIMP)*, vol. 8, no. 1, pp. 1–7, 2023.
- [6] S. Abdullah, W. A. Kusuma, dan S. H. Wijaya, "Prediksi protein-protein interaction berbasis sekuens protein menggunakan fitur autocorrelation dan machine learning," *Jurnal Teknologi dan Sistem Komputer*, vol. 10, no. 1, pp. 1–11, 2022. doi: 10.14710/jtsiskom.2022.13984, <https://jtsiskom.undip.ac.id>.
- [7] A.C.Bergstrom, D.Conran, dan D.W.Messinger, "Gaussian Blur and Relative Edge Response," arXiv preprint, arXiv:2301.00856, Jan.2023. doi:10.48550/arXiv.2301.00856.
- [8] A. Arnita, F. Marpaung, F. Aulia, N. Suryani, dan R. C. Nabila, *Computer Vision dan Pengolahan Citra Digital*, A. B. Surya, Ed. Surabaya, Indonesia: Pustaka Aksara, 2022. ISBN: 978-623-8230-27-3.
- [9] K. A. N., *Ekstraksi Warna Objek menggunakan Segmentasi Citra dan HSV*, Institut Teknologi Bandung, 2019. <https://informatika.stei.itb.ac.id/~rinaldi.munir/Citra/2019-2020/Makalah2019/13516108.pdf>.
- [10] A. R. Smith, "Color gamut transform pairs," *ACM SIGGRAPH Computer Graphics*, vol. 12, no. 3, pp. 12–19, Aug. 1978. <https://dl.acm.org/doi/10.1145/965139.807361>.
- [11] B. K. Shah, V. Kedia, R. Raut, S. Ansari, dan A. Shroff, "Evaluation and Comparative Study of Edge Detection Techniques," *IOSR Journal of Computer Engineering (IOSR-JCE)*, vol. 22, no. 5, ser. III, pp. 06–15, Sep.–Oct. 2020.
- [12] I. Sobel, "History and Definition of Sobel Operator," Scribd, 2014. <https://id.scribd.com/document/271811982/History-and-Definition-of-Sobel-Operator>.
- [13] M. Oni, B. Kanata, dan D. Ratnasari, "Menentukan Luas Objek Citra dengan Teknik Segmentasi Berdasarkan Warna pada Ruang Warna HSV," *Dielektrika*, vol. 8, no. 2, pp. 137–146, Agustus 2021.
- [14] A. Rosenfeld and A. C. Kak, *Digital Picture Processing*, illustrated ed. New York, NY, USA: Academic Press, 1976.
- [15] N. I. Humaira, M. Herman, N. Nurhikma, dan A. B. Kaswar, "Klasifikasi Tingkat Kualitas dan Kematangan Buah Tomat Berdasarkan Fitur Warna Menggunakan Jaringan Syaraf Tiruan," *Jurnal Nasional Teknik Elektro dan Teknologi Informasi (JNTETI)*, vol. 10, no. 2, pp. 79–86, Mei 2021. [Online]. Vol02-1-20329-49809-1-SM.
- [16] H. Wijaya, A. Marzuki, and S. Sitanggang, "Identifikasi Karakteristik Citra Berdasarkan pada Nilai Entropi dan Kontras" *Jurnal Ilmiah Teknik Informatika*, vol. 6, no. 2, pp. 79–88, Jul. 2021.
- [17] G. D. Cahyo, N. A. Thobroni, T. P. Gusti, and R. Wulaningrum, "Penggunaan OpenCV untuk Segmentasi Cabai menggunakan Konversi RGB ke HSV," *Seminar Nasional Teknologi & Sains (STAINS)*, vol. 4, no. 1, pp. 1–10, Jan. 2025, doi: 10.29407/cgkfp.10.
- [18] M. R. Qisthiano dan A. O. Pratiwi, "Deteksi Tepi pada Citra Objek Benda Menggunakan Algoritma Sobel dan Prewitt dengan Python," *Jurnal Informatika dan Teknik Elektro Terapan (JITET)*, vol. 13, no. 2, pp. 45–52, 2025. [Online]. <https://doi.org/10.23960/jitet.v13i2.6407>.
- [19] A. F. Hastawan, R. Septiana, dan Y. E. Windarto, "Perbaikan Hasil Segmentasi HSV pada Citra Digital Menggunakan Metode Segmentasi RGB Grayscale," *Edu Komputika Journal*, vol. 6, no. 1, pp. 1–8, 2019.
- [20] R. S. Firzanah, N. Z. Anugrani, Samsidar, I. A. Khalil, dan T. A. Ayyubi, "Implementasi Metode Thresholding untuk Segmentasi Citra Digital Kupu-Kupu dalam Lingkungan Pengembangan Visual Studio Code," *MAPLE: Mechatronics Journal in Professional and Entrepreneur*, vol. 6, no. 1, Politeknik Bosowa, Juni 2024.
- [21] H. Pangaribuan dan S. Sitohang, "Peningkatan Kualitas Deteksi Tepi dengan Metode Segmentasi Citra," *Remik: Riset dan E-Jurnal Manajemen Informatika Komputer*, vol. 7, no. 1, pp. 1–8, Jan. 2023. doi: 10.33395/remik.v7i1.12050.
- [22] A. K. Panggabean, A. Syahfaridzah, dan N. A. Ardiningih, "Mendeteksi Objek Berdasarkan Warna dengan Segmentasi Warna HSV Menggunakan Aplikasi MATLAB," *METHOMIKA: Jurnal Manajemen Informatika & Komputerisasi Akuntansi*, vol. 4, no. 2, pp. 94–97, Okt. 2020. doi: 10.46880/jmika.Vol4No2.pp94-97.
- [23] N. K. Ibrahim, "Review on Region-Based Segmentation Using Watershed and Region Growing Techniques and their Applications in Different Fields," *Journal La Multiapp*, vol. 3, no. 5, pp. 241–249, 2022. doi: 10.37899/journallamultiapp.v3i5.714.
- [24] S. T. Damanik, A. Setiawan, T. Simanjuntak, A. A. Tumanggor, dan F. Ramadhani, "Analisis Deteksi Tepi untuk Mengidentifikasi Jenis Mobil," *Explore: Jurnal Sistem Informasi dan Telematika (Telekomunikasi, Multimedia dan Informatika)*, vol. 14, no. 1, pp. 1–6, 2023. <https://jurnal.ubl.ac.id/index.php/explore/article/view/3082>.
- [25] F. M. Rafi, "Segmentasi Jenis Kawasan pada Citra Satelit Daerah dengan Split and Merge berdasarkan Kluster Warna Piksel," *Makalah IF4073 Interpretasi dan Pengolahan Citra*, Sekolah Teknik Elektro dan Informatika, Institut Teknologi Bandung, Mei 2022. [https://informatika.stei.itb.ac.id/~rinaldi.munir/Citra/2021-2022/Makalah/Makalah_IF4073_Citra_2022%20\(7\).pdf](https://informatika.stei.itb.ac.id/~rinaldi.munir/Citra/2021-2022/Makalah/Makalah_IF4073_Citra_2022%20(7).pdf)