SISTEM DETEKSI WARNA OBJEK BERBASIS OPENCV PYTHON

Rizko Al Amin[1]*, Rizky Nurman Aktan[2]
[1,2] Teknologi Informasi, Universitas Nurdin Hamzah
email: rizkoalamin56@gmail.com, nurmanaktanrizky@gmail.com

Abstract – Color detection is one of several techniques used for object tracking. It can also function as an object classifier in robotics and other significant projects. This paper discusses the detection of primary colors using the built-in webcam on a laptop. The initialization of the three HSV variables is crucial, as it allows for the precise definition of each object's color. The experiment showed that some unintended colors were detected due to the real-time nature of the system. To improve accuracy, incorporating machine learning or stabilization features could enhance the system's performance.

Keywords — Object Detection, Rgb Color, HSV, Real-time;

Abstrak— Deteksi warna adalah salah satu teknik yang digunakan untuk melacak objek dan mengklasifikasikan benda dalam bidang robotika serta berbagai aplikasi lainnya. Penelitian ini membahas identifikasi warna primer menggunakan webcam bawaan pada laptop. Proses inisialisasi variabel HSV sangat penting untuk menetapkan warna yang ingin dideteksi. Dengan mengacu pada variabel warna tersebut, objek dapat dikenali dan dianalisis lebih lanjut. Hasil eksperimen menunjukkan bahwa beberapa objek yang bukan target juga terdeteksi karena sistem bekerja secara real-time. Oleh karena itu, diperlukan penambahan fitur machine learning untuk menstabilkan data yang diterima dari kamera.

Kata Kunci—Deteksi object, Warna Rgb, HSV, Real-time

1.PENDAHULUAN

Pelacakan objek adalah komponen utama dalam visualisasi komputer yang memberikan informasi berharga untuk memahami data dalam bentuk gambar atau video, serta berhubungan dengan berbagai aplikasi [1][2]. Salah satu cara untuk melacak objek adalah dengan menggunakan deteksi warna RGB sebagai dasar dalam mengenali objek spesifik. RGB dipilih karena merupakan warna dasar yang sering kali digunakan untuk mewakili warna objek [3]. Warna itu sendiri adalah spektrum tertentu yang terdapat dalam cahaya putih sempurna. Nilai warna ditentukan oleh kecerahan atau kepekatan warna tersebut, yang dipengaruhi oleh penambahan putih atau hitam. Penelitian menunjukkan bahwa kombinasi warna yang menghasilkan rentang yang paling luas adalah merah (R), hijau (G), dan biru (B), yang dikenal sebagai warna primer dalam model RGB. Dengan mencampurkan ketiga warna dasar ini dalam perbandingan tertentu, warna lain dapat tercipta. Setiap warna dasar memiliki tingkat intensitas yang berbeda. Dalam pemrosesan citra, beberapa metode digunakan, salah satunya adalah model HSV. Model HSV terdiri dari tiga komponen: hue, saturation, dan value, yang memungkinkan deteksi objek berdasarkan warna tertentu, sekaligus mengurangi pengaruh cahaya luar dengan mengenali hanya enam warna utama: coklat, kuning, hijau, biru, hitam, dan putih.

Selain itu, metode pengenalan citra lainnya diterapkan untuk deteksi wajah, namun prosesnya tidak dilakukan secara real-time. Sebaliknya, gambar harus diproses terlebih dahulu dengan dimasukkan ke dalam komputer desktop untuk analisis lebih lanjut.

II TINJAUAN PUSTAKAN DAN METODE PENELITIAN

2.1 Tinjauan Pustaka

2.1.1 Warna RGB

Seperti yang telah dijelaskan sebelumnya, RGB adalah model warna yang terdiri dari tiga warna dasar: Merah (Red), Hijau (Green), dan Biru (Blue) [4]. Ketiga warna ini dikenal sebagai warna primer, yang jika digabungkan akan menghasilkan warna lain. Model RGB ini merepresentasikan cara manusia melihat warna. Oleh karena itu, objek dapat didefinisikan berdasarkan persentase kandungan ketiga warna primer tersebut [3]. Dalam konteks komputasi, deklarasi warna ini biasanya ditulis dalam bentuk array (merah, hijau, biru), dengan nilai yang berada dalam rentang 0 hingga 255.

a. Warna HSV

HSV terdiri dari tiga komponen utama, yaitu Hue (warna dominan yang terlihat oleh mata manusia), Saturation (tingkat keberadaan warna putih dalam Hue), dan Value (nilai keterangan atau intensitas warna) [5]. Sama halnya dengan RGB, ketiga komponen dalam model HSV ini juga dapat dinyatakan dalam bentuk array, dengan nilai masing-masing berada dalam rentang 0 hingga 255.

b. Webcam

Webcam adalah perangkat berbentuk kamera yang digunakan untuk menangkap gambar maupun video, dan dapat diaplikasikan dalam pemrosesan secara real-time [6]. Pengambilan video dilakukan dengan menggunakan webcam bawaan yang terdapat pada laptop pengguna.

c. Python

Python adalah bahasa pemrograman yang relatif baru dan dirancang untuk memudahkan pembuatan program secara cepat dan efisien. Setiap program yang dibuat umumnya memerlukan input dan menghasilkan output. Meskipun penginputan di Python terlihat mudah, hal ini tetap memerlukan pemahaman yang baik, karena banyak pemula yang menghadapi kesulitan dalam menulis program dengan bahasa ini. Penjelasan ini diharapkan dapat membantu para pemula yang tengah belajar bahasa pemrograman

2.2 Metode Perancangan Sistem

Pada pengembangan sistem ini, metode yang dilakukan menggunakan beberapa 6 (enam) tahapan, yaitu **tahap pertama** proses akuisisi citra, proses ini menggunakan kamera untuk menangkap gambar atau video. **Proses kedua** adalah preprocessing, proses ini adalah proses konversi citra ke format HSV dan penerapan filter untuk mengurangi noise. **Proses ketiga** yaitu segmentasi warna, yaitu proses menggunakan metode thresholding berdasarkan rentang warna HSV. **Proses keempat** yaitu deteksi

kontur: Identifikasi bentuk objek yang sesuai dengan warna target. **Proses kelima** yaitu proses ekstraksi Fitur. Proses Menentukan koordinat dan ukuran objek yang terdeteksi. **Proses keenam** adalah visualisasi Hasil. Proses ini menampilkan hasil deteksi secara real-time. Untuk lebih jelasnya, metode perancangan sistem terdapat pada Gambar 1 berikut.

Gambar1. Tahapan Peracangan sistem

a. Akuisisi Citra

Proses ini adalah langkah awal dalam pengolahan citra, di mana gambar atau video diambil menggunakan perangkat seperti kamera atau sensor citra. Data yang dihasilkan adalah citra digital yang akan digunakan untuk analisis lebih lanjut. Citra ini bisa berupa gambar statis atau rangkaian gambar (video) yang perlu diproses lebih lanjut.

b. Preprocessing

Preprocessing adalah langkah penting untuk meningkatkan kualitas citra agar lebih mudah dianalisis. Dalam hal ini, konversi citra ke format HSV (Hue, Saturation, Value) digunakan untuk mengubah citra dari format RGB (Red, Green, Blue) menjadi ruang warna yang lebih sesuai untuk analisis berbasis warna. Hue (H) mengukur warna, Saturation (S) mengukur kejenuhan warna, Value (V) mengukur kecerahan warna. Proses ini membantu mengurangi sensitivitas terhadap pencahayaan, karena model HSV lebih dekat dengan cara manusia melihat warna. Setelah itu, filter seperti Gaussian Blur dapat diterapkan untuk mengurangi noise (gangguan pada citra), sehingga hasil deteksi objek lebih akurat.

c. Segmentasi Warna

Segmentasi warna menggunakan metode thresholding, yang artinya kita menentukan batas-batas nilai warna tertentu dalam ruang warna HSV. Dengan cara ini, objek dengan warna yang mirip dapat dipisahkan dari latar belakang atau objek lain berdasarkan rentang warna tertentu. Misalnya, jika kita ingin mendeteksi objek berwarna merah, kita akan menentukan rentang nilai Hue yang sesuai dengan warna merah, dan memisahkan piksel-piksel yang berada dalam rentang tersebut dari piksel lainnya.

d. Deteksi Kontur

Deteksi kontur adalah proses untuk menemukan bentuk objek dalam citra berdasarkan warna atau intensitas tertentu. Setelah segmentasi warna, citra akan berisi area-area yang diidentifikasi sebagai objek. Teknik seperti thresholding atau edge detection digunakan untuk menemukan batas objek tersebut, yang sering disebut sebagai "kontur." Kontur ini menunjukkan bentuk dan ukuran objek dalam citra.

e. Ekstraksi Fitur

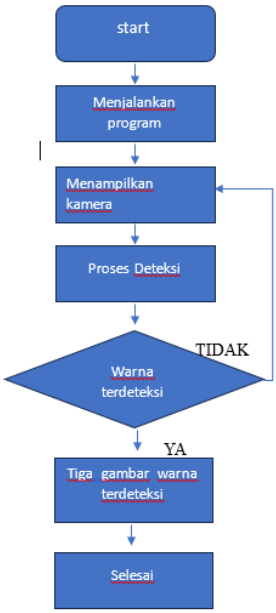
Pada tahap ini, informasi lebih lanjut tentang objek yang terdeteksi diambil, seperti koordinat posisi (misalnya titik tengah objek) dan ukuran objek (misalnya panjang, lebar, atau luas). Fitur-fitur ini digunakan untuk analisis lebih lanjut, misalnya untuk pelacakan objek atau untuk pengenalan pola. Fitur ekstraksi ini juga berguna untuk memahami orientasi dan bentuk objek yang terdeteksi.

f. Visualisasi Hasil

Setelah semua tahap pemrosesan selesai, hasil deteksi objek biasanya divisualisasikan dengan menggambar batas objek atau kontur di atas citra asli. Ini memungkinkan kita untuk melihat objek yang terdeteksi dalam konteks citra asli dan menganalisis hasil deteksi secara real-time. Biasanya, visualisasi ini dilakukan dengan menggunakan warna berbeda atau penanda lain pada objek yang terdeteksi. Proses ini umum digunakan dalam berbagai aplikasi, seperti pemrograman komputer untuk pengenalan objek, pelacakan objek dalam video, atau sistem penglihatan mesin.

2.3. Flowchart

Flowchart digunakan untuk proses menjalankan aplikasi yang sudah dirancang. Untuk flowchart dapat dilihat pada Gambar 2 berikut.



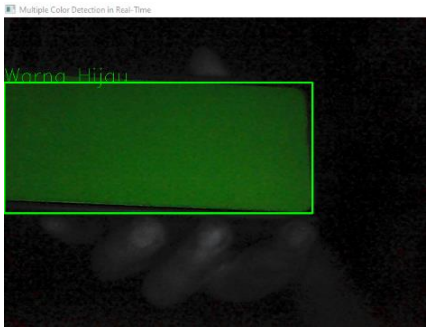
Gambar 2. Tahapan Peracangan sistem

III. Hasil dan Pembahasan

Pengujian dilakukan dengan memberikan 3 objek warna yang berbeda yaitu merah, hijau, biru dan warna selain 3 itu tadi tidak terdeteksi contohnya dinding di belakang. Ada beberapa uji coba sistem dalam menjalankan sistem, yaitu:

3.1 Uji coba Warna pertama

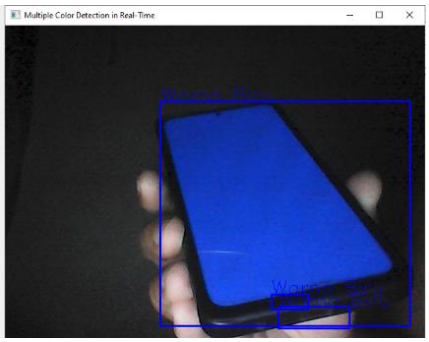
Sebelum melakukan pengujian deteksi gambar dengan kita warna, terlebih dahulu uji gambar warna pertama



Gambar 3. Pengujian warna hijau

Pengujian warna hijau Tunggal, video system realtime mampu mendeteksi warna hijau dengan sukses ,tanpa adanya kesalahan walaupun di belakang cenderung berwarna gelap

3.1.2 Uji warna Tunggal kedua

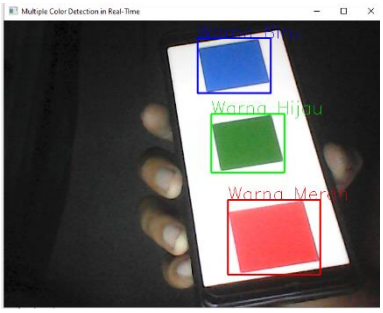


Gambar 4. Pengujian Warna Biru

Pengujian warna biru , sedikit berbeda ada 2 warna biru yang terdeteksi .cenderung ke warna gelap sehingga warna hitam terdeteksi menjadii warna biru

3.1.3. Mendeteksi 3 warna

Percobaan menjalankan 3 warna sekaligus, warna terdeteksi dengan baik dengan tambahan non objek yang terdeteksi di karenakan dengan Cahaya yang gelap di belakang, hasil deteksi bisa di liat di gambar bawah.



Gambar 5.Mendeteksi 3 warna

Pada hasil terakhir, system berhasil mendeteksi 3 warna yaitu Biru, Merah, Hijau, selain itu text pada warna gambar tersebut juga berada pada warnanya masing masing dan tanpa adanya melenceng.

3.1.4. Hasil Pengujian Sistem

Berikut adalah hasil pengujian yang dilakukan terhadap berbagai warna dalam kondisi pencahayaan yang berbeda

Tabel 1. Data hasil pengujian sistem

Warna objek	Kondisi Pencahayaan	Jumlah Objek	Objek Terdeteksi	Akurasi
Hijau	Terang	1	1	100%
Biru	Terang	1	1	100%
Merah	Terang	1	1	100%

Dari hasil pengujian, dapat disimpulkan bahwa kondisi pencahayaan sangat mempengaruhi akurasi deteksi warna. Sistem bekerja lebih baik dalam kondisi terang dibandingkan kondisi redup. Hal ini terjadi karena intensitas cahaya yang lebih tinggi membuat warna lebih kontras dengan latar belakang, sehingga lebih mudah dideteksi oleh metode segmentasi warna yang digunakan. Pada objek berwarna merah, biru, dan hijau sistem menunjukkan akurasi akurat 100 % karena kondisi lebih teran

3.1.4 Source Code

```
import numpy as np
import cv2
webcam = cv2.VideoCapture(0)
while(1):
    _, imageFrame = webcam.read()
    hsvFrame = cv2.cvtColor(imageFrame, cv2.COLOR_BGR2HSV)

    # define mask
    Merah_lower = np.array([136, 87,111], np.uint8)
    Merah_upper = np.array([180, 255, 255], np.uint8)
    Merah_mask = cv2.inRange(hsvFrame, Merah_lower, Merah_upper)

    # define mask
    Hijau_lower = np.array([25, 52, 72], np.uint8)
    Hijau_upper = np.array([102, 255, 255], np.uint8)
    Hijau_mask = cv2.inRange(hsvFrame, Hijau_lower, Hijau_upper)

    # define mask
    Biru_lower = np.array([94, 80, 2], np.uint8)
    Biru_upper = np.array([120, 255, 255], np.uint8)
    Biru_mask = cv2.inRange(hsvFrame, Biru_lower, Biru_upper)

    kernal = np.ones((5, 5), "uint8")

    # For Merah color
    Merah_mask = cv2.dilate(Merah_mask, kernal)
    res_Merah = cv2.bitwise_and(imageFrame, imageFrame,
                                mask = Merah_mask)

    # For Hijau color
    Hijau_mask = cv2.dilate(Hijau_mask, kernal)
    res_Hijau = cv2.bitwise_and(imageFrame, imageFrame,
                                mask = Hijau_mask)

    # For Biru color
    Biru_mask = cv2.dilate(Biru_mask, kernal)
    res_Biru = cv2.bitwise_and(imageFrame, imageFrame,
                               mask = Biru_mask)
```

Gambar 1. Program Pertama

3.1.4 Source Kode Kedua

```
# Creating contour to track Merah color

contours, hierarchy = cv2.findContours(Merah_mask,
                                       cv2.RETR_TREE,
                                       cv2.CHAIN_APPROX_SIMPLE)
for pic, contour in enumerate(contours):
    area = cv2.contourArea(contour)
    if(area > 300):
        x, y, w, h = cv2.boundingRect(contour)
        imageFrame = cv2.rectangle(imageFrame, (x, y),
                                   (x + w, y + h),
                                   (0 , 0, 225), 2)
        cv2.putText(imageFrame, "Warna Merah", (x, y),
                    cv2.FONT_HERSHEY_SIMPLEX, 1.0,
                    (0, 0, 225))

#Creating contour to track Hijau color

contours, hierarchy = cv2.findContours(Hijau_mask,
                                       cv2.RETR_TREE,
                                       cv2.CHAIN_APPROX_SIMPLE)
for pic, contour in enumerate(contours):
    area = cv2.contourArea(contour)
    if(area > 300):
        x, y, w, h = cv2.boundingRect(contour)
        imageFrame = cv2.rectangle(imageFrame, (x, y),
                                   (x + w, y + h),
                                   (0, 255, 0), 2)
        cv2.putText(imageFrame, "Warna Hijau", (x, y),
                    cv2.FONT_HERSHEY_SIMPLEX,
                    1.0, (0, 225, 0))

# Creating contour to track Biru color

contours, hierarchy = cv2.findContours(Biru_mask,
                                       cv2.RETR_TREE,
                                       cv2.CHAIN_APPROX_SIMPLE)
for pic, contour in enumerate(contours):
    area = cv2.contourArea(contour)
    if(area > 300):
```

Gambar 2. Program Kedua

3.1.5 Source Kode Kedua

```
        x, y, w, h = cv2.boundingRect(contour)
        imageFrame = cv2.rectangle(imageFrame, (x, y),
                                   (x + w, y + h),
                                   (255, 0, 0), 2)
        cv2.putText(imageFrame, "Warna Biru", (x, y),
                    cv2.FONT_HERSHEY_SIMPLEX,
                    1.0, (255, 0, 0))

# Program Termination
cv2.imshow("Multiple Color Detection in Real-TIme", imageFrame)
if cv2.waitKey(10) & 0xFF == ord('/'):
    cap.release()
    cv2.destroyAllWindows()
    break
```

Gambar 3.Program ketiga

IV KESIMPULAN

Kesimpulan dari penelitian ini

1. Dengan menggunakan mendeteksi warna secara *Realtime* warna yang di deteksi cukup baik
2. Intensitas terhadap Cahaya sangangat berpengaruh besar pada saat pengambilan gambar Rgb tersebut

3. *Karena system menggunakan Realtime* sering terjadinya perubahan text saat menjalankan program seharusnya tidak terdeteksi menjadi terdeteksi.

IV DAFTAR PUSTAKA

[1] A. Krizhevsky, I. Sutskever, and G. E. Hinton, "Imagenet classification with deep convolutional neural networks," in NIPS, 2012.

[2] D. A. Prabowo and D. Abdullah, "Deteksi dan Perhitungan Objek Berdasarkan Warna Menggunakan Color Object Tracking," *Pseudocode*, vol. 5, no. 2, pp. 85–91, 2018, doi: 10.33369/pseudocode.5.2.85-91

[3] Syahrudin, Akbar Nur, and Tedi Kurniawan. "Input dan output pada bahasa pemrograman python." *Jurnal Dasar Pemograman Python STMIK* 20 (2018): 1-7..

[4] Fauziah, Fauziah, and Widya Wisanty. "Color Palette menggunakan Python cv2 dan NumPy." *Seminar Nasional Teknik Elektro dan Informatika (SNTEI)*. Vol. 9. No. 1. 202