

PERBANDINGAN KINERJA YOLOV8 DENGAN SEGMENTASI BERBASIS WARNA HSV PADA DETEKSI DAN PELACAKAN OBJEK REAL TIME EMBEDDED

Heri Setiawan, Bagus Patria, Yohanes Dwi Cahyono

Elektronika Sistem Senjata, Program Studi Rekayasa Keamanan Siber, Politeknik Angkatan Darat
Jl. Raya Anggrek, Kesatrian Poltekad, Desa Pendem, Kec. Junrejo, Kota Batu, Jawa Timur 65324
mhs.cyber8.baguspatricia@poltekad.ac.id

ABSTRAK

Perkembangan teknologi pengolahan citra digital dan kecerdasan buatan telah mendorong kebutuhan sistem deteksi dan pelacakan objek real-time yang cepat, akurat, serta efisien pada perangkat embedded dengan sumber daya komputasi terbatas. Penelitian ini bertujuan membandingkan kinerja dua metode computer vision, yaitu algoritma deep learning Ultralytics YOLOv8 dan metode konvensional berbasis segmentasi ruang warna HSV, dalam mendeteksi serta melacak objek di lingkungan Politeknik Angkatan Darat (Poltekad) yang dinamis. Penelitian dilakukan melalui implementasi alur pemrosesan citra, ekstraksi koordinat pusat objek, serta perhitungan error posisi (ΔX dan ΔY) terhadap pusat frame kamera, dengan evaluasi berdasarkan distribusi spasial error, kestabilan pelacakan, dan latency komputasi. Hasil menunjukkan bahwa YOLOv8 unggul dalam akurasi dan kestabilan pelacakan dengan sebaran error yang rapat serta minim false positive, meskipun memiliki fluktuasi latency 60–200 ms. Sebaliknya, segmentasi HSV lebih unggul dalam efisiensi komputasi dan kecepatan pemrosesan yang stabil, tetapi rentan terhadap variasi pencahayaan, bayangan, serta kemiripan warna latar belakang. Secara keseluruhan, YOLOv8 lebih sesuai untuk aplikasi kritis di lingkungan kompleks, sedangkan HSV ideal untuk kasus sederhana dengan objek warna solid. Penelitian ini merekomendasikan pengembangan sistem hybrid serta optimalisasi model untuk menghasilkan sistem visi komputer yang lebih adaptif, efisien, dan mendukung keselamatan operasional manusia di lapangan.

Kata kunci : YOLOv8, Segmentasi HSV, Deteksi Objek Real-time, Embedded System, Computer Vision

1. PENDAHULUAN

Perkembangan teknologi pengolahan citra digital dan kecerdasan buatan telah secara signifikan mendorong penerapan sistem deteksi objek secara real-time di berbagai domain, termasuk keamanan publik, pemantauan lingkungan, otomasi industri, serta pengawasan keselamatan manusia. Kebutuhan akan sistem deteksi yang cepat, akurat, dan andal semakin mendesak, terutama pada perangkat dengan sumber daya komputasi terbatas seperti *embedded systems*. Pada platform tersebut, efisiensi komputasi, optimalisasi memori, serta minimisasi latensi pemrosesan menjadi faktor penentu keberhasilan implementasi di lapangan, yang pada akhirnya berkontribusi terhadap keselamatan dan kesejahteraan manusia.

Dalam beberapa tahun terakhir, algoritma berbasis deep learning seperti seri YOLO telah banyak diadopsi karena kemampuannya melakukan deteksi objek secara cepat dengan tingkat akurasi yang kompetitif [1], [3]. Pendekatan single stage detection yang diterapkan YOLO memungkinkan pemrosesan dalam satu tahap, sehingga sangat sesuai untuk aplikasi yang menuntut respons waktu nyata [3], [11]. Namun, model-model ini masih menuntut sumber daya komputasi yang relatif tinggi, sehingga performanya dapat berpengaruh pada perangkat embedded dengan keterbatasan hardware [1], [7], [12]. Tantangan ini menjadi semakin relevan ketika sistem harus beroperasi secara stabil di lingkungan nyata yang dinamis dan tidak terduga [8], [9].

Di sisi lain, metode segmentasi berbasis ruang warna HSV [2], [4] (Hue, Saturation, Value) masih digunakan secara luas untuk kasus deteksi objek tertentu. Metode ini menawarkan keunggulan berupa kompleksitas komputasi yang rendah, konsumsi memori yang hemat, serta kecepatan pemrosesan yang tinggi dibandingkan pendekatan deep learning [2], [6]. HSV dapat bekerja secara efektif ketika objek target memiliki karakteristik warna yang khas dan kondisi pencahayaan relatif terkendali [4], [10]. Akan tetapi, kerentanan terhadap variasi intensitas cahaya, bayangan, serta kemiripan warna dengan latar belakang sering menjadi keterbatasan utama, khususnya di lingkungan terbuka yang kompleks [6].

Berdasarkan kondisi tersebut, diperlukan studi eksperimen komparatif yang sistematis untuk mengevaluasi dan membandingkan kinerja algoritma YOLO serta segmentasi HSV dalam deteksi objek real-time pada perangkat embedded system [6], [9]. Penelitian ini mengambil case di lingkungan Politeknik Angkatan Darat (Poltekad), yang memiliki karakteristik operasional unik dengan variasi kondisi pencahayaan dan dinamika objek yang dapat mempengaruhi performa sistem secara langsung. Melalui pendekatan ini, diharapkan dapat dihasilkan rekomendasi teknis yang tidak hanya akurat secara ilmiah, tetapi juga memperhatikan aspek kepraktisan, keandalan, serta manfaat bagi keselamatan dan efisiensi kerja manusia di lapangan.

2. TINJAUAN PUSTAKA

Pengembangan sistem pelacakan objek secara real-time memerlukan pemahaman mendalam mengenai implementasi metode computer vision, baik yang berbasis kecerdasan buatan (deep learning) maupun pendekatan konvensional berbasis manipulasi ruang warna [3], [11]. Pemilihan metode yang tepat sangat berpengaruh terhadap akurasi, efisiensi komputasi, serta keandalan sistem dalam mendukung keselamatan dan operasional manusia di lapangan. Bagian ini merangkum penelitian terdahulu yang relevan, kemudian menguraikan dasar algoritma YOLO, segmentasi HSV, ekstraksi pusat massa, serta model error posisi yang digunakan.

2.1. Penelitian Terdahulu

Sejumlah penelitian terdahulu telah mengembangkan deteksi objek real-time, baik melalui pendekatan deep learning berbasis YOLO maupun segmentasi warna HSV. Terven dkk. meninjau evolusi arsitektur YOLO dari YOLOv1 hingga YOLOv8 dan YOLO-NAS, serta menegaskan pergeseran ke detektor single-stage yang menyeimbangkan akurasi dan kecepatan [3]. Afifah dan Erniwati merangkum kemajuan YOLOv8 pada domain pengawasan, kendaraan otonom, industri, dan citra medis, serta menyoroti pentingnya model yang ringkas untuk aplikasi praktis [11].

Pada sisi implementasi edge, Yang dan Fan mengusulkan YOLOv8-Lite untuk sistem mengemudi otonom waktu nyata dengan pengurangan parameter agar layak di perangkat terbatas [1]. Chen dkk. mengembangkan YOLO-Compact khusus sistem embedded dan industri, serta melaporkan throughput tinggi pada masukan beresolusi besar [7]. Martínez Otero dkk. membandingkan implementasi native dan generated code YOLOv8/YOLOv9 pada NVIDIA Jetson, sehingga memperjelas trade-off latensi, akurasi, dan alur deployment [9]. Suchý dan Turčaník meninjau efisiensi energi YOLOv8l dan RT-DETR pada Raspberry Pi 5 serta Jetson Orin NX, dan menunjukkan bahwa model besar belum tentu mencapai target real-time 25 FPS tanpa optimasi perangkat keras [12].

Beberapa studi juga memperkuat YOLOv8 untuk skenario sulit. Talib dkk. menambahkan Context Attention Block (YOLOv8-CAB) guna meningkatkan deteksi objek berukuran kecil tanpa menaikkan kompleksitas secara berlebihan [5]. Nimma dkk. menggabungkan backbone YOLOv8 dengan mekanisme atensi dan kepala Transformer untuk pengawasan video real-time pada adegan ramai dan pencahayaan bervariasi [8]. Temuan-temuan tersebut menegaskan keunggulan semantik YOLOv8, tetapi tetap menyisakan persoalan beban komputasi pada platform embedded.

Pada ranah metode konvensional, Bora dkk. membandingkan ruang warna $L^*A^*B^*$ dan HSV untuk segmentasi citra, dan menyimpulkan bahwa HSV umumnya lebih unggul berdasarkan metrik MSE

dan PSNR [2]. Kang dkk. memanfaatkan HSV untuk lokalisasi objek otomatis pada robot grasping tanpa pengetahuan awal, termasuk strategi sampling ROI agar threshold tidak selalu disetel ulang secara manual [4]. Shaik dkk. membandingkan deteksi warna kulit pada HSV dan YCbCr, yang relevan dengan pengujian isolasi wajah/kulit pada penelitian ini [10]. Oleynikov selanjutnya mengusulkan segmentasi HSV adaptif pada pencahayaan berubah-ubah, serta membandingkannya dengan YOLOv8 menggunakan metrik IoU [6].

Meskipun literatur tersebut telah membahas YOLO, HSV, maupun implementasi embedded secara terpisah, kajian komparatif langsung antara Ultralytics YOLOv8 dan segmentasi HSV pada pelacakan real-time dengan metrik error posisi (ΔX dan ΔY) di lingkungan operasional seperti Poltekad masih terbatas. Kesenjangan inilah yang menjadi dasar penelitian ini: menilai secara empiris akurasi pelacakan, sebaran error, dan latency kedua metode pada perangkat bersumber daya terbatas.

2.2. Algoritma YOLO (You Only Look Once)

YOLO merupakan salah satu algoritma deteksi objek berbasis deep learning yang mengadopsi pendekatan single stage detector [3]. Berbeda dengan metode berbasis wilayah (region-based) yang memisahkan proses klasifikasi dan penentuan kotak pembatas (bounding box), YOLO memandang deteksi objek sebagai suatu masalah regresi tunggal. Citra masukan dibagi menjadi grid, di mana setiap sel grid bertanggung jawab memprediksi kotak pembatas serta skor probabilitas kelas secara simultan [3], [11].

Pada penelitian ini, varian yang digunakan adalah Ultralytics YOLOv8 [1], [3], [11]. Varian ini dipilih karena menerapkan arsitektur anchor-free yang memprediksi pusat objek secara langsung, sehingga secara signifikan mengurangi beban komputasi pada perangkat embedded [1], [7]. Selain itu, integrasi Feature Pyramid Network (FPN) dan Path Aggregation Network (PAN) dalam strukturnya meningkatkan kemampuan deteksi terhadap objek berukuran kecil serta objek yang mengalami oklusi parsial [5], [8], sehingga mendukung aplikasi yang menuntut keandalan tinggi di lingkungan operasional yang dinamis [9], [12].

2.3. Segmentasi Ruang Warna HSV (Hue, Saturation, Value)

Segmentasi HSV merupakan teknik computer vision konvensional yang mengekstrak objek berdasarkan karakteristik warna spesifiknya [2], [4]. Dibandingkan dengan ruang warna RGB yang sangat sensitif terhadap perubahan intensitas cahaya karena pencampuran informasi warna dan kecerahan, ruang warna HSV memisahkan secara eksplisit komponen Hue (warna murni), Saturation (tingkat kepekatan warna), dan Value (tingkat kecerahan) [2], [10]. Pemisahan ini menjadikan HSV lebih intuitif untuk thresholding warna pada aplikasi real-time [4], [6].

Proses ekstraksi objek dilakukan melalui metode thresholding atau pengambangan statis. Citra masukan bertipe RGB dikonversi terlebih dahulu ke format HSV [2], [4]. Selanjutnya, piksel yang berada dalam rentang batas bawah (lower bound) dan batas atas (upper bound) diubah menjadi piksel putih (nilai 1), sedangkan piksel di luar rentang tersebut menjadi hitam (nilai 0), sehingga menghasilkan citra biner (mask) [4], [6]. Untuk objek dengan karakteristik warna kulit, rentang HSV juga telah banyak dikaji sebagai dasar isolasi area target [10].

2.4. Ekstraksi Kontur dan Pusat Massa Objek

Setelah terbentuk citra biner dari hasil segmentasi warna, tahap selanjutnya adalah menentukan koordinat absolut objek melalui algoritma pencarian kontur berbasis topologi struktur *pixel*. Titik tengah (*centroid*) dari kontur terbesar yang berhasil diisolasi dihitung menggunakan momen citra spasial orde nol (M_{00}) dan orde satu (M_{10}, M_{01}). Pendekatan ini memungkinkan perolehan posisi objek secara akurat dan efisien melalui persamaan berikut:

$$x^- = \frac{M_{10}}{M_{00}}, \quad y^- = \frac{M_{01}}{M_{00}}$$

2.5. Persamaan Matematika Koordinat Error Visi

Posisi titik tengah objek target (X_t, Y_t) selanjutnya dibandingkan dengan titik pusat *frame* kamera (X_c, Y_c) untuk menghitung simpangan (*error*) posisi horizontal (ΔX) dan vertikal (ΔY) sebagai berikut:

$$\Delta X = X_t - X_c \quad (1)$$

$$\Delta Y = Y_t - Y_c \quad (2)$$

Persamaan di atas dinomori secara berurutan. Nilai selisih posisi (ΔX dan ΔY) dari hasil komputasi visi ini akan dikirimkan ke mikrokontroler sebagai parameter koreksi sistem penggerak aktuator, sehingga mendukung respons yang tepat dan aman bagi operator manusia.

3. METODE PENELITIAN

Metode penelitian yang diterapkan dalam kajian ini dirancang untuk membandingkan dua pendekatan utama Computer Vision, yaitu berbasis deep learning Ultralytics YOLOv8 [1], [3] dan berbasis konvensional melalui manipulasi ruang warna (segmentasi HSV) [2], [4]. Alur eksperimen difokuskan pada pemrosesan citra digital, ekstraksi koordinat, hingga analisis efisiensi pengenalan objek secara real-time pada perangkat embedded [7], [9].

3.1. Alur Pemrosesan Visi YOLOv8

Alur pemrosesan visi YOLOv8 pertama menggunakan model YOLOv8 yang telah dilatih untuk mengenali objek target secara langsung [1], [5], [11]. Alur kerja komputasi visi berbasis deep learning ini meliputi tahapan sebagai berikut:

- Akuisisi Citra: Menangkap frame video interaktif dari sensor visual secara *real-time*.
- Normalisasi Dimensi: Mengubah resolusi citra secara otomatis ke ukuran matriks input standar

model (640×480) pixel untuk menjaga konsistensi komputasi.

- Ekstraksi Fitur Konvolusional: Citra diproses melalui lapisan *Backbone* dan *Neck* berbasis jaringan syaraf tiruan untuk mengekstrak fitur arsitektur geometri dan semantik objek.
- Non-Maximum Suppression* (NMS): Mengeliminasi *bounding box* yang tumpang tindih (*overlap*) berdasarkan ambang batas skor kepercayaan (*confidence score*), sehingga menyisakan satu kotak pembatas optimal di sekitar objek target.
- Kalkulasi Pusat Massa: Mengambil nilai koordinat ekstrem dari *bounding box* hasil NMS, yaitu (X_{min}, Y_{min}) dan (X_{max}, Y_{max}), untuk menghitung titik tengah spasial objek (X_t, Y_t) menggunakan persamaan regresi linear sederhana pada pusat kotak.

3.2. Alur Pemrosesan Visi Segmentasi HSV

Skenario kedua memanfaatkan manipulasi nilai piksel pada ruang warna HSV secara terstruktur [2], [4] melalui tahapan sebagai berikut:

- Konversi Ruang Warna: Mengubah format matriks citra bawaan dari ruang warna RGB (Red, Green, Blue) menjadi HSV (Hue, Saturation, Value) untuk memisahkan informasi kromatisitas dengan intensitas cahaya [2], [4].
- Operasi *InRange* (*Thresholding*): Menerapkan pemisahan biner di mana pixel yang berada dalam ambang batas spesifik (*lower bound* dan *upper bound*) dari objek target akan diisolasi menjadi area putih (nilai 1), sedangkan pixel latar belakang diubah menjadi hitam (nilai 0).
- Operasi Morfologi: Menerapkan fungsi *Erosion* dan *Dilation* (atau kombinasi *Opening/Closing*) untuk mereduksi noise titik-titik kecil terisolasi serta menutup lubang kosong (*hole*) pada area objek biner.
- Ekstraksi Kontur dan Momen Citra: Menemukan batas terluar dari objek biner menggunakan algoritma pelacakan topologi piksel, kemudian menghitung momen spasial orde nol dan satu untuk mendapatkan koordinat pusat massa (*centroid*) objek (X_t, Y_t).

3.3. Skema Perhitungan Error Pusat Frame

Kedua metode visi tersebut diintegrasikan ke dalam sebuah fungsi perhitungan deviasi posisi untuk mendeteksi sejauh mana objek bergeser dari pusat pengamatan kamera. Nilai simpangan (*error*) dihitung secara matematis menggunakan model koordinat pixel Kartesius 2D berdasarkan persamaan berikut:

$$\Delta X = X_t - X_c \quad (1)$$

$$\Delta Y = Y_t - Y_c \quad (2)$$

Di mana:

- ΔX dan ΔY merupakan nilai error pergeseran objek pada sumbu horizontal (X) dan vertikal (Y).

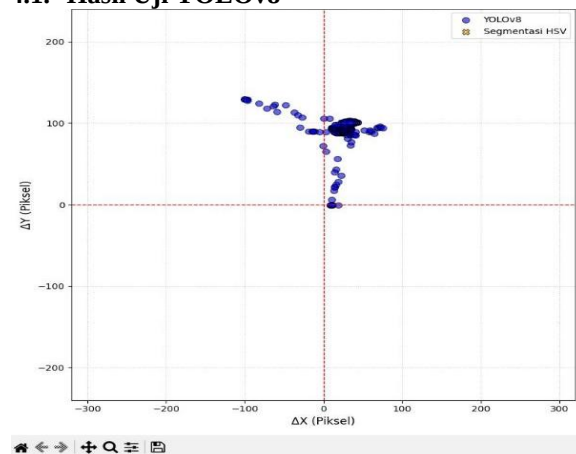
- b. X_t dan Y_t adalah koordinat titik tengah objek hasil komputasi dari masing-masing metode visi.
- c. X_c dan Y_c adalah koordinat titik pusat frame gambar yang bersifat statis, yaitu berada pada koordinat (320,240) untuk resolusi standar 640×480 pixel.

Nilai deviasi posisi yang diperoleh dari Persamaan (3) dan (4) ini selanjutnya dikirimkan secara sekuensial menuju sub-sistem kendali/mikrokontroler sebagai parameter masukan aktuator untuk diuji tingkat responsivitas dan kestabilannya.

4. HASIL DAN PEMBAHASAN

Evaluasi mendalam terhadap performa kedua metode deteksi dan pelacakan objek dilakukan melalui analisis distribusi spasial koordinat kesalahan posisi serta efisiensi waktu pemrosesan (*inference time*) per bingkai. Pendekatan ini tidak hanya menyoroti keunggulan teknis, tetapi juga implikasinya terhadap keandalan sistem dalam mendukung tugas-tugas manusiawi yang menuntut presisi dan keselamatan, seperti pengawasan taktis atau bantuan operasi lapangan

4.1. Hasil Uji YOLOv8



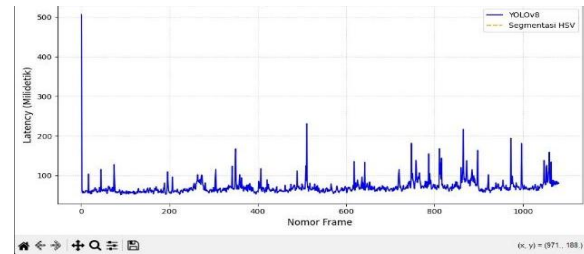
Gambar 1. Sebaran Koordinat Error Posisi Objek

Grafik sebaran koordinat pada Gambar 1 menggambarkan tingkat stabilitas algoritma dalam mengunci objek target pada area bidik serta frekuensi kesalahan deteksi (*false positive*). Titik pusat (0,0) yang ditandai garis putus-putus merah merepresentasikan titik tengah absolut lensa kamera sebagai acuan sasaran ideal.

Ultralytics YOLOv8 [1], [9] (Lingkaran Biru): Menunjukkan pengelompokan (clustering) yang sangat rapat dan konsisten di kuadran atas dengan rentang hingga 120 piksel pada sumbu horizontal maupun vertikal. Ketiadaan outlier yang signifikan mencerminkan stabilitas tinggi dalam mengunci centroid target secara presisi, sehingga mendukung pengambilan keputusan yang lebih aman dan andal bagi operator manusia [8], [9].

Segmentasi Warna HSV [2], [6] (Silang Oranye): Menunjukkan dispersi spasial yang lebih besar,

termasuk anomali lompatan koordinat. Dalam ranah computer vision, fenomena ini mengindikasikan kehadiran noise atau gangguan pelacakan akibat kemiripan spektrum warna latar belakang dengan ambang batas target, yang menyebabkan kotak pengunci (bounding box) terlepas dari objek utama [4], [6].

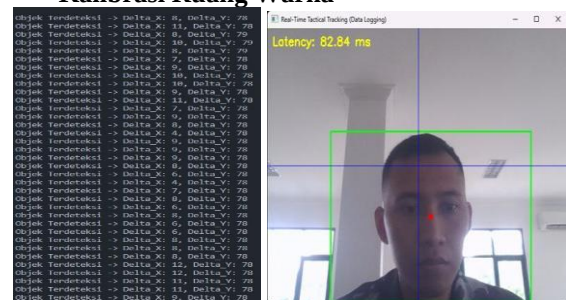


Gambar 2. Latency Komputasi Real Time YOLOv8

Efisiensi komputasi diukur melalui waktu pemrosesan setiap bingkai dalam satuan milidetik (ms), sebagaimana ditampilkan pada Gambar 2. Pada fase awal (frame 0), kedua metode mengalami lonjakan latensi hingga ~600 ms, yang merupakan fenomena normal model warm-up akibat inisialisasi memori dan arsitektur komputasi.

Setelah fase inisiasi, kecepatan pemrosesan stabil pada rentang 60–80 ms per frame. Namun, YOLOv8 menunjukkan fluktuasi dinamis hingga 200 ms, yang mencerminkan kompleksitas ekstraksi fitur konvolusional berlapis pada jaringan saraf tiruan [1], [7], [12]. Di sisi lain, segmentasi HSV yang berbasis manipulasi piksel linear cenderung lebih stabil dan hemat sumber daya, meskipun kurang tangguh terhadap variasi lingkungan [2], [6], [9].

4.2. Analisis Visual Antarmuka Eksperimen dan Kalibrasi Ruang Warna

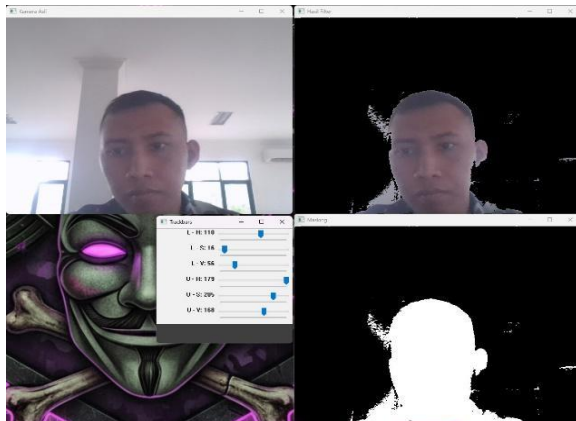


Gambar 3. Visualisasi antarmuka pelacakan objek real-time berbasis YOLOv8.

Gambar 3 memperlihatkan antarmuka sistem pelacakan berbasis YOLOv8 [1], [9]. Objek manusia terdeteksi secara real-time dengan bounding box hijau. Centroid target ditandai titik merah, sementara crosshair biru berfungsi sebagai acuan pusat kamera. Selisih geometris ΔX dan ΔY antara keduanya menghasilkan parameter koreksi yang dapat diteruskan ke aktuator, mendukung respons sistem yang cepat dan intuitif bagi pengguna manusia [8].

4.3. Hasil Uji HSV

Metode segmentasi HSV merupakan pendekatan berbasis warna yang sederhana namun efektif dalam computer vision, khususnya untuk lingkungan dengan pencahayaan terkendali [2], [4]. Keunggulannya terletak pada representasi warna yang intuitif (Hue untuk jenis warna, Saturation untuk kemurnian, Value untuk kecerahan), sehingga lebih tahan terhadap variasi intensitas cahaya dibandingkan RGB [2], [10].



Gambar 4. Kalibrasi dan Segmentasi Warna Kulit/Wajah

Pada pengujian metode segmentasi warna berbasis HSV [4], [10], dilakukan dua eksperimen utama untuk mendemonstrasikan kemampuan deteksi dan pelacakan objek berdasarkan rentang warna tertentu.

Metode HSV berhasil mengisolasi warna kulit secara efektif [10]. Dengan pengaturan trackbar sebagai berikut:

Lower (L): [110, 16, 56]

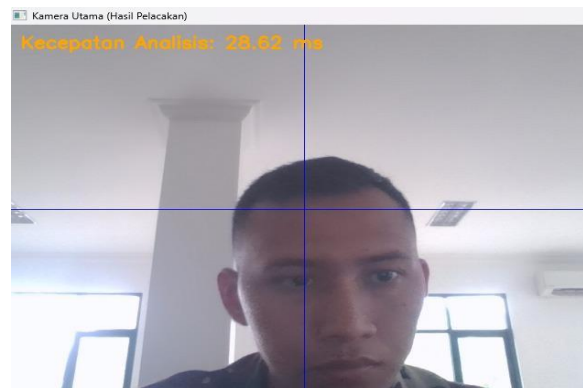
Upper (U): [179, 205, 168]

Jendela Masking menunjukkan area wajah dan telinga berwarna putih (nilai 1), sementara latar belakang, dinding, dan seragam berubah menjadi hitam (nilai 0). Hasil *bitwise AND* pada jendela "Hasil Filter" menghasilkan visualisasi hanya kepala yang terisolasi dengan bersih di atas latar hitam, tanpa noise berlebih. Nilai koordinat pusat (ΔX dan ΔY) dapat diekstrak secara real-time untuk menghasilkan data CSV.

Dalam eksperimen Gambar 4, kalibrasi berhasil mendeteksi warna kulit dengan akurasi tinggi karena rentang HSV yang disesuaikan secara manual via trackbar. Hal ini sesuai dengan prinsip bahwa HSV memisahkan komponen kromatik (Hue) dari luminansi, sehingga cocok untuk skin detection [2], [10]. Hasil masking yang bersih menunjukkan minimnya false positive pada latar belakang netral.

Dengan rentang default hijau terang (*lower_color* = [35, 100, 100]; *upper_color* = [85, 255, 255]), jendela Masking sepenuhnya hitam, menandakan tidak ada deteksi warna hijau terang. Seragam loreng TNI (hijau kusam/camo) tidak terdeteksi karena berada di luar rentang HSV yang ditentukan [2], [4]. Namun, saat objek hijau terang (misalnya stabilo) dihadirkan,

masking langsung menyala putih, kotak pelacakan oranye muncul, dan nilai $\Delta X/\Delta Y$ tercetak real-time. Ini membuktikan metode HSV bekerja dengan baik pada warna mencolok namun sensitif terhadap variasi pencahayaan dan saturasi [6].



Gambar 5. Segmentasi Warna

Secara keseluruhan, metode HSV menghasilkan segmentasi biner yang cepat dan ringan, cocok untuk pelacakan real-time dengan kalibrasi trackbar [2], [4].

Kegagalan deteksi seragam loreng menyoroti keterbatasan utama HSV, yaitu sensitivitas terhadap variasi warna akibat pencahayaan, bayangan, atau pola camo yang memiliki saturasi rendah [6]. Berbeda dengan YOLOv8 yang memanfaatkan fitur bentuk dan tekstur melalui deep learning [3], [5], [8], HSV "buta" terhadap bentuk dan hanya bergantung pada rentang warna [2], [4]. Oleh karena itu, HSV lebih unggul dalam kecepatan dan efisiensi komputasi untuk objek warna solid, tetapi memerlukan kalibrasi ulang untuk setiap kondisi pencahayaan atau objek baru [6], [10].

5. KESIMPULAN DAN SARAN

Penelitian ini telah membandingkan dua pendekatan deteksi dan pelacakan objek secara real-time pada perangkat embedded system, yaitu metode berbasis deep learning Ultralytics YOLOv8 [1], [3], [11] dan metode konvensional berbasis segmentasi ruang warna HSV [2], [4]. Hasil eksperimen menunjukkan bahwa YOLOv8 unggul dalam hal akurasi dan kestabilan pelacakan [8], [9]. Sebaran koordinat error posisinya sangat rapat dan konsisten di sekitar titik pusat frame dengan minimnya outlier atau false positive, sehingga lebih andal di lingkungan dinamis seperti area Politeknik Angkatan Darat yang memiliki variasi pencahayaan dan objek kompleks. Meskipun mengalami fluktuasi latency antara 60–200 ms, YOLOv8 mampu mempertahankan penguncian objek yang presisi berkat kemampuannya memahami fitur bentuk, tekstur, dan konteks melalui jaringan saraf tiruan [5], [7].

Di sisi lain, segmentasi HSV menunjukkan keunggulan dalam efisiensi komputasi dan kecepatan pemrosesan yang lebih stabil serta rendah konsumsi sumber daya, sehingga sangat cocok untuk perangkat embedded dengan keterbatasan hardware [2], [6], [12]. Namun, metode ini rentan terhadap variasi

pencahayaan, bayangan, serta kemiripan warna dengan latar belakang, yang menyebabkan dispersi error lebih besar dan kegagalan deteksi pada objek seperti seragam loreng [4], [6]. HSV bekerja optimal hanya pada objek dengan warna solid dan khas dalam kondisi pencahayaan yang terkendali [10]. Secara keseluruhan, YOLOv8 lebih sesuai untuk aplikasi yang mengutamakan keandalan dan presisi di lingkungan nyata yang tidak terduga [8], [9], sedangkan HSV tetap relevan sebagai solusi ringan dan cepat untuk kasus sederhana [2], [4]. Kedua metode mampu menghasilkan nilai error posisi (ΔX dan ΔY) yang dapat digunakan sebagai input sistem kendali aktuator untuk meningkatkan keselamatan dan efisiensi operasional manusia di lapangan.

Berdasarkan hasil penelitian, pemilihan metode sebaiknya disesuaikan dengan konteks aplikasi. YOLOv8 direkomendasikan untuk aplikasi kritis yang membutuhkan ketahanan terhadap variasi lingkungan, seperti pengawasan taktis dan keamanan publik [8], [11], [12], sementara HSV lebih tepat untuk aplikasi sederhana dengan objek warna mencolok dan pencahayaan stabil, atau sebagai lapisan pendukung untuk mengurangi beban komputasi [2], [6]. Pengembangan sistem hybrid yang menggabungkan HSV untuk deteksi awal cepat dan YOLOv8 untuk verifikasi serta pelacakan presisi di kondisi kompleks sangat dianjurkan agar dapat mengoptimalkan keseimbangan antara kecepatan dan akurasi [1], [6], [7].

Selain itu, diperlukan optimalisasi lebih lanjut pada YOLOv8 melalui teknik quantization dan pruning agar lebih ringan dan efisien di perangkat embedded [1], [7], [12]. Untuk HSV, disarankan penggunaan adaptive thresholding serta integrasi filter tambahan seperti Kalman Filter untuk meningkatkan ketahanan [6]. Penelitian lanjutan juga perlu dilakukan dengan pengujian di berbagai kondisi pencahayaan ekstrem, integrasi dengan sensor lain seperti kamera IR atau depth camera, serta evaluasi performa pada hardware spesifik seperti Raspberry Pi atau Jetson Nano dengan mempertimbangkan metrik FPS, konsumsi daya, dan akurasi jarak jauh [9], [12]. Pengembangan antarmuka pengguna yang lebih intuitif serta sistem kalibrasi otomatis untuk HSV juga menjadi langkah penting ke depan [4]. Penelitian ini diharapkan dapat menjadi referensi teknis bagi pengembangan sistem visi komputer berbasis embedded yang lebih aman, efisien, dan adaptif di lingkungan operasional militer maupun sipil.

DAFTAR PUSTAKA

- [1] M. Yang and X. Fan, "YOLOv8-Lite: A lightweight object detection model for real-time autonomous driving systems," *ICCK Trans. Emerg. Topics Artif. Intell.*, vol. 1, no. 1, pp. 1–16, Apr. 2024, doi: 10.62762/TETAI.2024.894227.
- [2] D. J. Bora, A. K. Gupta, and F. A. Khan, "Comparing the performance of L*A*B* and HSV color spaces with respect to color image segmentation," *Int. J. Emerg. Technol. Adv. Eng.*, vol. 5, no. 2, pp. 192–203, Feb. 2015.
- [3] J. Terven, D.-M. Córdova-Esparza, and J.-A. Romero-González, "A comprehensive review of YOLO architectures in computer vision: From YOLOv1 to YOLOv8 and YOLO-NAS," *Mach. Learn. Knowl. Extr.*, vol. 5, no. 4, pp. 1680–1716, 2023, doi: 10.3390/make5040083.
- [4] H.-C. Kang, H.-N. Han, H.-C. Bae, M.-G. Kim, J.-Y. Son, and Y.-K. Kim, "HSV color-space-based automated object localization for robot grasping without prior knowledge," *Appl. Sci.*, vol. 11, no. 16, Art. no. 7593, 2021, doi: 10.3390/app11167593.
- [5] M. Talib, A. H. Y. Al-Noori, and J. Suad, "YOLOv8-CAB: Improved YOLOv8 for real-time object detection," *Karbala Int. J. Modern Sci.*, vol. 10, no. 1, Art. no. 5, 2024, doi: 10.33640/2405-609X.3339.
- [6] A. A. Oleynikov, "Adaptive HSV segmentation for real-time object detection under varying lighting conditions," *Informacionnye Tehnologii*, vol. 32, no. 1, pp. 46–56, 2026.
- [7] Y. Chen, B. Zhao, X. Jia, and T. Ma, "Efficient real-time object detection for embedded and industrial systems: The You Only Look Once-Compact approach," *Eng. Appl. Artif. Intell.*, vol. 156, Art. no. 109927, Sep. 2025, doi: 10.1016/j.engappai.2024.109927.
- [8] D. Nimma et al., "Object detection in real-time video surveillance using attention based transformer-YOLOv8 model," *Alexandria Eng. J.*, vol. 118, pp. 482–495, Apr. 2025, doi: 10.1016/j.aej.2025.01.032.
- [9] P. Martínez Otero, A. Tellaeche, and M. Hernández Melero, "Performance analysis of the YOLO object detection algorithm in embedded systems: Generated code vs. native implementation," *Computation*, vol. 14, no. 3, Art. no. 67, 2026, doi: 10.3390/computation14030067.
- [10] K. B. Shaik, P. Ganesan, V. Kalist, B. S. Sathish, and J. M. M. Jenitha, "Comparative study of skin color detection and segmentation in HSV and YCbCr color space," *Procedia Comput. Sci.*, vol. 57, pp. 41–48, 2015, doi: 10.1016/j.procs.2015.07.362.
- [11] V. Afifah and S. Erniwati, "YOLOv8 for object detection: A comprehensive review of advances, techniques, and applications," *Int. J. Adv. Comput. Inform.*, vol. 2, no. 1, pp. 53–61, 2026.
- [12] I. Suchý and M. Turčaník, "Review of large YOLOv8 and RT-DETR energy efficiency on edge devices for real-time detection," *Sci. Rep.*, vol. 16, Art. no. 10908, 2026, doi: 10.1038/s41598-026-46453-6.