

Laporan Praktikum Kontrol Cerdas

Nama : Wahyu Agustina
NIM : 224308046
Kelas : TKA-6B
Akun Github : <https://github.com/Wahyuagustina24>
Student Lab Assistant : Mba Salma Halizah

1. **Judul Percobaan:** Deteksi Objek Sederhana dengan *OpenCV*

2. **Tujuan Percobaan:**

Tujuan dari percobaan Deteksi Objek Sederhana dengan *OpenCV* adalah :

- Memahami konsep dasar kontrol cerdas (*Intelligent Control Systems*).
- Mengenali peran AI, *Machine Learning* (ML), dan *Deep Learning* (DL) dalam sistem kendali.
- Mempelajari penerapan *Computer Vision* dalam sistem kontrol berbasis AI.
- Menggunakan Python dan *OpenCV* untuk mendeteksi objek secara sederhana.
- Memanfaatkan GitHub untuk *version control* dan Kaggle sebagai sumber dataset.

3. **Landasan Teori:**

Kecerdasan buatan atau *Artificial Intelligence* (AI) adalah sistem yang mempelajari bagaimana membuat komputer dapat berpikir, belajar, dan bertindak seperti manusia (Dosari & Abouellail, 2023). *Intelligent Control* adalah metode pengendalian sistem yang menggunakan AI, *Machine Learning*, dan *Deep Learning* untuk meningkatkan performa dan efisiensi. Contoh implementasi *Intelligent Control* yaitu pada robotika (kontrol gerak robot berbasis AI), otomotif (*autonomous driving system*), industri (prediksi dan optimasi proses manufaktur), dan medis (AI untuk kontrol peralatan kesehatan) (Marsella dkk., 2023).

Software yang digunakan dalam percobaan ini berupa Python dan Open CV. OpenCV (*Open Source Computer Vision Library*) adalah perpustakaan

perangkat lunak yang dirancang khusus untuk pengolahan citra dan visi komputer. Dikembangkan oleh Intel dan sekarang bersifat *open-source*, OpenCV menyediakan berbagai alat dan fungsi yang memudahkan pengembang untuk membuat aplikasi berbasis pengolahan citra dan visi komputer (Zebua & Rosyani, 2024). Python, sebagai salah satu bahasa pemrograman yang didukung oleh OpenCV, menjadi pilihan populer karena sintaksnya yang mudah dipahami dan kemampuannya untuk integrasi dengan berbagai pustaka lain, seperti NumPy, SciPy, dan *scikit-learn*. Dengan OpenCV Python, pengembang dapat mengakses berbagai algoritma canggih seperti deteksi objek, pelacakan, pengenalan wajah, dan segmentasi citra. Salah satu aplikasi penting dari OpenCV Python adalah dalam deteksi objek (Alam dkk., 2024). Deteksi objek adalah proses mengidentifikasi dan menemukan instansi objek tertentu dalam gambar atau video. Dalam konteks pengolahan citra dan visi komputer, deteksi objek melibatkan penggunaan algoritma dan teknik khusus untuk mengenali berbagai jenis objek.

Model warna RGB merupakan singkatan dari Red (Merah) Green (Hijau) Blue (Biru). Model warna RGB juga disebut additive color atau warna pencahayaan karena apabila RGB dikombinasikan maka akan menghasilkan warna putih. RGB merupakan model warna yang paling dasar dalam melakukan penyimpanan gambar (Goenawan dkk., 2022). Pada setiap pixel warna memiliki rentang nilai intensitas mulai dari 0 sampai dengan 255. Setiap titik yang berada pada ruang warna RGB merupakan warna dengan memiliki komponen R, G dan B. Untuk titik (0,0,0) merupakan titik warna yang berwarna hitam, sedangkan titik (1,1,1) merupakan titik warna yang berwarna putih.

4. Analisis dan Diskusi:

- Analisis

Praktikum percobaan pada minggu pertama ini adalah mendeteksi objek berwarna merah, kemudian dilakukan modifikasi program untuk mendeteksi objek berwarna merah, hijau dan biru secara *real-time* dengan menggunakan OpenCV. Selain itu, juga dilakukan modifikasi menggunakan fitur *bounding box* untuk menandai objek yang terdeteksi.

Tahap pertama yang dilakukan adalah menginisialisasi kamera menggunakan **cv2.VideoCapture** dengan kamera bawaan PC/Laptop atau dengan *webcam*. Kemudian, untuk menjaga konsistensi warna meskipun terdapat perubahan pencahayaan, setiap frame hasil tangkapan diubah format warnanya menjadi HSV. Proses konversi ini memanfaatkan kode **hsv = cv2.cvtColor(frame, cv2.COLOR_BGR2HSV)** untuk mengubah dari format BGR asli ke HSV.

Rentang warna merah didefinisikan dengan menggunakan kode **lower_red = np.array([0, 120, 70])** dan **upper_red = np.array([10, 255, 255])** serta dibuat sebuah mask dengan fungsi **mask_red = cv2.inRange(hsv, lower_red, upper_red)** dan **result_red = cv2.bitwise_and(frame, frame, mask=mask_red)**.

Rentang warna biru didefinisikan dengan menggunakan fungsi **lower_blue = np.array([100, 150, 0])** dan **upper_blue = np.array([140, 255, 255])** serta dibuat sebuah mask dengan memanfaatkan fungsi **mask_blue = cv2.inRange(hsv, lower_blue, upper_blue)** dan **result_blue = cv2.bitwise_and(frame, frame, mask=mask_blue)**.

Rentang warna hijau didefinisikan dengan menggunakan kode **lower_green = np.array([40, 40, 40])** dan **upper_green = np.array([80, 255, 255])** serta dibuat sebuah mask dengan memanfaatkan fungsi **mask_green = cv2.inRange(hsv, lower_green, upper_green)** dan **result_green = cv2.bitwise_and(frame, frame, mask=mask_green)**.

Setelah itu, digunakan kode **mask = mask_red + mask_blue + mask_green** dan **result = cv2.bitwise_and(frame, frame, mask=mask)** untuk menggabungkan ketiga mask, kode ini akan menghasilkan satu mask yang mencakup semua area yang terdeteksi berwarna merah, biru, atau hijau.

Kemudian penambahan fitur kontur dan *bounding box* dengan menggunakan kode **cv2.findContours()** untuk mendeteksi kontur area yang memiliki warna merah, biru atau hijau, sedangkan untuk menggambar *bounding box* menggunakan **cv2.boundingRect()** yang berfungsi untuk memberikan koordinat dan ukuran kotak pada objek yang

berwarna merah, biru atau hijau tersebut. *Bounding box* ini digambar pada frame asli menggunakan **cv2.rectangle()**. Penambahan fitur *bounding box* secara signifikan meningkatkan kemampuan sistem dalam memahami dan berinteraksi dengan informasi visual serta kejelasan dan akurasi deteksi, karena tidak hanya menampilkan area yang memiliki warna, tetapi juga menandai objek dengan jelas.

- Diskusi

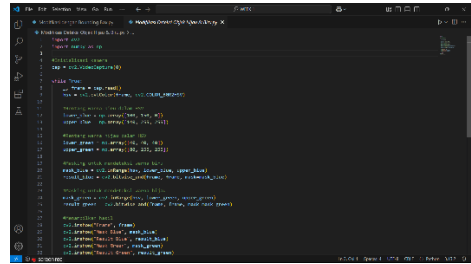
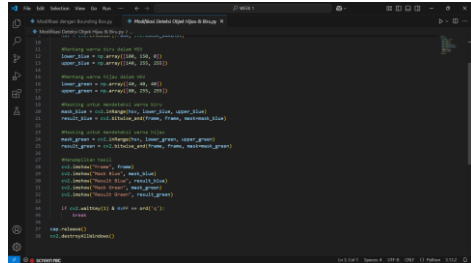
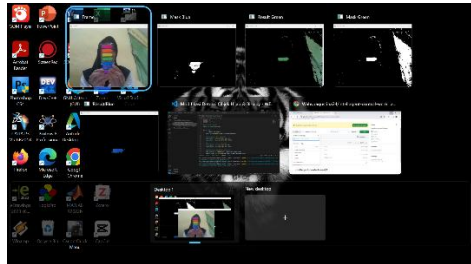
Metode untuk mendeteksi objek berwarna merah, hijau dan biru dengan penambahan fitur *bounding box* dengan OpenCV ini memiliki kelebihan dan kekurangan. Kelebihan dari metode ini adalah memudahkan dalam mendeteksi objek berdasarkan warna tertentu. Dengan penggunaan warna HSV lebih cepat dan efisien, sehingga cocok untuk aplikasi *real-time* seperti pelacakan objek atau pengawasan. Selain itu, penambahan fitur *bounding box* memberikan informasi penting tentang lokasi dan ukuran objek yang terdeteksi dengan warna berbeda beda dalam satu frame. Namun, di sisi lain dengan metode ini memiliki kekurangan atau keterbatasan salah satunya yaitu rentan terhadap *noise* atau gangguan pada citra. *Noise* dapat menyebabkan deteksi palsu atau mengurangi akurasi deteksi. Selain itu, sistem akan mengalami kesalahan deteksi jika lingkungan pencahayaan berubah atau objek lain dengan warna yang serupa.

5. Assignment:

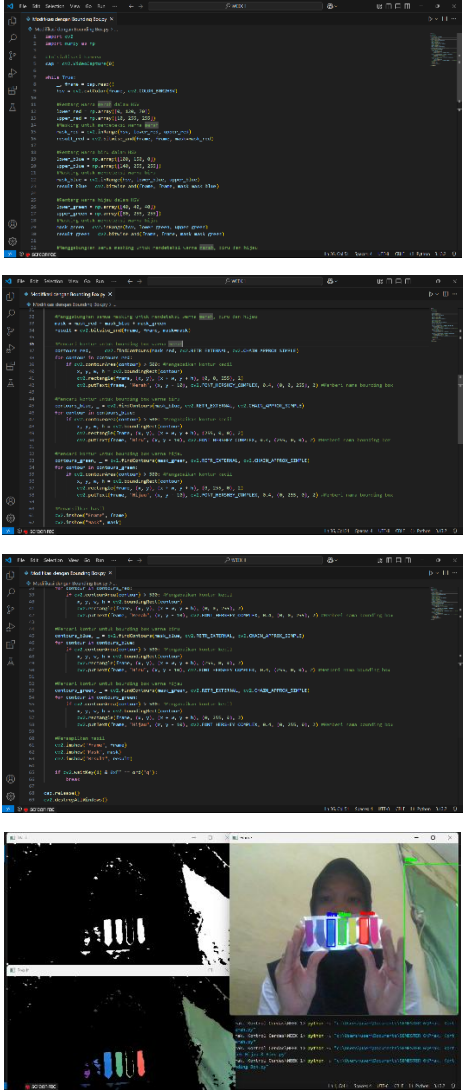
Dalam praktikum ini adalah untuk mendeteksi objek warna berbasis warna menggunakan OpenCV dan dengan menggunakan bahasa pemrograman Python. Pada sistem ini dirancang untuk mendeteksi warna biru dengan *bounding box*. Dimulai dengan menginisialisasi kamera, yang kemudian dikonversi ke warna HSV, setelah itu dilakukan masking berdasarkan rentang warna biru dan ditambahkan fitur *bounding box* menggunakan fungsi **cv2.boundingRect()** dan label teks “nama warna” ditambahkan untuk memberikan informasi visual mengenai hasil deteksi. Dengan adanya *bounding box*, sistem menjadi lebih informatif, karena pengguna dapat dengan jelas melihat objek yang dideteksi dalam kamera. Namun, terdapat kekurangan

seperti kemungkinan deteksi objek yang tidak diinginkan, seperti latar belakang atau pencahayaan memiliki warna yang serupa.

6. Data dan Output Hasil Pengamatan:

No	Variabel	Hasil Pengamatan
1.	<ul style="list-style-type: none"> - Menginisialisasi kamera, yang kemudian dikonversi ke warna HSV, kode program yaitu <code>cv2.cvtColor(frame, cv2.COLOR_BGR2HSV)</code> - Rentang warna biru dalam HSV yaitu <code>lower_blue = np.array([100, 150, 0])</code> <code>upper_blue = np.array([140, 255, 255])</code> - Rentang warna hijau dalam HSV yaitu <code>lower_green = np.array([40, 40, 40])</code> <code>upper_green = np.array([80, 255, 255])</code> - Masking untuk mendeteksi warna biru yaitu <code>mask_blue = cv2.inRange(hsv, lower_blue, upper_blue)</code> <code>result_blue = cv2.bitwise_and(frame, frame, mask=mask_blue)</code> - Masking untuk mendeteksi warna hijau yaitu <code>mask_green = cv2.inRange(hsv, lower_green, upper_green)</code> 	  

	<pre> result_green = cv2.bitwise_and(frame, frame, mask=mask_green) - Kemudian untuk menampilkan hasil yaitu cv2.imshow("Frame", frame) cv2.imshow("Mask Blue", mask_blue) cv2.imshow("Result Blue", result_blue) cv2.imshow("Mask Green", mask_green) cv2.imshow("Result Green", result_green) - Menekan tombol 'q' yang akan melepaskan akses kamera dan menutup tiga tab yaitu cv2.waitKey(1) & 0xFF == ord('q') </pre>	
2.	<pre> - Menginisialisasi kamera, yang kemudian dikonversi ke warna HSV, kode program yaitu cv2.cvtColor(frame, cv2.COLOR_BGR2HSV) - Rentang warna merah dalam HSV yaitu lower_red = np.array([0, 120, 70]) upper_red = np.array([10, 255, 255]) - Rentang warna biru dalam HSV yaitu lower_blue = np.array([100, 150, 0]) </pre>	

	<pre>upper_blue = np.array([140, 255, 255])</pre> <ul style="list-style-type: none"> - Rentang warna hijau dalam HSV yaitu <code>lower_green = np.array([40, 40, 40])</code> <code>upper_green = np.array([80, 255, 255])</code> - Masking untuk mendeteksi warna merah yaitu <code>mask_red = cv2.inRange(hsv, lower_red, upper_red)</code> <code>result_red = cv2.bitwise_and(frame, frame, mask=mask_red)</code> - Masking untuk mendeteksi warna biru yaitu <code>mask_blue = cv2.inRange(hsv, lower_blue, upper_blue)</code> <code>result_blue = cv2.bitwise_and(frame, frame, mask=mask_blue)</code> - Masking untuk mendeteksi warna hijau yaitu <code>mask_green = cv2.inRange(hsv, lower_green, upper_green)</code> <code>result_green = cv2.bitwise_and(frame, frame, mask=mask_green)</code> - Menemukan kontur yaitu <code>cv2.findContours(mask, cv2.RETR_TREE, cv2.CHAIN_APPROX_SIMPLE)</code> 	
--	---	---

	<ul style="list-style-type: none"> - Untuk memunculkan <i>bounding box</i> yaitu <code>cv2.boundingRect(contour)</code> - Kemudian untuk menampilkan hasil yaitu <code>cv2.imshow("Frame", frame)</code> <code>cv2.imshow("Mask", mask)</code> <code>cv2.imshow("Result", result)</code> - Menyelesaikan program, dengan menekan tombol 'q' yaitu <code>cv2.waitKey(1) & 0xFF == ord('q')</code> 	
--	---	--

7. Kesimpulan:

Berdasarkan praktikum dan analisis yang telah dilakukan, maka dapat ditarik kesimpulan yaitu:

- Sistem atau metode pendeteksi objek menggunakan OpenCV dan Python dapat mengidentifikasi objek berdasarkan warna tertentu seperti merah, hijau, dan biru.
- Penambahan fitur bounding box dapat meningkatkan kejelasan deteksi dengan memberikan batas visual atau kotak pada objek yang terdeteksi.
- Sistem atau metode ini memiliki keterbatasan terutama dalam hal sensitivitas terhadap perubahan pencahayaan serta kemungkinan mendeteksi objek dengan warna yang serupa.

8. Saran:

Untuk meningkatkan ketepatan deteksi objek dapat digunakan metode lain seperti deteksi tepi (*Canny Edge Detection*) atau model berbasis *machine learning*. Selain itu, apabila warna latar belakang mirip dengan objek yang ingin dideteksi, pertimbangkan penggunaan metode lain seperti deteksi bentuk atau fitur objek selain warna. Selain metode HSV, gunakan model *deep learning* seperti YOLO atau Faster R-CNN yang lebih handal dalam deteksi objek.

9. Daftar Pustaka:

- Alam, S., Zainal, M., & Fazil, E. (2024). *PERANCANGAN SISTEM PENGENALAN WAJAH MENGGUNAKAN PYTHON, OPENCV DAN HAARCASCADE*. 9.
- Dosari, F. H. M. A., & Abouellail, S. I. A. D. (2023). Artificial Intelligence (AI) Techniques for Intelligent Control Systems in Mechanical Engineering. *American Journal of Smart Technology and Solutions*, 2(2), 55–64. <https://doi.org/10.54536/ajsts.v2i2.2188>
- Goenawan, A. D., Rachman, M. B. A., & Pulungan, M. P. (2022). Identifikasi Warna Pada Objek Citra Digital Secara Real Time Menggunakan Pengolahan Model Warna HSV. *Jurnal Teknik Informatika dan Elektro*, 4(1), 68–74. <https://doi.org/10.55542/jurtie.v4i1.430>
- Marsella, M., Wijaya, C. S., Wijaya, I., Shidqi, M. T., & Novita, D. (2023). ANALISIS IMPLEMENTASI ARTIFICIAL INTELLIGENCE UNTUK BISNIS: SYSTEMATIC LITERATURE REVIEW. *DEVICE : JOURNAL OF INFORMATION SYSTEM, COMPUTER SCIENCE AND INFORMATION TECHNOLOGY*, 4(2), 133–145. <https://doi.org/10.46576/device.v4i2.4037>
- Zebua, E. T. P., & Rosyani, P. (2024). *Perancangan Deteksi Objek Kendaraan Bermotor Berbasis OpenCV Python menggunakan Metode HOG-SVM untuk Analisis Lahu Lintas Cerdas*. 2(1).