

Laporan Praktikum Minggu Ke-2 Kontrol Cerdas

Minggu ke-2

Nama : Putu Sriwahyuningsih
NIM : 224308066
Kelas : TKA 7C
Akun Github : <https://github.com/putusriwahyu>

1. Judul Percobaan

Penerapan Computer Vision dalam Color Detection dengan Machine Learning.

2. Tujuan Percobaan

- Memahami dasar-dasar Machine Learning dalam sistem kendali.
- Mengimplementasikan model ML sederhana untuk klasifikasi objek.
- Menggunakan Scikit-learn untuk membuat model ML dasar.
- Mengintegrasikan model ML dengan Computer Vision untuk deteksi objek.
- Mengelola dataset dan melakukan pelatihan model sederhana.

3. Landasan Teori

Machine learning merupakan cabang ilmu bagian dari kecerdasan buatan (*artificial intelligence*), dengan pemrograman untuk memungkinkan komputer menjadi cerdas berperilaku seperti manusia, dan dapat meningkatkan pemahamannya melalui pengalaman secara otomatis (Retnoningsih & Pramudita, 2020). Dimana model yang dibuat haruslah mempunyai akurasi tinggi dengan loss yang rendah sehingga hasil prediksi dapat seakurat mungkin. Dengan tujuan tersebut maka dibutuhkan data untuk melakukan proses machine learning. Dimana Data terbagi menjadi dua kategori yakni data training dan data testing. Data training berfungsi untuk melatih algoritma, sedangkan data testing berfungsi untuk menilai seberapa baik kinerja algoritma yang diajarkan sebelumnya ketika menemukan data baru (Nur Fajri et al., 2022).

Machine Learning memiliki beberapa jenis, seperti, supervised learning, unsupervised learning dan reinforcement learning (Roihan et al., 2020). Supervised learning merupakan salah satu teknik machine learning yang

menggunakan dataset (data training) yang sudah berlabel (labeled data) untuk melakukan pembelajaran pada mesin, sehingga mesin mampu mengidentifikasi label input dengan menggunakan fitur yang dimiliki untuk selanjutnya melakukan prediksi maupun klasifikasi. Klasifikasi yaitu proses mengelompokkan objek-objek dengan karakteristik yang mirip dalam beberapa kelas. Pada umumnya pengklasifikasian dokumen diwakili oleh kalimat-kalimat penting dengan menentukan ciri-ciri atau karakteristik (Septhya et al., 2023). Ada beberapa klasifikasi dalam supervised learning diantaranya *Decision Tree*, *K-Nearest Neighbor (KNN)*, *Naive Bayes*, *Regresi*, dan *Super Vector Machine* (Retnoningsih & Pramudita, 2020). Salah satu metode dalam klasifikasi adalah Super Vector Machine. Super Vector Machine berfungsi untuk memperoleh hasil prediksi pengujian, yang mana hasil dari prediksi untuk pengujian diperoleh dari kelompok yang berupa feature vector. Support Vector Machine (SVM) adalah teknik seleksi yang menghasilkan hasil dengan tingkat akurasi klasifikasi tertinggi dengan membandingkan sekumpulan parameter standar dengan nilai diskrit yang disebut sebagai himpunan kandidat. Kelebihan dari Support Vector Machine (SVM) selain dari populer juga sangat cocok untuk klasifikasi dikarenakan tidak bergantung pada jumlah atribut dan dapat menyelesaikan masalah dari dimensi (Septhya et al., 2023).

4. Analisis dan Diskusi

- Analisis Hasil

1. Bagaimana performa model dalam mendeteksi warna?

Performa model KNN dalam mendeteksi warna bisa dibilang sudah bagus. Cara kerjanya adalah membandingkan warna yang tertangkap kamera dengan data warna yang sudah ada, lalu memilih warna yang paling mirip. Kalau data latihnya sedikit, hasilnya sering kurang tepat dan tidak stabil. Tapi kalau datanya banyak, akurasi bisa sangat tinggi bahkan sampai 100% dan hasil prediksi jadi lebih konsisten. Jadi, KNN cocok dipakai untuk deteksi warna, meskipun hasilnya tetap bisa dipengaruhi oleh pencahayaan dan kualitas data.

2. Bagaimana perbedaan akurasi jika jumlah dataset ditambah?

Semakin banyak jumlah dataset yang digunakan, akurasi model akan cenderung meningkat. Hal ini karena model memiliki lebih banyak contoh data untuk belajar dan mengenali pola warna. Pada metode KNN, penambahan dataset sangat berpengaruh, karena KNN bekerja dengan cara mencari tetangga terdekat. Jika data warna lebih banyak dan beragam, peluang KNN menemukan tetangga yang benar juga semakin besar. Namun, jika dataset terlalu sedikit, akurasi KNN bisa turun drastis. Dibuktikan dengan melakukan uji coba datasheet pada KNN berikut hasilnya.

```
Evaluasi akurasi KNN dengan berbagai ukuran sampel:  
Sample size 10: Akurasi rata-rata = 77.67%, Std Dev = 14.69%  
Sample size 50: Akurasi rata-rata = 97.67%, Std Dev = 1.53%  
Sample size 100: Akurasi rata-rata = 100.00%, Std Dev = 0.00%
```

Hasil evaluasi menunjukkan bahwa ukuran sampel sangat memengaruhi kinerja KNN. Pada sampel kecil (10 data), akurasi hanya mencapai 77,67% dengan standar deviasi 14,69%, menandakan prediksi masih sering salah dan tidak konsisten. Ketika jumlah sampel ditingkatkan menjadi 50, akurasi meningkat tajam menjadi 97,67% dan standar deviasi turun drastis menjadi 1,53%, sehingga hasil prediksi lebih stabil. Dengan 100 sampel, akurasi mencapai 100% dengan standar deviasi 0%, artinya model mampu mengenali data dengan sempurna. Hal ini membuktikan bahwa semakin banyak data yang digunakan, akurasi KNN semakin tinggi dan hasil prediksi semakin konsisten.

➤ Bagaimana cara meningkatkan kinerja model klasifikasi?

Kinerja model klasifikasi seperti KNN dapat ditingkatkan dengan beberapa cara, antara lain menambah jumlah data latih agar model memiliki lebih banyak contoh untuk belajar, melakukan normalisasi data supaya setiap fitur memiliki skala yang seimbang, mencari nilai K yang optimal agar prediksi tidak terlalu sensitif maupun terlalu umum, serta melakukan augmentasi data untuk memperkaya variasi sehingga model lebih tahan terhadap perubahan seperti pencahayaan. Selain itu, pemilihan fitur yang lebih representatif, misalnya menggunakan HSV atau LAB selain RGB, juga dapat membuat klasifikasi warna lebih akurat.

Contoh peningkatan kerja klasifikasi terletak pada data dan output pada percobaan KNN. Yang dimana peningkatan klasifikasi nya ada beberapa langkah. Dari mulai tingkat non akurasi sampai akurasi maksimal.

- Diskusi

1. Apa keuntungan Machine Learning dibandingkan metode berbasis aturan (rule-based)?

Kalau pakai metode berbasis aturan, kita harus menuliskan semua aturan satu per satu. Ini ribet kalau kasusnya banyak atau rumit. Dengan Machine Learning, sistem bisa belajar sendiri dari data sehingga lebih fleksibel, akurat, dan bisa menyesuaikan dengan kondisi baru tanpa harus ditulis ulang aturannya.

2. Bagaimana ML dapat diintegrasikan lebih lanjut dalam sistem kendali? ML bisa dipakai untuk menganalisis data sensor secara real-time dan memberi keputusan yang lebih pintar. Misalnya, pada robot atau mesin otomatis, ML bisa membantu menyesuaikan pergerakan berdasarkan kondisi lingkungan, atau memprediksi kesalahan lebih awal sehingga sistem lebih efisien dan aman.

3. Apa saja tantangan dalam penerapan ML dalam sistem real-time?

Tantangannya adalah kecepatan dan keterbatasan hardware. ML butuh banyak perhitungan, jadi kalau sistemnya lambat, hasilnya bisa telat. Selain itu, data real-time sering bising (noise) atau berubah-ubah, sehingga butuh model yang benar-benar stabil. Tantangan lainnya adalah akurasi vs kecepatan, karena model yang sangat akurat biasanya lebih berat dijalankan.

5. Assignment

- Modifikasi program untuk menggunakan model ML lain (contoh: Decision Tree atau SVM)

Disini saya memodifikasi dengan menggunakan ML SVM yang dimana sebagai berikut, untuk penjelasan dari penerapan codingannya.

Kode Program	Penjelasan	Fungsi
import cv2, numpy, pandas, sklearn	Mengimpor library utama: OpenCV	Menyediakan alat bantu


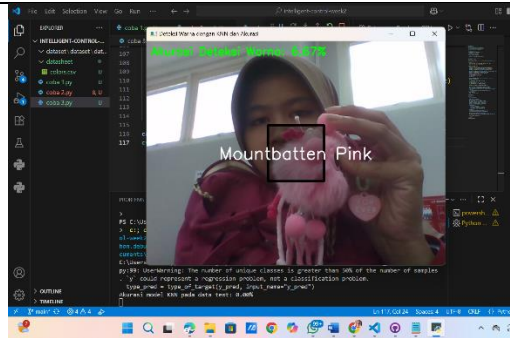
	untuk kamera & tampilan, NumPy untuk operasi matematis, Pandas untuk membaca dataset, dan Scikit-learn untuk machine learning.	utama agar program bisa jalan.
<code>color_data = pd.read_csv('datasheet/colors.csv')</code>	Membaca file CSV yang berisi daftar warna (RGB + nama). Data ini nantinya jadi bahan pembelajaran model.	Mengambil dataset warna agar bisa diproses.
<code>X = color_data[['R','G','B']].values y = color_data['color_name'].values</code>	Memisahkan data fitur (nilai R, G, B) dan label (nama warna).	Supaya model tahu mana yang jadi input (RGB) dan output (nama warna).
LabelEncoder & StandardScaler	LabelEncoder mengubah nama warna jadi angka, StandardScaler menormalkan nilai RGB agar semua fitur seimbang.	Agar data lebih mudah dipahami dan diolah model SVM.
<code>train_test_split(...)</code>	Membagi dataset jadi training (80%) dan testing (20%).	Mengecek apakah model bisa mengenali warna baru.
<code>svm_model = SVC(kernel='rbf', C=10, gamma='scale')</code>	Membuat model Support Vector Machine dengan kernel RBF yang cocok untuk data non-linear seperti warna.	Mesin utama untuk klasifikasi warna.
<code>svm_model.fit(X_train, y_train)</code>	Melatih model dengan data	Model belajar pola dari data

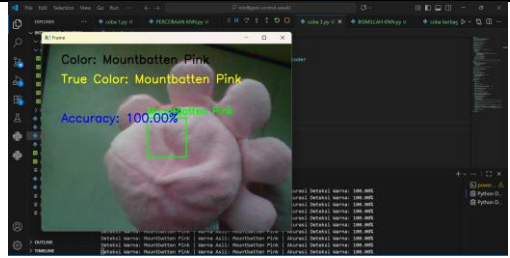
	training.	RGB dan nama warna.
<code>accuracy_score(y_test, y_pred)</code>	Menghitung akurasi model dari hasil testing.	Mengukur seberapa benar prediksi model.
<code>def find_nearest_color(rgb_color): ...</code>	Fungsi tambahan untuk mencari warna terdekat berdasarkan jarak Euclidean.	Jadi pembanding warna asli selain prediksi SVM.
<code>cap = cv2.VideoCapture(0)</code>	Mengaktifkan kamera laptop/PC untuk menangkap video.	Menyediakan input real-time dari kamera.
<code>roi1, roi2</code>	Menentukan area kotak (Region of Interest) di sisi kiri & kanan frame.	Area tempat sistem membaca warna objek.
<code>avg_color1 = np.mean(roi1, ...)</code>	Menghitung rata-rata warna dalam kotak.	Mendapatkan nilai RGB yang mewakili isi kotak.
<code>svm_model.predict(...)</code>	Menggunakan model SVM untuk memprediksi nama warna dari nilai RGB.	Menentukan nama warna hasil deteksi.
<code>accuracy_score(detected_true_labels, detected_colors)</code>	Membandingkan warna asli dan prediksi SVM secara live.	Mengukur akurasi deteksi secara langsung.
<code>cv2.rectangle, cv2.putText</code>	Menggambar kotak dan menampilkan teks warna di layar kamera.	Visualisasi hasil agar mudah dilihat.
<code>cv2.imshow('Frame', frame)</code>	Menampilkan jendela video dengan hasil deteksi warna.	Memberikan tampilan real-time ke pengguna.
<code>if cv2.waitKey(1) & 0xFF ==</code>	Program berhenti	Memberikan

ord('q'):	jika tombol q ditekan.	kontrol untuk keluar dari aplikasi.
cap.release(), cv2.destroyAllWindows()	Menutup kamera dan semua jendela OpenCV.	Membersihkan resource setelah program selesai.

6. Data dan Output Hasil Pengamatan

- Hasil modifikasi kode program untuk deteksi warna lain

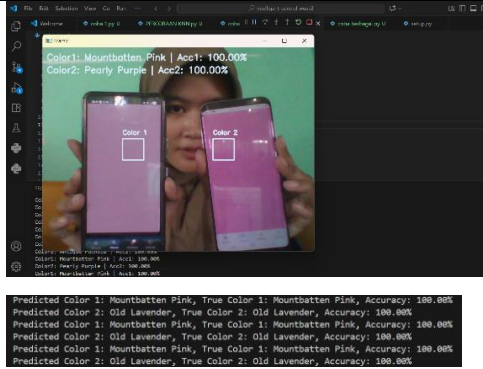
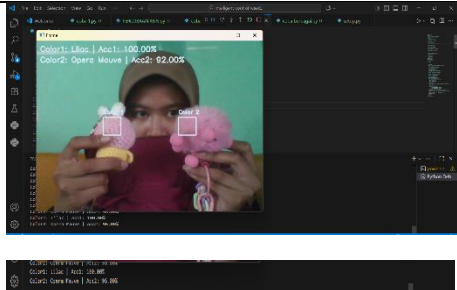
Percobaan KNN	
Keterangan	Gambar
Pada percobaan pertama, menggunakan codingan dari file canva diaplikasikan dengan data color.csv yang mana sistem mampu mengenali warna objek secara langsung melalui kamera dan memberikan output teks warna yang terdeteksi . Namun masih belum ada keterangan terkait tingkat keakurasiannya.	
Pada percobaan kedua, menggunakan codingan dari file warna2.py diaplikasikan dengan data color.csv. Sistem mampu mengenali warna objek secara langsung melalui kamera dan memberikan output teks warna yang terdeteksi , yaitu <i>Mountbatten Pink</i> . Selain itu, pada tampilan juga sudah	

ditambahkan keterangan tingkat akurasi deteksi warna sebesar 6,67%. Namun untuk tingkat keakurasian masih rendah.	
Codingan kedua diperbaiki dengan menambahkan label encoding, augmentasi data, serta pencarian K terbaik. Warna diambil dari rata-rata area (ROI), bukan satu piksel, dan akurasi dihitung dari banyak frame, sehingga hasilnya lebih stabil akurat, dan terdeteksi .	

Pada percobaan pertama, program yang dibuat dari file *canva* dengan data *colors.csv* sudah bisa mendeteksi warna objek lewat kamera dan menampilkan teks warna yang sesuai. Tapi, di tahap ini belum ada informasi soal akurasi, jadi kita belum tahu seberapa tepat hasil deteksinya. Sedangkan pada percobaan kedua, program dari file *warna2.py* dengan data yang sama sudah lebih berkembang karena selain menampilkan warna yang terdeteksi, misalnya “Mountbatten Pink”, juga ditambahkan keterangan tingkat akurasi, yaitu 6,67%. Walaupun begitu, tingkat akurasinya masih rendah sehingga perlu ditingkatkan lagi, misalnya dengan menambah data latih atau memperbaiki cara pengolahan datanya supaya hasil deteksinya bisa lebih tepat.

Pada codingan kedua terdapat beberapa perbaikan dibanding codingan pertama, yaitu penambahan label encoding agar warna lebih mudah diproses oleh model, augmentasi data untuk memperbanyak variasi warna sehingga model lebih tahan terhadap perubahan pencahayaan, serta pencarian nilai K terbaik sehingga klasifikasi lebih optimal. Selain itu, pengambilan warna dilakukan dengan menghitung rata-rata area (ROI), bukan hanya satu piksel, sehingga hasil deteksi lebih stabil. Akurasinya juga dihitung dari banyak frame secara berulang, bukan sekali uji, sehingga lebih mencerminkan kinerja

model secara keseluruhan.

Percobaan SVM	
<p>Dengan metode SVM, sistem diuji untuk mendeteksi dua warna secara bersamaan dari dua smartphone berbeda. Hasil nya terdeteksi menunjukkan Color1: <i>Mountbatten Pink</i> dan Color2: <i>Pearly Purple</i>, dengan tingkat akurasi masing-masing 100%. Hal ini menunjukkan SVM mampu memberikan hasil klasifikasi warna yang lebih akurat dibandingkan percobaan sebelumnya.</p>	 <p>Predicted Color 1: Mountbatten Pink, True Color 1: Mountbatten Pink, Accuracy: 100.00%</p> <p>Predicted Color 2: Old Lavender, True Color 2: Old Lavender, Accuracy: 100.00%</p> <p>Predicted Color 1: Mountbatten Pink, True Color 1: Mountbatten Pink, Accuracy: 100.00%</p> <p>Predicted Color 2: Old Lavender, True Color 2: Old Lavender, Accuracy: 100.00%</p> <p>Predicted Color 1: Mountbatten Pink, True Color 1: Mountbatten Pink, Accuracy: 100.00%</p> <p>Predicted Color 2: Old Lavender, True Color 2: Old Lavender, Accuracy: 100.00%</p>
<p>Program terdeteksi warna pada dua objek yang ditunjukkan ke kamera. Hasil deteksi bisa berbeda dengan warna aslinya karena pencahayaan memengaruhi tangkapan kamera.</p>	 <p>Color 1: Lilac Acc1: 100.00%</p> <p>Color 2: Opera House Acc2: 92.00%</p> <p>Predicted Color 1: Lilac, True Color 1: Lilac, Accuracy: 100.00%</p> <p>Predicted Color 2: Opera House, True Color 2: Opera House, Accuracy: 92.00%</p>

Percobaan dengan metode SVM dilakukan untuk mendeteksi dua warna sekaligus dari dua smartphone yang berbeda. Hasilnya, sistem berhasil mengenali warna pertama sebagai *Mountbatten Pink* dan warna kedua sebagai *Pearly Purple*, dengan akurasi 100% untuk keduanya. Hal ini menunjukkan bahwa SVM dapat mengenali warna dengan sangat tepat dan konsisten, bahkan saat ada lebih dari satu objek di layar. Jika dibandingkan dengan percobaan sebelumnya yang menggunakan KNN, metode SVM jelas lebih baik karena mampu memberikan hasil yang lebih akurat dan stabil.

7. Kesimpulan

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

1. Percobaan menunjukkan bahwa KNN mampu mendeteksi warna dengan

cukup baik, namun akurasi sangat bergantung pada jumlah dan kualitas dataset. Semakin banyak data latih yang digunakan, semakin tinggi pula akurasi model, bahkan bisa mencapai 100% ketika dataset besar dan bervariasi.

2. Metode SVM memberikan hasil klasifikasi yang lebih stabil dan akurat dibandingkan KNN, terbukti dengan akurasi 100% saat mendeteksi dua warna sekaligus. Hal ini membuktikan bahwa SVM lebih cocok digunakan pada kasus deteksi warna berbasis RGB.
3. Faktor pencahayaan dan kualitas kamera tetap memengaruhi hasil deteksi, sehingga meskipun model sudah baik, kondisi lingkungan juga perlu diperhatikan.
4. Integrasi machine learning dengan computer vision terbukti dapat digunakan untuk mendeteksi warna secara real-time, sehingga dapat diaplikasikan pada sistem kendali maupun aplikasi berbasis visi komputer lainnya.

8. Saran

1. Dataset warna sebaiknya ditambah dan dibuat lebih bervariasi supaya hasil deteksi lebih akurat.
2. Selain pakai RGB, bisa coba pakai warna lain seperti **HSV** atau **LAB** biar lebih stabil kalau cahaya berubah-ubah.
3. Program real-time sebaiknya dibuat lebih ringan dan cepat, jadi tidak terlalu berat saat dijalankan.
4. Bisa coba bandingkan dengan metode lain (misalnya Decision Tree, Random Forest, atau CNN) untuk lihat mana yang hasilnya paling bagus.
5. Ke depan, sistem ini bisa dipakai untuk aplikasi nyata, misalnya alat sortir barang berdasarkan warna atau cek kualitas produk.

9. Daftar Pustaka

Nur Fajri, F., Tholib, A., & Yuliana, W. (2022). Application of Machine Learning Algorithm for Determining Elective Courses in Informatics Study Program. *Jurnal Teknik Informatika dan Sistem Informasi*, 8(3). <https://doi.org/10.28932/jutisi.v8i3.3990>

- Retnoningsih, E., & Pramudita, R. (2020). Mengenal Machine Learning Dengan Teknik Supervised Dan Unsupervised Learning Menggunakan Python. *BINA INSANI ICT JOURNAL*, 7(2), 156. <https://doi.org/10.51211/biict.v7i2.1422>
- Roihan, A., Sunarya, P. A., & Rafika, A. S. (2020). Pemanfaatan Machine Learning dalam Berbagai Bidang: Review paper. *IJCIT (Indonesian Journal on Computer and Information Technology)*, 5(1). <https://doi.org/10.31294/ijcit.v5i1.7951>
- Septhya, D., Rahayu, K., Rabbani, S., Fitria, V., Rahmaddeni, R., Irawan, Y., & Hayami, R. (2023). Implementasi Algoritma Decision Tree dan Support Vector Machine untuk Klasifikasi Penyakit Kanker Paru: Implementation of Decision Tree Algorithm and Support Vector Machine for Lung Cancer Classification. *MALCOM: Indonesian Journal of Machine Learning and Computer Science*, 3(1), 15–19. <https://doi.org/10.57152/malcom.v3i1.591>