

Laporan Praktikum Kontrol Cerdas

Nama : Nefi Afif Sujatyana
NIM : 224308093
Kelas : TKA-7D
Akun Github : <https://github.com/nefiafif612-cmd>

1. Judul Percobaan:

Machine Learning for Control Systems

2. Tujuan Percobaan:

- Mahasiswa mampu memahami dasar-dasar Machine Learning dalam sistem kendali.
- Mahasiswa dapat mengimplementasikan model ML sederhana untuk klasifikasi objek.
- Mahasiswa dapat menggunakan Scikit-learn untuk membuat model ML dasar.
- Mahasiswa dapat mengintegrasikan model ML dengan Computer Vision untuk deteksi objek.
- Mahasiswa dapat mengelola dataset dan melakukan pelatihan model sederhana.

3. Landasan Teori:

Machine Learning adalah salah satu bidang cabang ilmu komputer yang memberikan kemampuan kepada komputer untuk dapat belajar tanpa diprogram secara eksplisit (**Daqiqil, 2021**). Untuk bisa mengaplikasikan teknik-teknik machine learning maka harus ada data. Tanpa data maka algoritma machine learning tidak dapat bekerja. Data yang ada biasanya dibagi menjadi dua, yaitu data training dan data testing. Data training digunakan untuk melatih algoritma, sedangkan data *testing* digunakan untuk mengetahui performa algoritma yang telah dilatih sebelumnya ketika menemukan data baru yang belum pernah dilihat. *Machine Learning* sendiri

terbagi menjadi beberapa algoritma yang berbeda-beda dan setiap dari algoritma tersebut memiliki fungsi dan tujuannya masing-masing, seperti *supervised learning*, *unsupervised learning*, dan *reinforcement learning* (Karimah, 2023). *Supervised Learning* adalah algoritma yang menyimpulkan fungsi dari data pelatihan berlabel yang terdiri dari sekumpulan contoh pelatihan, misalnya seperti *logistic regression* dan *K-Nearest Neighbors* (KNN). *Unsupervised Learning* adalah algoritma yang bertujuan untuk meningkatkan kinerja tugas pembelajaran yang diawasi dimana kita tidak memiliki banyak data, contohnya seperti *k-means*, *Apriori*, DBSCAN, dan *clustering*. *Reinforcement Learning* adalah algoritma yang mengumpulkan pengetahuan (sinyal penguatan) untuk memilih tindakan yang mengarah ke hasil tertinggi yang diharapkan. Dibandingkan dengan metodologi lain dalam *Machine Learning*, algoritma *Reinforcement Learning* memiliki kelebihan yang berbeda yaitu kemampuan untuk secara mandiri menjelajahi lingkungan yang sangat dinamis dan stokastik dan mengembangkan, dengan mengumpulkan umpan balik evaluatif dari lingkungan, kebijakan kontrol yang optimal (Fathurohman, 2021). *Machine Learning* memungkinkan sistem kendali untuk belajar dari data dan meningkatkan performanya seiring waktu. Beberapa contoh penerapannya yaitu *Predictive Maintenance* (mendeteksi potensi kegagalan sistem sebelum terjadi), *Adaptive Control* (sistem kendali yang dapat menyesuaikan parameter secara otomatis), dan *Anomaly Detection* (mendeteksi perilaku tidak normal dalam sistem industri) (Roihan dkk., 2020).

4. Analisis dan Diskusi:

- Analisis

Percobaan praktikum pada minggu kedua ini adalah mendeteksi objek berwarna menggunakan kamera dengan **Support Vector Machine** (SVM). Data dari warna tersebut adalah RGB dan nama warna. Kemudian, data akan diolah dengan **LabelEncoder** untuk mengkonversi label teks menjadi angka sehingga serasi dengan model SVM. Untuk fitur RGB ini menggunakan **StandardScaler** untuk meningkatkan akurasi model yang dilatih dengan menggunakan SVM dengan kernel RBF (Radial Basis

Function) yang mana dipilih karena kemampuan dalam menangani distribusi data non-linear (warna dalam RGB). Data dibagi menjadi *training* dan *testing* menggunakan *train_test_split*. Akurasi model akan diuji pada data testing dan hasilnya akan ditampilkan pada terminal VSCode. Selanjutnya, pada bagian real time color detection, program menggunakan OpenCV untuk mengakses kamera dan mengambil dua **ROI (Region of Interest)** pada layar yaitu, satu sisi sebelah kiri dan satu sisi sebelah kanan. Kemudian, rata-rata warna dalam **ROI** dihitung untuk prediksi oleh model SVM dan prediksi ini akan dibandingkan dengan warna yang terdekat pada dataset untuk menghitung akurasi secara *running accuracy* berdasarkan 50 prediksi terakhir. Dan hasilnya akan ditampilkan pada frame layar kamera dengan *bounding box* dan teks yang akan menunjukkan nama warna serta tingkat keakurasian warna yang terdeteksi.

- Diskusi

Pada program yang dipakai ini memanfaatkan SVM dengan **kernel RBF** untuk mengklasifikasikan warna berdasarkan warna RGB. Penggunaan **StandardScaler** untuk membantu model berkonversi lebih cepat dan memberikan hasil yang akurat. **Kernel RBF** mampu menangani distribusi warna yang kompleks pada RGB, sehingga hasilnya lebih akurat dalam pengklasifikasian warna. Selain itu, dengan *hyperparameter* berupa **C=10** digunakan untuk mengontrol regulasi dan memberikan keseimbangan yang baik. Kemudian, untuk penggunaan **ROI** memungkinkan deteksi warna pada dua titik berbeda pada simultan, dan perhitungan *running accuracy* memberikan evaluasi secara lebih akurat dan dinamis, serta stabil. Namun, model ini akan lebih rentan terhadap perubahan cahaya dan kualitas kamera. Akurasi prediksi warna dapat terpengaruh oleh pencahayaan lingkungan yang tidak konsisten. Oleh karena itu, *preprocessing* tambahan dapat membantu meningkatkan akurasi diberbagai kondisi pencahayaan. Secara keseluruhan, program ini menunjukkan implementasi *color detection* dengan **SVM** dan **OpenCV**.

5. Assignment:

Praktikum di minggu kedua ini yaitu melakukan sejumlah tugas yaitu (penugasan variabel) untuk menyimpan data dan konfigurasi penting yang diperlukan dalam deteksi warna. Pertama, **file_path** diisi dengan lokasi dataset dengan format warna CSV, yang selanjutnya dibaca lewat '**pd.read_csv()**' dan disimpan dalam bentuk variabel **color_data**. Kemudian, **X** diisi dengan nilai RGB yang diambil dari kolom 'R',

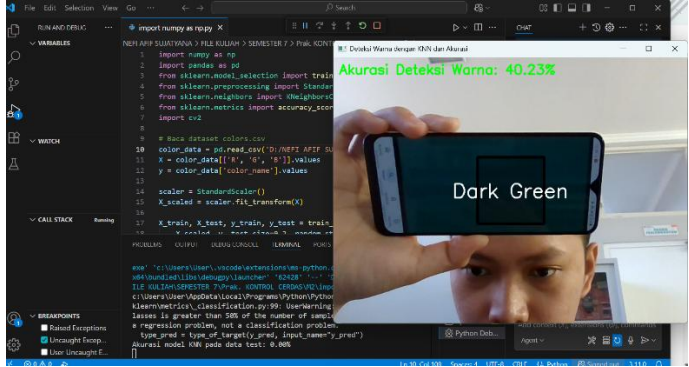
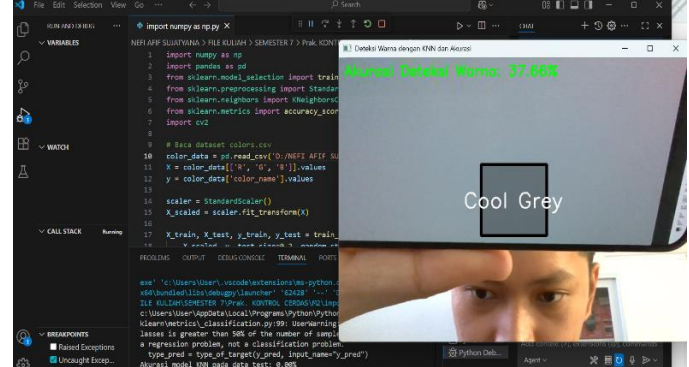
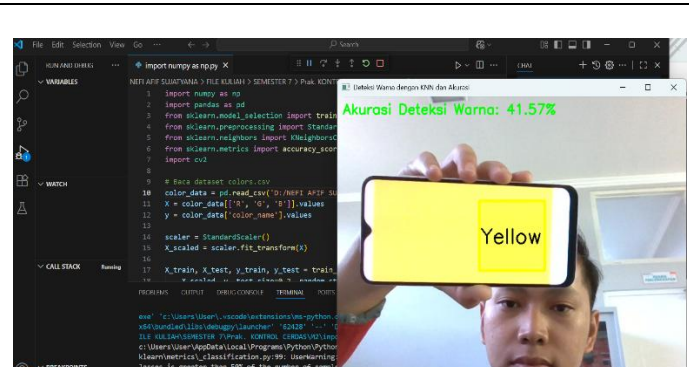
'G', dan 'B', sementara *y* menyimpan label warna yang terdapat dalam kolom '**ColorName**'. Karena label berbentuk teks, **LabelEncoder** diterapkan untuk mengubah label tersebut menjadi bentuk numerik yang disimpan dalam variabel **y_encoded**. Fitur RGB kemudian dinormalisasi menggunakan **StandardScaler**, dan hasilnya disimpan di **X_scaled** agar model SVM beroperasi secara optimal. Selanjutnya, dataset dipisahkan menjadi data pelatihan dan pengujian dengan menggunakan '**train_test_split()**', dan hasilnya disimpan dalam variabel **X_train**, **X_test**, **y_train**, serta **y_test**.

Dalam bagian model *machine learning*, **svm_model** mendapatkan objek '**SVC()**' yang diatur dengan **kernel RBF**, serta parameter '**C=10**' dan '**gamma=scale**'. Model ini kemudian dilatih dengan data pelatihan, dan hasil prediksinya disimpan dalam **y_pred** untuk tujuan evaluasi akurasi. Untuk deteksi warna secara *real-time*, diisi dengan objek '**cv2.VideoCapture(0)**' guna mengakses kamera. Variabel **detected_colors_1**, **detected_colors_2**, **detected_true_labels_1**, dan **detected_true_labels_2** digunakan untuk menyimpan hasil prediksi serta label asli secara dinamis, yang kemudian dipakai untuk menghitung *running accuracy*.

Setiap kali satu frame diambil dari kamera, dua **Region of Interest (ROI)** dihitung dan disimpan dalam **ROI1** dan **ROI2**, sementara warna rata-rata dalam ROI tersimpan di **avg_color1** dan **avg_color2**. Warna rata-rata ini dibalik urutannya dari **BGR** ke **RGB** dan dinormalisasi sebelum digunakan untuk prediksi warna oleh model SVM. Hasil prediksi disimpan dalam **color_pred1** dan **color_pred2**, yang kemudian dibandingkan dengan warna terdekat dari dataset melalui fungsi **find_nearest_color()**.

Hasil prediksi serta akurasi ditampilkan di layar kamera menggunakan OpenCV, dengan informasi yang diperbarui secara dinamis berdasarkan *running accuracy* dari 50 prediksi terakhir. Penugasan variabel yang terstruktur dengan baik dalam program ini memungkinkan aliran data yang lancar dari *preprocessing*, *training model*, hingga prediksi *real-time*.

6. Data dan Output Hasil Pengamatan:

No	Variabel	Hasil Pengamatan
1.	Pada percobaan pertama, saya menggunakan warna hijau untuk dilakukan uji coba. Ternyata program mendeteksi warna tersebut sebagai warna dark green dengan akurasi warna 40,23%.	 The screenshot shows a Jupyter Notebook with Python code for SVM classification. The code imports necessary libraries, loads a color dataset, scales the features, and trains a model. A video inset shows a person holding a smartphone that displays 'Dark Green' with an accuracy of 40.23%.
2.	Pada percobaan kedua, saya menggunakan warna putih dan melakukan uji coba. Didapatkan hasil warna tersebut termasuk dalam warna cool grey dengan akurasi warna 37.66%	 The screenshot shows the same SVM code in a Jupyter Notebook. A video inset shows a person holding a smartphone that displays 'Cool Grey' with an accuracy of 37.66%.
3.	Pada percobaan ketiga, saya menggunakan warna kuning untuk dilakukan uji coba. Didapatkan hasil warna tersebut termasuk dalam warna yellow dengan akurasi warna 41.57%.	 The screenshot shows the same SVM code in a Jupyter Notebook. A video inset shows a person holding a smartphone that displays 'Yellow' with an accuracy of 41.57%.

7. Kesimpulan:

- Program ini untuk mendeteksi warna dengan menggunakan *Support Vector Machine* (SVM) dengan kernel RBF.

- Menggunakan StandardScaler membantu meningkatkan akurasi model, sedangkan LabelEncoder untuk pengubahan label warna dari teks menjadi angka yang serasi dengan SVM.
- *Running accuracy* memberikan evaluasi hasil deteksi model yang lebih akurat dan dinamis.
- *Region of Interest* (ROI) memungkinkan deteksi warna pada dua area yang bersamaan, sehingga meningkatkan fungsi aplikasi pendeteksi warna.
- Hasil prediksi dan keakuratan objek warna bergantung pada pencahayaan dan kualitas kamera.

8. Saran:

Untuk meningkatkan akurasi yang tepat, terdapat beberapa hal yang dapat dilakukan yaitu di antaranya, menggunakan *color correction* pada tahap *processing*. Kemudian, agar agar model lebih adaptif terhadap perubahan pencahayaan dengan menambahkan *adaptive thresholding*. Selain itu, perluas dataseet warna dengan lebih banyak variasi untuk meningkatkan kemampuan generalisasi model dalam mengenali warna lebih akurat.

9. Daftar Pustaka:

- Daqiqil, I. (2021). *Machine Learning: Teori, Studi Kasus, dan Implementasi Menggunakan Python*. Riau: UR PRESS.
- Fathurohman, A. (2021). Machine Learning untuk Pendidikan: Mengapa dan Bagaimana. *I*(3).
- Karimah Tauhid, Volume 2 Nomor 1 (2023), e-ISSN 2963-590X. (2023). 2.
- 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>