

# Laporan Praktikum Kontrol Cerdas

Nama : Aureyza Pandu Qinara  
NIM : 224308078  
Kelas : TKA - 7D  
Akun Github (Tautan) : <https://github.com/aureyzapandu>  
Student Lab Assistant :-

Judul Percobaan

Week 3: Deep Learning for Intelligent Control Systems

## 1. Tujuan Percobaan

Tujuan dari praktikum “*Deep Learning for Intelligent Control Systems*”, mahasiswa diharapkan mampu:

- a) Memahami konsep dasar *Deep Learning* dalam sistem kendali.
- b) Mengimplementasikan *Convolutional Neural Network* (CNN) untuk klasifikasi objek.
- c) Menggunakan *TensorFlow* dan *Keras* dalam membangun model *Deep Learning*.
- d) Mengintegrasikan model CNN dengan *Computer Vision* (OpenCV) untuk deteksi objek secara *real-time*.
- e) Menguji kemampuan model dengan menambahkan mode *Night Vision*, sehingga objek dapat terdeteksi pada kondisi pencahayaan rendah.

## 2. Landasan Teori

- *Deep Learning* dalam Sistem Kendali

*Deep Learning* (DL) merupakan cabang dari *Machine Learning* yang menggunakan jaringan saraf tiruan dengan banyak lapisan untuk mempelajari representasi data secara otomatis dari data mentah. Dalam sistem kendali, DL berperan penting dalam meningkatkan kemampuan pengenalan pola dan pengambilan keputusan secara otomatis. Contohnya, DL digunakan dalam kendaraan otonom untuk mendeteksi dan mengenali objek di jalan, serta dalam robotika untuk navigasi berbasis visi komputer. Selain itu, DL juga diaplikasikan dalam pengawasan cerdas dan kontrol kualitas industri, serta

pengembangan sistem *night vision* untuk deteksi objek dalam kondisi pencahayaan rendah (LeCun et al., 2015).

- *Convolutional Neural Network (CNN)*

CNN adalah jenis jaringan saraf tiruan yang dirancang khusus untuk pengolahan data citra dengan memanfaatkan struktur spasial data. CNN menggunakan lapisan konvolusi untuk mengekstrak fitur lokal dari gambar, yang memungkinkan model belajar representasi hierarkis dari data visual secara otomatis tanpa perlu ekstraksi fitur manual (Krizhevsky et al., 2017).

Struktur CNN terdiri dari :

- a. *Convolutional Layer*: Melakukan operasi konvolusi dengan filter (kernel) untuk mengekstrak fitur lokal dari gambar, seperti tepi, sudut, dan tekstur.
- b. *Activation Function (ReLU)*: Fungsi aktivasi ReLU (*Rectified Linear Unit*) diterapkan setelah konvolusi untuk memperkenalkan non-linearitas dan mempercepat proses pelatihan.
- c. *Pooling Layer (Max Pooling)*: Mengurangi dimensi fitur dengan mengambil nilai maksimum dari area tertentu, sehingga mengurangi kompleksitas komputasi dan membantu menghindari *overfitting*.
- d. *Fully Connected Layer*: Menghubungkan fitur yang telah diekstrak ke neuron output untuk klasifikasi akhir.
- e. *Softmax Layer*: Menghasilkan probabilitas kelas untuk setiap kategori yang diprediksi.

- *Preprocessing Data & Augmentasi*

*Preprocessing data* merupakan tahap penting dalam pelatihan model *Deep Learning*, yang meliputi normalisasi data agar nilai pixel berada dalam rentang yang sesuai untuk mempercepat konvergensi model. Augmentasi data dilakukan untuk memperbesar variasi data pelatihan dengan teknik seperti rotasi, zoom, dan flipping horizontal. Hal ini membantu model menjadi lebih *robust* terhadap variasi data nyata dan mengurangi risiko *overfitting* (Shorten and Khoshgoftaar, 2019).

- *Training Model CNN*

*Training* model CNN melibatkan proses *forward propagation* untuk menghasilkan prediksi dan *backpropagation* untuk memperbarui bobot jaringan berdasarkan fungsi *loss*, seperti *categorical crossentropy* untuk klasifikasi multi-kelas (LeCun et al., 2015)

- Integrasi CNN dengan *Computer Vision*

Integrasi CNN dengan *Computer Vision* memungkinkan deteksi objek secara *real-time* menggunakan kamera. OpenCV digunakan untuk menangkap video dan melakukan *preprocessing* gambar sebelum diprediksi oleh model CNN. Mode *night vision* dikembangkan dengan mengubah citra menjadi *grayscale* dan menerapkan *color map* untuk meningkatkan visibilitas objek dalam kondisi pencahayaan rendah. Pendekatan ini memungkinkan sistem kendali untuk beroperasi efektif dalam berbagai kondisi lingkungan

### 3. Analisis dan Diskusi

Program yang dibuat merupakan implementasi *Convolutional Neural Network* (CNN) untuk klasifikasi citra secara *real-time* menggunakan OpenCV. Model CNN yang digunakan telah dilatih dengan *dataset Intel Image Classification* yang terdiri dari enam kelas, yaitu *buildings*, *forest*, *glacier*, *mountain*, *sea*, dan *street*.

Pada proses *real-time*, kamera laptop diaktifkan untuk menangkap citra secara langsung. Setiap frame yang masuk diproses melalui beberapa tahapan, yakni *preprocessing* berupa perubahan ukuran gambar menjadi (150x150) piksel, normalisasi ke skala [0–1], dan penyesuaian dimensi agar sesuai dengan format input CNN. Hasil prediksi kemudian ditampilkan pada layar dalam bentuk label kelas yang terdeteksi.

Selain itu, program ini dilengkapi dengan fitur *Night Vision* yang bekerja dengan cara mengubah citra ke skala abu-abu kemudian memberikan efek *color map* (COLORMAP\_JET) sehingga tampilan menyerupai penglihatan malam. Fitur ini memungkinkan sistem tetap dapat digunakan untuk deteksi objek meskipun dalam kondisi pencahayaan rendah, sesuai dengan tujuan praktikum Week 3.

Perbedaan utama dengan modul adalah pada saat pelatihan model. Jika

pada modul CNN dilatih selama 10 epoch, dalam implementasi ini digunakan 5 epoch. Alasan penggunaan epoch 5 adalah untuk efisiensi waktu training, mengingat proses pelatihan CNN membutuhkan sumber daya komputasi yang besar. Dengan mengurangi epoch, model dapat selesai dilatih lebih cepat tanpa mengurangi esensi praktikum, yaitu integrasi CNN dengan *Computer Vision* secara *real-time*. Selain itu, penggunaan epoch lebih sedikit juga membantu mengurangi risiko *overfitting*, sehingga model tetap dapat melakukan generalisasi dengan baik pada data baru seperti input kamera.

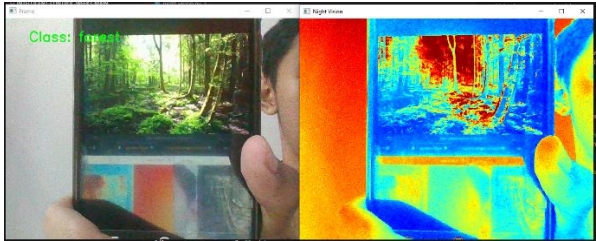
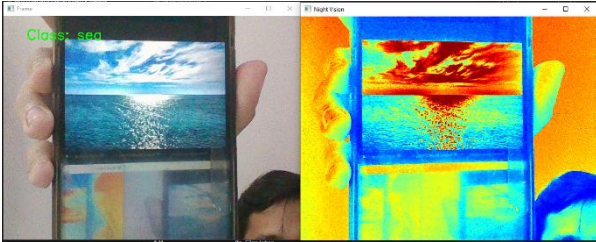
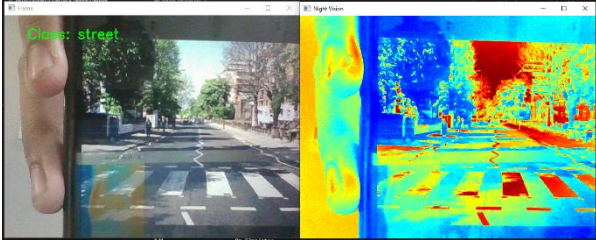
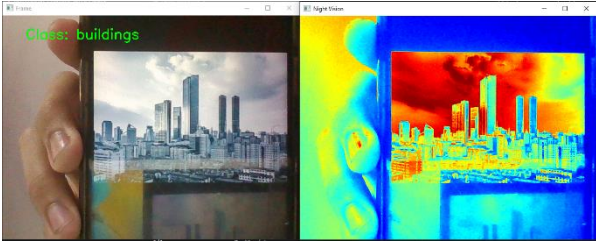
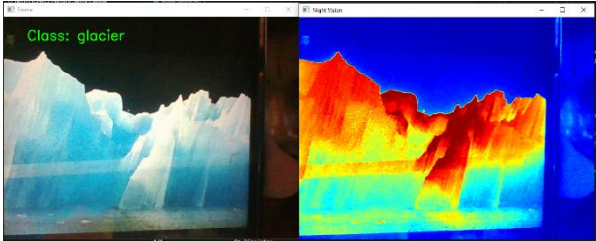
Secara keseluruhan, program yang dibuat telah sesuai dengan learning objectives Practice Week 3, yaitu memahami konsep CNN, mengintegrasikannya dengan Computer Vision, menambahkan fitur *Night Vision*, serta menjalankan sistem klasifikasi secara *real-time*.

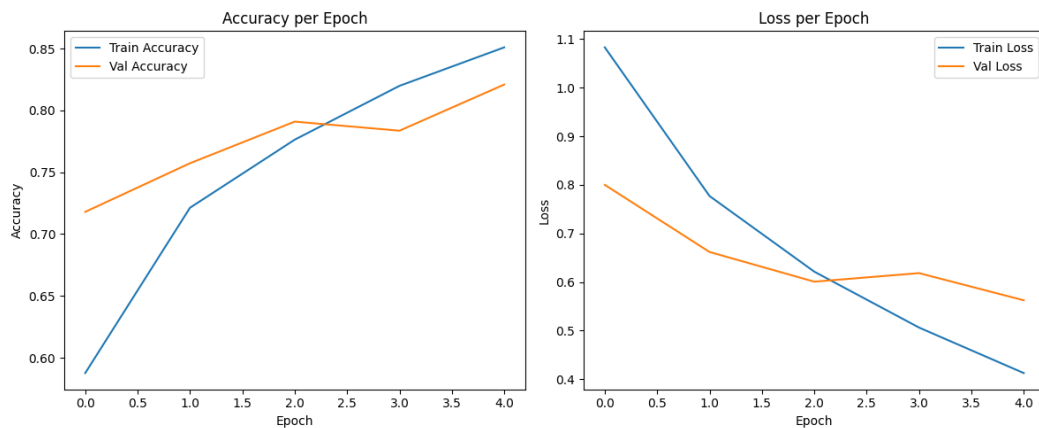
#### 4. Assignment

1. Melakukan studi literatur mengenai konsep dasar yang mendukung percobaan, antara lain *Convolutional Neural Network* (CNN) untuk klasifikasi citra, OpenCV sebagai pustaka pengolahan video *real-time*, *preprocessing data* gambar (*resize* dan *normalisasi*), mode *night vision* dengan konversi *grayscale* dan *color map*, serta integrasi model *TensorFlow/Keras* untuk prediksi kelas objek.
2. Membuat akun GitHub sebagai sarana penyimpanan kode program berbasis *version control system*.
3. Menginstal perangkat lunak pendukung, yaitu *VSCode*, *Python*, *Git*, *TensorFlow/Keras*, dan OpenCV.
4. Membuat repository di GitHub kemudian melakukan clone repository ke laptop.
5. Membuat kode program deteksi objek *real-time* menggunakan CNN dengan mode *night vision*, termasuk *load model*, *capture* video dari kamera, *preprocessing frame*, prediksi kelas, dan tampilan hasil pada dua jendela (*Frame* dan *Night Vision*).
6. Menjalankan program untuk menampilkan hasil deteksi pada jendela *Frame* dengan label kelas dan jendela *Night Vision* secara *real-time*.

7. Menghentikan program dengan menekan tombol q pada keyboard.
8. Melakukan *commit* dan *push* kode program ke *repository* GitHub sebagai dokumentasi akhir.

## 5. Data dan Output Hasil Pengamatan

No	Variabel	Hasil Pengamatan
1	Pengujian model <i>deep learning</i> klasifikasi <i>forest</i>	
2	Pengujian model <i>deep learning</i> klasifikasi <i>sea</i>	
3	Pengujian model <i>deep learning</i> klasifikasi <i>steet</i>	
4	Pengujian model <i>deep learning</i> klasifikasi <i>building</i>	
5	Pengujian model <i>deep learning</i> klasifikasi <i>glacier</i>	



Hasil training menunjukkan bahwa grafik akurasi cenderung meningkat di setiap epoch, meskipun kenaikannya tidak mulus (terlihat fluktuasi kecil) karena menggunakan 5 epoch. Kenaikan akurasi yang tidak sepenuhnya linier menandakan bahwa model sedang menyesuaikan bobot (*weights*) berdasarkan data yang diberikan, sehingga pada epoch tertentu akurasi dapat melonjak signifikan, sementara pada epoch lain kenaikannya lebih lambat.

Pada grafik *loss* menunjukkan pola yang berlawanan. Nilai *loss* cenderung menurun dari epoch ke epoch, meskipun juga tidak sepenuhnya linier. Hal ini menunjukkan bahwa kesalahan prediksi model semakin berkurang seiring bertambahnya epoch, sehingga model menjadi lebih baik dalam mengenali pola pada dataset.

## 6. Kesimpulan

- Konsep dasar *Deep Learning* berhasil dipahami melalui implementasi CNN untuk klasifikasi gambar, di mana model mampu belajar mengekstraksi fitur penting dari dataset citra.
- Model CNN yang dibangun dengan TensorFlow dan Keras menunjukkan tren peningkatan akurasi dan penurunan *loss* pada setiap epoch, meskipun grafik tidak sepenuhnya linier. Hal ini menunjukkan model belajar secara bertahap dari data yang diberikan.
- Integrasi CNN dengan OpenCV berjalan dengan baik, memungkinkan sistem mendeteksi objek secara *real-time* dari kamera.
- Penambahan fitur *Night Vision* terbukti meningkatkan visibilitas objek pada kondisi pencahayaan rendah, sehingga deteksi tetap dapat dilakukan secara efektif.

## 7. Saran

- a) Menambah jumlah epoch saat training agar model CNN memiliki waktu belajar yang lebih optimal sehingga akurasi lebih stabil.
- b) Memperbesar dan memperluas dataset (misalnya dengan menambah variasi kondisi pencahayaan dan sudut pandang objek) agar model lebih *robust* dalam mengenali pola.
- c) Menerapkan data *augmentation* yang lebih bervariasi, seperti rotasi, perubahan kontras, dan *shear transformation* untuk meningkatkan generalisasi model.

## 8. Daftar Pustaka

- Krizhevsky, A., Sutskever, I., Hinton, G.E., 2017. ImageNet classification with deep convolutional neural networks. *Commun. ACM* 60, 84–90.  
<https://doi.org/10.1145/3065386>
- LeCun, Y., Bengio, Y., Hinton, G., 2015. Deep learning. *Nature* 521, 436–444.  
<https://doi.org/10.1038/nature14539>
- Shorten, C., Khoshgoftaar, T.M., 2019. A survey on Image Data Augmentation for Deep Learning. *J. Big Data* 6, 60. <https://doi.org/10.1186/s40537-019-0197-0>