Laporan Praktikum Kontrol Cerdas Minggu Ke-3

Nama: M Ikhsan Nurdin

NIM: 234308103

Kelas: TKA-6D

Akun Github: https://github.com/icannn10/Praktikum-Skicit_Learn

Pendahuluan

MediaPipe merupakan sebuah framework yang dikembangkan untuk membangun sistem pemrosesan data berbasis machine learning dalam berbagai bentuk media seperti video, gambar, maupun audio. Framework ini dirancang untuk mempermudah pembuatan pipeline pemrosesan yang terstruktur sehingga proses deteksi dan analisis dapat berjalan secara efisien dan real-time.

Pada praktikum kali ini mendeteksi dan mengenali gerakan tangan secara otomatis menggunakan webcam. Sistem bekerja dengan mengambil gambar tangan secara real-time, kemudian mendeteksi titik-titik landmark pada tangan menggunakan MediaPipe. Data koordinat landmark tersebut diproses dan dinormalisasi sebelum dimasukkan ke dalam model machine learning yang telah dilatih sebelumnya menggunakan algoritma Random Forest. Model kemudian memprediksi jenis gerakan tangan, seperti “buka” atau “tutup”, dan hasil prediksi ditampilkan langsung pada layar. Dengan proses ini, sistem mampu mengenali gerakan tangan secara cepat dan akurat

A. Tujuan dan Manfaat

1. Memahami konsep dasar Computer Vision.
2. Merancang sistem pengenalan gerakan tangan berbasis webcam.
3. Mengimplementasikan deteksi landmark tangan menggunakan MediaPipe.
4. Mengolah data koordinat tangan menjadi fitur numerik untuk proses klasifikasi.
5. Menerapkan algoritma Random Forest untuk mengklasifikasikan gerakan tangan buka dan tutup.

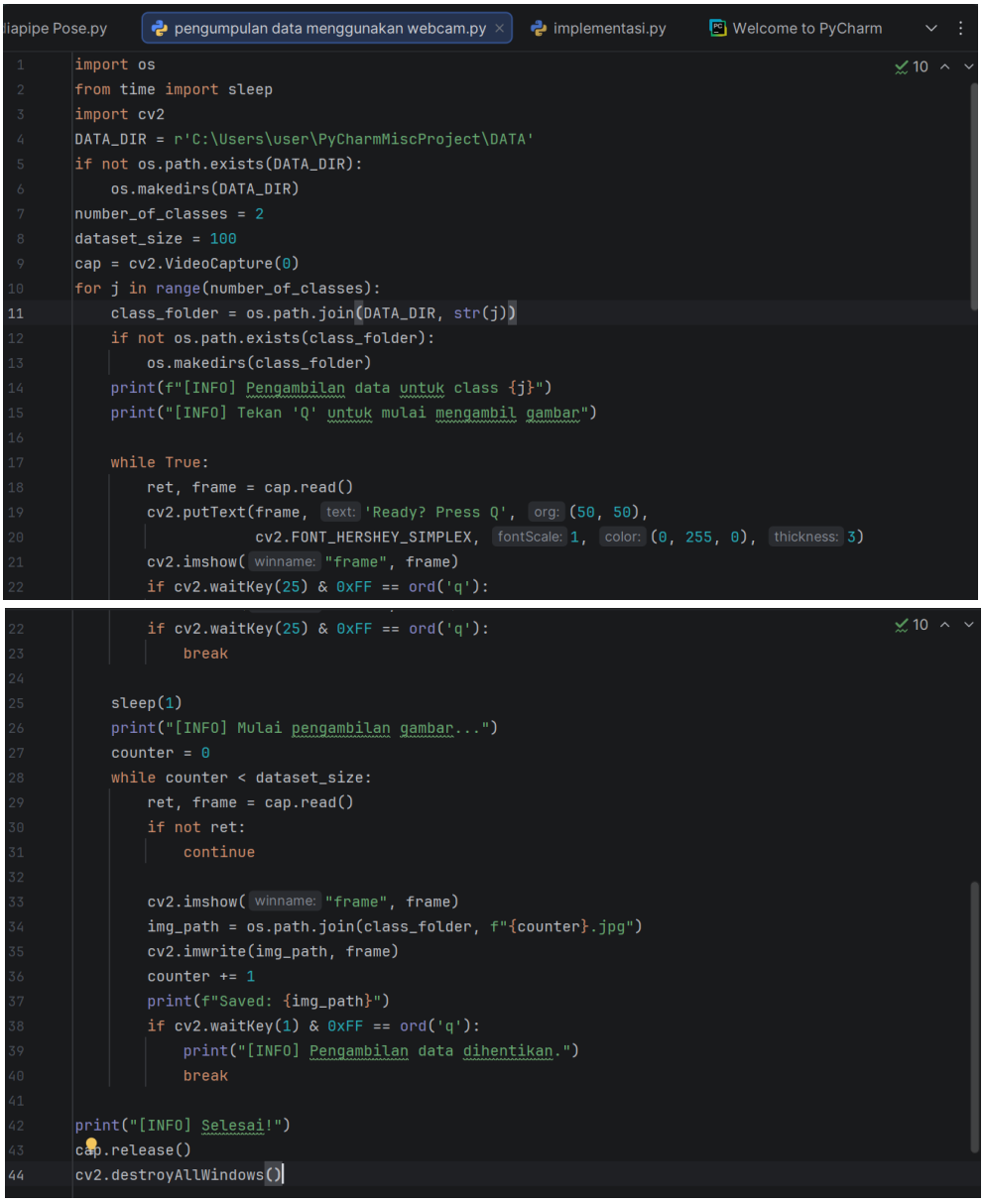
Manfaat

1. Mahasiswa dapat memahami bagaimana sistem mendeteksi objek berdasarkan citra digital.
2. Memberikan pemahaman tentang penerapan computer vision dalam pengenalan gerakan tangan.

3. Menambah wawasan mengenai penggunaan machine learning untuk klasifikasi data.
4. Melatih kemampuan dalam mengolah data dan membangun model prediksi.

B. Kode Program dan Analisis

Kode Program Pengumpulan Data Menggunakan Webcam



```
import os
from time import sleep
import cv2
DATA_DIR = r'C:\Users\user\PyCharmMiscProject\DATA'
if not os.path.exists(DATA_DIR):
    os.makedirs(DATA_DIR)
number_of_classes = 2
dataset_size = 100
cap = cv2.VideoCapture(0)
for j in range(number_of_classes):
    class_folder = os.path.join(DATA_DIR, str(j))
    if not os.path.exists(class_folder):
        os.makedirs(class_folder)
    print(f"[INFO] Pengambilan data untuk class {j}")
    print("[INFO] Tekan 'Q' untuk mulai mengambil gambar")

    while True:
        ret, frame = cap.read()
        cv2.putText(frame, text: 'Ready? Press Q', org: (50, 50),
                    cv2.FONT_HERSHEY_SIMPLEX, fontScale: 1, color: (0, 255, 0), thickness: 3)
        cv2.imshow( winname: "frame", frame)
        if cv2.waitKey(25) & 0xFF == ord('q'):
            break

    sleep(1)
    print("[INFO] Mulai pengambilan gambar...")
    counter = 0
    while counter < dataset_size:
        ret, frame = cap.read()
        if not ret:
            continue

        cv2.imshow( winname: "frame", frame)
        img_path = os.path.join(class_folder, f"{counter}.jpg")
        cv2.imwrite(img_path, frame)
        counter += 1
        print(f"Saved: {img_path}")
        if cv2.waitKey(1) & 0xFF == ord('q'):
            print("[INFO] Pengambilan data dihentikan.")
            break

print("[INFO] Selesai!")
cap.release()
cv2.destroyAllWindows()
```

Analisis Kode Program

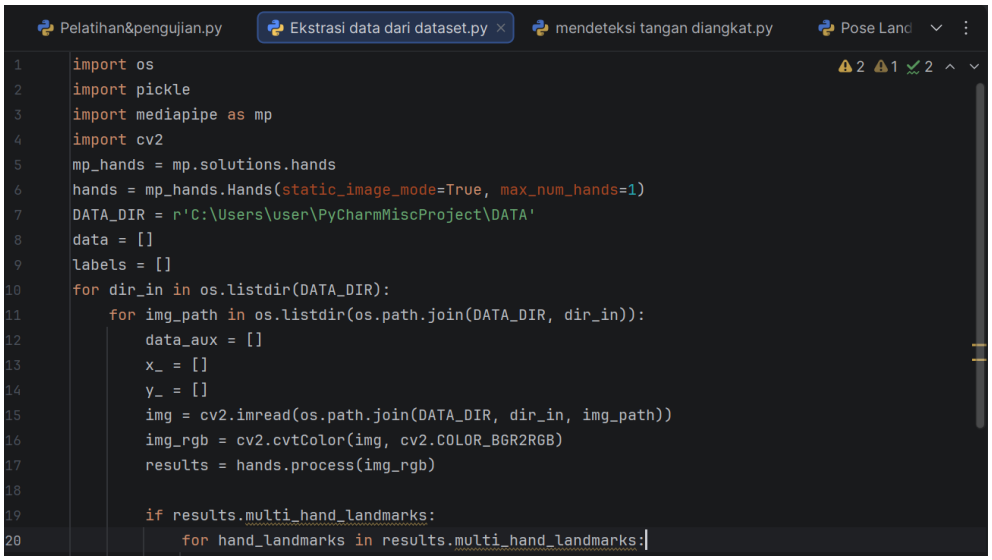
Program ini digunakan untuk melakukan proses pengambilan dataset gambar secara otomatis menggunakan webcam. Dataset tersebut nantinya digunakan sebagai data latih dalam proses machine learning. Pada awal program, sistem menentukan

lokasi folder penyimpanan data. Jika folder tersebut belum tersedia, maka program akan membuatnya secara otomatis agar gambar yang diambil dapat tersimpan dengan rapi.

Selanjutnya, program akan menentukan jumlah kelas dan jumlah gambar yang akan diambil. Dalam kode tersebut ditentukan terdapat dua kelas, dan masing-masing kelas akan memiliki 100 gambar. Artinya, total gambar yang akan dikumpulkan adalah 200 gambar. Setelah itu, kamera diaktifkan menggunakan OpenCV untuk mulai menangkap gambar secara real-time. Setelah kamera diaktifkan, program akan melakukan perulangan sesuai jumlah kelas yang telah ditentukan. Untuk setiap kelas, sistem membuat folder khusus agar data tersimpan secara terstruktur dan tidak tercampur. Sebelum pengambilan gambar dimulai, user diminta menekan tombol "Q" sebagai tanda siap.

Selanjutnya, program akan menangkap gambar secara berurutan dan menyimpannya dalam format JPG ke folder kelas yang sesuai hingga jumlah yang ditentukan terpenuhi atau dihentikan secara manual. Setelah seluruh proses selesai, kamera akan dimatikan dan semua jendela tampilan ditutup. Secara keseluruhan, program ini berfungsi sebagai tahap pengumpulan data sebelum masuk ke proses ekstraksi fitur dan pelatihan model machine learning.

Kode Program Ekstrasi Data dari Dataset



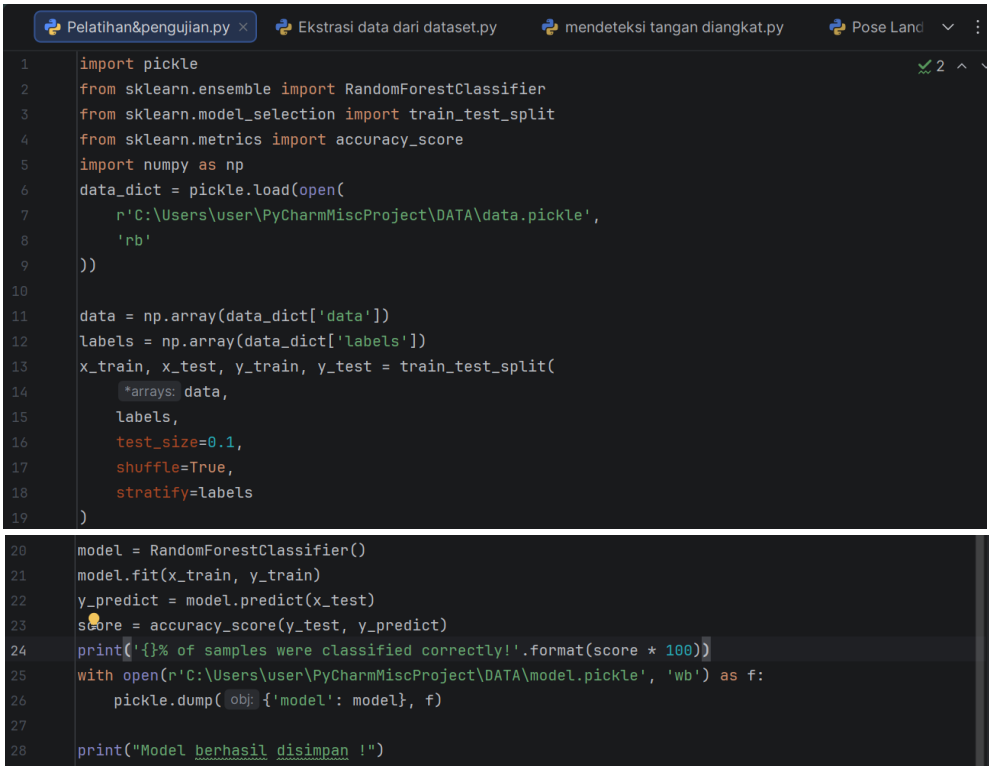
```
        if results.multi_hand_landmarks:
            for hand_landmarks in results.multi_hand_landmarks:

                for landmark in hand_landmarks.landmark:
                    x_.append(landmark.x)
                    y_.append(landmark.y)
                for landmark in hand_landmarks.landmark:
                    data_aux.append(landmark.x - min(x_))
                    data_aux.append(landmark.y - min(y_))

            data.append(data_aux)
            labels.append(dir_in)

with open(r'C:\Users\user\PyCharmMiscProject\DATA\data.pickle', 'wb') as f:
    pickle.dump( obj: {'data': data, 'labels': labels}, f)
print("Data berhasil disimpan")
```

Analisis Program

Program tersebut berfungsi untuk melakukan proses ekstraksi fitur dari dataset gambar yang telah dikumpulkan sebelumnya. Diawal, program mengimpor library yang dibutuhkan seperti *os* untuk membaca folder, *pickle* untuk menyimpan data hasil ekstraksi, *cv2* untuk mengolah gambar, serta mediapipe untuk mendeteksi landmark tangan. Sistem kemudian mengaktifkan modul Hands dari MediaPipe dalam mode gambar statis dan membatasi deteksi maksimal satu tangan pada setiap gambar. Setelah itu, program menentukan direktori dataset dan menyiapkan dua variabel, yaitu *data* untuk menyimpan fitur dan *labels* untuk menyimpan label kelas.

Selanjutnya, program membaca setiap folder kelas yang ada di dalam direktori dataset. Untuk setiap gambar di dalam folder tersebut, gambar akan dibaca menggunakan OpenCV lalu dikonversi dari format BGR ke RGB karena MediaPipe bekerja dalam format RGB. Kemudian gambar diproses menggunakan MediaPipe untuk mendeteksi landmark tangan. Kode program ini berfungsi untuk mengubah gambar mentah menjadi data numerik berupa koordinat landmark tangan yang nantinya dapat digunakan sebagai input dalam proses pelatihan model machine learning.

Kode Program untuk Pelatihan dan Pengujian



Analisis Program

Kode ini adalah program untuk melatih model machine learning yang bisa mengenali gestur tangan. Program dimulai dengan memuat dataset dari file data.pickle yang sebelumnya sudah dibuat, lalu mengubahnya menjadi array numpy agar bisa diproses. Data kemudian dibagi menjadi dua bagian: 90% untuk melatih model dan 10% untuk menguji seberapa akurat model tersebut, dengan opsi stratify=labels untuk memastikan proporsi setiap kelas tetap seimbang di kedua bagian.

Setelah data siap, program membuat model Random *Forest Classifier* yaitu algoritma yang bekerja dengan membangun banyak pohon keputusan lalu menggabungkan hasilnya kemudian melatihnya menggunakan data latih. Model yang sudah dilatih lalu diuji dengan data uji, dan hasilnya dihitung menggunakan accuracy_score untuk melihat berapa persen prediksi yang benar.

Terakhir, program mencetak persentase akurasi ke layar dan menyimpan model yang sudah dilatih ke file model.pickle. File ini nantinya bisa digunakan kembali di program lain (seperti program deteksi tangan secara real-time) tanpa perlu melatih ulang dari awal.

Kode Program Implementasi

```python
import pickle
import cv2
import mediapipe as mp
import numpy as np
model_dict = pickle.load(open(
    r"C:\Users\user\PyCharmMiscProject\DATA\model.pickle", 'rb'
))
model = model_dict['model']
cap = cv2.VideoCapture(0)
mp_hands = mp.solutions.hands
mp_drawing = mp.solutions.drawing_utils
hands = mp_hands.Hands(max_num_hands=1)
labels_dict = {0: 'buka', 1: 'tutup'}

while True:
    data_aux = []
    x_ = []
    y_ = []
    ret, frame = cap.read()
    if not ret:
        break

    frame_rgb = cv2.cvtColor(frame, cv2.COLOR_BGR2RGB)
    results = hands.process(frame_rgb)
    if results.multi_hand_landmarks:
        for hand_landmarks in results.multi_hand_landmarks:

            mp_drawing.draw_landmarks(
                frame,
                hand_landmarks,
                mp_hands.HAND_CONNECTIONS
            )
            for landmark in hand_landmarks.landmark:
                x_.append(landmark.x)
                y_.append(landmark.y)
            for landmark in hand_landmarks.landmark:
                data_aux.append(landmark.x - min(x_))
                data_aux.append(landmark.y - min(y_))

        prediction = model.predict([np.asarray(data_aux)])
        predicted_character = labels_dict[int(prediction[0])]
        cv2.putText(
            frame,
            predicted_character,
            org: (50, 100),
            cv2.FONT_HERSHEY_SIMPLEX,
            cv2.FONT_HERSHEY_SIMPLEX,
            fontScale: 1.3,
            color: (0, 0, 0),
            thickness: 3
        )
    cv2.imshow( winname: 'frame', frame)
    if cv2.waitKey(1) & 0xFF == ord('q'):
        break
cap.release()
cv2.destroyAllWindows()
```

Analisis Program

Kode ini adalah program implementasi real-time yang menggunakan model yang sudah dilatih sebelumnya untuk mendeteksi gestur tangan melalui webcam. Di awal, program memuat model dari file model.pickle, lalu membuka kamera

menggunakan OpenCV dan menyiapkan MediaPipe untuk mendeteksi landmark tangan. Tersedia juga kamus label {0: 'buka', 1: 'tutup'} yang berfungsi menerjemahkan hasil prediksi angka menjadi kata yang mudah dibaca.

Program kemudian berjalan dalam loop terus-menerus, membaca setiap frame dari webcam lalu mengonversinya ke format RGB agar bisa diproses oleh MediaPipe. Jika tangan terdeteksi, program akan menggambar titik-titik landmark beserta koneksinya di atas frame, lalu mengambil koordinat setiap titik dan menormalkannya dengan mengurangi nilai minimum agar posisi tangan tidak terpengaruh oleh lokasinya di layar.

Data koordinat yang sudah dinormalisasi tersebut kemudian dimasukkan ke model untuk diprediksi, dan hasilnya ditampilkan sebagai teks di layar (misalnya "buka" atau "tutup"). Frame hasil deteksi ditampilkan secara langsung di jendela bernama "frame", dan program akan berhenti serta menutup semua jendela apabila pengguna menekan tombol 'q' pada keyboard.

Latihan 3 Pose Berdiri dan Duduk

Kode program pengumpulan data pose

ngumpulan data menggunakan webcam.py | implementasi.py | data pose.py | ekstraksi pose.py | pe

```python
import os
from time import sleep
import cv2
import mediapipe as mp
DATA_DIR = r'C:\Users\user\PyCharmMiscProject\DATA'
if not os.path.exists(DATA_DIR):
    os.makedirs(DATA_DIR)

number_of_classes = 2
dataset_size = 100
mp_pose = mp.solutions.pose
mp_drawing = mp.solutions.drawing_utils

pose = mp_pose.Pose()
cap = cv2.VideoCapture(0)
for j in range(number_of_classes):
    class_folder = os.path.join(DATA_DIR, str(j))
    if not os.path.exists(class_folder):
        os.makedirs(class_folder)

    print(f"[INFO] Pengambilan data untuk class {j}")
    print("[INFO] Tekan 'Q' untuk mulai mengambil gambar")
    while True:
```

```
    while True:
        ret, frame = cap.read()
        cv2.putText(frame, text: 'Ready? Press Q', org: (50, 50),
                    cv2.FONT_HERSHEY_SIMPLEX, fontScale: 1, color: (0, 255, 0), thickness: 3)
        cv2.imshow( winname: "Pose Capture", frame)
        if cv2.waitKey(25) & 0xFF == ord('q'):
            break
    sleep(1)
    print("[INFO] Mulai pengambilan gambar...")
    counter = 0
    while counter < dataset_size:
        ret, frame = cap.read()
        if not ret:
            continue
        img_rgb = cv2.cvtColor(frame, cv2.COLOR_BGR2RGB)
        results = pose.process(img_rgb)
        if results.pose_landmarks:
            mp_drawing.draw_landmarks(
                frame,
                results.pose_landmarks,
                mp_pose.POSE_CONNECTIONS
            )
```

```
            img_path = os.path.join(class_folder, f"{counter}.jpg")
            cv2.imwrite(img_path, frame)
            print(f"Saved: {img_path}")
            counter += 1
        cv2.imshow( winname: "Pose Capture", frame)
        if cv2.waitKey(1) & 0xFF == ord('q'):
            print("[INFO] Pengambilan data dihentikan.")
            break

print("[INFO] Selesai!")
cap.release()
cv2.destroyAllWindows()
```

Kode program ekstraksi data pose

ngumpulan data menggunakan webcam.py | implementasi.py | data pose.py | ekstraksi pose.py | pe

```
import os
import pickle
import mediapipe as mp
import cv2
mp_pose = mp.solutions.pose
pose = mp_pose.Pose(static_image_mode=True)
DATA_DIR = r'C:\Users\PyCharmMiscProject\data_pose'
data = []
labels = []

for dir_ in os.listdir(DATA_DIR):
    class_dir = os.path.join(DATA_DIR, dir_)
    for img_path in os.listdir(class_dir):
        img = cv2.imread(os.path.join(class_dir, img_path))
        if img is None:
            continue
        img_rgb = cv2.cvtColor(img, cv2.COLOR_BGR2RGB)
        results = pose.process(img_rgb)
        data_aux = []
        x_ = []
        y_ = []
        if results.pose_landmarks:
            for lm in results.pose_landmarks.landmark:
```

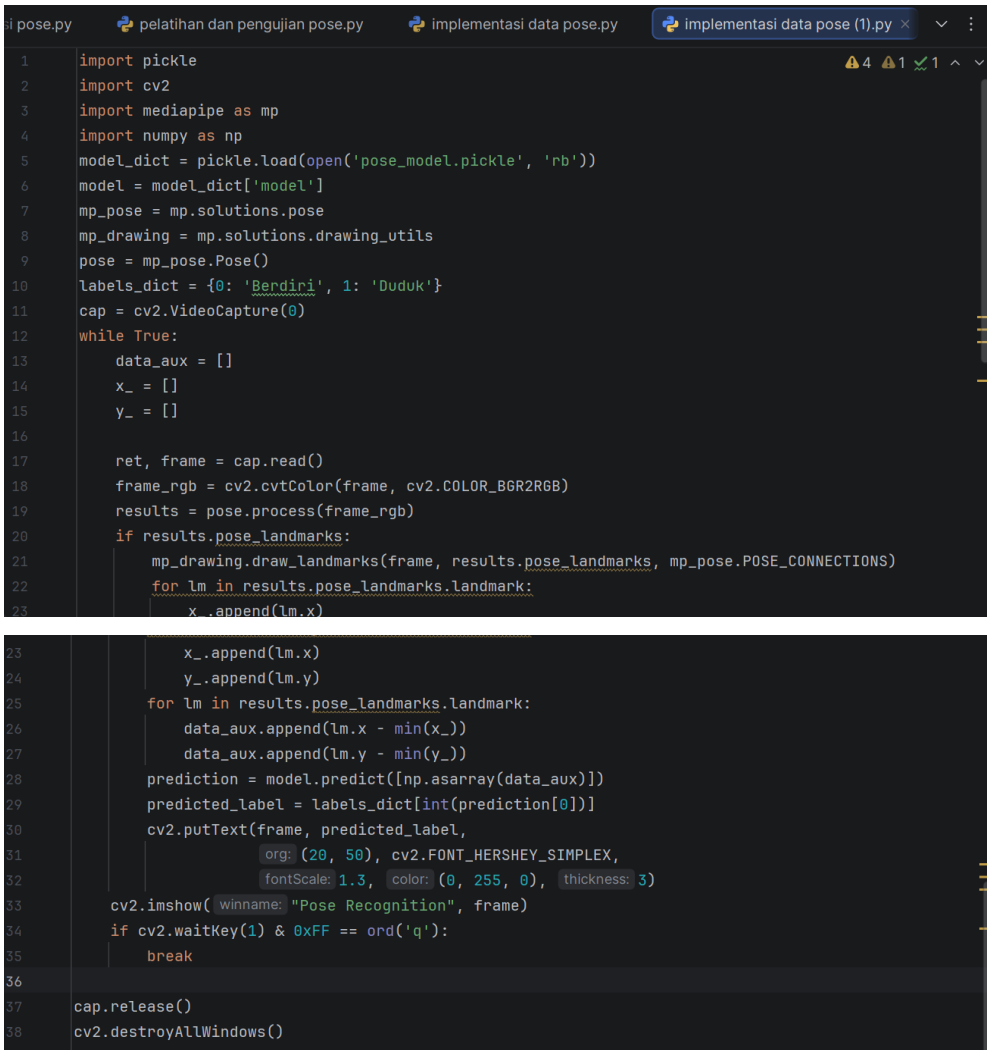
```
        if results.pose_landmarks:
            for lm in results.pose_landmarks.landmark:
                x_.append(lm.x)
                y_.append(lm.y)
            for lm in results.pose_landmarks.landmark:
                data_aux.append(lm.x - min(x_))
                data_aux.append(lm.y - min(y_))

            data.append(data_aux)
            labels.append(int(dir_))
with open('pose_data.pickle', 'wb') as f:
    pickle.dump( obj: {'data': data, 'labels': labels}, f)
print("Ekstraksi selesai. File tersimpan sebagai pose_data.pickle")
```

Kode program pelatihan dan pengujian data

```
entasi.py   data pose.py   ekstraksi pose.py   pelatihan dan pengujian pose.py   Welcome to PyChar
import pickle
from sklearn.ensemble import RandomForestClassifier
from sklearn.model_selection import train_test_split
from sklearn.metrics import accuracy_score, classification_report
import numpy as np
data_dict = pickle.load(open(
    r'C:\Users\user\PyCharmMiscProject\DATA\pose_data.pickle',
    'rb'
))
data = np.array(data_dict['data'])
labels = np.array(data_dict['labels'])
print(f"Jumlah data: {len(data)}")
print(f"Distribusi label: Berdiri={sum(labels==0)}, Duduk={sum(labels==1)}")
x_train, x_test, y_train, y_test = train_test_split(
    *arrays: data,
    labels,
    test_size=0.1,
    shuffle=True,
    stratify=labels
)
model = RandomForestClassifier(n_estimators=100, random_state=42)
model.fit(x_train, y_train)
y_predict = model.predict(x_test)
```

```
model = RandomForestClassifier(n_estimators=100, random_state=42)
model.fit(x_train, y_train)
y_predict = model.predict(x_test)
score = accuracy_score(y_test, y_predict)

print(f"\ndari sampel diklasifikasikan dengan benar!".format(score * 100))
print("\nLaporan Klasifikasi:")
print(classification_report(y_test, y_predict, target_names=['berdiri', 'duduk']))

with open(r'C:\Users\user\PyCharmMiscProject\DATA\pose_model.pickle', 'wb') as f:
    pickle.dump( obj: {'model': model}, f)

print("Model berhasil disimpan!")
```

Kode program Implementasi data pose



```
import pickle
import cv2
import mediapipe as mp
import numpy as np
model_dict = pickle.load(open('pose_model.pickle', 'rb'))
model = model_dict['model']
mp_pose = mp.solutions.pose
mp_drawing = mp.solutions.drawing_utils
pose = mp_pose.Pose()
labels_dict = {0: 'Berdiri', 1: 'Duduk'}
cap = cv2.VideoCapture(0)
while True:
    data_aux = []
    x_ = []
    y_ = []

    ret, frame = cap.read()
    frame_rgb = cv2.cvtColor(frame, cv2.COLOR_BGR2RGB)
    results = pose.process(frame_rgb)
    if results.pose_landmarks:
        mp_drawing.draw_landmarks(frame, results.pose_landmarks, mp_pose.POSE_CONNECTIONS)
        for lm in results.pose_landmarks.landmark:
            x_.append(lm.x)
            y_.append(lm.y)
        for lm in results.pose_landmarks.landmark:
            data_aux.append(lm.x - min(x_))
            data_aux.append(lm.y - min(y_))
        prediction = model.predict([np.asarray(data_aux)])
        predicted_label = labels_dict[int(prediction[0])]
        cv2.putText(frame, predicted_label,
                    org: (20, 50), cv2.FONT_HERSHEY_SIMPLEX,
                    fontScale: 1.3,  color: (0, 255, 0),  thickness: 3)
    cv2.imshow( winname: "Pose Recognition", frame)
    if cv2.waitKey(1) & 0xFF == ord('q'):
        break

cap.release()
cv2.destroyAllWindows()
```

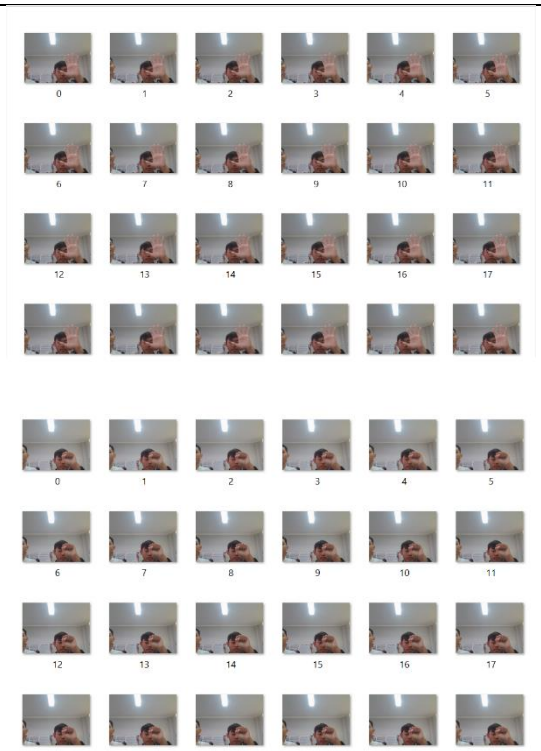
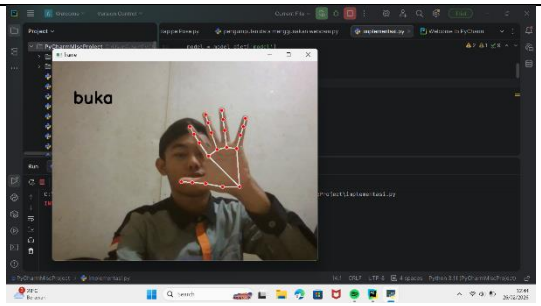
Analisis Program

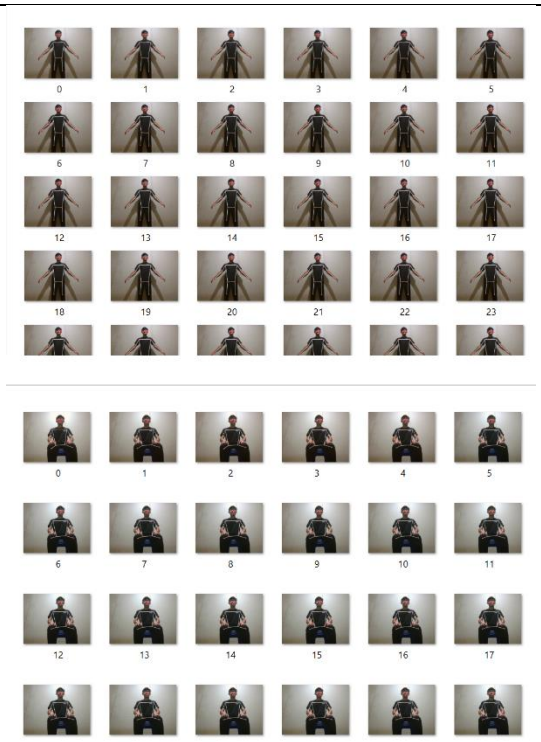
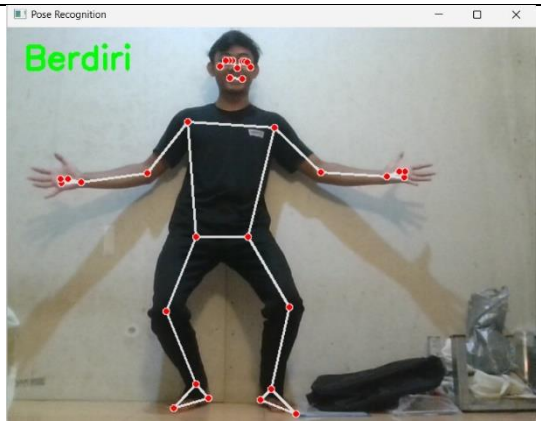
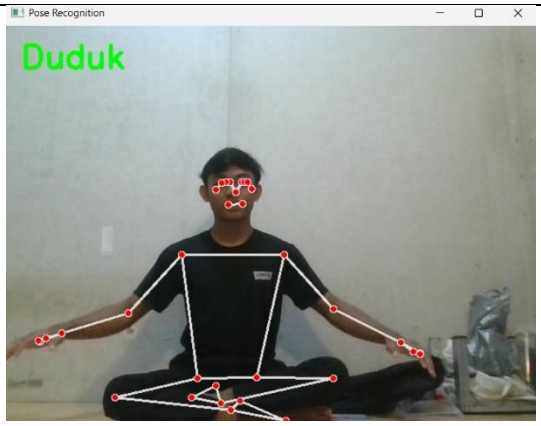
Program ini terdiri dari empat file yang bekerja secara berurutan untuk membangun sistem deteksi postur tubuh. File pertama bertugas mengumpulkan data, file kedua bertugas mengekstrak data dari foto-foto yang sudah disiapkan dalam dua folder yaitu "berdiri" dan "duduk". Setiap gambar diproses menggunakan MediaPipe Pose untuk mendeteksi 33 titik landmark tubuh manusia, kemudian koordinat setiap titik dinormalisasi agar posisi orang di foto tidak mempengaruhi hasil, lalu semua data disimpan ke file pose_data.pickle.

File ketiga mengambil data yang sudah disimpan tadi untuk melatih model machine learning menggunakan algoritma Random Forest. Data dibagi menjadi 90% untuk pelatihan dan 10% untuk pengujian, kemudian model dilatih dan dievaluasi

akurasinya. Setelah proses pelatihan selesai, model disimpan ke file pose_model.pickle sehingga bisa digunakan kembali tanpa perlu melatih ulang dari awal.

File keempat adalah program utama yang berjalan secara real-time menggunakan webcam. Program membuka kamera, mendeteksi pose tubuh di setiap frame, lalu mengirimkan data landmark ke model untuk diprediksi. Hasilnya langsung ditampilkan di layar dalam bentuk teks "Berdiri" berwarna hijau atau "Duduk" berwarna oranye, lengkap dengan gambaran skeleton tubuh. Jika tidak ada orang yang terdeteksi, program akan menampilkan pesan peringatan, dan pengguna bisa menghentikan program kapan saja dengan menekan tombol 'q'.

Pengumpulan data melalui webcam	
Mendeteksi tangan tertutup	
Mendeteksi tangan terbuka	

Pengumpulan data pose	
Mendeteksi pose berdiri	
Mendeteksi pose duduk	

Kesimpulan

Program ini adalah sistem pengenalan postur tubuh manusia secara real-time yang menggabungkan MediaPipe Pose, OpenCV, dan algoritma Random Forest. Sistem bekerja dalam tiga tahap yaitu pengumpulan data, pelatihan model, dan implementasi melalui webcam, dengan memanfaatkan 33 titik landmark tubuh untuk membedakan postur berdiri dan duduk. Program ini memiliki potensi besar untuk dikembangkan lebih lanjut, seperti menambah kelas postur baru, memperbanyak dataset, atau diterapkan dalam sistem monitoring kesehatan dan keselamatan kerja. Dengan struktur yang rapi dan terorganisir, program ini menjadi fondasi yang baik untuk pengembangan sistem pengenalan aktivitas manusia yang lebih kompleks ke depannya.

E. Referensi

- https://youtu.be/mSO2hJln0OY?si=zsPFHrv1iJ-ai19V
- https://youtu.be/mEwoAV5_dcA?si=5l-UqbsFZIyjRyBR