Program Studi Teknik Informatika
Institut Teknologi Sumatera

Nama: **Felix Ferdinandus, Kiagus M Roihan Ananta, Pricelia Putri** Tugas : **Final Project**
Mata Kuliah: **Multi Media (IF4021)** Tanggal: 30 Mei 2025

1 Deskripsi Project

Tebak Anomali merupakan filter interaktif berbasis Python yang memadukan audio, visual, dan pengenalan gesture tangan. Aplikasi ini menampilkan lima gambar anomali yang diacak posisinya, lalu memutar suara dari salah satu anomali tersebut. Pengguna diminta mencocokkan suara tersebut dengan gambar yang benar, dengan cara membentuk angka menggunakan jari tangan (1 hingga 5) yang mewakili pilihan gambar. Gesture angka ini akan dikenali oleh sistem menggunakan hand recognition berbasis MediaPipe, dan digunakan sebagai input untuk menjawab soal. Jika jawabannya benar, pengguna akan melanjutkan ke pertanyaan berikutnya dan mendapat poin. Jika salah, sistem tidak akan memberikan poin kepada pengguna dan akan melanjutkan ke pertanyaan berikutnya hingga pengguna menjawab semua pertanyaan.

2 Teknologi

Teknologi yang digunakan dalam mengerjakan *Final Project* ini, yaitu sebagai berikut:

- Bahasa Pemrograman Python sebagai dasar bahasa program yang digunakan.
- Mediapipe: Digunakan untuk deteksi dan pengenalan tangan serta untuk menggambar landmark tangan. Secara spesifik, ini digunakan untuk mengenali gestur angka jari (1 sampai 5).
- OpenCV (cv2): Digunakan untuk pemrosesan citra dan video, termasuk membaca dari kamera, membalikkan frame, menggambar bentuk dan teks pada frame, mengubah ukuran gambar, dan menampilkan frame.
- Pygame: Digunakan untuk memutar efek suara dan musik dalam game.
- NumPy: Digunakan untuk operasi numerik, terutama untuk memanipulasi array gambar.

3 Cara Kerja

Cara kerja pada proyek *Tebak Anomali* adalah sebagai berikut:

- Sistem menggunakan kamera *webcam* untuk menangkap video secara langsung.
- Sistem memanfaatkan *library* seperti MediaPipe untuk mendeteksi tangan dan mengenali gesture angka (1 hingga 5) berdasarkan posisi *landmark* pada jari-jari tangan.
- Aplikasi akan menampilkan lima gambar anomali secara acak dan memutar suara dari salah satu anomali tersebut.

- Pengguna diminta mencocokkan suara dengan gambar yang sesuai dengan membentuk gesture angka menggunakan tangan, yang kemudian dikenali sistem sebagai input jawaban.
- Jika gesture yang dikenali sesuai dengan jawaban yang benar, maka sistem akan menampilkan soal berikutnya. Jika salah, sistem memberikan umpan balik kepada pengguna dan tetap berada pada soal yang sama.

4 Penjelasan Kode Program

Kode program pada proyek *Tebak Anomali* ini ada pada **main.py**. Berikut ini adalah penjelasan dari **main.py**.

4.1 Import dan Inisialiasi

```
import cv2
import mediapipe as mp
import numpy as np
import time
import pygame as pg
import random
```

Import library utama:

- cv2: OpenCV, untuk menangani video dan gambar.
- mediapipe: untuk deteksi dan pelacakan tangan.
- numpy: operasi array.
- time: untuk timer dan penghitungan waktu.
- pygame: untuk memutar suara.
- random: untuk memilih soal secara acak.

```
mp_hands = mp.solutions.hands
mp_drawing = mp.solutions.drawing_utils

cap = cv2.VideoCapture(1)  # Membuka kamera (index 1, bisa disesuaikan)
```

- mp_hands: modul untuk pelacakan tangan.
- mp_drawing: utilitas untuk menggambar landmark tangan di frame.
- cap: membuka kamera, 1 berarti kamera kedua (biasanya 0 adalah kamera utama).

4.2 Fungsi draw$_s tart_s creen(frame)$

```
def draw_start_screen(frame):
    h, w, _ = frame.shape
    overlay = frame.copy()
    cv2.rectangle(overlay, (0, 0), (w, h), (0, 0, 0), -1)
    alpha = 0.7
    cv2.addWeighted(overlay, alpha, frame, 1 - alpha, 0, frame)
    cv2.putText(frame, 'TEBAK ANOMALI', (w//2-220, h//2-60), cv2.FONT_HERSHEY_SIMPLEX, 2,
    (255,255,255), 5)
    cv2.putText(frame, 'Tekan [Spasi] untuk mulai', (w//2-270, h//2+20), cv2.FONT_HERSHEY_SIMPLEX,
    1.2, (0,255,255), 3)
    return frame
```

- Membuat layar awal dengan overlay hitam transparan.
- Menampilkan judul game dan instruksi “Tekan Spasi untuk mulai”.
- Mengembalikan frame dengan tampilan layar awal.

4.3 Fungsi $\mathbf{draw}_{q}uestion_{s}creen(frame, choices, timer = None)$

```
def draw_question_screen(frame, choices, timer=None):
        h, w, _ = frame.shape
    # Area untuk gambar (5 gambar, horizontal), lebih kecil dan di bagian atas
    img_w, img_h = 110, 110  # Ukuran gambar diperkecil
    margin = 25
    total_width = img_w*5 + margin*4
    if total_width > w:
        scale = w / (img_w*5 + margin*4 + 1)
        img_w = int(img_w * scale)
        img_h = int(img_h * scale)
        margin = int(margin * scale)
        total_width = img_w*5 + margin*4
    start_x = max(0, (w - total_width) // 2)
    y = 10  # Gambar lebih ke atas
    for i, cid in enumerate(choices):
        img_path = f"aset/img/{cid}.png"
        img = cv2.imread(img_path, cv2.IMREAD_UNCHANGED)
        if img is not None:
            img = cv2.resize(img, (img_w, img_h))
            x = start_x + i*(img_w+margin)
            if x+img_w > w:
                continue
            if img.shape[2] == 4:
                alpha_s = img[:, :, 3] / 255.0
                alpha_l = 1.0 - alpha_s
                for c in range(3):
                    frame[y:y+img_h, x:x+img_w, c] = (alpha_s * img[:, :, c] + alpha_l * frame[y:y+
    img_h, x:x+img_w, c])
            else:
                frame[y:y+img_h, x:x+img_w] = img
            # Nomor urut lebih kecil
            cv2.rectangle(frame, (x, y+img_h+2), (x+img_w, y+img_h+25), (0,0,0), -1)
            cv2.putText(frame, str(i+1), (x+img_w//2-10, y+img_h+20), cv2.FONT_HERSHEY_SIMPLEX,
    0.9, (0,255,255), 2)
    # Timer di kanan bawah
    if timer is not None:
        timer_text = f"Timer: {timer}"
        text_size = cv2.getTextSize(timer_text, cv2.FONT_HERSHEY_SIMPLEX, 1.5, 4)[0]
        text_x = w - text_size[0] - 30
        text_y = h - 30
        cv2.putText(frame, timer_text, (text_x, text_y), cv2.FONT_HERSHEY_SIMPLEX, 1.2, (0,0,255),
    3)
    return frame
```

- Menampilkan 5 gambar pilihan (jawaban) di bagian atas layar.
- Gambar diambil dari folder aset/img/id.png.
- Ukuran gambar diatur agar pas di layar.
- Jika gambar memiliki alpha channel (transparansi), gambar digabung dengan frame secara halus.
- Nomor urut ditampilkan di bawah setiap gambar.

- Jika ada timer, tampilkan countdown timer di kanan bawah.
- Mengembalikan frame yang sudah diisi pilihan dan timer.

4.4 Inisialisasi pygame mixer

```
pg.mixer.init()
```

Menginisialisasi modul suara pygame untuk memutar efek suara.

4.5 State dan Variabel Game

```
state = 'start'  # State awal: tampilan start
current_question = 0
score = 0
max_questions = 5
question_data = []
user_answer = None
countdown_start = None
```

Variabel kontrol state permainan dan data soal, jawaban, skor, dll.

4.6 Fungsi generate_question()

```
def generate_questions():
    questions = []
    used = set()
    for _ in range(max_questions):
        answer = random.choice([i for i in all_ids if i not in used])
        used.add(answer)
        choices = [answer]
        while len(choices) < 5:
            c = random.choice(all_ids)
            if c not in choices:
                choices.append(c)
        random.shuffle(choices)
        questions.append({'answer': answer, 'choices': choices})
    return questions
```

- Membuat daftar soal berisi 5 soal.
- Setiap soal punya 1 jawaban benar (answer) dan 4 pilihan lain yang acak tapi unik.
- Pilihan dikocok urutannya.
- Mengembalikan list soal.

4.7 Main Loop dengan Mediapipe Hands

```
with mp_hands.Hands(
    max_num_hands=1,
    min_detection_confidence=0.7,
    min_tracking_confidence=0.7
) as hands:
```

- Membuka sesi deteksi tangan dengan parameter untuk kualitas deteksi dan tracking.
- max_num_hands=1: hanya satu tangan yang dideteksi.

4.8 Cek Kamera dan Setup Awal Game

```
if not cap.isOpened():
    print("Error: Could not open video capture device.")
    exit(1)

question_data = generate_questions()
state = 'start'
current_question = 0
score = 0
user_answer = None
countdown_start = None
last_gesture = 1
gesture_history = []
gesture_stable = 1
gesture_stable_count = 0
gesture_stable_required = 3
```

- Jika kamera tidak berhasil dibuka, keluar program.
- Menginisialisasi ulang state game dan variabel gesture untuk stabilisasi input gesture tangan.

4.9 Loop Utama Per Frame

```
while cap.isOpened():
    ret, frame = cap.read()
    if not ret or frame is None:
            print("Error: Could not read frame from camera.")
            break
        frame = cv2.flip(frame, 1)
        key = cv2.waitKey(1) & 0xFF
```

- Baca frame dari kamera.
- Flip frame supaya seperti cermin.
- Tangkap tombol keyboard yang ditekan.

4.10 State: start

```
if state == 'start':
    frame = draw_start_screen(frame)
    cv2.imshow('Hand Finger Count', frame)
    if key == ord(' '):
        state = 'question'
        current_question = 0
        score = 0
        question_data = generate_questions()
        user_answer = None
        countdown_start = None
        last_gesture = 1
        gesture_history = []
        gesture_stable = 1
        gesture_stable_count = 0
    continue
```

- Tampilkan layar awal.
- Jika tombol spasi ditekan, pindah ke state question (mulai permainan).

4.11 State: question

```
if state == 'question':
    q = question_data[current_question]
    if countdown_start is None:
        # Play sound soal dan suara countdown
    timer = 10 - int(time.time() - countdown_start)
    frame = draw_question_screen(frame, q['choices'], timer=timer)

    # Deteksi tangan dengan mediapipe
    results = hands.process(rgb)
    finger_tips = [4, 8, 12, 16, 20]
            gesture_count = None
            if results.multi_hand_landmarks and results.multi_handedness:
                hand_landmarks = results.multi_hand_landmarks[0]
                handedness = results.multi_handedness[0].classification[0].label
                landmarks = hand_landmarks.landmark
                fingers_up = []
                if handedness == 'Right':
                    fingers_up.append(1 if landmarks[finger_tips[0]].x < landmarks[finger_tips[0] -
     1].x else 0)
                else:
                    fingers_up.append(1 if landmarks[finger_tips[0]].x > landmarks[finger_tips[0] -
     1].x else 0)
                for tip in finger_tips[1:]:
                    fingers_up.append(1 if landmarks[tip].y < landmarks[tip - 2].y else 0)
                count = sum(fingers_up)
                if 1 <= count <= 5:
                    gesture_count = count
                    gesture_history.append(count)
                    if len(gesture_history) > gesture_stable_required:
                        gesture_history.pop(0)
                    # Cek stabilisasi
                    if len(gesture_history) == gesture_stable_required and all(g == gesture_history
    [0] for g in gesture_history):
                        gesture_stable = gesture_history[0]
                        gesture_stable_count = gesture_stable_required
                    else:
                        gesture_stable_count = 0
                # Visualisasi landmark tangan
                mp_drawing.draw_landmarks(frame, hand_landmarks, mp_hands.HAND_CONNECTIONS)
            # Tampilkan gesture yang terbaca di kiri bawah
            h, w, _ = frame.shape  # pastikan h dan w terdefinisi
            if gesture_count is not None:
                cv2.putText(frame, f'Gesture: {gesture_count}', (30, h-80), cv2.
    FONT_HERSHEY_SIMPLEX, 1.2, (255,255,255), 3)
            else:
                cv2.putText(frame, 'Gesture: -', (30, h-80), cv2.FONT_HERSHEY_SIMPLEX, 1.2,
    (255,255,255), 3)
            # Tampilkan gesture stabil di kiri bawah
            cv2.putText(frame, f'Stable: {gesture_stable if gesture_stable_count else "-"}', (30, h
    -30), cv2.FONT_HERSHEY_SIMPLEX, 1, (255,255,255), 3)
            # Countdown
            if timer <= 0:
                user_answer = gesture_stable if gesture_stable_count else last_gesture
                state = 'result'
                countdown_start = None
                pg.mixer.music.stop()
    # Jika waktu habis, simpan jawaban gesture stabil dan pindah state ke result
    cv2.imshow('Hand Finger Count', frame)
    continue
```

- Tampilkan soal dan pilihan jawaban dengan gambar.
- Putar suara soal dan suara countdown satu kali per soal.
- proses deteksi tangan:
  - Hitung jumlah jari terbuka (gesture).
  - Stabilkan gesture: harus muncul selama 3 frame berturut-turut agar valid.
- Tampilkan gesture dan gesture stabil di frame.
- Jika waktu timer habis, ambil gesture stabil sebagai jawaban user, pindah ke state result.

4.12 State: result

```
if state == 'result':
    q = question_data[current_question]
    idx = q['choices'].index(q['answer'])
    correct = (user_answer == idx+1)
    if correct:
        score += 20
        feedback = 'BENAR!'
        pg.mixer.music.load("aset/sound/correct.mp3")
        pg.mixer.music.play()
    else:
        feedback = 'SALAH!'
        pg.mixer.music.load("aset/sound/wrong.mp3")
        pg.mixer.music.play()
    frame = draw_question_screen(frame, q['choices'])
    # Tampilkan feedback, jawaban benar, dan skor
    cv2.imshow('Hand Finger Count', frame)
    cv2.waitKey(1200)  # jeda 1.2 detik sebelum soal berikutnya
    current_question += 1
    if current_question >= max_questions:
        state = 'end'
    else:
        state = 'question'
    continue
```

- Cek apakah jawaban gesture user sama dengan jawaban benar.
- Jika benar, tambahkan skor dan mainkan suara benar.
- Jika salah, mainkan suara salah.
- Tampilkan hasil di layar selama 1.2 detik.
- Jika soal habis, pindah ke state end, kalau belum lanjut soal berikutnya.

4.13 State: end

```
if state == 'end':
    # Buat overlay hitam transparan
    # Tampilkan teks skor akhir dan instruksi untuk main lagi atau keluar
    cv2.imshow('Hand Finger Count', frame)
    if key == ord(' '):
        # Restart game
    elif key == ord('q'):
        break
    continue
```

- Tampilkan layar akhir dengan skor total.
- Instruksi tekan spasi untuk main lagi atau Q untuk keluar.
- Jika spasi ditekan, restart game.
- Jika Q ditekan, keluar dari program.

4.14 Fallback

```
cv2.imshow('Hand Finger Count', frame)
if key == ord('q'):
    break
```

- Jika tidak ada kondisi di atas, hanya tampilkan frame.
- Jika Q ditekan kapan pun, keluar dari program.

4.15 Cleanup

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

Setelah loop selesai, tutup kamera dan hapus semua jendela OpenCV.

5 Implementasi Program

5.1 Instalasi

- Clone Repository.
  Buka Terminal di VisualCode, lalu ketik ini:

  ```
  git clone https://github.com/ForZa77123/Tubes_mulmed
  ```

- Install dependencies di Terminal.
  Di terminal ketik ini:

  ```
  pip install -r requirements.txt
  ```

- Jalankan program di Terminal

  ```
  python \texttt{main.py}
  ```

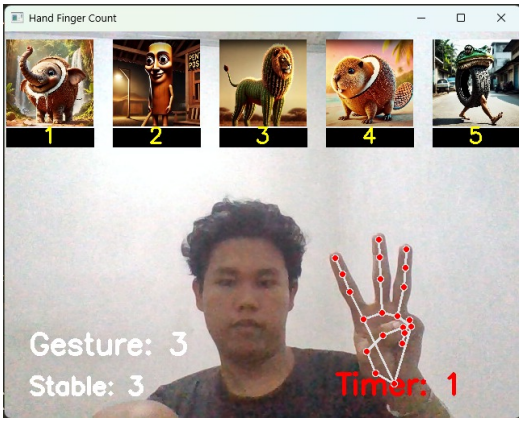
5.2 Antarmuka Program

Antarmuka program terdiri dari beberapa bagian utama, yaitu:

1. **Menu Start:** Menu utama dari aplikasi yang menampilkan Nama dari aplikasi ini dan arahan untuk menekan tombol "Spasi" untuk memulai permainan.

Gambar 1: Menu Start

2. **Gameplay:** Pada saat permainan dimulai, sistem akan meng*generate* 5 gambar berbeda dan 1 audio *random*. Setelah itu pengguna dapat memberikan *hand gestures* untuk membuat angka 1 sampai 5 dan menebak anomali dari ke 5 gambar tersebut berdasarkan audio yang diberikan. Jika benar, maka pengguna akan mendapatkan skor dan melanjutkan ke soal berikutnya. Jika salah, maka pengguna tidak akan mendapatkan poin dan melanjutkan ke soal berikutnya sampai 5 soal.



(a) Tampilan awal gameplay

(b) Pengguna menjawab Benar

(c) Pengguna menjawab Salah

Gambar 2: Gambar Jika pengguna menjawab benar atau salah.

3. **End screen** Pada saat permainan berakhir, sistem akan menampilkan score akhir dan opsi untuk memulai game kembali ataupun keluar.

Gambar 3: End screen

References

[1] Ivan Goncharov. (2022, Maret 14). *Custom Hand Gesture Recognition with Hand Landmarks Using Google's Mediapipe + OpenCV in Python.* [Video]. YouTube. Diakses pada 31 Mei 2025, dari https://www.youtube.com/results?search_query=mediapipe+hand+gesture+recognition+number

[2] Google AI. (2025, Januari 13). *Hand Landmarker.* Diakses pada 31 Mei 2025, dari https://ai.google.dev/edge/mediapipe/solutions/vision/hand_landmarker

[3] Gautam, A. (2020, September 21). *Hand Recognition using OpenCV.* Medium. Diakses pada 31 Mei 2025, dari https://gautamaditee.medium.com/hand-recognition-using-opencv-a7b109941c88