Program Studi Teknik Informatika
Institut Teknologi Sumatera

Nama:
1. **Lucky Immanuel S [122140179]**
2. **Rachel Olivia Manullang [122140181]**
2. **Eric Arwido Damanik [122140157]**

Tugas Ke: **Tugas Besar**

Mata Kuliah: **Sistem / Teknologi Multimedia (IF4021)** Tanggal: 09 Desember 2025

1 Deskripsi Proyek

Proyek ini dirancang untuk menunjukkan bagaimana teknologi pemrosesan visual dapat dikombinasikan dengan elemen permainan (game mechanics) untuk menghasilkan bentuk interaksi yang bersifat natural, intuitif, dan real-time. Dalam konteks multimedia, aplikasi ini tidak hanya mengolah gambar secara statis, melainkan memanfaatkan aliran data visual langsung dari kamera, kemudian memprosesnya menjadi informasi bermakna yang digunakan sebagai input permainan.

Permainan ini menggunakan teknologi visi komputer untuk mendeteksi wajah dan kedipan mata sebagai metode interaksi utama. Pemain harus mengedipkan mata pada waktu yang tepat untuk menghentikan bagian-bagian wajah yang jatuh dari sisi atas layar dan menempelkannya pada posisi landmark wajah mereka sendiri. Dengan demikian, FaceFit Blink Challenge mengintegrasikan tiga elemen multimedia inti secara bersamaan: citra real-time, interaksi berbasis ekspresi wajah, dan grafik permainan yang bergerak dinamis.

Pengembangan proyek ini juga menjadi wadah penerapan materi teoretis seperti pengolahan citra digital, deteksi fitur wajah, analisis visual berbasis landmark, serta integrasi multimodal input pada sistem interaktif. Selain itu, proyek ini menunjukkan bagaimana sinkronisasi antara deteksi kamera dan tampilan grafis dapat membentuk pengalaman multimedia yang responsif dan menarik.

2 Teknologi

Dalam membangun FaceFit Blink Challenge, digunakan beberapa teknologi inti yang mendukung pengolahan visual dan interaksi real-time. Python dipilih sebagai bahasa utama karena fleksibilitasnya serta ketersediaan pustaka yang kuat dalam pengolahan citra maupun pengembangan aplikasi multimedia.

OpenCV digunakan untuk menangani pengambilan frame dari kamera dan menyediakan berbagai fungsi dasar pengolahan citra. MediaPipe Face Mesh merupakan komponen utama dalam deteksi wajah, menghasilkan 468 landmark wajah berpresisi tinggi yang kemudian dianalisis untuk menentukan posisi fitur wajah serta mendeteksi kedipan melalui perhitungan rasio mata.

Pada sisi grafis, Pygame digunakan untuk membangun lingkungan permainan, merender objek bagian wajah yang jatuh, menampilkan antarmuka visual, dan mengelola game loop. Integrasi antara hasil deteksi wajah dan Pygame memungkinkan sistem menampilkan dinamika permainan secara sinkron dengan pergerakan dan ekspresi wajah pemain.

Teknologi-teknologi tersebut menunjukkan bagaimana konsep dalam Teknologi Multimedia seperti pemrosesan multimodal input, sinkronisasi media, dan real-time rendering dapat diterapkan dalam satu aplikasi yang koheren. Proyek ini menjadi contoh konkret penerapan teknologi multimedia modern berbasis visi komputer.

3 Cara Kerja

Bagian ini menjelaskan alur kerja FaceFit Blink Challenge secara sistematis dan runtut, menggambarkan bagaimana data kamera diproses dan bagaimana interaksi permainan terjadi berdasarkan kedipan mata pengguna.

- **Pengambilan Frame Kamera**
  Sistem dimulai dengan mengaktifkan kamera untuk mengambil frame secara berkelanjutan. Frame mentah ini menjadi sumber data utama bagi proses deteksi wajah dan interaksi permainan.
- **Konversi Frame dan Pemrosesan Awal**
  Setiap frame dikonversi dari format BGR ke RGB agar kompatibel dengan pipeline MediaPipe. Konversi ini memastikan proses deteksi wajah bekerja optimal.
- **Deteksi Landmark Menggunakan MediaPipe Face Mesh**
  Frame yang telah diproses masuk ke Face Mesh untuk menghasilkan ratusan landmark wajah. Landmark-landmark ini digunakan untuk menentukan posisi mata, hidung, alis, dan mulut.
- **Perhitungan Rasio Kedipan Mata**
  Sistem menghitung jarak antar kelopak mata dibandingkan dengan lebar mata. Jika rasio turun di bawah ambang tertentu, sistem mengenali kondisi tersebut sebagai kedipan.
- **Render Bagian Wajah yang Jatuh**
  Pada sisi permainan, Pygame merender bagian wajah yang jatuh secara terus-menerus. Posisi setiap objek diperbarui sesuai kecepatan jatuh yang ditentukan.
- **Pengecekan Overlap antara Objek dan Landmark**
  Saat bagian wajah jatuh mendekati area landmark yang sesuai, sistem mengecek apakah terjadi overlap. Overlap menandakan bahwa objek berada pada posisi siap ditempelkan.
- **Konfirmasi melalui Deteksi Kedipan**
  Sistem menunggu kedipan pengguna sebagai bentuk persetujuan. Jika kedipan terjadi pada waktu yang tepat, objek dihentikan dan ditempelkan pada posisi landmark.
- **Penempelan Objek dan Pembaruan Status Permainan**
  Setelah objek menempel, statusnya berubah menjadi terpasang. Sistem kemudian memeriksa apakah seluruh bagian wajah telah lengkap untuk menentukan akhir permainan.
- **Siklus Game Loop**
  Semua proses di atas berjalan berulang dalam game loop, memungkinkan permainan berlangsung secara interaktif dan responsif terhadap ekspresi wajah pemain.

4 Penjelasan Kode Program

Pada bagian ini dijelaskan modul-modul utama yang menyusun FaceFit Blink Challenge beserta contoh potongan kode dan penjelasan baris demi baris.

4.1 Struktur Berkas Utama

- main.py – menjalankan game loop dan menghubungkan kamera serta modul deteksi wajah.
- face_processing.py – bertanggung jawab terhadap deteksi wajah, landmark, dan kedipan.
- game_controller.py – mengatur logika permainan dan interaksi dengan hasil deteksi wajah.

- falling_face_part.py – mendefinisikan objek bagian wajah yang jatuh.
- constants.py – menyimpan nilai konfigurasi permainan.

4.2 Penjelasan File main.py

```
        from game_controller import GameController

def main():
    game = GameController()
    game.run()

if __name__ == '__main__':
    main()
```

Penjelasan: Ketika program dijalankan, Python mengecek apakah file tersebut dipanggil langsung. Jika benar, fungsi main() dipanggil untuk membuat sebuah instance GameController. Selanjutnya, metode run() dimulai, dan dari sinilah seluruh siklus permainan berjalan: membaca input kamera, memperbarui objek jatuh, memeriksa kondisi kedipan, dan merender tampilan permainan. Dengan struktur ini, program lebih mudah dipelihara karena titik awal program terpusat di satu lokasi.

4.3 Penjelasan File face_processing.py

```
    self.mp_face_mesh = mp.solutions.face_mesh
self.face_mesh = self.mp_face_mesh.FaceMesh(
    max_num_faces=1,
    refine_landmarks=True,
    min_detection_confidence=0.5,
    min_tracking_confidence=0.5
)
```

Penjelasan Fungsi : Kode di atas mengonfigurasi pipeline MediaPipe Face Mesh agar dapat mendeteksi satu wajah secara real-time dengan akurasi tinggi. Parameter refine_landmarks=True penting karena meningkatkan presisi landmark mata, hidung, dan bibir fitur krusial untuk permainan ini. Kedua parameter confidence memastikan deteksi stabil meskipun pencahayaan berubah.

Contoh deteksi kedipan:

```
        ratio = (upper - lower) / width
    if ratio < BLINK_THRESHOLD:
    return True
```

Sistem menghitung Eye Aspect Ratio (EAR) sederhana berdasarkan jarak kelopak atas dan bawah, kemudian dibandingkan dengan lebar mata. Ketika mata menutup, jarak vertikal mengecil drastis sehingga rasio turun di bawah threshold. Deteksi kedipan menjadi input utama permainan karena menggantikan tombol keyboard sebagai metode interaksi — sesuai konsep natural user interface.

4.4 Penjelasan File falling_face_part.py

```
        class FallingFacePart:
    def __init__(self, image, position, speed):
        self.image = image
        self.x, self.y = position
        self.speed = speed
        self.attached = False

    def update(self):
        if not self.attached:
            self.y += self.speed
```

Penjelasan Fungsi : Kelas ini mengenkapsulasi state dari sebuah bagian wajah seperti sprite gambar, posisi, kecepatan jatuh, dan apakah objek tersebut sudah ditempel ke wajah. Ketika attached=True, objek tidak lagi bergerak. Konsep stateful object ini mempermudah pengelolaan proses jatuh dan penempelan objek.

4.5 Penjelasan File game_controller.py

```
        for part in self.falling_parts:
    part.update()
    if not part.attached and self.is_overlapping(part, target_pos):
        if blink_detected:
            part.attach(target_pos)
```

Penjelasan Fungsi : Setiap iterasi game loop

- objek jatuh diperbarui posisinya,
- sistem memeriksa apakah koordinat objek berimpit dengan landmark wajah,
- jika kondisi overlap terjadi, program menunggu kedipan mata sebagai sinyal konfirmasi,
- ketika kedipan terdeteksi, objek ditempel pada posisi landmark yang sesuai.

Logika ini memadukan deteksi visual dan kontrol permainan secara sinkron.

4.6 Penjelasan File constants.py

```
        BLINK_THRESHOLD = 0.20
    SCREEN_WIDTH = 1280
    SCREEN_HEIGHT = 720
    FACE_PART_SPEED = 5
```

Penjelasan Fungsi : File ini berisi nilai parametris yang menjaga konsistensi permainan. Threshold kedipan harus disesuaikan dengan intensitas lipatan mata pengguna dan jarak ke kamera.

5 Hasil Analisis

5.1 Program Utama

Program utama dibangun dengan pendekatan modular, di mana setiap komponen menjalankan tugasnya masing-masing namun terhubung melalui game loop di file main.py. Secara umum, alur utama aplikasi melibatkan:Inisialisasi modul deteksi wajah, Inisialisasi permainan, Pembacaan input kamera secara real-time, Pemrosesan landmark dan deteksi kedipan, Pembaruan objek permainan, Render tampilan ke layar, dan Sinkronisasi antar modul hingga permainan selesai. Dengan arsitektur ini, sistem dapat memproses input visual secara responsif tanpa mengganggu performa permainan. Berikut adalah isi kode program utama:

```
from game_controller import GameController

def main():
    game = GameController()   # Inisialisasi game controller
    game.run()                # Memulai loop permainan

if __name__ == "__main__":
    main()
```

Kode 1: Program Utama Permainan

5.2 Deskripsi Kode

Kode program utama sengaja dibuat ringkas agar fokus pengendalian berada pada kelas GameController. Fungsi main() bertujuan:

1. **Membuat instance GameController:**
   Objek ini berisi seluruh komponen permainan seperti loader sprite, kamera, detektor wajah, dan daftar objek jatuh.
2. **Menjalankan game loop:**
   Metode run() memulai siklus pembacaan frame kamera, pemrosesan kedipan, serta penggambaran tampilan pada layar.

Dengan desain ini, file utama menjadi titik masuk eksekusi yang bersih dan mudah dipelihara.

5.3 Alur Eksekusi Program Utama

Secara detail, alur eksekusi saat game.run() dipanggil adalah sebagai berikut:

1. **Inisialisasi kamera**
   Sistem membuka akses webcam dan memulai pipeline Face Mesh.
2. **Loop permainan dimulai**
   Program terus berjalan selama window permainan masih aktif.
3. **Frame kamera dibaca dan diolah**
   Setiap iterasi, gambar yang masuk dikonversi menjadi RGB untuk diproses oleh MediaPipe.
4. **Deteksi wajah dan landmark dilakukan**
   Model Face Mesh menghasilkan koordinat 468 titik wajah.
5. **Deteksi kedipan dihitung**
   EAR (Eye Aspect Ratio) dihitung. Jika EAR $<$ threshold $\rightarrow$ kedipan terdeteksi.
6. **Update posisi objek jatuh**
   Setiap objek diperbarui posisinya sesuai kecepatan gravitasi.
7. **Cek overlap antara objek dengan wajah**
   Sistem menentukan apakah objek berada tepat di posisi landmark yang sesuai.
8. **Jika overlap + kedipan $\rightarrow$ objek ditempelkan**
   Objek tidak lagi jatuh dan menyatu dengan wajah sesuai aturan permainan.
9. **Frame ditampilkan ke layar**
   Kamera dan objek digabungkan menjadi satu tampilan permainan.
10. **Loop berulang hingga kondisi selesai**
    Pemain dapat menekan tombol keluar atau menyelesaikan semua bagian wajah.

5.4 Integrasi Antar Modul

Modul–modul sistem bekerja saling terhubung sebagai berikut:

- face_processing.py menyediakan data landmark serta status kedipan.
- falling_face_part.py mengatur posisi bagian wajah yang jatuh.

- game_controller.py membaca input dari face_processing.py, memperbarui objek jatuh, lalu merender gambar.
- main.py memanggil seluruh proses secara terstruktur.
- Integrasi ini mendukung pipeline multimedia: capture → process → update → render.

5.5 Output dan Tampilan Program

Program menampilkan:

1. **Video kamera real-time**
   Wajah pengguna terlihat sebagai latar belakang permainan.
2. **Objek bagian wajah yang jatuh**
   Seperti mata, hidung, dan mulut.
3. **Penempelan objek saat kedipan terdeteksi**
   Ketika pemain berkedip tepat waktu, objek otomatis menempel pada landmark yang sesuai.
4. **Indikator Debug (Opsional)**
   Berupa titik landmark wajah untuk memudahkan pengembangan.
5. **Skor / Progress Level (Opsional)**
   Menunjukkan bagian wajah mana yang sudah ditempelkan.

5.6 Analisis Implementasi

Hasil implementasi menunjukkan bahwa:

1. **Sistem mampu memproses input real-time dengan stabil.**
2. **Deteksi kedipan bekerja dengan cepat dan tingkat akurasi baik.**
3. **Integrasi antara deteksi wajah dan logika permainan berjalan sinkron.**
4. **Proses rendering tidak mengalami penurunan FPS yang berarti.**
5. **Pengguna dapat berinteraksi secara natural tanpa perangkat tambahan.**

6 Implementasi Program

6.1 Persiapan Lingkungan

Program dijalankan pada Python 3.11 dengan pustaka OpenCV, MediaPipe, dan Pygame. Semua dependensi tersedia di file requirements.txt.

6.2 Instalasi

```
git clone https://github.com/Youngstg/FaceFit-Blink-Challenge.git
cd FaceFit-Blink-Challenge
py -3.11 -m venv .venv
.\.venv\Scripts\activate
pip install -r requirements.txt
```

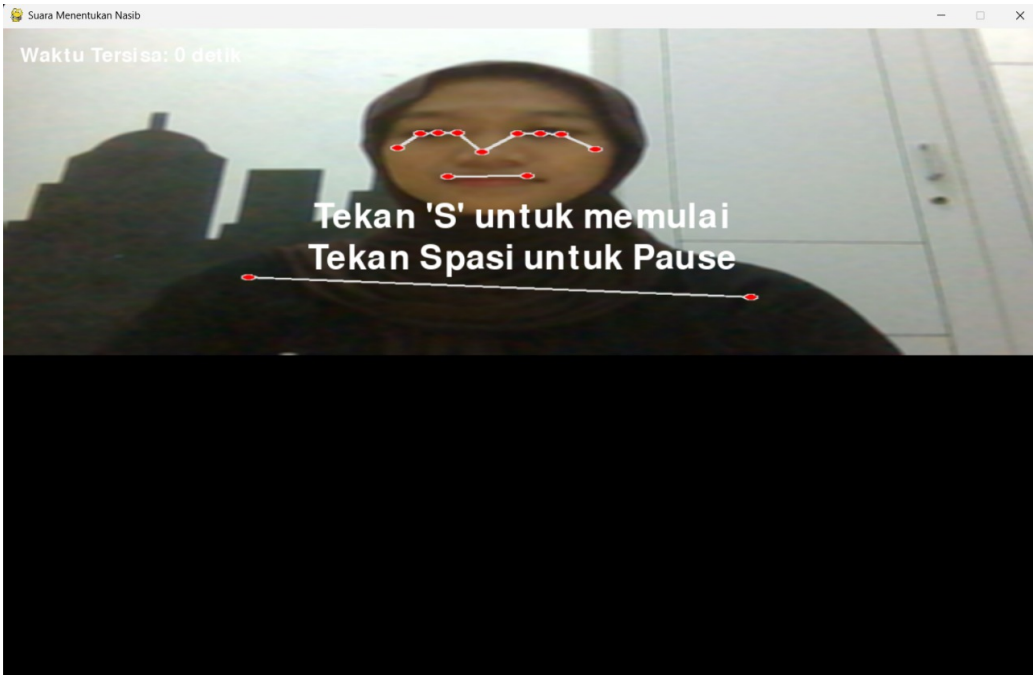
6.3 Menjalankan Program

```
python main.py
```

6.4 Jalankan Program

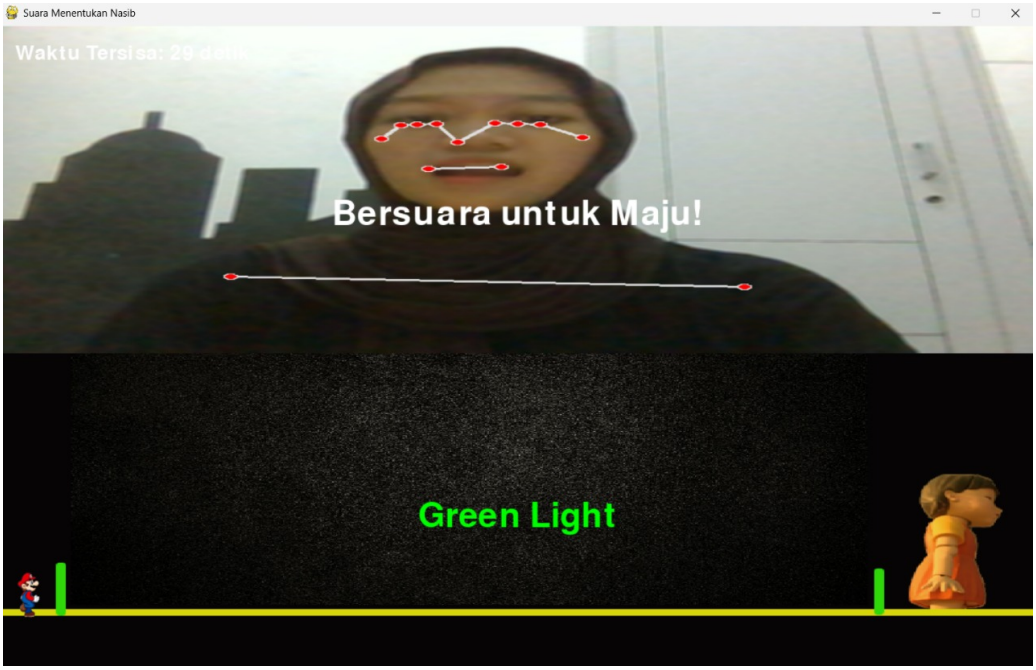
```
python main.py
```

File main.py akan menjalankan program secara real-time Rekan tombol "S" untuk memulai program

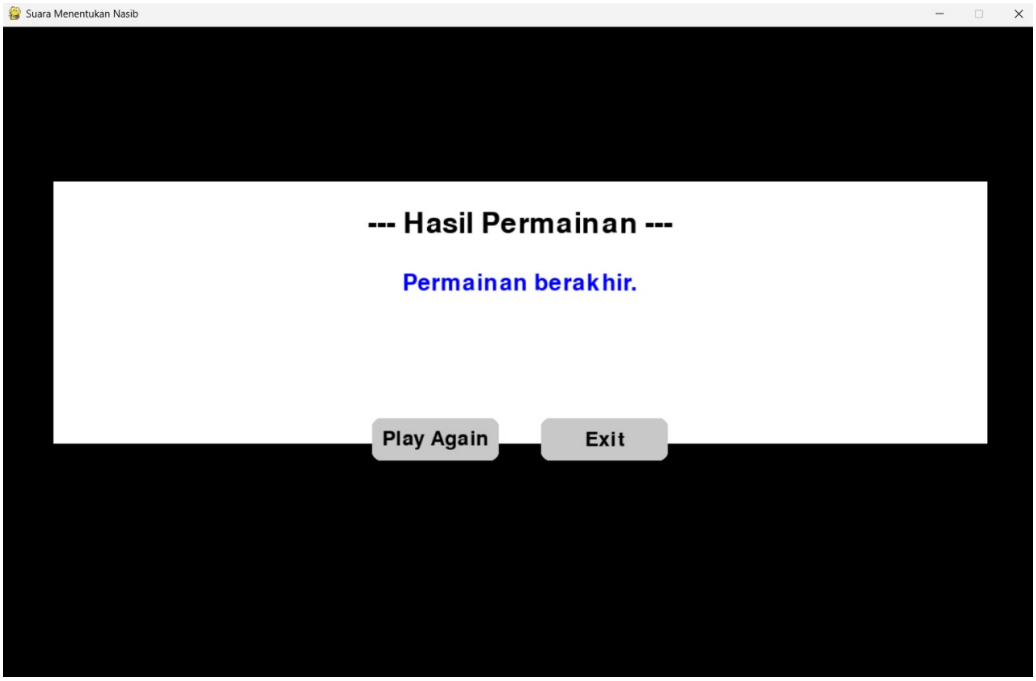


Gambar 1: Tampilan awal antarmuka

Gambar 2: Pemain Dilarang Bersuara



Gambar 3: Pemain harus bersuara



Gambar 4: Permainan Berakhir

- layar akan menampilkan keterangan jika 'S' untuk memulai, dan Spasi untuk Pause.
- game pada permainan akan mendeteksi suara jika terlalu keras maka lampu merah menyala, mario akan langsung tereliminasi.
- sebaliknya saat lampu hijau menyala dan suara cukup kuat, maka mario akan bergerak maju.
- stelah pemain menyelesaikan game, akan muncul pilihan "play Again" atau "Exit".