



Mata Kuliah: Sistem / Teknologi Multimedia (IF40305)

Tugas: Tugas Besar

Nama Anggota:

1. Dinda Joycehana (122140048)
2. Asavira Azzahra (122140067)
3. Hizkia Christovita Siahaan (122140110)

Tanggal: 11 Desember 2025

1 Deskripsi Projek

Live Captcha Challenge adalah sebuah permainan berbasis CAPTCHA yang menggunakan kamera dan gerakan tangan untuk menyusun puzzle secara live. Tidak seperti CAPTCHA puzzle biasa yang menggunakan gambar statis, proyek ini menampilkan potongan-potongan dari video webcam pengguna secara real-time. Pemain harus mengatur ulang potongan-potongan tersebut hingga membentuk tampilan kamera yang utuh.

Interaksi pada game ini dilakukan menggunakan gesture pinch (menggabungkan ibu jari dan telunjuk). Gesture tersebut dideteksi menggunakan MediaPipe Hands, sehingga pemain dapat mengambil dan memindahkan blok puzzle hanya dengan menggerakkan tangan di depan kamera. Hal ini menjadikan permainan lebih interaktif dan lebih sulit ditiru oleh bot.

Dalam proyek ini digunakan beberapa teknologi multimedia, seperti OpenCV untuk menangkap dan menampilkan video webcam, MediaPipe untuk mendeteksi posisi jari dan gerakan tangan, serta Pygame untuk memberikan efek suara saat pemain menukar blok atau menyelesaikan puzzle. Game ini juga dilengkapi fitur hitungan langkah, indikator posisi puzzle yang benar, dan efek kemenangan.

Secara sederhana, Live Captcha Challenge adalah permainan puzzle berbasis webcam yang menggabungkan teknologi computer vision dan gesture recognition untuk menciptakan pengalaman bermain yang unik, interaktif, dan lebih aman dibanding CAPTCHA puzzle tradisional.

2 Alat dan Bahan

Pada pengembangan proyek Live Captcha Challenge, digunakan beberapa alat dan bahan pendukung yang terdiri dari perangkat keras, perangkat lunak, serta library pendukung untuk menjalankan sistem pelacakan tangan dan puzzle berbasis webcam.

2.1 Perangkat Keras

Pengembangan dan pengujian proyek ini dilakukan menggunakan perangkat keras sebagai berikut:

- **Komputer/Laptop:** Digunakan untuk menulis kode, menjalankan aplikasi, dan menguji performa game.
- **Webcam:** Kamera eksternal atau bawaan laptop yang berfungsi menangkap video secara real-time. Webcam feed akan dipotong menjadi puzzle dan digunakan sebagai input bagi sistem *hand tracking*. Direkomendasikan menggunakan webcam beresolusi minimal 720p, karena kualitas webcam mempengaruhi akurasi deteksi tangan dan pengalaman bermain.

2.2 Bahasa Pemrograman

Proyek ini dibuat menggunakan Python versi 3.10. Versi ini dipilih karena stabil, kompatibel dengan library terbaru, dan mendukung penggunaan OpenCV serta MediaPipe tanpa masalah.

2.3 Library

Beberapa library Python yang digunakan untuk membangun fitur-fitur inti pada proyek ini yaitu:

- **OpenCV**: Digunakan untuk menangkap video dari webcam, memproses frame video, dan menampilkan antarmuka permainan.
- **MediaPipe**: Library ini menyediakan model *hand tracking* yang digunakan untuk mendeteksi posisi jari dan gerakan tangan secara real-time.
- **Pygame**: Digunakan untuk mengelola audio dalam game, seperti efek suara saat pemain menukar blok puzzle atau menyelesaikan permainan.
- **NumPy**: Digunakan untuk manipulasi array dan operasi matematika yang diperlukan dalam pemrosesan gambar dan logika permainan.

2.4 Struktur Proyek

Proyek Live Captcha Challenge memiliki struktur direktori sebagai berikut:

```
Live_Captcha_Challenge/
|-- game/
|   |-- __init__.py
|   |-- game_renderer.py
|   |-- hand_tracker.py
|   |-- puzzle_pieces.py
|   |-- puzzle.py
|   |-- sound.py
|-- assets/
|   |-- sounds/
|-- main.py
|-- requirements.txt
|-- README.md
```

3 Penjelasan

Program Live Captcha Challenge dibangun berdasarkan beberapa modul Python yang saling terintegrasi untuk menjalankan seluruh fitur permainan. Setiap modul memiliki perannya masing-masing, mulai dari pendekripsi tangan, logika permainan puzzle, rendering tampilan, hingga pemrosesan suara. Bagian ini menjelaskan lingkungan pengembangan, instalasi dependensi, serta penjelasan setiap modul yang digunakan dalam sistem.

3.1 Setup Lingkungan Pengembangan

Proyek ini dikembangkan menggunakan Python 3.10 pada environment virtual untuk menjaga isolasi dependensi, sehingga tidak mengganggu environment sistem utama. Pada pengembangan proyek ini menggunakan UV. Berikut ini langkah-langkah untuk menyiapkan UV environment :

```
1 pip install uv #Instalasi uv
2 uv venv --python=python3.10 #Membuat environment virtual dengan python 3.10
3 .venv\Scripts\activate #Aktifkan environment virtual
4
```

Kode 1: Membuat UV environment

3.2 Instalasi *Library*

Agar program dapat berjalan dengan baik, seluruh *library* pendukung harus dipasang terlebih dahulu.

```
1 uv pip install numpy opencv-python mediapipe pygame
2
```

Dependensi yang diperlukan pada proyek ini sudah disediakan dalam file **requirements.txt**, sehingga proses instalasi *library* dapat dilakukan melalui satu perintah berikut:

```
1 uv pip install -U -r requirements.txt
2
```

Kode 2: Instalasi dependensi melalui file requirements.txt

3.3 Mekanisme Kerja Sistem

Permainan ini menggunakan konsep scrambled glass blocks, yaitu potongan-potongan puzzle yang tetap menampilkan video webcam secara langsung (live feed), bukan gambar statis. Dengan konsep ini, pemain tetap dapat melihat gerakan real-time di setiap blok puzzle.

Perbedaan utama dengan Captcha puzzle biasa :

- **Live Feed pada Setiap Blok:** Setiap potongan puzzle adalah "window" yang menampilkan bagian tertentu dari video webcam secara langsung, sehingga pemain dapat melihat gerakan real-time di setiap blok (seluruh video dari webcam selalu aktif di balik potongan puzzle).
- **Interaksi dengan Gesture Tangan:** Pemain menggunakan gesture pinch (menggabungkan ibu jari dan telunjuk) untuk memilih dan memindahkan potongan puzzle, yang dideteksi menggunakan MediaPipe Hands.
- **Tantangan Keamanan Lebih Tinggi:** Dengan menggunakan video live feed dan gesture recognition, sistem ini lebih sulit untuk diotomatisasi oleh bot dibandingkan dengan CAPTCHA puzzle tradisional.

3.4 Hand Tracker menggunakan MediaPipe

Modul **hand_tracker.py** berfungsi sebagai sistem pendekripsi tangan yang digunakan untuk interaksi di dalam permainan. Modul ini memuat kelas **HandTracker** yang menangani proses inisialisasi model MediaPipe Hands, pemrosesan frame, pendekripsi gesture pinch, serta pengambilan koordinat jari pemain.

Kode berikut digunakan untuk menginisialisasi model deteksi tangan:

```
1 self.mp_hands = mp.solutions.hands
2 self.hands = self.mp_hands.Hands(
3     static_image_mode=False,
4     max_num_hands=1,
5     min_detection_confidence=0.5,
6     min_tracking_confidence=0.5
7 )
8
```

Fungsi utama dalam modul ini:

- **process_frame(frame_rgb)** Memproses sebuah frame RGB menjadi objek hasil deteksi Mediapipe.

```
1     return self.hands.process(frame_rgb)
2
```

- **get_hand_position(hand_landmarks, frame_shape)** Mengambil koordinat piksel jari telunjuk (landmark ke-8) untuk menentukan posisi tangan di layar.

```
1     index_tip = hand_landmarks.landmark[8]
2     return (int(index_tip.x * w), int(index_tip.y * h))
3
```

- **is_pinching(hand_landmarks)** Menentukan apakah pemain sedang melakukan gesture *pinch* dengan menghitung jarak Euclidean antara ibu jari (landmark 4) dan telunjuk (landmark 8).

```
1     distance = math.sqrt(
2         (thumb_tip.x - index_tip.x)**2 +
3         (thumb_tip.y - index_tip.y)**2
4     )
5     return distance < self.pinch_threshold
6
```

- **draw_landmarks(frame, hand_landmarks)** Menggambar titik dan koneksi landmark tangan pada frame.

Modul ini menjadi fondasi interaksi dalam game karena semua aksi pemain dilakukan melalui gesture tangan.

3.5 Webcam Puzzle

Modul **puzzle_pieces.py** bertanggung jawab untuk membuat, mengacak, dan memvalidasi potongan puzzle. Setiap potongan puzzle memiliki dua atribut penting: **correct_pos** (posisi yang benar) dan **current_pos** (posisi saat ini).

Pembuatan potongan puzzle dilakukan sebagai berikut:

```
1     self.pieces.append({
2         'correct_pos': (col, row),
3         'current_pos': (col, row),
4         'id': row * self.grid_size + col
5     })
6
```

Pengacakan puzzle dilakukan menggunakan **random.shuffle** terhadap seluruh kemungkinan posisi:

```
1     positions = [(col, row) for row in range(...) for col in range(...)]
2     random.shuffle(positions)
3
```

Proses memilih potongan berdasarkan posisi tangan dilakukan melalui fungsi:

```
1     grid_x = x // piece_size
2     grid_y = y // piece_size
3
```

Pertukaran posisi dua potongan puzzle dilakukan dengan:

```
1     piece1['current_pos'], piece2['current_pos'] = \
2         piece2['current_pos'], piece1['current_pos']
3
```

Pengecekan apakah puzzle sudah selesai dilakukan dengan memeriksa apakah seluruh `current_pos` sama dengan `correct_pos`.

Modul ini menyediakan logika inti permainan: struktur data puzzle, pengacakan, pemilihan, dan pengecekan kondisi selesai.

3.6 Tampilan Game

Modul `game_renderer.py` mengatur seluruh proses visualisasi puzzle. Fungsi yang paling penting adalah `draw_puzzle()`, yang menampilkan potongan video webcam ke posisi masing-masing pada grid puzzle.

Berikut adalah cara program menampilkan potongan video webcam:

```
1 piece_view = cropped_frame[  
2     src_y:src_y+self.piece_size,  
3     src_x:src_x+self.piece_size  
4 ].copy()  
5
```

Potongan tersebut kemudian digambar ke tampilan puzzle sesuai posisi saat ini:

```
1 display[y:y+self.piece_size, x:x+self.piece_size] = piece_view  
2
```

Renderer juga menggambar border berwarna hijau apabila potongan sudah berada pada posisi yang benar:

```
1 color = (0, 255, 0) if piece['current_pos'] == piece['correct_pos'] else (255, 255, 255)  
2
```

Selain itu renderer juga menyediakan:

- `draw_start_screen()` — menampilkan instruksi awal.
- `draw_move_count()` — menampilkan jumlah langkah pemain.
- `draw_solved_screen()` — overlay hijau dan teks ketika puzzle selesai.

Renderer memastikan permainan terlihat jelas, informatif, dan responsif.

3.7 Pemrosesan Suara

Modul `sound.py` menggunakan *Pygame Mixer* untuk memuat dan memainkan efek suara. Inisialisasi mixer dilakukan dengan:

```
1 pygame.mixer.init()  
2
```

Memuat file audio dilakukan menggunakan path relatif:

```
1 click_sound = pygame.mixer.Sound(os.path.join(SOUND_DIR, "click.wav"))  
2 win_sound   = pygame.mixer.Sound(os.path.join(SOUND_DIR, "win.wav"))  
3
```

Suara klik dimainkan dengan:

```
1 click_sound.play()  
2 click_sound.set_volume(1.0)  
3
```

Sedangkan suara kemenangan dimainkan dengan:

```
1 win_sound.set_volume(0.5)  
2 win_sound.play()  
3
```

Modul ini memberikan umpan balik audio sehingga permainan terasa lebih hidup dan interaktif.

3.8 Program Utama

File `main.py` merupakan titik masuk permainan. Program akan mencetak instruksi awal, membuat objek `LiveGlassPuzzle`, dan menjalankan permainan melalui fungsi `run()`.

Pemanggilan utama dilakukan melalui:

```
1 puzzle = LiveGlassPuzzle(grid_size=3)
2 puzzle.run()
3
```

Di dalam `run()`, terdapat game loop yang menangani:

- pembacaan frame webcam,
- pemrosesan gesture tangan,
- pembaruan tampilan puzzle,
- pengecekan kondisi selesai,
- serta input keyboard (SPACE, R, Q).

Game loop terus berjalan hingga pemain menekan tombol Q:

```
1 while cap.isOpened():
2     ret, frame = cap.read()
3     ...
4
```

Program utama ini menghubungkan seluruh modul menjadi satu sistem permainan yang interaktif.

3.9 Pembahasan Implementasi Sistem

Pada implementasi sistem, filter Live Captcha Challenge dibangun dengan mengintegrasikan beberapa modul utama yaitu `hand_tracker.py`, `puzzle_pieces.py`, `game_renderer.py`, `sound.py`, serta `main.py` sebagai pengendali utama. Setiap modul memiliki peran penting dalam menjalankan fungsionalitas permainan secara menyeluruh.

3.9.1 Deteksi Tangan dan Gesture Pinch

Deteksi tangan menggunakan MediaPipe Hands merupakan komponen inti dalam interaksi permainan. Modul `hand_tracker.py` mengolah frame yang ditangkap dari webcam dan menghasilkan posisi koordinat jari pemain secara real-time. Gesture *pinch* (menggabungkan ibu jari dan telunjuk) digunakan sebagai bentuk “klik virtual” untuk memilih sebuah potongan puzzle.

Proses pinch dihitung dengan mengukur jarak Euclidean antara landmark jari yang relevan. Jika jarak di bawah ambang batas tertentu, sistem menganggap pemain sedang melakukan pinch. Pendekatan ini memungkinkan interaksi yang natural dan responsif tanpa menggunakan perangkat tambahan.

3.9.2 Pemotongan Frame Webcam Menjadi Puzzle

Modul `puzzle_pieces.py` bertanggung jawab untuk membagi frame webcam menjadi beberapa potongan dengan ukuran yang sama. Setiap potongan direpresentasikan sebagai sebuah objek yang memiliki:

- posisi benar (`correct_pos`),
- posisi saat ini (`current_pos`),

- dan identitas unik.

Setelah potongan dibuat, modul ini juga melakukan pengacakan posisi sehingga susunan awal puzzle tidak beraturan. Tantangan pemain adalah mengembalikannya ke susunan awal seperti tampilan kamera sebenarnya.

3.9.3 Mekanisme Pertukaran Potongan Puzzle

Ketika sistem mendeteksi bahwa koordinat pinch berada pada salah satu potongan puzzle, potongan tersebut dianggap terpilih. Jika pemain kemudian melakukan pinch pada potongan lain, maka kedua potongan tersebut dipertukarkan.

Proses pertukaran dilakukan dengan menukar nilai **current_pos** antar potongan. Interaksi ini dirancang menyerupai permainan puzzle klasik, tetapi dengan tambahan tantangan karena semua potongan menampilkan bagian video yang masih aktif bergerak.

3.9.4 Proses Rendering Tampilan Puzzle

Modul **game_renderer.py** melakukan rendering seluruh tampilan permainan. Setiap potongan puzzle ditampilkan berdasarkan posisi saat ini, sambil tetap mengambil konten video secara langsung dari webcam. Sistem juga memberikan indikator visual berupa:

- **border hijau** ketika potongan berada di posisi yang benar,
- **border putih** ketika potongan belum sesuai,
- teks instruksi seperti “Pinch untuk memilih puzzle” atau “Puzzle Solved”.

Renderer bekerja pada setiap frame sehingga permainan tampak dinamis dan interaktif.

3.9.5 Umpaman Balik Audio

Pemrosesan audio dilakukan oleh modul **sound.py**. Dua suara utama digunakan dalam permainan:

- efek suara klik saat pemain menukar potongan,
- efek suara kemenangan ketika puzzle berhasil diselesaikan.

Penambahan audio meningkatkan pengalaman bermain dan membuat interaksi terasa lebih imersif.

3.9.6 Alur Program Utama

File **main.py** menghubungkan seluruh modul dan menjalankan game loop. Pada alur utama, program melakukan:

1. membaca frame webcam,
2. mendeteksi tangan dan gesture,
3. memperbarui logika permainan (pertukaran puzzle),
4. menampilkan tampilan puzzle,
5. memberikan feedback audio,
6. menghentikan permainan jika pemain menekan tombol tertentu.

Game loop ini berjalan terus-menerus hingga pemain keluar dari permainan.

3.10 Analisis Performa Sistem

Berdasarkan hasil implementasi, sistem dapat berjalan secara real-time pada laptop dengan spesifikasi standar. MediaPipe Hands mampu mendeteksi gesture dengan cukup stabil selama pencahayaan ruangan memadai. Proses rendering puzzle juga berjalan lancar karena setiap potongan hanya memerlukan operasi pemotongan frame dan penyalinan matriks sederhana.

Tantangan utama pada performa muncul ketika ukuran grid diperbesar (misalnya 5x5), karena jumlah operasi pemotongan dan rendering meningkat. Meskipun demikian, sistem tetap dapat berjalan dengan baik tanpa penurunan frame rate yang signifikan pada pengujian.

3.11 Keterbatasan Implementasi

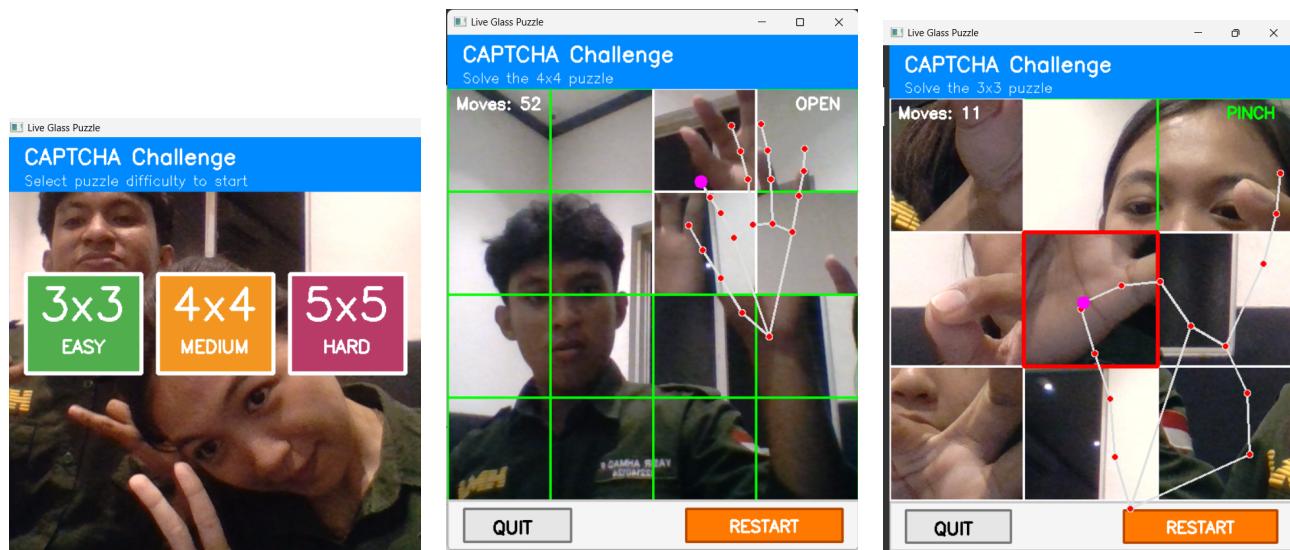
Beberapa keterbatasan pada sistem ini antara lain:

- Akurasi gesture dapat menurun pada kondisi pencahayaan yang buruk.
- Sistem hanya mendukung satu tangan aktif; deteksi dua tangan belum diimplementasikan.
- Puzzle menggunakan batasan kotak persegi sehingga tidak mendukung bentuk puzzle yang kompleks.

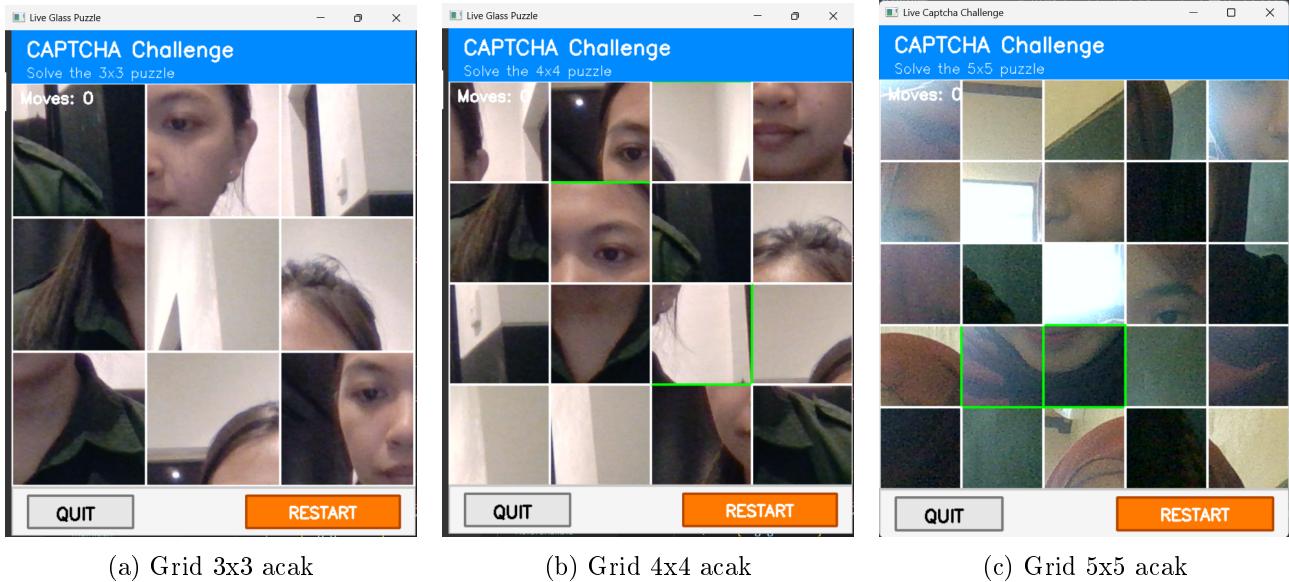
Keterbatasan ini dapat diperbaiki pada versi pengembangan selanjutnya dengan menambahkan model gesture yang lebih canggih serta optimasi tampilan.

4 Hasil

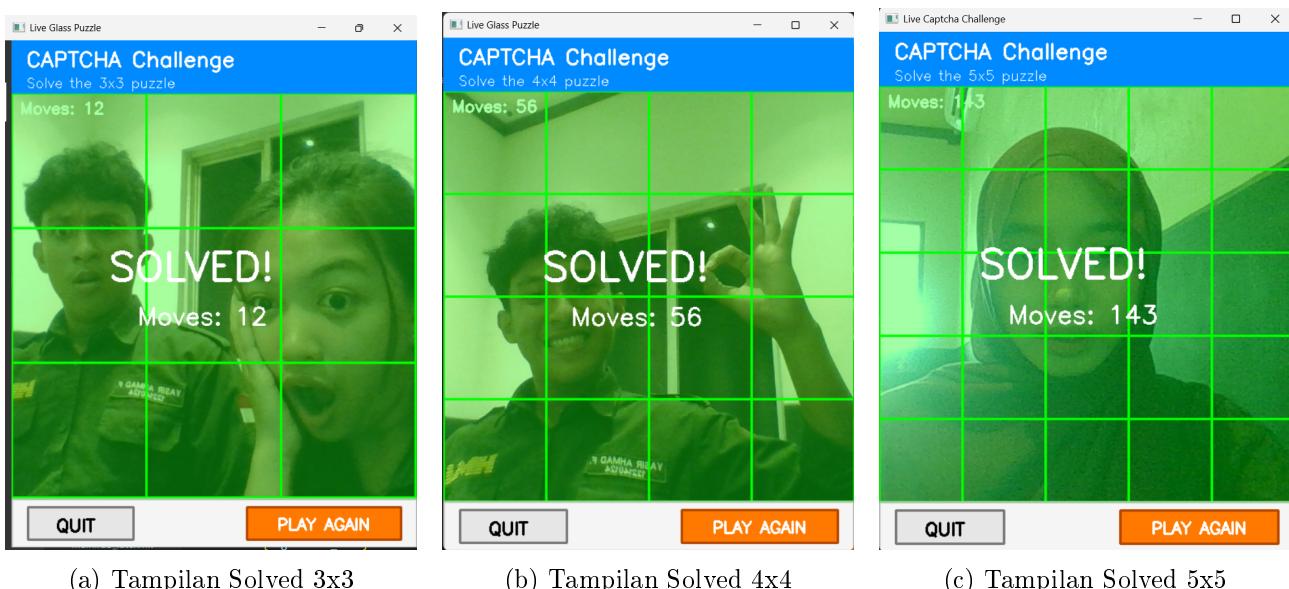
Hasil penerapan dari filter yang kami kembangkan ditunjukkan pada Gambar berikut.



Gambar 1: Tampilan start, open hand, dan pinch pada Live Captcha Challenge.



Gambar 2: Tampilan grid acak 3x3, 4x4, dan 5x5.



Gambar 3: Tampilan grid solved 3x3, 4x4, dan 5x5.

(Penjelasan gambar hasil)

5 Kesimpulan

Berdasarkan proses perancangan, implementasi, dan pengujian yang telah dilakukan, proyek *Live Captcha Challenge* berhasil dikembangkan sebagai permainan interaktif berbasis *computer vision* yang menggabungkan teknologi hand tracking dan video webcam secara real-time. Sistem mampu mendeteksi gesture *pinch* secara akurat sehingga pengguna dapat berinteraksi langsung dengan potongan puzzle yang berasal dari *live feed* kamera.

Integrasi seluruh modul utama, yaitu `hand_tracker.py`, `puzzle_pieces.py`, `game_renderer.py`, `sound.py`, dan `main.py`, menghasilkan permainan yang responsif, stabil, serta mudah dimainkan. MediaPipe Hands mendukung deteksi posisi jari secara real-time, sementara OpenCV menyediakan pengolahan dan rendering puzzle yang halus. Penambahan efek suara melalui Pygame turut meningkatkan pengalaman bermain melalui umpan balik audio pada setiap interaksi.

Hasil pengujian menunjukkan bahwa permainan dapat berjalan dengan baik pada perangkat dengan spesifikasi standar, baik untuk konfigurasi grid 3x3, 4x4, maupun 5x5. Penggunaan *live feed* pada setiap potongan puzzle memberikan pengalaman yang dinamis serta menciptakan tingkat keamanan lebih tinggi dibandingkan CAPTCHA berbasis gambar statis, sehingga lebih sulit ditiru oleh sistem otomatis.

Meskipun demikian, permainan ini masih memiliki beberapa keterbatasan, seperti sensitivitas terhadap pencahayaan, deteksi satu tangan saja, serta bentuk puzzle yang masih sederhana. Keterbatasan ini sekaligus membuka peluang pengembangan lanjutan, seperti peningkatan robustness deteksi gesture, penambahan variasi interaksi, maupun pengembangan bentuk puzzle yang lebih kompleks.

Secara keseluruhan, proyek ini berhasil mencapai tujuan yang ditetapkan, yaitu mengembangkan permainan CAPTCHA interaktif berbasis webcam dengan gesture tangan sebagai mekanisme kontrol utama. Implementasi yang dihasilkan tidak hanya memberikan pengalaman bermain yang menarik, tetapi juga memiliki potensi sebagai pendekatan CAPTCHA yang lebih aman dan sulit dieksplorasi oleh sistem otomatis.

References

https://www.tiktok.com/@stereo.drift/video/7541439754634661138?is_from_webapp=1&sender_device=pc&web_id=7564777874063132161