Drowsy Driver Warning System Program Based Raspberry Pi 4 Model B With Speaker Audio Output And I2C 128x64 OLED
Program Sistem Peringatan Pengemudi Mengantuk Berbasis Raspberry Pi 4 Model B Dengan Output Audio Speaker Dan I2C OLED 128x64

Achmad Arif Dwi Prasetyo[1)], Indah Sulistiyowati*[2),] Shazana Dhiya Ayuni[3),] Arief Wisaksono[4)]

[1,2,3,4)]Program Studi Teknik Elektro, Universitas Muhammadiyah Sidoarjo, Indonesia
*Email Penulis Korespondensi: indah_sulistiyowati@umsida.ac.id

Abstract. *Drowsy driving is a serious hazard that significantly increases the risk of road accidents, impairs cognitive function, and reduces driver ability. Computer vision-based image processing has emerged as an effective method to detect drowsiness by analyzing facial features, especially eye movements and blink duration. Raspberry Pi 4 Model B is a device that supports to implement a real-time drowsiness detection system using computer vision-based image processing. Using Raspberry pi 4 model B as a computer used for computer vision as well as digital image processing, using two outputs as a warning for drowsy drivers, namely audio speakers to provide warnings in the form of sound and 128x64 OLED I2C as a visual warning to the driver. Shape Predictor 68 Facial Landmark a technique used in computer vision to detect and localize key facial features based on pre-trained machine learning models. This method is calculated using the distance between the main points around each eye, which is extracted using the Shape Predictor 68 Facial Landmark model in dlib. as long as the eyes are closed, the EAR value is close to the value 0, when the eyes are open the EAR value can be said to be greater than the value 0. The results of these tests obtained an average response time of 0.75 for audio speakers on. with the fastest response time of 0.81. This research can prove that the Raspberry Pi 4 model B can be used for drowsiness detection and warning. In the future, the system can be developed in terms of more complex software so that the system can be implemented with varying conditions.*

Keywords – *Raspberry Pi 4 Model B; Eye Aspect Ratio(EAR);Drowsinnes*

Abstrak. *Mengemudi dalam keadaan mengantuk merupakan bahaya serius yang secara signifikan meningkatkan risiko kecelakaan di jalan raya, merusak fungsi kognitif, dan mengurangi kemampuan pengemudi. Pemrosesan gambar berbasis visi komputer telah muncul sebagai metode yang efektif untuk mendeteksi rasa kantuk dengan menganalisis fitur wajah, terutama gerakan mata dan durasi berkedip. Raspberry Pi 4 Model B adalah perangkat yang mendukung untuk mengimplementasikan sistem deteksi kantuk secara real-time menggunakan pemrosesan gambar berbasis visi komputer. Menggunakan Raspberry pi 4 model B sebagai komputer yang digunakan untuk computer vision sekaligus pengolahan citra digital, dengan menggunakan dua buah output sebagai peringatan bagi pengemudi yang mengantuk, yaitu audio speaker untuk memberikan peringatan berupa suara dan OLED I2C 128x64 sebagai peringatan visual kepada pengemudi. Shape Predictor 68 Facial Landmark merupakan teknik yang digunakan dalam computer vision untuk mendeteksi dan melokalisasi fitur-fitur wajah utama berdasarkan model pembelajaran mesin yang telah dilatih sebelumnya. Metode ini dihitung dengan menggunakan jarak antara titik-titik utama di sekitar setiap mata, yang diekstraksi menggunakan model Shape Predictor 68 Facial Landmark pada dlib. selama mata tertutup, nilai EAR mendekati nilai 0, saat mata terbuka nilai EAR dapat dikatakan lebih besar dari nilai 0. Hasil dari pengujian tersebut didapatkan rata-rata waktu respon sebesar 0.75 untuk audio speaker on. dengan waktu respon tercepat sebesar 0.81. Penelitian ini dapat membuktikan bahwa Raspberry Pi 4 model B dapat digunakan untuk deteksi dan peringatan kantuk. Untuk kedepannya, sistem dapat dikembangkan dari segi perangkat lunak yang lebih kompleks sehingga sistem dapat diimplementasikan dengan kondisi yang bervariasi.*

Kata Kunci - *Raspberry Pi 4 Model B; Eye Aspect Ratio(EAR); Mengantuk*

I. PENDAHULUAN

Mengemudi dalam keadaan mengantuk merupakan bahaya serius namun sering diabaikan yang secara signifikan meningkatkan risiko kecelakaan di jalan raya. Kelelahan mengganggu fungsi kognitif, memperlambat waktu reaksi, dan mengurangi kemampuan pengemudi untuk mengambil keputusan dengan cepat [1]. Penelitian menunjukkan bahwa kurang tidur dan jam mengemudi yang panjang merupakan kontributor utama terhadap kecelakaan yang disebabkan oleh rasa kantuk, yang sering kali mengakibatkan kewaspadaan saat mengemudi secara alamiah menurun

[2]. Tidak seperti mengemudi dalam keadaan mabuk, mengemudi dalam keadaan mengantuk seringkali tidak disadari, karena gejala-gejala seperti sering berkedip, menganggukkan kepala, dan keluar jalur mungkin tidak langsung terlihat [3]. Mengingat bahaya yang terkait dengan kelelahan, solusi berbasis teknologi seperti sistem pendeteksi kantuk secara real-time menjadi semakin penting.

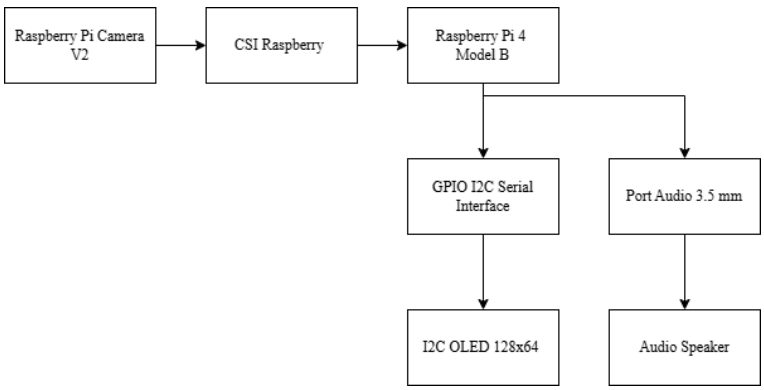
Deteksi kantuk merupakan area penelitian yang penting dalam keselamatan di jalan raya, yang bertujuan untuk mencegah kecelakaan yang disebabkan oleh kelelahan pengemudi melalui sistem pemantauan waktu nyata. Pemrosesan gambar berbasis visi komputer telah muncul sebagai metode yang efektif untuk mendeteksi rasa kantuk dengan menganalisis fitur wajah, terutama gerakan mata dan durasi kedipan [4]. Namun, pendekatan alternatif melibatkan penggunaan sensor denyut nadi untuk memantau sinyal fisiologis, seperti variabilitas detak jantung, yang berubah ketika seseorang mengantuk [5]. Meskipun metode berbasis sensor denyut nadi menawarkan akurasi yang lebih tinggi dalam mendeteksi kelelahan fisiologis, metode ini membutuhkan kontak langsung dengan pengemudi, sehingga kurang praktis untuk penggunaan sehari-hari [6]. Sebaliknya, deteksi kantuk berbasis visi komputer sepenuhnya tanpa kontak, sehingga memungkinkan pemantauan terus menerus tanpa ketidaknyamanan atau intervensi pengemudi.

Raspberry Pi 4 Model B telah menjadi perangkat yang mendukung untuk mengimplementasikan sistem pendeteksi kantuk secara real-time dengan menggunakan pemrosesan gambar berbasis visi komputer. Dengan prosesor quad-core, GPU yang telah tertanam dalam perangkat, dan dukungan untuk modul kamera beresolusi tinggi, Raspberry Pi memungkinkan eksekusi yang efisien dari algoritma pendeteksian tengara wajah untuk melacak gerakan mata dan pemrosesan untuk mendeteksi kantuk [7]. Visi komputer berbasis Raspberry Pi 4 model B menjadi solusi yang tidak mengganggu, terjangkau, dan mudah digunakan untuk lingkungan mengemudi di dunia nyata. Dengan mengintegrasikan efisiensi komunikasi antar perangkat keras dengan teknik pemrosesan gambar.

Mengemudi dalam keadaan mengantuk merupakan penyebab utama kecelakaan di jalan raya, sehingga diperlukan pengembangan sistem pemantauan pengemudi secara real-time untuk meningkatkan keselamatan di jalan raya. Penelitian ini menyajikan Sistem Peringatan Pengemudi Mengantuk yang diimplementasikan pada Raspberry Pi 4 Model B, memanfaatkan pemrosesan gambar berbasis visi komputer untuk mendeteksi kelelahan pengemudi. Kombinasi perangkat keras dan perangkat lunak ini memberikan solusi pendeteksian kantuk yang tidak mengganggu, hemat biaya, dan waktu nyata, memastikan umpan balik langsung dari pengemudi. Dengan mengintegrasikan komputasi tertanam, pemrosesan gambar, dan mekanisme peringatan multimodal, sistem berbasis Raspberry Pi ini menawarkan pendekatan efisien dan praktis yang diharapkan dapat mengurangi kecelakaan yang berkaitan dengan kelelahan.

II. METODE

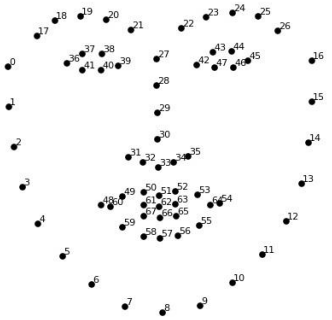
Sistem peringatan pengantuk berbasis Raspberry Pi 4 model B dengan output audio speaker dan I2C OLED 128x64, merupakan penelitian yang berbasiskan metode R&D (*Research and Development*). Menggunakan Raspberry pi 4 model B sebagai komputer yang digunakan untuk visi komputer dan juga pemrosesan gambar digital, menggunakan dua outputan sebagai peringatan pengemudi mengantuk yakni audio speaker untuk memberikan peringatan berupa suara dan I2C OLED 128x64 sebagai peringatan secara visual kepada pengemudi. Pengolahan gambar digital untuk mendeteksi wajah maupun mata menguunakan model Shape Predictor_68_Facial Landmark , untuk mengukur tingkat kantuk pada pengemudi menggunakan metode EAR (*Eye Aspect Ratio*) yakni nilai yang didapatkan berdasarkan rasio terbuka ataupun tertutup aspek mata [8].



Gambar 1. Desain Perangkat Keras

Gambar 1. Menunjukan bagaiaman desain perangkat keras yang dilustrasikan dengan blok blok persegi panjang. Pada tahap awal untuk proses inputan terdapat Raspberry pi camera V2 yakni perangkat difungsikan untuk menangkap gambar yang terhubung melalui komunikasi CSI (*Camera Serial Interface*) yang tersedia pada perangkat komputer papan tunggal Raspberry Pi 4 model B [9]. Tahap proses menggunakan Raspberry Pi 4 model B, menerima umpan inputan dan melakukan pengolahan gambar digital dari kamera raspberry lalu melakukan tindakan komputasi untuk mengidentifikasi mengantuk atau tidak untuk melakukan tindakan mengaktifkan outputan.

Tahap terakhir yakni outputan dari sistem terdiri dari 2 komponen. Komponen pertama ouputan audio speaker terhubung dengan Raspberry Pi 4 model B melalui port audio diameter 3.5 mm. Komponen kedua yakni I2C OLED 128x64 koomponen menampilkan karakter melalui layar OLED dengan layar grafis yang memiliki 128 kolom dan 64 baris, terhubung dengan pin GPIO (*General Purpose Input Output*) melalui komunikasi I2C yang tersedia pada Raspberry Pi 4 model B [10].



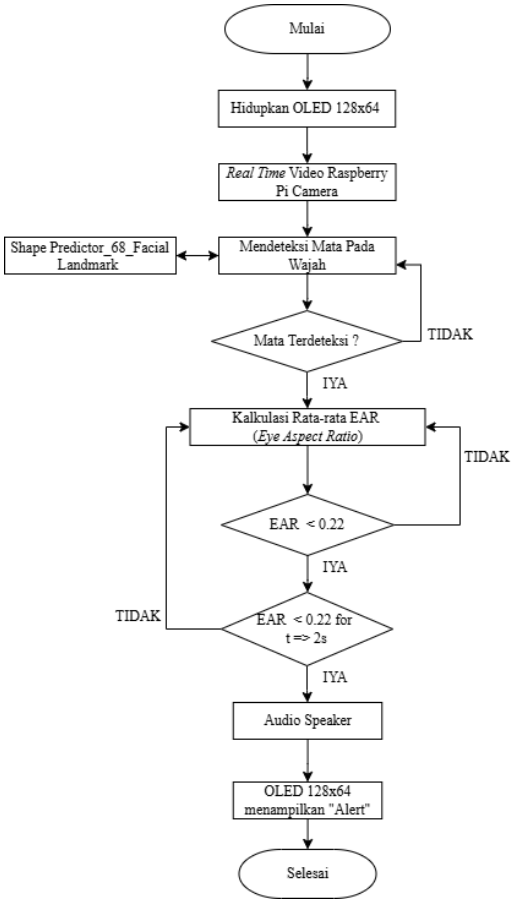
Gambar 2. Ilustrasi *Shape Predictor_68_Facial Landmark*

Gambar 2. Merupakan ilustrasi dari model Shape Predictor_68_Facial yang diwakili oleh titik koordinat. Shape Predictor 68 Facial Landmark teknik yang digunakan dalam visi komputer untuk mendeteksi dan melokalisasi fitur-fitur wajah utama berdasarkan model pembelajaran mesin yang telah dilatih sebelumnya. Dikembangkan menggunakan algoritme prediktor bentuk dlib, algoritme ini memetakan 68 titik tengara spesifik pada wajah, termasuk mata, alis, hidung, mulut, dan garis rahang [11]. Model ini dilatih pada set data besar gambar wajah yang dianotasi dan menerapkan pohon regresi ensembel untuk memprediksi posisi tengara secara akurat. Dalam sistem peringatan pengemudi mengantuk yang diimplementasikan dalam program di atas, model Shape Predictor 68 digunakan untuk mendeteksi dan mengekstrak area mata dari wajah pengemudi. Dengan menggunakan pustaka OpenCV dan dlib, sistem mengisolasi koordinat mata kiri dan kanan dan menghitung EAR untuk menentukan durasi penutupan mata [12].

$$EAR = \frac{\left||P2-P6|\right|+||P3-P5||}{2||P1-P4||} \qquad (1)$$

Eye Aspect Ratio (EAR) adalah metode yang banyak digunakan dalam deteksi kantuk berbasis visi komputer untuk mengukur keterbukaan dan penutupan mata melalui analisis geometris landmark wajah. Metode ini dihitung menggunakan jarak antara titik utama di sekitar setiap mata, yang diekstraksi menggunakan model Shape Predictor 68 Facial Landmark di dlib. Rumus EAR dapat didefinisikan seperti pada persamaan (1), di mana p2-p6 dan p3-p5 adalah jarak vertikal antara landmark kelopak mata, dan p1-p4 adalah jarak horizontal antara sudut mata [13].

Dalam sistem peringatan pengemudi mengantuk yang diimplementasikan dalam program di atas, EAR dikomputasi dalam waktu nyata dari frame video langsung yang ditangkap oleh raspberry pi camera v2 dan diproses menggunakan. Sistem mengekstrak koordinat mata, menghitung EAR untuk kedua mata, dan rata-rata nilainya untuk menentukan penutupan mata secara keseluruhan. Jika nilai EAR turun di bawah ambang batas yang telah ditentukan pada penelitian ini penulis menggunakan amabneg batas 0,22 untuk jangka waktu yang lama, sistem akan mengidentifikasi pengemudi mengantuk dan mengaktifkan peringatan audio dan visual [14].



Gambar 3. Diagram Alir

Ditujukan pada Gambar 3. Merupakan diagram alir menguraikan sistem pendeteksi pengemudi yang mengantuk yang diimplementasikan pada Raspberry Pi 4 Model B, mengintegrasikan pelacakan mata berbasis visi komputer dan peringatan waktu nyata. Prosesnya dimulai dengan menginisialisasi layar OLED 128x64, yang berfungsi sebagai output visual untuk status dan peringatan sistem. Modul Picamera2 menangkap video real-time, yang kemudian diproses menggunakan model Shape Predictor 68 Facial Landmark dari pustaka dlib untuk mendeteksi fitur wajah, khususnya mata. Jika tidak ada mata yang terdeteksi, sistem akan mengulang kembali untuk terus memproses frame baru. Setelah deteksi mata berhasil, Eye Aspect Ratio (EAR) dihitung berdasarkan penanda mata. Langkah ini menentukan persentase keterbukaan mata, yang merupakan metrik penting untuk mendeteksi rasa kantuk [15].

Jika nilai EAR yang dihitung turun di bawah 0,22 sistem akan melanjutkan pemantauan untuk penutupan mata yang berkelanjutan. Jika EAR tetap di bawah 0,22 selama setidaknya 2 detik, yang mengindikasikan potensi kantuk, sistem akan memicu alarm audio yang memperingatkan pengemudi untuk bangun. Secara bersamaan, pesan peringatan visual (“Alert”) ditampilkan pada layar OLED. Jika EAR kembali di atas ambang batas sebelum mencapai tanda 2 detik, sistem akan mengatur ulang dan melanjutkan pemantauan. Pendekatan ini memastikan bahwa kedipan sesaat tidak memicu alarm palsu, sehingga pendeteksian menjadi lebih efisien.

III. HASIL DAN PEMBAHASAN

A. Pustaka Program

```
from scipy.spatial import distance
from imutils import face_utils
from pygame import mixer
import imutils
import dlib
import cv2
import time
from picamera2 import Picamera2
from luma.core.interface.serial import i2c
from luma.oled.device import sh1106
from luma.core.render import canvas
from PIL import ImageFont

```

Gambar 4. Pustaka Program Python

Pustaka Python yang digunakan dalam program ini memiliki peran yang berbeda dalam visi komputer, pemrosesan audio, dan antarmuka perangkat keras. Modul spasial SciPy menyediakan fungsi distance.euclidean, yang sangat penting untuk menghitung jarak Euclidean antara tengara wajah. Imutils adalah pustaka kemudahan yang menyederhanakan operasi OpenCV yang umum seperti mengubah ukuran dan ekstraksi tengara wajah. Dlib adalah pustaka pembelajaran mesin yang kuat yang digunakan untuk deteksi wajah dan prediksi tengara. OpenCV (cv2) adalah pustaka utama untuk pemrosesan gambar, mengubah bingkai menjadi skala abu-abu dan menggambar hamparan visual seperti lambung cembung di sekitar mata yang terdeteksi. Waktu digunakan untuk mengukur durasi penutupan mata untuk memicu peringatan setelah ambang batas yang ditetapkan.

Untuk integrasi perangkat keras, pustaka Picamera2 memungkinkan pengambilan gambar yang efisien dari kamera Raspberry Pi, mendukung pengambilan bingkai langsung untuk pemrosesan waktu nyata. Modul mixer Pygame digunakan untuk pemutaran audio, memungkinkan audio berbunyi ketika kantuk terdeteksi. Luma.core dan modul terkait, termasuk i2c dan sh1106, menyediakan antarmuka untuk mengendalikan layar OLED melalui I2C, memungkinkan pembaruan status waktu nyata ditampilkan di layar. ImageFont dari PIL memungkinkan program untuk membuat font khusus pada layar OLED, meningkatkan visibilitas. Bersama-sama, pustaka ini membentuk sistem terintegrasi untuk deteksi kantuk pengemudi secara real-time dengan menggabungkan visi komputer, peringatan audio, dan interaksi perangkat keras.

B. Kalkulasi EAR (*Eye Aspec Ratio*)

```
def eye_aspect_ratio(eye):
    A = distance.euclidean(eye[1], eye[5])
    B = distance.euclidean(eye[2], eye[4])
    C = distance.euclidean(eye[0], eye[3])
    ear = (A + B) / (2.0 * C)
    return ear

```

Gambar 5. Program Python Kalkulasi Nilai EAR

Ditujukan pada Gambar 5. A dan B mewakili jarak vertikal antara tengara kelopak mata atas dan bawah, sedangkan C mewakili jarak horizontal antara sudut mata. Jarak-jarak ini diperoleh dengan menggunakan fungsi distance.euclidean dari SciPy, yang menghitung jarak garis lurus antara dua titik dalam bidang 2D. EAR kemudian dihitung dengan menggunakan rumus (A + B) / (2.0 * C). Rumus ini rata-rata dua jarak vertikal dan menormalkannya dengan jarak horizontal, membuat skala metrik tidak berubah-ubah, yang berarti tetap konsisten terlepas dari ukuran wajah atau jarak dari kamera. Nilai EAR yang lebih tinggi mengindikasikan mata yang terbuka, sedangkan nilai yang lebih rendah menunjukkan bahwa mata sedang menutup atau terpejam. Karena EAR tetap relatif konstan saat mata terbuka dan menurun secara signifikan saat mata tertutup, EAR berfungsi sebagai ukuran berbasis ambang batas yang dapat diandalkan untuk mendeteksi rasa kantuk. Dengan terus memantau EAR, sistem dapat menentukan kapan mata pengemudi tertutup untuk waktu yang lama, sehingga memicu peringatan jika diperlukan.

C. I2C OLED 128x64

```
#   *Modul Oled*
# Inisialisasi koneksi I2C
serial = i2c(port=1, address=0x3C)
# Inisialisasi perangkat OLED
device = sh1106(serial)
# Menggunakan font kustom
font_path = "/usr/share/fonts/truetype/dejavu/DejaVuSans-Bold.ttf"
font = ImageFont.truetype(font_path, 24)
# Buat konten untuk ditampilkan di OLED
with canvas(device) as draw:
    draw.text((0, 0), "Hello", font=font, fill="white")
```

Gambar 6. Program Python Menampilkan OLED

Inisialisasi dan mengonfigurasi layar OLED menggunakan pustaka Luma.OLED, yang menyediakan antarmuka untuk mengontrol layar OLED kecil berbasis I2C model SH1106. Langkah pertama membuat koneksi I2C (Inter-Integrated Circuit) menggunakan serial = I2C (port=1, address=0x3C), di mana port =1 menentukan bus I2C pada Raspberry Pi, dan address=0x3C mewakili alamat I2C default untuk banyak modul OLED. Selanjutnya, device = sh1106(serial) menginisialisasi driver OLED SH1106, yang memungkinkan sistem untuk mengirim data grafis atau tekstual ke layar. Driver ini menangani komunikasi tingkat rendah yang diperlukan untuk merender konten pada panel OLED.

Untuk menampilkan teks, kode memuat font TrueType khusus menggunakan Python Imaging Library (PIL). File font terletak di "/usr/share/fonts/truetype/dejavu/DejaVuSans-Bold.ttf", dan dimuat dengan ukuran 24 piksel menggunakan ImageFont.truetype(). Context manager canvas(device) kemudian digunakan untuk membuat permukaan gambar sementara, di mana teks "Hello" dirender pada posisi (0,0) dalam warna putih. Hal ini memastikan bahwa ketika layar OLED diperbarui, teks akan terlihat. Penggunaan font khusus meningkatkan keterbacaan dan estetika, sehingga lebih mudah menyampaikan peringatan atau pesan status. Integrasi OLED ini sangat penting dalam sistem pendeteksi kantuk, karena memberikan output visual waktu nyata.

D. Raspberry Pi Camera V2

```
# Inisialisasi kamera
picam2 = Picamera2()
picam2.configure(picam2.create_preview_configuration(main={"format": "RGB888"}))
picam2.start()
```

Gambar 7. Program Python Menangkap Gambar Raspberry Pi Camera V2

Kode ini menginisialisasi dan mengkonfigurasi modul Picamera2, yang merupakan pustaka Python modern untuk berinteraksi dengan Kamera Raspberry Pi. picam2 = Picamera2() membuat instance dari kelas Picamera2, yang memungkinkan program untuk mengambil gambar dan bingkai video. Picam2.configure picam2.create_preview_configuration(main={"format":"RGB888"}) mengatur mode pratinjau kamera, menetapkan bahwa format gambar utama harus RGB888, yang berarti setiap piksel diwakili oleh 8 bit per saluran warna (Merah, Hijau, dan Biru), memberikan gambar berwarna. Format ini banyak digunakan dalam pemrosesan gambar karena kompatibel dengan OpenCV, yang mengharapkan gambar dalam format RGB atau BGR. Terakhir, picam2.start() mengaktifkan kamera, secara terus menerus mengambil frame yang dapat diproses dalam waktu nyata.

E. Program Facial Landmark

```
# Inisialisasi variabel
thresh = 0.22
frame_check = 20
detect = dlib.get_frontal_face_detector()
predict = dlib.shape_predictor("//home/pi/Desktop/Smadav_EAR//shape_predictor_68_face_landmarks.dat")
(lStart, lEnd) = face_utils.FACIAL_LANDMARKS_68_IDXS["left_eye"]
(rStart, rEnd) = face_utils.FACIAL_LANDMARKS_68_IDXS["right_eye"]
```

Gambar 8. Program Python Facial Landmark

Kode ini menginisialisasi variabel kunci untuk sistem pendeteksi kantuk. Variabel thresh diatur ke 0.22, mewakili ambang batas Eye Aspect Ratio (EAR) di bawahnya, di mana sistem menganggap mata pengemudi dalam keadaan tertutup. Variabel frame_check = 20 dalam beberapa implementasi, variabel ini membantu melacak jumlah frame yang berurutan di mana EAR tetap berada di bawah ambang batas sebelum memicu peringatan. Fungsi dlib.get_frontal_face_detector() menginisialisasi detektor wajah, yang bertanggung jawab untuk mendeteksi wajah dalam umpan kamera. Sementara itu, dlib.shape_predictor memuat detektor tengara wajah 68 titik yang sudah dilatih sebelumnya, yang sangat penting untuk menemukan mata pada wajah yang terdeteksi. Indeks untuk landmark mata kiri dan kanan diambil dari face_utils.FACIAL_LANDMARKS_68_IDXS, untuk memastikan bahwa wilayah mata yang benar diekstraksi untuk perhitungan EAR.

F. Perulangan Deteksi Mata

```
54  # Main Loop
55  while True:
56      frame = picam2.capture_array()
57      frame = imutils.resize(frame, width=450)
58      gray = cv2.cvtColor(frame, cv2.COLOR_BGR2GRAY)
59      subjects = detect(gray, 0)
60          # Eye Aspect Ratio
61      for subject in subjects:
62          shape = predict(gray, subject)
63          shape = face_utils.shape_to_np(shape)
64          leftEye = shape[lStart:lEnd]
65          rightEye = shape[rStart:rEnd]
66          leftEAR = eye_aspect_ratio(leftEye)
67          rightEAR = eye_aspect_ratio(rightEye)
68          ear = (leftEAR + rightEAR) / 2.0
69          # Tambahkan setelah penghitungan nilai 'ear'
70          cv2.putText(frame, f"EAR: {ear:.2f}", (300, 30),
71              cv2.FONT_HERSHEY_SIMPLEX, 0.6, (255, 255, 255), 2)
72          print (f"EAR: {ear:.2f}")
```

Gambar 9. Operasi Perulangan Raspberry Pi Mendeteksi Kantuk

Kode ini beroperasi dalam perulangan while yang tak terbatas, secara terus menerus menangkap dan memproses frame dari Kamera Raspberry Pi untuk deteksi kantuk secara real-time. Fungsi picam2.capture_array() mengambil satu frame sebagai larik NumPy, yang kemudian diubah ukurannya menjadi lebar tetap 450 piksel menggunakan imutils.resize(frame, width=450). Pengubahan ukuran membantu menstandarisasi dimensi input, mengurangi beban komputasi sambil mempertahankan detail yang cukup untuk analisis wajah. Bingkai kemudian dikonversi ke skala abu-abu menggunakan cv2.cvtColor(frame, cv2.COLOR_BGR2GRAY), karena gambar skala abu-abu menyederhanakan pemrosesan dan meningkatkan efisiensi model pendeteksian wajah Dlib. Fungsi detect(gray, 0) dari detektor wajah berbasis HOG Dlib memindai gambar untuk mencari wajah manusia, mengembalikan kotak pembatas untuk setiap wajah yang terdeteksi. Parameter 0 menentukan faktor peningkatan default, menjaga kinerja pendeteksian tetap dioptimalkan.

Untuk setiap wajah yang terdeteksi (subjek), prediktor tengara wajah 68 titik (predict(gray, subject)) diterapkan untuk mengekstrak fitur wajah utama, yang kemudian dikonversi ke dalam larik NumPy menggunakan face_utils.shape_to_np(shape). Penanda mata diambil menggunakan indeks yang telah ditentukan sebelumnya, dengan leftEye = shape[lStart:lEnd] dan rightEye = shape[rStart:rEnd]. Rasio Aspek Mata (EAR) untuk kedua mata dihitung secara terpisah menggunakan fungsi eye_aspect_ratio (), dan rata-ratanya disimpan dalam variabel ear. Rata-rata ini memperhitungkan asimetri kecil dalam pendeteksian mata dan memastikan pembacaan yang lebih stabil. EAR berfungsi sebagai indikator penting keterbukaan mata, di mana nilai yang lebih tinggi menandakan mata terbuka, dan nilai yang lebih rendah menunjukkan mata tertutup.

G. Outputan Apabila EAR dibawah Tresh

```
        #Output
        if ear < thresh:
            # Cek apakah alert_start_time belum diatur
            if alert_start_time is None:
                alert_start_time = time.time()
            # Hitung durasi mata tertutup
            if time.time() - alert_start_time >= alert_duration:
                cv2.putText(frame, "***************ALERT!***************", (10, 30),
                            cv2.FONT_HERSHEY_SIMPLEX, 0.7, (0, 0, 255), 2)
                mixer.music.play()
                print("Alert")
                with canvas(device) as draw:
                    draw.text((0, 0), "Alert", font=font, fill="white")

        else:
            # Reset alert jika mata terbuka
            alert_start_time = None
            with canvas(device) as draw:
                draw.text((0, 0), "HELLO", font=font, fill="white")
```

Gambar 10. Program Python Output Peringatan Audio Speaker dan OLED

Bagian perulangan utama yang bertanggung jawab untuk memicu peringatan ketika sistem mendeteksi bahwa mata pengemudi terpejam dalam waktu yang lama, yang mengindikasikan potensi kantuk. Kondisi pertama memeriksa apakah Rasio Aspek Mata (EAR) telah turun di bawah ambang batas yang telah ditentukan (thresh). Jika ear < thresh, sistem mencurigai bahwa mata tertutup, dan tindakan lebih lanjut diperlukan. Untuk memastikan sistem tidak memicu peringatan untuk kedipan singkat, kode melacak durasi waktu penutupan mata. Jika alert_start_time belum diatur (yaitu, mata sebelumnya terbuka), kode akan mencatat waktu saat ini menggunakan time.time(), memulai pengatur waktu untuk penutupan mata. Pencatat waktu ini akan digunakan untuk mengukur berapa lama mata tetap tertutup.

Setelah pengatur waktu dimulai, sistem akan terus memeriksa apakah durasi penutupan mata telah melebihi waktu ambang batas (alert_duration = 2 detik dalam kasus ini). Jika kondisi time.time() - alert_start_time >= alert_duration terpenuhi, hal ini mengindikasikan bahwa mata telah tertutup selama 2 detik atau lebih. Pada titik ini, peringatan akan dipicu. Peringatan ada dua: peringatan visual muncul di layar dengan teks “ALERT!” yang ditampilkan pada frame menggunakan cv2.putText(), dan peringatan audio diputar melalui speaker menggunakan mixer.music.play(). Warna teks (0, 0, 255) sesuai dengan warna merah, memastikan visibilitas yang tinggi. Print(“Alert”) memberikan umpan balik di konsol, yang mengonfirmasi bahwa peringatan telah dipicu.

Layar OLED juga diperbarui untuk menampilkan pesan peringatan, draw.text((0, 0), “Alert”, font=font, fill=“white”), menggunakan pustaka Luma.OLED, yang memberikan umpan balik waktu nyata pada perangkat. Jika rasio aspek mata berada di atas ambang batas, yang mengindikasikan mata terbuka, sistem akan mengatur ulang alert_start_time ke None, yang secara efektif menghapus status peringatan sebelumnya. Layar OLED diperbarui lagi, kali ini menampilkan “HELLO,” yang menunjukkan bahwa sistem dalam keadaan normal. Dengan mengatur ulang kondisi peringatan saat mata terbuka, sistem memastikan bahwa sistem hanya memicu peringatan setelah mata tetap tertutup dalam waktu yang ditentukan.

H. Hasil Outputan Program

Gambar 11. Kondisi Mata Terbuka (EAR = 0.23)

Gambar 12. Kondisi Mata Tertutup (EAR = 0.19)

Gambar 13. OLED Menampilkan "Alert"

Gambar 11. dan Gambar 12. merupakan tampilan hasil video real-time, terlihat bahwa sistem dapat mendeteksi jarak vertikal mata dan jarak horizontal mata. Nilai EAR untuk kedua mata dihitung berdasarkan persamaan (1), kemudian nilai EAR dari kedua mata ditampilkan dalam frame. Berdasarkan gambar 12. selama mata tertutup, nilai EAR mendekati nilai 0, ketika mata terbuka nilai EAR dapat dikatakan lebih besar dari nilai 0. Pada gambar. terlihat bahwa nilai EAR yang muncul pada frame pada gambar sebesar 0.28 pada kondisi mata terbuka (tidak mengantuk) sehingga sistem pada alat tidak akan memberikan peringatan. Gambar 12. menunjukkan nilai EAR sebesar 0.19 pada kondisi mata tertutup (mengantuk) pada frame akan merespon untuk menampilkan kalimat peringatan. Sedangkan pada Gambar 12. merupakan respon dari I2C OLED 128x64 jika nilai EAR dibawah ambang batas 0.22 akan menampilkan kalimat “Alert”, dan juga akan memberikan peringatan suara melalui audio speaker.

Tabel 1. Waktu Yang Dibutuhkan Audio Sepaker OLED Untuk Merespon

No	Waktu Yang Dibutuhkan Audio Sepaker Untuk Merespon	Waktu Yang Dibutuhkan OLED Untuk Merespon
1	0.81	0.87
2	0.81	0.89
3	0.83	0.85
4	0.70	0.79
5	0.83	0.89
6	0.73	0.78
7	0.71	0.75
8	0.74	0.78
9	0.73	0.78
10	0.70	0.75
Rata-rata	**0.75**	**0.81**

Intervensi untuk pengemudi yang mengantuk adalah dengan menggunakan speaker audio Tabel 1. menunjukkan waktu yang dibutuhkan oleh speaker audio untuk merespon. satuan waktu yang digunakan dalam pengujian adalah detik. Nilai waktu didapatkan ketika subjek memejamkan mata, kemudian fitur stopwatch pada smartphone digunakan untuk mengukur nilai waktu audio speaker menyala. Hasil dari pengujian tersebut didapatkan rata-rata waktu respon sebesar 0.75 untuk audio speaker on. dengan waktu respon tercepat sebesar 0.81. Dengan waktu respon kurang dari 1 detik, maka dapat dikatakan bahwa audio speaker dapat berfungsi untuk mengintervensi pengemudi yang mengantuk.

VII. SIMPULAN

Program ini ditujukan untuk sistem deteksi pada pengemudi berbasiskan Raspberry Pi 4 model B sebagai platform komputasi papan tunggal. Melalui cara visi komputer Raspberry Pi 4 model memproses pengolahan gambar yang didapatkan pada Raspberry pi camera V2, jika komputer mendeteksi kantuk komputer raspberry pi mengaktifkan outputan berupa suara melalui audio speaker dan peringatan secara visual melalui I2C OLED 128x64. Cara mengukur tingkat kantuk pengemudi digunakan model Shape Predictor_68_Facial Landmark untuk mendeteksi bagian bagian wajah teriutama mata. EAR (*Eye Aspect Ratio*) digunakan untuk parameter mengukur tingkat kantuk malalui nilai rasio terbuka tertutupnya kelopak mata. Eye Aspect Ratio pada kondisi ambang batas maksimum dalam waktu 2 detik, Smart Alarm Driver Assistance melakukan interupsi berupa peringatan suara dan visual yang akan membantu pengemudi untuk tetap sadar. Pengujian data mendapatkan rata-rata respon output peringatan audio speaker sebesar 0.82 detik dan rata-rata OLED 128x64 sebesar 0.90 detik. Penelitian ini dapat membuktikan bahwa Raspberry Pi 4 model B dapat digunakan untuk deteksi dan peringatan kantuk. Kedepannya, sistem dapat dikembangkan dari segi perangkat lunak lebih kompleks sehingga sistem dapat diimpelementasikan dengan kondisi yang bervariasi.

REFERENSI

[1] M. A. Kamran, M. M. N. Mannan, and M. Y. Jeong, "Drowsiness, Fatigue and Poor Sleep's Causes and Detection: A Comprehensive Study," *IEEE Access*, vol. 7, pp. 167172–167186, 2019, doi: 10.1109/ACCESS.2019.2951028.

[2] M. Chrisnatalia, D. K. Putri, and S. B. B. Liwun, "Perilaku Mengemudi Berisiko," *Humanit. (Jurnal Psikologi)*, vol. 7, no. 3, pp. 305–318, Dec. 2023, doi: 10.28932/humanitas.v7i3.7550.

[3] M. R. Zielinski, D. M. Systrom, and N. R. Rose, "Fatigue, Sleep, and Autoimmune and Related Disorders," *Front. Immunol.*, vol. 10, Aug. 2019, doi: 10.3389/fimmu.2019.01827.

[4] J. Jose, J. S. Vimali, P. Ajitha, S. Gowri, A. Sivasangari, and B. Jinila, "Drowsiness Detection System for Drivers Using Image Processing Technique," in *2021 5th International Conference on Trends in Electronics and Informatics (ICOEI)*, 2021, pp. 1527–1530. doi: 10.1109/ICOEI51242.2021.9452864.

[5] H. Lee, J. Lee, and M. Shin, "Using Wearable ECG/PPG Sensors for Driver Drowsiness Detection Based on Distinguishable Pattern of Recurrence Plots," *Electronics*, vol. 8, no. 2. 2019. doi: 10.3390/electronics8020192.

[6] F. Rundo *et al.*, "An Innovative Deep Learning Algorithm for Drowsiness Detection from EEG Signal," *Computation*, vol. 7, no. 1. 2019. doi: 10.3390/computation7010013.

[7] S. Laki, R. Stoyanov, D. Kis, R. Soulé, P. Vörös, and N. Zilberman, "P4Pi," *ACM SIGCOMM Comput. Commun. Rev.*, vol. 51, no. 3, pp. 17–21, Jul. 2021, doi: 10.1145/3477482.3477486.

[8] S. Mehta, S. Dadhich, S. Gumber, and A. J. Bhatt, "Real-Time Driver Drowsiness Detection System Using Eye Aspect Ratio and Eye Closure Ratio Fatigue Detection Non-Intrusive Methods Driver monitoring system," *Proceeding Int. Conf. Sustain. Comput. Sci. Technol. Manag.*, pp. 1333–1339, 2019, [Online]. Available: https://ssrn.com/abstract=3356401

[9] P. Fard Moshiri, R. Shahbazian, M. Nabati, and S. A. Ghorashi, "A CSI-Based Human Activity Recognition Using Deep Learning," *Sensors*, vol. 21, no. 21. 2021. doi: 10.3390/s21217225.

[10] B. Balon and M. Simić, "Using Raspberry Pi Computers in Education," in *2019 42nd International Convention on Information and Communication Technology, Electronics and Microelectronics (MIPRO)*, 2019, pp. 671–676. doi: 10.23919/MIPRO.2019.8756967.

[11] P. Wang, Q. Zhou, Y. Zhao, S. Zhao, and J. Ma, "A Robust Facial Landmark Detection in Uncontrolled Natural Condition," *J. Phys. Conf. Ser.*, vol. 1693, no. 1, 2020, doi: 10.1088/1742-6596/1693/1/012202.

[12] B. Vaishali and S. J. J. Thangaraj, "Design of facial feature extraction for age classification in criminal identification system using DLib library over OpenCV library," 2024, p. 020147. doi: 10.1063/5.0197433.

[13] C. Dewi, R. C. Chen, X. Jiang, and H. Yu, "Adjusting eye aspect ratio for strong eye blink detection based on facial landmarks," *PeerJ Comput. Sci.*, vol. 8, no. 2020, pp. 1–21, 2022, doi: 10.7717/peerj-cs.943.

[14] C. Dewi, R. C. Chen, C. W. Chang, S. H. Wu, X. Jiang, and H. Yu, "Eye Aspect Ratio for Real-Time Drowsiness Detection to Improve Driver Safety," *Electron.*, vol. 11, no. 19, 2022, doi: 10.3390/electronics11193183.

[15] B. Borah and S. Mukherjee, "D-Alarm: An Efficient Driver Drowsiness Detection and Alarming System," in *2023 6th International Conference on Advances in Science and Technology (ICAST)*, 2023, pp. 203–208. doi: 10.1109/ICAST59062.2023.10454968.

Conflict of Interest Statement:

The author declares that the research was conducted in the absence of any commercial or financial relationships that could be construed as a potential conflict of interest.