Universitas
Muhammadiyah
Sidoarjo

Seminar Proposal Skripsi

Program Sistem Peringatan Pengemudi Mengantuk Berbasis Raspberry Pi 4 Model B Dengan Output Audio Speaker Dan I2C OLED 128x64

OLEH :

- Achmad Arif Dwi Prasetyo (211020100011)
- Dosen Pembimbing Indah Sulistiyowati, ST., MT.

BAB 1. Pendahuluan

- Latar Belakang

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. 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. (M. Chrisnatalia, D. K. Putri, and S. B. B. Liwun) “Perilaku Mengemudi Berisiko”.

BAB 1. Pendahuluan

- Pengolahan Citra

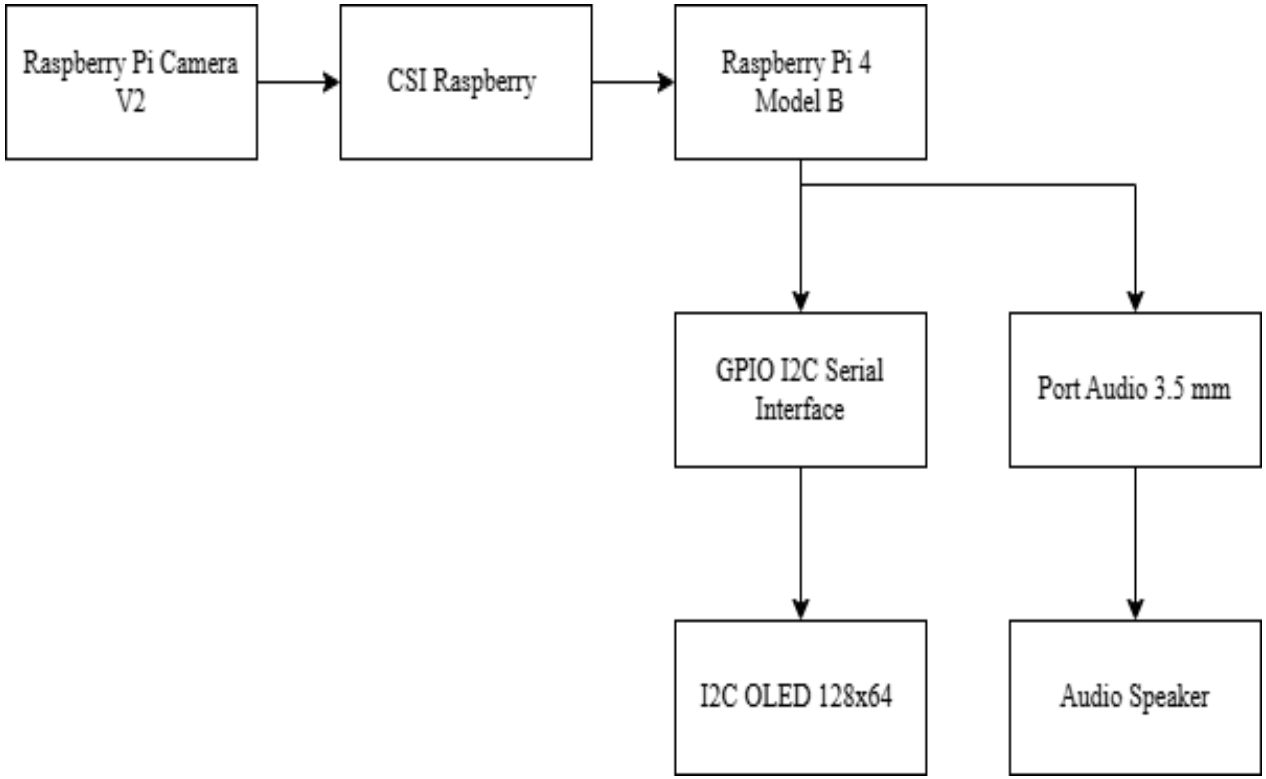
Deteksi kantuk merupakan area penelitian yang penting dalam keselamatan di jalan raya, yang bertujuan untuk mencegah kecelakaan yang disebabkan oleh kelelahan pengemudi melalui sistem pemantauan waktu nyata. Pemrosesan gambar berbasis visi komputer telah muncul sebagai metode yang efektif untuk mendeteksi rasa kantuk dengan menganalisis fitur wajah, terutama gerakan mata dan durasi kedipan.

- Raspberry Pi 4 Model B

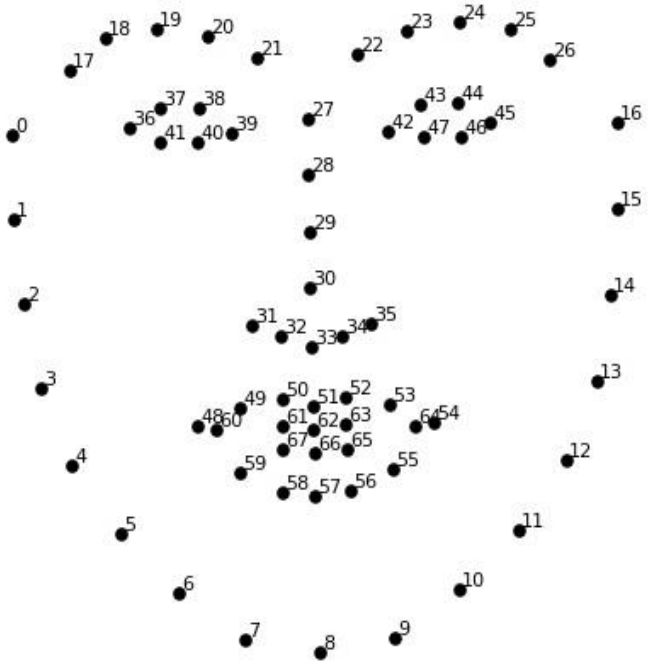
Raspberry Pi 4 Model B telah menjadi perangkat yang mendukung untuk mengimplementasikan sistem pendeteksi kantuk secara real-time dengan menggunakan pemrosesan gambar berbasis visi komputer. Dengan prosesor quadcore, GPU yang telah tertanam dalam perangkat, dan dukungan untuk modul kamera beresolusi tinggi, Raspberry Pi memungkinkan eksekusi yang efisien dari algoritma pendeteksian tengara wajah untuk melacak gerakan mata dan pemrosesan untuk mendeteksi kantuk.

BAB 2. Metodologi

- **Desain Perangkat Keras**



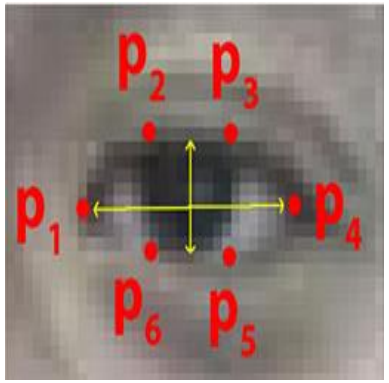
BAB 2. Metodologi

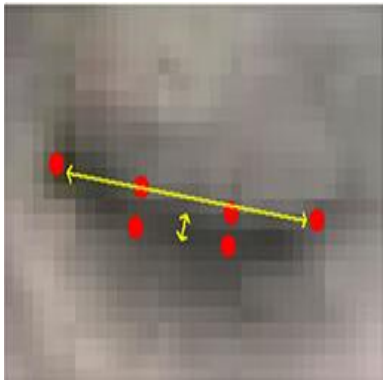


Shape Predictor 68 Facial Landmark algoritme prediktor bentuk dlib, teknik yang digunakan dalam visi komputer untuk mendeteksi dan melokalisasi fitur fitur wajah utama.

- **Shape Predictor_68_Facial Landmark Detection**

BAB 2. Metodologi





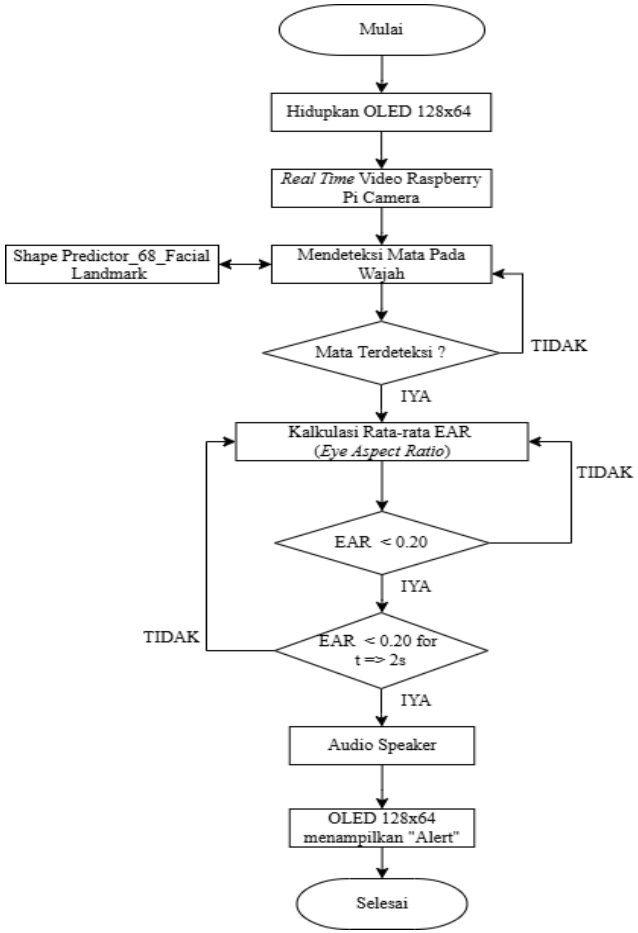
EAR
(Eye Aspect Ratio)

Metode untuk mengukur keterbukaan dan penutupan mata melalui analisis geometris landmark wajah. Dihitung menggunakan jarak antara titik utama di sekitar setiap mata, yang diekstraksi menggunakan model Shape Predictor 68 Facial Landmark.

$$EAR = \frac{||P2-P6|| + ||P3-P5||}{2||P1-P4||}$$

BAB 2. Metodologi

- **Flowchart**



BAB 2. Metodologi

- **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

```

BAB 3. Hasil

- **Fungsi Kalkulasi Nilai**

```
18    def eye_aspect_ratio(eye):
19        A = distance.euclidean(eye[1], eye[5])
20        B = distance.euclidean(eye[2], eye[4])
21        C = distance.euclidean(eye[0], eye[3])
22        ear = (A + B) / (2.0 * C)
23        return ear
24
```

- **Detektor 68 Titik Wajah**

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

BAB 3. Hasil

- **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")

```

- **Raspberry Pi Camera V2**

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

```

BAB 3. Hasil

- **RGB ke Abu-Abu dan Kode Deteksi Wajah Beserta 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}")
```

- RGB ke Gray menyederhanakan pemrosesan dan meningkatkan efisiensi model pendeteksian wajah
- Numpy Mengestrak model Shape Predictor_68_Facial Landmark Detection untuk pendeteksian wajah dan juga mata.

BAB 3. Hasil

- **Convex Hull**

```
74          # Convex Hull
75          leftEyeHull = cv2.convexHull(leftEye)
76          rightEyeHull = cv2.convexHull(rightEye)
77          cv2.drawContours(frame, [leftEyeHull], -1, (0, 255, 0), 1)
78          cv2.drawContours(frame, [rightEyeHull], -1, (0, 255, 0), 1)
79
```

Convex Hull digunakan untuk mendefinisikan dan memvisualisasikan bentuk mata yang terdeteksi, yang sangat penting untuk pelacakan tengara wajah dan deteksi kantuk.

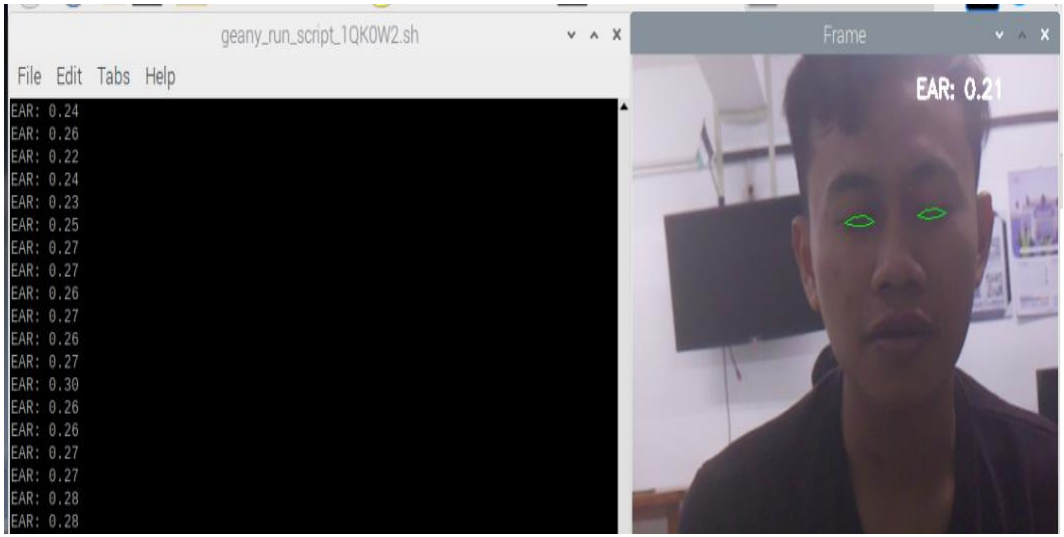
BAB 3. Hasil

- **Kode Output Peringatan**

```
80          #Output
81          if ear < thresh:
82              # Cek apakah alert_start_time belum diatur
83              if alert_start_time is None:
84                  alert_start_time = time.time()
85              # Hitung durasi mata tertutup
86              if time.time() - alert_start_time >= alert_duration:
87                  cv2.putText(frame, "****************ALERT!****************", (10, 30),
88                              cv2.FONT_HERSHEY_SIMPLEX, 0.7, (0, 0, 255), 2)
89                  mixer.music.play()
90                  print("Alert")
91                  with canvas(device) as draw:
92                      draw.text((0, 0), "Alert", font=font, fill="white")
93
94          else:
95              # Reset alert jika mata terbuka
96              alert_start_time = None
97              with canvas(device) as draw:
98                  draw.text((0, 0), "HELLO", font=font, fill="white")
99
```

BAB 3. Hasil

- **Output Program**



No	Waktu Yang Dibutuhkan Audio Sepaker Untuk Merespon	Waktu Yang Dibutuhkan Audio Sepaker 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

Universitas
Muhammadiyah
Sidoarjo

TERIMA KASIH