IGNITE : Journal Islamic Global Network for Information Technology and Entrepreneurship
Volume. 2, Nomor.4 Tahun 2024
e-ISSN :3025-2857; dan, p-ISSN :3025-2849, Hal 15-27
DOI: https://doi.org/10.59841/ignite.v2i4.2011
Available online at : https://jurnal.stikes-ibnusina.ac.id/index.php/IGNITE

Facial Landmarks and Face Detection in Python With OpenCv

Supiyandi[1] , Icha Miranti Irzan[2] , Risma Hidayati[3] , Rosa Prahasti[4] , Natria Selina[5]
[1-5] Fakultas Sains dan Teknologi, Ilmu Komputer, Universitas Islam Negeri Sumatera Utara, Indonesia
E-mail: supiyandi.mkom@gmail.com[1] , icameranti2017@gmail.com[2] , rismaaja201@gmail.com[3] , rosaprahasti423@gmail.com[4] , natriaselina06@gmail.com[5]

Abstract. *Face detection and facial landmarks are an important technique in the field of computer vision with a wide range of potential applications, including expression recognition, security systems, and human-computer interaction. This study explores the implementation of facial landmarks detection using Python and OpenCV, focusing on the use of the Haar Cascade algorithm for face detection and the Local Binary Features (LBF) model for the identification of landmarks. The proposed method implements real-time detection via webcam, capable of recognizing 68 important points on the human face. The results show that the approach using OpenCV and LBF models has good accuracy in detecting and tracking facial features in different lighting conditions and viewing angles. This research contributes to the development of efficient and reliable facial detection methods, with wide application potential in the fields of computer vision, security, and behavioral analysis.*

Keywords*: Facial Landmarks, Computer Vision, OpenCV , Python, Face Detection*

Abstrak. Deteksi wajah dan facial landmarks merupakan teknik penting dalam bidang computer vision dengan berbagai aplikasi potensial, termasuk pengenalan ekspresi, sistem keamanan, dan interaksi manusia-komputer. Penelitian ini mengeksplorasi implementasi deteksi facial landmarks menggunakan Python dan OpenCV, dengan fokus pada penggunaan algoritma Haar Cascade untuk deteksi wajah dan model Local Binary Features (LBF) untuk identifikasi titik-titik landmark. Metode yang diusulkan mengimplementasikan deteksi real-time melalui webcam, mampu mengenali 68 titik penting pada wajah manusia. Hasil menunjukkan bahwa pendekatan menggunakan OpenCV dan LBF model memiliki akurasi yang baik dalam mendeteksi dan melacak fitur-fitur wajah dalam kondisi pencahayaan dan sudut pandang yang berbeda. Penelitian ini memberikan kontribusi dalam pengembangan metode deteksi wajah yang efisien dan dapat diandalkan, dengan potensi aplikasi yang luas dalam bidang computer vision, keamanan, dan analisis perilaku.

Kata kunci: Landmark Wajah, Visi Komputer, OpenCV, Python, Deteksi Wajah

1. PENDAHULUAN

Perkembangan teknologi computer vision telah mengalami kemajuan signifikan dalam beberapa dekade terakhir, terutama dalam deteksi wajah dan analisis fitur wajah. Wajah manusia memiliki bentuk yang unik, dan proses identifikasi wajah sering dilakukan dengan mengenali pola-pola tertentu pada wajah tersebut. Salah satu teknik dasar dalam bidang visi komputer adalah deteksi landmark wajah, yaitu proses menemukan area penting pada wajah, seperti sudut mata, hidung, mulut, dan fitur wajah lainnya (Wibowo, Karima, Wiktasari, Yobioktabera, & Fahriah, 2020). Deteksi ini menjadi penting dalam berbagai aplikasi seperti penyelarasan wajah, deteksi emosi, analisis ekspresi wajah, augmented reality, dan pengenalan wajah, yang semuanya memerlukan kemampuan untuk mendeteksi dan melacak landmark wajah secara akurat.

Received: November 17, 2024; Revised: November 31, 2024; Accepted: Desember 03, 2024; Published : Desember 05, 2024;

Tantangan utama dalam deteksi facial landmarks meliputi variabilitas ekspresi wajah, kondisi pencahayaan yang berbeda, sudut pandang, dan kompleksitas struktur wajah manusia (Maulana, Khairunisa, & Mufidah, 2023). Algoritma modern seperti Haar Cascade dan Local Binary Features (LBF) telah memberikan terobosan dalam mengatasi sebagian besar tantangan ini, menawarkan metode yang lebih akurat dan efisien dalam mengenali dan melacak fitur wajah.

Penelitian ini bertujuan untuk mengeksplorasi implementasi teknik deteksi facial landmarks menggunakan pustaka OpenCV dan bahasa pemrograman Python. Fokus utama penelitian ini adalah mengembangkan metode deteksi yang dapat mengenali 68 titik landmark pada wajah manusia secara real-time dan beradaptasi dengan berbagai kondisi pencahayaan dan sudut pandang. Pendekatan yang diusulkan memanfaatkan kombinasi algoritma Haar Cascade untuk deteksi wajah dan model Local Binary Features (LBF) untuk identifikasi presisi titik-titik landmark, menawarkan solusi komputasional yang efisien dan dapat diandalkan.

Penelitian sebelumnya oleh (Sejati & Mardhiyyah, 2021) tentang "Deteksi Wajah Berbasis Facial Landmark Menggunakan OpenCV dan Dlib" menunjukkan bahwa metode facial landmark dapat melakukan deteksi pada area wajah dengan baik, menyesuaikan titik facial landmark yang telah ditentukan. Selain itu, penelitian oleh (Farokhah , 2021) menunjukkan bahwa Konvolusional Jaringan Saraf (CNN) dapat digunakan untuk memprediksi landmark wajah dengan akurat. Proposal sistem identifikasi landmark wajah real-time yang ditampilkan dalam penelitian ini menunjukkan hasil yang menggembirakan pada berbagai kumpulan data, memperlihatkan kekokohan dan keakuratannya dalam menemukan fitur wajah.

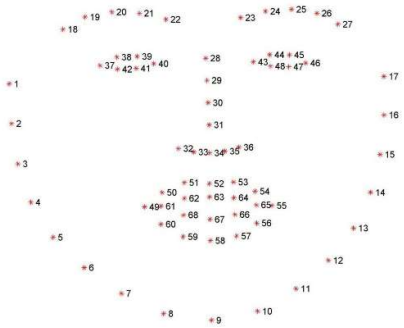
A. Facial Landmark

Facial landmark adalah serangkaian titik koordinat yang menandai lokasi fitur penting pada wajah manusia. Titik-titik ini digunakan untuk mendeteksi dan menganalisis bagian-bagian wajah seperti mata, alis, hidung, mulut, dan garis rahang (Pranoto, 2018). Dalam implementasi umum, facial landmark terdiri dari 68 titik yang sering digunakan dalam berbagai aplikasi, seperti yang tersedia dalam pustaka OpenCV. Setiap titik memiliki posisi spesifik yang merepresentasikan bagian tertentu dari wajah. Secara lebih rinci, 68 titik landmark tersebut terbagi menjadi beberapa kelompok area wajah, sebagai berikut:

- Rahang (jaw): titik 1-17
- Alis (eyebrows): titik 18-22 untuk alis kiri dan 23-27 untuk alis kanan
- Hidung (nose): titik 28-36

- Mata (eyes): titik 37-42 untuk mata kiri dan 43-48 untuk mata kanan
- Mulut (mouth): titik 49-68

Proses ini menjadi dasar penting dalam berbagai aplikasi computer vision seperti pengenalan ekspresi wajah, estimasi pose wajah, pengenalan identitas, dan augmented reality. Keakuratan dalam mendeteksi titik-titik landmark ini sangat penting karena akan mempengaruhi kualitas hasil analisis pada tahap selanjutnya.



Gambar 1 Facial Landmark

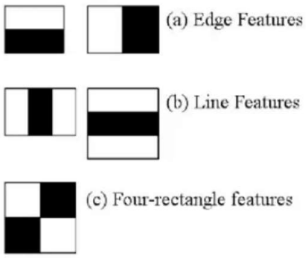
B. Face Detection

Deteksi wajah atau face detection adalah teknologi komputer yang digunakan dalam berbagai aplikasi yang mengidentifikasi wajah manusia dalam gambar digital. Deteksi wajah juga mengacu pada proses psikologis yang digunakan manusia untuk menemukan dan memperhatikan wajah dalam pemandangan visual (Hidayat, 2021). Face detection merupakan proses mendeteksi keberadaan wajah manusia dalam sebuah citra. Pada penelitian ini, metode yang digunakan adalah Haar Cascade Classifier, algoritma yang dikembangkan oleh Paul Viola dan Michael Jones pada tahun 2001.

1. Haar Cascade Classifier

Haar Cascade adalah algoritma pembelajaran mesin untuk mendeteksi objek yang diusulkan oleh Paul Viola dan Michael Jones pada tahun 2001. Algoritma ini adalah pendekatan berbasis pembelajaran mesin yang menggunakan fungsi cascade, dimana fungsi ini dilatih dari berbagai citra positif dan negatif. Citra positif merupakan citra yang memiliki objek yang akan dideteksi, sedangkan citra negatif merupakan citra yang tidak memiliki objek deteksi (Susim & Darujati, 2021). Sehingga fungsi ini dapat digunakan untuk mendeteksi objek pada citra yang lain. Saat ini, OpenCV sudah memberikan library untuk algoritma Haar Cascade serta sudah dikategorikan kedalam beberapa kategori seperti wajah, mata, dan sebagainya, tergantung pada gambar yang telah dilatih. Haar Cascade mengekstraksi fitur dari gambar

menggunakan sebuah "filter" mirip dengan konsep kernel konvolusional. Filter ini disebut fitur Haar dan terlihat seperti gambar 2.



Gambar 2 Fitur Haar

Filter ini akan memeriksa satu bagian pada satu waktu. Kemudian untuk tiap bagian, semua intensitas piksel pada bagian hitam dan putih akan dijumlahkan, lalu menghitung selisih dari tiap nilai yang dijumlahkan. Nilai tersebut merupakan nilai fitur yang diekstraksi. Karakteristik dari algoritma ini adalah adanya klasifikasi bertingkat. Klasifikasi ini terdiri dari beberapa tingkatan dimana tiap tingkatan mengeluarkan subcitra yang diyakini bukan wajah. Setelah wajah terdeteksi menggunakan Haar Cascade, proses pemetaan landmark wajah menjadi lebih terfokus pada area wajah yang relevan (Yulina, 2021).

C. Implementasi dengan OpenCV

OpenCV menyediakan pustaka yang telah dilatih untuk algoritma Haar Cascade, seperti haarcascade_frontalface_alt2.xml, yang digunakan untuk mendeteksi wajah dalam citra (Prakoso, Rasyid, Deannova, & Rahmawan, 2024). Proses implementasinya meliputi:

- Menggunakan citra berwarna sebagai input, lalu mengonversinya menjadi grayscale untuk mempercepat proses deteksi.
- Menerapkan Haar Cascade Classifier untuk mendeteksi wajah pada citra.

Haar Cascade dipilih karena memiliki akurasi tinggi dan proses deteksi yang cepat. Dengan menggunakan OpenCV, implementasi algoritma ini menjadi lebih mudah dan praktis. Selain OpenCV, pustaka lain yang digunakan dalam penelitian ini meliputi:

1. urllib: Untuk mengunduh model-model yang diperlukan dari URL (misalnya, Haar Cascade XML dan LBF Model).
2. NumPy: Untuk menangani data array dan mengonversi hasil deteksi wajah menjadi format yang sesuai untuk pemrosesan lebih lanjut dengan model LBF.

D. Pemetaan Landmark Menggunakan LBF Model

Penelitian ini juga mengimplementasikan LBF Model (Local Binary Features) untuk pemetaan titik landmark wajah. LBF adalah model yang digunakan untuk mendeteksi titik-titik penting pada wajah, seperti mata, hidung, dan mulut, dan dilatih menggunakan gradient boosting. Keunggulan LBF Model adalah efisiensi dan akurasi dalam pemetaan landmark, terutama dalam aplikasi real-time, karena lebih ringan dibandingkan dengan deep learning, menjadikannya pilihan yang baik untuk aplikasi yang membutuhkan respons cepat.

2. METODE PENELITIAN

1. Pengumpulan Data

Data dikumpulkan melalui proses pengambilan gambar wajah secara real-time menggunakan webcam. Pengambilan gambar dilakukan dengan posisi wajah menghadap ke depan (frontal view), untuk memastikan semua fitur wajah terlihat dengan jelas. Selama proses pengambilan gambar, kondisi pencahayaan dijaga agar tetap normal, sehingga kualitas deteksi wajah tetap terjaga.

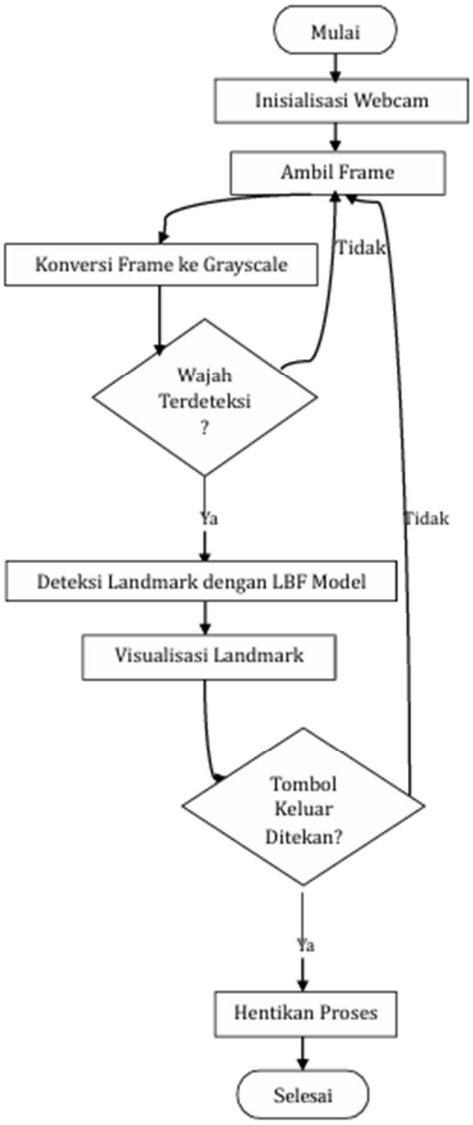
2. Perancangan Sistem

Tahap perancangan melibatkan penyusunan alur kerja sistem deteksi wajah dan facial landmarks. Titik-titik landmark didefinisikan berdasarkan 68 ciri-ciri khas wajah:

a. Wilayah rahang (jaw line): Titik 1 – 17 mendefinisikan bentuk dan struktur garis rahang dan menggambarkan kontur bawah wajah.
b. Wilayah alis (eyebrows): Alis kiri titik 18 – 22 dan alis kanan titik 23 – 27 mendeteksi kurva dan ketinggian alis, membantu analisis ekspreksi wajah.
c. Wilayah hidung (nose): Titik 28 – 36 (9 titik) menggambarkan struktur hidup termasuk ujung hidung, cuping dan tulang hidup.
d. Wilayah mata (eyes): Mata kiri titik (37 – 42) dan mata kanan titik (43 – 48) mendeteksi sudut dalam mata, sudut luar mata, kelopak mata atas dan bawah.
e. Wilayah mulut (mouth): Titik 49 – 68 (20 titik) untuk representasi mulut mencakup bibir atas dan bawah, sudut mulut dan garis tengah mulut.

Sistem dirancang untuk mengenali dan menandai titik-titik landmark secara real-time dengan menggunakan algoritma:

- Deteksi wajah: Haar Cascade Classifier
- Deteksi landmarks: Local Binary Features (LBF) Model



Gambar 3 Flowchart

3. Implementasi

Implementasi dilakukan dengan mengembangkan program menggunakan bahasa pemrograman Python. Pustaka utama yang digunakan adalah:

- OpenCV (cv2): Digunakan untuk pengolahan citra, deteksi wajah menggunakan Haar Cascade

- NumPy: Digunakan untuk manipulasi array dan data numerik membantu dalam konversi data wajah yang terdeteksi ke dalam format yang sesuai untuk pemetaan landmark.
- Urllib: Digunakan untuk mengunduh model deteksi secara otomatis

Proses deteksi dimulai dengan inisialisasi webcam dan pembacaan frame secara real-time. Setiap frame diubah menjadi gambar grayscale untuk mempercepat proses deteksi. Setelah itu, Haar Cascade digunakan untuk mendeteksi wajah, diikuti dengan identifikasi 68 titik landmark pada wajah menggunakan LBF. Selanjutnya, titik-titik landmark tersebut ditampilkan pada frame untuk visualisasi.

A. Konfigurasi Lingkungan Pengembangan

- Versi Python: 3.10
- OpenCV versi: 4.10.0. 84
- NumPy versi: 1.26.0
- Sistem Operasi: Windows 11

B. Proses Inisialisasi Model

- Mekanisme unduh otomatis model:
  - Haar Cascade: Diunduh dari link LFBModel.rar - Google Drive
  - LBF Landmark Model: Diunduh dari link LFBModel.rar - Google Drive
- Proses verifikasi integritas file model
- Manajemen penyimpanan model dalam direktori 'data'

C. Konfigurasi Deteksi Wajah

- Metode konversi warna: CV2.COLOR_BGR2GRAY

D. Konfigurasi Landmark Detection

- Model: LBF (Local Binary Features)
- Jumlah landmark: 68 titik
- Metode fitting: Landmark detector fit()
- Warna marker: Biru (RGB: 255,0,0)
- Ukuran marker: Radius 1 piksel
- Ketebalan garis: 2 piksel

E. Mekanisme Pemberhentian

- Tombol pemberhenti: Kunci 'q'
- Proses pembersihan sumber daya:
  - Melepaskan capture webcam
  - Menutup semua jendela OpenCV

4. Pengujian

Pengujian dilakukan untuk mengevaluasi sistem melalui beberapa parameter, yaitu akurasi deteksi wajah, akurasi facial landmarks, dan performa sistem. Akurasi deteksi wajah dievaluasi melalui perhitungan jumlah wajah yang berhasil terdeteksi, kemampuan deteksi dalam berbagai kondisi pencahayaan, serta tingkat kesalahan deteksi. Akurasi facial landmarks meliputi presisi penandaan 68 titik landmark, stabilitas pelacakan titik landmark, dan konsistensi deteksi dalam berbagai sudut wajah. Sementara itu, performa sistem mencakup kecepatan pemrosesan per frame, penggunaan sumber daya komputasi, dan responsivitas sistem secara real-time. Metrik pengujian yang digunakan meliputi jumlah wajah yang terdeteksi, persentase keberhasilan deteksi landmark, waktu proses per frame, dan kestabilan deteksi.

3. HASIL DAN PEMBAHASAN

```
# Facial Landmark Detection with Python and OpenCV

# Import Packages
import cv2
import os
import urllib.request as urlreq
import numpy as np

# Save the URL for the face detection algorithm in the haarcascade_url variable
haarcascade_url =
"https://raw.githubusercontent.com/opencv/opencv/master/data/haarcascades/haarcascade_frontalface_alt2.xml"
haarcascade = "haarcascade_frontalface_alt2.xml"
haarcascade_clf = "data/" + haarcascade

# Check if 'data' folder exists in the current directory
if os.path.isdir('data'):
```

```
  # Check if haarcascade is in the 'data' folder
  if haarcascade not in os.listdir('data'):
    urlreq.urlretrieve(haarcascade_url, haarcascade_clf)
    print("File downloaded")
  else:
    print("File already exists")
else:
  os.mkdir('data')
  urlreq.urlretrieve(haarcascade_url, haarcascade_clf)
  print("File downloaded")

# Create an instance of the Face Detection Cascade Classifier
detector = cv2.CascadeClassifier(haarcascade_clf)

# Save the URL for the facial landmark detection model in the LBFmodel_url variable
LBFmodel_url = "https://github.com/kurnianggoro/GSOC2017/raw/master/data/lbfmodel.yaml"
LBFmodel = "lbfmodel.yaml"
LBFmodel_file = "data/" + LBFmodel

# Check if 'data' folder exists in the current directory
if os.path.isdir('data'):
  # Check if Landmark detection model is in the 'data' folder
  if LBFmodel not in os.listdir('data'):
    urlreq.urlretrieve(LBFmodel_url, LBFmodel_file)
    print("Landmark detection model downloaded")
  else:
    print("Landmark detection model already exists")
else:
  os.mkdir('data')
  urlreq.urlretrieve(LBFmodel_url, LBFmodel_file)
  print("Landmark detection model downloaded")

# Create an instance of the Facial Landmark Detector with the model
landmark_detector = cv2.face.createFacemarkLBF()
landmark_detector.loadModel(LBFmodel_file)

# Get image from webcam
print("Checking webcam connection...")
webcam_cap = cv2.VideoCapture(0)

while(True):
  # Read the webcam frame
  _, frame = webcam_cap.read()

  # Convert frame to grayscale
  gray = cv2.cvtColor(frame, cv2.COLOR_BGR2GRAY)

  # Detect faces using the haarcascade classifier on the grayscale image
  faces = detector.detectMultiScale(gray)

  for (x, y, w, h) in faces:
    # Detect landmarks on the face in the grayscale image
    _, landmarks = landmark_detector.fit(gray, np.array([faces]))

    for landmark in landmarks:
```

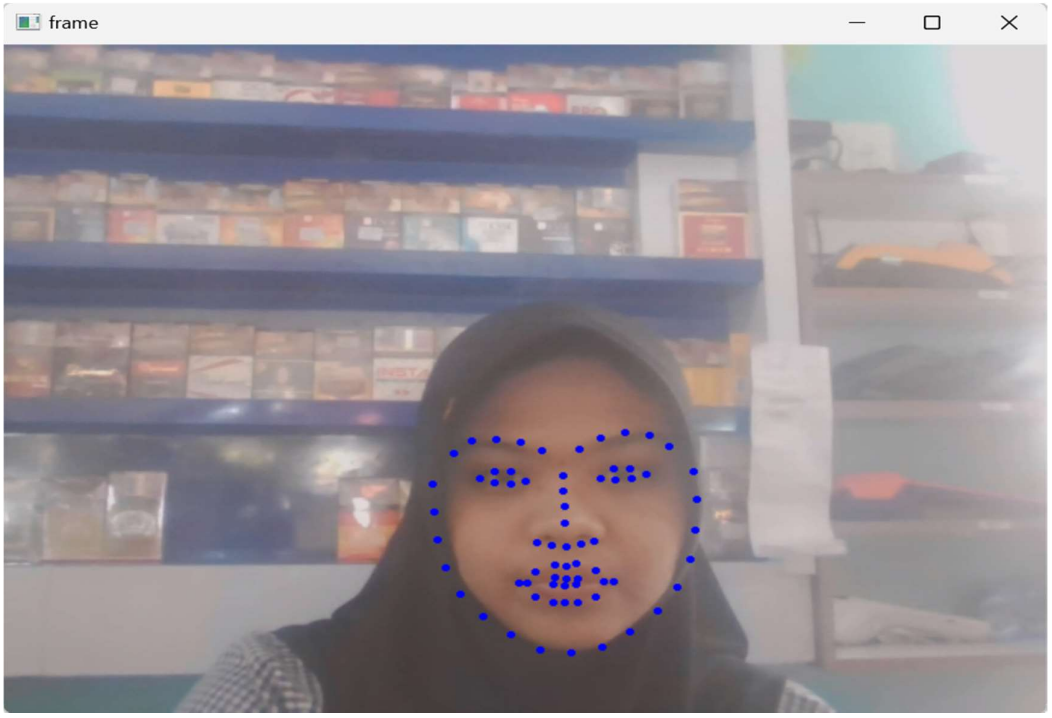
```
# Save the last instance of the detected image
cv2.imwrite('face-detect.jpg', frame)

# Show the image
cv2.imshow("frame", frame)

# Terminate the capture window when 'q' key is pressed
if cv2.waitKey(20) & 0xFF == ord('q'):
   webcam_cap.release()
   cv2.destroyAllWindows()
   break
```

Program ini bertujuan mendeteksi wajah dan landmark secara real-time menggunakan webcam dengan algoritma Haar Cascade untuk deteksi wajah dan Local Binary Features (LBF) Model untuk deteksi landmark. Haar Cascade, yang diimplementasikan menggunakan pustaka OpenCV, bekerja dengan mengunduh file model `haarcascade_frontalface_alt2.xml`, sedangkan LBF Model menggunakan file `lbfmodel.yaml`. Keduanya disimpan dalam folder `data`, yang secara otomatis dibuat jika belum ada. Detektor wajah dibuat menggunakan `cv2.CascadeClassifier`, sedangkan landmark wajah diimplementasikan menggunakan `cv2.face.createFacemarkLBF` (Al-Aidid & Pamungkas, 2018).

Proses deteksi dimulai dengan membaca frame dari webcam, mengubahnya menjadi grayscale untuk mempermudah pemrosesan, lalu mendeteksi wajah menggunakan `detector.detectMultiScale()`. Setelah wajah terdeteksi, detektor landmark menggunakan model LBF untuk mengidentifikasi 68 titik landmark pada wajah. Landmark ini kemudian digambarkan pada frame asli sebagai lingkaran biru kecil pada fitur wajah, seperti mata, hidung, mulut, dan rahang. Setiap frame yang diproses ditampilkan secara real-time menggunakan `cv2.imshow`, dan hasil deteksi terakhir disimpan sebagai file `face-detect.jpg`. Program akan berhenti jika tombol 'q' ditekan.



Gambar 4. Keluaran Kode Program

Hasil deteksi menunjukkan bahwa Haar Cascade dapat mendeteksi wajah dengan baik pada orientasi frontal dan pencahayaan normal, sedangkan LBF Model memberikan deteksi landmark yang cukup akurat. Dalam gambar hasil yang disertakan, dua wajah terdeteksi dengan jelas, dan landmark seperti mata, hidung, mulut, serta kontur wajah berhasil ditandai dengan baik menggunakan titik-titik biru.

Analisis Performa

- Keunggulan:
    - Haar Cascade terbukti efisien untuk mendeteksi wajah dengan orientasi frontal.
    - Model LBF mampu memberikan deteksi landmark yang cukup presisi pada kondisi pencahayaan normal.
    - Algoritma ini cukup ringan dan cocok untuk aplikasi real-time sederhana pada perangkat dengan spesifikasi menengah.
- Keterbatasan:

- Performanya menurun pada kondisi pencahayaan rendah atau jika wajah tidak menghadap langsung ke kamera (non-frontal).
- Beberapa landmark dapat gagal dideteksi jika ada perubahan ekspresi wajah yang ekstrem.
- Pemrosesan dapat melambat jika dijalankan pada perangkat dengan spesifikasi rendah.

Program ini memberikan gambaran implementasi deteksi wajah dan landmark yang sederhana namun efektif menggunakan pustaka OpenCV.

4.KESIMPULAN

Penelitian ini berhasil mengimplementasikan deteksi facial landmarks menggunakan Python dan OpenCV dengan algoritma Haar Cascade dan Local Binary Features (LBF) Model. Sistem yang dikembangkan mampu mendeteksi 68 titik landmark wajah secara real-time dengan akurasi yang baik, terutama pada kondisi pencahayaan normal dan orientasi wajah frontal. Metode ini memiliki potensi aplikasi yang luas dalam bidang computer vision, seperti pengenalan ekspresi, sistem keamanan, dan interaksi manusia-komputer. Meskipun demikian, terdapat keterbatasan pada kondisi pencahayaan rendah dan sudut pandang non-frontal. Untuk peningkatan lebih lanjut, dapat dipertimbangkan penggunaan model deteksi yang lebih modern seperti Deep Learning berbasis Convolutional Neural Networks (CNN) untuk meningkatkan keakuratan, khususnya dalam kondisi pencahayaan buruk atau sudut wajah yang tidak frontal. Selain itu, analisis kecepatan pemrosesan dan perbandingan dengan metode deteksi lain juga dapat dilakukan untuk evaluasi performa yang lebih mendalam.

5. DAFTAR PUSTAKA

Al-Aidid, S., & Pamungkas, D. (2018). Sistem Pengenalan Wajah dengan Algoritma Haar Cascade dan Local Binary Pattern Histogram. *Jurnal Rekayasa Elektrika*, 62-67. doi:10.17529/jre.v14i1.9799

Farokhah , L. (2021). Perbandingan Metode Deteksi Wajah Menggunakan OpenCV Haar Cascade, OpenCV Single Shot Multibox Detector (SSD) dan DLib CNN. *JURNAL RESTI (Rekayasa Sistem dan Teknologi Informasi)*(3), 609-614. doi:10.29207/resti.v5i3.3125

Hidayat, A. N. (2021). *Aplikasi Python untuk Face Detection dengan OpenCV menggunakan Algoritma Haar-Cascade*. (Medium) Diambil kembali dari https://annisanoorhidayat.medium.com/aplikasi-python-untuk-face-detection-dan-face-recognition-dengan-opencv-456aa16f8b3

Maulana, I., Khairunisa, N., & Mufidah, R. (2023). DETEKSI BENTUK WAJAH MENGGUNAKAN CONVOLUTIONAL NEURAL NETWORK (CNN). *JATI (Jurnal Mahasiswa Teknik Informatika), No. 7*(6), 3348-3355. doi:10.36040/jati.v7i6.8171

Prakoso, A. P., Rasyid, A., Deannova, A., & Rahmawan, A. E. (2024). DETEKSI WAJAH MENGGUNAKAN CASCADE CLASSIFIER DENGAN OPENCV-PYTHON. *Jurnal Pengabdian Kepada Masyarakat*, 23-29. doi:10.61488/sikama.v2i1.35

Pranoto, H. (2018). *Pengenalan Wajah: Tahapan Mempelajari Wajah*. (BINUS UNIVERSITY) Diambil kembali dari https://socs.binus.ac.id/2018/12/10/pengenalan-wajah-tahapan-mempelajari-wajah/

Sejati, H. P., & Mardhiyyah, R. (2021). Deteksi Wajah Berbasis Facial Landmark Menggunakan OpenCV Dan Dlib. *Jurnal Teknologi Informasi, Vol.5*, 144-148. doi:10.36294/jurti.v5i2.2220

Susim, T., & Darujati, C. (2021). Pengolahan Citra untuk Pengenalan Wajah (Face Recognition) Menggunakan OpenCV. *Jurnal Syntax Admiration*, 534-545. doi:10.46799/jsa.v2i3.202

Wibowo, A. W., Karima, A., Wiktasari, Yobioktabera, A., & Fahriah, S. (2020). Pendeteksian dan Pengenalan Wajah Pada Foto Secara Real Time Dengan Haar Cascade dan Local Binary Pattern Histogram. *JTET (Jurnal Teknik Elektro Terapan), Vol. 9*, 6-11.

Yulina, S. (2021). Penerapan Haar Cascade Classifier dalam Mendeteksi Wajah dan Transformasi Citra Grayscale Menggunakan OpenCV. *Jurnal Komputer Terapan*, 100-109.