***Artificial Intelligent***



**Supervisor:**

Mr. Rebaz Najeeb

**Submission Date:**

August 3, 2024

Aryan Bestun

Hera Sabir

Objective of the Project:

The primary objective of this project was to develop a real-time system capable of detecting and recognizing facial features such as faces, eyes, noses, and mouths using a webcam feed. The system was to be implemented using OpenCV, a library for computer vision tasks, leveraging pre-trained Haar Cascade classifiers for feature detection.

Tools and Libraries:

•OpenCV: An open-source computer vision library used for image and video processing. It provides pre-trained models and tools for tasks such as object detection and image manipulation.

•Python: The programming language used to implement the system due to its simplicity and the availability of extensive libraries for computer vision.

Methods:

1.Importing Libraries: The OpenCV library was imported to utilize its functions for image processing and feature detection.

2.Defining Functions:

* draw\_boundary: This function converts an input image to grayscale, detects features using a given classifier, draws rectangles around detected features, and annotates them with text.
* detect: This function detects faces in the input image and subsequently detects eyes, noses, and mouths within the detected faces. It calls the draw\_boundary function for each feature.

3.Loading Haar Cascade Classifiers: Pre-trained Haar Cascade classifiers for detecting faces, eyes, noses, and mouths were loaded using OpenCV.

4.Capturing Video: The system captures video frames from the default webcam using OpenCV’s VideoCapture method.

5.Real-Time Detection Loop:

* For each frame captured from the video feed, the detect function is called to identify and annotate facial features.
* The annotated frame is displayed in a window.
* The loop continues until the user presses the 'a' key to terminate the program.

6. Releasing Resources: After breaking the loop, the video capture is released, and all OpenCV windows are destroyed to clean up resources.

Challenges Faced and Solutions:

1.Accuracy of Feature Detection:

* Challenge: Achieving accurate and reliable detection of facial features can be difficult due to variations in lighting, occlusions, and different angles.
* Solution: Fine-tuning the parameters of the Haar Cascade classifiers (such as scaleFactor and minNeighbors) helped improve the accuracy. Additionally, using well-trained classifiers specific to each feature (face, eyes, nose, mouth) ensured better detection.

2.Performance and Real-Time Processing:

* Challenge: Processing video frames in real-time while maintaining high detection accuracy.
* Solution: Efficient implementation of the detection functions and minimizing computational overhead by focusing only on regions of interest (ROI) for subsequent feature detection.

3.Resource Management:

* Challenge: Properly managing the capture and release of video resources and ensuring the program exits cleanly.
* Solution: Implementing appropriate resource release functions (release for video capture and destroyAllWindows for OpenCV windows) ensured no resources were left hanging.

Summary of Results:

The implemented system successfully detected and annotated facial features in real-time using the webcam feed. The key results and observations are as follows:

•The face detection function effectively identified faces in the video feed and marked them with blue rectangles.

•Within the detected faces, the system accurately located and annotated eyes, noses, and mouths with red, green, and white rectangles, respectively.

•The system operated efficiently in real-time, displaying the annotated video feed with minimal latency.

•The project demonstrated the practical application of OpenCV’s Haar Cascade classifiers for real-time feature detection and provided a robust framework for further enhancements, such as integrating more advanced models for improved accuracy.