Report: Face Detection and Feature Localization

Author: Syed Rahmath Ullah Hussaini

Guide: Dr. Ahmed Rimaz Faizabadi

Institution: Delloyd R&D

---

1. Introduction

Face detection and facial feature localization are key tasks in computer vision, used in applications such as biometric authentication, augmented reality, emotion recognition, and human-computer interaction. This project aims to:

- Detect faces in images.
- Locate key facial features:
  - Tip of the nose
  - Centers of the eyes
- Annotate the images with detected features for visualization.

The implementation uses **MediaPipe Face Mesh**, a state-of-the-art framework for real-time facial landmark detection.

---

2. Problem Statement

Develop a program to:

1. Detect faces in static images.
2. Identify and localize specific facial landmarks (nose tip, eye centers).
3. Annotate images with bounding boxes and feature points.
4. Classify the computer vision problem type and justify it.

---

3. Computer Vision Problem Type

This task is a **facial landmark detection problem**, which is a combination of:

- **Object Detection:** Detecting the face in the image.
- **Keypoint Regression:** Localizing specific facial landmarks (nose tip, eye centers) by predicting their (x, y) coordinates.

Justification:
Unlike simple image classification, the algorithm outputs **exact coordinates** for facial features rather than a single class label. This makes it a **regression-based detection problem**.

---

4. Tools and Technologies

- **Python 3.8+**
- **MediaPipe**: For face mesh and landmark detection.
- **OpenCV**: For image processing and annotation.
- **NumPy**: For mathematical operations and landmark calculations.
- **Conda environment**: To manage dependencies and package versions.

---

5. Methodology

1. **Face Detection:**
    - MediaPipe Face Mesh detects up to 5 faces per image.
    - Faces are identified by their landmarks.
2. **Landmark Extraction:**
    - Nose tip index = 1
    - Left and right eyes = sets of predefined landmark indices.
    - Compute the center of eyes using the mean of landmark coordinates.
3. **Annotation:**
    - Draw bounding box around the face.
    - Draw circles at the nose tip and eye centers.

    - Label each feature appropriately.
    - Optional: Draw the full facial mesh for visualization.

4. **Batch Processing:**
    - Input images are taken from a testImage folder.
    - Annotated results are saved to a result folder.
    - Handles multiple images automatically.

---

6. Code Implementation

Key functions:

- compute_landmark_center(landmarks, indices, image_shape)
  Computes average coordinates for a set of landmark indices.
- annotate_face_landmarks(image_path, output_path, draw_mesh=True)
  Detects faces, locates features, and annotates the image.

Example usage:

```
input_folder = "testImage"

output_folder = "result"


for filename in os.listdir(input_folder):

    if filename.lower().endswith((".jpg", ".jpeg", ".png")):

        input_path = os.path.join(input_folder, filename)

        output_path = os.path.join(output_folder, f"annotated_{filename}")

        annotate_face_landmarks(input_path, output_path, draw_mesh=True)
```

---

7. Results

- The program successfully detects faces in images.
- Nose tip and eye centers are accurately localized.
- Annotated images show:

    - Green face bounding box
    - Red circle for nose tip
    - Blue circles for left and right eye centers
- Works for multiple images in batch processing.

Sample Output Image:

(Replace with actual image path in report)

---

8. Observations

- Accuracy depends on image quality and face visibility.
- Works best with frontal faces.
- Can detect multiple faces per image.
- Optional face mesh provides a clear visual understanding of landmark detection.

---

9. Conclusion

This project successfully demonstrates **face detection and facial landmark localization** using MediaPipe. The problem is classified as a **facial landmark detection problem**, a combination of **object detection and keypoint regression**. Annotated images provide clear visualization of detected features, making the system suitable for applications like biometrics, AR, and HCI.

---

10. References

1. MediaPipe Face Mesh Documentation
2. OpenCV Python Documentation
3. NumPy Documentation