University of the Punjab
Gujranwala Campus
Department of Information Technology

Project: Face Detection

Prepared by:
AYESHA SHAHBAZ(BIT21242)

Submitted to:
Miss Fouqia Zaheer

Abstract

This project focuses on facial detection using OpenCV's pre-trained Haar cascade classifier. It abstracts the complex process of image processing and object detection into a simple function that takes an image as input, detects faces, and highlights them.

Key Components

1. **Input Handling:** Loads an image file and converts it to grayscale for processing.
2. **Face Detection:** Uses OpenCV's Haar cascade classifier to detect faces.
3. **Visualization:** Draws rectangles around detected faces and displays the processed image.

- The internal complexities of computer vision, such as image filtering, feature extraction, and classification, are hidden from the user. The function `detect_faces(image_path)` simplifies the process by only requiring an image file as input and automatically performing detection and visualization.

- This simple project loads an image, detects faces using OpenCV's Haar cascade, and draws rectangles around detected faces. Replace 'example.jpg' with your image file.

ACKNOWLEDGEMENT

I would like to express my sincere gratitude to my respected teacher, Ms. Fouqia Zaheer, for her invaluable guidance, support, and encouragement throughout this project. Her expertise and insights have played a crucial role in shaping the direction of my work.

Face Detection using OpenCV

This script detects faces in an image using OpenCV's Haar cascade classifier.

Requirements:
- OpenCV (`pip install opencv-python`)
- A valid image file to process

Functions:
- detect_faces(image_path): Reads an image, detects faces, and displays the result.

Usage:
Call detect_faces('example.jpg') with the path to your image file.

```
import cv2
def detect_faces(image_path):
```

Detects faces in an image and displays the result.

Parameters:
- image_path (str): Path to the input image file.
The function loads an image, converts it to grayscale, detects faces using OpenCV's pre-trained Haar cascade classifier, and draws rectangles around detected faces.

CODE:

```
# Load the Haar cascade classifier for face detection
face_cascade = cv2.CascadeClassifier(cv2.data.haarcascades + 
'haarcascade_frontalface_default.xml')

    # Read the image
    image = cv2.imread(image_path)
    gray = cv2.cvtColor(image, cv2.COLOR_BGR2GRAY)

    # Detect faces in the image
    faces = face_cascade.detectMultiScale(gray, scaleFactor=1.1, minNeighbors=5, minSize=(30, 
30))

    # Draw rectangles around detected faces
    for (x, y, w, h) in faces:
        cv2.rectangle(image, (x, y), (x + w, y + h), (255, 0, 0), 2)


  # Display the result
    cv2.imshow('Face Detection', image)
    cv2.waitKey(0)
    cv2.destroyAllWindows()

# Example usage
detect_faces('example.jpg')
```

SUMMARY

Face Detection using OpenCV

This script detects faces in an image using OpenCV's Haar cascade classifier.

Requirements:

- OpenCV (pip install opencv-python)

- A valid image file for processing

Function:

detect_faces(image_path): Reads an image, detects faces, and displays the result.

Usage:

Call detect_faces('example.jpg') with the path to your image file.