

**Project Report**

Student Name : Nasir Mehmood | Nokhaiz Ahmed ID : CSC-21F-097 | CSC-21F-099

Semester : 6-C Session : Spring 2024

Course : Digital Image Processing Teacher : Sir. Anwar Ali Sathio

## 

## 

## **Project Report: Digital Image Processing for Home Security System**

### **Introduction**

The Home Security System project leverages digital image processing techniques to enhance home security by implementing a facial recognition system. This system captures and recognizes faces using a webcam, and alerts the homeowner through a text-to-speech (TTS) system if an unauthorized person is detected.

### **Objectives**

* Develop a facial recognition system to identify known and unknown individuals.
* Capture and save images of known faces.
* Implement real-time face detection and recognition.
* Use TTS to announce the presence of known individuals or alert about unauthorized access.

### **Technologies Used**

* **Python:** Primary programming language for the project.
* **OpenCV:** Library for computer vision tasks such as capturing and processing images.
* **face\_recognition:** Library for facial recognition tasks.
* **pyttsx3:** Library for text-to-speech conversion.
* **NumPy:** Library for numerical operations.
* **OS and Time:** Standard libraries for file handling and timing functions.

### **System Design**

#### **1. Capturing and Saving Known Faces**

The capture\_face function captures an image of a person's face using a webcam and saves it in the known\_faces directory. The image is saved with the person's name, facilitating future recognition.

Code:  
  
CODE FOR IMAGE SAVE:

import cv2

import os

def capture\_face(name):

cap = cv2.VideoCapture(0)

if not cap.isOpened():

print("Error: Could not open webcam.")

return

print(f"Capturing face for {name}. Press 's' to save and 'q' to quit without saving.")

while True:

ret, frame = cap.read()

if not ret:

print("Error: Could not read frame.")

break

# Display the frame

cv2.imshow('Capture Face', frame)

key = cv2.waitKey(1) & 0xFF

if key == ord('s'):

# Save the frame as an image

known\_faces\_dir = 'known\_faces'

if not os.path.exists(known\_faces\_dir):

os.makedirs(known\_faces\_dir)

image\_path = os.path.join(known\_faces\_dir, f"{name}.jpg")

cv2.imwrite(image\_path, frame)

print(f"Saved {name}'s face to {image\_path}")

break

elif key == ord('q'):

print("Quit without saving.")

break

cap.release()

cv2.destroyAllWindows()

# Example usage

name = input("Enter the name of the person to authorize: ")

capture\_face(name)

#### **2. Verify the face:**

**Code:**

MAIN CODE:

import cv2

import face\_recognition

import pyttsx3

import numpy as np

import os

import time

# Initialize text-to-speech engine

tts\_engine = pyttsx3.init()

tts\_engine.setProperty('rate', 150) # Set speech rate

tts\_engine.setProperty('volume', 0.9) # Set volume level

# Load known faces and their names

known\_face\_encodings = []

known\_face\_names = []

# Path to known faces directory

known\_faces\_dir = 'known\_faces'

# Check if known faces directory exists

if not os.path.exists(known\_faces\_dir):

os.makedirs(known\_faces\_dir)

print(f"Directory '{known\_faces\_dir}' created. Please add images of known individuals and rerun the script.")

exit()

# Load known faces from the directory

for file\_name in os.listdir(known\_faces\_dir):

image\_path = os.path.join(known\_faces\_dir, file\_name)

image = face\_recognition.load\_image\_file(image\_path)

face\_encodings = face\_recognition.face\_encodings(image)

if face\_encodings: # Ensure there's at least one face encoding

known\_face\_encodings.append(face\_encodings[0])

known\_face\_names.append(os.path.splitext(file\_name)[0])

# Check if any known faces were loaded

if not known\_face\_encodings:

print("No known faces loaded. Please add images of known individuals to the 'known\_faces' directory and rerun the script.")

exit()

def detect\_features():

cap = cv2.VideoCapture(0) # Change the index if your webcam is not at index 0

if not cap.isOpened():

print("Error: Could not open webcam.")

return

last\_announcement\_time = 0

announcement\_delay = 10 # seconds

while True:

ret, frame = cap.read()

if not ret:

print("Error: Could not read frame.")

break

# Flip the image horizontally for a later selfie-view display

frame = cv2.flip(frame, 1)

rgb\_frame = cv2.cvtColor(frame, cv2.COLOR\_BGR2RGB)

# Resize frame to improve speed

small\_frame = cv2.resize(rgb\_frame, (640, 480))

# Detect faces

face\_locations = face\_recognition.face\_locations(small\_frame)

face\_encodings = face\_recognition.face\_encodings(small\_frame, face\_locations)

for (top, right, bottom, left), face\_encoding in zip(face\_locations, face\_encodings):

matches = face\_recognition.compare\_faces(known\_face\_encodings, face\_encoding)

name = "Unknown"

if True in matches:

face\_distances = face\_recognition.face\_distance(known\_face\_encodings, face\_encoding)

best\_match\_index = np.argmin(face\_distances)

if matches[best\_match\_index]:

name = known\_face\_names[best\_match\_index]

current\_time = time.time()

if current\_time - last\_announcement\_time > announcement\_delay:

tts\_engine.say(f"Hello, {name}")

tts\_engine.runAndWait()

last\_announcement\_time = current\_time

else:

# Save unknown face

unknown\_face\_dir = 'unknown\_faces'

if not os.path.exists(unknown\_face\_dir):

os.makedirs(unknown\_face\_dir)

unknown\_face\_path = os.path.join(unknown\_face\_dir, f"unknown\_{top}{right}{bottom}\_{left}.jpg")

cv2.imwrite(unknown\_face\_path, frame[top:bottom, left:right])

current\_time = time.time()

if current\_time - last\_announcement\_time > announcement\_delay:

tts\_engine.say("Unauthorized person detected")

tts\_engine.runAndWait()

last\_announcement\_time = current\_time

# Draw a box around the face

cv2.rectangle(frame, (left, top), (right, bottom), (0, 255, 0), 2)

# Draw a label with a name below the face

cv2.rectangle(frame, (left, bottom - 35), (right, bottom), (0, 255, 0), cv2.FILLED)

font = cv2.FONT\_HERSHEY\_DUPLEX

cv2.putText(frame, name, (left + 6, bottom - 6), font, 0.9, (255, 255, 255), 1)

cv2.imshow('HomeSecurity', frame)

if cv2.waitKey(1) & 0xFF == ord('q'):

break

cap.release()

cv2.destroyAllWindows()

# Example usage

detect\_features()

### **Future Work (Continued)**

* **Automated Learning:** Implement a feature that can learn and update the known faces database automatically over time.
* **Security Enhancements:** Incorporate additional security measures such as encrypted data storage and secure transmission protocols to protect sensitive information.
* **Environmental Adaptations:** Improve the system’s robustness to various lighting conditions and angles of view to enhance its performance in diverse environments.
* **Integration with Home Automation:** Link the facial recognition system with other home automation systems, such as smart locks or alarm systems, to automatically respond to recognized or unrecognised faces.
* **Behavioral Analysis:** Integrate behavioral analysis to monitor and recognize unusual patterns of activity, adding an extra layer of security.

### **Challenges**

* **Lighting Conditions:** Varying lighting conditions can affect the accuracy of face recognition, requiring robust algorithms to handle such variations.
* **Real-Time Processing:** Ensuring the system processes video feeds in real-time without significant delay is crucial for timely alerts and responses.
* **Data Privacy:** Managing and storing biometric data securely is critical to prevent unauthorized access and misuse.

### **Results and Evaluation**

To evaluate the system's performance, several tests were conducted under different conditions. The key metrics used for evaluation include:

* **Accuracy:** The system's ability to correctly identify known and unknown individuals.
* **Response Time:** The time taken to process and announce recognition results.
* **User Feedback:** Satisfaction levels of users interacting with the system.

### **Conclusion**

The Home Security System project effectively combines digital image processing and facial recognition to create a robust and user-friendly security solution. With continuous improvements and integration of advanced features, this system has the potential to significantly enhance home security and provide peace of mind to homeowners.