Automatic Door Open on Face Recognition

Most doors are controlled by people who use keys, security cards, a password, or any other pattern to open the door, In this project I helped users* to improve *door security in sensitive locations, using artificial intelligence techniques in face detection and recognition.

Users' facial images can be stored in the database.

This system mainly consists of three subsystems:

1) Face detection and recognition:

At this stage, the face area in the image is detected. If the face is recognized, the value of true will be returned to the variable in which we save the results. But if the face is unknown, the value of false is attributed to it.

2) Sending data to the microcontroller:

After processing the data and identifying whether the face is known or not, we send a signal to the microcontroller from the serial port through the pyserial library

In the first case, if the face is known, we send ON, and in the second case, if the face is unknown or there is no face in the picture, we send OFF.

3) Receiving data in the microcontroller and opening or closing the door:

The data sent from the serial port is read into the microcontroller

If it is ON, it will command the door to be opened and the green light will turn on, but if it is OFF, the red light will turn on and the door will not be opened.

Python code:

```python
# import libraries _________________
import cv2
import face_recognition
import os
import glob
import serial
import time
import serial.tools.list_ports
# Extract the port connected to the Arduino _________________
ports = serial.tools.list_ports.comports()
for i in ports:
    port=(i[0])
# Define people's photos in databases _________________
known_faces = []
known_names = []
known_faces_paths = []
registered_faces_path = 'dataset/'
for name in os.listdir(registered_faces_path):
    images_mask = '%s%s/*.jpg' % (registered_faces_path, name)
    images_paths = glob.glob(images_mask)
    known_faces_paths += images_paths
    known_names += [name for x in images_paths]
def get_encodings(img_path):
    image = face_recognition.load_image_file(img_path)
    encoding = face_recognition.face_encodings(image)
    return encoding[0]
known_faces = [get_encodings(img_path) for img_path in known_faces_paths]
# Camera selection _________________
vc = cv2.VideoCapture(0)
prev_frame_time = 0
new_frame_time = 0
# data transfer protocol
try:
    ser = serial.Serial(port, 115200)
except:pass
finally:
    time.sleep(0.5)
```

```
# real time tracking face _________________
while True:
    ret, frame = vc.read()
    if not ret:
        break
    frame_rgb = cv2.cvtColor(frame, cv2.COLOR_BGR2RGB)
    faces = face_recognition.face_locations(frame_rgb)
    new_frame_time = time.time()
    fps = 1 / (new_frame_time - prev_frame_time)
    prev_frame_time = new_frame_time
    fps = int(fps)
    fps = f'FPS:{str(fps)}'
    font = cv2.FONT_HERSHEY_SIMPLEX
    cv2.putText(frame, fps, (10, 60), font, 2, (100, 200, 200), 5, cv2.LINE_AA)
    if len(faces)==1:
        for face in faces:
            top, right, bottom, left = face
            cv2.rectangle(frame, (left, top), (right, bottom), (0, 0, 255), 2)
            face_code = face_recognition.face_encodings(frame_rgb, [face])[0]
            results = face_recognition.compare_faces(known_faces, face_code, tolerance=0.6)
            if any(results):
                name = known_names[results.index(True)]
                cv2.putText(frame, name, (left, bottom + 20), cv2.FONT_HERSHEY_PLAIN, 2, (0, 255, 0), 2)
                try:
                    ser.write("ON\r".encode())
                except:pass
            else:
                name = 'unknown'
                cv2.putText(frame, name, (left, bottom + 20), cv2.FONT_HERSHEY_PLAIN, 2, (0, 0, 255), 2)
                try:
                    ser.write("OFF\r".encode())
                except:pass
    elif len(faces)>0:
        try:
            ser.write("OFF\r".encode())
        except:pass
        cv2.rectangle(img=frame, pt1=(15, 70), color=(10, 200, 0), pt2=(543, 120), thickness=5)
        cv2.putText(frame, 'Please one person', (20, 110), cv2.FONT_HERSHEY_PLAIN, 3, (0, 0, 255), 5)
    else:
        try:
            ser.write("OFF\r".encode())
        except:pass
        cv2.rectangle(img=frame, pt1=(85, 175), color=(10, 200, 0), pt2=(550, 300), thickness=3)
        cv2.putText(frame, 'there is no one', (124, 245), cv2.FONT_HERSHEY_PLAIN, 3, (0, 0, 255), 2)
    cv2.imshow('face recognition', frame)
```

```
            k = cv2.waitKey(1)
            # The program closes if you press the letter q on the keyboard
            if ord('q') == k:
                try:
                    ser.write("OFF\r".encode())
                except:pass
                break
        cv2.destroyAllWindows()
        vc.release()
```

Arduino code:

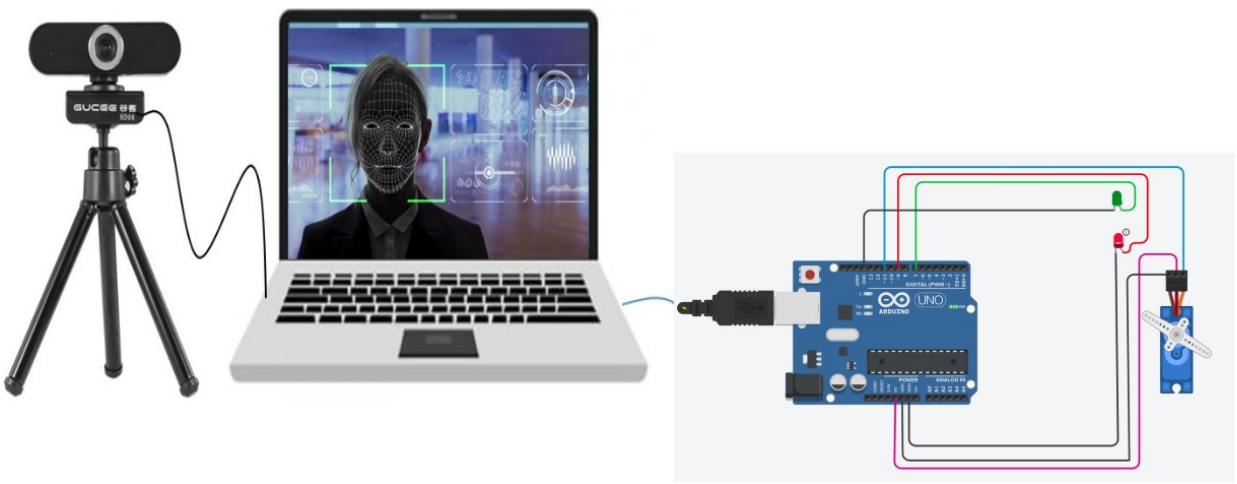
Fichier Édition Croquis Outils Aide

sketch_aug03a

```
Servo servo;
int green = 7;
int red = 9;
void setup() {
servo.attach(11);
Serial.begin(115200);
pinMode(red,OUTPUT);
pinMode(green,INPUT);
}
void loop() {
while (Serial.available()==0){
}
myCMD=Serial.readStringUntil('\r');
  if (myCMD=="ON"){
  servo.write(180);
  digitalWrite(green,1);
  digitalWrite(red,0);
  }
  else if (myCMD=="OFF") {
  servo.write(0);
  digitalWrite(red,1);
  digitalWrite(green,0);
}
}
```

Compilation terminée.

Schematic:



Automatic Door Open on Face Recognition