CONSTRUCTOR UNIVERSITY

MCO0015-340112
Data Acquisition Technologies and Sensor
Fall 2022

Prof. Dr.-Hu, Fangning

Group Project Work

	Group Members	
1	Samson, Jhealyn Bautista	Matr Nr. 30006053
2	Zamora, Justin Dohn Goyena	Matr Nr. 30005958
3	Sundar, Anurag	Matr Nr. 20332594

Date: 30.12.2022

Table of Contents

List of Figures

Abstract

Smart systems that leverage technological advancements and sensor data is an integral element of automation, and this report looks at automous authentication for security purposes. Authentication is used in every sector of operations and in our day to day lives, whether that be entering your PIN number in an ATM machine or getting your documents verified in immigration. The application of authentication into different usecases are limitless and what we propose in the long run is an easy plug and play system that combines live sensor data and a facial recognition model to authenticate users. As a start an MVP model for a homeowner that will use the facial recognition system to automatically open a door was built. The simple system consisted of Arduino I/O components to simulate a doorbell and door of a house, as well as python packages to capture the image, connect to the remote server and run the facial recognition model.

1 Introduction

A great example of utilizing a facial recognition system for autonomous authentication is looking into the usecase of convenience for your average homeowner. A system that can verify the identities of individuals removes the need to bring a key or worry about it from ever getting stolen, as well as report to the homeowner any suspicious activities. This type of system provides both security and convenience. Taking that into account this report contains a breakdown of the proposed system, which includes the Arduino I/0 components to simulate the system, the methodology, results and discussion of how parts of the system can be reused for other security applications and a conclusion that also includes a future outlook into how the system can be further improved and applied to different usecases.

2 Components

Input components of the model:

Arduini Uno (any model) x 1

Arduino Push Button x1

For the camera, an inbuilt camera or external usb camera connected to the laptop can be used.

Output components of the model:

Arduini Uno (any model) x 1

HC-05 Bluetooth Serial module x 1

9g Micro Servo x 1

3 Methodology

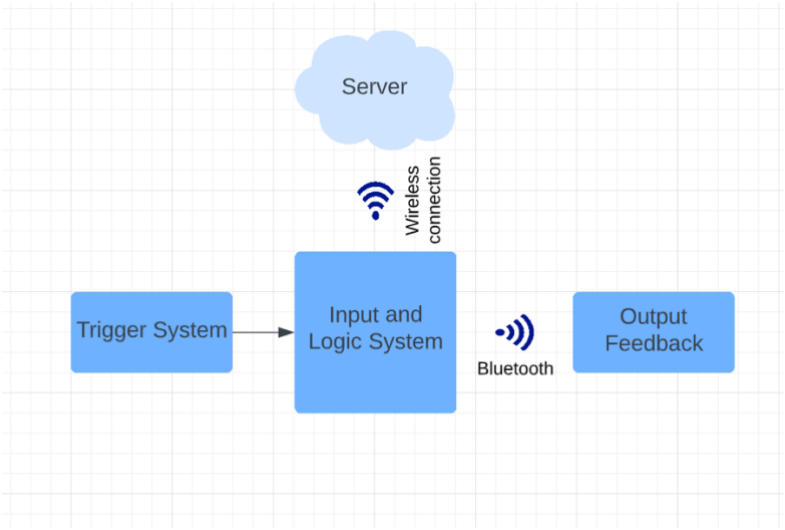

Figure 1: Overview oft he MVP system for the homeowner usecase

The system can be broken down into 3 major parts: the Trigger system, the Input & Logic System that also connects to the remote server through a wireless connection, and the Output Feedback system as depicted in Figure 1.

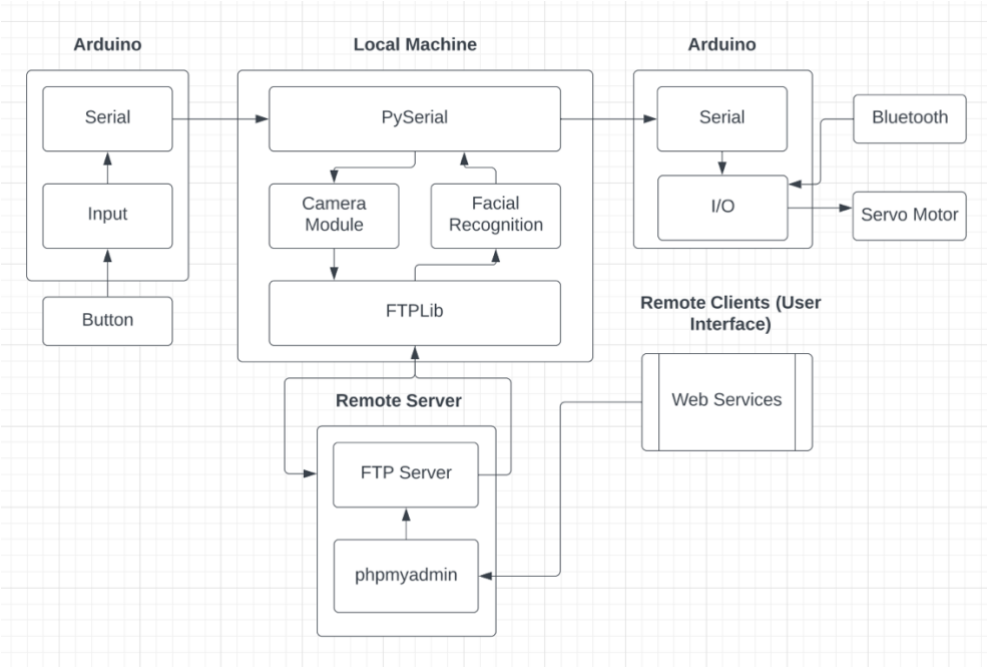

Figure 2 Further Breakdown of the system

3.1 Trigger System

The trigger system consists of an Arduino board that takes in analog data as the input from a button as shown in Figure **3**. A closer look of the circuits of the Arduino components are illustrated in Figure 3. In relation to the usecase the button represents someone ringing the doorbell. Then once the button is pressed an analog signal is taken in by the Arduino board and converted into string. The string is then displayed in the serial monitor of the connecting local machine. The trigger system can be replaced with motion sensors or other types. It was kept simple as the trigger is interchangeable, depending on the usecase and is not the core part of the plug and play feature of the future system.

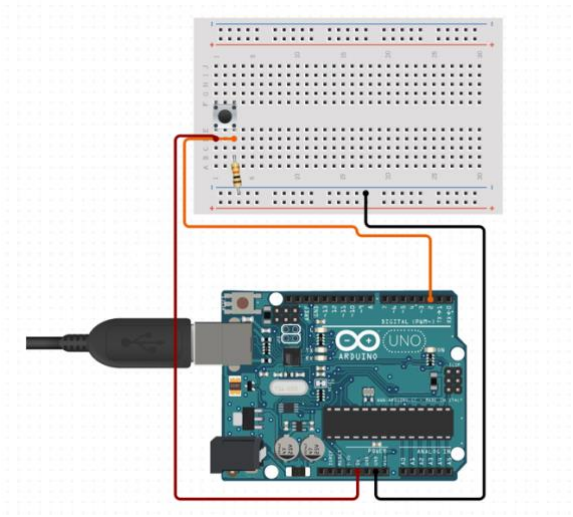

Figure 3: Trigger System

3.2 Input and Logic System

The input and logic system are run in a python environment and the script is broken into four critical modules as shown in the Local Machine block in Figure 2. These include the PySerial, Camera Module, FTPLib and the Facial Rcognition Modules. Below is a sequence of how the modules are run and the function of each of them:

1. PySerial: PySerial is a python library that “encapsulates the access for the serial ports” (PySerial) of the local machine. A function in the script waits for a string from the trigger system and using the pyserial python package, the string is read and passed to the Camera Module of the script.

2. Campure Module: This module consist of a function that captures images using the OpenCV package. The library utilizes the inbuilt camera on the local machine. It can also capture images from an external USB camera. The image gets saved on the local machine.

3. FTPLib: The new image captured in the Camera Module then gets uploaded to the Remote FTP server wirelessly using the FTPlib package as depicted in Figure 2. The new uploaded image can be viewed by remote clients once they have entered the right username/password combination, which is saved in the remote database (accessed in the backend through phpmyadmin).

Afterward, The library of known images are fetched from the remote FTP server using the FTPLib again. The library of known images are saved in the local machine and later deleted after the Facial Recognition Module is run.

4. Facial Recognition module: Once the library of known images is fetched, using the Face recognition python package that has a pre-trained model, the human features from the images are extracted, encoded, then compared to the new image or the image captured in the Capture Module. If there is a match, a string is printed out to the serial monitor and if not, an error message is shown. It is important to note down that the package has limitations, such as spitting out an error if there are no visible distinct human features. Therefore, it is sensitive to too much light or sudden movements.

5. Pyserial: the string is printed out to the serial monitor by using Pyserial. Once it shows up in the serial monitor this will trigger the Output Feedback system.

The logic component of the entire workflow is what would be continuously improved as it can be integrated into other usecases where facial recognition can be used for authentication.

The Remote Client component in Figure 2 refers to the simple user interface, where the user can log in to the website and upload new images in the library of ‘known images’.

The server was hosted in Jacobs university’s network and a VPN (Cisco) had been used to connect to the network remotely.

3.3 Output Feedback System

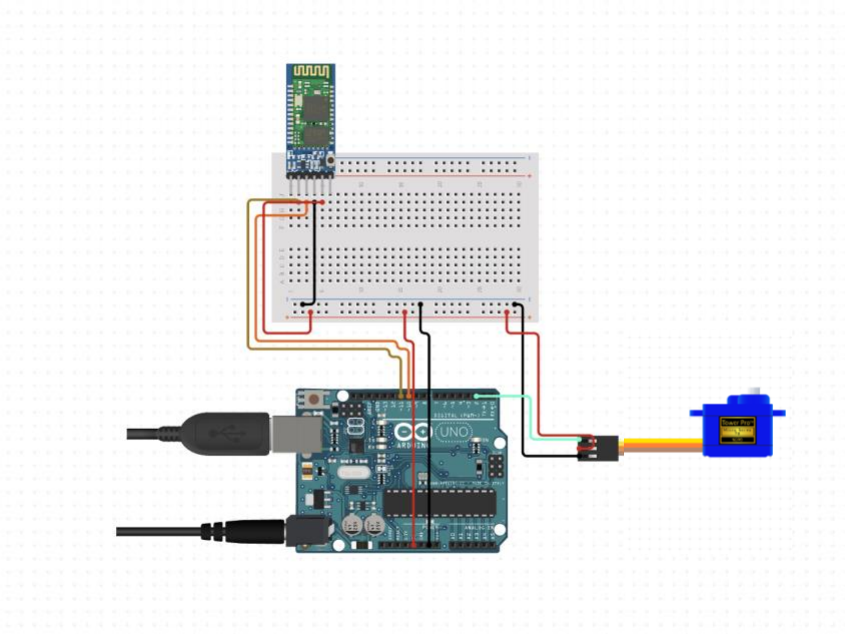

Figure 4 Output Feedback System

As shown in Figure 2 the Output Feedback system consists of an Arduino board, connected to a Bluetooth serial module and a servo motor. The Bluetooth module is the input component and the servo motor is the output component. A better look at the circuits is depicted in Figure 4. The Bluetooth module needs to first pair and connect with the local machine for it to start waiting in a specified port for a string that will trigger the output feedback system and move the servo motor. The local machine itself will determine the outgoing COM port. The servo will then move 90^{o} and back again to its original starting point.

4 Results and Discussion

The original workflow consisted of using an OV70x series camera module to capture the images and send the images via Bluetooth to the logic component. However, due to constraints on the hardware, the workflow had to be adapted to using an external USB webcam. The new workflow also posed some issues and a few workarounds needed to be made to a few of the components such as the button connected to the microcontroller that triggers the system to capture the image, including a delay so that the system will not capture multiple images as a chattering phenomenon sometimes occurred. The captured image from OpenCV also needed to have a few milliseconds of delay to give time for the webcam to "prepare". Without the delay, the images appeared dark since it needed a second or two to "start-up". The new workflow in Figure 1 works after the workarounds though the timing of the overall system takes around three to four seconds before the user receives any feedback. This is expected because of the delays added, but it is something that can be further improved. Though the response time of the model if the library of known images is large has not been tested, it is expected that a significant lag could occur.

As an MVP model parts of the system, mostly the logic component can be further improved and reused for other applications or use cases. The input and output components can be modeled using the current workflow too, but with improvements that are further discussed in Section 5 of the report.

5 Conclusion and Future Outlook

The project was successful in applying a small-scale facial recognition system that can be used for homeowners. The future outlook for this project is for the system to be adapted in various usecases, such as authenticating the person making transactions on an ATM to automating immigration processes in airports for faster authentication using facial recognition. Though further improvements are needed for the next iteration of the project. The facial recognition model should be improved to lessen the possibility of the system matching lookalikes and delay the processing time of the model if the library of known images is large. The camera python package needs to be improved as well to reduce the time it takes for it to capture images and for it to be compatible with most common camera models. The remote user interface to see images can be improved too by including editing access and better user authentication.

Bibliography

Sources from the internet:

Welcome to pySerial's documentation (no date) *Welcome to pySerial's documentation - pySerial 3.4 documentation*. Available at: https://pyserial.readthedocs.io/en/latest/ (Accessed: December 30, 2022).

Face-recognition (no date) *PyPI*. Available at: https://pypi.org/project/face-recognition/ (Accessed: December 30, 2022).

Ftplib - FTP protocol client (no date) *Python documentation*. Available at: https://docs.python.org/3/library/ftplib.html (Accessed: December 30, 2022).

OpenCV-python tutorials (no date) *OpenCV*. Available at: https://docs.opencv.org/4.x/d6/d00/tutorial_py_root.html (Accessed: December 30, 2022).