Facial Recognition Door Lock

Singularity University iLab

July 2017

**Project**

Facial Recognition Door Lock

**Materials**

Hardware:

* computer
* USB webcam with microphone
* USB speaker
* Lockitron Bolt
* Lockitron Bridge (to access Lockitron Bolt through wifi)

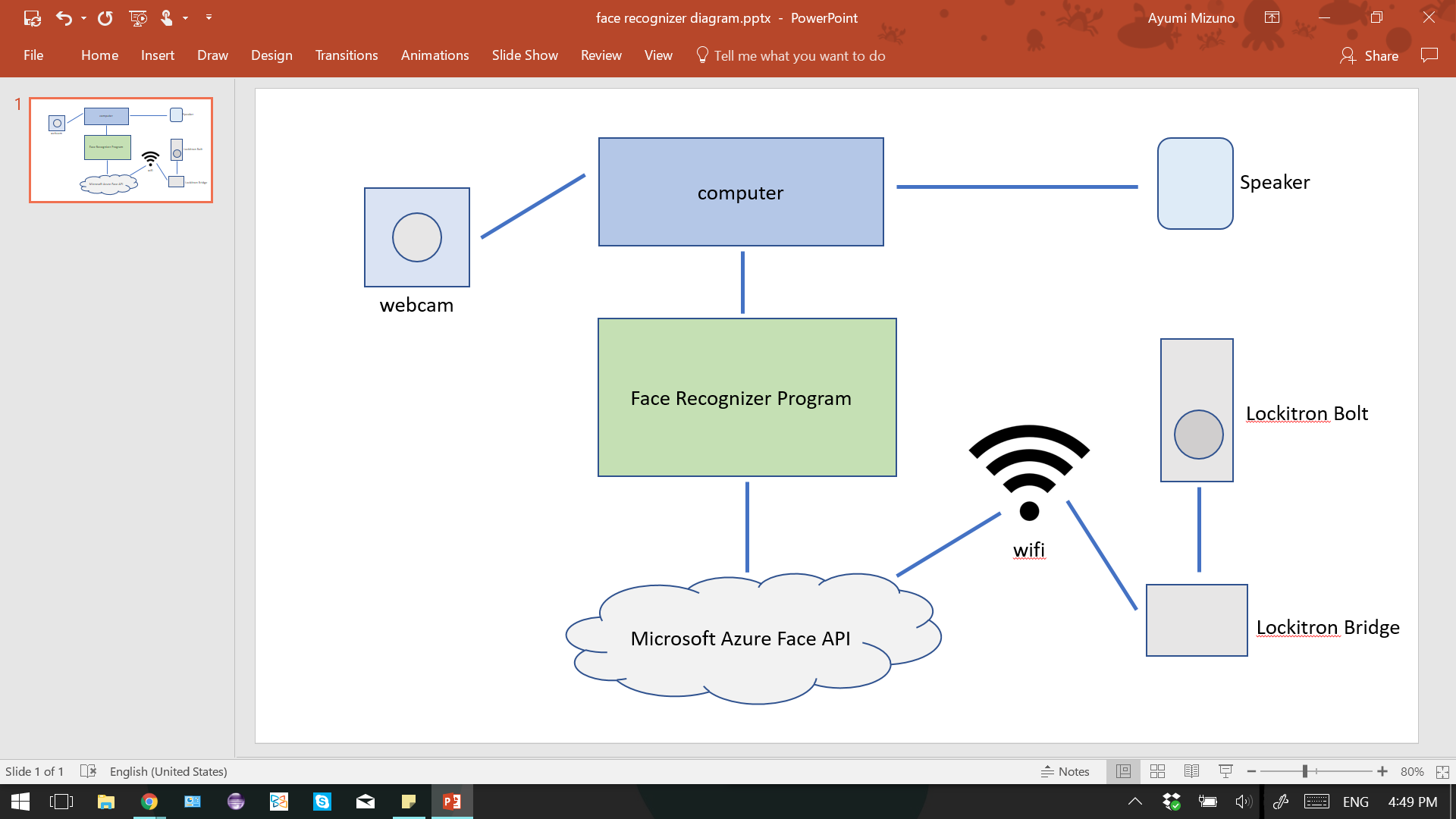
Software:

* Microsoft Face API
* Python 2.7
* Lockitron API
* IBM Bluemix API (speech-to-text and text-to-speech)

Files:

* createPersonGroup.py
* create&addperson.py
* trainPersonGroup.py
* faceRecognizer.py
* getLanguage.py (works with faceRecognizer)
* Webcam.py (works with faceRecognizer)
* deletePerson.py
* deletePersonGroup.py
* directory.txt (works with faceRecognizer)
* timesheet.txt (works with faceRecognizer)

**Setup**



**Method**

The Facial Recognition Door Lock allows a door to be unlocked when a user is recognized. The program begins by taking a picture of the user’s face using a webcam. The picture is sent to the Microsoft Azure Face server to determine if the user is recognized or not. The key methods used from Microsoft Azure Face API include “create”, “train”, and “identify”. In order for the program to recognize a certain user and unlock the door upon recognition of that user, they must first be “created” as a person. The facial recognizer is then “trained” by a picture of the user’s face. If a user has not been “created” and “trained”, the server responds that the user is not recognized.

A directory text file was made in order to keep track of the names of the users who have been created and trained. This allows the program to respond to the user by their name instead of their id number, which is what the server uses to recognize users. Another text file also records a user’s name and the date and time they used the system. This timesheet file enables for more personalized responses. If a user has never used the facial recognizer before, the program greets the user by their full name. After a user’s initial use however, the program greets them by their first name.

The main output of the program is the controlling of the lock. The Lockitron Bolt is a lock connected to the project through wifi by the Lockitron Bridge. The Lockitron API is used to control the lock through the program. If the user is not recognized, the lock remains locked. However, if the user is recognized, the lock becomes unlocked for 3 seconds and locks afterwards.

Another output besides controlling the lock is a speech response, which allows the user to interact with the program and receive personalized greetings. IBM’s Bluemix API is implemented to allow the program to implement speech-to-text as well as text-to-speech. When a user is recognized, the initial greeting asks the user to choose a language for the second, personalized response. The languages include English, Japanese, French, and German, along with the option to skip, which leads to a text response. After the user responds through the microphone connected to the computer, the program outputs a second greeting in which the user’s name is included, along with their predicted age and their emotion. In addition, if the facial recognizer detects makeup on the user, the output includes a statement about the makeup.

**Limitations**

A limitation is that the face recognizer may not provide complete security. The program could be fooled by using pictures of a user instead of the user physically being in front of the webcam. One solution could be to add another security measure such as a keypad to the door.

Another limitation is that the program relies on wifi since there are three APIs implemented in the program. If there is no wifi access, the program will not run. In addition, if the wifi is slow, the Lockitron Bolt will be slow to respond, since the lock is connected to the wifi through the Lockitron Bridge. This could prolong the time the user has to wait in order for the program to unlock the door. A keypad could be added to the door so that the user unlock the door through a code. By combining the facial recognition program with a keypad, both the security and wifi issue can be solved.

**Future work**

* Move program to Raspberry Pi
  + To do this, Windows 10 IOT must first be installed on the Raspberry Pi and the program must be ported to Visual Studio to create a Windows Universal App which can then be deployed onto the Raspberry Pi.
* Figure out way to continuously run program and to detect when to take picture
  + The program should run continuously to prevent users from having to manually start the process of unlocking the door
  + A detector that detects when a person is in front of the webcam may be added so the program knows when to initialize the program and take the picture

**Sources**

<https://docs.microsoft.com/en-us/azure/cognitive-services/face/apireference>

<https://api.lockitron.com/v2/documentation>

<https://www.ibm.com/watson/services/speech-to-text/>

<https://www.ibm.com/watson/services/text-to-speech/>

<https://github.com/Microsoft/Cognitive-Face-Python>

<http://hanzratech.in/2015/02/03/face-recognition-using-opencv.html>