COMP3012
Robotics Term Project

Salim Manji

April 18, 2021

Contents

1 Project Outline

Recently, I have been exploring home security camera systems, and have tried some options commercially available options including the Arlo 2 system. Often, these commercial offerings are expensive, bulky and not suited for Calgary's weather. Not wanting to damage my property by drilling holes through my walls for Ethernet cables, I have been looking for a way to mount the cameras indoors where they will be protected from the elements and from vandalism or theft. Unfortunately, passive infrared camera systems are unable to detect motion through my windows nor capture any footage at night, forcing me to search for alternative approaches. In my efforts, I stumbled up on some microwave or Doppler radar sensors on Amazon. These sensors are very interesting, in that they are able to detect motion through walls, and up to seven meters away from their position. Since no IR would be required, motion could be detected without a pixel based image comparison.

The concern of sensitivity became very obvious during my testing of the sensors, where I realized me (or even my neighbours) moving around the house would trip detection, leading to an unnecessary amount of data being recorded and stored. This data would need to be evaluated manually, and a large amount of storage would be required.

Having some experience with cloud computing and machine learning from my experiences in COMP3504 and COMP3505, I explored the opportunity to implement some cloud functionality through Amazon Web Services (AWS). Professor Yasaman Ammanejad was very kind in extending some unused AWS credit to me, and I have tried to integrate a few of the AWS services into this project.

In addition, I have attempted to integrate openCV, an open source facial detection and recognition system that the RPi is able to use. The setup for this is quite complicated, and my build failed a number of times. Over the summer break, I intend to use docker to mount a container supporting a pre-made openCV build.

2 Required Hardware

1. Raspberry Pi 3B+ (RPi).
2. Raspberry Pi Camera (with ribbon).
3. Arudino Uno (with USB cable).
4. Breadboard.
5. RCWL-0516 Microwave/Doppler Radar sensor.
6. 3x wires to connect sensor to breadboard.
7. Internet access.
8. AWS Account.
9. QNAP NAS.

3 High Level Work Flow

Option 1:

1. If motion is detected by the RCWL-0516 sensor/Arduino, trigger the RPi capture an image.
2. Upload the image to AWS Rekognition Service to analyze the image for facial detection.
3. Return results from AWS Rekognition.
4. Parse JSON results in Python script.
5. IF a person was detected in the image, take a short video clip to monitor intruder actions.
6. Send an email via an SMTP server installed on the RPi.
7. Upload image and video file to QNAP NAS for storage and later retrieval.
8. ELSE delete the image and await the next motion detection.

Option 2:

Please note, I chose to complete the first option only. Over the break, I plan to implement this functionality to avoid any costs associated with Amazon Web Services, in particular Rekognition.

1. If motion is detected by the RCWL-0516 sensor/Arduino, trigger the RPi capture an image.
2. Run openCV on the RPi to see if a face is recognized.
3. IF the length of the detected faces array is greater than 0 (or use a boolean if the length is greater than 0), there is at least one person in the frame, so trigger Pi Camera to record video.
4. Send an email via SMTP server installed on RPi to owner.
5. Upload image and video file to QNAP NAS for storage and later retreival.
6. ELSE delete the image and await the next motion detection.

4 Setup

1. Setup your Raspberry Pi.

   I choose Raspbian as my OS, but there are other operating systems to choose from. Ensure all packages are up-to-date, in particular Python. Once setup, power the RPi down and connect the camera. A bit of dexterity may be required to build the camera housing as the pieces are quite small and the screws short. Lastly, I opted to disable the red led from showing when the camera is active:

   ```
   sudo nano /boot/config.txt
   ```

   (substitue nano for your preferred text editor), then add the following line to the bottom of the text file:

   ```
   disable_camera_led=1
   ```

2. Install ncftp.

   ```
   sudo apt install ncftp
   ```

3. Connect the components.

   This is a pretty basic setup. I have connected 5V from the Arduino into the breadboard, and of course grounded the circuit. One other cable connects the RCWL-0516 sensor to the an input pin on the Arduino to receive data. In addition, the USB cable is connected from the Arduino to one of the USB ports on the RPi for serial communication.

4. Create an account on the NAS for file transfer.

   The Qnap OS is very user friendly, and creating a new account was pretty straightforward. I created a new username and password, and restricted folder access.

   Once done, edit the upload bash file with relevant login information.

5. Create an email account for SMTP use.

   Head over to gmail, create a new account. You will want to setup Two Factor Authentication (2FA), and also adjust account settings to allow for less secure apps to access the account. Lastly, create an App Password to use for email alerts.

   Once done, edit the cctv.py file with relevant login information.

6. Create an AWS account. You will need access to:

   - AWS Rekonition
   - AWS S3 (storage buckets)

   Once your account is setup, create a new bucket that will store images that are uploaded for facial recognition. The bucket name is inputted into the Python script upload.py.

   You will need to run the

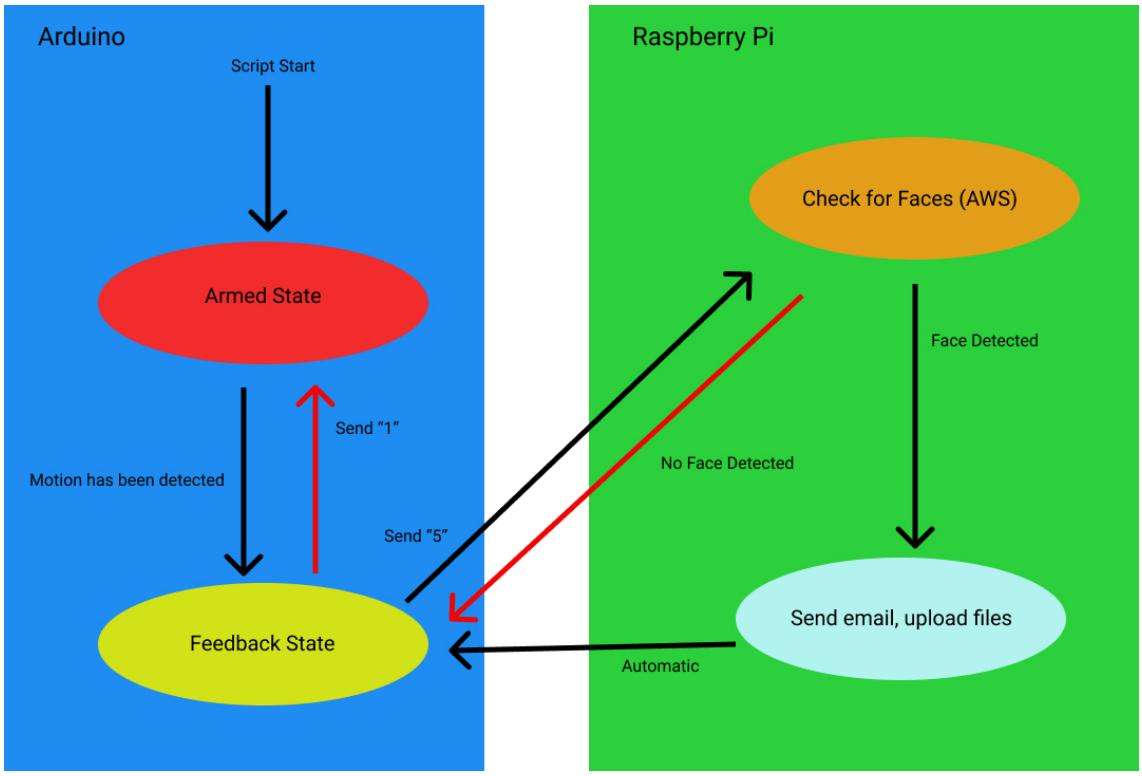
   ```
   aws configure
   ```

   command on the RPi, where you will need to enter your access key and other credential information to link the RPi to your AWS account.

7. Write some scripts!

   I opted to write the bulk of my code in Python. As this was my first exposure to the language, it took a bit of learning to understand the syntax, and it is pretty ugly since I don't know how to write functions in Python yet. Two additional short bash script were written to upload the video file to my QNAP unit via FTP, which deletes the local files upon successful completion and another to delete an image if no faces are detected.

5 State Diagram



6 Code Walk-through

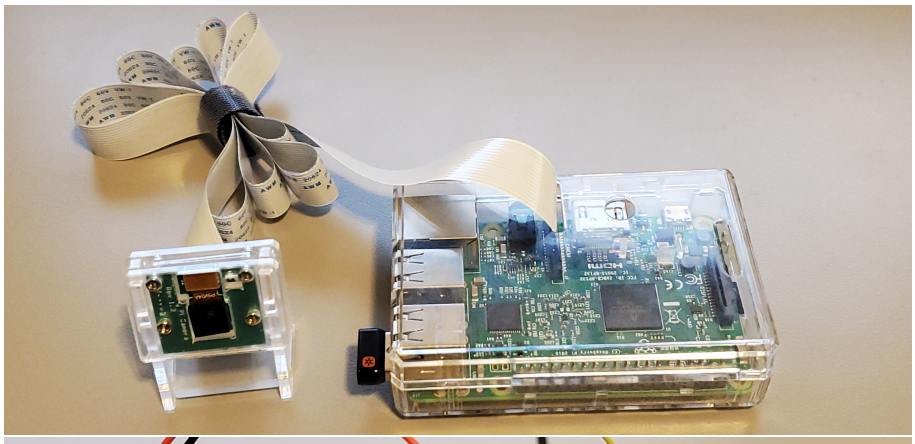
Using the RPi camera is very simple; import the camera library, then create a filename. I opted to use a time stamp to easily identify the motion event. Next, the resolution for the camera is set; I am using very high quality resolution to capture at around 2 MB per image files. Only a minute difference in time was noted when checking for faces through AWS Rekognition, and I thought it would be best to have a large image file for later evaluation. This resolution can be changed to decrease upload time to AWS and for FTP. Lastly, the capture function is called to capture an image.

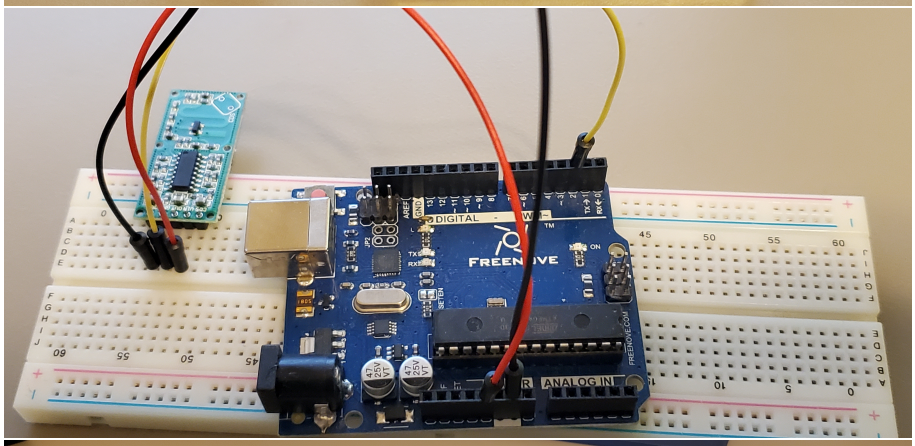
Next, an object is created to handle the AWS upload to an AWS S3 storage bucket. Once the image is uploaded, a response is received from AWS. My code simply checks to see how many faces are detected in the frame, and as long as at least one has been found, the rest of the code executes. Additional information can be extracted, such as emotion and physical attributes, discussed in a following portion.

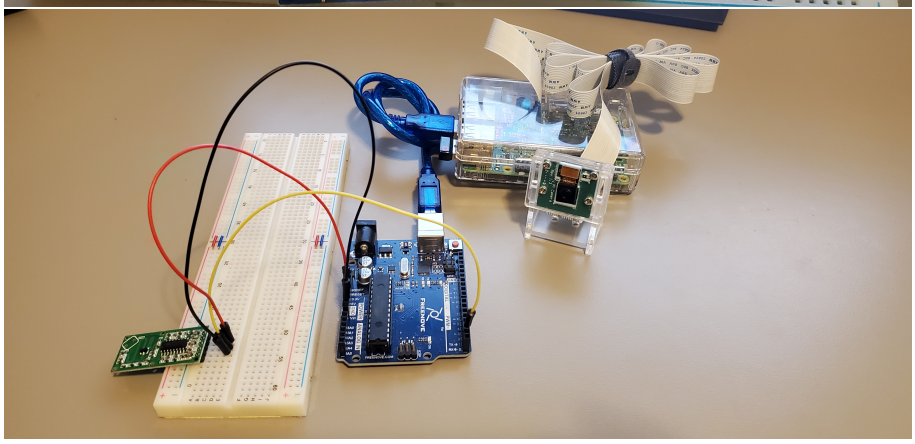
Assuming at least one face has been detected by AWS Rekognition, the camera is set to sleep for a short period of time to adjust for light sensitivity, then a video is captured. Once complete, an alert message is sent to the owner relating to the recent capture, then the "upload" bash script is called which pushes the photo and video to the QNAP NAS.

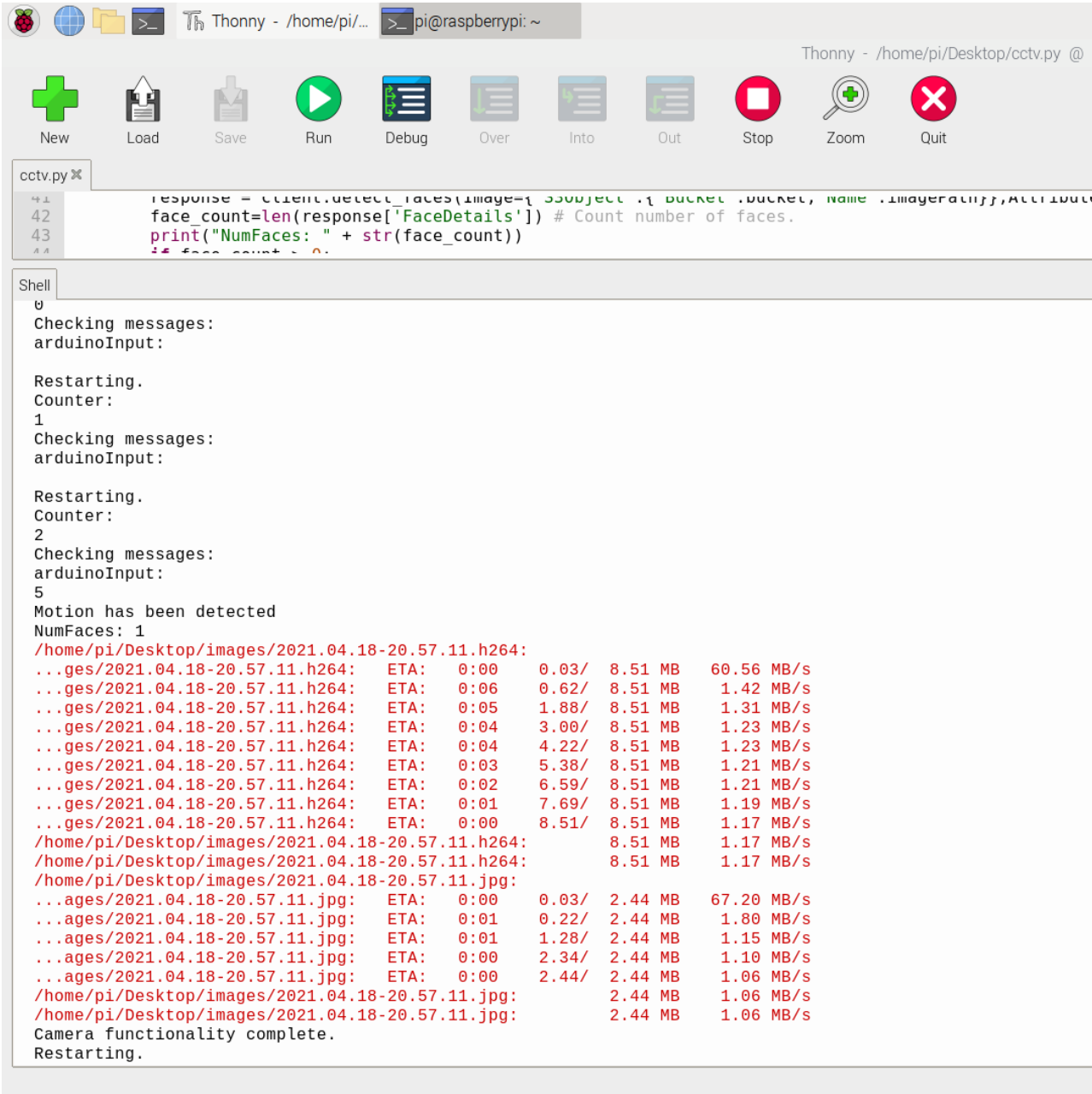
If no faces have been found, the "delete" bash script runs, which simply deletes any files in the images folder on the RPi.

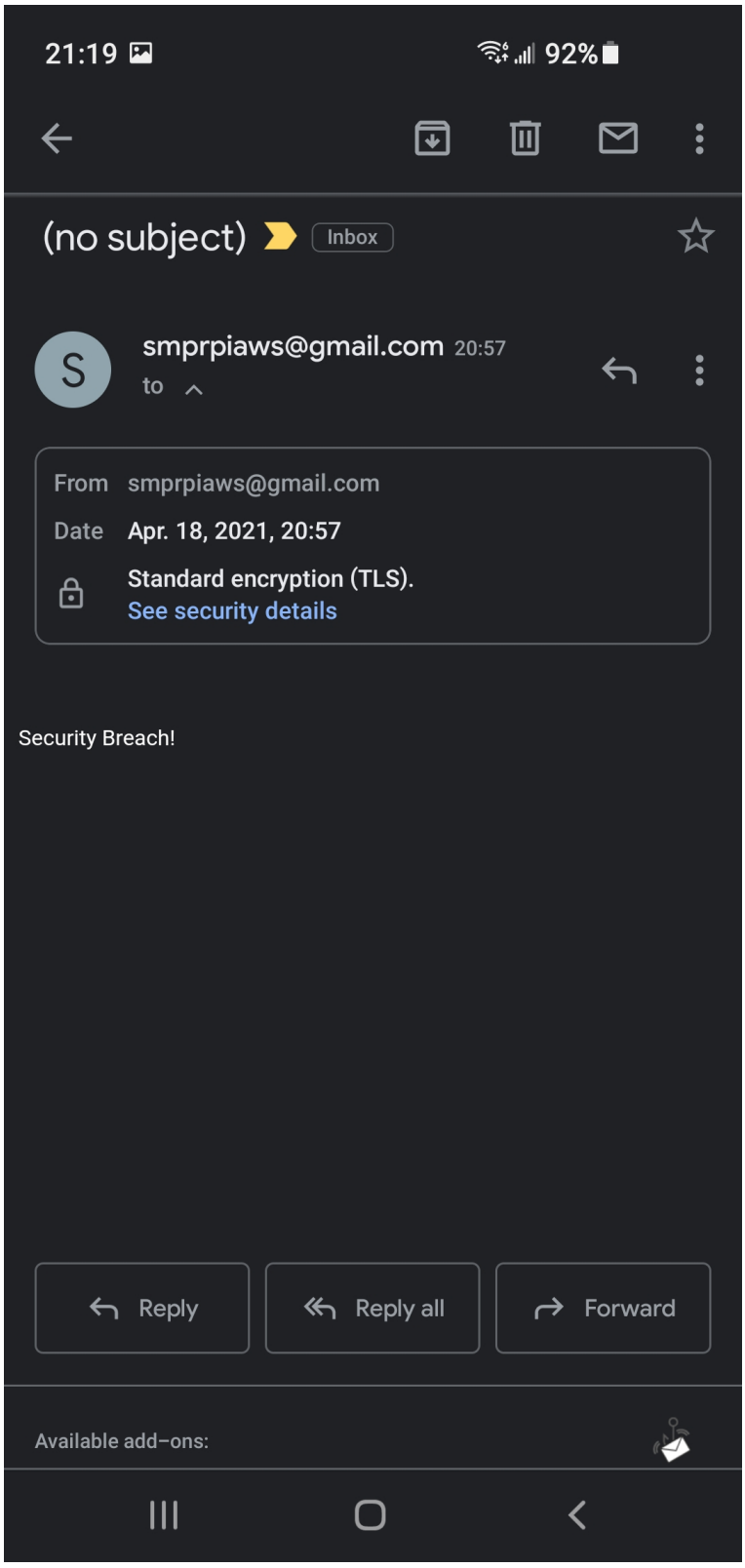
7 Hardware Configuration and Output







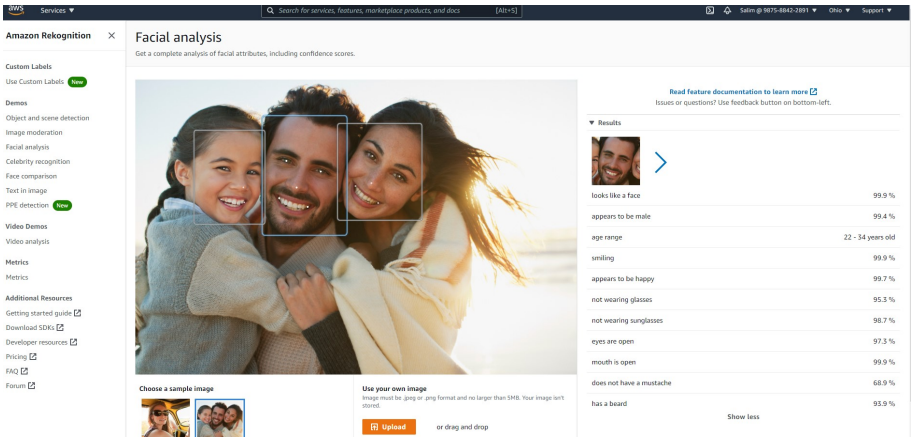

Thonny - /home/pi/...
pi@raspberrypi: ~
Thonny - /home/pi/Desktop/cctv.py @
New
Load
Save
Run
Debug
Over
Into
Out
Stop
Zoom
Quit
cctv.py
face_count=len(response['FaceDetails']) # Count number of faces.
print("NumFaces: " + str(face_count))
Shell
0
Checking messages:
arduinoInput:
Restarting.
Counter:
1
Checking messages:
arduinoInput:
Restarting.
Counter:
2
Checking messages:
arduinoInput:
5
Motion has been detected
NumFaces: 1
/home/pi/Desktop/images/2021.04.18-20.57.11.h264:
...ges/2021.04.18-20.57.11.h264: ETA: 0:00 0.03/ 8.51 MB 60.56 MB/s
...ges/2021.04.18-20.57.11.h264: ETA: 0:06 0.62/ 8.51 MB 1.42 MB/s
...ges/2021.04.18-20.57.11.h264: ETA: 0:05 1.88/ 8.51 MB 1.31 MB/s
...ges/2021.04.18-20.57.11.h264: ETA: 0:04 3.00/ 8.51 MB 1.23 MB/s
...ges/2021.04.18-20.57.11.h264: ETA: 0:04 4.22/ 8.51 MB 1.23 MB/s
...ges/2021.04.18-20.57.11.h264: ETA: 0:03 5.38/ 8.51 MB 1.21 MB/s
...ges/2021.04.18-20.57.11.h264: ETA: 0:02 6.59/ 8.51 MB 1.21 MB/s
...ges/2021.04.18-20.57.11.h264: ETA: 0:01 7.69/ 8.51 MB 1.19 MB/s
...ges/2021.04.18-20.57.11.h264: ETA: 0:00 8.51/ 8.51 MB 1.17 MB/s
/home/pi/Desktop/images/2021.04.18-20.57.11.h264: 8.51 MB 1.17 MB/s
/home/pi/Desktop/images/2021.04.18-20.57.11.h264: 8.51 MB 1.17 MB/s
/home/pi/Desktop/images/2021.04.18-20.57.11.jpg:
...ages/2021.04.18-20.57.11.jpg: ETA: 0:00 0.03/ 2.44 MB 67.20 MB/s
...ages/2021.04.18-20.57.11.jpg: ETA: 0:01 0.22/ 2.44 MB 1.80 MB/s
...ages/2021.04.18-20.57.11.jpg: ETA: 0:01 1.28/ 2.44 MB 1.15 MB/s
...ages/2021.04.18-20.57.11.jpg: ETA: 0:00 2.34/ 2.44 MB 1.10 MB/s
...ages/2021.04.18-20.57.11.jpg: ETA: 0:00 2.44/ 2.44 MB 1.06 MB/s
/home/pi/Desktop/images/2021.04.18-20.57.11.jpg: 2.44 MB 1.06 MB/s
/home/pi/Desktop/images/2021.04.18-20.57.11.jpg: 2.44 MB 1.06 MB/s
Camera functionality complete.
Restarting.

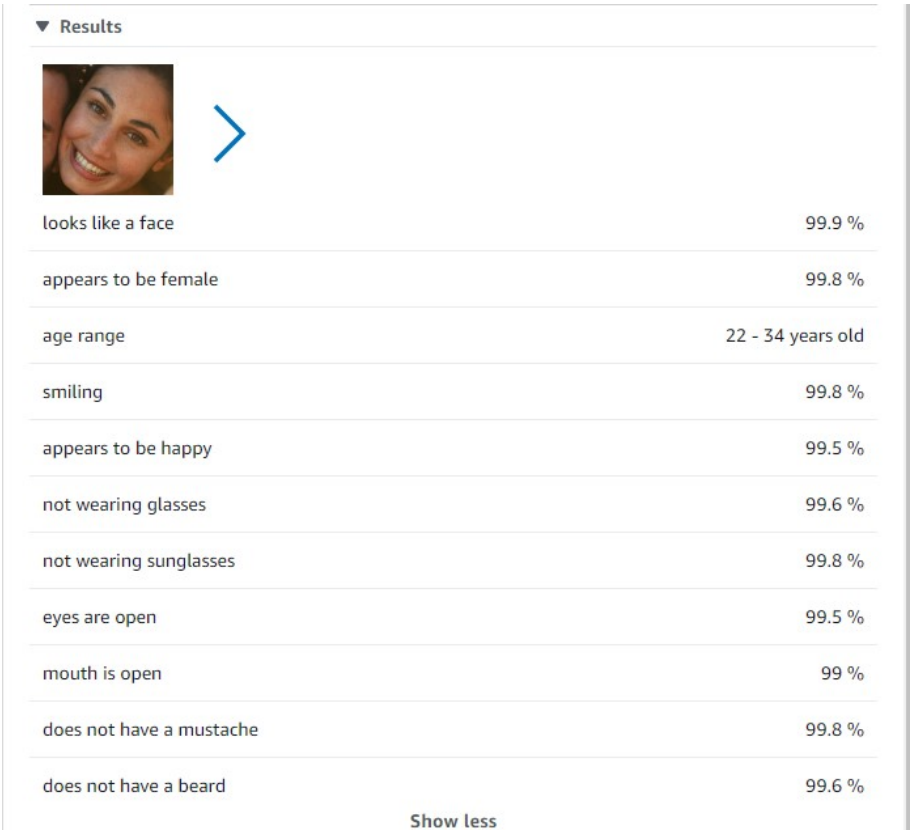

21:19
92%
(no subject)
Inbox
smprpiaws@gmail.com 20:57
to
From smprpiaws@gmail.com
Date Apr. 18, 2021, 20:57
Standard encryption (TLS).
See security details
Security Breach!
Reply
Reply all
Forward
Available add-ons:

8 Notable Discoveries

Please note, it is best to turn debug mode OFF on the Arduino. As a serial connection is established between the Arduino and the RPi, debug information is transmitted to the RPi. As no error handling for this is in place, code execution deviates from normal flow.

While I was previously aware of the fact that providers such as AWS, FaceBook and Google harvest massive amounts of data from us, I was surprised to see just how detailed a response AWS Rekognition was able to provide. Labels such as "not wearing sunglasses", "does not have a mustache" and "age range" have given me a good amount of insight into how much data is being recorded, and how invasive tech companies really are... They provide us will very cool goods and services, but the cost associated with individual privacy is high. Additional attributes, including mood, can also be determined (probability the individual is happy, sad, angry, etc), which worries me with regard to corporate-defined stereotypes being programmed into these algorithms; AI is pre-judging us through the use of developer written machine learning algorithms, creating new divides in social classes. A new feature now included with AWS Rekognition is PPE detection, which can detect whether workers are wearing PPE equipment.




Results
looks like a face 99.9 %
appears to be female 99.8 %
age range 22 - 34 years old
smiling 99.8 %
appears to be happy 99.5 %
not wearing glasses 99.6 %
not wearing sunglasses 99.8 %
eyes are open 99.5 %
mouth is open 99 %
does not have a mustache 99.8 %
does not have a beard 99.6 %
Show less

9 References

- https://docs.aws.amazon.com/sns/latest/dg/sns-setting-up.html
- https://aws.amazon.com/rekognition/getting-started/
- https://boto3.amazonaws.com/v1/documentation/api/latest/guide/s3-uploading-files.html
- https://docs.aws.amazon.com/lambda/latest/dg/welcome.html
- https://docs.aws.amazon.com/code-samples/latest/catalog/python-rekognition-rekognition_image_detection.py.html
- https://nerdynat.com/programming/2019/how-to-install-opencv-on-raspberry-pi-3b/
- https://nerdynat.com/programming/2019/facial-recognition-on-raspberry-pi-using-amazon-rekognition/
- https://www.linkedin.com/pulse/using-aws-ai-detect-people-faces-home-security-camera-khalid-abdulla/?trackingId=HgzdrsaF9q4ETlra5Jh%2FoQ%3D%3D
- https://medium.com/@matt.collins/facial-recognition-with-a-raspberry-pi-and-kinesis-video-streams-part-1-662f0bec5488
- https://medium.com/@matt.collins/facial-recognition-with-a-raspberry-pi-and-kinesis-video-streams-part-2-9c9a631e8c24
- https://www.pyimagesearch.com/2018/06/25/raspberry-pi-face-recognition/
- https://pythonprogramming.net/raspberry-pi-camera-opencv-face-detection-tutorial/
- https://www.raspberrypi-spy.co.uk/2013/05/how-to-disable-the-red-led-on-the-pi-camera-module/
- https://myhydropi.com/send-email-with-a-raspberry-pi-and-python
- https://thesolaruniverse.wordpress.com/2020/10/01/the-rcwl-0516-doppler-radar-motion-sensor-an-arduino-nano-and-an-event-led/
- https://www.arrow.com/en/research-and-events/articles/raspberry-pi-to-arduino-serial-communication-via-usb