Supporting Patients with Amnesia

Yuktha Deesan, Qianyun Wang, Matt Zheng

ECE 196 SP2024

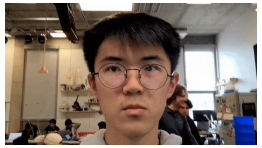
Problem Statement

People with amnesia often struggle to recognize familiar faces.

This can lead to difficulties in identifying friends, family, or regular visitors at their home.

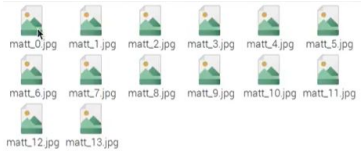
This inability can severely **disrupt their social interactions and compromise their safety**.

Proposed Idea: Door Camera with Facial Recog.



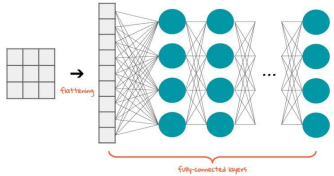
Take a short video of different angles of **subject's** face

Send to Rasp. Pi Via Telegram API

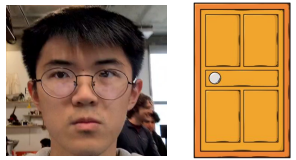


User is prompted to label the **subject**

Train neural net using Rasp. Pi



User is prompted to turn on camera

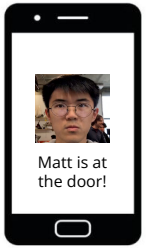


Subject is identified at the door

Proximity Sensing

Is the subject within 1m of the camera?

User receives a smartphones notification



Yes

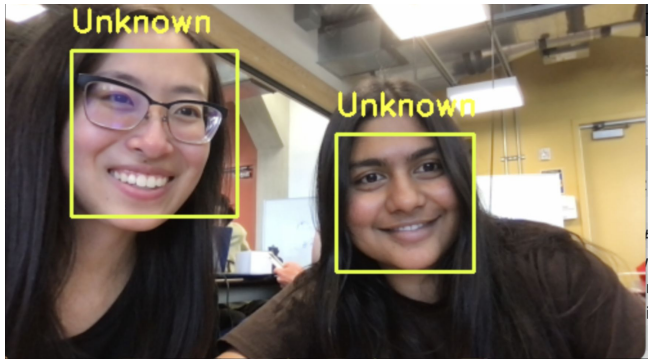
No

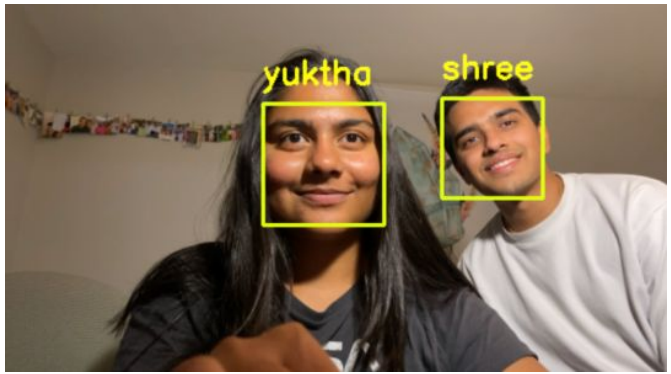
No smartphone notification sent

Testable Hypothesis

Hypothesis: Users who experience amnesia who use our facial recognition door camera experience a higher level of confidence when answering the door as opposed to without one.

Measurement: Noting the ability of our algorithm in combination with the camera module to **detect and classify** people.



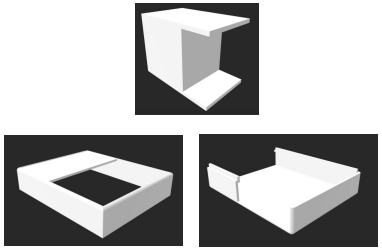


Milestones

【The Ambition Zone】

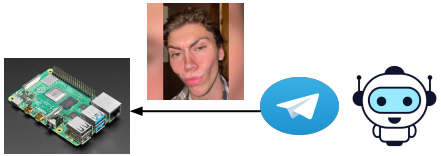
Week 4
Implementing Telegram Bot base functionality

Week 8
Using DLib and OpenCV libraries to implement model training and face recognition functionality.

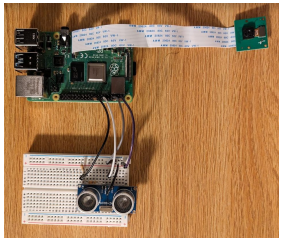


Early Week 10
Using Fusion to design a enclosure and print it

Week 6
Adapting Telegram Bot code to function on Raspberry Pi



Early Week 9
Integrate ultrasonic distance sensor breakout board



Using Swift instead of the Telegram API to make an app that does what the Telegram bot does

Technical Challenges

Telegram Bot

SSL certificate verification errors on UCSD networks prevent the bot from functioning when connected to UCSD WiFi.

solution →

Use a mobile hotspot to bypass this network restriction.

PCB

Overlooked design error resulted in frying ToF sensor with high voltage

solution →

Using an ultrasonic distance sensing module with a breadboard

Technical Challenges

Raspberry Pi

Have trouble setting up the coding environment.

solution

Spent over 20 hours resolving multiple library dependency issues when it came to installing OpenCV and DLib.

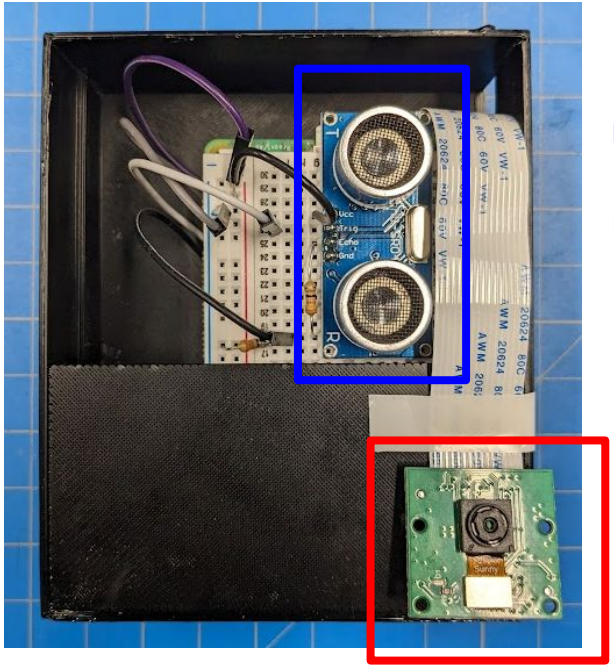
Other

Video frames extracted from video were always rotated due to metadata not being interpreted

solution

Prompt the user to send landscape video.

Prototype: Hardware Aspects



Ultrasonic distance sensor module

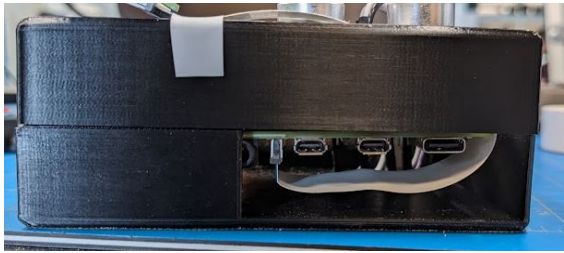
Used to determine subject proximity

Camera Module

Controlled by user through Telegram bot commands

Space-filling Clip

Keeps everything in place



Prototype: Software Aspects

```python
#GPIO Mode (BOARD / BCM)
GPIO.setmode(GPIO.BCM)

#set GPIO Pins
GPIO_TRIGGER = 18
GPIO_ECHO = 24

#set GPIO direction (IN / OUT)
GPIO.setup(GPIO_TRIGGER, GPIO.OUT)
GPIO.setup(GPIO_ECHO, GPIO.IN)

def distance():
    # set Trigger to HIGH
    GPIO.output(GPIO_TRIGGER, True)

    # set Trigger after 0.01ms to LOW
    time.sleep(0.00001)
    GPIO.output(GPIO_TRIGGER, False)

    StartTime = time.time()
    StopTime = time.time()

    # save StartTime
    while GPIO.input(GPIO_ECHO) == 0:
        StartTime = time.time()

    # save time of arrival
    while GPIO.input(GPIO_ECHO) == 1:
        StopTime = time.time()

    # time difference between start and arrival
    TimeElapsed = StopTime - StartTime
    # multiply with the sonic speed (34300 cm/s)
    # and divide by 2, because there and back
    distance = (TimeElapsed * 34300) / 2

    return distance
```

Measure distance

```python
# loop over the facial embeddings
for encoding in encodings:
    # attempt to match each face in the input image to our known
    # encodings
    # matches = face_recognition.compare_faces(data["encodings"], encoding)
    matches = face_recognition.compare_faces(knownEncodings, encoding)
    name = "Unknown" # if face is not recognized, then print Unknown

    # check to see if we have found a match
    if True in matches:
        # find the indexes of all matched faces then initialize a
        # dictionary to count the total number of times each face
        # was matched
        matchedIdxs = [i for (i, b) in enumerate(matches) if b]
        counts = {}

        # loop over the matched indexes and maintain a count for
        # each recognized face
        for i in matchedIdxs:
            name = knownNames[i]
            counts[name] = counts.get(name, 0) + 1

        # determine the recognized face with the largest number
        # of votes (note: in the event of an unlikely tie Python
        # will select first entry in the dictionary)
        name = max(counts, key=counts.get)

        # If someone in your dataset is identified, print their name on the screen
        if currentname != name:
            currentname = name
            # print(currentname)
            message = f"It's {name} at the door."
        else:
            message = "There's an unregistered face at the door."
    else:
        message = "There's an unregistered face at the door."
```

Face recognition with camera

```python
async def open_cam(update: Update, context: ContextTypes.DEFAULT_TYPE):
    global camera_process, flag, message_queue
    flag.value = False
    camera_process = multiprocessing.Process(target=camera_loop, args=(flag, message_queue))
    camera_process.start()
    await update.message.reply_text(f'Camera on. Send `/stop_cam` to stop.')

    # Start a background task to check for messages in the queue
    context.job_queue.run_repeating(check_queue, interval=1, first=0, data=update)
```

```python
async def check_queue(context: ContextTypes.DEFAULT_TYPE):
    global message_queue
    update = context.job.data
    try:
        while True:
            message = message_queue.get_nowait()
            await update.message.reply_text(message)
    except Empty:
        pass
```

any messages placed in the message_queue are periodically checked and sent as replies in the chat

References, Citations

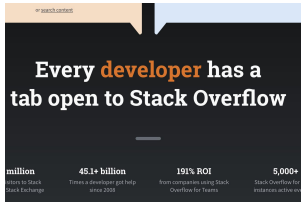
- Face recognition with Raspberry Pi and OpenCV
- **Motivation**: Impaired Facial Recognition and Dementia

IEEE Papers:

- Face Recognition-Based Smart Glass for Alzheimer's Patients
- The Amnesia Atlas. An immersive SenseCam interface as memory-prosthesis
- Selective Amnesia: On Efficient, High-Fidelity and Blind Suppression of Backdoor Effects in Trojaned Machine Learning Models

Helpful sources:

- https://core.telegram.org/bots/api
- https://core.telegram.org/bots/tutorial
- https://picamera.readthedocs.io/en/release-1.13/
- https://stackoverflow.com/
- https://tutorials-raspberrypi.com/raspberry-pi-ultrasonic-sensor-hc-sr04/



Questions?