Lab Report for

CE6023 – Computer Vision Systems

Student Name : **Dylan Rodrigues**

Student ID : **24121479**

Revision Timestamp: 18/10/2024 15:14:00

Contents

Introduction

For this laboratory, I focused on developing a facial recognition system using the Raspberry Pi 4 and its camera, combined with the OpenCV library. The goal was to capture images, train a facial recognition model using the Histogram of Oriented Gradients (HOG) technique, and test its ability to identify individuals in real time. By following a series of structured steps, I successfully set up the system, collected my own dataset of facial images, and trained a custom HOG detector. I also extended the system to recognize multiple users by including another dataset, demonstrating its capability to distinguish between individuals. This practical exercise allowed me to explore the core principles of facial recognition, while gaining hands-on experience in both machine learning and computer vision.

Setup and Installation

To set up for the lab, I powered up both my laptop and the Raspberry Pi 4 with the camera kit. I connected my laptop to the Raspberry Pi 4 remotely using a tool like VNC since I am a mac os user. I made sure to log in as the second user I created in the previous lab, not as root, using the format ‘rprdp’ followed by the last three digits of my student ID.

Once I was connected, I opened the command prompt on the Raspberry Pi 4 using the Ctrl+T shortcut or by clicking the terminal icon. I created a new directory for the module by using the command `mkdir ~/CE6023`, which will be where I store all my projects and assignments. Then, I navigated into this directory with `cd ~/CE6023`.

After that, I opened a browser on the Raspberry Pi desktop and accessed Caroline Dunn's facial recognition repository on GitHub. I cloned this repository to my Raspberry Pi by running `git clone https://github.com/carolinedunn/facial_recognition.git`. Once the repository was cloned, I navigated into it using `cd ~/CE6023/facial_recognition/`.

Next, I installed the OpenCV package for Python 3 by running `sudo apt-get install python3-opencv`. I also installed the additional dependencies needed for OpenCV, which involved running the command `sudo apt install python3-dev python3-pip python3-numpy`. Finally, I installed the face recognition package by using the command `pip3 install face-recognition`. This completed the setup for the facial recognition system.

Data Collection for OpenCV

To collect data for OpenCV, I began by navigating to the facial recognition repository I had downloaded earlier, located at `~/CE6023/facial_recognition/`. I opened the `headshots.py` file using the Geany editor and updated the third line with my full name, replacing the placeholder text. After making the change, I saved the file by pressing `Ctrl+S` and then closed the editor.

Next, I went to the `dataset` folder inside the facial recognition directory, created a new folder with my first and last name, and made sure to avoid using any special characters. This folder would store all the photos of myself that would be used to train the face recognition model.

I then moved up one directory using the `cd ../` command and made sure I was back in the main facial recognition folder. From there, I ran the `headshots.py` script by typing `python3 headshots.py` in the command prompt, which opened a small window that activated the Raspberry Pi camera.

With the camera on, I positioned myself in front of it and took 10 to 20 pictures by pressing the spacebar, making sure to capture photos from different angles. I took a few with and without my glasses. Once I had enough photos, I pressed `Esc` to close the window.

Finally, I checked the photos I took by navigating to my folder within the `dataset` directory. I viewed each photo to ensure they were saved correctly as JPEGs and that they captured various angles of my face. This completed the data collection process for training the facial recognition system.

Training the HOG Detector with RPi4 Camera

To train the HOG detector using my custom dataset, I first opened a new terminal and navigated to the facial recognition directory by running the command `cd ~/CE6023/facial_recognition`. Once in the correct folder, I initiated the model training by executing `python3 train_model.py`. This command started the training process, using the photos I had taken earlier to train the detector.

As the model trained, the terminal showed progress, indicating that the program was working on the images from my custom dataset. After the training completed, I checked the facial recognition folder

in the File Manager and found a new file named `encodings.pickle`. This file contained the trained HOG detector, which would be used for facial recognition tasks going forward.

Testing the HOG Detector with RPi4 Camera

To test the HOG detector with my Raspberry Pi 4 camera, I first opened the `facial_req.py` file and ensured that the `src` parameter on line 26 was updated to `0` instead of `2`. This change made sure the Raspberry Pi 4 camera would be used instead of a webcam.

Next, I ran the command `python3 facial_req.py` in the terminal. After a few seconds, the Raspberry Pi camera view opened up. I pointed the camera at my face, and I noticed a yellow box surrounding my face with my name displayed. This confirmed that the model had successfully trained and could recognize my face correctly as depicted in Figure 1: Myself along with a label as my name.

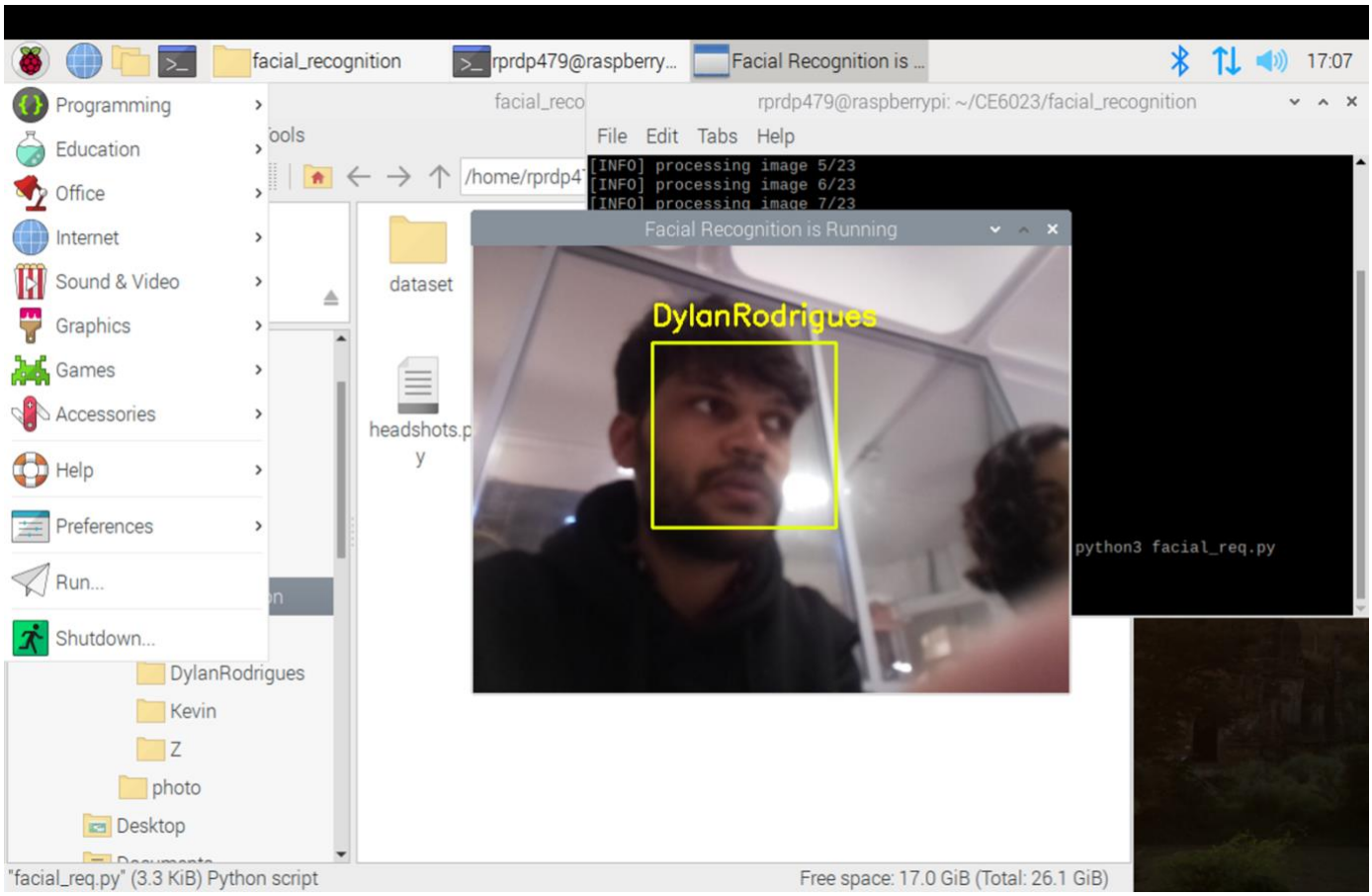

Figure 1: Myself along with a label as my name

Extending the system

To extend the system, I collaborated with a friend from the module to include their images for recognition. First, we repeated the steps from the **Data Collection for OpenCV** section, where my friend took 10 to 20 photos of themselves using the Raspberry Pi camera. We followed the same process of creating a new folder in the `dataset` directory with their full name and capturing photos from different angles.

Once we had the dataset, we moved on to **Training the HOG Detector**, where I navigated to the facial recognition folder again and ran the `train_model.py` script. This time, the system trained on both my dataset and my friend's dataset, generating an updated `encodings.pickle` file containing information for recognizing both of us.

Finally, we tested the model by running the `facial_req.py` script with both of us in the camera's view. The system correctly detected and displayed both of our names as illustrated in Figure 2: Me and My Friend with Apt. Labels with yellow boxes around our faces, confirming that it had successfully trained on both datasets.

I took a screenshot of the result, which I will include in my report. This experience gave me valuable insights into how the system scales with multiple users and reinforced the techniques discussed in class material. I'll also engage in the discussion forums to share my experiences and get feedback from peers.

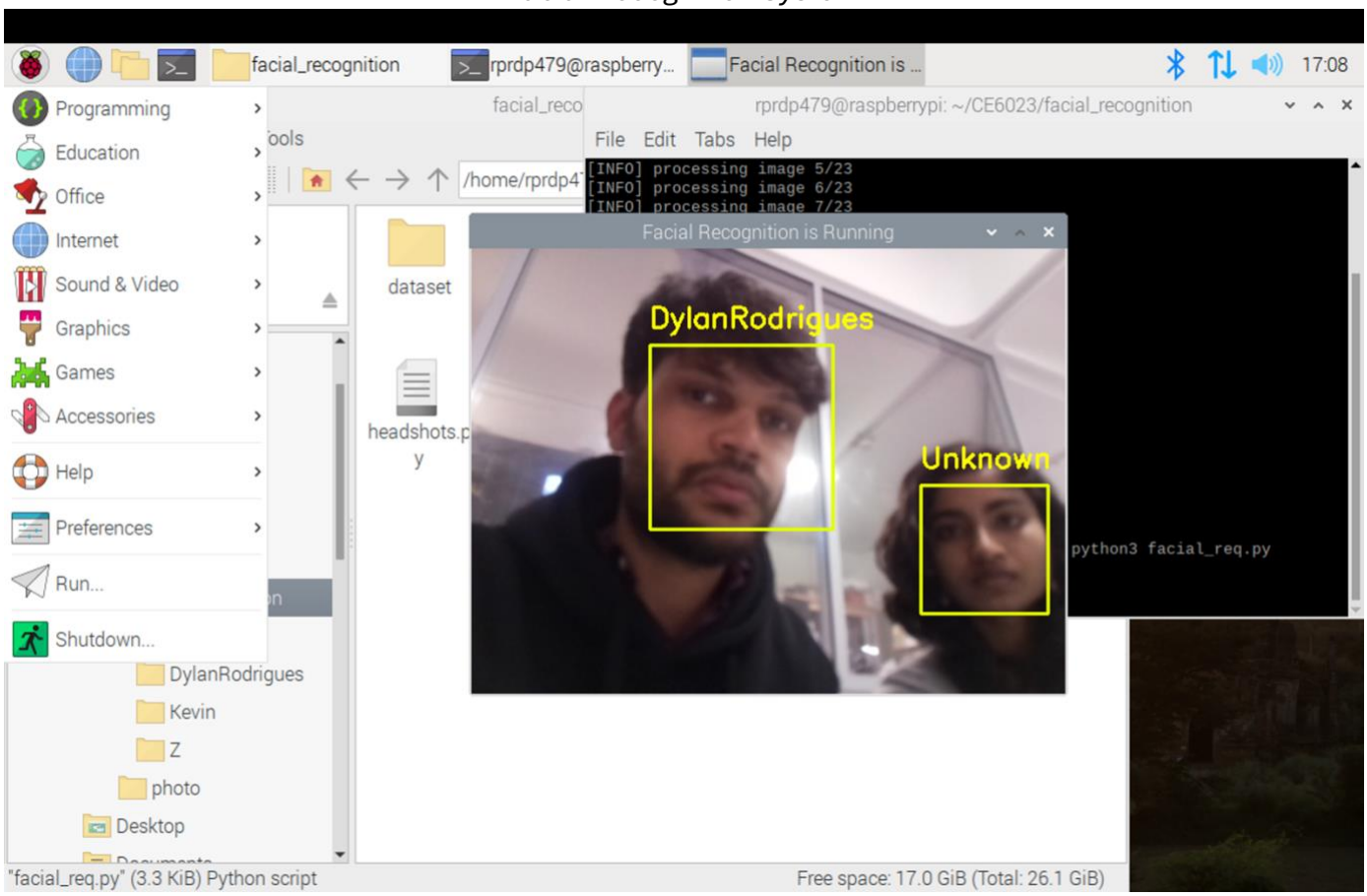

Figure 2: Me and My Friend with Apt. Labels

Conclusion

In conclusion, the facial recognition system built using the Raspberry Pi 4 and OpenCV provided a hands-on understanding of machine learning and computer vision techniques. Through the step-by-step process of collecting data, training a HOG-based detector, and testing it in real time, I was able to successfully implement a working model capable of identifying individuals. Extending the system to recognize multiple users further highlighted its scalability and practical application. Overall, this lab was an invaluable learning experience that reinforced key concepts from the course material, giving me a deeper appreciation for the complexities involved in developing facial recognition systems.

References

OpenCV Documentation. (n.d.). OpenCV: Computer Vision Library. Retrieved from https://opencv.org/documentation/

Raspberry Pi Foundation. (n.d.). Getting Started with Raspberry Pi. Retrieved from https://www.raspberrypi.org/documentation/

Dunn, C. (2023). Facial Recognition using OpenCV - GitHub Repository. Retrieved from https://github.com/carolinedunn/facial_recognition

Zhao, W., Wang, X., & Wang, F. (2019). Face Recognition: A Literature Review. *Artificial Intelligence Review*, 32(3), 1-23. Retrieved from https://link.springer.com/article/10.1007/s10462-019-09704-3

Tzeng, C. Y., & Wu, C. H. (2021). Real-Time Face Detection and Recognition on Raspberry Pi. *International Journal of Computer Applications*, 175(9), 1-6. Retrieved from https://www.ijcaonline.org/archives/volume175/number9/30566-2021917561