Software Formalization

Year: 2019 Semester: Fall Team: 01 Project: IntelliFace

Creation Date: October 1, 2019 Last Modified: December 9, 2019

Author: Abhay Sasidharan Email: asasidh@purdue.edu

Assignment Evaluation:

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Item** | **Score (0-5)** | **Weight** | **Points** | **Notes** |
| **Assignment-Specific Items** | | | | |
| **Third Party Software** | 5 | x2 | 10 |  |
| **Description of Components** | 5 | X3 | 15 |  |
| **Testing Plan** | 5 | x3 | 15 |  |
| **Software Component Diagram** | 5 | x4 | 20 |  |
| **Writing-Specific Items** | | | | |
| **Spelling and Grammar** | 5 | x2 | 10 |  |
| **Formatting and Citations** | 5 | x1 | 5 |  |
| **Figures and Graphs** | 5 | x2 | 10 |  |
| **Technical Writing Style** | 5 | x3 | 15 |  |
| **Total Score** | 100 | | |  |

5: Excellent 4: Good 3: Acceptable 2: Poor 1: Very Poor 0: Not attempted

General Comments:

1.0 Utilization of Third Party Software:

We are not using any third-party software or open-source software on our STM32F0 MCU besides the basic *string, unistd, stdio, and stdlib* libraries provided by STMicroelectronics, and used the ARMV6-M Architecture Reference Manual, along with the STM32F051R8T6 Datasheets for reference on interfacing *GPIO* along with the *USART, ADC and TIM* peripherals.

The libraries used on the Nvidia Jetson Nano have been listed as follows:

|  |  |  |  |
| --- | --- | --- | --- |
| **Name** | **License** | **Description** | **Use** |
| TKinter | BSD License | “The [tkinter](https://docs.python.org/3/library/tkinter.html#module-tkinter) package (“Tk interface”) is the standard Python interface to the Tk GUI toolkit. Both Tk and [tkinter](https://docs.python.org/3/library/tkinter.html#module-tkinter) are available on most Unix platforms, as well as on Windows systems.)” [1] | Used to build and display the Interface dashboard, and run parallel processes to synchronize with the latest updates on news, mails, stocks, and time. |
| OpenCV | 3-clause BSD | “OpenCV is an open source computer vision and machine learning software library. OpenCV was built to provide a common infrastructure for computer vision applications and to accelerate the use of machine perception in the commercial products.” [2] | Used to detect the user’s face using Haar Cascade Filters, isolate the relevant regions, and utilize the Keras network to authenticate the user. |
| Keras | MIT | “Keras is a high-level neural networks API, written in Python and capable of running on top of [TensorFlow](https://github.com/tensorflow/tensorflow), [CNTK](https://github.com/Microsoft/cntk), or [Theano](https://github.com/Theano/Theano). It was developed with a focus on enabling fast experimentation. Being able to go from idea to result with the least possible delay is key to doing good research.” [3] | Used to design and run a lightweight image classifier neural network, seeded with a training set of images of recognized users, and will return the confidence rating given a captured image from the camera using OpenCV. |
| OpenWeatherMap | Creative Commons ShareAlike-4.0 | “Get current weather, daily forecast for 16 days, and 3-hourly forecast 5 days for your city. Helpful stats, graphics, and this day in history charts are available for your reference. Interactive maps show precipitation, clouds, pressure, wind around your location.” [4] | Used to aggregate weather conditions required for the Interface dashboard. |
| NVIDIA Cuda | GNU GPL-3.0 License | “The NVIDIA CUDA Toolkit provides command-line and graphical tools for building, debugging and optimizing the performance of applications accelerated by NVIDIA GPUs, runtime and math libraries.” [5] | Used for image processing and parallel computing acceleration in the OpenCV Haar Cascades. |
| Intrinio SDK | GNU GPL-3.0 License | “The Intrinio API uses HTTPS verbs and a RESTful endpoint structure, which makes it easy to request stock data from Intrinio. Basic Authentication is administered over HTTPS. Responses are delivered in JSON format.” [6] | Required for the Stock buy prices requesting from the TKinter Interface, free in Sandbox mode with a rate limiter on number of requests per minute (5). |
| NumPy | 3-clause BSD | “NumPy is a library for the Python programming language, adding support for large, multi-dimensional arrays and matrices, along with a large collection of high-level mathematical functions to operate on these arrays.” [7] | We are using NumPy for large dataset handling in the neural network and video parser segment of the Interface backend. |

2.0 Description of Software Components:

The software of the Intelligent Interface consists of four primary components, namely the MCU Sensor Data Aggregator, the Facial Recognition Module, the Dashboard Module, and the Video Filter Module, each described below:

**2.1 MCU Sensor Data Aggregator**

The STM32F0 MCU is used to aggregate data from the Infrared, Temperature, and Ambient Light sensors using ADC and Timers, and transmits these to the Jetson Dashboard Module using a USART Channel. Anomalies on the temperature sensor will trigger a PagerDuty API request from the Jetson to the user, alerting them of a potentially dangerous situation. The Light Sensor values gathered using a photoresistor will control the monitor backlight through the Jetson, for low power optimizations. The Infrared Sensor is used to read the authenticated user’s gestures, namely the on/off signal and the dashboard swipe control.

2.2 Dashboard Module

The Dashboard is a Python object oriented designed GUI application using the Tkinter library which will run on the Jetson. The Dashboard will display the welcome message, time, date, day, temperature, location, news, stocks, mail and a randomly generated quote. Various APIs have been used to get the necessary information for the display (for example, BBC news is used as the source for the News API). Facial Recognition will also be merged with the Dashboard module so that the mirror can identify when to and not to invoke the GUI. The Dashboard will be dynamic and is the central core of the Mirror display.

**2.3 Facial Recognition Module**

The facial recognition module was written in Keras adapted from the Inception Convolutional Neural Network model written in Python using the Keras library. It relies on the webcam feed and OpenCV video processing using the Haar Cascade Classifiers to isolate a user’s face from the feed, and feeds the resultant image through the network which yields a similarity confidence level based on a number of trained images in the recognized user list (currently a photo each of the group members, requires data set expansion along with images of course staff). If the confidence level of a certain user’s recognition is high enough, the current bar being set at a minimum of 70% match, the module will pass and allow the dashboard to be opened. Repeated unauthorized accesses will lock the dashboard and notify the users through a Slack Channel or PagerDuty request.

**2.4 Video Filter Module**

The project will also include various additional features to improve the functionality of the product and add more points of interaction between the mirror and the user. The video filtering module will incorporate functions aimed towards the purpose of adding image filters using convolution and Gaussian blurs. The module can be used to adjust the contrast and hue of the images being processed. Moreover, various Snapchat Lens effects can be recreated for the purpose of adding Augmented Reality functionality. This can be done in Python using Dlib, OpenCV and NumPy processing libraries, allowing the mirror to conduct real-time face effects.

3.0 Testing Plan:

To simplify testing on the MCU and the Jetson, we have attempted to divide and conquer each segment as defined below, splitting the MCU component into a Base Board and Sensor Board PCB each:

**3.1 MCU Sensor Data Aggregator**

The most important feature is the MCU sensor data aggregator. This is the main hardware component of the project and it is important that data being interpreted is accurate. The temperature data will be verified with a thermometer. The infrared sensor will be tested by ensuring that it can detect the presence of the user. This is important because it is the starting point of waking up the mirror. In addition, the light sensor will be studied by adjusting the ambient room light and testing to make sure that the screen’s brightness increases in relation to the room brightness.

3.2 Dashboard Module

Testing the Dashboard will be very crucial towards the completion of this project. The UI is going to be the display for each user, and it needs to be tested in all the possible states to make sure there are no unexpected crashes. User experience is very important and one way to test for it is by adding dynamic icons to the interface and seeing how the different possible arrangements fit on the display. Also, when the dashboard module is made to interact with the Facial Recognition module, necessary usability testing can help ensure that the UI for the IntelliFace is error free and that all the users are satisfied with the product.

**3.3 Facial Recognition Module**

The facial recognition module is the third part that needs to be tested. The validity of this module will determine the quality of the user experience. The module is only designed to be functional in well-lit environments. It will be tested by certifying that users can seamlessly register their faces into the system. Moreover, the testing process will also ensure that registered users can use the facial recognition module to login to the system.

**3.4 Video Filter Module**

The fourth most important aspect of the project to be tested is the video filter module. This will be tested by ensuring that the effects in the module are placed in the correct region by affecting the correct pixels in the images. For instance, if a user were to add a filter to his/her face, the coordinates of the face should be recognized accurately, and the appropriate function should affect only the region of the face.

4.0 Sources Cited:

[1] Docs.python.org. (2019). *tkinter — Python interface to Tcl/Tk — Python 3.7.4 documentation*. [online] Available at: https://docs.python.org/3/library/tkinter.html [Accessed 27 Sep. 2019].

[2] Opencv.org. (2019). *About*. [online] Available at: https://opencv.org/about/ [Accessed 27 Sep. 2019].

[3] “Keras: The Python Deep Learning library,” *Home - Keras Documentation*. [Online]. Available: https://keras.io/. [Accessed: 27-Sep-2019].

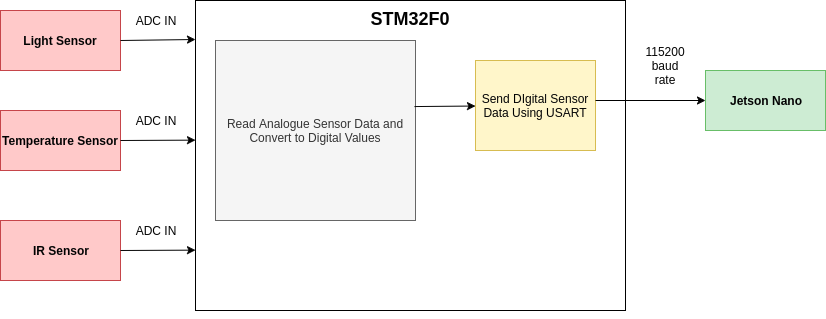
[4] OpenWeatherMap.org, “Сurrent weather and forecast,” *About OpenWeatherMap* . [Online]. Available: https://openweathermap.org/. [Accessed: 27-Sep-2019].

[5] “CUDA Zone,” *NVIDIA Developer*, 12-Sep-2019. [Online]. Available: https://developer.nvidia.com/cuda-zone. [Accessed: 27-Sep-2019].

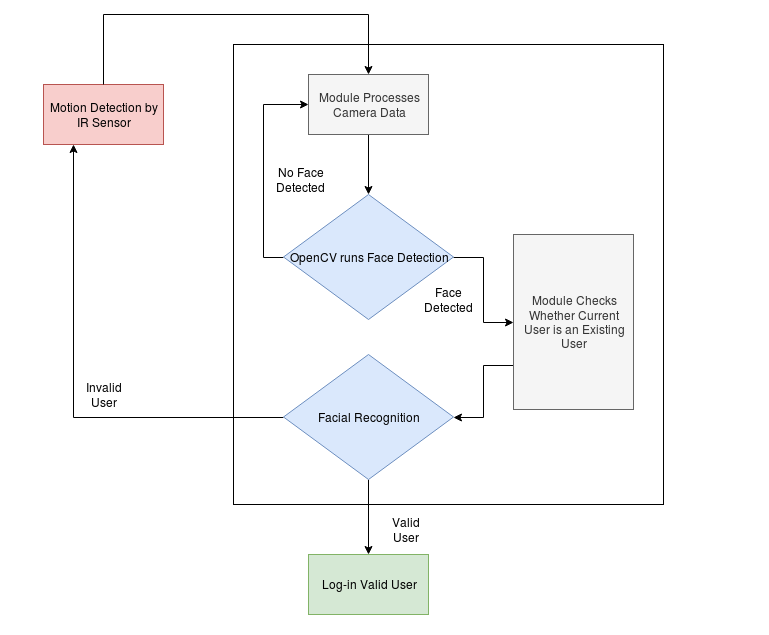
[6] “Terms of Service: Intrinio,” *Intrinio Fintech Marketplace*. [Online]. Available: https://about.intrinio.com/terms. [Accessed: 27-Sep-2019].

[7] “NumPy,” *NumPy*. [Online]. Available: https://numpy.org/. [Accessed: 27-Sep-2019].

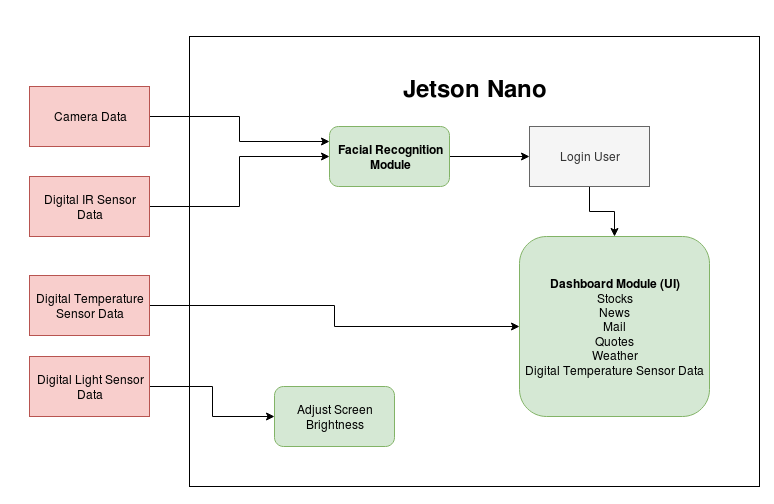
Appendix 1: Software Component Diagrams

**

*Figure 1. Microcontroller Software Flow Diagram*

**

*Figure 2. Facial Recognition Software Flow Diagram*



*Figure 3. Jetson Nano Software Flow Diagram*