CSE 321 Term Project Report Details Fall 2017

1. Executive Summary

Face-ID is a software that allow users to capture a face and analyze with its database to determine if the captured face is in the system or not. The software is not build for commercial use but instead it can be used for education. The software showcase how you can use simple math with pixels to create a very basic facial identification product using processing i3 language. The open source code uses little to no library with easy to understand data structure targeting beginner coder. The project is a real time/embedded system because it use RPi3 with functionality performing at close to real time.

2. Requirements

Raspberry Pi 3, Processing i3 and USB Webcam are three minimal product you need to get started.

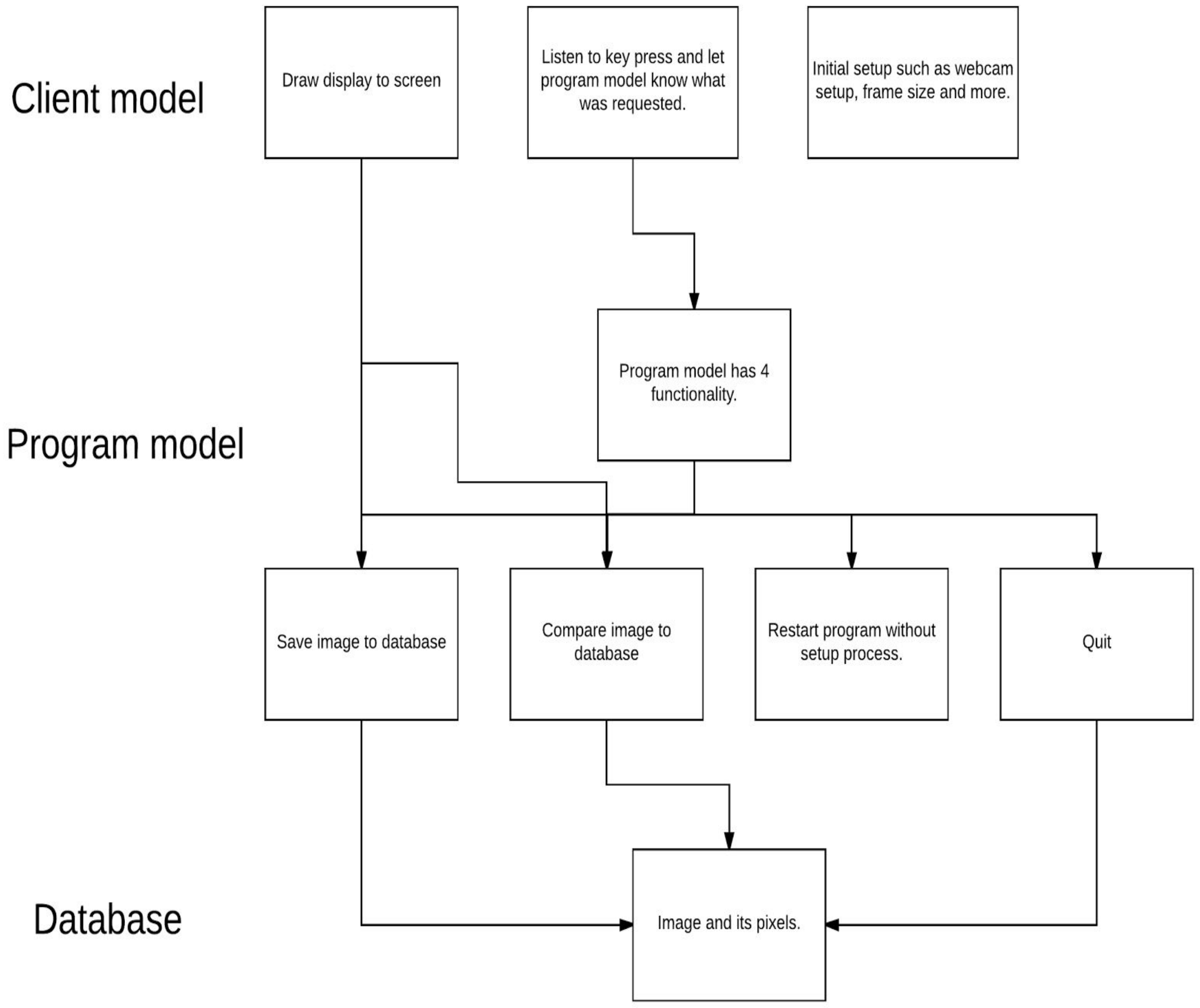
- Raspberry Pi 3 - You can can get them online by searching for Raspberry Pi Model B. You also need Raspbian or Noobs install on your RPi3 where you can download on their official website.
- Processing i3 - This is the language we use for our Face-ID software. It is similar to Java and you can download it on their official website with Linux Armv6hf version.
- USB Webcam - Webcam is use to capture faces to analyze with the database. You can get them online or an electronic shop and configure RPi3 to recognize it.

3. Project Approach

We develop the project in a familiar language and easy to use PDE to target beginner coders. Processing i3 is a PDE (processing development environment) software that is fitting for people who are into graphics but might not be very good coder. The PDE uses Java as its language which is a common language to start as for a beginner and simple game design. The program was also developed to be user friendly where functionality are separated function such as comparing image will use read, calculate, compare and more functions.

4. Project Description

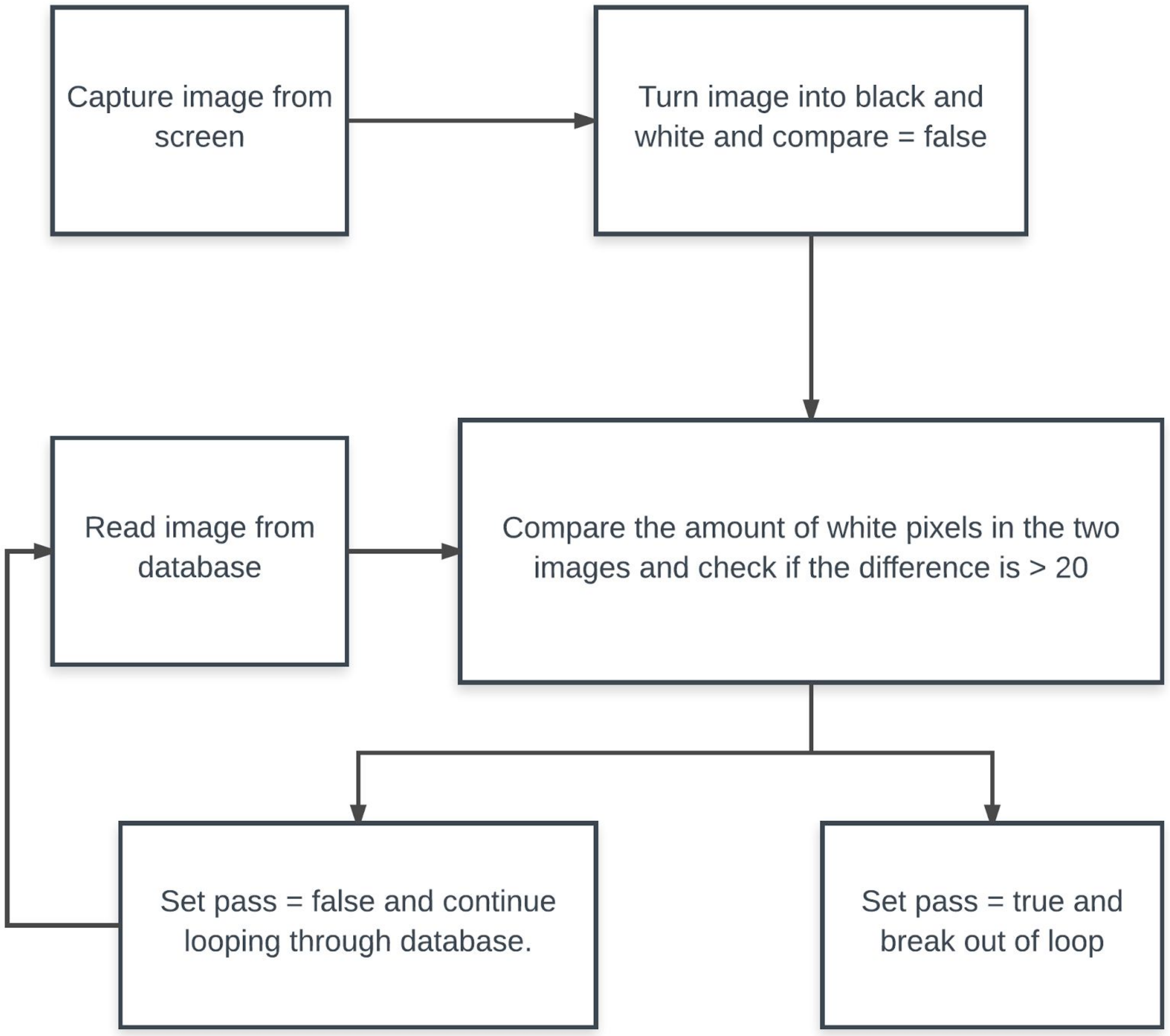
There are 3 model in our project, client, program and database. The client model is simply an interface for the user (GUI). The program is where the processing happen when user interact with the software. The program takes in user input and process them and tell what the client model should display.The database is use when user wants to save the image captured or compare image we currently have with database image.



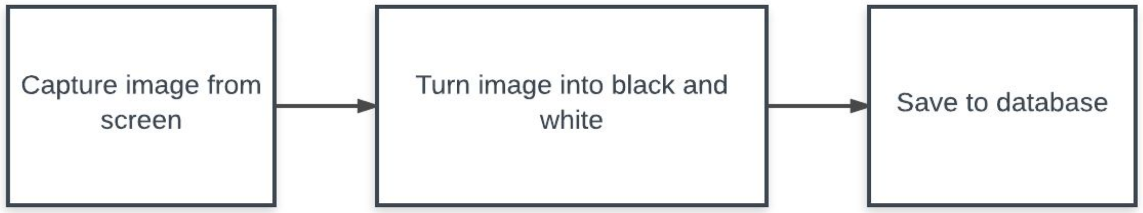
Client model
Draw display to screen
Listen to key press and let program model know what was requested.
Initial setup such as webcam setup, frame size and more.
Program model
Program model has 4 functionality.
Save image to database
Compare image to database
Restart program without setup process.
Quit
Database
Image and its pixels.

5. Design Details

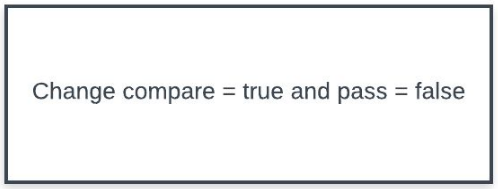
Compare:


Capture image from screen
Turn image into black and white and compare = false
Read image from database
Compare the amount of white pixels in the two images and check if the difference is > 20
Set pass = false and continue looping through database.
Set pass = true and break out of loop

Save:

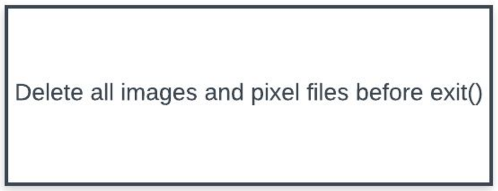


Restart:



By changing compare = true, pass = false it changes how draw() function work.

Quit:



6. User's Manual

Set-up

- Install processing i3 (ARMv6hf) at https://processing.org/download/
- Follow instruction at https://processing.org/tutorials/gettingstarted/ to untar and run processing on Raspberry pi 3
- Clone our repository and open "face_id" from processing to start coding or run.
- To make it an executable, you have to export pde and choose linux version.

Functionality

- The current version provide "Save", "Restart", "Compare", "Quit"
- To save your face to the "database" simply press 's' or 'S' when running the program.
- To restart the face id process simply press 'r' or 'R' when running the program.
- To compare your face to the "database" simply press SPACEBAR when running the program.

7. Programmer's Manual

Set-up

- Install processing i3 (ARMv6hf) at https://processing.org/download/
- Follow instruction at https://processing.org/tutorials/gettingstarted/ to untar and run processing on Raspberry pi 3
- Clone our repository and open "face_id" from processing to start coding or run.
- To make it an executable, you have to export pde and choose linux version.

Functionality description

setup:

setup() uses gohai library in order to enable usb webcam. It sets up the size of the display and search for webcam devices. It also configure the first devices and use the device that has 640*480 with rate of 30fps.

key pressed :

The valid keys are **s**, **r**, **spacebar**, **q** where lower or upper cases are valid.
'**S**' represent save and it takes a picture within the box display and change the image into black and white. It also calculate the amount of white pixels in each row and save them as Image#.jpg and Image#.txt.
'**R**' represent restarting without running through setup. Setup takes a long time to run therefore by pressing 'r' we just change how draw function work by setting to its beginning state with compare = true, pass = false.
'**Spacebar**' is the key for comparing current image capture to image in database. It first set compare = false and capture the image within the box, turns it into black and white. Then it loop through the database images and compare the total white pixels in the current image and the database images. If difference is less than 20 it will set pass = true and break out of the loop. If difference is greater than 20, it will keep finding unless if its end of database, it will set pass = false. This will change how draw display to user.
'**Q**' is the quit key when you are done with the program. When you quit the program, it delete all images and text files that was saved while the program was running.

draw:

The draw function is processing what is display on the screen for the user. It display different scenes based on two boolean variable compare and pass.
If **compare = true**, it will display the initial scene where a box appear in the middle of the frame with what is recording from the webcam.
If **compare = false** and **pass = false** it will display a "denied" red box letting the user know that the person is not in the database.
If **compare = false** and **pass = true** it will display a "pass" green box letting the user know that the person is in the database.

calculateImage:

This function calculate the amount of white pixels in each row of an image and append them into a global int array (**white_pixel**). This function is use when '**spacebar**' and '**s**' is press because in compare it needs two array of white pixels to compare and in save it needs to save the values of the white pixels array in a text file.

readImageData:

This function calculate the amount of white pixels in each row of an image in the database and append them into a global int array (**dbWhitePixel**). This function is use when ‘**spacebar**’ is press because in compare it needs two array of white pixels to compare (**dbWhitePixel** and **white_pixel**) .

saveImageData:

This function creates a new text file with the name base on a variable **dfilenum** which is a variable that keep counts of how many files are in the database. After creating the file it print all the value of **white_pixel** array into the file then close it after finish printing.

compareImageData:

This function calculate the total white pixel value of two image (database and current) by adding all value of **white_pixel** and save it in float **cValue** and adding all value of **dbWhitePixel** and save it in float **dValue**. After adding all white pixels value of both image it will calculate the difference between **cValue** and **dValue** then save the percentage difference to a global float variable **matchP.**

blackwhite:

This function changes the color of the image capture to black and white. It uses threshold of float value 160 and test them against each pixel in the image. If pixel is greater than threshold, it change the pixel value into white else change the pixel value into black.

8. References:

Processing reference:

https://processing.org/reference

GL video usb webcam setup for raspberry pi 3:

https://github.com/gohai/processing-glvideo/blob/master/examples/SimpleCapture/SimpleCapture.pde