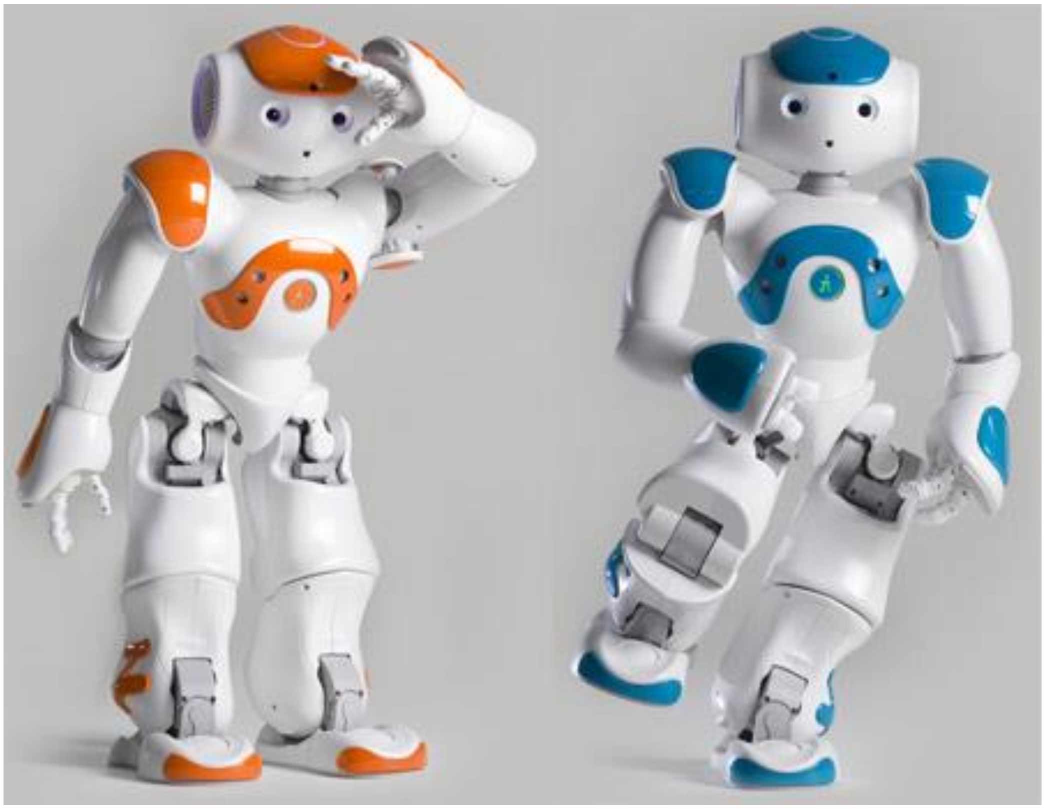


Implementing the Face Detection Authentication in the NAO Robot

Kristi Prifti | Krutarth Patel | Hue Do

Faculty Advisor: Hao Loi

Department of Computer Science and Engineering, Quinsigamond Community College

Course Title: Introduction to Software Engineering

Abstract of the Project

The NAO robot has a wide range of applications to automate many of our daily tasks. These robots are used for many different purposes. Shortly, they will become an essential education element. The NAO robot has built-in cameras that could be used to recognize faces. Our project is to use the NAO robot to take the attendance of students of any class using the NAO robot. This project will have two different methods including several steps. First, the NAO robot will learn different faces. These faces will be saved in its memory. Programs based on python language will be created to compare the faces from the memory of the NAO with the faces of the students present in the class. This can be done also by using Choregraphe software. The second method is to compare pictures to recognize faces. The NAO robot will capture a picture of a person and compare it with the pictures existing in our machine. After training NAO, a webpage will be build based on the Blackboard interface. The main purpose of this webpage is to keep the attendance report of all students. These two different methods will take attendance and will store the data on one common webpage. This project will be based on the Python programming language and Choregraphe software. The application of this project will have potential beyond the classroom. Eventually, it could be programmed to use in hotels, hospitals, airports, at work, etc.

Introduction to the NAO Robot

NAO robot is the first robot created by the SoftBank Robotics. It is well-known around the world and it is a great programming tool. Shortly, he will become a standard element in education and research centers. The NAO robot is also used as an assistant by corporations and healthcare centers to greet, inform, and entertain guests. This robot is built-in with very great features and hardware. Some of them are as follows:

- 25 degrees of freedom which enables him to move and adapt to his environment.
- 7 touch sensors located on the top , hands, and feet, sonars, and an inertial unit to perceive his environment and locate himself in space.
- 4 directional microphones and speakers to interact with humans.
- Speech recognition and dialogue are available in 20 languages including English, French, Spanish, German, Arabic, and many more.
- Open and fully programmable platform.
- Two 2D cameras to acknowledge shapes, objects, and even people [1].

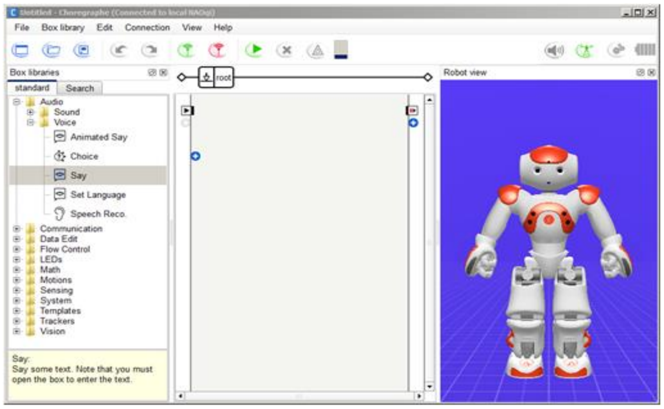
The NAO robot has two built-in cameras. Using this cameras NAO can learn and recognizes objects, paths, and faces of humans.

Starting Our Project

Our project starts by training the NAO to learn face. By researching on the NAO Software 1.14.5 documentation, we found that this task can be performed in two ways.

- Using Choregraphe software
- By creating a program that recognize faces by comparing pictures of faces

Firstly, we started by using Choregraphe on the base of the documentation provided. After connecting the NAO to our machine, we can start exploring on Choregraphe. There are different types of behaviors available in boxes libraries [2]. One of them is Learn Face Box. We run that box in the flow diagram panel. While running, its askes the string from the user. Here, we can type the name of the person of whom the face is been learned. The NAO will memorize this face and the string provided. This is how we made the NAO to learn face with the help of Choregraphe Software.



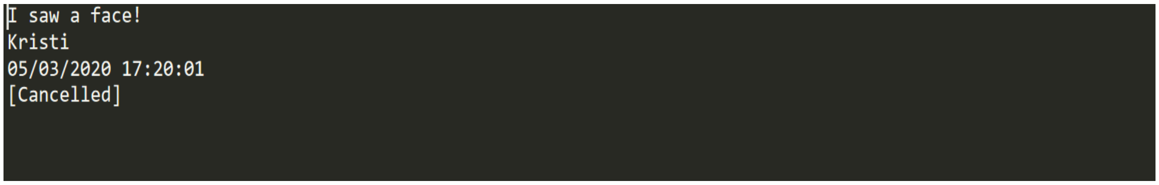
Say:
Say some text. Note that you must open the box to enter the text.

The another way to train NAO to learn face is by creating a python script. This can be done with the help of the AlFaceDetection API and the AlFaceDetection tutorial. In this script, we must import time and AlProxy from NAOqi. This tutorial explains how to run the ALFaceDetection module using Python. We use the following approach: we regularly check the ALMemory's result variable. Information about the detected faces is printed on the screen [3]. Comparing both methods, we found that Choregraphe software is more effective and less time-consuming. So, we will use Choregraphe in our project to train NAO to learn faces.

Face Recognition Using Choregraphe Software

The NAO robot is learning faces and storing them in its memory. Our next task is to compare the faces of the students with the faces in the memory. We can perform by creating a python script named as face_recognition_using_choregraphe.py. This program is created with the help of the Softbank Robotics Documentation, NAOqi 2.5 [4]. This python program is using NAOqi as a library. The class is created named attendanceTaker, which includes various functions. The particularly important function of this class is on_human_tracked function. When any face is detected, the Boolean present in the function changes to true and it calls the memory of NAO. If the robot had learned that face before using Choregraphe, then it will able to recognize that face, and the program will execute the print command. Else it will ask to detect the face again.

This function can generate a text file or can directly uploads the report on the webpage. When any face is detected and matched with faces learned by NAO, then the person's information will be saved in this file with time and class information. This can also be modified as needed. The file generated will later use by the webpage. All the names of the student in this file will be marked as present on the webpage and thus the attendance report will be stored in the database of the website.



```
I saw a face!
Kristi
05/03/2020 17:20:01
[Cancelled]
```

When the program is run, and the NAO recognizes the face, this is how the output looks like in the terminal. It includes name, date, and time. The same information is saved in the file created.

Using NAO's built-in camera to take pictures

Our group performed face recognition through the software. There is not just one way to perform this project. Hence, we started researching for another way that could be less time-consuming as well as more efficient. While researching, we found that the other way to implement face recognition is by comparing pictures of a person's face. Let us go in for more detail.

The first most step for this method was to make NAO take pictures. We can use Choregraphe here as well. But that will only take pictures and after it stops running. Our aim was to capture a picture and compare it with the pictures already existing in our machine. Thus, Choregraphe was not a good idea. We need to create an algorithm based on python language that can capture pictures using NAO's built-in cameras. We referred to Aldebaran Documentation [6] and created a program named Use_NAO_Camera_To_Take_Pics.py. It consists of one main function which performs the task of capturing a picture. It takes a picture by calling getImageRemote function, converts it in PNG file using ImageDraw package, and saves it in our machine.

We run many tests on this program and from them we concluded that these pictures have low image size and good pixel quality.

Face Recognition Using Pictures

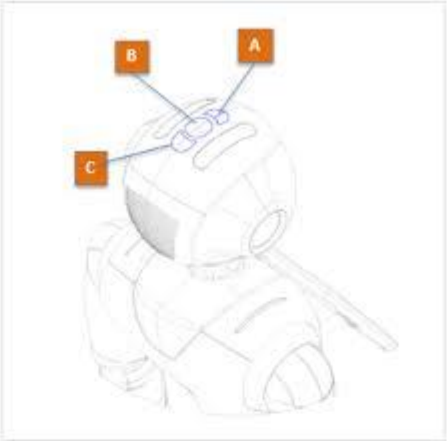
Moving further in performing face recognition using the method of comparing pictures, we will continue by adding more functions to our algorithm. This python code is saved and named as face_recognition_module.py. We created a folder named ‘Faces’, that consists of pictures of faces of different people. The picture that is captured by using the Use_NAO_Camera_To_Take_Pics.py program is named by the name of the person. Let us get through functions of the face recognition module algorithm.

After capturing a picture, the classify_faces(IMG) function is called. This function calls get_encoded_faces(), which enters into the Face folder and starts encoding all the faces one after the other in this folder. The classify_faces(IMG) function compares the picture captured by the NAO with the picture encoded from the Faces folder. If the picture consists of multiple faces, then the program tries to recognize all the faces. If it matches any face, then with the help of coordinates, it draws the box around the person’s face with the name written below of that box. The next step is taking the name, course name, date and time of class, and saving all information in a text file. Sometimes the program fails to recognize the face. At that time, it prints a message saying, “I don’t recognize you.”

At the end of this algorithm, the text file generated is sent to the webpage with all students' information. This webpage sorts all the student's information accordingly and displays the attendance report in an organized manner. If the student is recognized that implies, he is present in the class. So, the attendance report shows that students as present in class on that date and time.

Using Touch Sensors to Run the Programs

After the completion of creating algorithms of both methods, we started to modify it for more improvement. We referred to NAO Software Documentation [7] to get more details of the NAO robot. As we mentioned that a NAO robot has seven touch sensors located on different parts of the body, we thought to utilize them. Firstly, starting with the touch sensor on the top of head of the robot, an algorithm named as Start_attendence_with_back_click_of_naos_head.py is created that calls that sensor.



There is one important function called onTouched(value) in this algorithm. As shown in the introduction that NAO robot has many sensors. These sensors aim to determine whether the robot is touched. It is a Boolean behavior, an event is raised when for example the head is touched the value will be True, and when the head is touched again the value will become False. So, whenever the touch is detected on NAO's head, the program will start running in our machine. This makes attendance taking easier. This function is based on AlTouch from Aldebaran Documentation [5].

Attendance Report Webpage

As said this project has two different methods. First is by using Choregraphe software and second one by using pictures. Both this method creates a text file after the are compiled. This file is then uploaded automatically on a webpage. The name of this webpage is given as Attendance Report Webpage. Couple of the files are mapped around to make this webpage. They are described below:

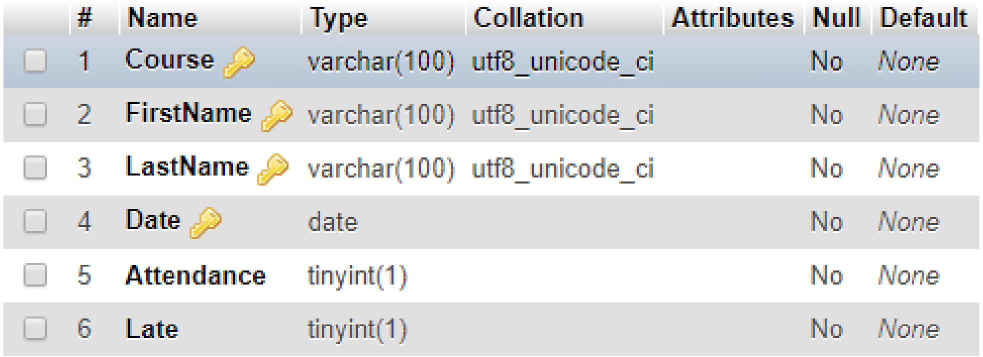
Webhost

- We approached to 000webhost for hosting our webpage and started working on the Attendance Report webpage that supports PHP, and MySQL.

Database

- Firstly, we need to prepare a database to store the data in, which we receive from the NAO robot. The phpMyAdmin in MySQL is used to create the database.
- There are six columns created to store the information of Course name, First Name,

Last Name, Date, Attendance, Late comers. Structure of the table is shown below:



	#	Name	Type	Collation	Attributes	Null	Default
□	1	Course 🔑	varchar(100)	utf8_unicode_ci		No	*None*
□	2	FirstName 🔑	varchar(100)	utf8_unicode_ci		No	*None*
□	3	LastName 🔑	varchar(100)	utf8_unicode_ci		No	*None*
□	4	Date 🔑	date			No	*None*
□	5	Attendance	tinyint(1)			No	*None*
□	6	Late	tinyint(1)			No	*None*

- The Course Name, First Name, and Last Name are set to VARCHAR (100) to allow them to secure a good deal of characters. The reason we use VARCHAR for all these domains is that it allows a variable amount of characters. Date has the type of date and that of domains like Attendance and late are Boolean, which shows 1 if it's true, 0 if it's false.

HTML Code

- Index.html:

  This is the first level having an interface of the website that recites and inserts the information of students into the database. As robot can send the file directly to the database, but if the robot misses ant student or is not performing the tasks, then one can manually upload the attendance file to the database with the help of this interface. We set the action attribute to the file update.php. When the Submit tab is clicked, the report is passed to update.php file.

- **PHP Code:**

  Two PHP files are created to support the data from Index.html:

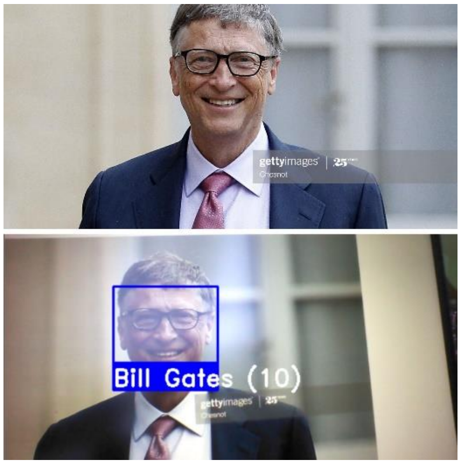
  1. Update.php: This file transfers the report to MySQL table to store in the memory. MySQL is connected to this file with the help of mysql_connect function.
  2. Report.php: The main purpose of this file is to take the data from the memory and display it on the main output page. To display all data in one table, mysqli_fetch_assoc() function is used that returns an associative array of strings representing the fetched row. A loop is being functioned here to iterate through each row. A filter tab is also created that allows users to search for a specific name, class, or date.

Results from our Testing face_recognition_module.py

We performed 20 different tests with different faces. The result of five of them are shown below.

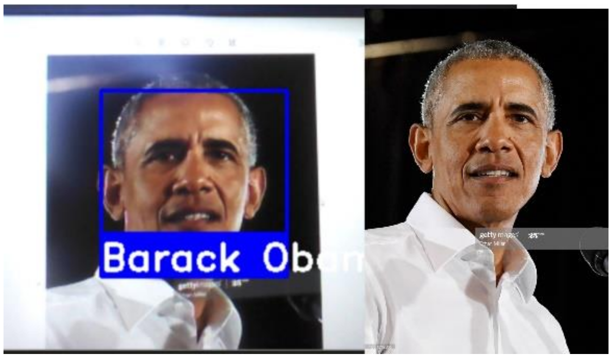
- Face of Bill gates:

  Result: Passed



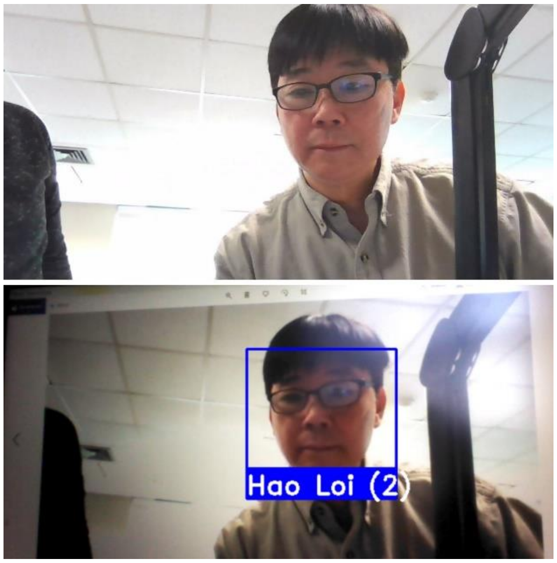
- Face of Ex-President Obama:

  Result: Passed



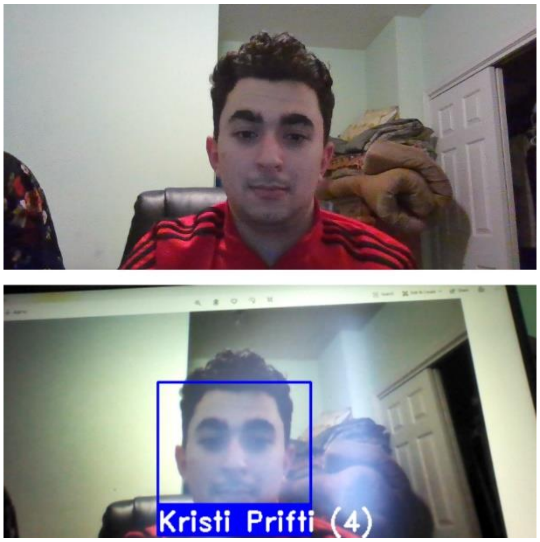
- Face of Professor Loi:

  Result: Passed



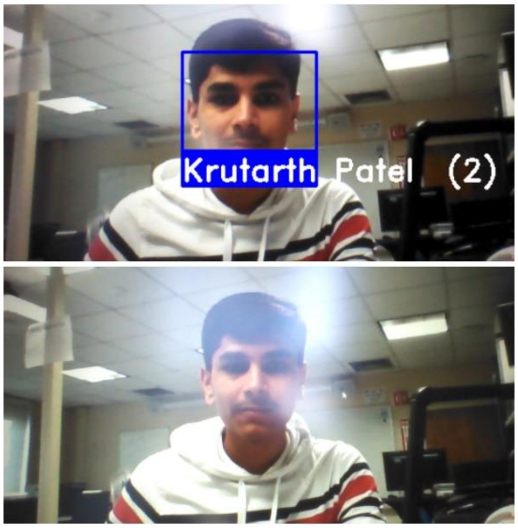
- Face of Kristi:

  Result: Passed

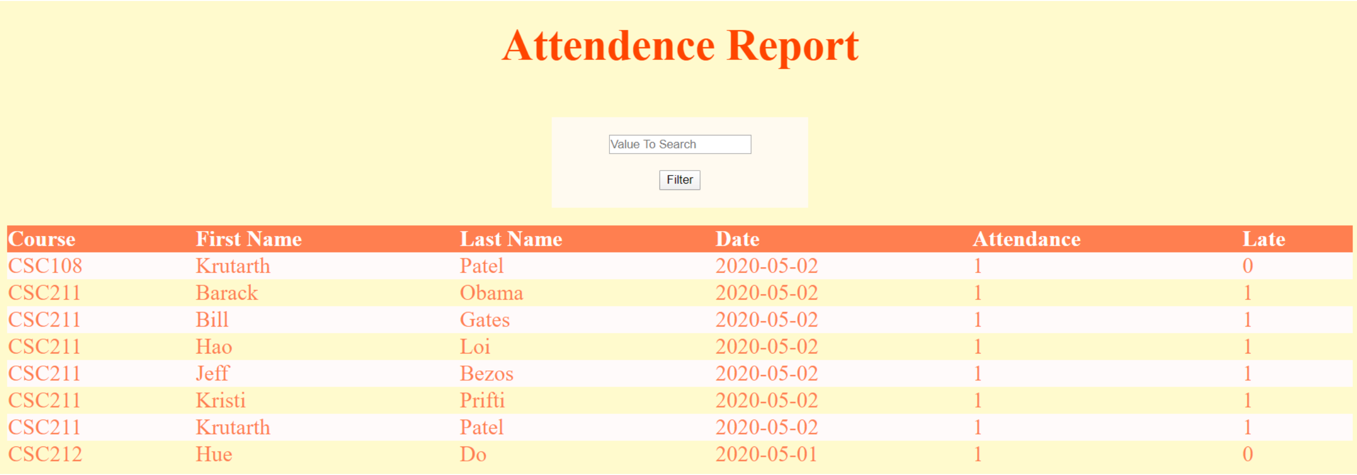


- Face of Krutarth:

  Result: Passed



If their good light condition and proper angle, then it the probability for getting fail is very low. An example of result on the attendance report webpage is shown below:



Attendence Report

Course	First Name	Last Name	Date	Attendance	Late
CSC108	Krutarth	Patel	2020-05-02	1	0
CSC211	Barack	Obama	2020-05-02	1	1
CSC211	Bill	Gates	2020-05-02	1	1
CSC211	Hao	Loi	2020-05-02	1	1
CSC211	Jeff	Bezos	2020-05-02	1	1
CSC211	Kristi	Prifti	2020-05-02	1	1
CSC211	Krutarth	Patel	2020-05-02	1	1
CSC212	Hue	Do	2020-05-01	1	0

Works Cited

1. Softbank Robotics. https://www.softbankrobotics.com/emea/en/nao
2. NAO Software 1.14.5 Documentation, Choregraphe Software. http://doc.aldebaran.com/1-14/software/choregraphe/interface.html
3. AlFaceDetection API and AlFaceDetection Tutorial Documentation. http://doc.aldebaran.com/1-14/naoqi/vision/alfacedetection-tuto.html
4. Softbank Robotics Documentation used for creating face recognition program. http://doc.aldebaran.com/2-5/dev/python/reacting_to_events.html
5. Aldebaran Documentation for touch sensors on the NAO robot. http://doc.aldebaran.com/2-1/naoqi/sensors/altouch.html
6. Aldebaran Documentation for making NAO robot take pictures. http://doc.aldebaran.com/2-4/dev/python/examples/vision/get_image.html
7. NAO Software Documentation for touch sensors on NAO robot. http://doc.aldebaran.com/1-14/family/nao_h25/contact-sensors_h25.html#h25-contact-sensors