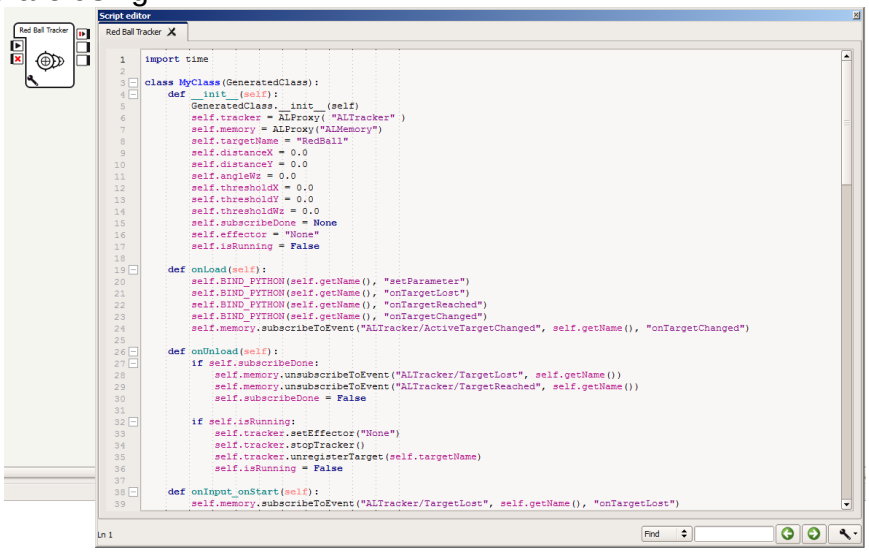
CP/PC 497
Working with Nao and 3D printing

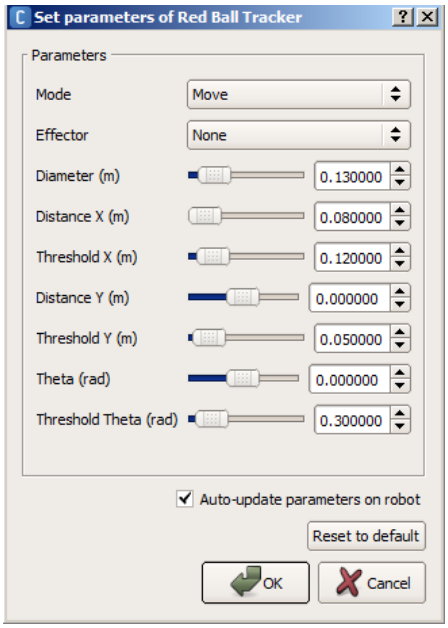
Michael Murray, Wesley Crick, Matt Erickson, Alex Carinci, Joe Staller

Block Programing

Nao is a basic artificial intelligence robot that can be programmed to do a wide variety of tasks. Nao uses block programming where each block contains hard python code within. These codes can be manipulated if need be by simply double clicking the block code you are using.



However, you can set the codes to specific needs by clicking the wrench in the bottom left corner of the box. This brings up different variables depending on each box's output. For the same block as above (Red Ball Tracker) the wrench would display;



By switching these values, you can have Nao simply look at a ball and follow it; he can walk toward the ball until a certain distance which he would then stop. It is a personal preference on what you wish Nao to accomplish and dependent on the size of the ball, the numbers will change. This is a specific case towards the Red Ball Tracker, but the same explanation applies to other blocks. Also, by holding the mouse over the bar it will tell you more specifically what that value does.

The program we use to access Nao is Choregraphe. It comes with lots of pre coded blocks in which you can combine to have Nao do a routine, walk somewhere, track a ball to recognizing human faces. For more information on how to program Nao with python, read the pdf document “Programming Nao using Python”.

However, you don’t need to rely on these programs alone. You can create your own position simply by two methods;

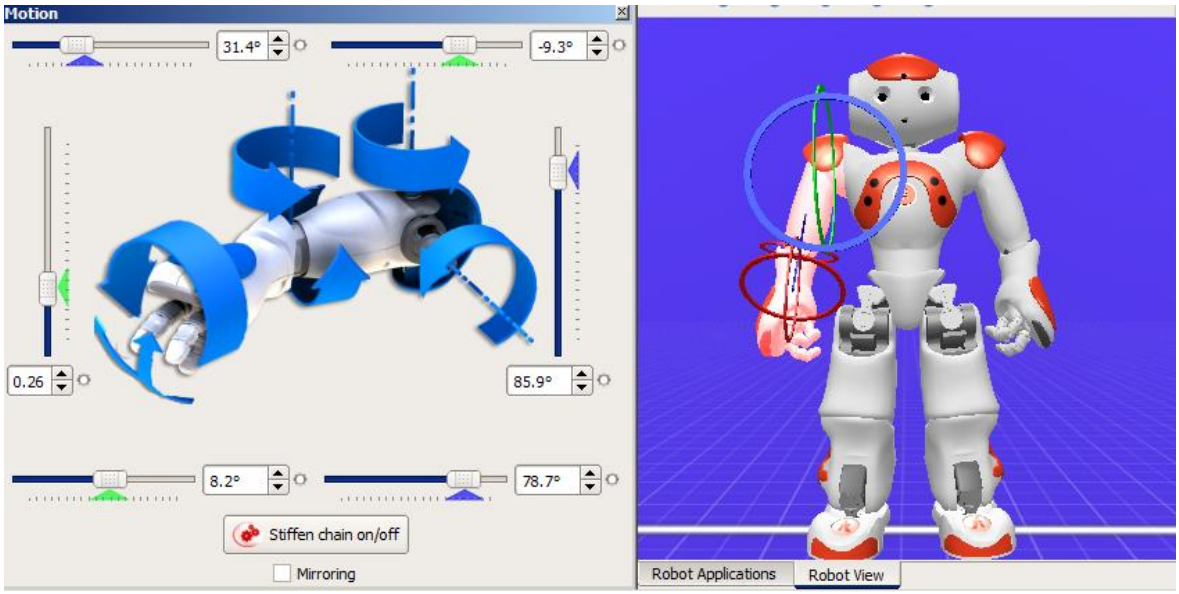
Animation mode:

- For this mode, you would want to turn off autonomous life so Nao stays still.
- Next, click the green icon of the person like figure. This will allow you to move his joints freely.
-

- You will see his eyes turn green which means he is loose.
- Now you will be able to move his head, arms, legs etc. by manually moving them yourself.
- Once you have your position, click the gear box beside the animation mode so it turns red. His eyes should change colour telling you he is stiff and won’t move.
- Save his position by going into “Pose Library” and clicking on the Add position. Name your position and then drag it into the template to have Nao access that position.

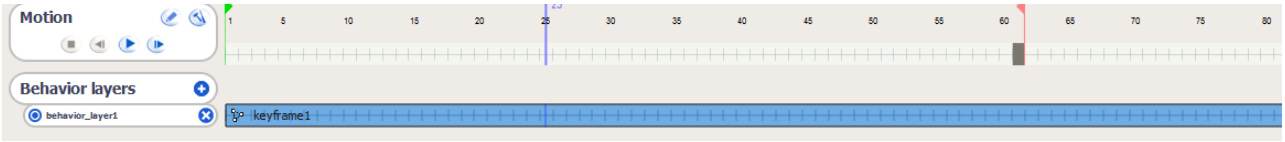
Motion Position window:

- This position allows you to get Nao in a symmetrical position.
- Start by turning off autonomous mode.
- Go into the robot view where you can see Nao on the screen.
- By clicking his body parts, a window comes up with multiple values and arrows.
    - Each bar pivots a specific joint in Nao’s arm for example;



  - The way this is better is that you can click the mirror box which will move both Nao's arms into the same location. This gives him a nice symmetrical look instead of estimating it in animation mode.
- Make sure someone watches Nao and is ready to catch him while another tampers with him in the position window. He could take a nasty fall.
- Save Nao's position by the same method as above.

After making these positions, double check that Nao won't perform them too quickly causing him to lose balance. Simply double click on the block you just created and move the red flag past the blue line. Start with it way past it and move it in closer, slowly picking up the pace until you find your perfect speed.



Problems:

Throughout the two terms, Nao slowly began to what we thought mechanically break down. When we say this, he tended to overheat quickly causing him to not complete tasks we asked him to. He would have trouble walking a straight line, and wouldn't be able to hold positions (Would not be able to keep his arm up; it would always fall back down). He would often need to be turned off after 30 minutes of working with him to cool down. It became very difficult to work with him, for he would not be able to do his normal capable movements.

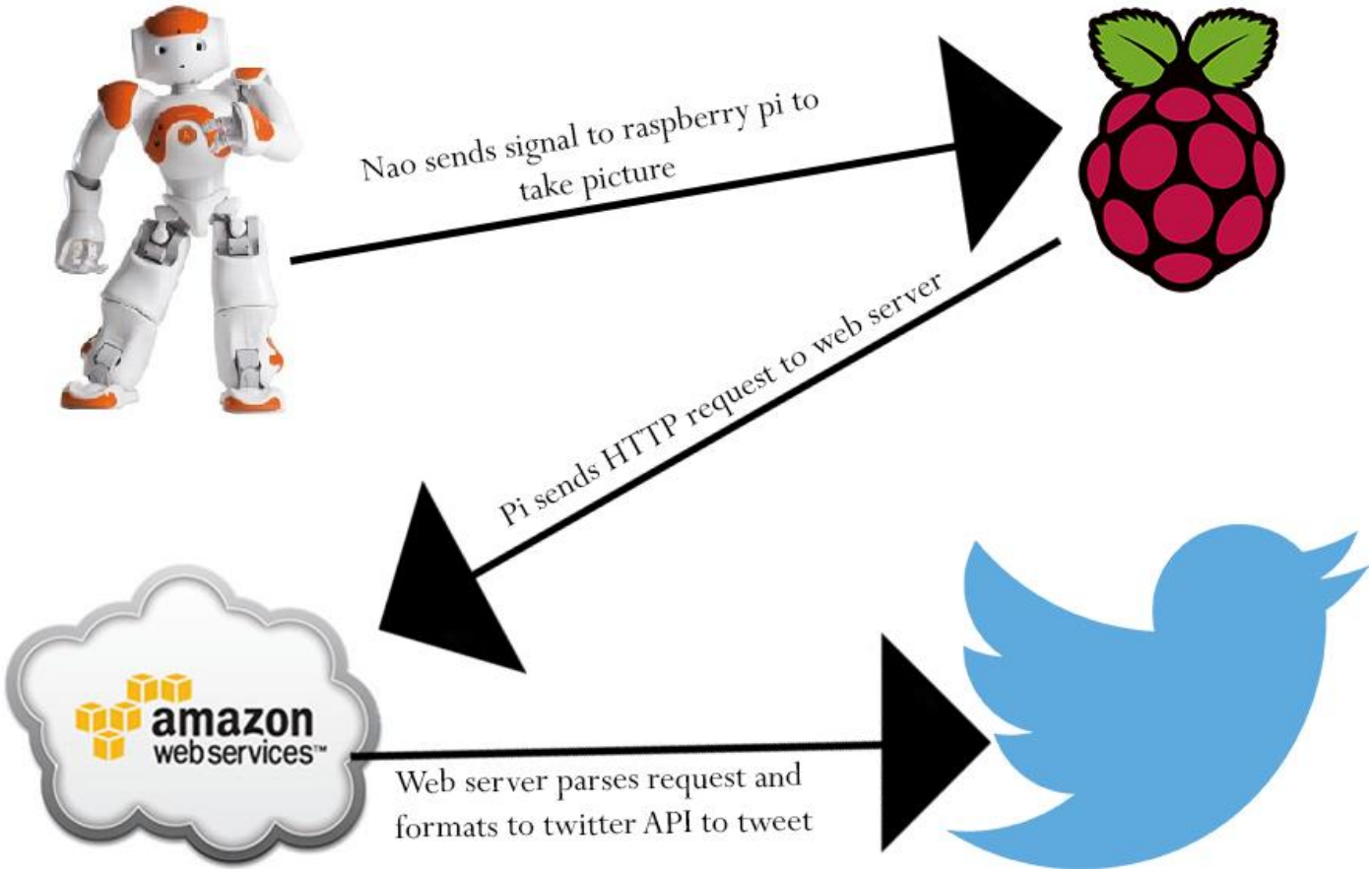
His routines would be affected as well. If he had been turned on for a while, he would not fully complete his routines. Meaning, he would not walk a certain distance requested or he would do a lazy position of what you requested. He also became very twitchy, causing him to follow over more and look less fluent.

To counter act this, we had to put in y values to hold him to the right more which in outcome, made him walk straight. If Nao was to be fixed up, these problems would not arise, at least not nearly as often. This would allow for more work sessions with him and less frustration waiting for him too cool down. Nao capabilities would be drastically easier to work with and could accomplish much more in shorter periods of time.

Selfie and Twitter

Last semester we had Nao tweeting from his camera, this semester we moved that code off of Nao and onto the raspberry pi so that we could take selfies with Nao. The process is generally the same with a slight exception of the Nao program. When the program starts, Nao will do a countdown, move the camera into position, and then send a signal to the pi to take the picture. The pi will then create the HTTP request and send it to an external server. The server is hosting a php file, in an Apache web server, which uses the Twitter API to tweet a message and an image.

Source Code:
https://github.com/cric5890/Nao-OCR/tree/master/Organized/Twitter%20and%20Selfie



Tour

Our tour program has all the software components to work, however we lack the hardware to complete it properly. The idea is to have Nao walk through the hallway and when he encounters a room number he recognizes, he will then give a little description about the room. We first wanted Nao to take the images and then do the image processing, but he was too slow to be able to make it look natural. To fix this we attached a raspberry pi to Nao so that he wouldn't have to worry about the images. All Nao would receive is a string of what to say, from a socket. The raspberry pi camera is a 1080p 5MP camera, which takes good pictures, however it has no stability in it. When Nao would walk and try to take pictures they would be extremely blurry. If we had stability for the camera then this could easily work, with the only possible downside is that Nao is a slow walker.

Source Code:
https://github.com/cric5890/Nao-OCR/tree/master/Organized/Tour

3D Printing

3D printing is a great tool to help accomplish tasks where one needs unique pieces. We used the 3D printer to make the raspberry pi interface with Nao. We designed a backpack to hold the raspberry pi and a battery which powered the pi. The camera was attached to a wooden dowel, which we cut from a wooden spoon. To connect this to Nao we needed two pieces, one that would attach the dowel to Nao's arm, and the other to hold the camera and attach at the end of the stick. 3D printing allowed us to make these custom parts with ease and precision, and that let us design the parts to be easily removable and put back on when needed.

However with new technology there are always some problems that can occur. One of the most frustrating problems was when the 3D printer would, what seemed to be randomly, stop extruding filament from the print head. We first thought that the filament was not catching between the two gears that pull the filament into the heated print head. It was easy to take the front off and use compressed air to clean it out. This however, did not solve the problem. It is an odd problem with an even odder solution. It seems that some filament spools are not consistently wound around the spool. Therefore the solution was to unwind some filament and then manually wind it back up. The way they wind the spool may be too tight so the printer cannot pull it hard enough to loosen it. After this we had consistent prints with no errors in our printed objects.

Basic 3D printing information

The image below shows the options that the MakerBot has the following important options for the object one will print. Starting with number 1, the infill percent of the object; this is the density of the object, unless you need a really strong and small object, you should leave it at under 20%. Next, number 2, the number of shells is how thick the outside of the object is, this should be used if you want a sturdy object but It doesn't need the inside to be filled. Third the layer height is the difference between the first printed

layer and the next. The MakerBot can do 0.1mm at most, which is good if you want your object to look good, otherwise print at 0.2mm. The first three majorly change how long it takes to print your object. Infill will affect the print time the most, which is why shells is a better idea to use.

The next two are not hugely important but can allow for easier prints. The temperature should usually stay at 230 degrees Celsius. However for coloured filament you will need at minimum 250 degrees, we have not gone over that, but it made our black filament print easier. We never changed the speed of what was printed; you may only want to change the speed when you are printing a very complex object.

Finally there are a couple of check options that you can decide to use to make print easier. First always use a raft; this will print a bed like object which your object will then be printed on. The raft comes off easily and always makes sure your print does not fall over. The next option is structures, these are used to hold up certain overhangs in your print, they can be good and bad. It is a good idea to use structures if you don't have a complex object or an artistic one. Structures have to be removed after your print and can leave marks. Finally there is a preview option which should always be used before printing. It generates the raft and the structures for you with your object, then you can see if it could possibly make any mistakes.