

CMPT 433 Project

Iteration 2

Team SFU Mindstorm

Jaspal Sandhu	jss24@sfu.ca
Gener Iglesias	giglesia@sfu.ca
Cathy Li	cgli@sfu.ca
Amandeep Sandhu	ass17@sfu.ca

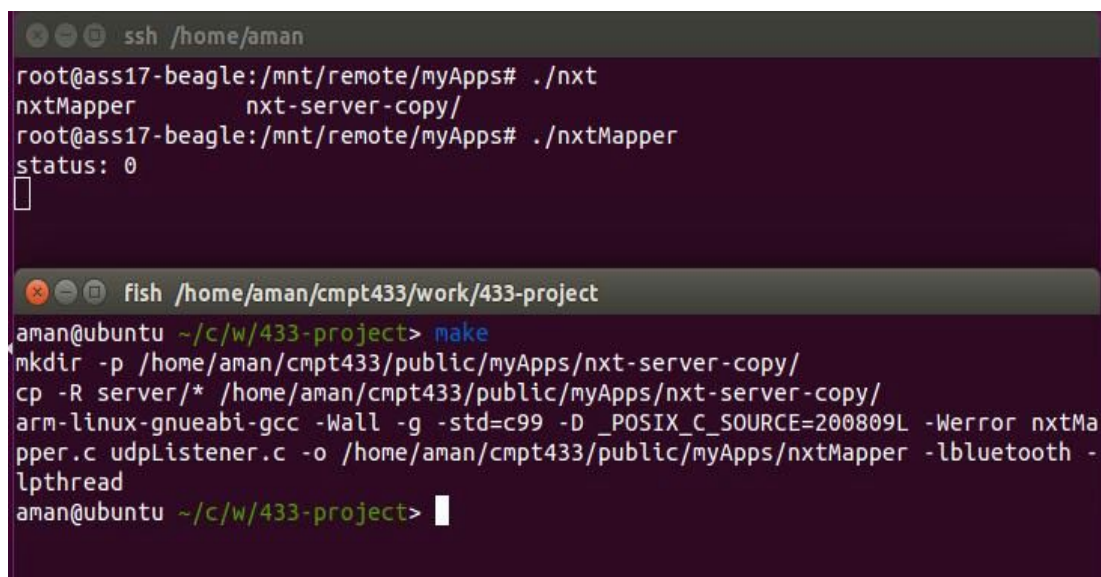
Project description

Our project is an NXT Lego robot which maps a room by gathering distances using an NXT IR proximity sensor. Other functionalities include displaying the map information on a website using node.js and manually controlling the NXT robot using the zencape joystick and increasing it's speed using the potentiometer.

Significant Accomplishments

1. Cross-compiling and Connecting Beaglebone to NXT via Bluetooth

In the previous iteration, we were unable to cross-compile and connect with the NXT brick successfully. However, we have managed to conquer this roadblock. (This will be further explained in the roadblocks section below). We were able to cross-compile successfully and establish a bluetooth connection with the robot, since the connection function (taken from the bluetooth library) returns a '0' status as seen in Figure 1 below.



```
ssh /home/aman
root@ass17-beagle:/mnt/remote/myApps# ./nxt
nxtMapper          nxt-server-copy/
root@ass17-beagle:/mnt/remote/myApps# ./nxtMapper
status: 0
█

fish /home/aman/cmpt433/work/433-project
aman@ubuntu ~/c/w/433-project> make
mkdir -p /home/aman/cmpt433/public/myApps/nxt-server-copy/
cp -R server/* /home/aman/cmpt433/public/myApps/nxt-server-copy/
arm-linux-gnueabi-gcc -Wall -g -std=c99 -D _POSIX_C_SOURCE=200809L -Werror nxtMa
pper.c udpListener.c -o /home/aman/cmpt433/public/myApps/nxtMapper -lblueooth -
lpthread
aman@ubuntu ~/c/w/433-project> █
```

Figure 1 - Connection established with status:0

One of the guides that we followed mentioned that the program should return a '0' if connection is established. Our bluetooth connection can also be proven when we run the our program which moves the motors in our next point.

We consider this as one of our significant accomplishments since most of our time has been dedicated to fixing this issue. This the most important component of our project.

2. Working NXT Motors for Movement

Another significant accomplishment we achieved was motor movement. We researched the nxt lego manual and it mentioned using command packets to send commands to NXT. We were able to use C's `uint8_t` byte array to send the command to the NXT brick through bluetooth and have the NXT behave appropriately.

<https://www.youtube.com/watch?v=LoFYP2WXrkl&feature=youtu.be>

3. Joystick and UDP/Node.js Functionality

Finally, our last significant accomplishment was utilizing the zencape joystick and a node.js server to move the robot. Once we discovered how to move the motors, we decided to implement some of our planned features.

The joystick controls the robot's movements. Pressing up will move the robot forward. Pressing down moves the robot backwards, and pressing left/right will make the robot rotate a set amount of degrees to that side. This can be seen in the video youtube link below: (*Note: this is the same video as above)

<https://www.youtube.com/watch?v=LoFYP2WXrkl&feature=youtu.be>

The node.js server has buttons that have the same functionality as the joystick inputs mentioned above for now.

Roadblocks and Unexpected Challenges

From the first iteration until now, one of our major drawbacks was trying to cross compile simple code to the BeagleBone Black and running it. Following past CMPT 433 guides, we were able to cross compile with much difficulties due to the uncertainties of which libraries and versions to use. After successfully cross-compiling on the host to create an executable, we ran into our second most time-consuming roadblock which was the "libbluetooth.so.3" error for missing library. This error occurred when we tried running the executable on the BeagleBone Black. Not knowing the cause of the error and with very little information of it in relation to the project we are doing, we tried various installations and reinstallations of libraries in attempt to solve it. In the end, we figured out that to solve the "libbluetooth.so.3" error, an additional library "**libbluetooth-dev: armel**" is required for successful running of the program on the target (which was not suggested in any of the past guides or online help pages).

Project Proposal and Current Accomplishments

Iteration 1 goals

- Assemble robot
- Set up and configure bluetooth
- Send simple movement commands to the NXT robot
- Receive data from NXT robot

Iteration 2 goals

- Make a mapping system that the NXT robot uses to map its surrounding environment
- Design a system where the robot traverses the entire area efficiently
- Use map data to generate a web page displaying the map
- Use generated map to give coordinate for the robot to travel to
- Use the BeagleBone joystick to manually control robot and the potentiometer to manually increase motor speeds.

The above goals were extracted from our original proposal. By iteration 2, we expected our project to be near complete with all the mapping design, web page and code for the robot to be finished; however, due to unexpected roadblocks in the connection/ communication aspect (cross compiling, running program on target and using bluetooth), we are a bit behind in our original schedule. We are mainly behind in the mapping aspect of our project and incorporating it with everything else. Currently, we are also experimenting and figuring out the code for the NXT robot's controls which would eventually lead us into figuring out the mapping procedure.