IIIT B Internship Project Report

Submitted By:
Manoj T Kumar
4th Year
NIT Trichy

Introduction

The internship between the months of May to July, focused on the applications of robotics. A readymade robotics kit, **The LEGO NXT 2.0** was provided for use for the duration of the internship. The kit included a **NXT Intelligent Brick**, which runs on **32 bit Atmel AT91SAM microcontroller**, and **8 bit Atmel ATmega48 microcontroller**. The brick also provided input and output ports to connect the various sensors and motors available. The brick had a USB port to connect to the computer, as well as Bluetooth connectivity. The UltraSonic sensor, and 3 motors were mainly used for all projects.

The firmware used for programming purposes was **Lejos 0.9.1**. This firmware enables Java programming on the brick, enabling the programmer to apply Java OOP concepts directly, as well as other concepts such as multithreading with synchronization. This also enables the use of **Java Swing** for **Graphical User Interface** design. The software used was **Eclipse Luna**, on a Macbook Pro. During the course of the internship, projects which focused on Android Development required the use of software such as **Android Studio and Netbeans**.

The projects attempted and completed successfully during the internship are listed below.

- **Mapping:**
  This involved the robot being able to sense it's surroundings, using the ultrasonic sensor and measure the distance of points from itself, to create an accurate map of the physical environment it is situated in.

- **Orientation and Heading Determination using Android Phone Sensors:**
  This project attempted to solve the inaccuracies of the built in position controller of the robot, by obtaining the heading, and orientation values from the accelerometer and magnetometer sensors of any Android Phone.

- **Remote Control Race Car:**
  The aim behind this project was to bring out the multithreading capabilities of the NXT Brick. A PS4 controller acted as a remote, and perfect synchronization of the NXT Race Car was achieved.

- **Bluetooth Connectivity**
  This small scale project was done just to test the Bluetooth connectivity of the NXT Robot. This project is not further

mentioned in the project report, but the entire code has been provided.

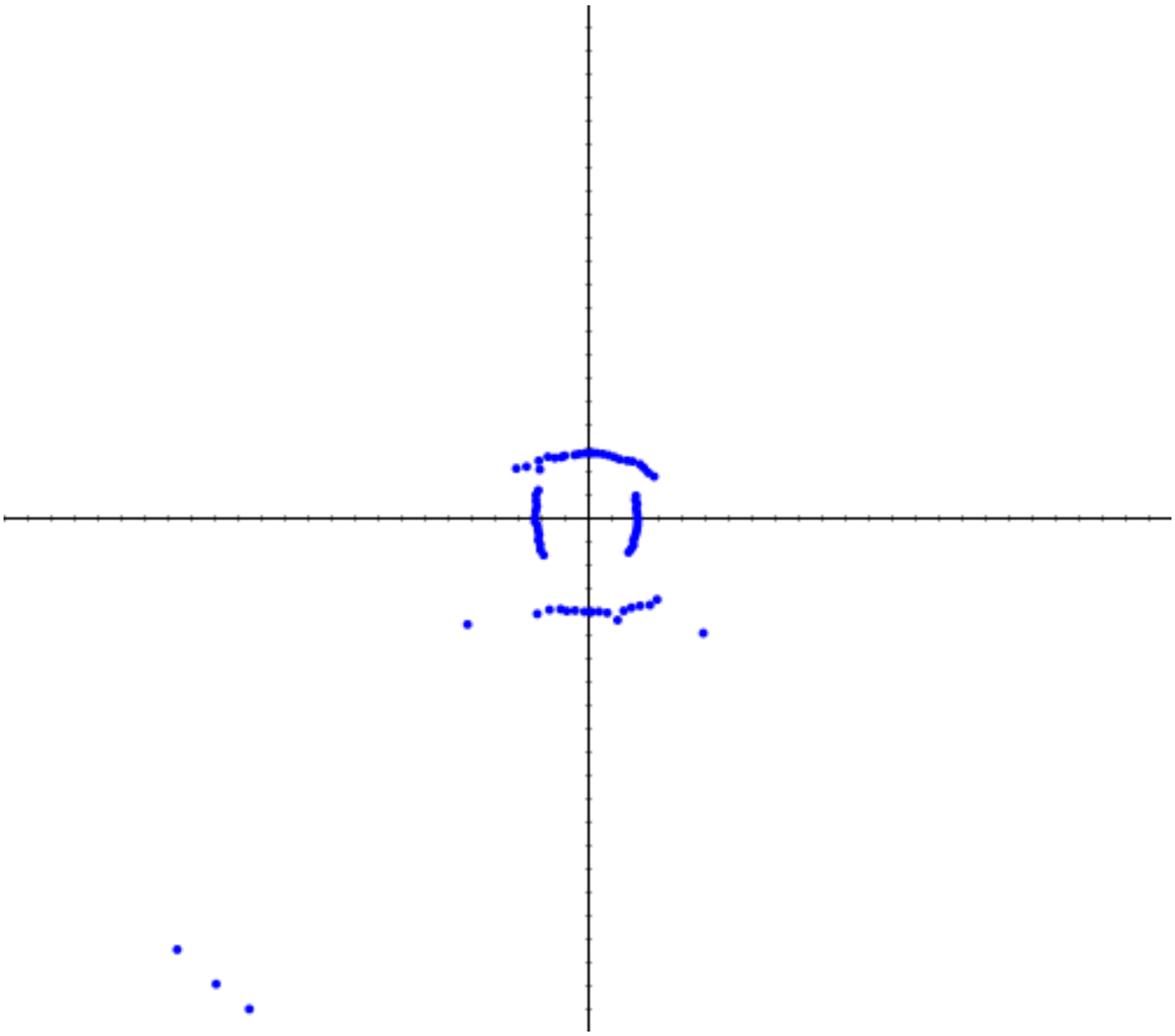
Mapping

The aim of the project was to enable the NXT Brick to plot an accurate Cartesian Graph representing its physical environment.
An UltraSonic Sensor was used to measure distances of points from the robot. 3 NXT Motors were used. 2 Motors were to enable movement, while the third motor was responsible for the rotation of the UltraSonic sensor to provide a 360 degree field of vision. After drawing a basic blueprint of the robot, as per the requirements stated above, the robot was physically built. The figure below is depicts the robot used for this project.

The Java program was constructed in such a manner that the program mainly ran on the CPU itself, thereby using the PC computational abilities, and reducing the workload on the NXT Robot as such. The robot was mainly used to provide the sensor readings to the PC. The readings were then used for the calculations, and in turn plot the map. The PC issued basic motor and sensor related commands to the

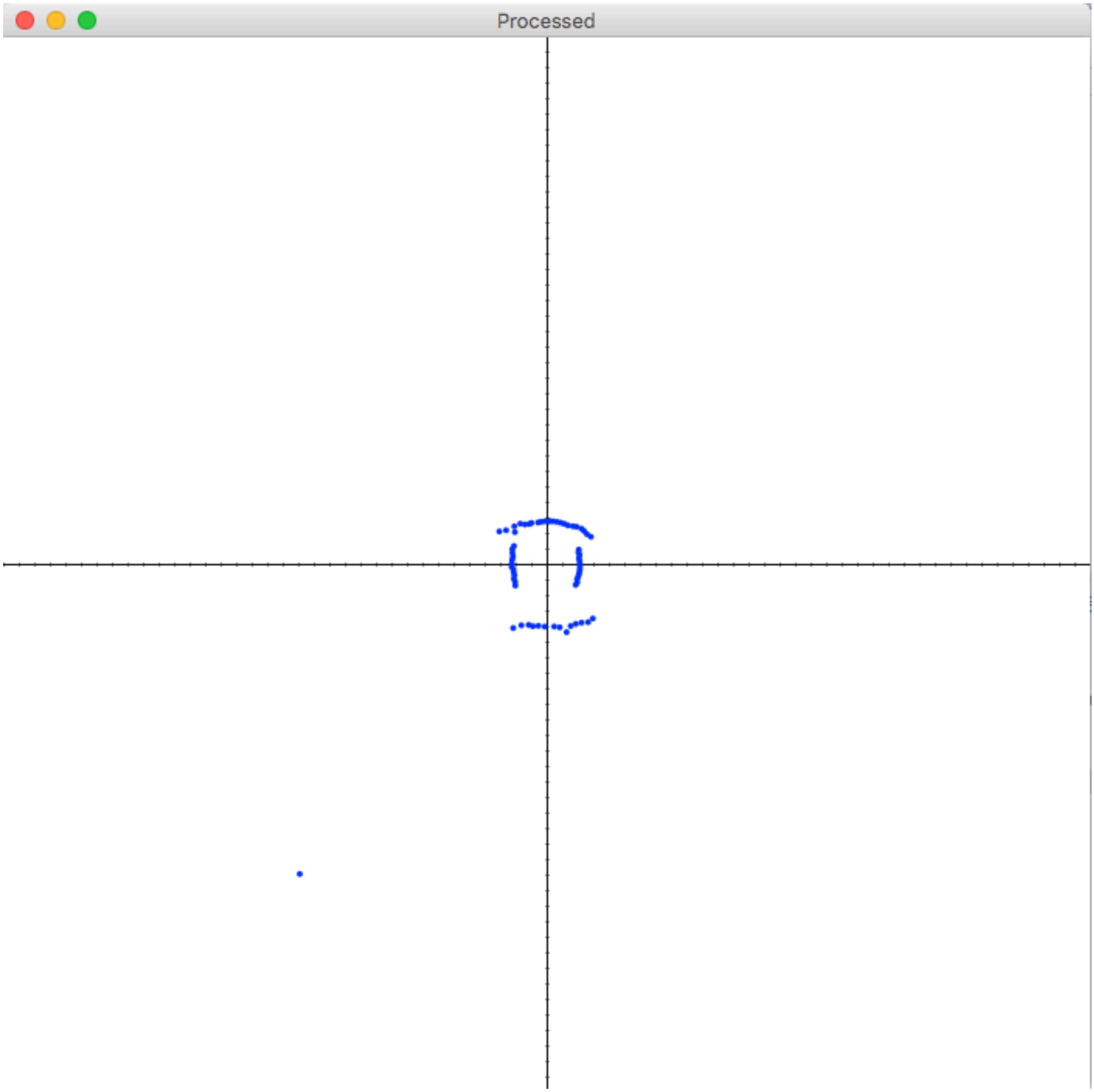
NXT. This method of execution architecture was advantageous, and enabled the use of Java Swing Library to plot the map.
The entire project code is also provided, along with a small video demonstrating the entire mapping procedure.

The Raw Data obtained from the UltraSonic sensor when mapped is shown in the figure below.



This mapped is obtained when the Robot is kept in a box, which is closed on all 4 sides.

Basic processing is done afterwards to this Raw Data, to remove the outliers. The processed image is shown in the figure below.



The following issues were encountered during this project. Possible and viable solutions are listed alongside the issues.

- **The ultrasonic sensor was not accurate enough.**

  The sensor provided only integer values for the distance of points from the robot. This issue caused the straight lines of an object in the physical world to be mapped as curved lines instead in the map.

  Possible Solution:

  Image processing should be applied to the raw data, and the curved lines should be converted to straight lines.

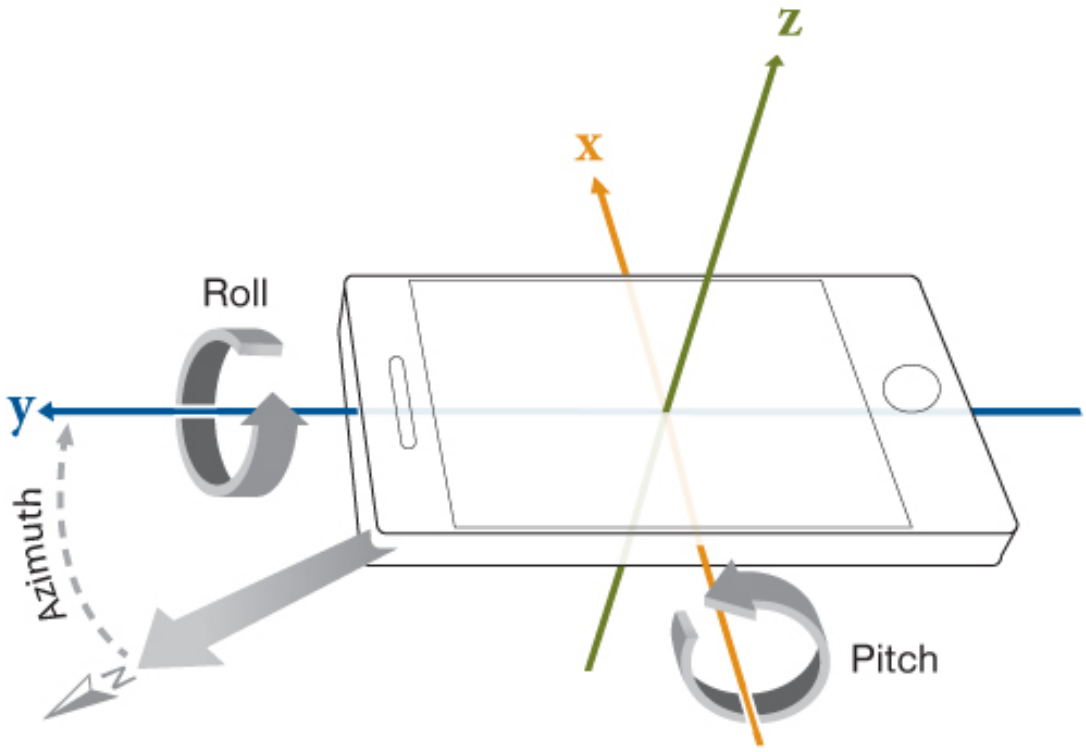
Orientation and Heading Detection

The aim of this project was to utilize an Android phone's accelerometer and magnetometer sensors to provide orientation, heading and position detection for the NXT Robot. The in-built NXT Robot's position detection and controller displayed many inaccuracies and anomalies. Therefore this issue can be resolved by utilizing the phone's far more sensitive and error free sensors.

The application was developed for an Android Phone using Java on Android Studio. A client server architecture was created, were the phone acted as the server, sending values from the sensors to the client which was the PC. The values were sent over a socket connection, through a WiFi hotspot created between the phone and PC. The NXT Robot in turn directly received these values from the PC.

The phone's accelerometer and magnetometer working in tandem, provided three axis values, roll, pitch and azimuth.

The figure below shows the three axis of the phone as per the Android Developers Guide.



These three axis readings when given to the NXT Robot can therefore be used to obtain an accurate orientation, heading, and position detection of the NXT Robot.

The entire code for this project has also been provided.

Remote Control Race Car

This project was mainly done to test the multithreading capabilities of the NXT Robot.
After creating a blueprint of the NXT Race Car, the robot was physically built. The robot designed is shown in the figure below.

The robot used a total of three motors. 2 motors were used for the forward/backward movement, and the third motor attached to a rack and pinion system was used for steering.

A PS4 Controller was used as the remote control. The left trigger button was used for braking/reverse. The right trigger button was used for acceleration. The left joystick was used for turning.
The values of the trigger buttons had a range of -1.0 to 1.0. Any value above 0, was used to calculate the percentage of power that should be supplied to the motor. Therefore the harder the trigger was pulled the faster the racecar would accelerate. Similarly the left joystick provided x-axis and y-axis values in relation to the movement of the joystick. These values were then converted to polar coordinates to determine the angle through which the racecar should turn. However the joystick had to be moved to such an extent that the x and y coordinates always laid on the perimeter of a circle of unit radius, otherwise no turn operation would take place.

The program worked as intended. A total of 3 threads were running side by side. One thread continuously polled the controller input, to obtain the readings. The second thread, which was responsible for the forward and backward movement of the robot, was notified whenever a change in controller values was detected. The third thread worked similarly, taking care of the turning operation whenever a change in the controller value was detected.

The entire project code, and a small demonstration of the NXT Race Car have been provided.

Conclusion

The above projects were completed successfully, and the entire project has been provided.

The NXT 2.0 provided a great platform to develop a deeper insight in Robotics and programming in Java.

I highly appreciate this opportunity provided by Dr.Sachit Rao, and IIIT Bangalore, allowing me to learn, and gain knowledge in Robotics, by providing me with the necessary resources, and mentorship.
The internship was a great experience, and I leave, with a deeper and greater understanding of Robotics.