Course Project Documentation

CS308 Project

Android Interface – Firebird API

TEAM 2:

Pararth Shah (09005009)

Aditya Ayyar (09005001)

Darshan Kapashi (09005004)

Siddhesh Chaubal (09005008)

Table Of Contents

1) Introduction:

- To create applications on Firebird, one requires an understanding of the hardware specifics of Firebird
- This limitation has become the motivation for our the project
- Our objective in this project is to provide an easy-to-use Firebird API for Android
- Our API allows a programmer to create Android apps to control the Firebird V
- We have made available all features of Firebird to the Android developer through our API
- This project has brought the two avenues of robotics and Android programming closer
- Due to our work, now any interesting use of the Firebird bot is exposed as an app written for any Android device

2) Problem Statement:

- To create an Android library app which acts as an API for accessing the features of Firebird
- This library should be available to Android developers for importing in their Android apps to do interesting stuff with the Firebird
- To write a generic command interpreter which runs on the Firebird, and waits for command messages over the communication interface, performing various actions as required
- To create a Bluetooth communication interface for communication between the Android device and the Firebird
- To design and implement a two way communication protocol for reliably and efficiently sending and receiving data between the Android device and the Firebird
- To create sample applications on the Android which demonstrate the use of the Android API for Firebird

3) Requirements:

A) Hardware Requirements:

1. Firebird V
2. Android device (with API 10 or higher)
3. Bluetooth module for Firebird

B) Hardware Requirements for Sample Applications:

1.Sharp Sensors	:	Used in the PeterParker application, to detect empty parking spaces. Used in the MapperBot application, to detect distances from wall.
2.Proximity Sensors	:	Used in the MapperBot application, to detect distances from wall.

C) Software Requirements:

1. AVR Studio	:	To program instruction onto the bot

2. Android 2.3.3 SDK (API 10)
3. Eclipse IDE with ADT Plugin

4) Implementation:

a) Firebird Module:

1. *Bluetooth Communication*
    - There is an interrupt handler for receiving data via the bluetooth. The interrupt handler adds the the received data to an input buffer
2. *Command Processor*
    - A function goes over the input buffer, and converts the input received to commands that the Firebird can execute. This involves removing irrelevant control data that the Bluetooth module sends, and parsing the other input according to our protocol
    - Our protocol is such that it allows single byte and multi byte commands. For simple commands like go forward, this is accomplished by only a single byte. For complex commands like set port X to value V, we first send a byte corresponding to the command “set port”. The next bytes will tell the Firebird which port and what value
3. *The Main Loop*
    - The bot takes these commands one at a time, and executes them. Examples of commands include buzzer on, buzzer off, move forward, move left, set port X to value V, get value from port X, get value of sensor S, etc
    - The Firebird is modelled as a finite state machine. It can be in multiple modes at a time. For example, its adaptive cruise control

may be on and it may be following a white line along with its buzzer on. Even in this state, one can command it to get the value of some particular port without having to “wait” or “get blocked” for the motion to stop or adaptive cruise control to be turned off before getting the port value

b) Android Module :

Implemented as an Android library which can be imported by any other application

1. *Bluetooth Communication*
   - It uses the phone's Bluetooth to connect to the Firebird and open a Bluetooth socket which is used for further communication with the Firebird
   - Any data to be sent can be directly sent using the Bluetooth socket opened while connecting
   - There is separate thread which runs in parallel to listen to any data received on the Bluetooth
   - In case the application has specified a handler to send the received data to, then, it passes on the data to the application, otherwise, it is stored in a buffer
2. *Firebird Class*
   - This is the main class to be used on the Android. It has methods to connect to a Firebird with a specified Bluetooth MAC address, and send commands to it

- There are methods to send common commands which use the protocol and send the required bytes to the Firebird
- It also has a method to send raw data to the Firebird, for e.g. Send 12, 83, 23
- This class encapsulates all the commands that can be given to the Firebird and is the entry point to the API.

Note: The Firebird Module and the Android Module remain the same for ALL applications.

c) Sample Applications :

1. *FBController*
   - This is a very basic application which takes as input, a byte and sends it to the Firebird
   - We can see the functioning of the Firebird on receiving any data using this application

2. *MissionControl*
   - This applications gives an interface to the user with buttons to move the bot forward, backward, left, and right, buzzer beep, adjust velocity, get IR sensor reading continuously
   - It keeps polling the Firebird continuously to get the value of the IR proximity sensors and displays on the screen

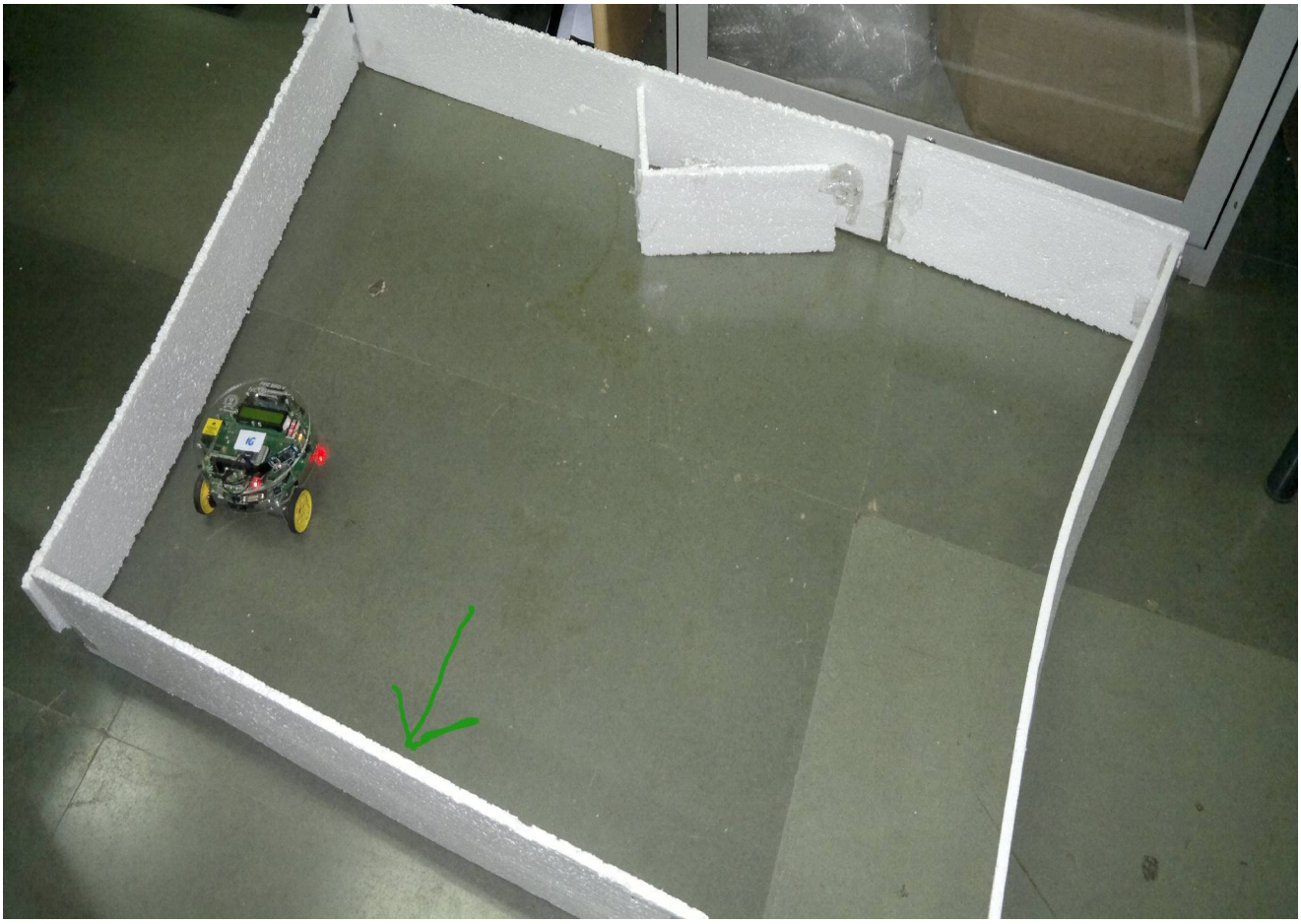
3. *PeterParker*
    - This turns the Firebird into an autonomous parking bot
    - Whiteline following is turned on and the bot sends back an “acknowledgement” when it reaches an intersection
    - The Android polls the Firebird to get readings from its sharp sensors
    - It determines if the spot to its left is available for parking
    - If not, it turns on whiteline following again
    - When it finds an available slot, it turns left by 90 degrees, moves to the spot, makes a u-turn and reverses into the parking space
    - All the required commands are sent from the Android

4. *MapperBot*
    - This turns the Firebird into a wall following bot
    - The Android continuously polls the Firebird to get readings of its front and left, Sharp and IR Proximity sensors
    - It then determines what it should do to keep following the wall to its left
    - In case of a change in direction, the Android also gets the interrupt count on one of the wheels to know how much distance it travelled
    - Now, it has sufficient information to draw the trajectory that the Firebird followed, onto the screen of the Android
    - This way, as the bot keeps following the wall in the room, a map of the room is created on the Android phone which can be saved as an image or emailed

5) Testing Strategy and Data:

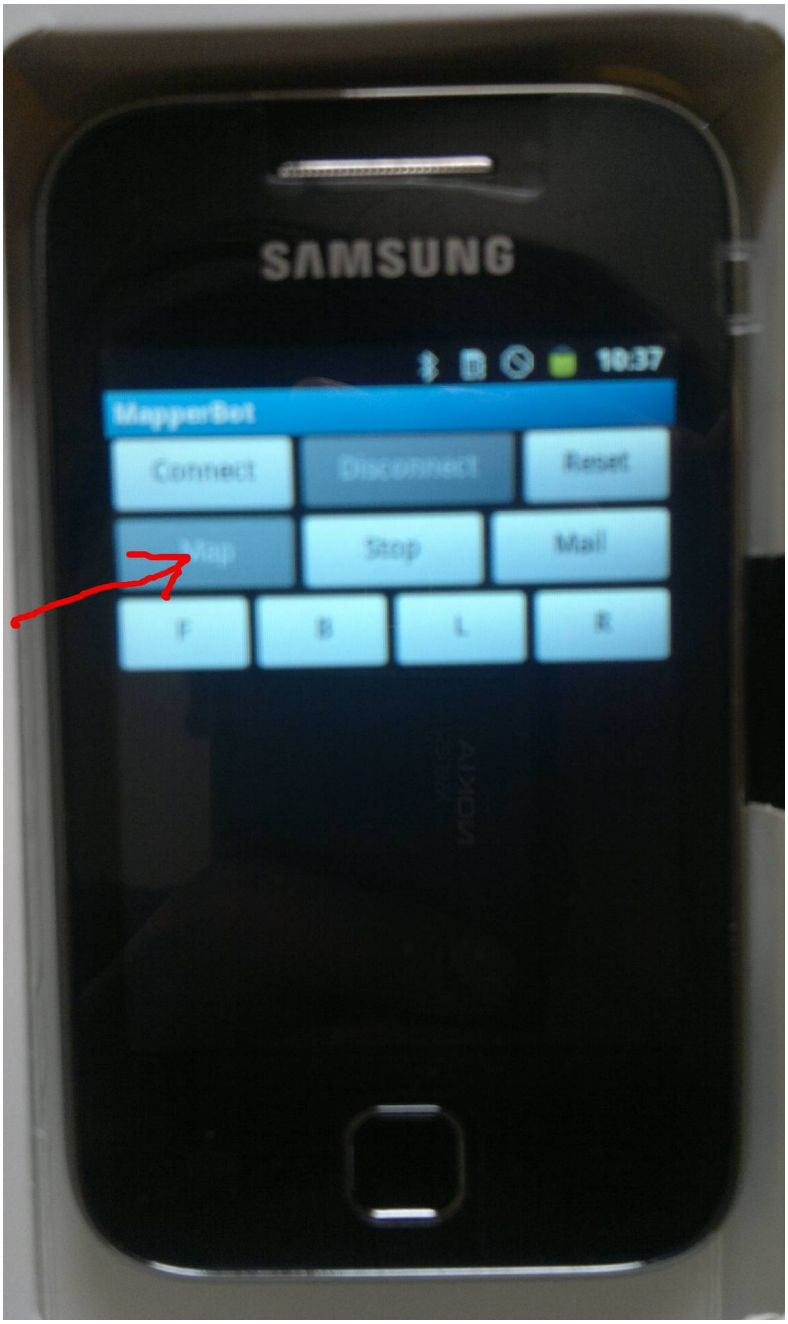
1) Mapper bot:

The bot is kept in an arena.

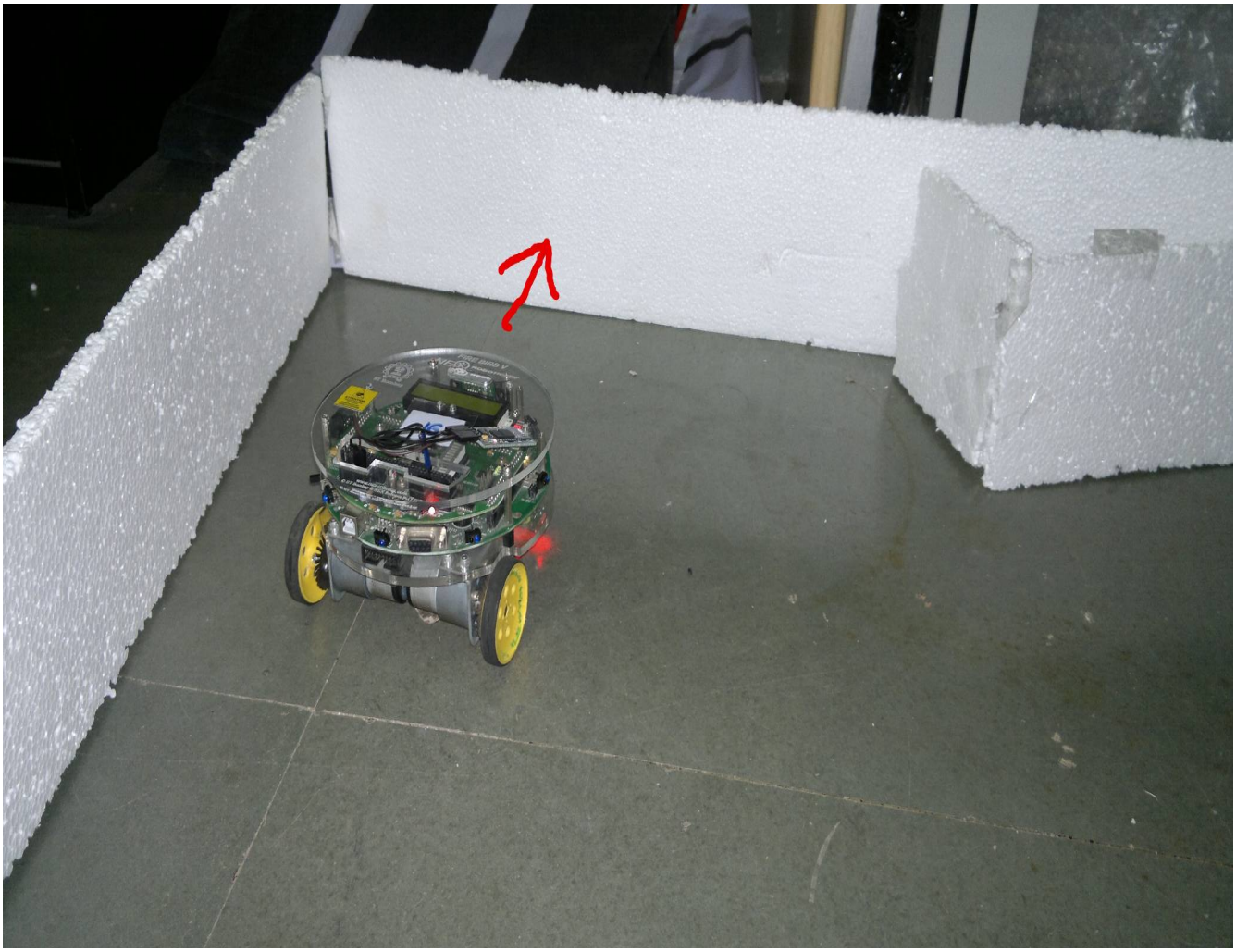


A wall has been depicted by the arrow.

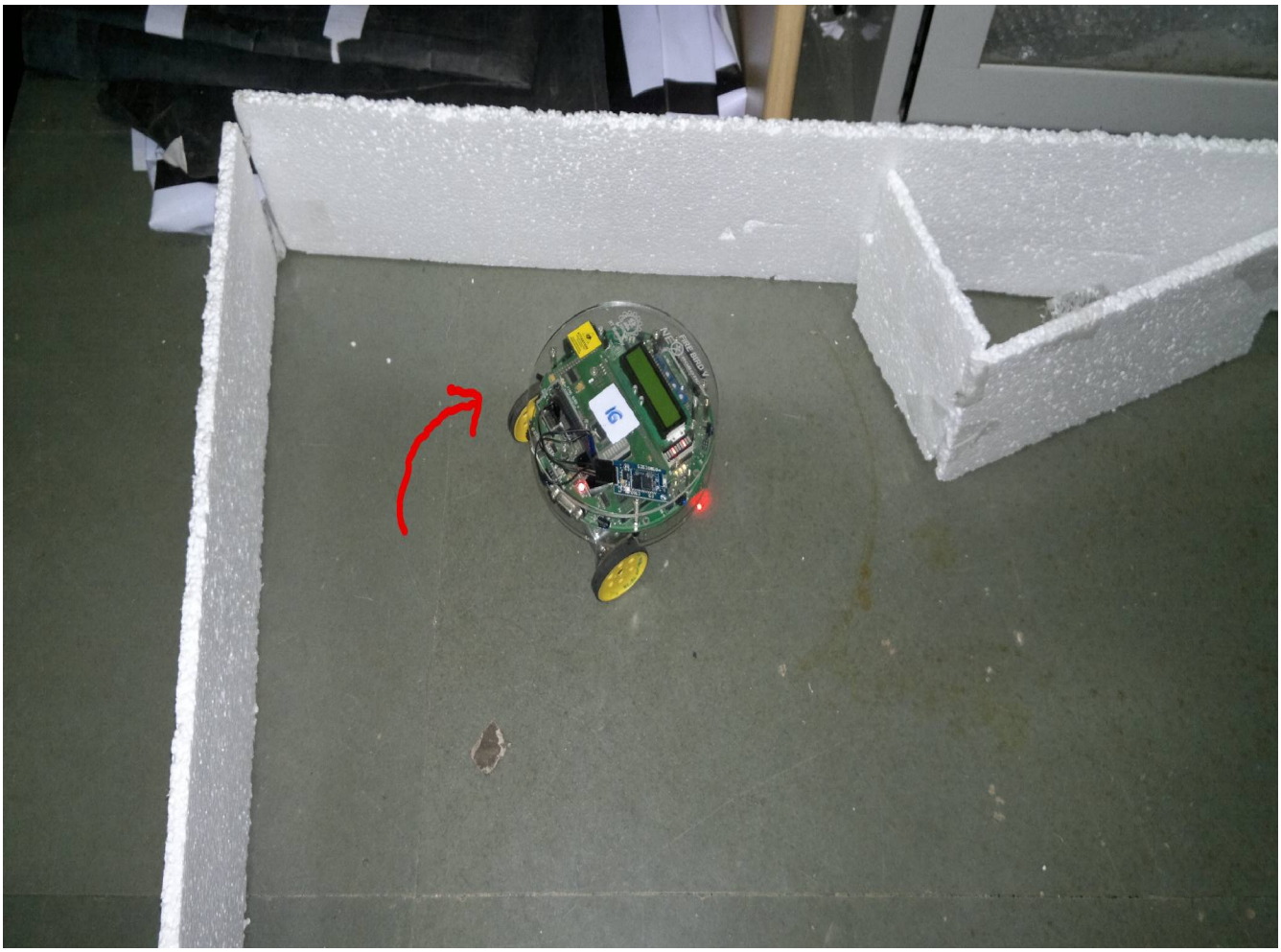
Our android interface has a map button.



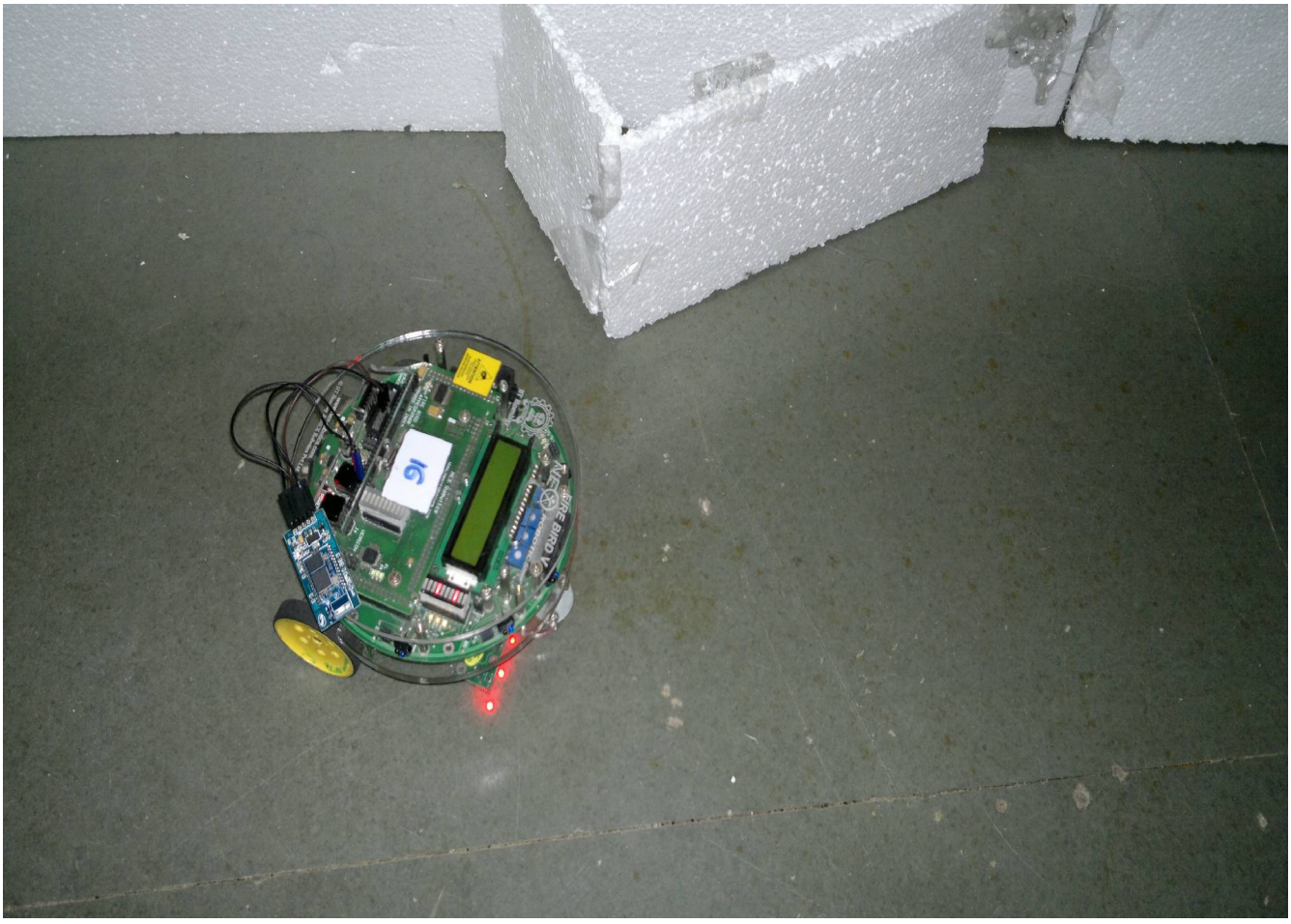
On pressing the map button, the bot starts moving along the wall. It sends sensor values to the android device and follows the left wall till it encounters a wall in the front.



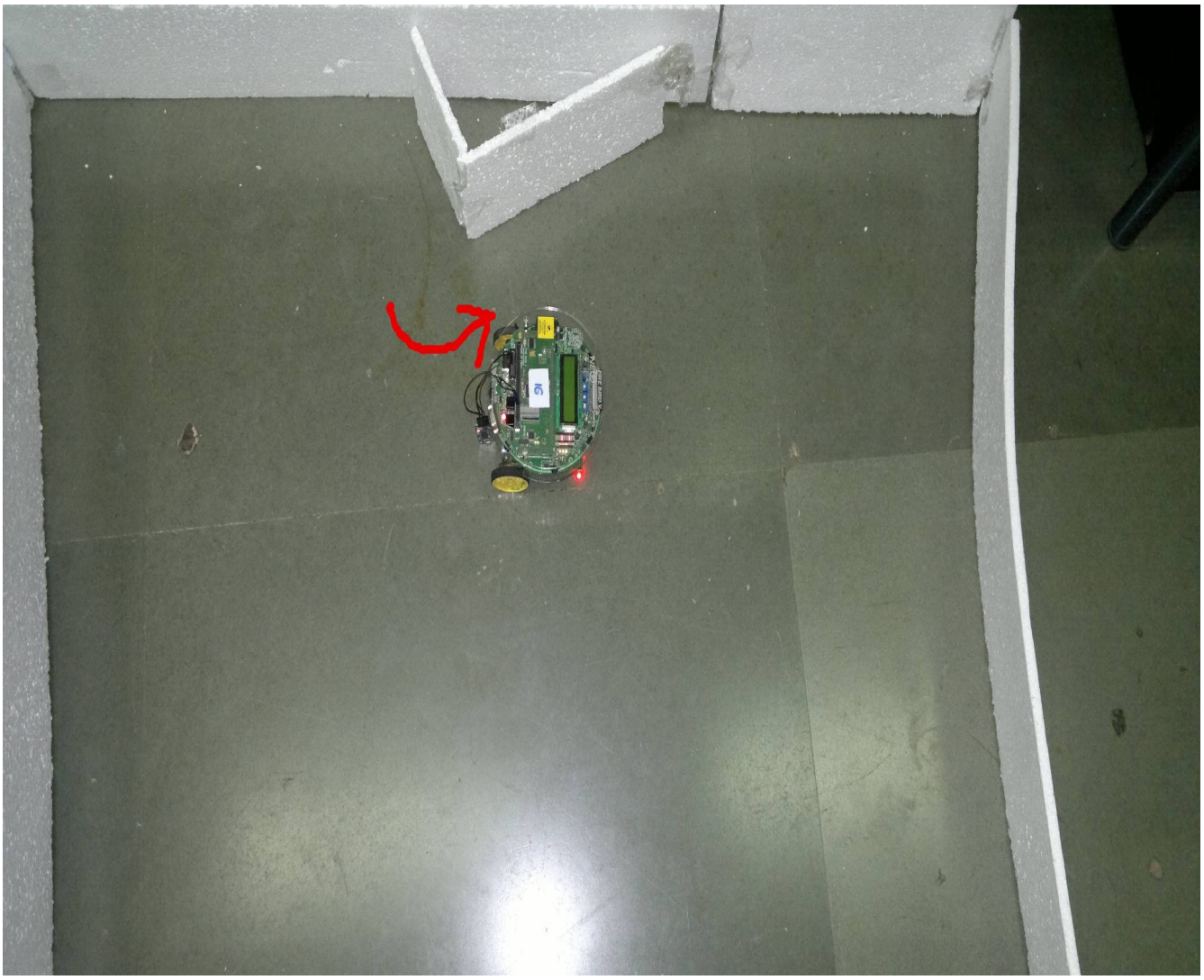
It now takes a right turn, and continues following the left wall.



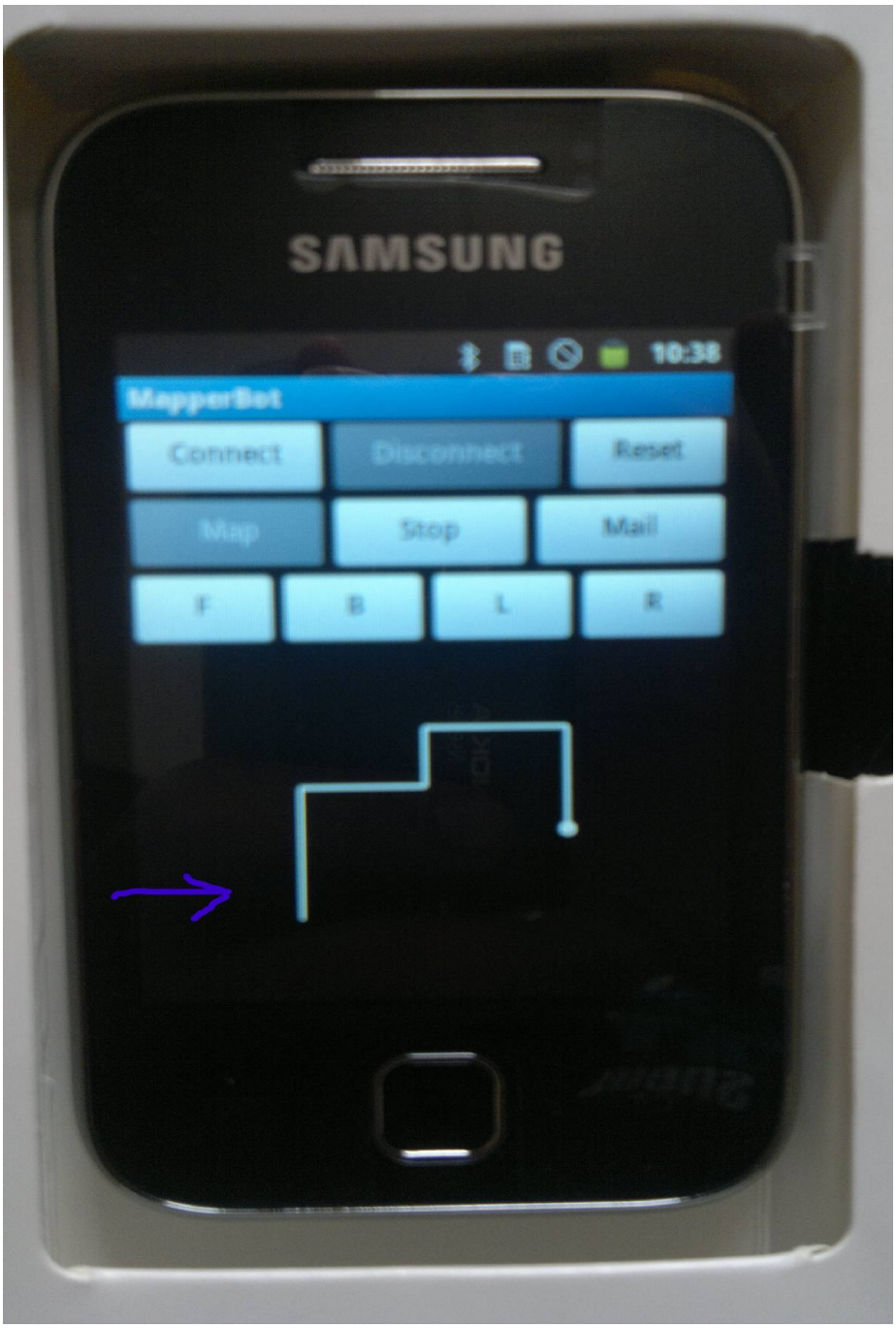
On moving further, it finds that there is no wall to its left.



And it then takes a left turn, and keeps following the left wall.

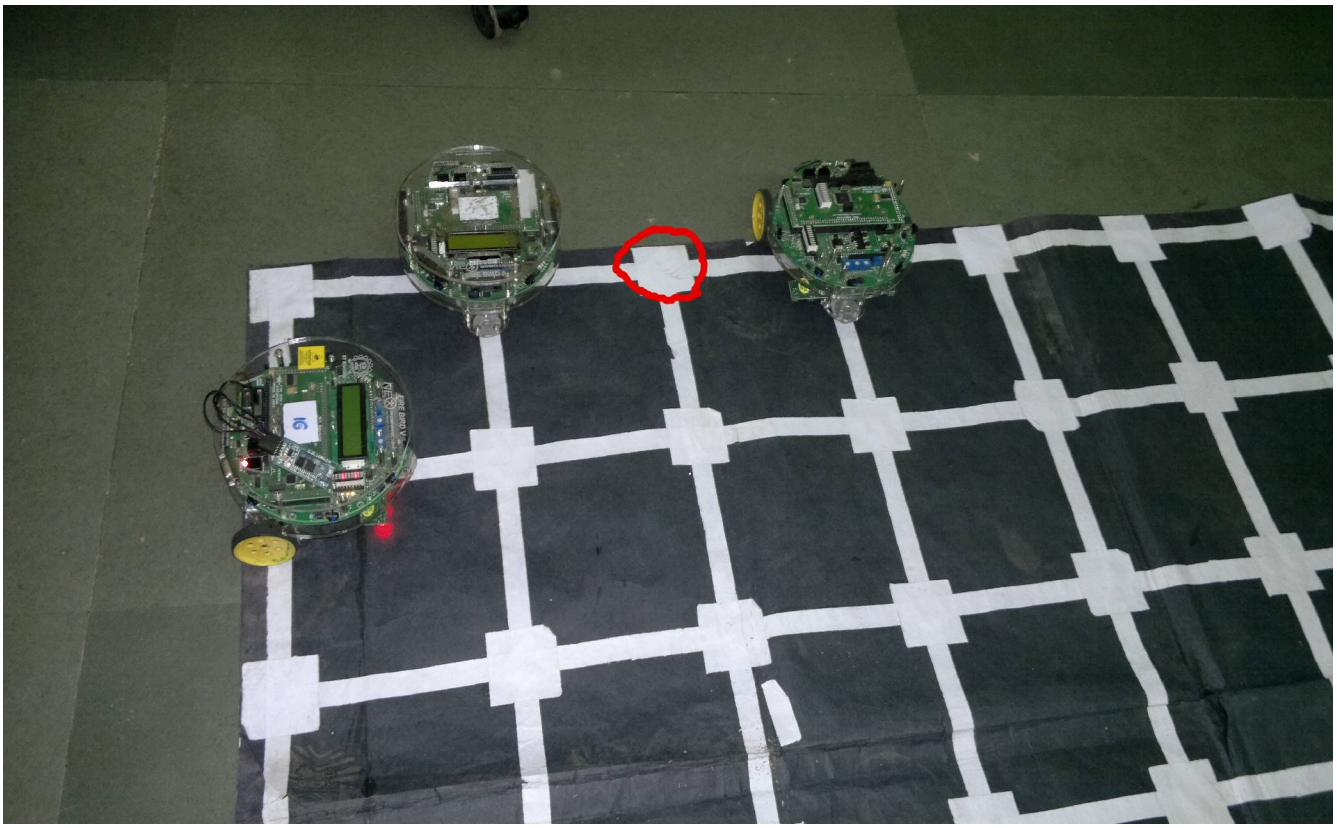


The android device creates a map of the arena that it traversed.

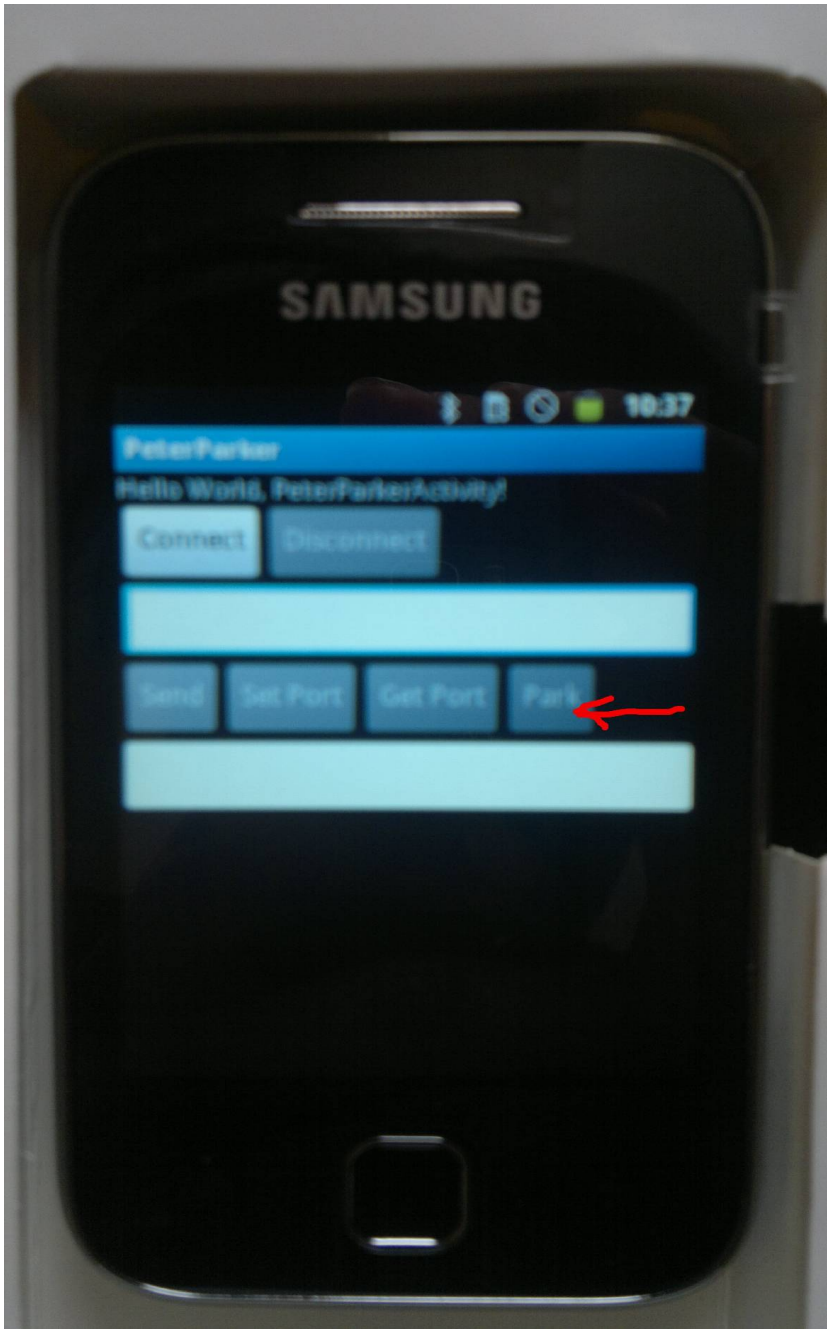


2) Peter Parker (The parking app):

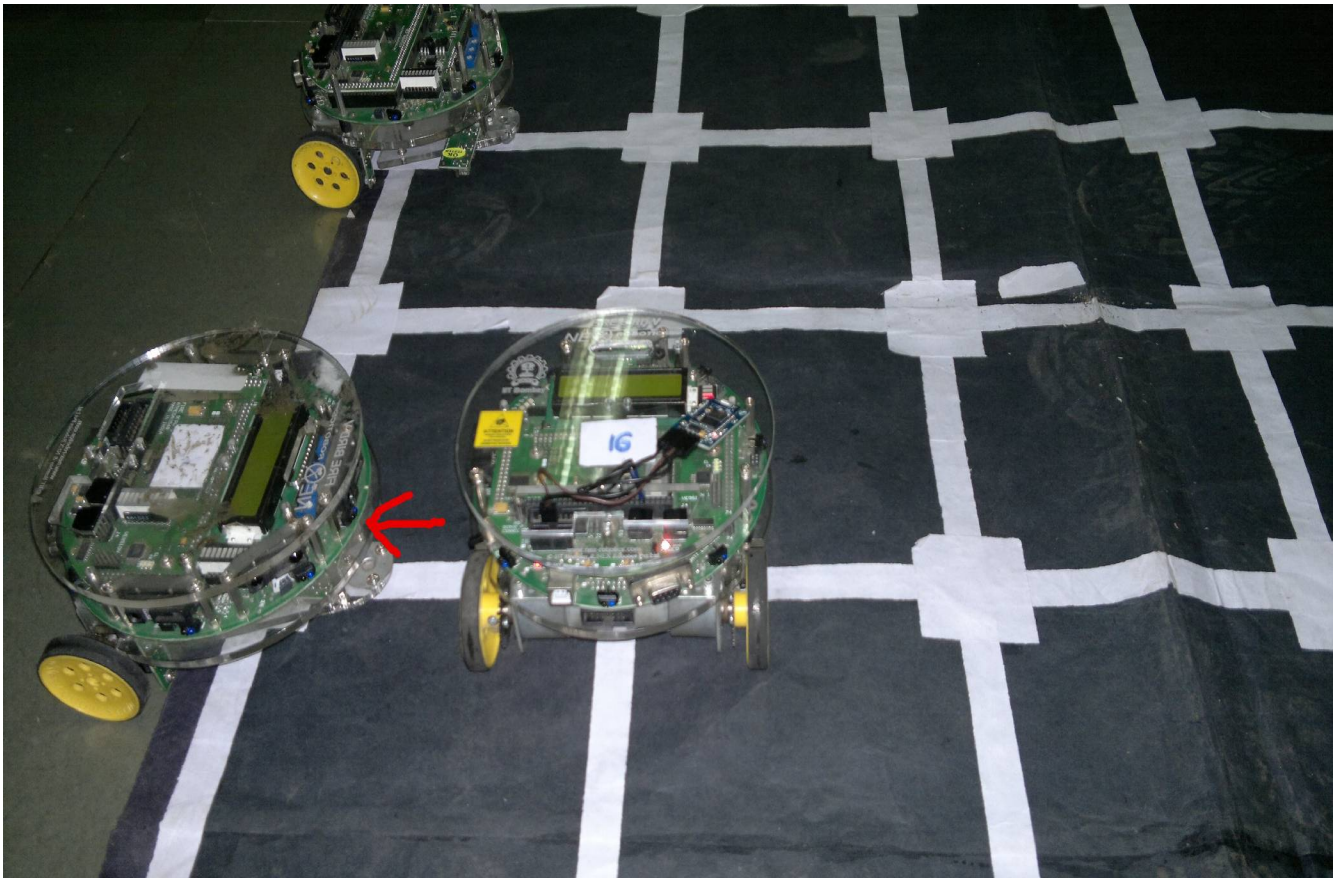
The bot is kept in a parking arena, with parking spots to its left, as shown.



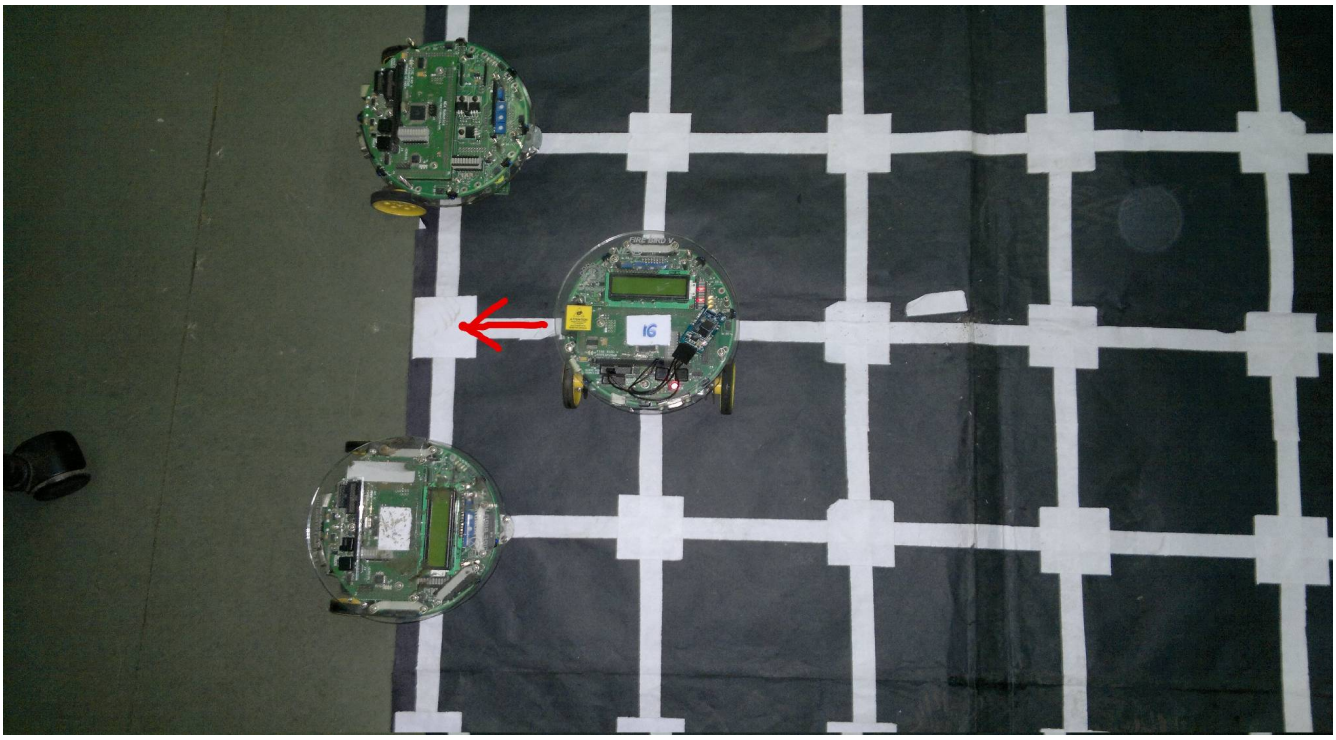
Our interface has a park button



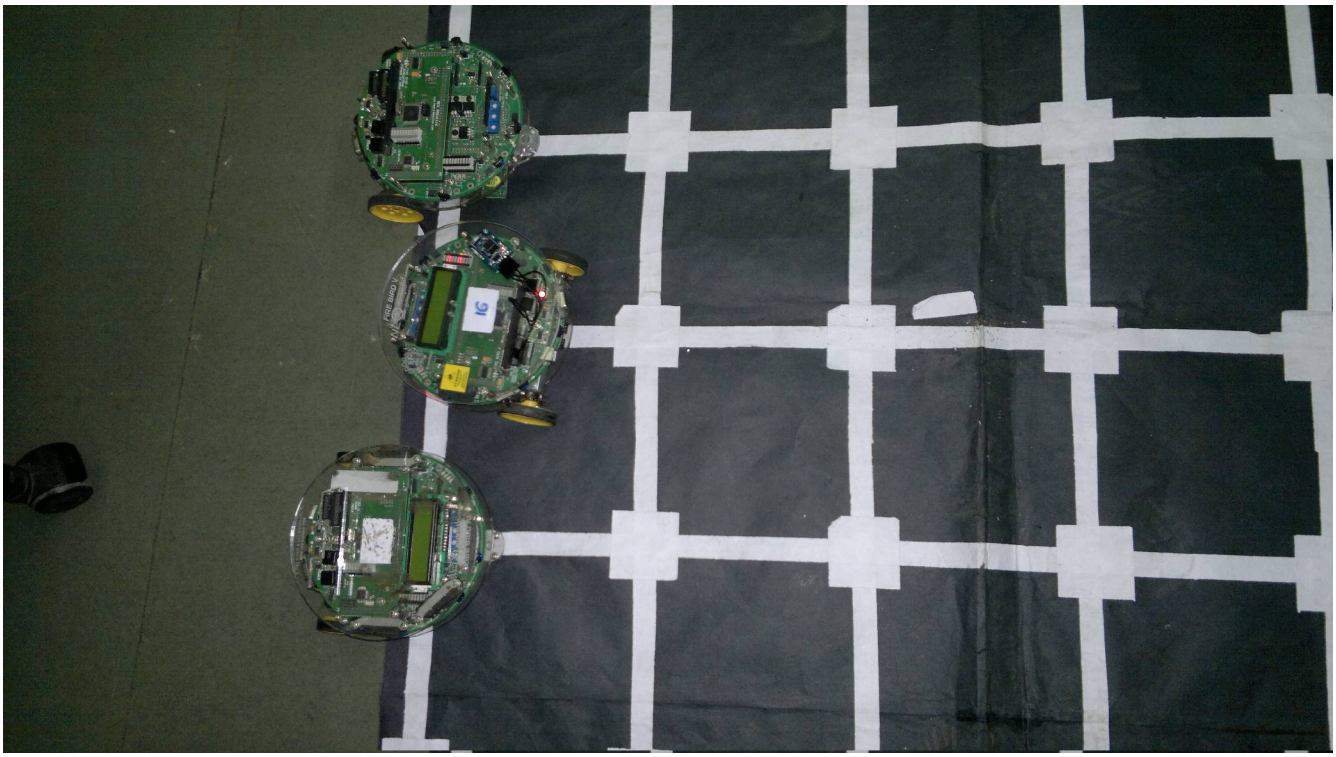
On pressing the park button, the bot starts moving forward along the whiteline and finds that a bot is parked to its left.



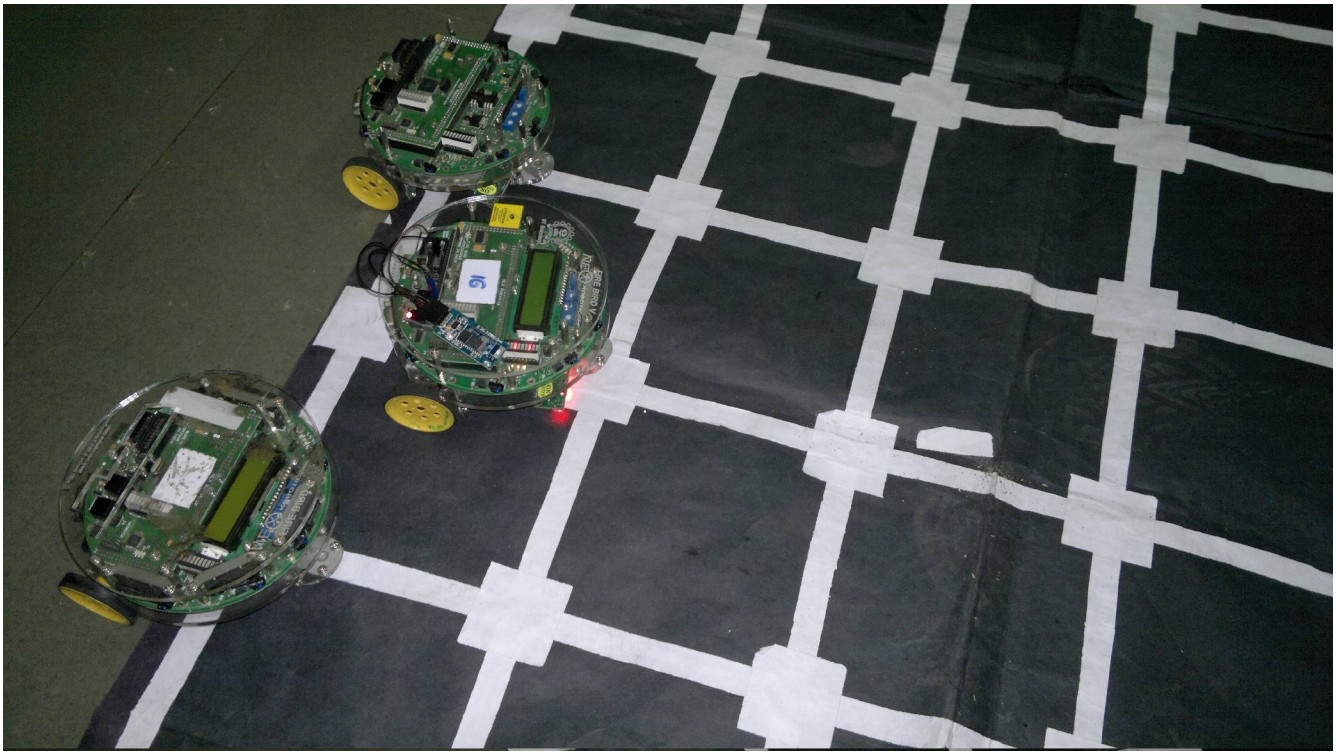
So, it goes further ahead and finds an empty parking lot.



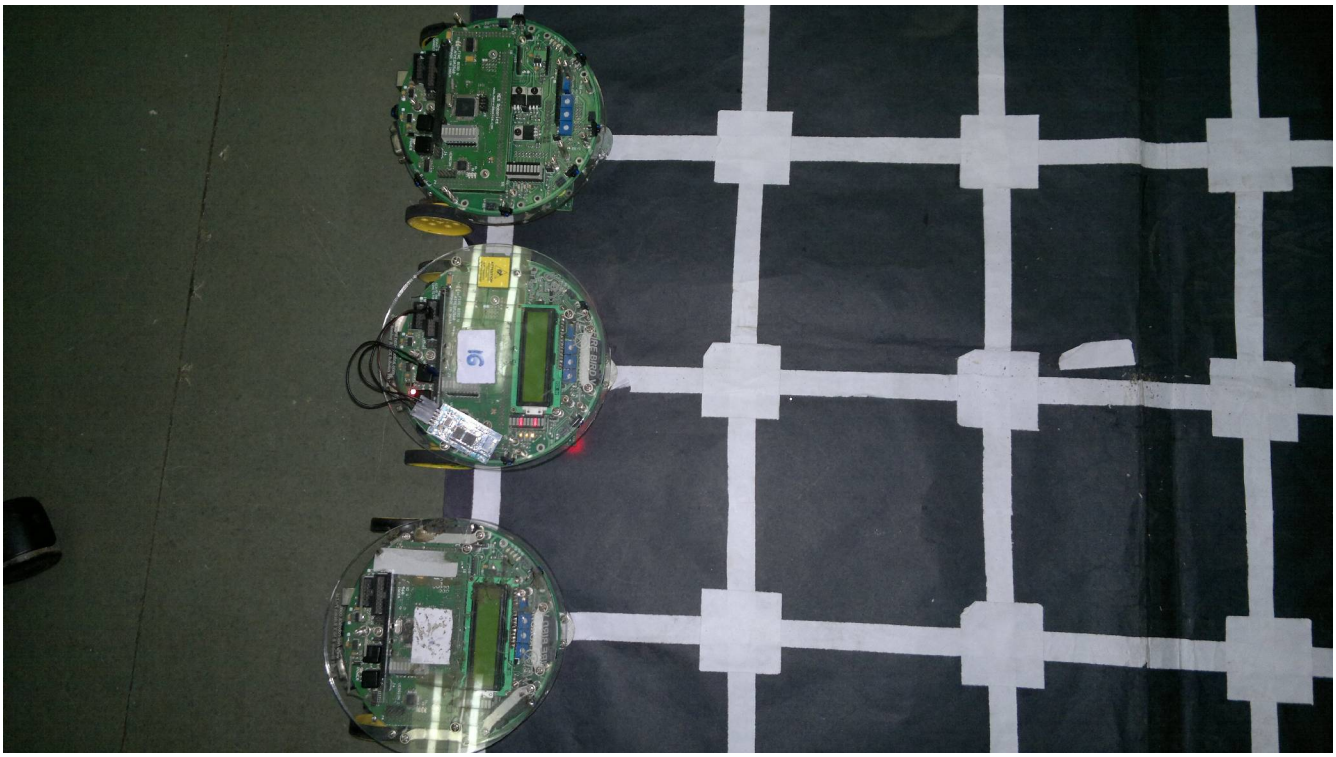
So, it goes to that spot, turns around and parks there.



It takes a U-turn to park.



It has parked successfully.



3) Mission Control:

Our mission control app has buttons that direct the bot to move forward, backward, right and left, and also for velocity control and buzzer. It displays sensor values at the top.

4) Fb Control app:

This app has buttons to send and get port values of the firebird.

6) Discussion of System:

a) <u>What has worked as per plan?</u>

1. *Generic Java API For Android*
    - We successfully implemented a generic API for the Android platform which can be easily included by Android apps to manipulate the Firebird

2. *Generic Firebird Command Interpreter*
    - We have implemented generic Firebird C code encapsulating all features of the Firebird
    - It is common between all possible apps, so it is required to burn the code only once one the Firebird, after which everything can be done via the Android API

3. *Bluetooth Communication Interface*
    - We could successfully implement a Bluetooth Communication Module for transferring command and data messages between Android and Firebird
    - The throughput given by the BT module is roughly 5-6 commands per second
    - This translates to a precision of 150-200 milliseconds in reading sensor values

4. *Sample Applications*
    - We have provided four sample applications to illustrate the use of our API
    - As promised, we have coded a parking application, PeterParker, which turns the Firebird into automated parking bot
    - Writing the application was made easy by the use of our API, since we only had to code up the GUI and application logic on Android

5. *Modularity For Easy Extensibility*
    - Our code is designed in modules which enables easy extensibility by tweaking any module independently
    - For example, if the Bluetooth communication hardware is not available, then the user can implement his/her own communication interface by writing a new CommunicationModule class, and with no changes to the other parts of the API
    - The Firebird Command Interpreter code can be modified to work on a different robot, without any changes to the Android API or the communication protocol. This will allow same API to control different robots (eg hexapod) using an Android phone
    - The Android API itself can be implemented for various API levels, so that is configurable for use with any Android device (especially the innovative Aakash tablets)

b) What we added more than discussed in SRS?

1. *Message Queueing*
    - We implemented queueing of messages on both Firebird as well as Android
    - Initially we were facing losses and unreliable transfer of data over Bluetooth due to latency of the BT link
    - By implementing message queues and designing the application to send packets in bursts, we are making full use of the BT channel, and this gives a reliable and efficient communication channel

2. *MapperBot Application*
    - We have implemented the MapperBot application which enables the Firebird to be used as a spy bot
    - This application makes extensive use of the API, and also its logic is nontrivial
    - Sensor values are continuously read from the bot, while all logic is implemented on the Android device, and the command for next action is computed
    - Also, as the bot moves, a map of the bot's path is drawn on the Android display, which can be later emailed to a remote address
    - This app showcases the power of the Android API for building non-trivial and useful applications for the Firebird

c) Changes made in plan:

- It was expected that having a single buffer on the firebird to store and process the received input would be sufficient.
- But as it turned out, the bluetooth module also sent control data, and due to processing delay in the interrupt service routine, some of the commands were dropped
- So we came up with a new scheme involving 2 buffers.
- The ISR would add the received data to one buffer and we would process the input in the main loop and add the extracted commands in a second "command buffer" which would be used by the executor module.

7) Future Work:

a) Extensions in the code:

- Add modules for alternative communication modes. e.g Zigbee, Serial Communication, Wifi, etc. and add functionality to control these from the Android
- Modify our API to work for any other similar bot (e.g Hexapod)

b) More Applications:

Any general application can be developed using our API. Some specific app ideas are:

Apps involving mounted android phone on Firebird

1. Surveillance Application
   - A surveillance app which allows the user to select locations on a map
   - The android mounted on the bot will determine its position using GPS and find a route connecting these points
   - The bot will now move along the route taking pictures on its way or relaying the video

2. Greenhouse Pest Control Application
    - Cameras are mounted in various parts of the greenhouse for surveillance of pests
    - When the camera detects a pest attack, it sends a message to the Firebird
    - The Firebird then goes to the the location of the pest infected area.

Apps which can control multiple Firebird bots

- This can be accomplished through WiFi module on the Firebird or a central server which communicates to Android devices sitting on the Firebird
- **Multiple Parking Bot**: In a parking arena, where there are multiple bots waiting in a queue to park, the Android can control these bots separately and park all of them in the arena

8) Conclusions:

- Basically, every application on the Firebird is now an Android app!
- The code on the Firebird is burnt only once before distributing
- This enables users to use the Firebird without having much knowledge about its hardware
- An Android user can concentrate on developing more complex apps rather than worrying about the hardware
- Hopefully our project will give rise to a new community of active Android developers aiming to build innovative apps for the Firebird

9) References:

- Android Developer Reference
  http://developer.android.com/reference/packages.html
- USB interface tutorial covering basic fundamentals bluetooth
  http://www.eeherald.com/section/design-guide/esmod14.html
- An Introduction to Bluetooth Programming
  http://people.csail.mit.edu/albert/bluez-intro/
- E-Yantra
  http://www.e-yantra.org
- Swiss Army Knife Project
  http://www.e-yantra.org/