CS308 - 2014 - Final Report

Localization of bot in an arena based on Sensor - Fusion

TU-01
T Ashok Kumar - 100050083
Prajwal A N - 100050085
Karthik KSS - 100050072
Vamsi Krishna - 100050080 ©

Project Website: https://github.com/prajwal-aithal/localization-on-firebird/

Table of Contents

1. Introduction

This document is the final report for the project “Localization of bot in an arena based on Sensor - Fusion”. This project deals with the problem of localization using multiple sensors. We try to find the location of the bot based on the sensor information at the location of the bot, which in our case is the wifi strength (in terms of RSSI value) of different routers at the bot’s location.

The problem of indoor localization is a difficult problem to solve. This kind of system has a lot uses in indoor locations. Some of such uses are - localizing location of a bot in an arena and taking action according to the position (useful in a greenhouse, malls, and other similar settings), scanning the arena for fields at different locations (though not through localization this problem can be solved by the system built through this project), studying effects of multiple sensor fields in an arena, derivatives of the above problems, etc. This motivated us to take up this project.

The system consists of a bot (Firebird V), a computing system (python shell), a wifi shield, and a communicating module (XBee in this case). The working of this system can be briefly explained as follows - the wifi shield connected to the bot requests for RSSI values from the various routers on demand from the bot and returns the same to the bot. The bot communicates these values to the computing system through the communication module. The computing system can do data analysis on this data and use the data as it wishes. We currently solve the problem of localization by using the data sent by the bot and trying to fit it to a model (Naive Bayes) learnt by the bot using sample data.

The organization of this document is as follows. Current section deals with introduction (abstract, terms, definitions, references). The next section gives the problem statement and the items delivered. The 3rd section explains details such as functionality, design constraints, and interfaces being used. The 4th section explains the overall design of the system. The 5th section explains the working of the system and some test results. The 6th section provides some insight into the development of this project. The penultimate section discusses the possible future directions for this project. We then conclude this report with some conclusions.

1.1. Definitions, Acronyms and Abbreviations

- Localization - The process of localizing the user location using different parameters.
- FireBird V - The bot used in the project.
- XBee - The communication module used in the project (to communicate between the bot and the computing device).
- Wifi Shield - The wifi module used in the project. This connects to the routers on demand from the bot and returns RSSI values for each router after the connection is established.
- RSSI - Received Signal Strength Indicator. This is a metric measuring the power present in a received radio signal.

2. Problem Statement

The problem statement is to achieve localization using multiple sensor information. The purpose of the product is to use various parameters (like layout definition, wifi strength, proximity sensors, etc) to localize the bot. On successful localization, we can navigate the bot to a target location. This could be used in locations such as greenhouses to navigate from one location to another with minimal human intervention. An ideal usage scenario would be to navigate from one location to another in a greenhouse to transport essential ingredients (like manure, chemicals, etc).

A by-product of this project is the interface on the bot to read sensor signals (currently supported sensors are Wifi, XBee, and wheel encoder). This interface could be used to create a signal field of a layout which itself has interesting applications in various problems.

3. Requirements

3.1. Functional Requirements

The functional requirements are:

- At a given location, the bot should be able to collect various sensor information (like proximity sensor information, wifi signals and strengths, wheel encoder information).
- Using the above information the system should be able to derive the location of the bot.
- Given a layout, the bot should be able to move from one location to another aided by the sensors on the bot.

3.2. Hardware Requirements

The Hardware requirements are:

- FireBird V - This is the bot that we are going to use to demonstrate this project.
- XBee interface - We need this to communicate with the computing system.
- Wifi Sensors - Needed to approximate the strength of the wifi servers at the current location. The current implementation of the system uses a **Wifi shield** in place of these sensors.
- Proximity sensors - Needed to sense any obstacles in the immediate path of the bot. These are going to be fit onto the FireBird bot. Though the current implementation of the project does not use these sensors, the future plans are to add these sensors and try to improve the localization.

Pictures of these components are present in Section 4.

3.3. Software Requirements

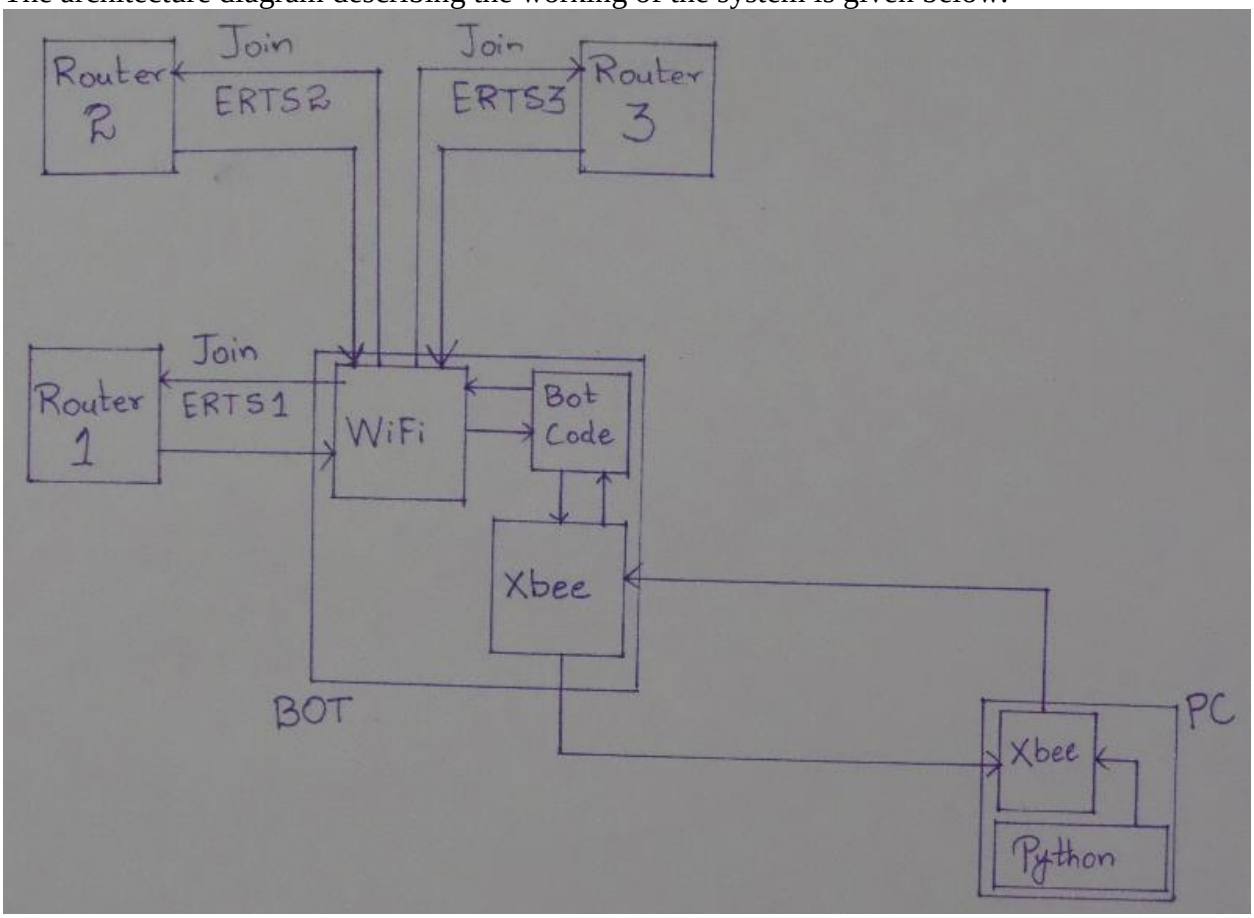
The software requirements are:

- Keil - Software to work on and build the bot code.
- Python - The computing system runs a python script to carry out localization. We are using Python 2.7.
- Scipy - Needed as a prerequisite to Scikit-Learn.
- Numpy - Needed as a prerequisite to Scikit-Learn.
- Scikit-Learn - This is the Machine Learning module. We can model the localization

system as different Machine Learning models using this module.

4. System Design

The architecture diagram describing the working of the system is given below.



As shown in the diagram.

- The bot consists of the Wifi shield and the XBee components.
- The computing system runs the python script.
- When the python script requests for the RSSI values, it sends the instruction through the XBee 1 module. The bot receives this instruction via the XBee 2 module (on the UART0 channel), and requests the Wifi component (on the UART1 channel) to ping the routers and return the RSSI values. The Wifi shield does the same and returns the values to the bot which then forwards it to the computing system via the XBee 2 module.
- The python script receives the above values and analyzes them to return the location of the bot.

The system design can be analyzed using the following interfaces.

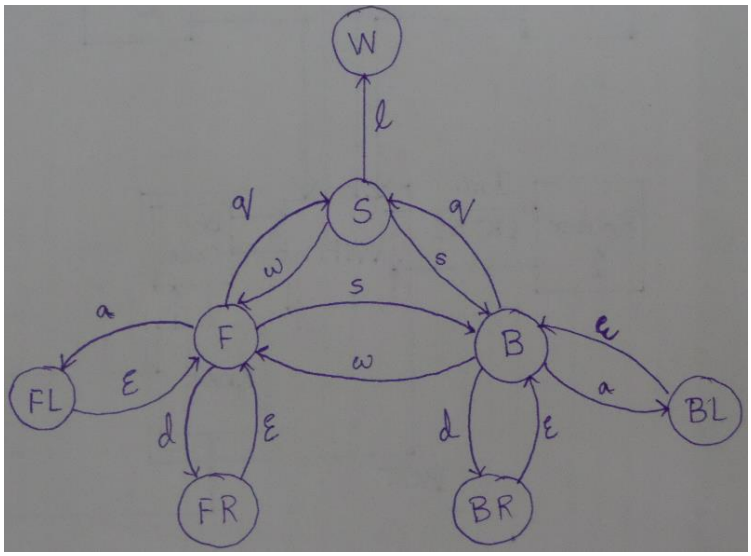
- **Localization Module**: This is the module that takes care of the localization. The function that does this is the *get_coordinates()* function in localization.py. This module can be further broken down into the following modules.
    - RSSI Interface: This is the interface to get the actual RSSI values. The function

that does this is the *connect_get_RSSI()* function in localization.py.

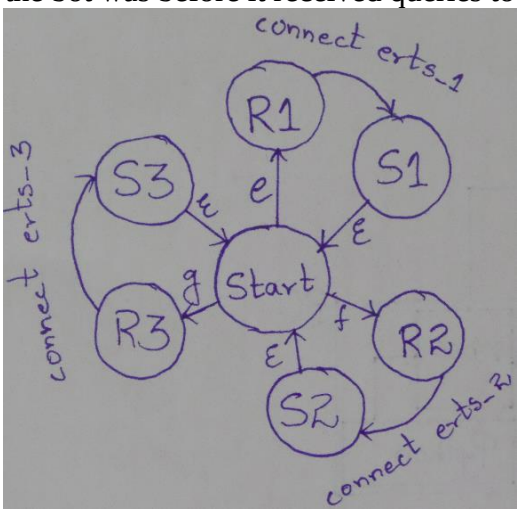
- RSSI calculation interface: This is the interface to calculate the aggregate RSSI value at a location. The function that does this is the *get_RSSI()* function in localization.py. This calls *connect_get_RSSI()* required number of times and collects the RSSI values from each router. It then calculates the RSSI value of each router at the bot by taking the average of the range of values that have the least variance.
- Location Prediction: This is the interface that takes values returned by the *get_RSSI()* function and predicts the location. Currently this is done by modelling the system as a Naive Bayes model and trying to fit the values into this model. The function that does this is the *predict_label()* function in localization.py. As can be seen the localization module has been coded in such a way that in case another algorithm is implemented in the future, the changes need to be made are minimal (change the *predict_label()* function to implement the new algorithm).

- **Navigation Module**: This is the module that takes care of navigation. Navigation on user command has been supported and the commands can be found in the initial message that the system displays (alternatively the function *init_print()* in localization.py contains the commands supported by the system). The module does not yet support automatic navigation given the source and the target coordinates.
- **Wifi Module on the bot:** This is the module that connects the Wifi shield to the routers and returns the RSSI values to the bot. This sends instructions on the UART1 channel to connect to the appropriate router ("*join ERTS_<number>*") and returns the messages that router sends to the Wifi shield through UART0 channel. A piece of code that does this can be found from Line 316 to Line 320 in the *IRQ_UART0()* function in main.c.
- **Communication Module**: This is the module that is responsible for the communication between the bot and the python script. On the bot side, the XBee is present on the UART0 channel, and the functions that take care of communication are *IRQ_UART0()*, *UART0_SendStr()* and *UART0_SendByte()* in main.c. The commands accepted by this module are commented in the *IRQ_UART0()* function.
  In the python script a variable (of type Serial) connects the port on which the XBee module is connected to the computer and this variable takes care of communication. The variable *ser* in localization.py is the variable, and the function *read()* and *write()* take care of reading and writing to the port respectively.

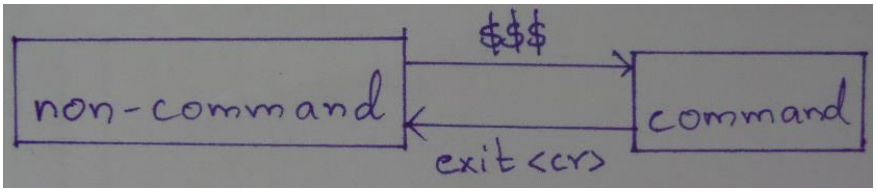
The main system behavior can be illustrated using the following FSM. This illustrates the states of the bot and corresponding actions. The states are F - forward, B - back, L - left, R - right, S - stop indicate the latest action the bot was instructed to do, and the state W indicates that the bot is in signal collection state.



The collection of RSSI values from the routers can be expressed as the following FSM. The states here are R1, R2, R3 indicating the latest router to which the bot has requested RSSI values, S1, S2, S3 indicating the latest router from which the bot has received RSSI values and Start indicating the state in which the bot was before it received queries to provide RSSI values.

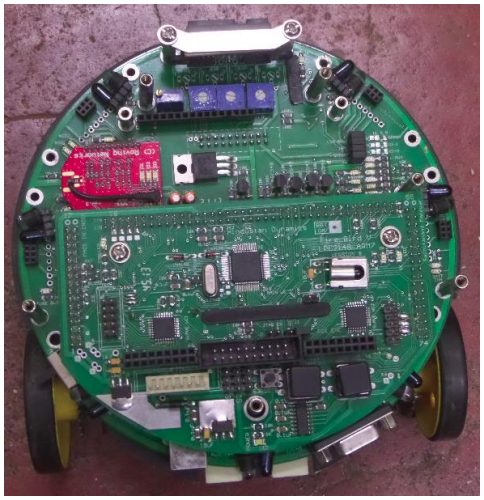


For the bot to reciprocate to the instructions sent, it should be in command mode. The FSM representing the state of the bot is as follows.

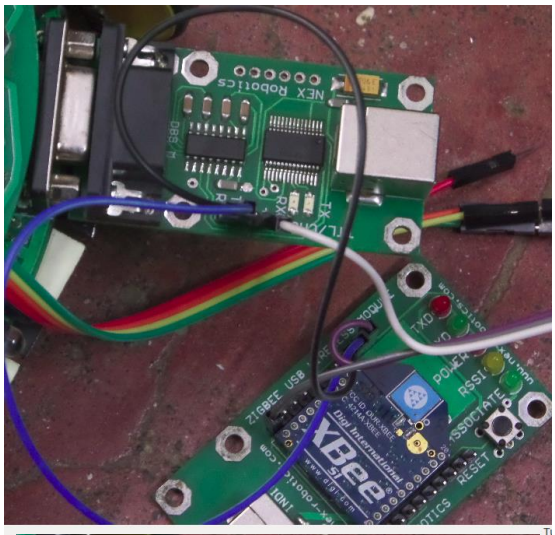


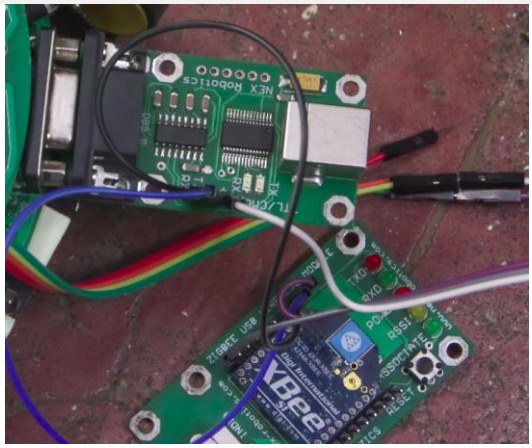
To make the bot move into the command mode, the script sends a “c” to the bot. The comments in the function *IRQ_UART0()* illustrate the working on the bot.

The following are the snapshots of the hardware components used in the project.

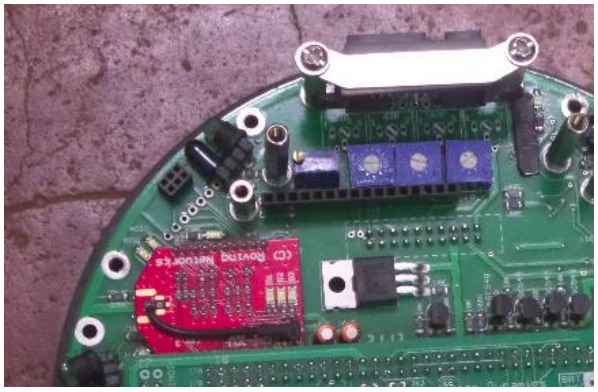


FireBird V

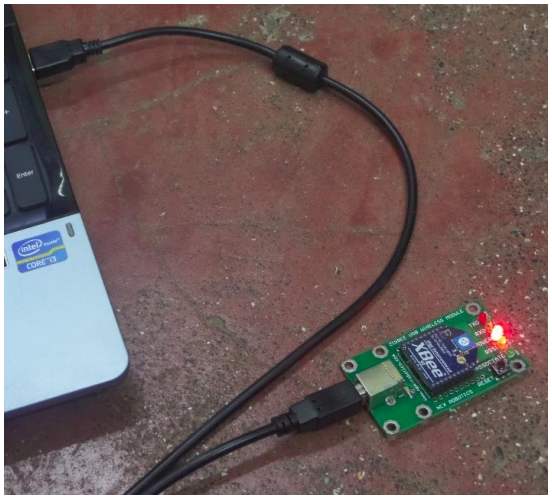




XBee on the bot



Wifi Shield



XBee connected to the PC

Wifi Router

5. Working of the System and Test results

The control flow of the system has been discussed above and has been illustrated in the architecture diagram in Section 4. We now discuss how the analysis of the RSSI data is carried out in our system.

Our localization module uses machine learning to predict the coordinates of the bot given the RSSI values. The model used is a Gaussian Naive Bayes model. The sample data can be found here. The format of the sample data can be found here.

The setup of the system for testing is as follows. The arena was an 8X8 grid, with each cell being 20cm X 20cm. There were 3 routers used for localization. The sample data was collected as follows. At each cell, we collected the RSSI values for each of the 3 routers three times. We did this for all the 64 cells. We use this data as training data.

The status of the promised functionalities is as follows.

Functionality	Status
Collection of various sensor information.	The system currently has interfaces to collect Wifi strength and support communication through XBee. Wheel encoder information has not been supported yet.
Localization of the bot using the above information.	The system localizes the bot using the Wifi strength (in terms of RSSI values). The accuracy has been discussed below.
Automated navigation of the bot in the given layout.	This module has been developed independently and has not been integrated into the system yet.

The testing was carried out as follows. The bot was placed in a random cell of the grid and the system was run. The distance between the output of the system and the actual position is the error of the system. When tested immediately after the sample data was taken, the accuracy was around 90%. When the settings were changed, the error ranges from 20cm to 90cm from the original position.

6. Development of the Project

The components delivered as promised are:

- Wifi strength determination: This module has been implemented and has been explained in section 4.
- Navigation module: This module has been implemented independently, but has not been integrated into the system and tested.
- Multiple sensors: Currently only wifi shield has been used to localize the bot. The other

sensors that could be used are proximity sensors, and wheel encoders. These have been mentioned in the future work section below.

The changes made in plan from SRS are as follows:

- The initial focus of the project was to automate the navigation and localization process. Due to problems in setting up the bot for our project (as explained below), we lost some time and had to shift the focus on just localization.
- Thus the navigation module was dropped, and localization was carried out with the help of Wifi sensors only.

The challenges that we faced during the development of the project were:

- We were given Wifi shield for the purposes of Wifi strength collection. We tried to communicate with the PC using the Wifi, but were not able to establish connection. We thus decided to use a different communication module (Bluetooth or XBee in our case).
- The XBee module was originally connected to the UART1 channel. But the Wifi shield used up this channel. This resulted in a problem of installing the XBee on the bot. We finalized on using Bluetooth for communication.
- The Bluetooth module was highly unstable. The module took a lot of time to connect with the PC. And the connection was highly unstable lasting for a maximum of 10 seconds.
- Thus we decided switch to XBee for communication. We had to use the UART0 channel. We modified the code to achieve the same.
- The bot leaves command mode unexpectedly. This transition is difficult to figure out. We solved it using a coding work-around and it works most of the time.
- It takes a lot of time to connect to the router and retrieve RSSI values. This forced us to drop support for collection of RSSI values while the bot is in motion.
- The RSSI values are highly dependent on the environment. The RSSI values are mostly dependent on:
    - Electromagnetic/Radio signals
    - Interference between the router and Wifi shield
    - Room temperature
    - Orientation of the bot w.r.t the router. This forced us to limit the orientation of the bot throughout the testing and sampling phase.

7. Future Work

The future directions of this project are:

- Analyze the effect of objects in the arena on the field.
- Analyze the effect of number of routers and their placement in the arena.
- The current arena is a flat surface. We need to analyze the working of the system in uneven surfaces (which is a highly possible use case).
- Analyze the error in the system when introduced in a greenhouse where there are a lot of interferences (like plants, sprinklers, etc).
- The Wifi interface can be used in Network Strength Analysis. This has applications in a lot of places (example, finding out where/which floor in a mall).
- We can use more sensor information (like wheel encoder data and proximity sensors) to increase the accuracy of localization. We can improve the localization algorithm itself by

using standard algorithms like Monte-Carlo algorithm.

- We can use the Wifi interface to find the field map of the arena.
- Orientation of the bot can be found out using an electronic compass.

8. Conclusions

Though we started to build an automated localization system along with navigation support, due to problems in both hardware and software support, we were not able to achieve all of our objectives. We built an interface to get RSSI values from the routers and communicate the same to a computing device. We also studied the behavior of the Wifi field w.r.t external stimuli. We also implemented a Machine Learning model for localization and have presented its working, and shortcomings. We have also explored and proposed a couple of possible future directions for the project.

9. References

- Received Signal Strength Indication, http://en.wikipedia.org/wiki/Received_signal_strength_indication
- Monte-Carlo Localization, http://en.wikipedia.org/wiki/Monte_Carlo_localization
- Tutorial to connect Wifly to the router, http://cairohackerspace.blogspot.in/2011/05/beginners-guide-to-connecting-and.html
- Wifly datasheet, https://www.sparkfun.com/datasheets/Wireless/WiFi/WiFlyGSX-um.pdf