ECE498: Senior Capstone Project I

Project Proposal

Project Title: ZigBee–Based Indoor Robot Localization and Mapping

Kyle Hevrdejs and Jacob Knoll
Advisor: Dr. Suruz Miah

Electrical and Computer Engineering Department
Caterpillar College of Engineering and Technology
Bradley University

Table of Contents

1 Introduction

As the use of mobile robots becomes more common in the workplace, the need for accurate navigation rises. There are systems currently on the market that aim to meet this need. The major drawback however is the cost, modularity between mobile robots, and scalability to larger environments. In this project, we propose a cost–effective, modular, and easily scalable system utilizing off the shelf components that will allow a mobile robot to navigate and map its environment using an EKF-SLAM (Extended Kalman Filter - Simultaneous Localization and Mapping) algorithm. This system will allow navigation and mapping of an indoor environment with an acceptable level of accuracy. Applications of this system include mining vehicles, research and education, deliveries within office builings, cleaning robots, and warehouses.

2 Background Study

A large amount of research in the field of robotics is concentrated in the areas of localization and navigation (see [1], [2], [3]). Currently, much of the work being done focuses on the use of passive RFID tags [1][2]. These methods are highly accurate but also expensive, both in hardware and computational efficiency. In [1] and [2], an algorithm is implemented using UHF (ultra high frequency) RFID in order to localize a mobile robot. In there, the phase information is adopted in order to calculate the angle between the passive tags and the mobile robot. Overall, the system worked well, however, the experiments performed did not fully demonstrate the localization algorithm, as the mobile robot was preprogrammed to follow a line marked on the floor of the environment. This approach also requires understanding of RF such as phase shift and RFID. Another method, seen in [3], shows a similar system but instead using wireless sensors as beacons. This setup is easier to implement on a large scale due to the minimal RF knowledge required and ease of implementation of beacons. The system presented in [3] also uses the technique of partitioning the environment into a grid in order to more accurately calculate signal strength and bearing using a forward mounted directional antenna. This approach shows accuracy over after long operation times, an undesired characteristic of our project. Although not directly used for localization, the work done in [4] shows a possible alternative method for localization and mapping. By utilizing a rotating reflector, the authors were able to accurately estimate the angle-of-arrival of wireless signals from multiple wireless sensors. Furthermore, there are similar systems currently in use by the military that implement this technique (rotating radar antennas). There are also similar systems available for civilian use in areas such as commercial marine radar. Using this method, the authors were able to estimate the angle-of-arrival with an accuracy of $\pm 3^{\circ}$.

In this project we aim to combine a rotating reflector system similar to the one presented in [4] and the widely used EKF-SLAM algorithm. The major challenges of this implementation will be obtaining a high enough accuracy in the range and

bearing measurements. As more interference enters the environment, the system will become less accurate. This can, however, be reduced by the introduction of more beacons, at an increase in total cost.

3 Standards and Patents

The standards that are applicable to our project include the IEEE 802.15.4 protocol and the ZigBee protocol. The IEEE 802.15.4 protocol[1] outlines the physical layer (PHY) and the medium access control layer (MAC) specifications for low-data-rate wireless communication in personal area networks (LR-WPAN). The PHY layer contains the radio frequency transceiver and its low-level controller. The MAC layer provides access to the physical channel for transfer and defines an IEEE address for the device. Its goal is to maintain a simple and flexible protocol while also being easy to install, having reliable data transfer, being low cost, and having reasonable battery life. The ceiling mounted beacons will use XBee modules to communicate wirelessly with the mobile robot and uses the 802.15.4 protocol by default. The 802.15.4 protocol allows for a star network topology which is made up of one established personal area network (PAN) coordinator and other functional devices. In our system, the XBee mounted on the mobile robot will act as a coordinator and the ceiling mounted XBees will be the other functional devices. During our testing, we have established basic peer-to-peer communication with this protocol.

The other standard known as ZigBee[2][3] is an open global standard for low-power, low-cost, low-data-rate, wireless networking based on the IEEE 802.15.4 standard. It maintains the specifications of the PHY layer and the MAC layer while defining the higher network layer and setting the framework for the application layer. The network layer takes care of network structure, routing, and security. The application layer has an application support sub-layer (APS) which defines various addressing objects and has other features for device specific applications. The ZigBee protocol allows for star, peer-to-peer, and mesh network topologies. We will still use the star network topology which is made up of a coordinator who creates a network with a channel and a unique PAN ID and end devices who will join this network with extra power saving features. The XBee mounted on the mobile robot will act as the coordinator and the ceiling mounted beacons will act as the end devices. The fully functional system will be implemented with a ZigBee network. It offers well defined packetized communication, power saving features which is important for the battery powered ceiling mounted beacons, and future applications which may require a higher level mesh network.

There are several patents that outline designs that are comparable to our project. The first patent [5] describes systems and methods for localization of a mobile device

[1] IEEE Standard for Low-Rate Wireless Networks, 2015
[2] Zigbee Pro Stack User Guide, 2014
[3] XBee/XBee-Pro S2C ZigBee User Guide, 2016

using beacons. This design uses a computer, a location signal receiver and beacons to determine the location of the mobile device. The beacons transmit the location signal and the computer determines the location from the received location signal strength and a unique key based on the particular transmitting beacon. This patent is relevant to our work as it uses received signal strength from beacons and individual beacon IDs for localization of a mobile device in an indoor environment, but the patent is more general to what the mobile device is and uses a different localization algorithm. The second patent [6] outlines positioning of an indoor mobile robot, and mapping of its environment and the beacons it detects. The mobile robot performs several positioning modes depending on the number on beacons it detects in its line of sight. This patent is more applicable to our project than the last patent. It utilizes a simultaneous localization and mapping technique for a mobile robot and beacons. However, the patent's positioning method is different than ours. The third patent [7] describes a localization method for a mobile robot with a system of beacons. It uses the signal strength and angle of arrival of the signal to localize and determine a coarse correction. This design is very similar to our approach by uses signal strength and angle, and similar localization algorithm, but we will be rotating a antennae reflector instead of the entire robot. A similar patent [8] to the previous explains an indoor localization method for a mobile device using directional reflectors. The mobile device transmits a signal and the signal is delayed and returned by reflectors in the environment. The location of the mobile device is then determined from the range by the amount of delay and the angle of arrival at the reflector. This patent is relatively similar to our project as it describes an indoor localization technique using range and angle for a mobile device.

4 Functional Requirements

4.1 System Architecture

For our project, we consider a scenario shown in Figure 1 where a mobile robot is deployed in a 3-dimensional environment with beacons distributed throughout. The mobile robot will also be given a list of waypoints located on the floor of the environment. Due to its widespread use in research applications, we have chosen the Pioneer 3-DX to implement our proposed project.

4.2 Block Diagrams

The system we propose for our project is fairly self–contained in the sense that it will handle its operation once it has started. This consideration will further promote modularity when the system is used with other mobile robots. A high level system block diagram can be seen in Figure 2. Many of the outputs shown are used for debugging purposes as well as being used for the proposed algorithm. The subsystem block diagram of the system shown in Figure 2 can be seen in Figure 3. Due to the nature of the algorithm, we arranged our subsystem block diagram to resemble a con-

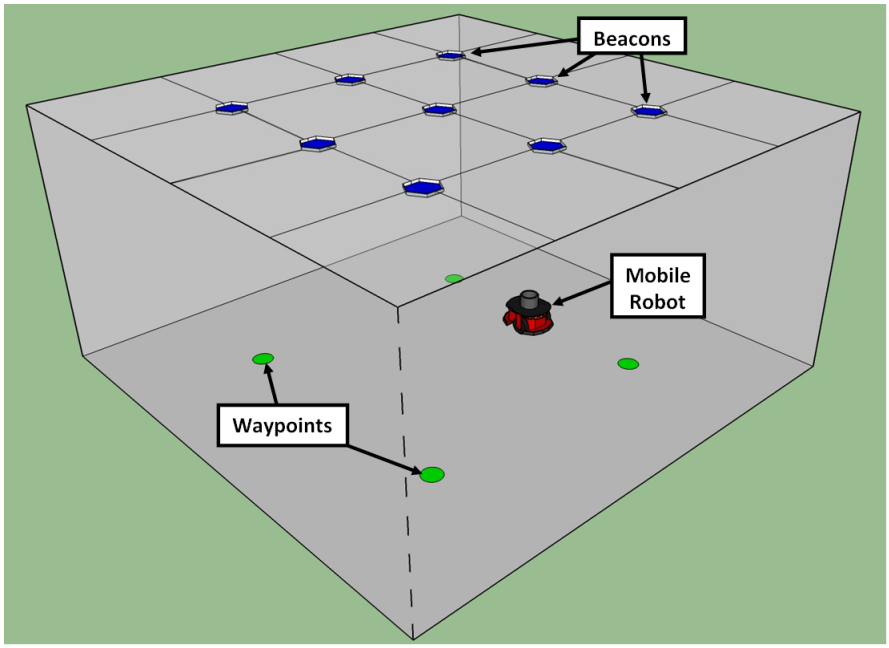


Figure 1: Environment with mobile robot, ceiling mounted beacons, and waypoints

ventional control system. This helps to better illustrate how data is being used within the system in order to localize the mobile robot and map the indoor environment.

As shown in Figure 3, the outputs labelled *estimated pose* and *estimated positions of XBees* are also used internally. This feedback design allows the system to use prior data and predictions to estimate the mobile robot's position with more accuracy each iteration. At the top of Figure 3, there are three separate subsystems. The first is the controller, which will be further explained in subsection 5.1. The internal values output by this subsystem are the linear velocity and the steering angle. These values are then used as inputs to the preprocessor subsystem where they are converted to

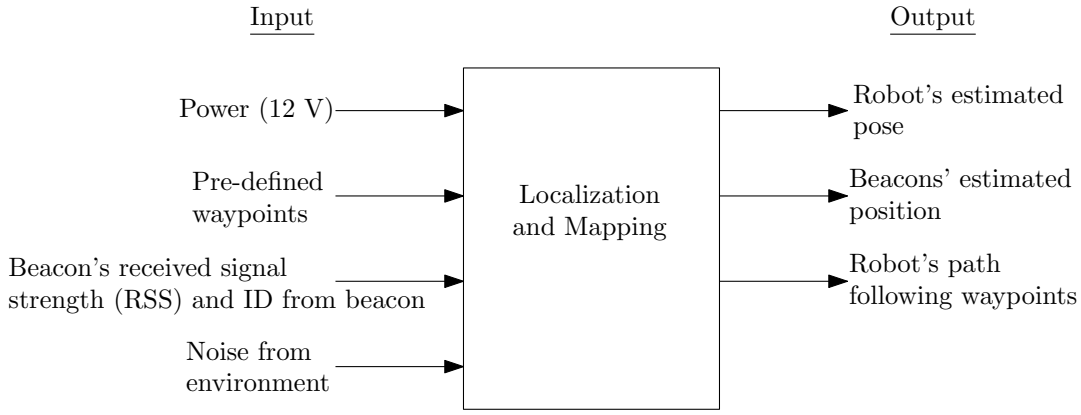


Figure 2: System level block diagram detailing inputs and outputs to the system

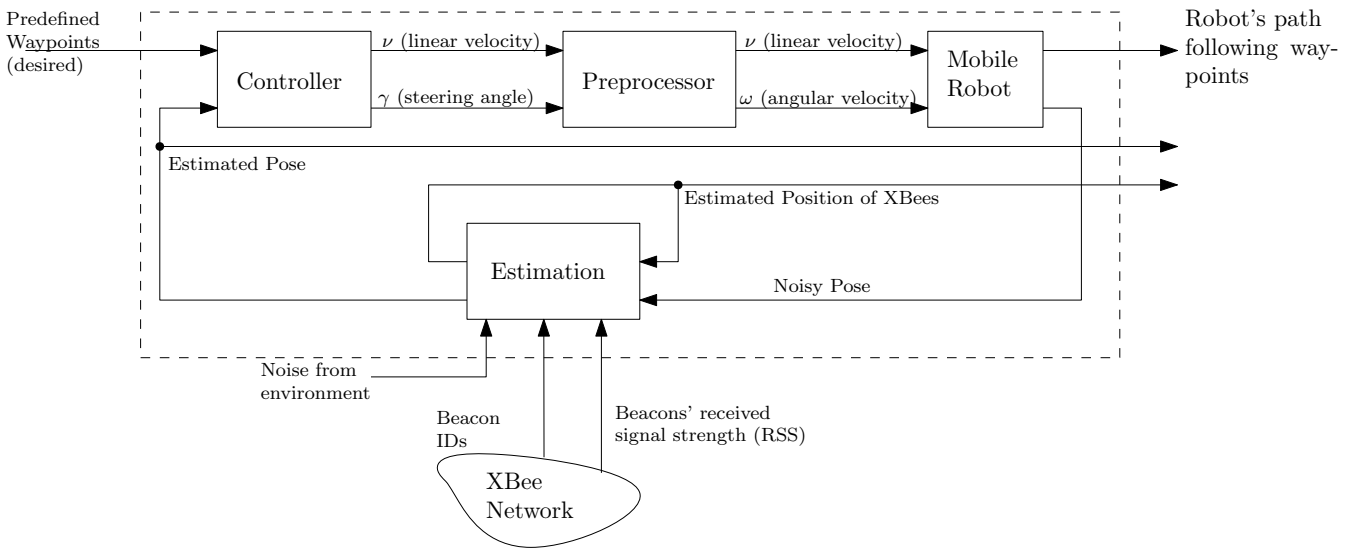


Figure 3: Subsystem level block diagram

control inputs accepted by the Pioneer 3-DX. This conversion is detailed in subsection 5.1. Once these control inputs are applied and the mobile robot has completed its movement, the noisy pose is sent to the estimation block where it is used to estimate the mobile robot's true pose. In this subsystem, the mapping of the XBee beacons within the environment is also done. The estimated pose is then used by the controller in order to repeat the process. As seen in Figure 3, the estimated position of the XBees is stored internally for use in the next iteration.

4.3 Specifications

For the proposed system, there are three main specifications that should be met. First, the overall cost should be less that $500 in order to stay within the guidelines set by the department. Secondly, the mobile robot must be able to be localized within 30 cm of its true position. Thirdly, the position estimates of each beacon should be within 30 cm of its true position with respect to the indoor operating space of the robot.

5 Preliminary Work

We began working in the lab as soon as we knew we had been assigned this project. We are both interested in the fields of robotics, controls, and embedded programming and are motivated to complete this project. The following sub-sections will cover the work we have done so far. It includes the modelling of mobile robot, system simulation results, the current and future design work, and some experiments we have run in the lab.

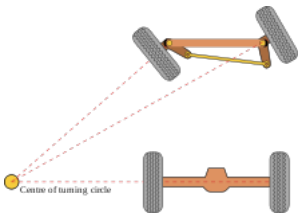


Figure 4: An example of the Ackermann steering vehicle

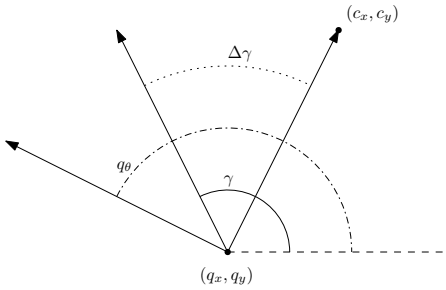


Figure 5: A simple illustration of the relationship between the robot's orientation, current steering angle, and the change in steering angle

5.1 Modelling

Since the algorithm for EKF-SLAM is widely-used and well known, we did not need to create the model from scratch. We based the model used in the controller subsystem on an Ackermann steering vehicle, shown in Figure 4. This model is only used for calculating the steering angle of the mobile robot with pose $\mathbf{q} = [q_x, q_y, q_\theta]^T$ where (q_x, q_y) is the robot's 2D position and q_θ is its orientation with respect to the global coordinate frame. This model will calculate the necessary steering angle to converge toward its current waypoint with location $\mathbf{c} = [c_x, c_y]^T$ where . In this model the linear velocity is kept at a constant value for simplicity. The steering angle is calculated using

$$\gamma = \gamma + \Delta\gamma \tag{1}$$

where $\Delta\gamma$ is the change in the steering angle calculated by

$$\Delta\gamma = \tan^{-1} \frac{c_y - q_y}{c_x - q_x} - q_\theta - \gamma \tag{2}$$

An illustration of the relationship between these angles can be in Figure 5.

Since a differential drive mobile robot is being used in the project, the steering angle γ must be converted to a usable value, in this case angular velocity, ω. The angular velocity is calculated by

$$\omega = \nu \frac{\sin\gamma}{\ell} \tag{3}$$

where ν is the linear velocity and ℓ is the distance between the wheels.

5.2 Simulation Results

For the simulation of our algorithm, we chose to use the free, multi-platform robot simulator V-REP[4]. The V-REP robot simulator follows industry standard robotic platforms in order to produce consistent theoretical results in a way that is accurate to their real-world counterparts. This simulator is popular within the robotics community due its integrated development environment, large number of robot models available, the ability to write in a variety of programming languages, its ROS compatability, and overall ease of use.

The model presented in the previous section was then implemented in V-REP. To obtain these theoretical results, the provided Pioneer 3-DX model was used for the simulation. Using V-REP's remote API, the Pioneer model is able to be controlled through MATLAB.

Since the algorithm operates under the assumption that there will be noise in the environment, we added Gaussian noise in MATLAB. In our simulation, noise is introduced as process noise and observation noise, each with a noise covariance matrix. The process noise covariance matrix is represented by

$$\mathbf{Q} = \begin{bmatrix} \sigma_\nu^2 & 0 \\ 0 & \sigma_\omega^2 \end{bmatrix} \tag{4}$$

where σ_ν is the standard deviation of linear velocity calculated by $\sqrt{\frac{1}{T^2} 2(\frac{1\pi r}{3N})^2}$ where r is the radius of the wheels, T is the time step, and N is the number of encoder ticks per wheel revolution. For the Pioneer we use the following values: $r = 0.075\,\text{m}$, $T = 0.25\,\text{s}$, and $N = 500$ which gave us the calculated value of σ_ν to be 8.8858×10^{-4}. σ_ω is the standard deviation of angular velocity. For this simulation we chose a value of $\sigma_\omega = 0.0012\,\text{rad}$.

The observation noise covariance is represented by

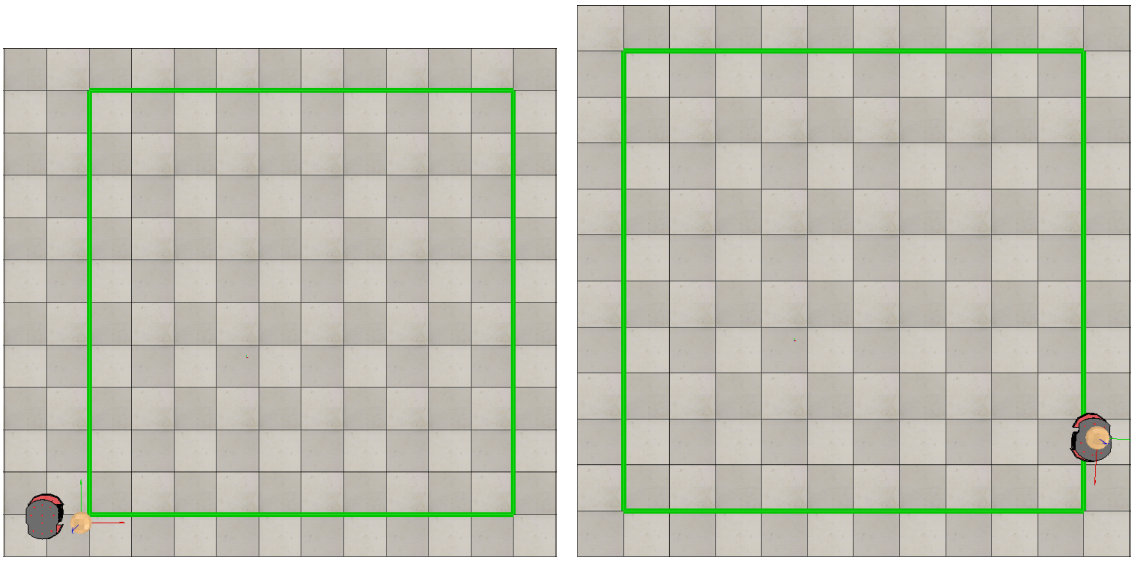
$$\mathbf{R} = \begin{bmatrix} \sigma_R & 0 \\ 0 & \sigma_B \end{bmatrix} \tag{5}$$

where σ_R is the standard deviation of the range measurement and σ_B is the standard deviation of the bearing measurement. We chose $\sigma_R = 1\,\text{m}$ and $\sigma_B = 0.0873\,\text{rad}$ for the simulation.

Using these values, the simulation was run for 2000 iterations with the Pioneer starting 0.5 m to the left of the desired starting position, in order to show the algorithm more clearly. This starting position can be seen in Figure 6. In these images, the floating orb follows the path of an ideal reference robot.

The error for the robot is defined as

[4] `http://www.coppeliarobotics.com/`



(a) (b)

Figure 6: Simulation showing EKF-SLAM algorithm performance at (a) $t = 0\,\text{s}$, and (b) $t = 250\,\text{s}$.

$$e_P(t) = \sqrt{(\hat{x}(t) - x^d(t))^2 + (\hat{y}(t) - y^d(t))^2} \tag{6}$$

where $\hat{x}(t)$ and $\hat{y}(t)$ are the estimated position values and $x^d(t)$ and $y^d(t)$ are the desired position values. As seen in 7(a), the error between the estimated position and the desired position is decreasing as the number iterations increases. This can also be seen when the error is plotted, as shown in 7(a). Although the error fluctuates, it is still quite low with consideration to the size of the robot and the amount of noise added in MATLAB. Using (8), the RMSE of the robot's error was calculated to be $0.0928\,\text{m}$.

Similar to (6), the error for the beacons can be calculated using

$$e_B(t) = \Sigma_{i=1}^{n} \sqrt{(\hat{x}_i(t) - x_i^d(t))^2 + (\hat{y}_i(t) - y_i^d(t))^2} \tag{7}$$

where n is the number of beacons, $\hat{x}_i(t)$ and $\hat{y}_i(t)$ are the estimated position of the ith beacon and $x_i^d(t)$ and $y_i^d(t)$ are the true position of the ith beacon.

$$\text{RMSE}_P = \sqrt{\frac{1}{t_f}\int_0^{t_f} [e_P(t)]^2 dt} \tag{8}$$

$$\text{RMSE}_B = \sqrt{\frac{1}{t_f}\int_0^{t_f} [e_B(t)]^2 dt} \tag{9}$$

Using (9), the RMSE of the beacon error was calculated to be $1.7254\,\text{m}$. This number

is deceiving since the combined error of all beacons was used, instead of a single beacon's error. For comparison, the range of calculated RMSE values for individual beacons started at 0.2047 m being the lowest and 0.5363 m being the highest.

5.3 Design

In the fully functioning system, a BeagleBone Black microcomputer[5] will control the Pioneer 3-DX, and interface with the XBee network. Several commands will be sent from the microcomputer to the robot mounted XBee to receive the necessary information to perform the algorithm. The first command will send a ping signal to the ceiling mounted XBees. When received, each XBee will set the RSSI of the ping signal into an internal register. The next command requests the ceiling mounted XBees to return its individual ID and its registered RSSI. We have written a C program to interface with the robot mounted XBee with the help of *libxbee*[6], open source C/C++ XBee interfacing library, and our testing confirms working order of this part of the design.

The planned connections between the hardware components in our design is shown in Figure 8 and a visual representation of the subsystem connections is shown in Figure 9.

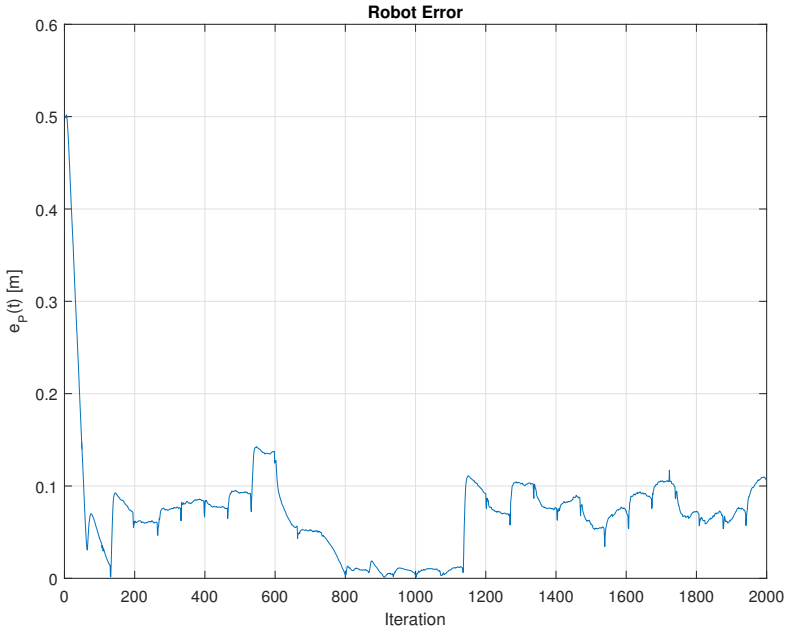
The future work required in the overall design is controlling the mobile robot with the BeagleBone Black microcomputer, and determining the angles of the beacons in relation to the mobile robot. The C program will need expand to a C++ program when implementing the Pioneer 3-DX controller. The Robot Operating System (ROS) is an open source C++ library that simplifies the code needed to control robots. The ROS library should be able to integrate into the existing network interfacing program seamlessly into one network interfacing and control program. To determine the angles of the beacon in relation to the mobile robot, we plan to implement a rotating antenna reflector for the robot mounted XBee in a similar design as [4]. This will focus the ping signal in one direction thus resulting in a higher RSSI for beacons in the direction of the reflector and lower RSSI for the beacons in other positions. After taking several RSSI measurements while rotating the reflector, we will be able to determine the angle of a beacon based on its strongest received signal strength.

5.4 Experimental Activities

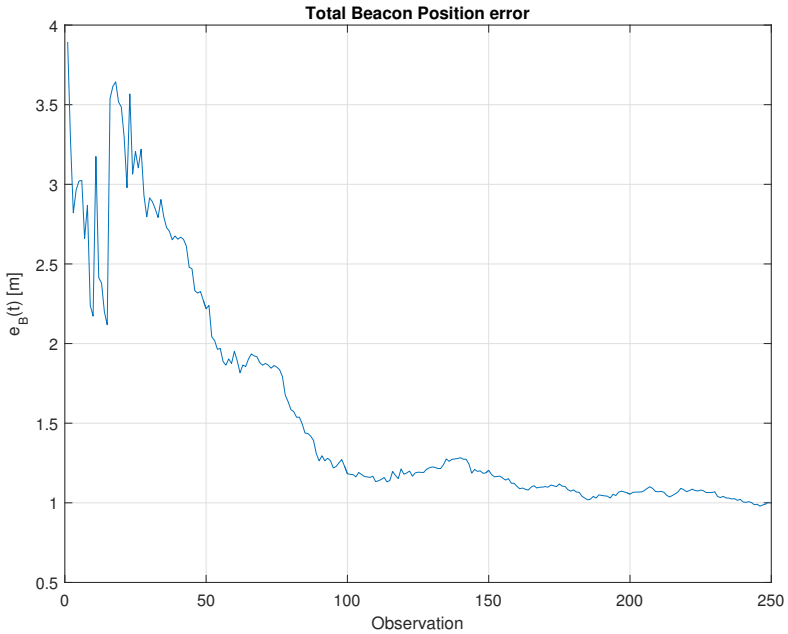
Thus far, much of our work has been based on interfacing with the XBee modules that we will be using as our beacons. In order to obtain the experimental data shown in Figure 10, we set up two XBee modules to run in API mode. Our beacon (the static XBee module) was powered with a power supply and placed on the floor. Our transmitter (the stationary XBee module) was connected to a BeagleBone Black running

[5] `https://beagleboard.org/black`

[6] `https://github.com/attie/libxbee3`



(a)



(b)

Figure 7: Plots showing the error in (a) the robot's position and (b) the estimated positions of the beacons

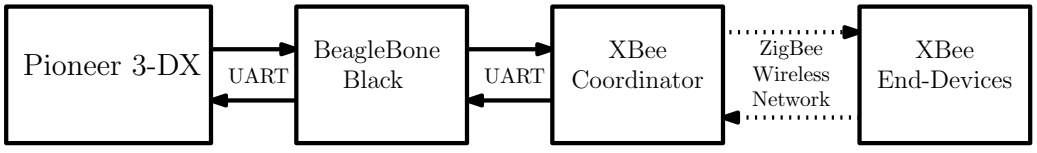


Figure 8: Hardware connections between subsystems

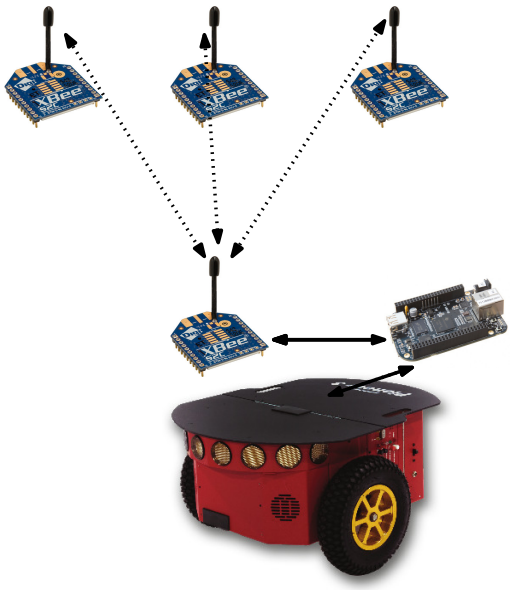


Figure 9: Visual representation of our design

the *libxbee* library. This module was powered directly from the BeagleBone Black connected to a laptop computer. The transmitter was then moved away from the beacon in set intervals and the RSSI was recorded at each interval. This experiment was repeated three times, for three different power levels, represented by *PL* in the legend of Figure 10.

Since the units used to measure RSSI are difficult to determine, we used the following equation to convert to a distance in meters

$$d = 10^{\frac{-\text{RSSI}-P_{\text{ref}}}{10\eta}} \tag{10}$$

where d is the calculated distance in meters, RSSI is the measured signal strength in *dBm*, P_{ref} is the absolute value of the RSSI at 1 m, and η is the propagation constant. For our experiment we found that $P_{\text{ref}} = 60$ and $\eta = 4$. Due to our limited RF knowledge, we do not know how to accurately find the propagation constant but found that the aforementioned value gave a reasonable estimation of the distance. The data in Figure 10 was converted using (10). This data can be seen in Figure 11.

It can be seen that the power level has little effect on the RSSI measurements at various distances. However, data gathered past a distance of 2.5 m becomes irregular. We believe that this is due to the environment where the experiments were done. In the lab there are two large metal cabinets against one wall. For our experiment we started from the opposite wall and moved the beacon toward the cabinets. We believe these cabinets introduced some interference and therefore altered our data. A possible solution to decrease the amount of interference is a reflector around the transmitting antenna in order to decrease the amount of backscattered signal we receive. Another easy change we could make would be using a very high power level, since the data

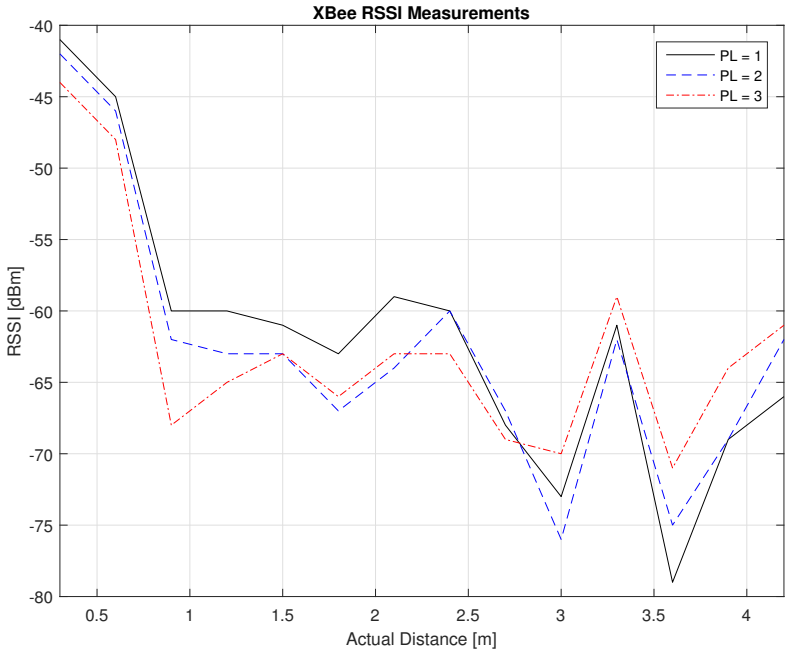


Figure 10: XBee ranging data plotted as RSSI vs. actual distance

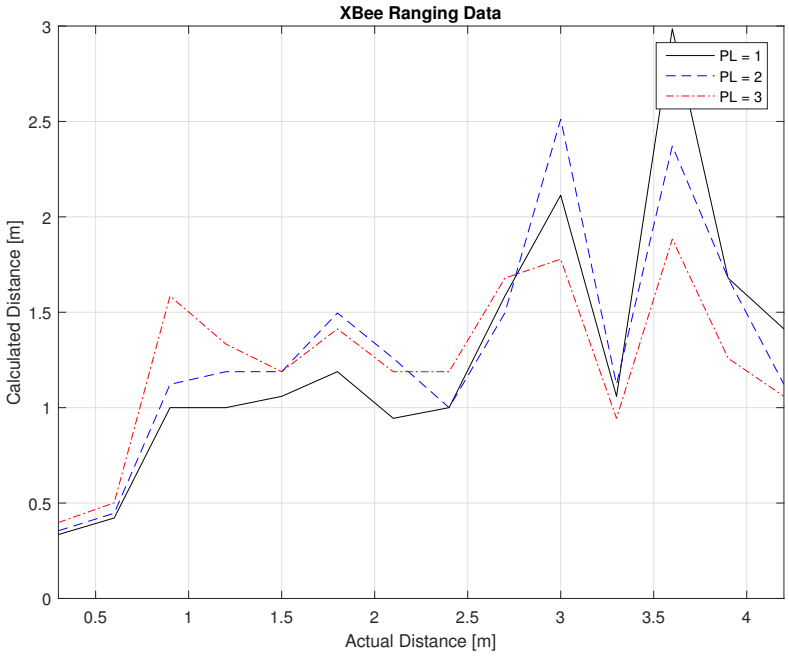


Figure 11: XBee ranging data plotted at calculated distance vs. actual distance

Part	Price	Quantity
XBee S2C w/ whip antenna	$18.19	6
XBee Interface Board	$5.31	6
Perforated Circuit Board	$2.84	4
Stepper Motor	$14.95	1
Motor Controller	$19.95	1
3.3 V Regulator	$0.79	10
9 V Battery Clip	$0.39	10

Table 1: List of ordered parts

seen in Figure 11 suggests that a higher power level is less susceptible to interference.

6 Parts List

We were fortunate enough to have access to some of our necessary parts as soon as our project began. These parts include 4 XBee S2C modules, 2 with chip antennas and 2 with whip antennas, XBee interfacing boards, a BeagleBone Black, and USB-to-serial cables. This allowed us to perform experiments using the XBee modules without placing a part request. However, now that we are starting to implement more of the system we need more parts. Not including the parts we already have, a list of parts that are to be ordered can be seen in Table 1.

7 Timeline and Milestones

The Gantt chart in Figure 13 shows our planned schedule to complete this project. Each month is split into 4 pieces represented by the dotted grid to approximate weekly schedule. The first milestone is to complete the necessary work for our angle estimation technique. This includes building an antenna reflector, then testing the reflector to obtain optimal results. The expected functionality is taking several RSSI samples while rotating the reflector a maximum of 180° and using the RSSI sample data to determine an estimated angle. The next milestone is to have full functionality of the mobile robot controller using ROS running on a BeagleBone Black. Controlling the Pioneer 3-DX with MATLAB will simplify testing and control, but our end goal is to have the system self-contained on the mobile robot platform. The last milestone is to integrate the subsystems into one fully functioning system. This will require that each subsystem has met its specifications and is ready to implemented. We expect that testing the system will require debugging and tuning, thus we have set aside a large amount of time dedicated to testing. We have also set time for completing documentation and any other work necessary to complete our project.

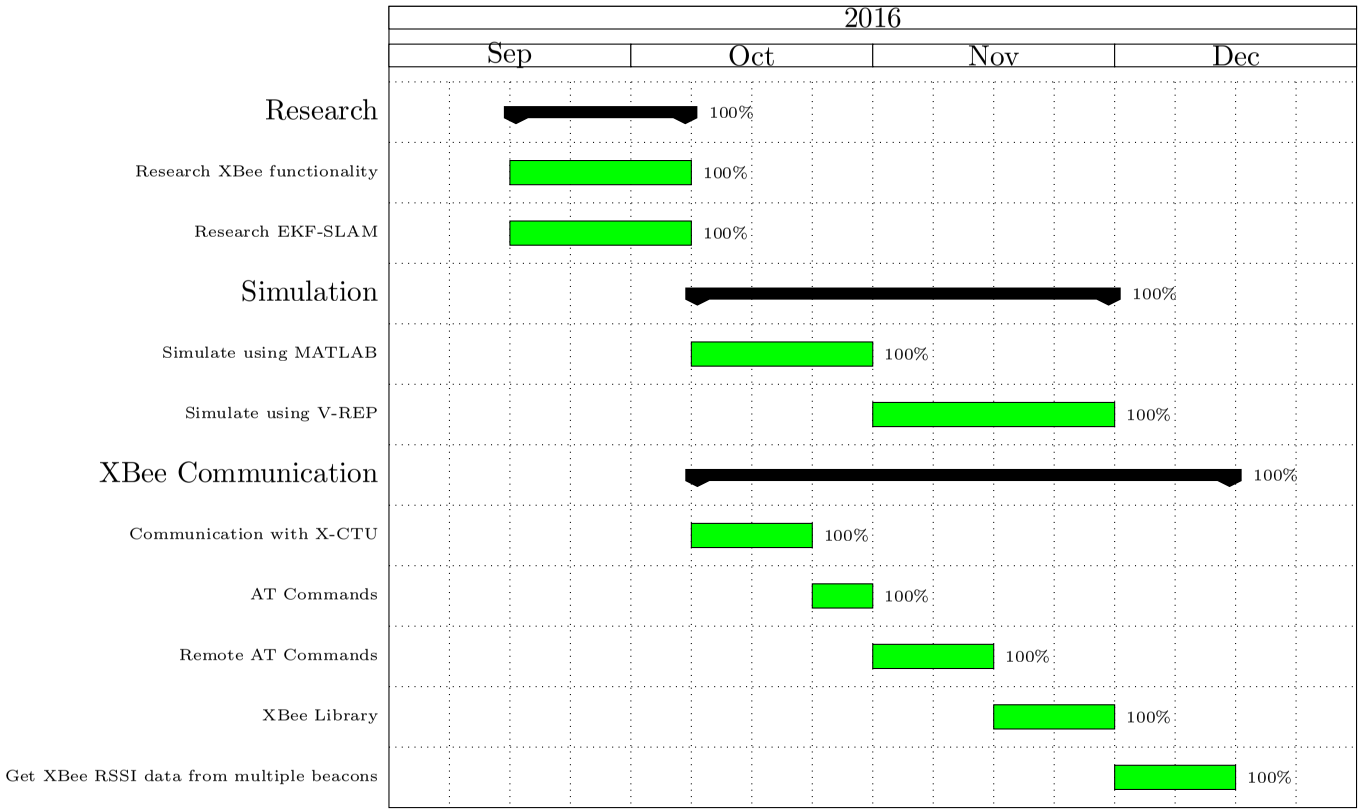


Figure 12: Gantt chart for Fall 2016

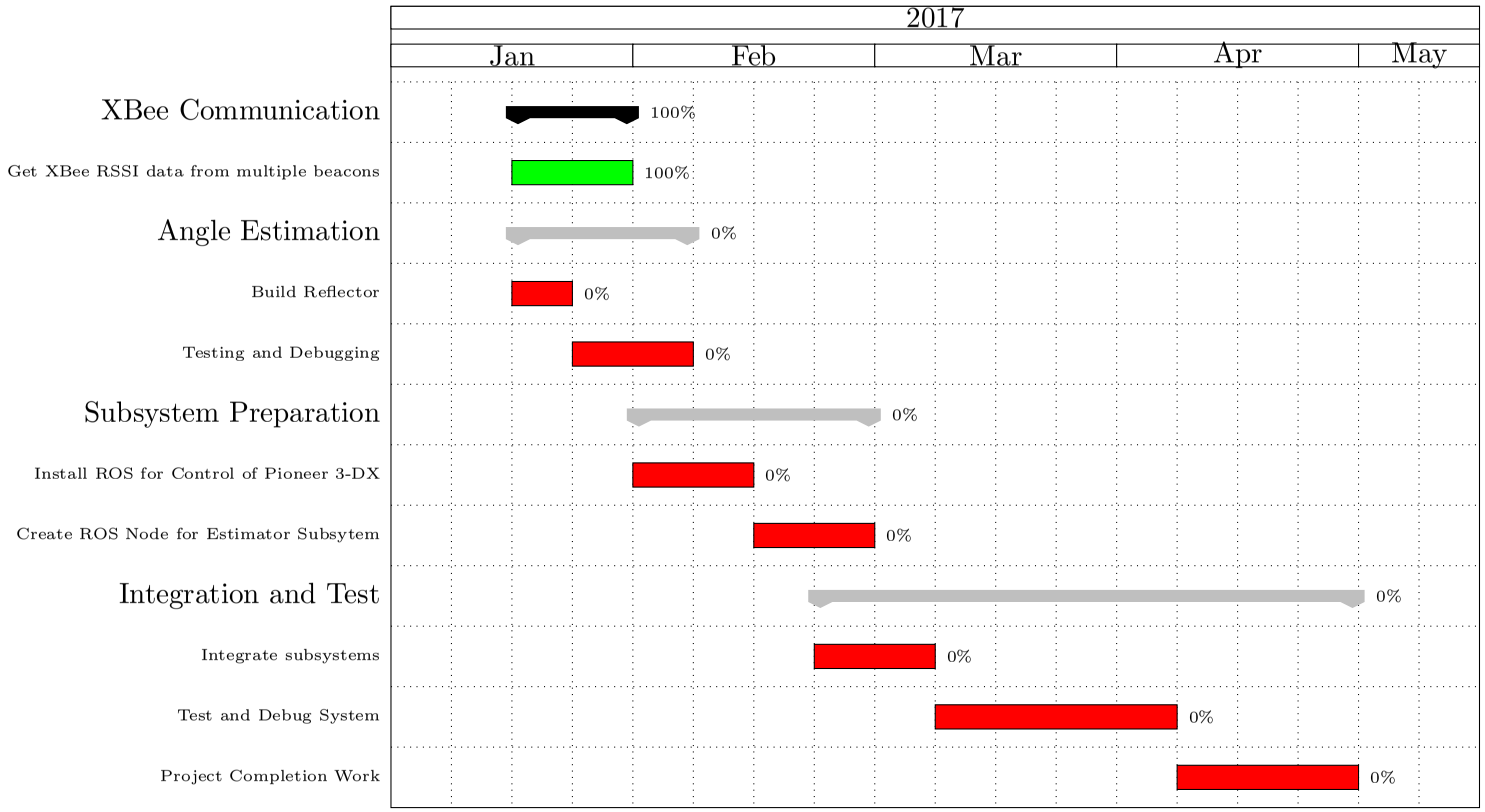


Figure 13: Gantt Chart for Spring 2017

References

[1] F. Martinelli, "A robot localization system combining rssi and phase shift in uhf-rfid signals," *IEEE Transactions on Control Systems Technology*, vol. 23, no. 5, pp. 1782–1796, Sept 2015.

[2] E. DiGiampaolo and F. Martinelli, "Mobile robot localization using the phase of passive uhf rfid signals," *IEEE Transactions on Industrial Electronics*, vol. 61, no. 1, pp. 365–376, Jan 2014.

[3] D. Song, C. Y. Kim, and J. Yi, "Simultaneous localization of multiple unknown and transient radio sources using a mobile robot," *IEEE Transactions on Robotics*, vol. 28, no. 3, pp. 668–680, June 2012.

[4] B. N. Hood and P. Barooah, "Estimating doa from radio-frequency rssi measurements using an actuated reflector," *IEEE Sensors Journal*, vol. 11, no. 2, pp. 413–417, Feb 2011.

[5] J. Biehl, M. L. Cooper, and G. Filby, "Systems and methods for mobile device location verification using beacons," U.S. Patent 9 491 588, Nov. 8, 2016.

[6] C.-K. Lin, H.-P. Chen, Y.-L. Ting, S.-C. Hsu, and C.-W. Chen, "Indoor robot and method for indoor robot positioning," U.S. Patent 9 442 177, Sep. 13, 2016.

[7] L. Edlund, R. Berlin, and C. R. Davidsson, "Method for localization of beacons for an autonomous device," U.S. Patent 5 682 313, Oct. 28, 1997.

[8] S. G. Younis, "Indoor position location using delayed scanned directional reflectors," U.S. Patent 9 383 441, Jul. 5, 2016.