Indoor Robot Localization and Mapping using ZigBee Radio Technology

Kyle Hevrdejs and Jacob Knoll
Advisor: Dr. Suruz Miah

October 31, 2016

1 Functional Description of the Project

In this project, we will implement a localization and mapping algorithm using a differential drive wheeled mobile robot operating in an indoor environment. The mobile robot will be equipped with an onboard microcontroller for implementing the proposed localization and mapping algorithm. The mobile robot is supposed to receive signal strength information from a set of ZigBee[1] radio modules (called herein active beacons) mounted on three-dimensional (3D) coordinates (ceiling of the robot's workspace, for example). The proposed localization and mapping algorithm is expected to determine the position and orientation (pose) of the mobile robot and 3D coordinates of active beacons (in this project, these are XBee modules[2] that support the ZigBee protocol). Appropriate actuator commands are then applied for the mobile robot to follow a set of two-dimensional waypoints within the robot's workspace. The information from the active beacons will allow the mobile robot to estimate the locations of beacons and its pose in an indoor environment simultaneously using an extended Kalman filter[3] (simultaneous localization and mapping). Based on the estimated pose of the mobile robot and the position of beacons, the necessary linear and angular velocities (actuator commands) will be applied to the mobile robot's left and right wheels in order to follow the path defined by pre-determined waypoints.

2 Related Work

One of the major areas in the field of robotics is localization.To control any type of robot (in our case a mobile robot), certain aspects of the environment must be known or able to be detected through the use of various sensors. Generally, there are two pieces of information that are gained by localization, the current location and the current orientation. By far the most common way to gather these two values is through the use of dead reckoning. This method uses the mobile robot's onboard odomettry to measure the distance traveled and

[1]https://en.wikipedia.org/wiki/ZigBee
[2]See http://www.digikey.com/ for details.
[3]https://en.wikipedia.org/wiki/Kalman_filter

convert that to a position and orientation (pose). For small environments, this method is adequate, but once the distance traveled becomes greater is becomes less accurate. The decreasing accuracy mostly stems from error in measurements since they will accumulate over time. Another limitation to this method is that the mobile robot must know its initial position, otherwise all subsequent calculations will be wrong. Therefore, dead reckoning is usually paired with another localization method.

Over the years there has been a sizeable amount of work done to increase the accuracy of robot localization. Much of this work is based on the same algorithm, called SLAM (Simultaneous Localization and Mapping) and its variants such as EKF (Extended Kahlman Filter) SLAM and fastSLAM. Since the algorithm is already well established, it makes sense that the work centered around is focusing on new ways to achieve the localization aspect. Some of these methods include the use of RFID [4][1][2], directional antennas [5], and XBee networks [3].

The method presented in [1] uses the phase of ultra high frequency RFID signals in order to find the position and orientation of the mobile robot. The work presented shows the potential of such a system in real world use. However, due to the nature of RFID and signal processing, certain errors are produced in their algorithm. Since only two passive RFID tags were used, there were uncertainties in the localization algorithm. The system was also developed purely to test the feasability of the technique and was not used to do any sort of navigation. However this method does present a viable option for solving the localization problem.

In [4], the author seeks to improve upon the accuracy of the method shown in [1]. In this paper, the author builds upon the previous method by adding the use of phase shift measurements in combination with RSSI. This improves accuracy significantly. Another improvement from the old method is the use of more passive tags in the beacon network. However, like the previous paper, the system was implemented to simply test the method and was not used to navigate an environment and had the mobile robot follow a line marked on the floor.

Another paper presenting mapping and localization using RFID can be found in [2]. This method presents refined method of mapping an environment and localizing a mobile robot within it. Similar to the work presented in [4] and [1], RFID tags are mounted on the ceiling. The main difference is the use of two, forward-facing directional antennas. This allows for the detection of RFID tags in certain zones within a known range of angles, simplifying the calculations. This paper also used the fastSLAM method, which is more robust than regular SLAM. This allowed the mobile robot to localized itself relatively quickly. The main disadvantage of this method (and the previous two) is the use passive RFID tags that have a range of only a couple meters in optimal conditions. In a real-world scenario, beacons may not be as numerous or densely placed in an environment.

The logical solution to this issue is to change wireless technologies. In the work presented in [5], the authors replace the passive RFID tags with a large number of wireless sensor nodes. This increases the size of the operating area. The setup of this system is also simpler, utilizing only a single directional antenna for sensor detection. The advantage of this system is clearly the simplicity of its hardware which requires minimal RF knowledge. However, the diisadvantage comes in the complexity of the calculations and computing power required. It also appears that it takes more time to localize the robot as more radio sources are introduced, which is not desired.

3 System Level Block Diagrams

A system level block diagram is shown in Figure 1. The main source of power for this system will be the 12V connection available within the mobile robot. The user will also be able to supply a set of predefined waypoints to the controller, which will define the mobile robot's trajectory through the environment. In order to follow this trajectory, the system will also need the received signal strength and IDs from the active beacons. This data will be sent through a 2.4GHz wireless signal using ZigBee protocol. Since the RSS measurements are subject to environmental noise, there is a certain amount of noise picked up by the signal as it radiates from the active beacons.

Since the main function of the system is navigation, most of the outputs are used for debugging and status indication to the user. The mobile robot's estimated pose is the location the mobile robot thinks it is currently located at. The active beacons' estimated positions is similar, except it outputs the estimated positions of all active beacons within the XBee network. The last output is the mobile robot's path following waypoints, which is simply the path the mobile robot follows as it navigates to its current waypoint.

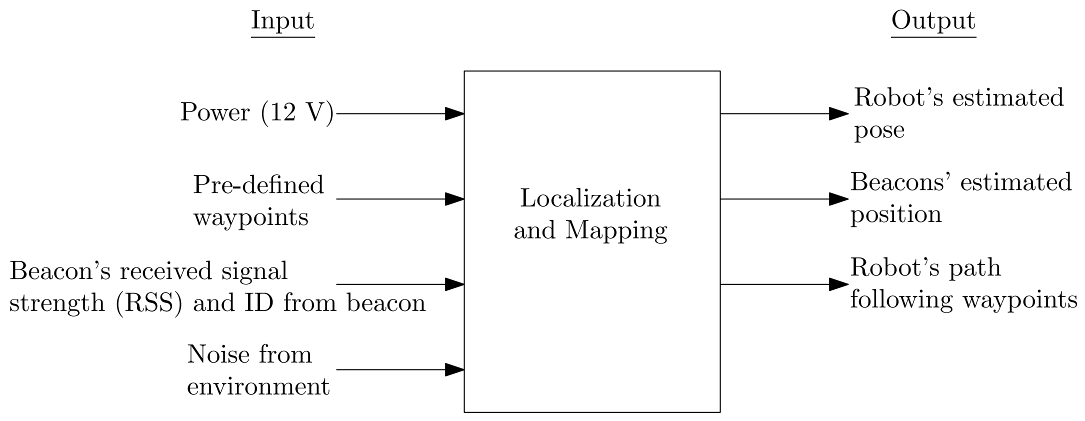
Input
Output
Power (12 V)
Pre-defined waypoints
Beacon's received signal strength (RSS) and ID from beacon
Noise from environment
Localization and Mapping
Robot's estimated pose
Beacons' estimated position
Robot's path following waypoints

Figure 1: High level block diagram of the Robot System

4 Modes of Operation

- *Initialization* - This mode will run at startup and will handle the activation/initialization of subsystems. This mode will also reset any components that require it. The only input required for this mode is power.

- *Calibration* - Once the system has been initialized, this mode will be entered. During this mode, the system will perform a preliminary discovery of all active beacons within a specified range. It will then run calibration functions to calculate necessary constants and parameters used in later modes. The two inputs required in this mode are power, beacon received signal strength and ID, and environment noise. This mode will also utilize the output for requesting signal strength and ID.
- *Discovery* - This mode will handle the discovery of active beacons in the environment. Unlike the calibration mode, this mode will run at set intervals during operation of the system in order to prevent loss of wireless signal. The required inputs for this mode of operation are power, active beacon's received signal strength and ID, and noise from environment. The output during this mode of operation is the request for signal strength and ID.
- *Navigation* - This is the main mode of the system, it will handle the navigation of the mobile robot along the path defined by the pre-determined waypoints. This mode utilizes all inputs and outputs of the system, with the outputs being displayed for status information.

5 Subsystem Level Block Diagrams

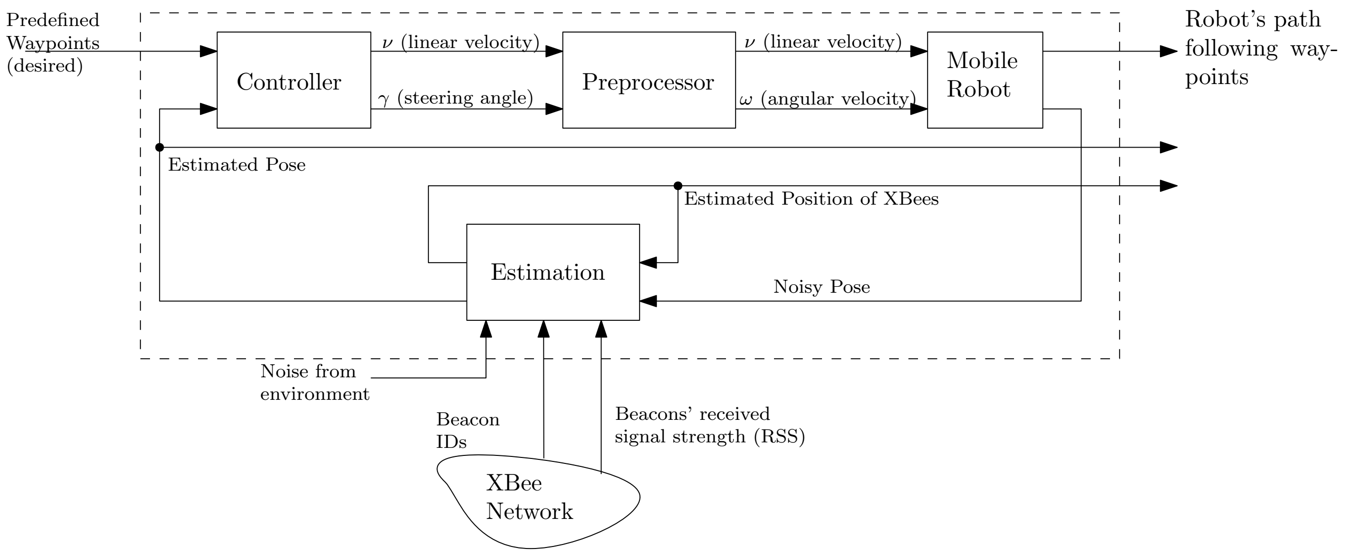

Figure 2: Subsystem block diagram of the Robot System

6 Functional Descriptions of Subsystem Blocks

6.1 Robot

This subsystem block handles the movement of the system within its environment. For our project, the mobile robot will be the Pioneer 3-DX running the ROSARIA[4] package.

6.2 Mapping and Localization

This subsystem contains the interface between the transmitter (XBee) and the Beaglebone Black microcomputer. The transmitter will communicate received signal strength indicator (RSSI) information and ID tags from the XBee netowrk to the microcomputer via UART where the estimated positions of the active beacons will be determined (mapping). Angle information of the active beacons with respect to the positive x-axis of the (x, y) plane will also communicated to the microcomputer. The implementation of getting angle information about the active beacons has not been finalized. Using the estimated active beacon positions and angles, the subsystem will estimate the position and orientation of the robot platform with respect to the (x, y) plane (localization). A flowchart showing the procedure used by the estimation subsystem can be seen in Figure 3.

6.3 Controller

This subsystem will handle the control of the mobile robot through linear and angular velocity inputs. These inputs are calculated so that when they are applied, the mobile robot will converge towards its current waypoint. Once the waypoint is within a certain distance, it will automatically switch to the next input so the mobile robot will continue to follow the predetermined path. The controller will be able to communicate with other subsystems via ROS, a widely used pseudo-operating system offering high-level abstraction from low level systems, which will simplify the communication and control aspect of the system. A flowchart showing the procedure used by the controller can be seen in Figure 4.

6.4 Preprocessor

This subsytem is not very complicated. Its only purpose is to convert the linear velocity and steering angle produced by the controller into the linear and angular velocity values needed by the mobile robot. Since this is a simple conversion, there is no flowchart included for this block in section 7.

7 Flowcharts

7.1 Mapping and Localization

The flowchart showing the process of the estimation block is show in Figure 3. The steps are detailed below.

[4]http://wiki.ros.org/ROSARIA

Step 0: This step initializes the variables for SLAM state, SLAM state co-variance, sampling time interval, data association table, data association gates, observation parameters, and others so they may be used by other parts of the mapping and localization block.
Step 1: This step handles the receiving of the noisy pose data in the form of RSSI and angle value.
Step 2: Using the information received in step 1, the EKF predict state covariance are calculated. This is the first step of the Extended Kalman filter for performing simultaneous localization and mapping.
Step 3: If enough time has passed since the last localization, localize again by continuing to step 5. If not, wait until the necessary amount of tie has passed.
Step 4: Observe RSSI and ID tags from beacons that are visible within the robot's semi-circular view.
Step 5: Associate new observations with previous observations.
Step 6: Calculate extended Kalman filter update using Cholesky factorization given the prior SLAM state and state covariance.
Step 7: Augment the SLAM state and state covariance using the data from step 6.
Step 8: Save states for next loop.
Step 9: Output estimated mobile robot pose and estimated beacon positions from SLAM state. After this step is complete, the algorithm returns to step 3.

7.2 Controller

The flowchart showing the process of the controller is shown in Figure 4. The steps are detailed below.
Step 0: The first step executed by this block is the initialization of variables. As seen in the figure, this step only occurs immediately after the block starts. This step mostly consists of environment setup and verifying connections.
Step 1: Once the block has started, it must check if this is the first iteration it has run since this step will be repeated every time the mobile robot moves. This step defines how the main loop will handle the waypoints initially.
Step 2: This step can be reached in two ways. If it is reached through step 1, it will set the new waypoint to be the first in the list of predefined waypoints (the start of the mobile robot's path). If it is reached through step 7, it will increment to the next waypoint in the list provided by the user.
Step 3: This step uses the estimated pose of the mobile robot provided by the mapping and localization block and the position of the current waypoint. The only calculations made here are the distance between the mobile robot and the waypoint and the angle between the mobile robot's current orientation and its desired orientation. This information is sent on the step 4.
Step 4: Using the information provided by step 3, this step calculates the necessary linear velocity and steering angle in order to drive to the current waypoint.
Step 5: This is the last step of the main loop. It will take the calculated linear velocity and steering angle and output them to the preprocessor block shown in Figure 2. Once this is complete, the process returns to step 1.

Step 6: This step is very simple and only handles the input of the estimated pose of the mobile robot and sends it directly to step 3. This information is obtained from the estimation block, specifically from step 9 in Figure 3.
Step 7: This step is reached every time after the first iteration. It checks to see if the mobile robot is within a certain distance to the waypoint. If the answer is yes, the next step is step 2. If the answer is no, the next step is step 8.
Step 8: When the mobile robot is not close enough to the current waypoint, this step is reached. Instead of changing to a new waypoint as in step 2, this step simply keeps the current waypoint and continues on with the loop.

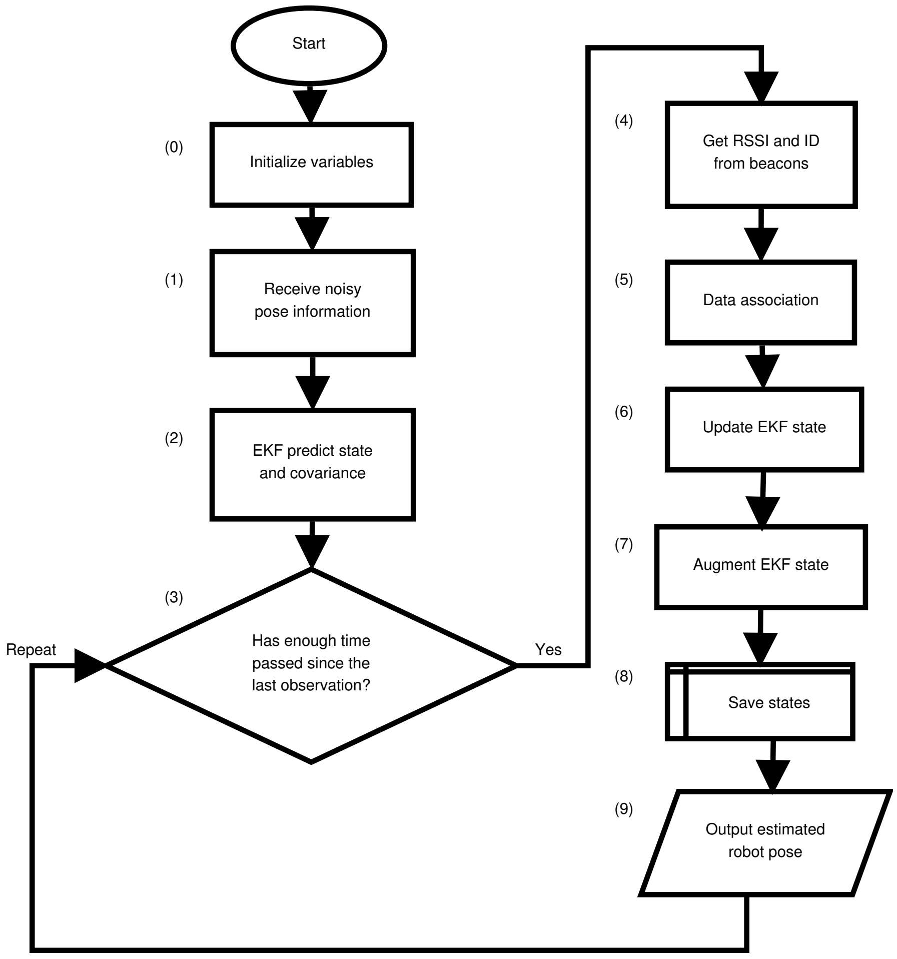
Start
(0)
Initialize variables
(1)
Receive noisy pose information
(2)
EKF predict state and covariance
(3)
Has enough time passed since the last observation?
Repeat
Yes
(4)
Get RSSI and ID from beacons
(5)
Data association
(6)
Update EKF state
(7)
Augment EKF state
(8)
Save states
(9)
Output estimated robot pose

Figure 3: A flowchart showing the process of the estimation block

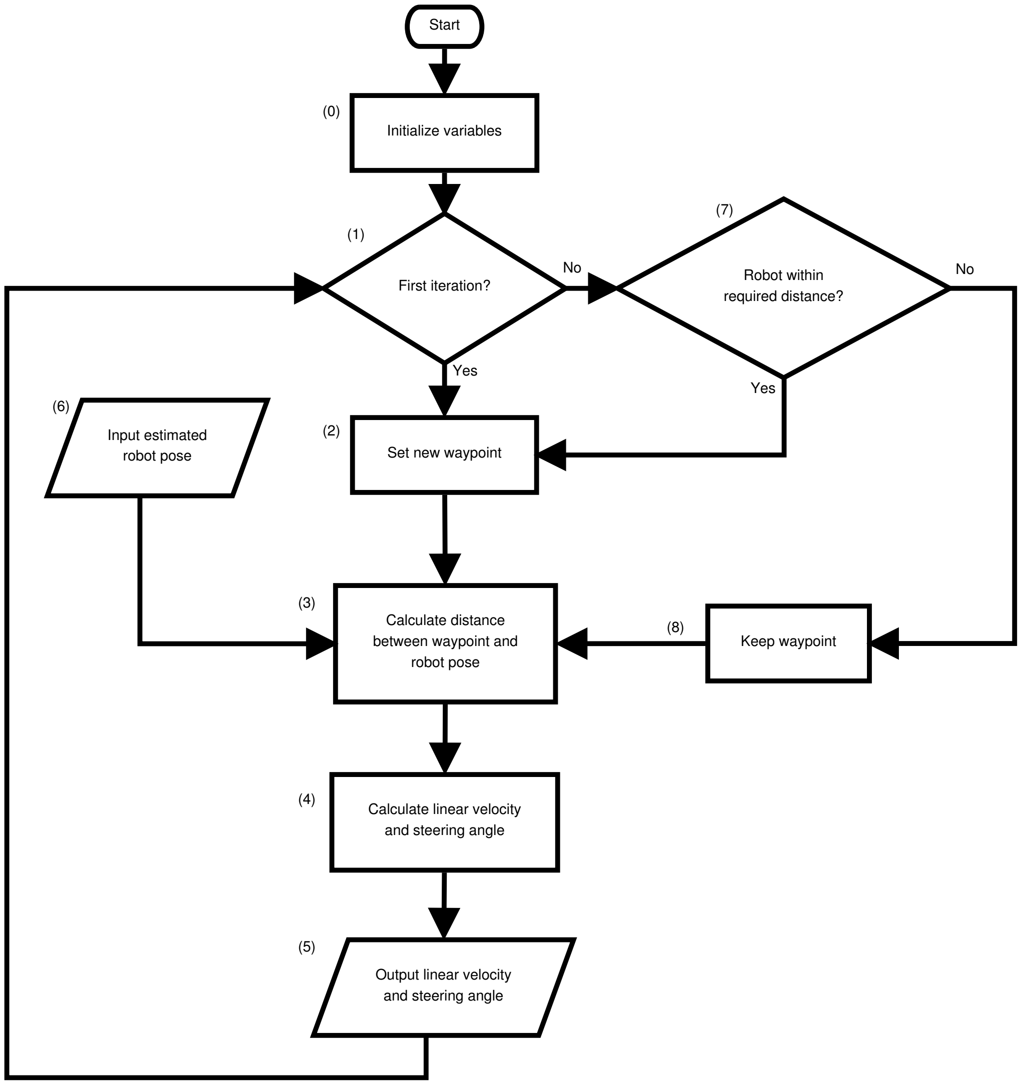
Start
(0)
Initialize variables
(1)
First iteration?
No
Yes
(7)
Robot within required distance?
No
Yes
(6)
Input estimated robot pose
(2)
Set new waypoint
(3)
Calculate distance between waypoint and robot pose
(8)
Keep waypoint
(4)
Calculate linear velocity and steering angle
(5)
Output linear velocity and steering angle

Figure 4: A flowchart showing the process of the controller block

References

[1] E. DiGiampaolo and F. Martinelli. Mobile robot localization using the phase of passive uhf rfid signals. *IEEE Transactions on Industrial Electronics*, 61(1):365–376, Jan 2014.

[2] D. Hahnel, W. Burgard, D. Fox, K. Fishkin, and M. Philipose. Mapping and localization with rfid technology. In *Robotics and Automation, 2004. Proceedings. ICRA '04. 2004 IEEE International Conference on*, volume 1, pages 1015–1020 Vol.1, April 2004.

[3] B. N. Hood and P. Barooah. Estimating doa from radio-frequency rssi measurements using an actuated reflector. *IEEE Sensors Journal*, 11(2):413–417, Feb 2011.

[4] F. Martinelli. A robot localization system combining rssi and phase shift in uhf-rfid signals. *IEEE Transactions on Control Systems Technology*, 23(5):1782–1796, Sept 2015.

[5] 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*, 28(3):668–680, June 2012.