LOCALIZATION IN NOISY ENVIRONMENT

USING EXTENDED KALMAN FILTER

by

ANEEKET SURESH PATKAR

Presented to the Faculty of the Graduate School of

The University of Texas at Arlington in Partial Fulfillment

of the Requirements

for the Degree of

MASTER OF SCIENCE IN ELECTRICAL ENGINEERING

THE UNIVERSITY OF TEXAS AT ARLINGTON

December 2007

ACKNOWLEDGEMENTS

I would like to thank my parents and family for all the love and care they have showered on me. I especially am thankful to my dad who has more belief in me than myself and who has influenced my life in a big way. I am thankful to my good friend, Prasanna Ballal, and Dr. Lewis for providing me the opportunity to work with them. I would like to thank Dr.Stephanou and Dr.Popa for their co-operation in allowing me to work at ARRI. I am thankful to Dr.Pritpal Dang, Prasanna and Abshishek Trivedi, for my thesis work would not been completed without their assistance. I would like to thank Norman Spayd for helping me out quite a few times and once in particular. I would like to thank the staff at ARRI including Kathleen and Dana for their help and providing a nice work environment.

I am grateful to Dr.Lewis, Dr.Graff and Dr.Liang for taking time out to attend my thesis defense.

I am indebted to Prasanna, Mohammed Mysorewala, Sandeep and Rahul for the numerous times that they have offered rides to ARRI. I am especially indebted to Rahul for the helping hand that he has extended whenever it was needed.

Last but not the least, I am thankful to all the friends who have made my stay at UTA a memorable part of my life.

I dedicate this thesis to my grandmother.

November 26, 2007

ABSTRACT

LOCALIZATION IN NOISY ENVIRONMENT

USING EXTENDED KALMAN FILTER

Publication No. _____

Aneeket Suresh Patkar, M.S.

The University of Texas at Arlington, 2007

Supervising Professor: Dr. Frank Lewis.

Localization is an important aspect of Wireless Sensor Networks. Information regarding the position of the sensor nodes is not always known. Without the position information of the sensors, the data reported by the sensors is of little use. Various approaches have been used to perform localization using some information about the sensor node. Potential field approach for localization, using distance information has been successfully tested with satisfactory results.

However in case of noisy environment, the range measurements have greater inaccuracy. In such cases, localization using the above algorithm can provide some inaccuracy. To rectify such erroneous localization situations, Extended Kalman Filters

are used to estimate the position. The Extended Kalman Filter has been used as the process for estimation of coordinates is a non-linear process. The EKF is a recursive filter which only needs the information from the previous state to predict the next state.

My Contributions to the thesis :

1. Programmed the Cricket in TinyOS to store the distance measurement in arrays and then broadcast the same over radio to other crickets. The cricket programmed as a base station only listens on the radio and then send the received message to the PC over the serial UART.
2. Created a LabVIEW application which processes the message received from the cricket base station and deciphers the message to extract the distance measurements. The node id is used to identify the transmitting node and the distances are stored in corresponding arrays. The ranging information in then written into a file.
3. Created a LabVIEW application to read the distances stored in the file and then arrange the readings depending on the number of nodes.
4. Implemented the Extended Kalman Filter Localization algorithm for relative and absolute localization algorithm.
5. Implemented V-shaped swarm behavior demonstration using three Garcia as the robot and cricket as the motion guiding tool.

TABLE OF CONTENTS

Appendix

LIST OF ILLUSTRATIONS

LIST OF TABLES

CHAPTER 1

INTRODUCTION

1.1. Introduction

In the recent years Wireless Sensor Networks (WSNs) have become prevalent in a lot of places such as environmental monitoring, warehouse monitoring, hazardous environment, to name a few. A WSN is essentially a network of sensors which monitor the environment for a physical quantity such as light, heat, humidity, etc. The sensors periodically sample the environment for the quantity and generally report the readings to a central mote, called the base. The base then processes the data from various sensors and decisions are made to take a set of actions.

In spite of having the cutting edge sensor technology, if the network can not point the location of the reporting sensor node then the network will not be able to complete the assigned set of tasks, in case of the reported event having occurred. This is where localization comes into picture. Localization is the process of assigning a fixed position co-ordinates to the nodes in the network based on measuring some aspect of the relative location of the nodes.

Various localization methods have been developed over the years, using the distances or angles or time of flight of radio signals to perform localization. The approach to localization using Potential Field to simulate a virtual force field is one of

the popular methods. This method localizes any number of nodes, with a minimum of three, when the distances between the nodes are known.

1.2. Problem Statement

The potential field approach to localization initially assigns random coordinates for the node locations and then tries to minimize the error between the actual and estimated distances between the nodes. The relationship between the coordinates and the distances is nonlinear. The approach does not take noise into consideration. Noise affects the distance readings and leads to inaccurate distance measurements which then lead to incorrect localization. A minor error in the distance measurement can lead to larger inaccuracy in the localization as the method assumes that distances between the nodes are accurately known and that is the only information available prior to the localization.

To overcome the effect of noise, some mechanism must be implemented which can filter out the effect of noise in the measurements. Thus, even if noise is present, the localization is performed accurately.

The Kalman Filter is exactly such a filter. It recursively calculates the process states, in this case the node coordinates, to minimize the error. The Kalman filter, though, is applied to linear processes and the localization involves a nonlinear relationship between the process states, position coordinates, and the process, measurement. The Extended Kalman Filter which extends the Kalman filter to nonlinear processes can be used instead. This thesis is an implementation of Localization using

EKF so that the potential field theory for localization can be applied even in noisy environments. Localization has been performed for a set of three nodes. The theory supports more than three, but a minimum of three, node localization. Both relative and absolute localization can be performed. In absolute localization, the coordinates, for a standard frame of reference, of one node are known before performing the localization along with the distances between the nodes.

1.3. Outline

The second chapter introduces localization in WSN and lists various aspects such as methods for localization, various ranging techniques. The third chapter discusses the potential field approach to localization. It provides the necessary formulae for the theory and also provides the algorithm for performing both relative and absolute localization. The fourth chapter sheds light on the Kalman filter theory as well as the Extended Kalman Filter theory and the filter formulation for the pertinent case of localization. The fifth chapter presents the results of the experiments. The results of localization for two three-node formations have been presented along with graphs of the error between actual distance and estimated distance for all the six distance measurements. The sixth chapter presents a swarm formation demonstration with three Garcia robot. The seventh chapter presents the conclusion, application and future work.

CHAPTER 2

LOCALIZATION

2.1. Introduction

Sensor Networks, by definition [1], is an infrastructure comprised of sensing (measuring), computing and communication elements that give an administrator the ability to instrument, observe and react to events and phenomena in a specified environment. The sensors monitor some environment parameter and relay the measured data either directly to the base station or through a repeater node. In Wireless Sensor Networks (WSN) the communication between the sensors and the base station occurs through wireless channels as the sensors need to be deployed in environments where going "wireless" is preferable over "wired" communication. The base station can then make a decision based on the sensor readings and decide on a set of action. However if the location of the sensor is not known to the base station, then the readings from the sensor are hardly of any use. Localization is a mechanism for autonomously discovering and establishing spatial relationships among objects- sensors in case of WSN[2]. There are various techniques which use different algorithms and physical phenomenon to perform localization. These techniques and the potential field approach, in particular will be discussed in this chapter.

2.2. Localization Methods

There are two main approaches for sensor node localization [3,4,5,6]. One is based on measuring the distances between the sensor nodes which requires special hardware equipment to measure distances and the other is a simpler one, in which such measurement is not required.

2.2.1. Range-based or Geometry-based

This method involves measurement of either the distance or angle between different nodes. In this case the nodes have special hardware using which the distance or angle between different nodes can be measured. The method provides the actual co-ordinates of the position on the nodes. There are two different types :

2.2.1.1. Trilateration

Trilateration is a method in which the distances between the nodes are used to find their co-ordinates. This method requires distance measurements between a minimum of three nodes. This is based on the concept that a node location is uniquely specified if the location of a minimum of three nodes is available to the node.

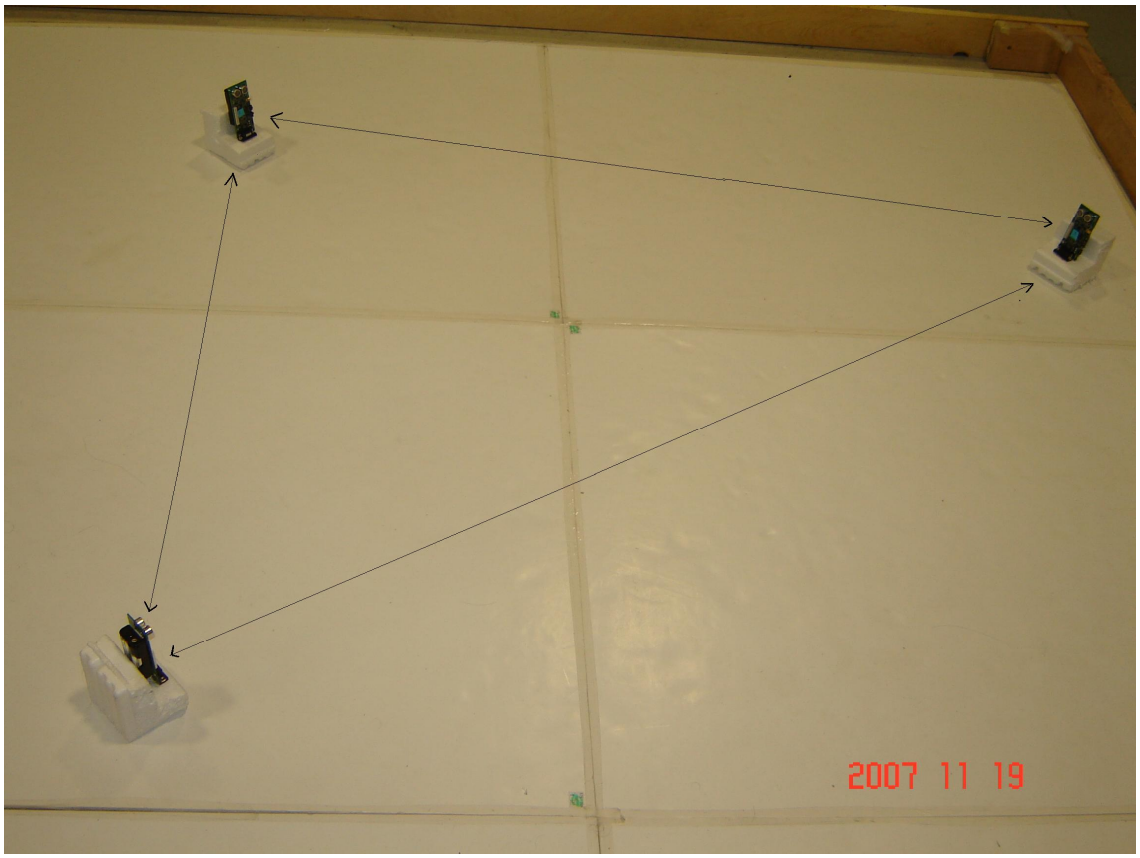


Fig 2.1 Trilateration

2.2.1.2. Triangulation

Triangulation is a method in which the angles between nodes are used to find the co-ordinates.

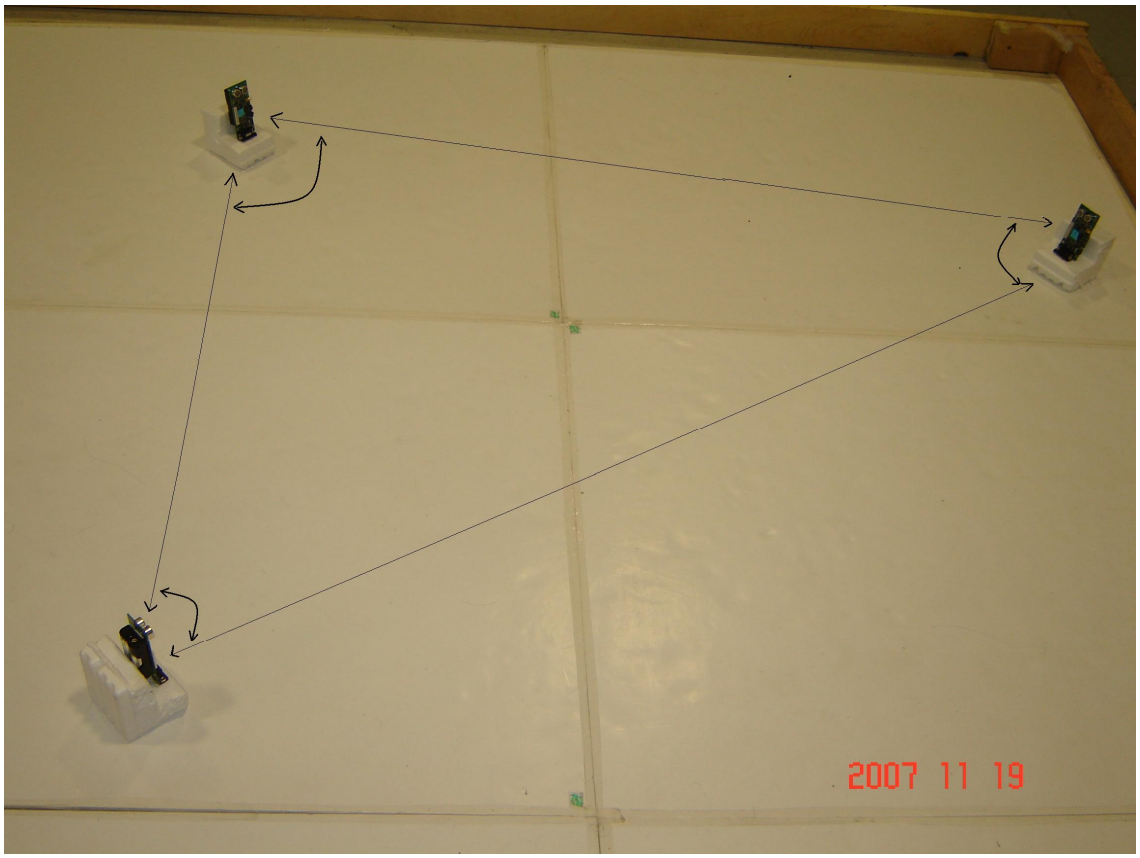


Fig 2.2 Triangulation.

2.2.2. Range-free or Proximity based

The proximity based method is simpler as compared to the geometry based method. In this case, information regarding the proximity of the node to a location with known co-ordinates rather the actual co-ordinates of the node is important. Hence no special hardware is needed to measure either distance or angle. This method provides an approximate estimate of the position.

Localization can also be qualified as being *absolute*, wherein the co-ordinates are defined with respect to a absolute frame of reference, or *relative* wherein a local frame of reference is used to assign the co-ordinates to the motes. An example of absolute frame of reference would be the Global Positioning System.

2.3. Ranging Methods

Ranging is the process of finding the distance or angle between the motes in a WSN. Range measurement is performed on the basis of time intervals between light, radio or acoustic signals between the motes.

2.3.1. Range Measurement Using Light

Laser based range measurement is amongst the popular and accurate range measurement. The apparatus is not cost-effective and is also bulky. Laser , by essence is highly directional, and hence while measuring distances the laser source and the target must either have line of sight path or reflecting mirrors can be used. In either case lasers impose a limitation on the alignment of the motes while taking range measurements.

Cameras could also be used to measure distances but the sensors need high processing power to process the data from cameras. This increases the cost and power consumption.

2.3.2. Range Measurement Using Radio

In this case the same apparatus used for radio communication can be used to measure distances between the motes. Distance measurement can be performed using either of the two following approaches:

2.3.2.1. Received Signal Strength Indicator

The power of a radio signal is inversely related to the distance traveled. Using this underlying principle the distance traveled by the radio signal from the transmitter to the receiver can be approximated. This approach, however, is fraught with many unpredictable parameters. The wireless signals undergo *fading* which can affect the received signal power. The effect of the fading however cannot be accurately predicted and it depends on the frequency, as well as the environmental conditions. The transmit power also affects the received power.

2.3.2.2. RF Time of Flight

The problems faced in RSSI measurement can be alleviated by measuring the duration between transmission and reception of the radio frequency signal. Since the effect of the environment on the velocity of radio frequency signals is negligible this method is quite accurate. The equipment needs to be high precision to be able to detect the small time intervals between transmission and reception. The transmitter and receiver must be synchronized to avoid any errors in measurement.

2.3.3. Range Measurement Using Acoustics

Ultrasound signals can be used to measure distance between the motes. An accurate measurement of the distance between the transmitting and receiving motes is achieved when a direct line of sight path is present. This method is quite accurate and cheap. As compared to RF time of flight, timing and synchronization is easily achieved in this case.

2.4. Cricket Hardware for Localization and TinyOS

In this thesis the trilateration method based has been used for geometric localization of the sensor motes. The "Cricket" sensor motes manufactured by CrossBow Technology have been used in the thesis to measure the distances using Ultrasound signals. Once the distances between the sensors are available the potential field approach can be applied to localize the network. The potential field approach and localization using the distance readings are discussed in the nest chapter.

2.4.1 Cricket Hardware

The "Cricket" hardware manufactured [7, 8, 9, 10, 11] by CrossBow Technology is basically a sensor with an embedded controller with the capability of wireless communication. The Cricket module has following components which have been illustrated in the figure below:

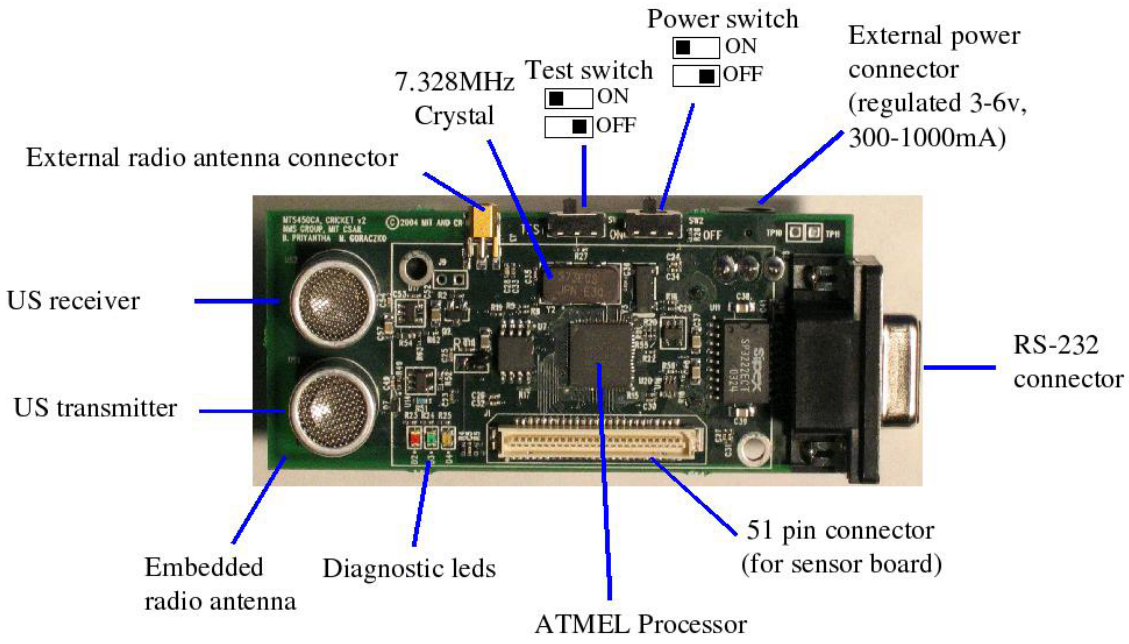


Fig 2.3 Cricket v2 hardware components and layout. [11]

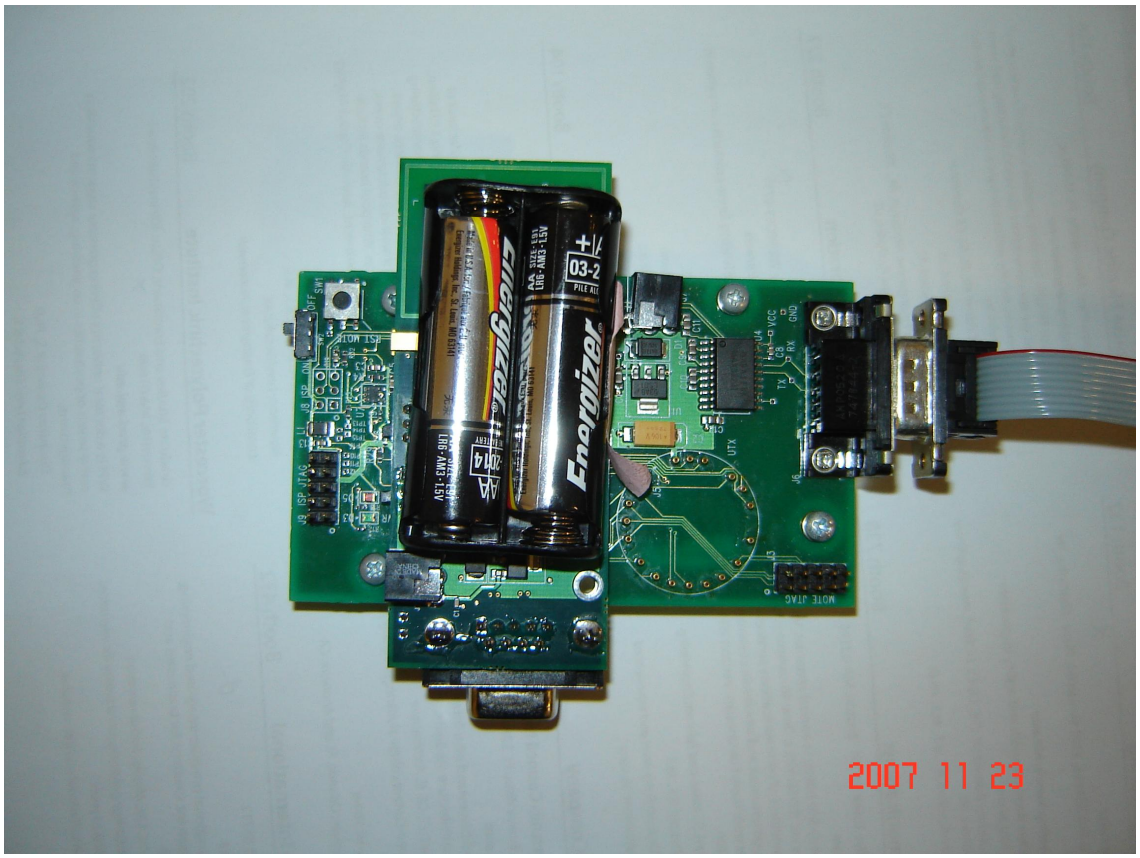


Fig 2.4 Cricket with MIB510 programming board.

2.4.1.1. Radio Communication System

The cricket wireless communication operates around 433 MHz. Each cricket is equipped with a multi-channel transceiver.

2.4.1.2. Ultrasound Range Measurement

Each cricket has an ultrasound transmitter and a receiver. Whenever the cricket broadcasts any radio signal, the cricket has the facility send an ultrasound signal as well. Since the speed of sound (330 m/sec) is lesser than that of light (2.97×10^8 m/sec) the instances of arrival of both signals at the receiver have a time lag. The range

measurement is performed by measuring the delay between the instance of arrival of the radio signal and the ultrasound signal. A simple calculation can then give the distance between the two crickets to an accuracy of a few centimeters. The measurement is performed on the basis that a direct line of sight path is present between the two crickets.

2.4.1.3. Embedded Microcontroller

The processor is an Atmel 128L processor which stores the code in 128 K bytes internal Compact Flash memory. There is also 4K bytes SRAM and 4K bytes EEPROM.

2.4.1.4. ADC channels.

There are 8 10-bit ADC channels present. An external sensor board can be connected to the cricket mote. This board, MTS310CA, has a light, sound, temperature, magnetic as well as acceleration sensors. The ADC channels are used to collect the signals from these sensors.

2.4.1.5. Serial UART

A serial UART port is provided for direct communication with the cricket. The cricket programming is not done through the serial port directly. The port can be used to get data from the mote if wireless communication cannot be performed.

2.4.1.6. Power

The Cricket mote is powered by two AA batteries. The optimal operational voltage is above 1.4 V.

2.4.2. TinyOS

TinyOS is a small, open-source, energy-efficient operating system developed by UC Berkeley[12]. TinyOS is essentially a scaled down version of UNIX. Cygwin, a software which emulates UNIX file system in Windows, is also available along with the TinyOS source code at [13]. A tutorial is also available at [14].

TinyOS is an event based operating system in the sense that, the occurrence of an event is detected, and then the set of assigned ”tasks” are then executed. The event occurrence is generally triggered by hardware but can also be “signaled” in software. TinyOS is ideal for WSN as it requires very little resources and the hardware springs into action only after “event” occurs. The rest of the time the system is in power down mode, which enhances the battery life. The coding of the programs to be executed can be done either in JAVA or using nesC. Nesc or Network Embedded Software C is a derivative of the C programming language. A description as well as tutorial can be found at [15].

2.4.3. nesC

Network **E**mbedded **S**ystem **C** is essentially an extension to the C software language designed with the intent of supporting the structure and execution model of TinyOS[17]. As mentioned above TinyOS is an event-driven operating system which works ideally for sensor networks. It supports most of the functionalities and mathematical functions of C. nesC was developed to create an event-driven execution,

concurrency model and component-oriented application design. At compilation time of nesC code, a detailed analysis of the code is performed and data races can be detected as well the total data RAM and program ROM consumed is specified.

nesC is similar to a hardware description in that the concept of components and wiring is present. [17]The *components* represent the hardware present on the sensors whereas *interfaces* describe the behavior of the component. An *interface* has *events* and *commands* which are used to implement the functionality. An *interface* can be *used* or *provided* : a an *interface* that is *provided* implements the functionality provided by the component to the user , an *used interface* on the other represents the functionality needed by the component to perform it's task. *Tasks* are defined as a set of actions which the hardware is intended to perform once an expected event has occurred. The flow of execution passes intermittently between *events* and *tasks* and in the period when neither is being executed the processor is in *idle* mode. The components and interfaces are bound to the target hardware at compile time and the tasks to be executed when defined events occur are burned into the device ROM.

CHAPTER 3

POTENTIAL FIELD APPROACH TO LOCALIZATION

3.1. Introduction

In this thesis the method of trilateration will be used to localize a network of three cricket motes. The motes transmit the distances between each other to the base. The base then performs the localization using potential field approach to get the co-ordinates of the three motes. If there is fixed frame of reference, then the localization performed is absolute. Thus, if the co-ordinates of one mote are known absolutely then the co-ordinates of the other two motes can also be calculated and the absolute localization of all nodes is possible. If however the localization is performed with no fixed frame of reference, then the localization is said to be relative localization.

3.2. Potential Field Localization

The potential field approach to localization has been discussed in detail in [19]. The theory for the localization has been developed in the paper as well. The main aspects of the theory have been reproduced here with brevity.

It is assumed that the distance measurements between the static nodes to be localized, is available. The method uses a dynamic model for position estimates of each node and is driven by a fictitious virtual force based on range errors. The virtual dynamics have states which are estimates of the relative positions and are calculated

recursively till a steady-state value that provides an optimal estimate of the relative positions of the nodes under consideration.

3.2.1. System Description

The position estimate for the i^{th} sensor node is given by ,

$$X_i = [x_i \quad y_i]^T \qquad .. (3.1)$$

where, x_i and y_i are the co-ordinates.

The position estimation dynamics are given as

$$\ddot{X}_i = \bar{f}_i \qquad ..(3.2)$$

where $\bar{f}_i = [f_i^x \quad f_i^y]$ is the virtual force in x and y direction.

The state variable description form for the position estimate of the i^{th} node is given by

$$\begin{bmatrix} \dot{X}_i \\ \ddot{X}_i \end{bmatrix} = \begin{bmatrix} O_2 & I_2 \\ O_2 & O_2 \end{bmatrix} \begin{bmatrix} X_i \\ \dot{X}_i \end{bmatrix} + \begin{bmatrix} O_2 \\ I_2 \end{bmatrix} \begin{bmatrix} f_i^x \\ f_i^y \end{bmatrix} \qquad ..(3.3)$$

where $O_2 = \begin{bmatrix} 0 & 0 \\ 0 & 0 \end{bmatrix}$ and $I_2 = \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix}$

3.2.2. Potential Field for Optimal Position Estimation

A potential field is introduced to determine the virtual force in(3.2) which provides an optimal estimate for the relative position of the nodes. The potential field is defined as

$$V_{ugs} = \sum_{i=1}^{N} \sum_{\substack{j=1 \\ j \neq i}}^{N} K_{ij} (r_{ij} - \bar{r}_{ij})^2 \qquad .. (3.4)$$

where $r_{ij} = \sqrt{[(x_i - x_j)^2 + (y_i - y_j)^2]}$ is the estimated range

$\bar{r}_{ij}$ is the actual measured range between i^{th} and j^{th} nodes.

The gradient of the potential with respect to the sensor node state is given by,

$$\frac{\partial V_{i_{ugs}}}{\partial X_i} = \sum_{\substack{j=1 \\ i \neq j}}^{N} K_{ij}(r_{ij} - \bar{r}_{ij}) \left[\frac{x_{i-x_j}}{\|X_i - X_j\|} \hat{x} \frac{y_i - y_j}{\|X_i - X_j\|} \hat{y} \right] \tag{3.5}$$

The virtual force acting for the i^{th} sensor node is given as,

$$\vec{f}_i = -\sum_{\substack{j=1 \\ i \neq j}}^{N} K_{ij}(r_{ij} - \bar{r}_{ij}) \frac{(X_i - X_{j)}}{\|X_i - X_j\|} - K_v \dot{X}_i \tag{.3.6}$$

The position estimates reach steady-state values that provide optimal estimates of the actual relative localization of the nodes so that V_{ugs} is minimized. The proof has been provided in [21].

The algorithm for relative localization has been reproduced here from [21].

1 Initialize N =3, the number of nodes in the network to start with the localization process.
2 Localize the network with N=3.
3 Increment N by 1 depending on the number of sensor nodes.
4 Initialize the next sensor node position.
5 Localize the network with new sensor node.
6 Repeat steps 3-5 till all nodes are localized.

Fig 3.1 Relative Localization Algorithm.

3.3. Distance Measurement Using Cricket

As mentioned in the earlier chapter, the cricket mote comes equipped with an Ultrasound ranging device. From the discussion of the potential field method above, if the distances between a minimum of three nodes were to be known, then with just the distance information, the nodes can be localized. The object of programming the crickets is for a set of three cricket to measure distances between each other, store them and then transmit the same to the base. The workstation attached to the base then localizes the nodes.

3.3.1. Cricket Modes

The cricket code developed at [19] and the tutorial [11] specify that the cricket can be programmed to work as either a *Beacon* or *Listener*. The mode can be specified in the *CricketConfig_s* structure while programming the cricket. In *beacon* mode, the cricket transmits a radio signal accompanied by an ultrasound signal repeatedly. A cricket, which has been configured in listener mode, detects the radio signal and starts an internal timer. The timer is stopped when the ultrasound signal is detected. The time interval between the detection of the two signals is directly proportional to the distance between the two crickets. A cricket can be programmed to work in a different mode by sending commands over the serial UART using a HyperTerminal. However, changing

the mode on the fly without the serial port didn't guarantee expected operation and the delay between the mode change could be large.

To find the distance between any two crickets measured by both crickets, a cricket needs to alternately listen for beacons from other crickets and then transmit its beacon to other crickets. This frequent changing of the modes could not be accomplished properly with the code. Hence, a version of cricket code developed at [20] was tested. This version supported alternate transmission and reception of beacons.

3.3.2. Cricket Programming

Each cricket is assigned a unique *id* while programming using the following command

make cricket install.x mib510,/dev/ttyS0

where "x" indicates the *id*. When a cricket receives a beacon, it identifies the id of the transmitting cricket and stores both the id and the measured distance. This information about the id and distances is transmitted as a radio message. When the base receives this message, it calculates the distance of the transmitting cricket. The radio message contains a list of id and distances of all the cricket from the cricket whose radio message has been received at the base.

3.3.2.1. Code Execution

The functioning of the cricket is based on two main *events*. A timer, *BeaconTimer*, is started once the cricket is turned on. After the specified time duration is over, the *BeaconTimer.fired* event occurs and then the radio message alongwith the

ultrasound signal is transmitted. A *RadioReceive.receive event* occurs once the cricket receives a radio message. A task to calculate the distance and store the id and distance is then put on the task handler. The *BeaconTimer* is then started again.

The base station code execution is a bit different. A base station is identified as the cricket having an id =0. In the case of the base station, no *BeaconTimer* is used. The code execution depends completely on the *RadioReceive.receive event* . Whenever a radio message is received the id of the transmitting cricket node as well as the data transmitted by the node is stored and then transmitted to the PC using the UART. The parsing of the message received by the PC is performed using LabVIEW.

3.3.3. Experiment Setup

The setup for localization using trilateration comprises of three cricket beacons and a base station cricket. The setup is shown in the following picture.

Fig 3.2 Setup for the Localization.

3.3.4. LabVIEW : Distances.vi

A LabVIEW vi was designed to extract the id and distance measurements from the message sent by the base station UART. The following image is a screenshot of the front panel of the designed vi.

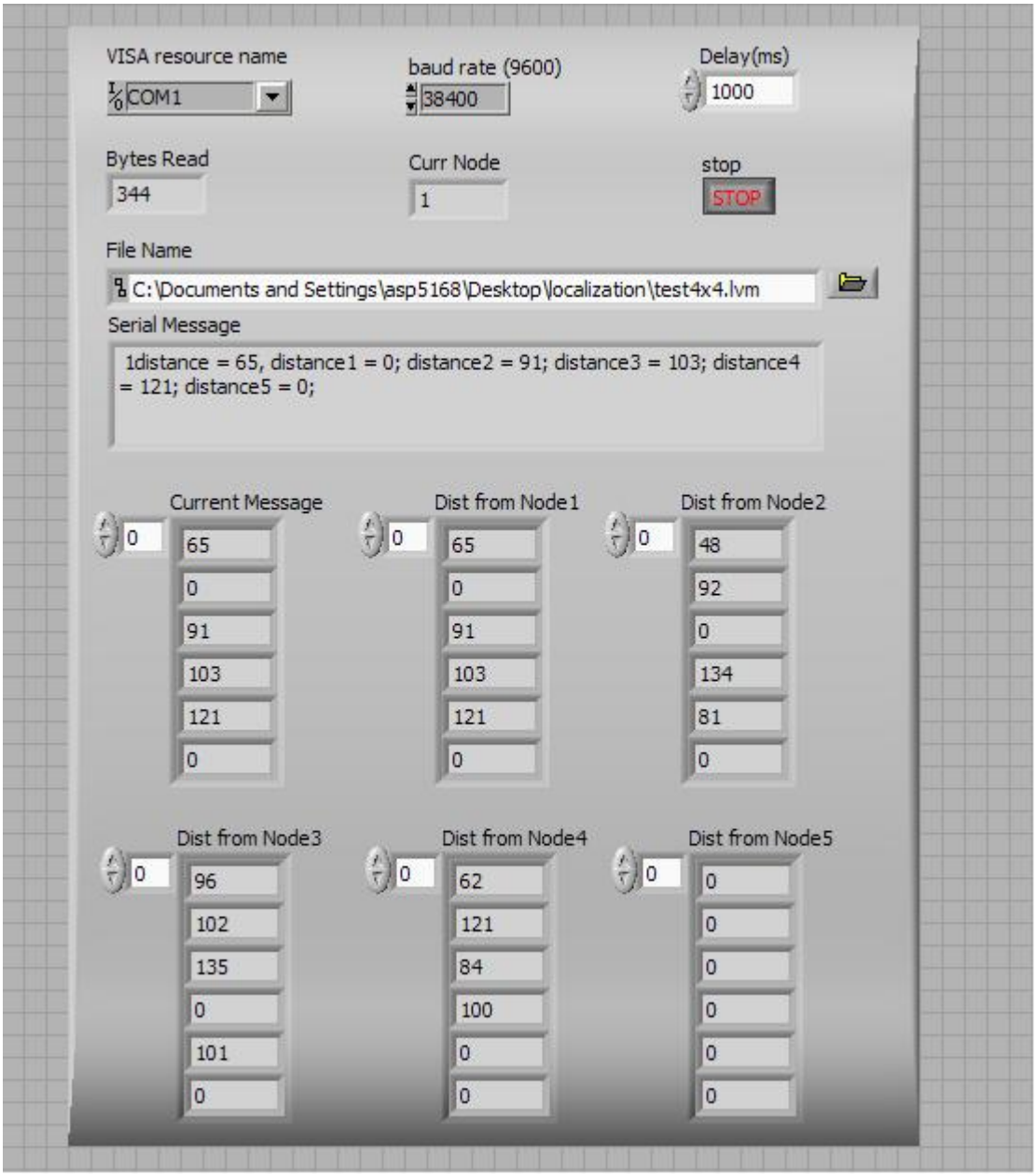


Fig 3.3 Front panel of LabVIEW application to store distances between nodes.

The contents of the message being received are shown in the center. The message is then deciphered and the distances extracted as shown in a screenshot of the block diagram shown below.

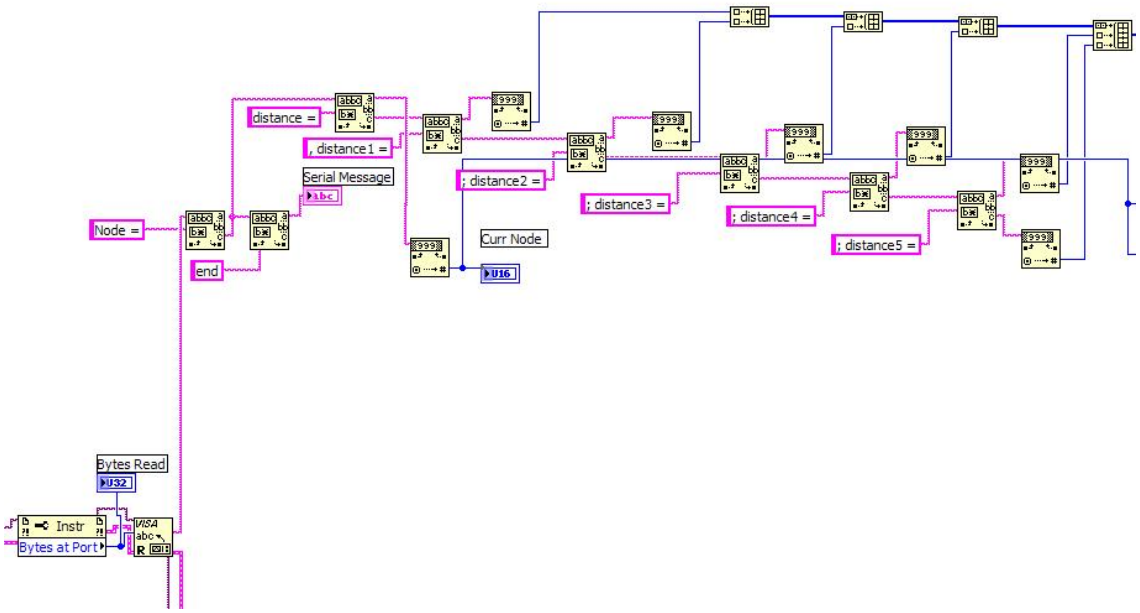


Fig 3.4 Block diagram of VI for extracting distance measurement from message received from cricket mote.

The extracted distances are then stored in an array based on the id of the nodes transmitting the message. The distance of the transmitting node from the base is also calculated at the base station.

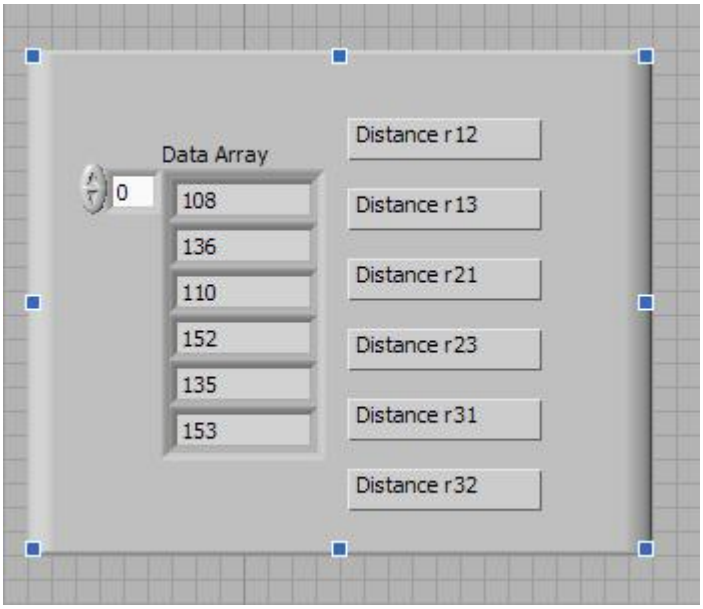


Fig 3.5 Illustration of the array of distance measurements between 3 Nodes

From the distance array for each node, the distances from the nodes under consideration for localization are extracted to form another array. This array is then sent to the LabVIEW vi which performs the localization using potential field approach.

3.3.5. LabVIEW : Localization-nonEKF.vi

This vi was developed by Prasanna Ballal at ARRI. The application performs localization of three nodes when the distances between the nodes are provided. In this case the distance measurements are already available from the Distances.vi. The localization vi was modified to read the distance measurements from the distances vi, rather than being manually entered.

3.3.5.1. Relative Localization

A screenshot of the vi to perform relative localization is provided below. The distance measurement are read from a file created by the distances vi to store the readings. An average for each distance reading is calculated from the multiple readings available and the relative localization is performed. The resulting graph is shown below.

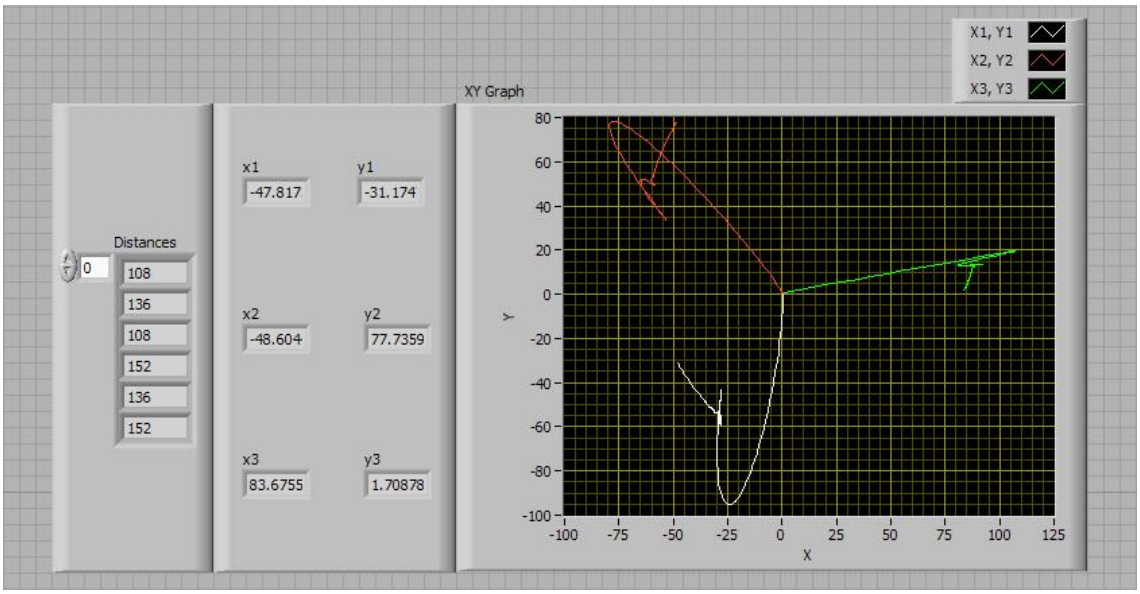


Fig 3.6 Relative Localization of 3 nodes using Potential field approach.

3.3.5.2. Absolute Localization

A small change to the vi used for relative localization results in absolute localization. In absolute localization, the absolute co-ordinates of one node are known. The localization algorithm then provides the absolute co-ordinates of the two other nodes. A screenshot of the front panel of the vi used for absolute localization is shown in the figure below.

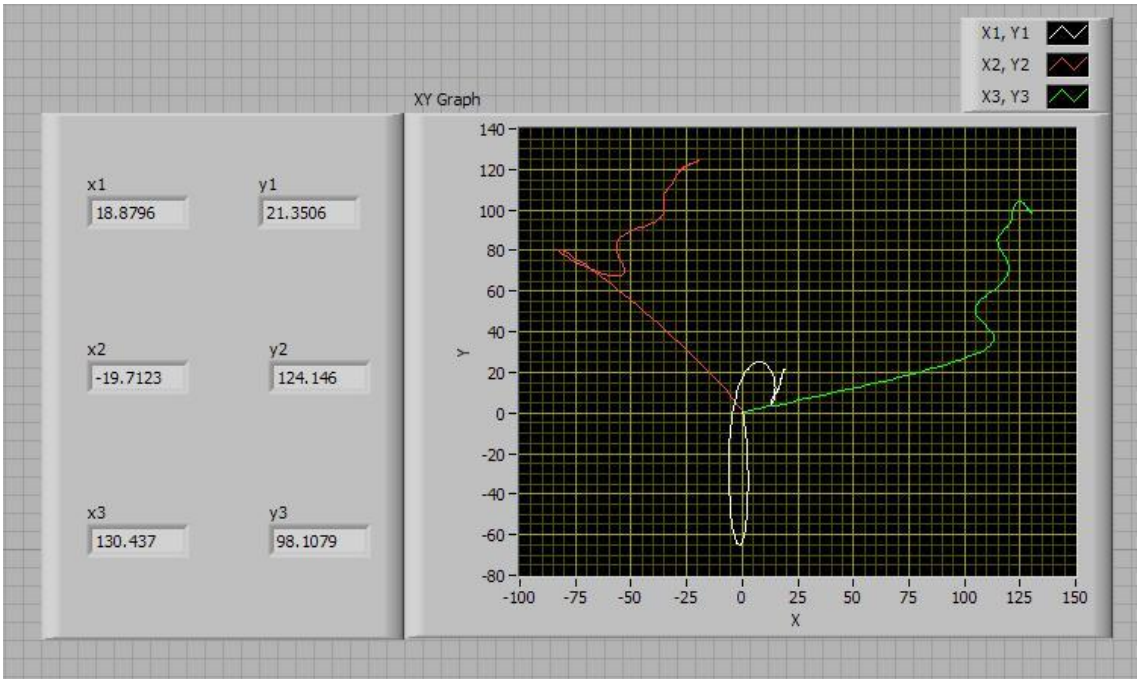


Fig 3.7 Absolute Localization of 3 nodes using Potential field approach

3.4. Limitation of the Potential Field Algorithm

The algorithm used to perform localization assumes that the distance of one cricket from a second cricket will be the same when measured by both. The table of readings from distances vi shown above proves that this assumption is not true, most of the times.

There is a difference of a few centimeters between the same distance measured by the two crickets motes. Also there is a difference between the readings at different time intervals performed between the same cricket motes without any change in the physical positions of the motes. One reason for this error is process noise.

3.5. Conclusion

The above algorithm does provide absolute as well as relative localization when the distance readings are available. However the basic premise, that the distance between two motes measured by both is equal, does not hold true. The distance readings also vary with time due to the measurement noise. Some sort of noise filtering must be implemented. Since the measurement function is a non-linear function ,an algorithm that performs recursive estimation using Extended Kalman Filtering has been proposed[21]. Localization performed using this algorithm has been implemented in the this thesis.

CHAPTER 4

KALMAN FILTER AND EXTENDED KALMAN FILTER

The Kalman filter is a set of mathematical equations which provide an effective recursive computational means to estimate the state of a process by minimizing the mean of the squared error[22]. The Kalman filter is based on the paper published in 1960 by R.E.Kalman, describing a recursive solution to linear filtering of discrete data. The Kalman filter provides a good estimate of the following state based on knowledge of the earlier state, unlike other filter implementations, such as the Weiner filter which need information from all the previous states.

4.1. Kalman Filter : Theory

The Kalman filter tries to estimate the state $x \epsilon R^n$ of a discrete-time controlled process governed by the linear stochastic difference equation

$$x_k = A\, x_{k-1} + B\, u_{k-1} + w_{k-1}\,, \qquad \text{.. (4.1)}$$

A measurement matrix $z \epsilon R^m$ is provided for the estimation.

$$z_k = Hx_k + v_k\,. \qquad \text{..(4.2)}$$

The variables w_k and v_k are random variables representing the process noise and measurement noise and are assumed to be mutually independent, white and with normal probability distributions.

$$p(w) \sim N(0,Q) \quad .. (4.3)$$

$$p(v) \sim N(0,R) \quad .. (4.4)$$

where Q is the process noise covariance and R is measurement noise covariance. Although they are assumed zero, in reality, they change with each time step or measurement.

The matrix *A* is a square matrix relating the current process state estimate with the state in the previous time instant. The matrix *B* is not necessarily a square matrix, and relates the current process state estimate with the system input at the previous time instance. The matrix *H*, which also is a rectangular matrix, relates the measurement to the state estimate.

The process noise w_k term can be removed by taking measurements of the system process and making a suitable estimate. The working if the filter is based on the idea that an a *posteriori* state estimate $\hat{x}_k$ is calculated as a linear combination of an a *priori* estimate $\hat{x}_k^-$ and a weighted difference between actual measurement z_k and a measurement prediction $H\hat{x}_k^-$ as shown in the following equation.

$$\hat{x}_k^- = \hat{x}_k^- + K(z_k - H\hat{x}_k^-) \quad .. (4.5)$$

The difference term $(z_k - H\hat{x}_k^-)$ is the measurement *innovative* or the *residual*. This term indicates the difference between actual measurement, z_k, and state estimate $H\hat{x}_k^-$. The matrix *K* is the *gain* or *blending factor* which is designed to minimize the a *posteriori* covariance.

$$K = \frac{P_k^- H^T}{HP_k^- H^T + R} \qquad ..(4.6)$$

4.1.1. Discrete Kalman Filter Algorithm

The equations of the filter can be divided into two steps: *Time update* equations and *Measurement update* equations. The time update equations involve the estimation of the next process state as well as the error covariance to get the a *priori* estimates. The measurement update equations are a feedback means to calculate the error between the estimated and actual measured state. This feedback is utilized in estimation of the next step for the time update equations. Thus the filter performs time update followed by measurement update recursively to minimize the error in the estimated and actual measured state.

The equations for the two steps can be tabulated as follows:

Table 4-1 Discrete Kalman filter time update (predictor) equations.

$$\hat{x}_k^- = A\hat{x}_k + Bu_{(k-1)}$$

$$P_k^- = AP_{(k-1)}A^T + Q$$

Table 4-2 Discrete Kalman filter time update (predictor) equations.

$$K_k = P_k^- H^T (HP_k^- H^T + R)^{-1}$$

$$\hat{x}_k = \hat{x}_k^- + K(z_k - H\hat{x}_k^-)$$

$$P_k = (I - K_k H)P_k^-$$

An analogy can be drawn to the *corrector-predictor* algorithm with the time update equations being predictor and the measurement update being corrector. The

algorithm can then be represented to recursively call two steps in a chart as shown below.

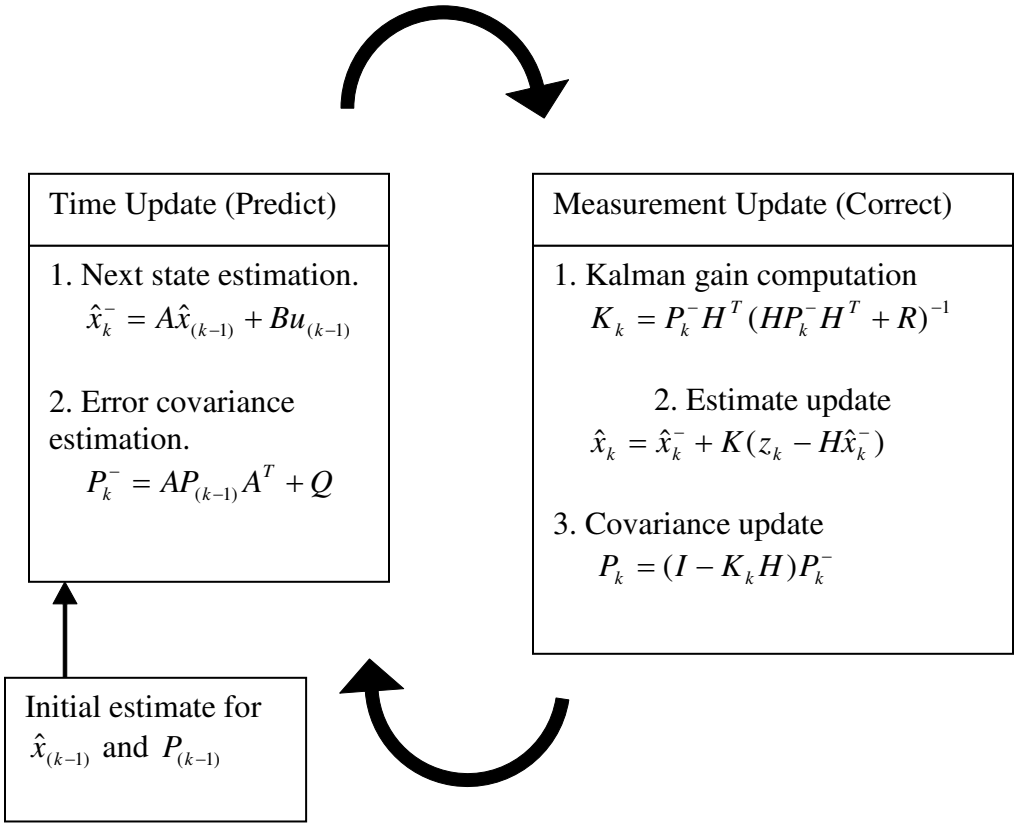


Figure 4.1 Operation of the Kalman Filter.

4.2. Extended Kalman Filter Theory

The discrete-time Kalman filter is defined for a process with a linear stochastic difference equation. The extended Kalman filter extends the Kalman filter equations to processes which have a non-linear relationship for measurement and and/or estimation.

A Kalman filter that linearizes about the current mean and covariance is referred to as *extended Kalman filter* or *EKF* [22].

In a non-linear estimation process, the current state estimate has a non-linear relationship with the previous state, input at previous time step. The process noise, generally, can be exactly measured and hence is neglected. The measured state also has a non-linear relationship with the previous state and measurement noise. As in the Kalman filter above, the relationship between current and previous state and current measurement and previous state can be given as

$$x_k = f(x_{(k-1)}, u_{(k-1)}, w_{(k-1)}) \quad .. (4.7)$$

$$z_k = h(x_k, v_k) \quad .. (4.8)$$

f and h above are non-linear functions relating the current process state with previous state, input and process noise , and the current measured state with current estimated state. The above nonlinear relationships can be expressed in linear forms as

$$x_k = \tilde{x}_k + A(x_{(k-1)} - \hat{x}_{(k-1)}) + Ww_{(k-1)} \quad .. (4.9)$$

$$z_k = \tilde{z}_k + H(x_k - \tilde{x}_k) + Vv_k \quad .. (4.10)$$

where

x_k and z_k are the actual state and measurement vectors,

$\tilde{x}_k$ and $\tilde{z}_k$ are the approximate state and measurement vectors.

$\tilde{x}_k$ is an a *posteriori* estimate of the state at step k,

w_k and v_k represent the process noise and measurement noise.

A is the Jacobian matrix of partial derivatives of f with respect to x

$$A_{[i,j]} = \frac{\partial f_{[i]}}{\partial x_{[j]}}(\hat{x}_{(k-1)}, u_{(k-1)}, 0) \quad .. (4.11)$$

W is the Jacobian matrix of partial derivatives of f with respect to w

$$W_{[i,j]} = \frac{\partial f_{[i]}}{\partial w_{[j]}}(\hat{x}_{(k-1)}, u_{(k-1)}, 0) \quad .. (4.12)$$

H is the Jacobian matrix of partial derivatives of h with respect to x

$$H_{[i,j]} = \frac{\partial h_{[i]}}{\partial x_{[j]}}(\tilde{x}_k, 0) \quad .. (4.13)$$

V is the Jacobian matrix of partial derivatives of h with respect to v

$$V_{[i,j]} = \frac{\partial h_{[i]}}{\partial v_{[j]}}(\tilde{x}_k, 0) \quad .. (4.14)$$

The state estimation equation can be simplified as

$$\tilde{x}_k = \tilde{x}_k + K_k \tilde{e}_{z_k} \quad .. (4.15)$$

where

$$\tilde{e}_{z_k} = z_k - \tilde{z}_k \quad .. (4.16)$$

is the measurement residual.

The table of EKF equations for the time update (predict) and measurement update (correct) state are shown below.

Table 4-3 Extended Kalman filter time update (predictor) equations.

$$\hat{x}_k^- = f(\hat{x}_{(k-1)}, u_{(k-1)}, 0)$$

$$P_k^- = A_k P_{(k-1)} A_k^T + W_k Q_{(k-1)} W_k^T$$

A_k and W_k are the process Jacobians at step k and Q_k is the process noise covariance at step k.

Table 4-4 Extended Kalman filter measurement update (corrector) equations.

$$K_k = P_k^- H_k^T (H_k P_k^- H_k^T + V_k R_k V_k^T)^{-1}$$

$$\hat{x}_k = \hat{x}_k^- + K_k (z_k - h(\hat{x}_k^-, 0))$$

$$P_k = (I - K_k H_k) P_k^-$$

H_k and V_k are the measurement Jacobians at step k and R_k is the measurement noise covariance at step k.

The algorithm can be represented in a figure below.

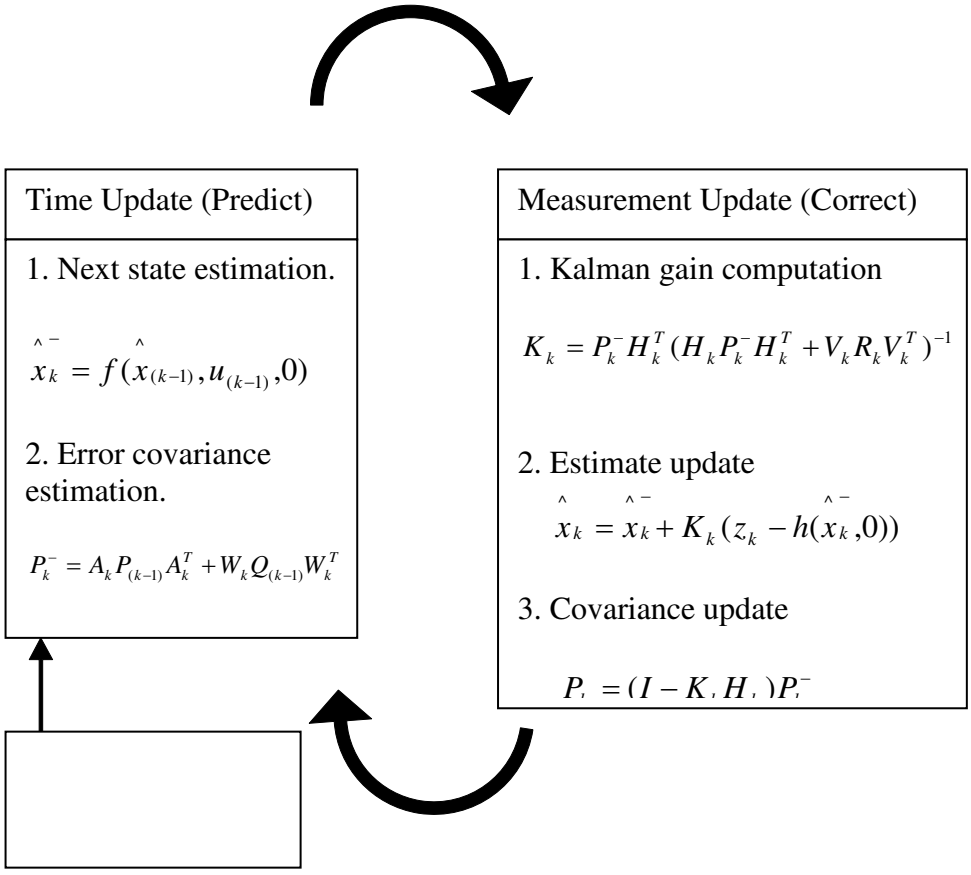


Figure 4.2 Operation of Extended Kalman Filter.

4.3. Localization using Extended Kalman Filter

The localization model presented in the earlier chapter does not take the implementation noise into consideration. The noise causes erroneous measurement of distance between crickets, with the result that there is a slight difference in the measurement of the same distance. A Kalman filter model has been proposed [21] to minimize the error. The block diagram of the localization scheme bundled with the Kalman filter for error minimization is shown below.

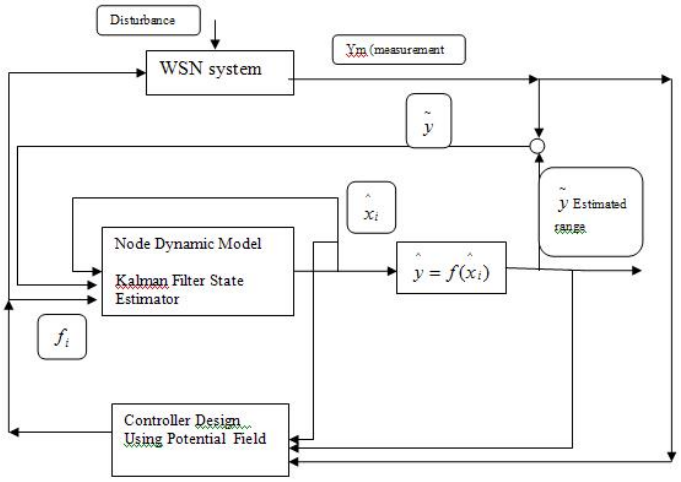


Figure 4.3 Block diagram of Localization scheme with Kalman filter.

The system model and measurement model for the Kalman filter are given as below,

4.3.1. System Model

Position estimate for the i[th] sensor node is given by,

$$X_i = \begin{bmatrix} x_i \\ y_i \end{bmatrix} \quad \text{.. (4.17)}$$

where x_i and y_i are the x co-ordinate and y co-ordinate of node i .

The position dynamics are given as

$$\ddot{X}_i = f_i = \begin{bmatrix} f_i^x \\ f_i^y \end{bmatrix} \quad \text{.. (4.18)}$$

The state space model is then given as

$$\begin{bmatrix} \dot{X}_i \\ \ddot{X}_i \end{bmatrix} = \begin{bmatrix} \dot{x}_i \\ \dot{y}_i \\ \ddot{x}_i \\ \ddot{y}_i \end{bmatrix} = \underbrace{\begin{bmatrix} 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \end{bmatrix}}_{\textit{Matrix A}} \begin{bmatrix} x_i \\ y_i \\ \dot{x}_i \\ \dot{y}_i \end{bmatrix} + \underbrace{\begin{bmatrix} 0 & 0 \\ 0 & 0 \\ 1 & 0 \\ 0 & 1 \end{bmatrix}}_{B} \begin{bmatrix} f_i^x \\ f_i^y \end{bmatrix} \quad \text{.. (4.19)}$$

4.3.2. Measurement Model

Let Z be the state of range measurements

$$Z = r_{ij} = \sqrt{(x_i - x_j)^2 + (y_i - y_j)^2} \quad \text{. . (4.20)}$$

$$Z = h(x_i, x_j) \quad \text{.. (4.21)}$$

The extended Kalman filter is used as the output is a nonlinear function.

4.3.3. Formulation of EKF

The equation for the filter model are

$$\dot{X}_i = Ax_i + Bu + Gw(t) \quad ; \qquad w(t) \sim N(0, Q) \qquad .. (4.22)$$

$$Z = h(x_i) + v(t) \quad ; \qquad v(t) \sim N(0, R) \qquad .. (4.23)$$

where

$$A = \begin{bmatrix} 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \end{bmatrix} \quad ; \quad B = \begin{bmatrix} 0 & 0 \\ 0 & 0 \\ 1 & 0 \\ 0 & 1 \end{bmatrix} \quad ; \quad u = \begin{bmatrix} f_i^x \\ f_i^y \end{bmatrix} \qquad .. (4.24)$$

The gain is given as

$$K_k = P_k^- H_k^T (H_k P_k^- H_k^T + P_k)^{(-1)} \qquad .. (4.25)$$

where

$$H_k = \left(\frac{\partial h(x)}{\partial X}\right)_{\hat{x}_k^-} \qquad .. (4.26)$$

The state update equation is

$$\hat{X}_k^+ = \hat{X}_k^- + K_k (Z - \hat{Z}) \qquad .. (4.27)$$

The co-variance update equation is

$$P_k^+ = (I - K_k H_k) P_k^- \qquad .. (4.28)$$

The state propagation equation is

$$\hat{X}_i = A\hat{X}_i + Bu \qquad .. (4.29)$$

The covariance propagation equation is

$$\dot{P} = AP + PA^T + GQG^T \qquad .. (4.30)$$

The matrix H is given as

$$H = \frac{\partial h(x)}{\partial X_i} = \frac{\partial (r_{ij})}{\partial X_i} = \frac{\partial}{\partial X_i}(\sqrt{(x_i - x_j)^2 + (y_i - y_j)^2}) \quad \text{.. (4.31)}$$

It is assumed that the process noise only affects the velocity.

CHAPTER 5

SIMULATION AND RESULTS

5.1. Introduction

Based on the EKF formulation in [21], as mentioned in the previous chapter, the localization using EKF was performed. The distances between the cricket motes are made available as discussed in chapter 3. The EKF implementation was performed was done using LabVIEW code. Relative and absolute localization has been performed as will be mentioned in this chapter. The results of the simulation have been presented.

5.2. Relative Localization with EKF Implementation in LabVIEW

The front panel of the LabVIEW implementation is shown below.

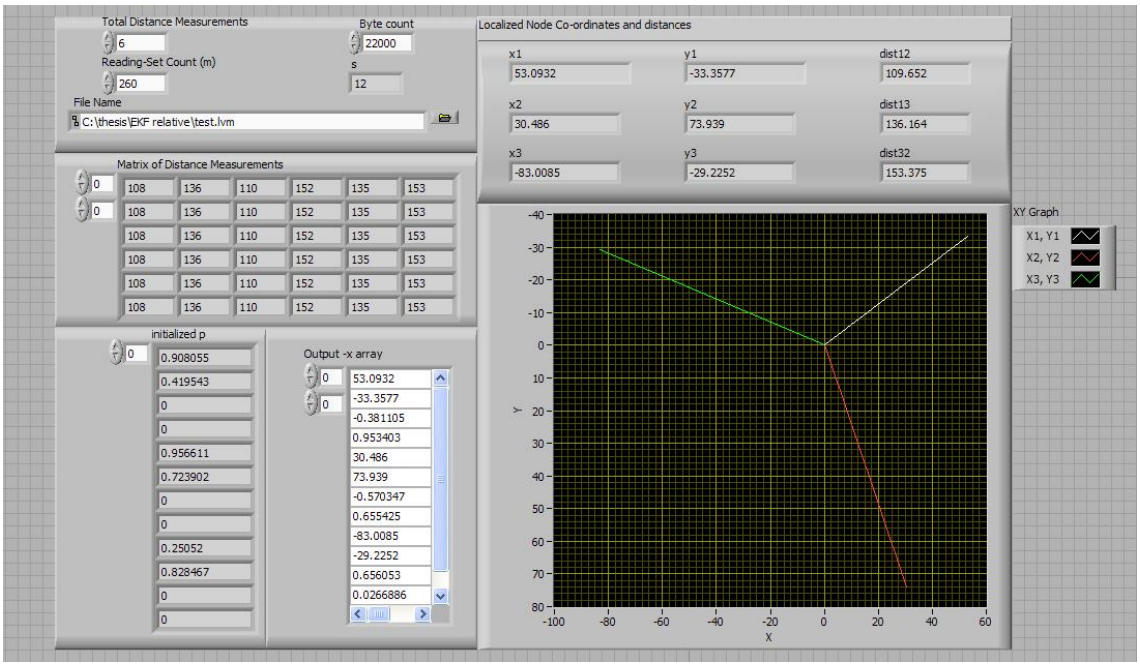


Fig 5.1 Front panel of the Relative Localization panel(EKF).

The above diagram lists the main input and the result of the EKF localization algorithm. The top left block contains the parameters to read the distance measurement readings obtained from the LabVIEW application mentioned in chapter 3. The distances are read by another *vi* as has been mentioned in chapter 3. The parameters specified are number of readings in each set (*Total Distance Measurements*), number of sets of distance readings (*Reading-Set count*) total bytes being read (*Byte count*). *s* is a parameter used for matrix generation.

The middle block on the left side displays the re-arranged distance readings in sets of readings with distance measured by nodes in ascending order in each row. Thus each row specified the distances of other nodes as measured by node 1, then node 2 and finally node 3.

The last block on the left specifies the input position and velocity parameters specified for the EKF. The final position and velocity vector is specified on right.

The block on the right top specifies the localized nodes. The (x, y) position co-ordinates of the nodes 1, 2 and 3 are specified as also the distances between the nodes after the localization using EKF.

The actual positions of the nodes localized above are shown in the figure below.

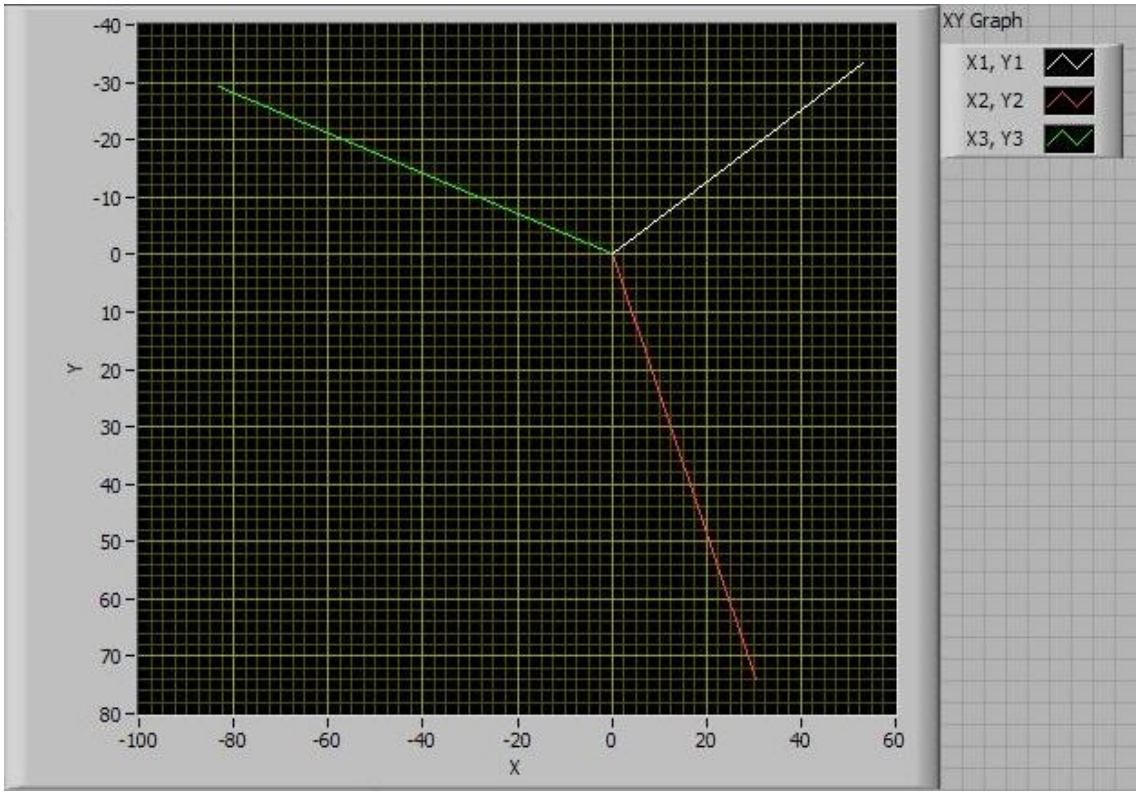


Fig 5.2 Resultant locations of the nodes.

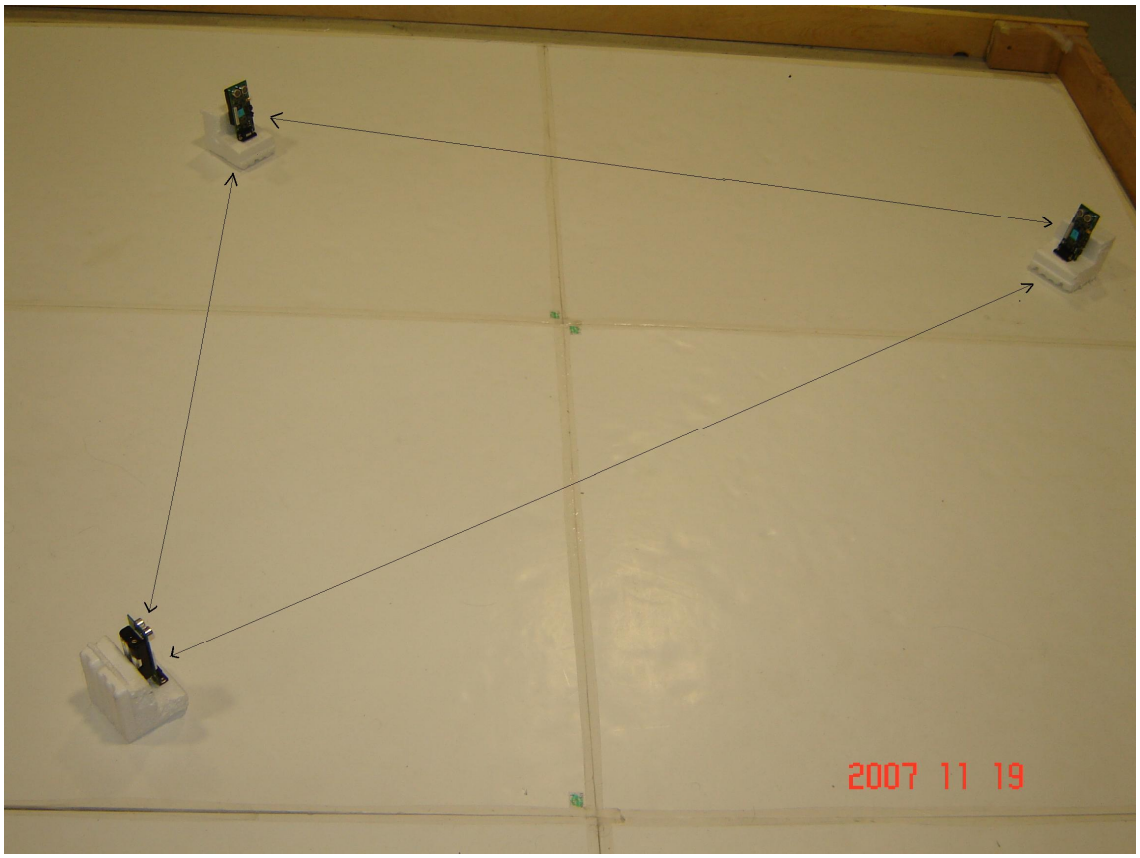


Fig 5.3 Actual locations of the nodes.

The Matlab code for the above algorithm has been written by Pritpal Dang. The above algorithm has been applied to a couple more mote arrangements as shown below. The actual node location and the result of the above algorithm are shown below.

Fig 5.4 Actual location of the nodes (scenario 2).

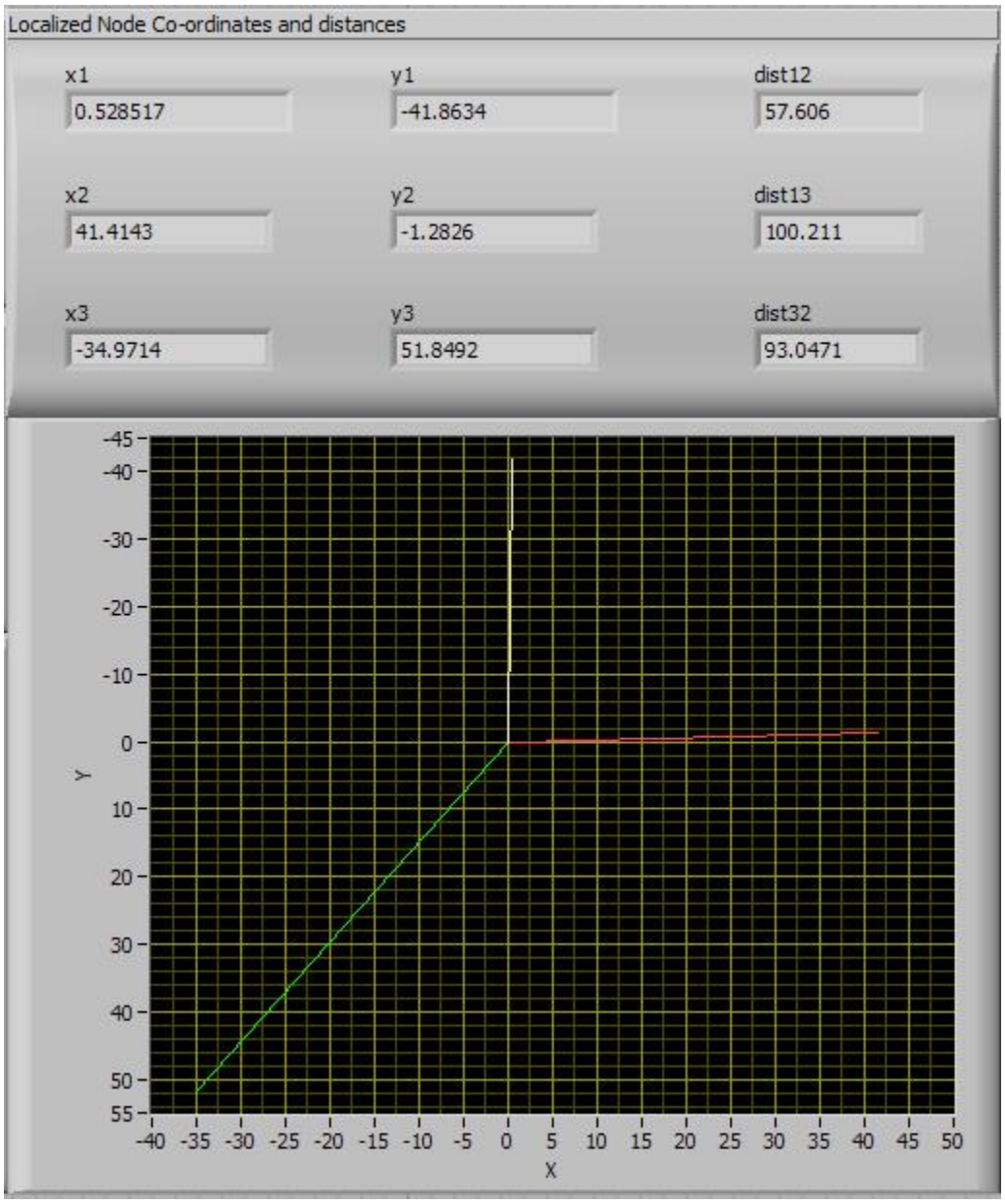


Fig 5.5 Resultant location of the nodes (scenario 2).

5.3. Error in Implementation

A comparison with the actual distance measured by the cricket motes and that after the localization shows a small error. The table of actual distances between the motes and the graph after simulation have been reproduced from fig 5.1 above.

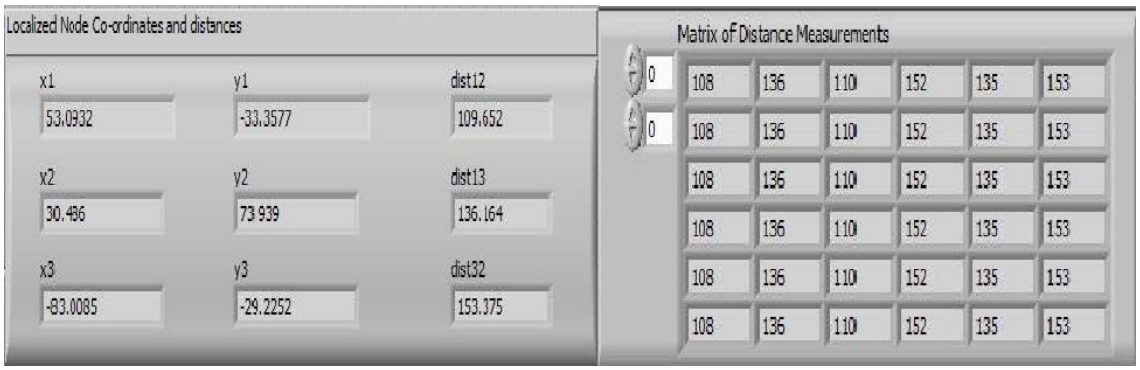


Fig 5.6 Illustration of error between actual distances and implementation.

There is an error in each distance measurement. A plot of the error between the actual distance and the distance estimation for each distance is shown below.

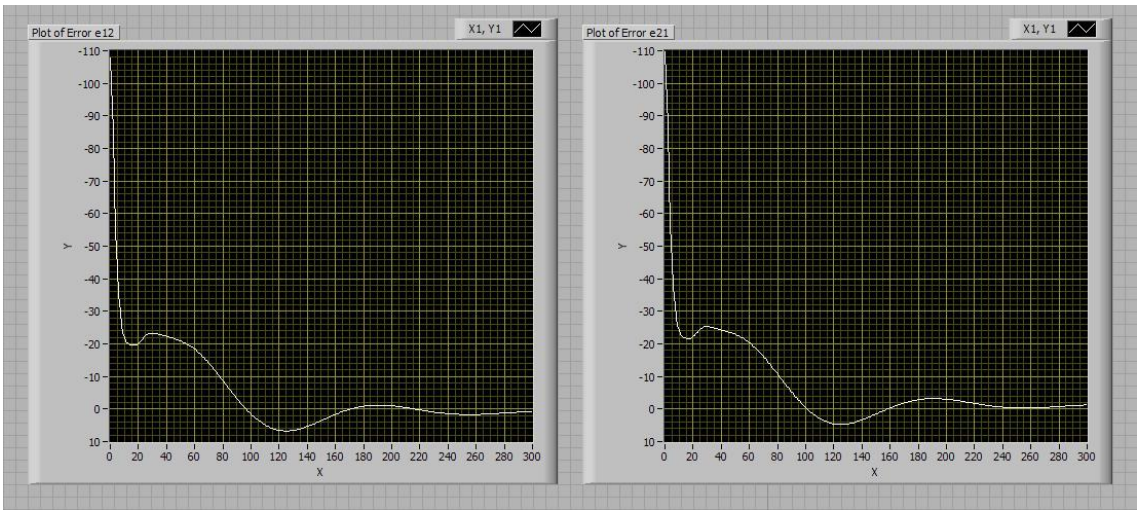


Fig 5.7 Illustration of the error between the actual distance and estimated distance between nodes 1 and 2.

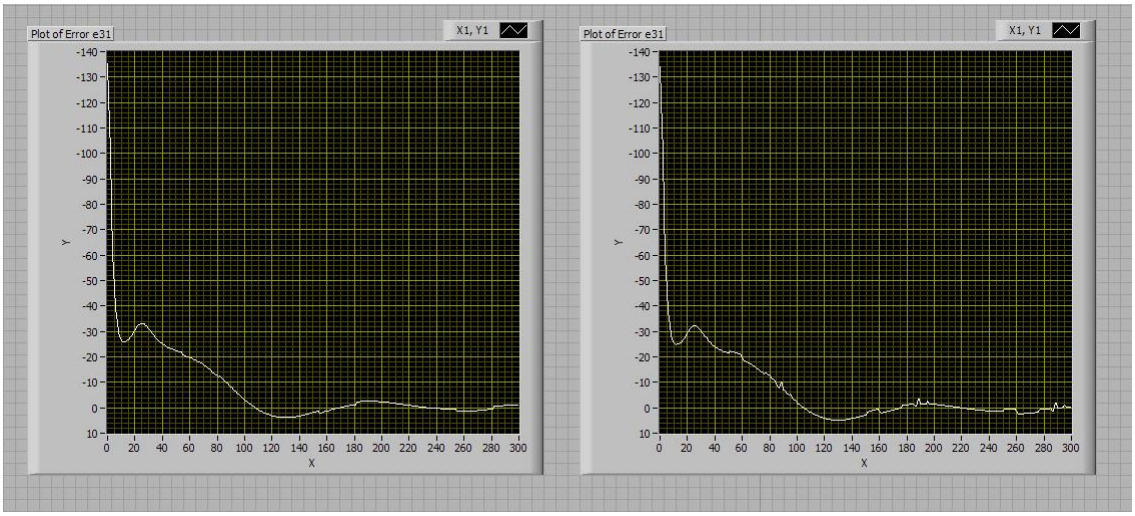


Fig 5.8 Illustration of the error between the actual distance and estimated distance between nodes 1 and 3.

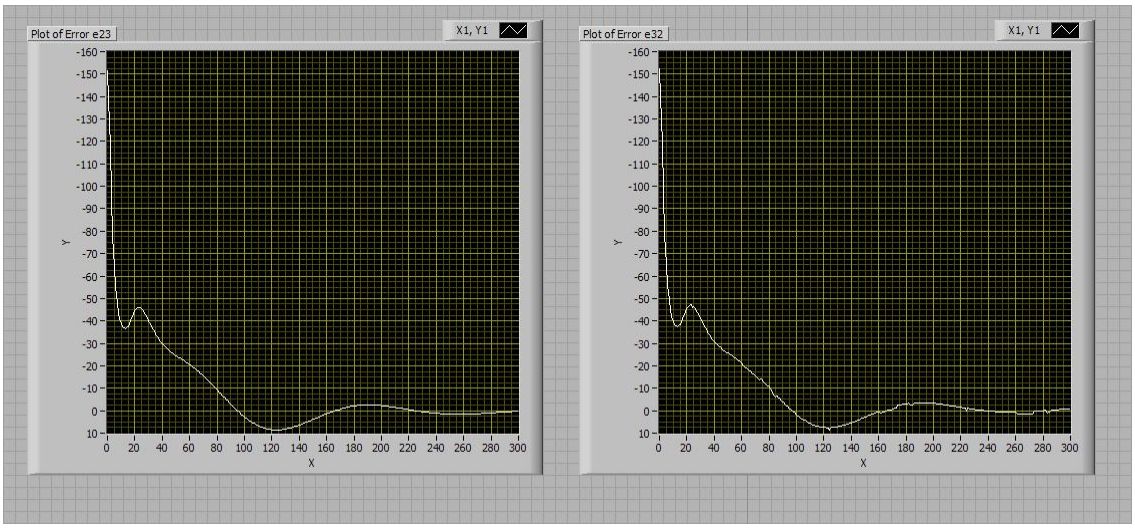


Fig 5.9 Illustration of the error between the actual distance and estimated distance between nodes 3 and 2.

5.4. Absolute Localization with EKF Implementation in LabVIEW

In absolute localization, the coordinates, with a standard frame of reference, of one or more nodes are known beforehand. Absolute localization is achieved in almost the same fashion as relative localization, with the difference that the force equation affecting the node with known coordinates is modified. The front panel for performing absolute localization is shown below.

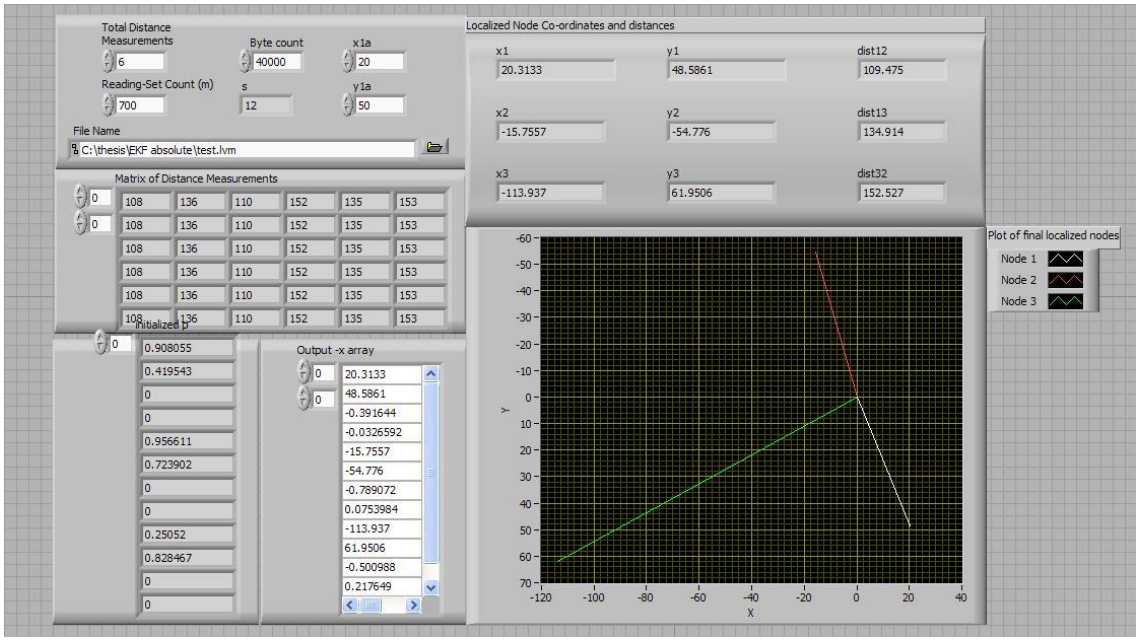


Fig 5.10 Front panel for Absolute Localization. The parameters "x1a" and "y1a" represent the available coordinates of node1.

The final co-ordinates of node 1 after localization are very close to the predefined ones. The number of times the simulation is run is also longer as compared to the relative localization. The figure below illustrates the distances between the nodes after the localization algorithm is applied (on left) and the readings obtained from measurements.

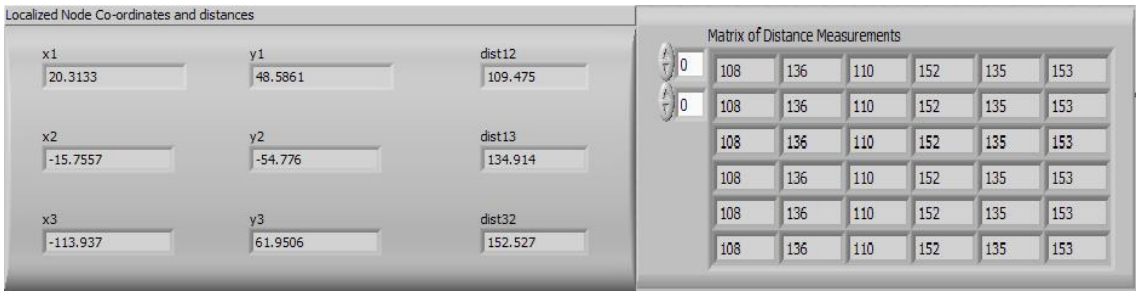


Fig 5.11 Illustration of error between actual distances and implementation.

5.5. Error in Implementation for Absolute Localization

As in the relative localization above, a plot of the error for the distances between the nodes is shown below.

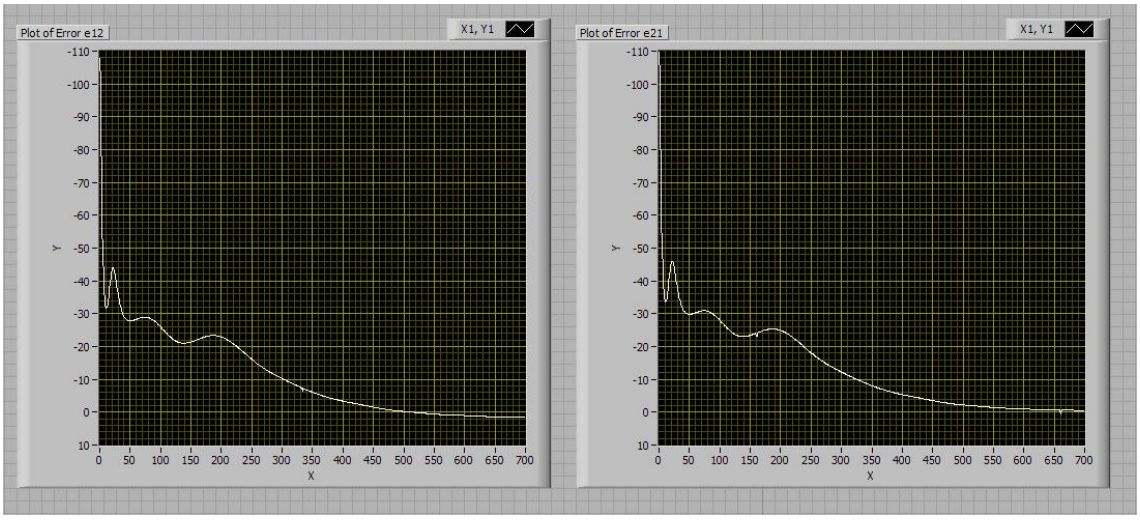


Fig 5.12 Illustration of the error between the actual distance and estimated distance between nodes 1 and 2 for absolute localization.

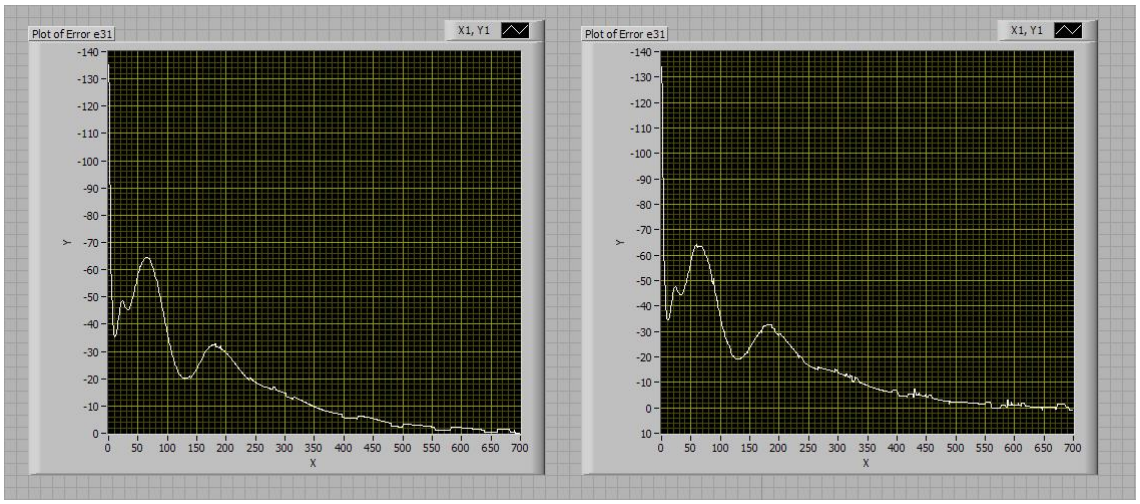


Fig 5.13 Illustration of the error between the actual distance and estimated distance between nodes 1 and 3 for absolute localization.

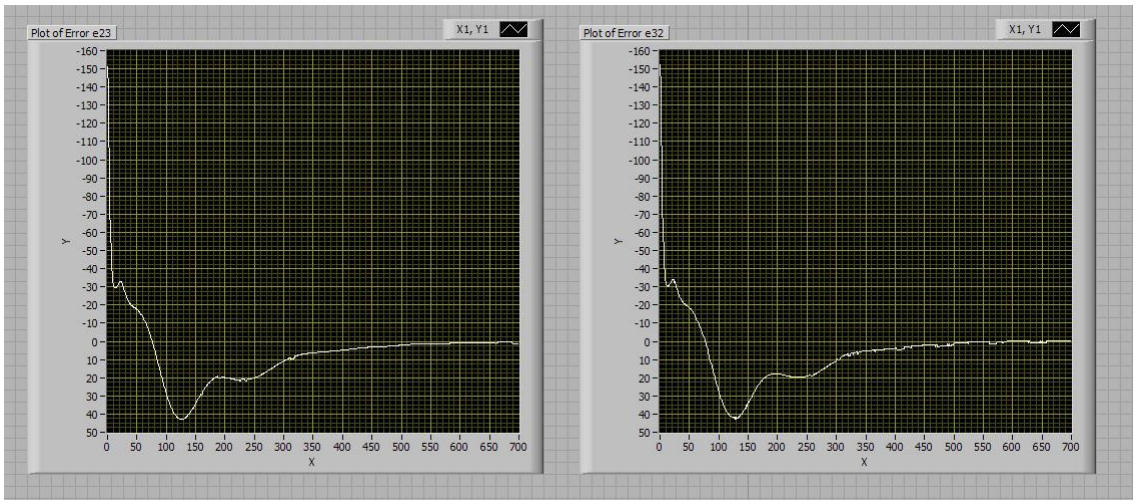


Fig 5.14 Illustration of the error between the actual distance and estimated distance between nodes 2 and 3 for absolute localization.

5.6. Conclusion

The error in case of the distance estimation for all sets of distances between the nodes for both relative and absolute localization algorithms have been reduced considerably. Also since the localization is performed using EKF over a set of readings, any effect of noise in a few readings will be corrected by the recursive filter. Thus both relative and absolute localization has been successfully achieved using Extended Kalman Filter.

CHAPTER 6

SWARM FORMATION

6.1. Introduction

Robot navigation has been receiving a lot of attention recently. The idea of using automated and intelligent machines capable of motion for performing tasks in hazardous environment has spurred a lot of research into the field of robotics. Robots have also been deployed to monitor warehouses. In such cases, it is important that the robot has a good idea about it's location as well as the location of the target. The positions can be determined by using standard localization schemes like GPS. However, for indoor environment, GPS can't be used. Instead other localization schemes like the one mentioned in the previous chapter can be used to localize the robot as well as track its motion. Sometimes a swarm of robots are deployed to perform a task. In this case the motion of all the robots has to be coordinated so that all reach the target. This chapter mentions on basic swarm behavior theory and considerations, the robot used for the demonstration of swarm behavior, and the demonstration.

6.2. Robots and Swarm Formation

Swarm robotics is the study of how large number of relatively simple physically embodied agents can be designed such that a desired collective behavior emerges from the local interactions among agents and between the agents and the environment[23].

The aim is to perform a set of tasks through coordination between multiple robots. Swarm behavior is inspired from observation of animal behavior. Migratory birds fly in formations. A figure [24] of geese flying in V- formation is shown below.

Fig. 6.1 Geese flying in V-formation.

Swarm formation involves a leader and the remaining robots as followers. The leader decides the motion trajectory for the entire swarm. In case of obstacle free environments or in a fixed-trajectory motion where no obstacles are present, the leader can send commands to the followers which can then move using a pre-determined set of

motions. As there is no obstacle, the followers needn't be assigned motion processing capability. Path planning is performed by the leader in such cases.

Communication between the leader and the followers is important in swarm formations. The leader needs to send motion commands to the followers and update the path to the target with current motion. If there is error in communication then the swarm formation doesn't stay in place as the follower motion isn't synchronized with that of the leader. Wireless communication along with proper node identification can be used to establish communication channels between the swarm.

Tracking the motion of either the swarm or the leader can be performed by using the localization method mentioned above or by the leader itself. In the earlier approach, the base station tracks the leader and updates its position and projected trajectory and then sends commands to the robot. In the latter approach the leader is provided with coordinates of it initial position and it updates its position as it moves. This can be performed if the leader can determine its displacement over a period of time. In case of most robots the motion motors have some sort of encoders which track the motion and update the count.

6.3. Robot Garcia

The robot Garcia [25] manufactured by Acroname Inc was used as the target for the implementation of the swarm demo along with the Cricket sensor mote. Garcia is a fully operational robot platform. The hardware to guide and control the motion of Garcia is already present. The software to plan the motion needs to be programmed to make Garcia move. By passing commands to the motion controller on-board Garcia the swarm formation algorithm can be tested.

6.3.1. Features of Garcia

6.3.1.1. Processor and Memory

Garcia has two separate 40 MHz processors : BrainStem Moto 1.0 , which handles motion control and sensor inputs, and BrainStem GP 2.0 which controls the serial interface, IR communication capability and additional I/O. The processors are, in fact, PIC18C252 microcontrollers with 32K EEPROM for storing code. [26, 25]

6.3.1.2. Battery

Garcia is powered by a standard 6-cell 7.2 V 3000mAH NiMH battery pack. There is also an option for direct power source connection.

6.3.1.3. Serial Port

Garcia has a serial port with which a serial communication channel with external devices is established. Using this port, commands can be sent to direct the motion of Garcia. This port has been used to send commands through the cricket to Garcia for the swarm demo.

6.3.1.4. Motor

Garcia has two Maxon A-max motors which can also be programmed to be used in PID Encoder mode for controlling Garcia motion besides normal mode of operation. The motors have quadrature encoders which keep track of the movement of the Garcia. Thus the distance traveled by Garcia can be calculated by taking the difference of the encoder readings at the starting point and the finishing point. [27] provides the exact count of the encoder readings for a 360^{o} turn as well a translational motion of a foot by the Garcia.

6.3.2. Garcia Motion Command

A list of commands which have been used to direct the motion of Garcia has been reproduced here from [7] . The commands, which are a series of hexadecimal bytes, and the corresponding effect on Garcia have been presented in a tabular form.

Table 6.1 Garcia Motion Commands and effect.

Garcia Command Array	Effect on Garcia
{0x02,0x02,0x00,0x00,0x02,0x04,0x1B,0x01,0x00,0x04, 0x04, 0x3E,0x00,0x00,0x07,0x04,0x04,0x3E, 0x01,0x00,0x07}	Move forward.
{0x02,0x02,0x00,0x00, 0x04,0x04,0x3E,0x00, 0xFF,0xF9, 0x04,0x04,0x3E,0x01,0x00,0x07}	Turn right.
{0x02,0x02,0x00,0x00, 0x04,0x04,0x3E,0x00, 0xFF,0x07, 0x04,0x04,0x3E,0x01,0x00,0xF9}	Turn left
{0x02,0x02,0x00,0x00, 0x04,0x04,0x3E,0x00, 0xFF,0x00, 0x04,0x04,0x3E,0x01,0x00,0x00}	Stop
{0x02,0x02,0x00,0x00, 0x04,0x04,0x3E,0x00, 0xFF,0x09, 0x04,0x04,0x3E,0x01,0x00,0x07}	Left curve
{0x02, 0x02, 0x00, 0x02, 0x02,0x02,0x19, 0x82,0x04,0x02,0x42,0x02}	Encoder readings access.

These are the basic set of commands sent to Garcia on the serial port by Cricket to get the encoder readings as well as for motion. A variation in the Garcia speed can be achieved by changing a couple of bytes in the command to move Garcia forward.

The figure below show Garcia robot with Cricket mote attached to its serial port.

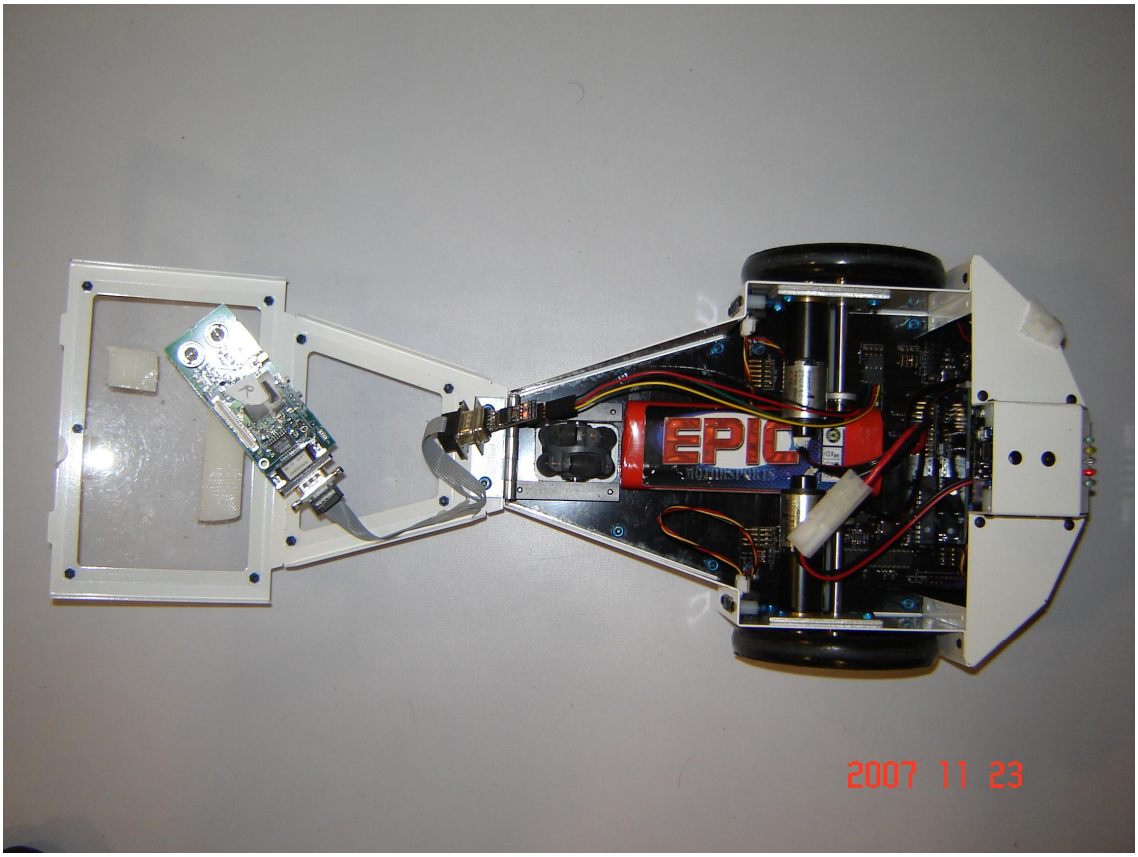


Fig 6.2 Garcia with Cricket mote attached to Serial port.

6.4. Swarm Demo

The swarm demo is built with three Garcia which form a triangular swarm as shown in the figure below.

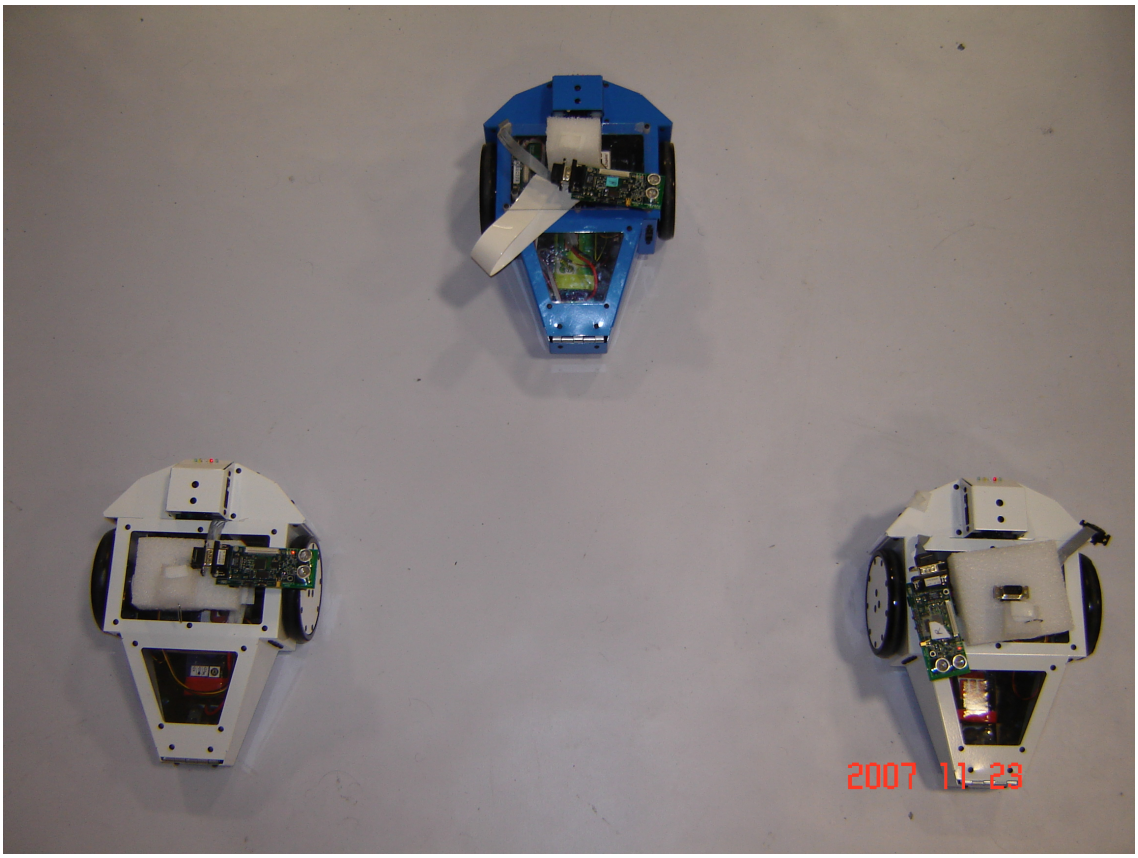


Fig 6.3 V-Swarm formation with three Garcia.

Initially, the cricket connected to the leader and two other crickets, the exact position of one of which is known, can be localized by using the above localization scheme. Except for the cricket connected to the cricket the other crickets are stationary. Once the localization is performed, the co-ordinates of current position and the

destination co-ordinates are sent to the leader over radio. A LabVIEW interface has been implemented to send these commands to the leader.

The crickets connected to the leader and the left and right follower are programmed with different code. The leader decides the path to be taken to reach the target and it sends the commands to the followers over radio. Once the cricket connected to the followers receive the commands they instruct the connected Garcia to perform a fixed set of movements. The communication between the Garcia and the cricket sensors occurs over the serial port as indicated in above diagram. The motion of the Garcia is achieved by sending the commands listed in table 6.1. The cricket connected to the leader uses the command to get encoder readings and then calculate it's displacement from the difference in encoder readings. Thus, the leader can calculate it's exact position and reach the destination. The leader sends commands to the followers and directs the entire swarm to the target.

CHAPTER 7

CONCLUSION, APPLICATION AND FUTURE WORK

7.1. Conclusion

The algorithm for the localization using extended Kalman filter has been successfully tested for both relative as well as absolute localization. The potential field approach for localization can now be applied even in noisy environment where noise affects the distance measurement and ranging. The experiment has been performed using three nodes for distance measurement, but the approach implies no limitation on the number of nodes being localized.

The nodes capable of measuring distance from each other can then be deployed in any environment and their relative and/or absolute positions can be determined. This approach to localization is most suitable for wireless sensor networks. A lot of techniques can be used to measure distances, one of the easiest being Ultrasound signals, which need line of sight communication for accurate estimation, and any one can be chosen. If the sensors have enough processing capacity then the localization can be performed in the nodes themselves.

The problem of localization in noisy environments with potential field approach can be solved using extended Kalman filters.

7.2. Application

Wireless sensor networks have been deployed in a lot of places. Environmental monitoring is an application where there can be lot of noise sources which affect the ranging. In such cases EKF can be used to decide the position of the sensors. Another example is monitoring of animals in a closed environment such as nature parks. A networks of sensors can be deployed in the closed environment. The animal is tagged with a ranging device. When the animal is near other ranging devices, the ranging can be performed to fix the location as the location of the other nodes is known already.

Ranging devices can be deployed in warehouses and also attached to valuable equipment. Any movement of the equipment can be detected by a change in its location. The algorithm can be used to track the location of the equipment within a large facility as well.

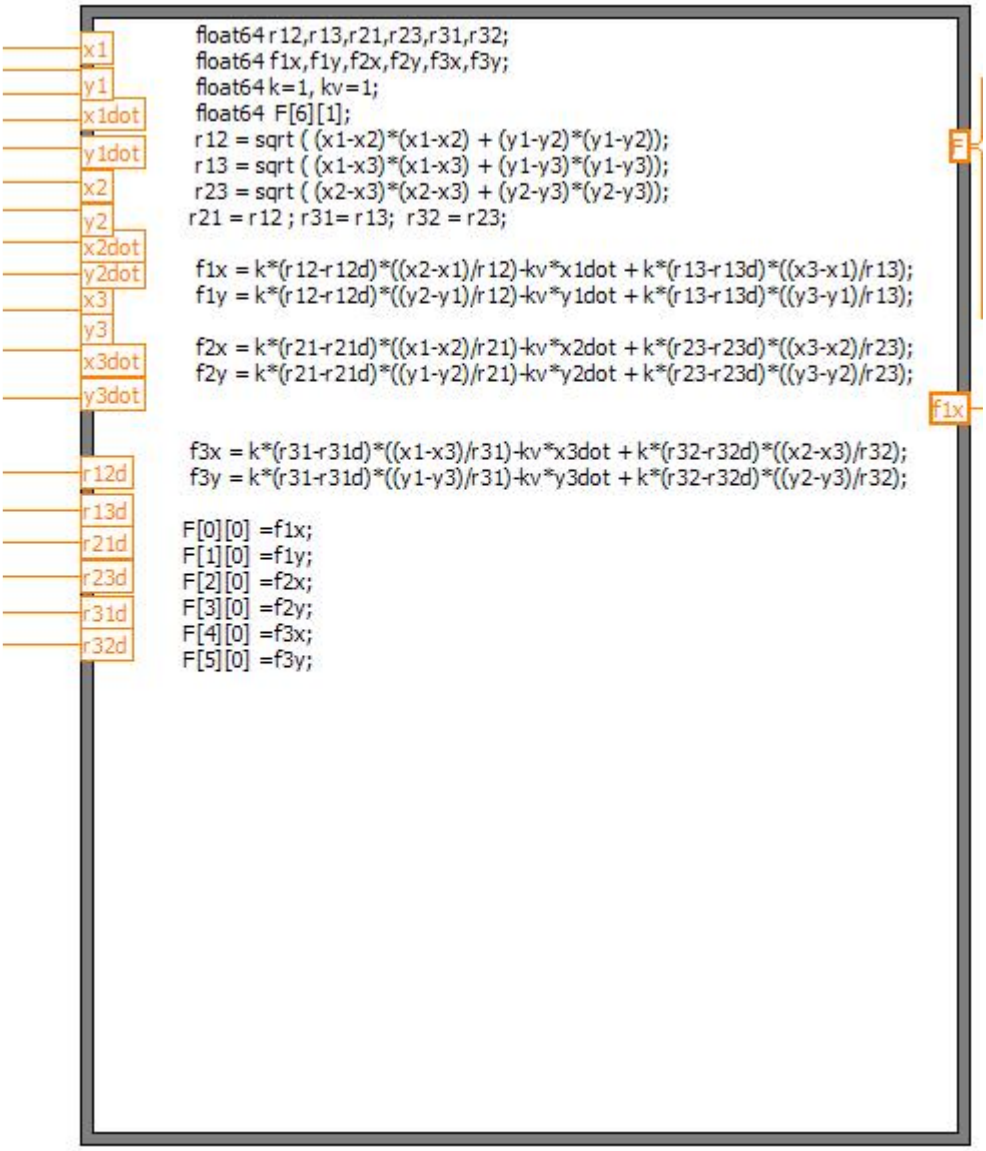
7.3. Future Work

Swarm can be formed with more robots and with different formations. Using motes with more storage than cricket localization can be implemented on the same motes as the ones governing the robot motion.

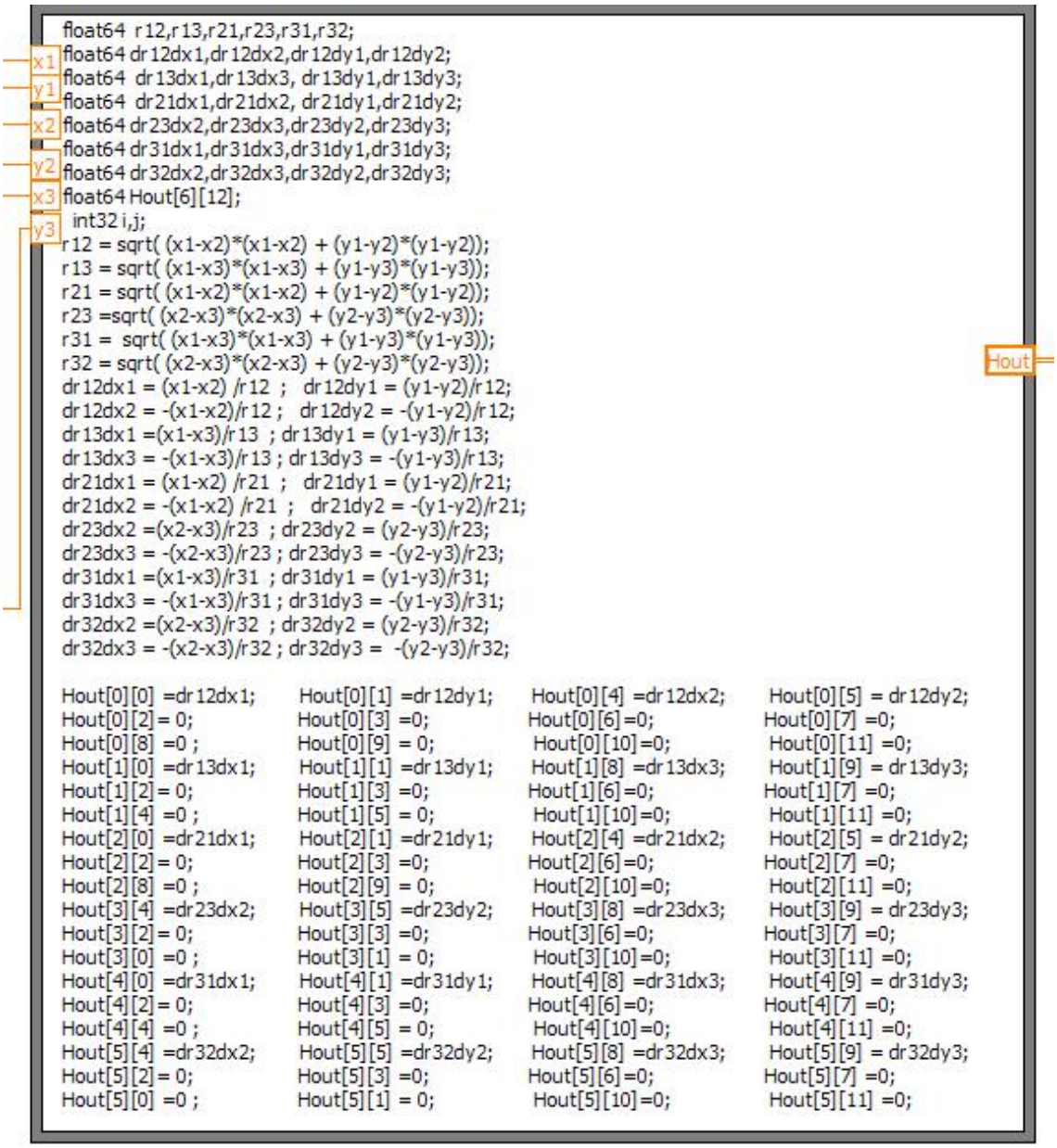
Localization using EKF can be performed with more nodes. As more nodes are added to the network they will be automatically localized.

APPENDIX A

LabVIEW APPLICATION FOR LOCALIZATION



A part of the LabVIEW application to calculate the motion update in EKF.



Part of the LabVIEW application to calculate the Jacobian for EKF.

REFERENCES

[1] Kazem Sohraby, Daniel Minoli & Taieb Znati, *Wireless Sensor Networks. Technology, Protocols and Applications*. John Wiley & Sons, Hoboken, New Jersey,2007.

[2] Nirupama Bulusu, Sanjay Jha, Wireless *Sensor Networks*, A Systems Perspective. Artech House, MA,2005.

[3] Karl H. and Willing A., "*Protocols and Architectures for Wireless Sensor Networks,*" John Wiley & Sons,2005.

[4] Savvides A., Girod L., Srivastava M and Estrin D., "Localization in Sensor Networks," in *Wireless Sensor Networks*, CS Raghavendra, K.M. Sivalingam, and T.Znati, Eds. Norwell,MA:Kluwer,2004.

[5] Masoomeh Rudafshani, Suprakash Datta, Localization in Wireless Sensor Networks IPSN'07 April 25-27 2007. Cambridge, MA, USA. ACM,2007.

[6] P.Ballal, *Wireless Sensor Networks* (to be published) NST Publications,2007

[7] A.Trivedi . *Discrete Event Controller : Application Using Dynamic Resource Allocation*, Masters Thesis, UTA,2007.

[8] CrossBow Technology, http://www.xbow.com

[9] Cricket Sensor Resource , http://cricket.csail.kit.edu

[10]CricketDatasheet,http://www.xbow.com/Products/Product_pdf_files/Wireless_pdf/MCS410_Cricket_Datasheet.pdf

[11] Cricket User Manual. http://cricket.csail.mit.edu/v2man.pdf

[12] Mica Product information.

http://www.xbow.com/Products/Product_pdf_files/Wireless_pdf/MICA.pdf

[13] TinyOS source . http://tinyos.net

[14] TinyOS Tutorial. http://www.tinyos.net/tinyos-1.x/doc/tutorial/index.html

[15] NesC Tutorial. http://www.tinyos.net/tinyos-1.x/doc/nesc/ref.pdf

[16] Wikipedia . http://en.wikipedia.org/wiki/Labview

[17] nesC : http://nescc.sourceforge.net

[18] *The nesC Language : A Holistic Approach to Networked Embedded System.s,* David Gay, Philip Levis, Robert von Behren, et al. PLDI '03, June 9-11, 2003, San Diego,California,USA. ACM 1-58113-662-5/03/0006.

[19]Connor Todd, et al. *WISER Group*, University of Houston, Houston,Texas,USA.

[20] MIT Cricket webpage : http://cricket.csail.mit.edu

[21] Pritpal Dang, Prasanna Ballal, Frank Lewis, Dan Popa, *Real Time Relative and Absolute Dynamic Localization of Air-Ground Wireless Sensor Networks*.

[22] Greg Welch, Gary Bishop., *An Introduction to the Kalman Filter*. http://www.cs.unc.edu/~welch/media/pdf/kalman_intro.pdf

[23] http://www.swarm-robotics.org/

[24] http://www.aerospaceweb.org/question/nature/q0237.shtml

[25] http://www.acroname.com/garcia/garcia.html

[26] Acroname Inc., http://www.acroname.com/garcia/man/man.html , Garcia Manual.

[27] Barnes B., *A BrainStem Tutorial using Palm Vx and Garcia* , July 2005.

BIOGRAPHICAL INFORMATION

Aneeket S Patkar was awarded his Bachelor of Engineering degree in Electronics and Telecommunication from Mumbai University, India in 2003. He pursued his Master of Science Degree at University of Texas at Arlington. He developed a liking for embedded programming, which combined with his interest in wireless communication, inspired him to join Wireless Sensor Network Lab at Automated Robotics Research Center at UTA.