12-2012

Rule-Based Precise Localization Services in Mobile Environments

Mohammed Elbes
Western Michigan University, cie_just@hotmail.com

RULE-BASED PRECISE LOCALIZATION SERVICES IN MOBILE ENVIRONMENTS

by
Mohammed Elbes

A Dissertation
Submitted to the
Faculty of the Graduate College
in partial fulfillment of the
requirements for the
Degree of Doctor of Philosophy
Department of Computer Science
Advisor: Ala Al-Fuqaha, Ph.D.

Western Michigan University
Kalamazoo, Michigan
December 2012

RULE-BASED PRECISE LOCALIZATION SERVICES IN MOBILE ENVIRONMENTS

Mohammed Elbes, Ph.D.

Western Michigan University, 2012

In recent years there has been great interest in the positioning of people and assets in indoor and outdoor environments. Many systems were constructed to accomplish this. The application of these systems depends on the environment and the precision required for localization. The Global Positioning System (GPS) cannot be used for a precise indoor localization due to the attenuation and scattering of the signals. For indoor environments, other technologies like Wireless LAN, laser, camera images and motion sensors are used in precise localization. GPS can be used in outdoor environments but if localization accuracy is a critical requirement, other technologies should be used.

An intersecting application scenario for precise pedestrian localization is for the blind and visually impaired. This important segment of our society faces several challenges during daily activities like path planning, navigation and obstacle avoidance. Many of the blind still trust and rely on the white cane to explore their environments. The state-of-the-art white canes have audio systems to guide the blind through their environment. Continuously wearing a headset and hearing sounds can

be irritating for some people. Our work provides techniques to fuse data from multiple cost-effective sensors such as Accelerometers, Gyroscopes, Time Difference of Arrival (TDoA) sensors, Wi-Fi and AM radio signals of opportunity. These techniques allow for building autonomous navigation systems for the blind and visually impaired.

ACKNOWLEDGMENTS

First and foremost, I thank God for giving me the health and the chance to accomplish my work on time.

I would like to especially thank my thesis advisor, Professor Ala Al-Fuqaha for his guidance to me throughout my master study at Western Michigan University. I thank him for his support and encouragement for me during my research. I gained so much knowledge and experience from him which increased my research abilities and experience. I hope this work will go some way toward expressing my thanks.

I would like to thank my thesis committee members: Professor Dionysios Kountanis, Professor Ammar Rayes, and Professor Driss Benahddou. Their valuable feedback on my work highly improved my thesis quality.

Last but not least, I want to extend my warm thanks for my beloved family. Especially my parents who are major reason for my success in my studies, thanks to my wife for her support during my study at WMU. Also, thanks to my brother Ahmad and for my beloved sisters as they didn't stop encouraging me. Finally I would like to dedicate this work for my beloved mother, father, wife and my buddy, brother Ahmad.

Mohammed Elbes

TABLE OF CONTENTS

Table of Contents–Continued

CHAPTER

Table of Contents–Continued

CHAPTER

Table of Contents–Continued

CHAPTER

Table of Contents–Continued

CHAPTER

LIST OF TABLES

LIST OF FIGURES

List of Figures–Continued

List of Figures–Continued

CHAPTER 1

INTRODUCTION

1.1 Background and General Overview

The fast growing manufacturing of mobile devices instrumented with positioning technologies stimulated the development of Location Based services (LBS) [1]. These services provide users with their geographical location as they navigate through their environment. Many technologies are already available to develop these services like Wi-Fi [2, 3], AoA based systems [4] and UWB [5]. In our work we utilize low cost MEMS sensors along with TDoA based sensors to perform PDR.

The main goal of localization is to track moving objects. Localization can be performed in buildings and closed environments (Indoor Localization) or outside buildings (Outdoor Localization). Many surveillance and tracking applications rely completely on the knowledge of the position of objects in the environment.

There are many real world applications that depend on this automation process. Location detection of personnel or medical equipment inside a hospital, moving assets inside a store, intelligent guidance, location-aware multimedia services are examples of these applications[6-8].

1.2 Problem Statement and Research Goals

The positioning of people and assets in indoor and outdoor environments has been the target of many research groups recently. To accomplish this, many systems

were constructed and the application of these systems depends on the environment and the precision required for localization. The Global Positioning System (GPS) cannot be used for a precise indoor localization due to the attenuation and scattering of the signals. For indoor environments, other technologies like Wireless LAN, laser, camera images and motion sensors are used in precise localization. GPS can be used in outdoor environments but if localization accuracy is a critical requirement, other technologies should be used.

The complexity of indoor/outdoor environments resulting from multipath propagation and frequent environment changes requires using more than one technique to improve localization accuracy. Data from wireless LANs, Ultra Wide Band (UWB), Infrared (IR), Ultrasound, camera images and Inertial Measurement Units (IMUs) were fused by multiple data fusion techniques to provide more precise and robust localization systems. These systems are evaluated based on location accuracy, cost, range, and the data rate. Tradeoffs between these evaluation parameters exist in each of these systems.

Our target application for precise pedestrian localization is for the blind and visually impaired. This important segment of our society faces several challenges during daily activities like path planning, navigation and obstacle avoidance. Many of the blind still trust and rely on the white cane to explore their environments. The state-of-the-art white canes have audio systems to guide the blind through their environment. Continuously wearing a headset and hearing sounds can be irritating for some people.

Current blind assistance navigation systems provide the position or the orientation of the pedestrian which comprises only a part of a complete and comprehensive social network. These systems do not provide sharing of user experience with their environment and neither have they provide tagging for points of interest in a certain environment. In this work, we provide a software architecture for a complete social network for the blind and visually impaired and will illustrate how the user experience will be shared with all users to maximize network benefit.

Many of the state-of-the-art smart phones are equipped with accelerometers, gyroscopes and magnetometers. Our work provides techniques to fuse data obtained from the low-cost sensors on these smart phones. These techniques allow for building autonomous navigation systems for the blind and visually impaired.

The conventional approaches for Pedestrian Dead0Reckoning (PDR) used in [9, 10] track the position of a pedestrian by integrating the accelerometer and gyroscope signals. Heading is obtained from a heading sensor like a digital compass. However, using signals from these technologies alone is susceptible to error due to inherent instrumental errors and environmental disturbances.

In this work we proposed gyroscope drift correction approach using Time Difference of Arrival (TDoA) technology. The advantages of using TDoA over magnetometers for drift correction is that it is less susceptible to magnetic noise sources in the environment. This enables the usage of the proposed drift correction approach along with the PDR techniques described in this work in order to perform precise localization of pedestrians in both indoor and outdoor environments.

1.3 Dissertation Outline

Our dissertation is structured in eight chapters including the introduction one. In this section we provide the overall outline of our Rule-Based localization services in mobile environments.

In chapter two, we provide a detailed literature review of the previous work in localization for both indoor and outdoor environments. We present the approaches to localization and discuss the advantages and disadvantages of these approaches. Furthermore, we discuss the approaches to PDR and discuss the pros and cons of these approaches.

Since we use Particle Swarm Optimization (PSO) as our efficient tool for optimization, we also present a survey about PSO in this chapter.

Our approach to gyroscope drift correction based on TOA technology in support of pedestrian dead reckoning is presented in chapter three. The chapter starts by providing a brief introduction about PDR and discusses our approach to pedestrian step detection. Our approach to symbolic representation of accelerometer signal and the Flip-Flop filter design is also presented in this chapter. Our novel approach to PDR using TDoA technology is discussed in detail and the chapter concludes with the results obtained from simulation about PDR.

Our approaches to indoor and outdoor localization are presented in chapters four and five respectively. The chapters starts by providing a brief introduction about indoor/outdoor localization and then discuss our Budgeted Dynamic Exclusion (BDE) technique that is used both in indoor and outdoor approaches.

The Particle Filter (PF) approach to fuse data from multiple sensor technologies is discussed next followed by a discussion of the results obtained both in indoor and outdoor localization techniques.

Chapter six presents an example application of our precise localization service which is a social network for the blind and visually impaired. We present our vision to this social network and provide preliminary software architecture for this social network and present some basic use cases.

Chapter seven discusses the software testing approach. The chapter starts by listing some of the most common definitions of software testing. The most important requirements to software testing are discussed next followed by an explanation of the general software testing principles and techniques. The chapter also provides and illustration of the V-Model approach to software testing and explains each stage of this model.

Rule-Based expert systems are presented in chapter eight. The chapter starts by providing a brief introduction about Rule-Based expert systems and lists some definitions for the expert system concept. Expert system components and the functionality of each component is discussed next followed by an explanation of the Rule-Based expert systems.

The validation and verification of Rule-Based expert systems, challenges to Rule-Based expert system's testing and recommendations to the process of verifying and validating Rule-Based expert systems are also discussed in this chapter. The

dissertation concludes in chapter nine in which we provide an explanation for all the results that we obtained in this research

CHAPTER 2

LITERATURE REVIEW

2.1 Overview

In this chapter we provide a literature review of the localization techniques in both indoor and outdoor environments. We also discuss the various sensor technologies used in these approaches and discuss the advantages and disadvantages of these technologies. Since we use Particle Swarm Optimization (PSO) as our optimization tool in this research, we start this chapter by presenting a detailed survey about PSO and provide some applications of PSO in engineering and networking.

2.2 Review of Optimization and its Applications in Engineering and Networking

Swarm Intelligence is a type of Artificial Intelligence that is based on the cooperative performance of the decentralized systems. Such systems are modeled by a set of agents that interact with each other and interact with its environment. Despite the fact that there is no centralized control on the behavior of these systems, the localized random behavior of these system leads to universal system intelligence. Examples of these systems are bird flocking, ant colonies and, fish schooling [11].

This section provides a detailed review of PSO. The general concept of optimization is discussed first followed by a review of other optimization techniques like the Genetic Algorithms (GA), Simulated Annealing (SA), Tabu Search (TS) and many other optimization techniques. A detailed description of PSO in the continuous

and binary spaces is provided next and the application of PSO in engineering and networking is also described in this section.

2.2.1 Optimization in Mathematics

Optimization is the study of maximizing or minimizing some objective function by finding the best variables' values. Generally, a representation of an optimization problem is of the form [12]:

$$min\ f(x) \qquad x \in S \subset R^n \tag{2.1}$$

According to the linear or nonlinear constraints

$$g_i(x) \leq 0,\ i = 1, \ldots, m, \text{ where m is the number of constraints in the problem} \tag{2.2}$$

A penalty function is used to solve most of the optimization problems. In this approach, two types of points exists in the search space of the problem; infeasible and feasible points. Infeasible points are the points which violate at least one of the problem constraints, while feasible points are the points satisfying the set of all the constraints.

The penalty functions are divided into two different sub categories: stationary and non-stationary functions. The non-stationary functions add vigorously alternating value for the penalty based in its distance of the infeasible point from the constraint. On the other hand, in the stationary functions, unchanging value is added to the constraint if it violates the objective function. [12].

The penalty function can be formulated as [12]:

$$F(x) = f(x) + h(k)H(x) \quad x \in S \subset R^n \tag{2.3}$$

Where *f(x)* is the objective function in equation 2.1, *h(k)* is a dynamically changing penalty function, *k* is the current algorithm iteration and *H(x)* is a penalty factor defined as [12] :

$$H(x)=\sum_{i=1}^{m} \Theta\,(qi(x))qi\,(x)\,\exp\,(\gamma\,(qi(x))\,) \qquad (2.4)$$

Where $q_i\,(x) = \max\,\{0, g_i(x)\}$, $\gamma\,(q_i(x))$ is the power of the penalty function, $\Theta(q_i(x))$ is a multi-stage assignment function and $g_i(x)$ are the constraints described in equation 2.2.

2.2.2 Taxonomy of Optimization Algorithms

In this section, we provide a detailed taxonomy of optimization. In general, optimization can be classified into three major categories [11]:

- **Optimization Algorithms**: The majority of these algorithms are designed for linear programming. Examples of optimization algorithms are simplex algorithm of George Dantiz, extensions of the simplex algorithms designed for Quadratic programming and Combinatorial Algorithms.
- **Iterative methods:** These methods are used for non-linear programming problems. Examples of these methods are the Newton's Method, Conjugate Gradient Method, Interior Point Methods and The Gradient Descent Method.
- **Heuristics:** In addition to the *finitely* terminating optimization algorithms and the *convergent* iterative method, Heuristic approaches provide an approximate solution to optimization problems. Examples of these approaches are the

Genetic Algorithms, Tabu Search, Harmony Search and Particle Swarm Optimization.

The aforementioned and more approaches are presented in Fig. 2.1. Since the main topic of this section is a heuristic-based approach (PSO), in this section we provide a detailed description for the major heuristic approaches. We also compare the performance of PSO and other heuristic approaches such as Genetic Algorithms (GA), Simulated Annealing (SA), Tabu Search (TS) , Harmony Search (HS), Stochastic Tunneling (ST) and the Cross Entropy (CE) Method.

1) Genetic Algorithms (GA): A search method that mimics the biological evolution. Given a target problem, the input to the GA is a set of candidate solutions generated randomly and evaluated according to a fitness function. In general, most of these solutions do not survive the evaluation process and are killed instantly. Multiple randomly modified copies of the survivors are generated and a pool of new generations of candidate solutions is constructed. This process of generating, evaluating and modifying the best solutions is repeated for several hundreds or thousands of rounds until the algorithm converges to an acceptable solution [13-16].

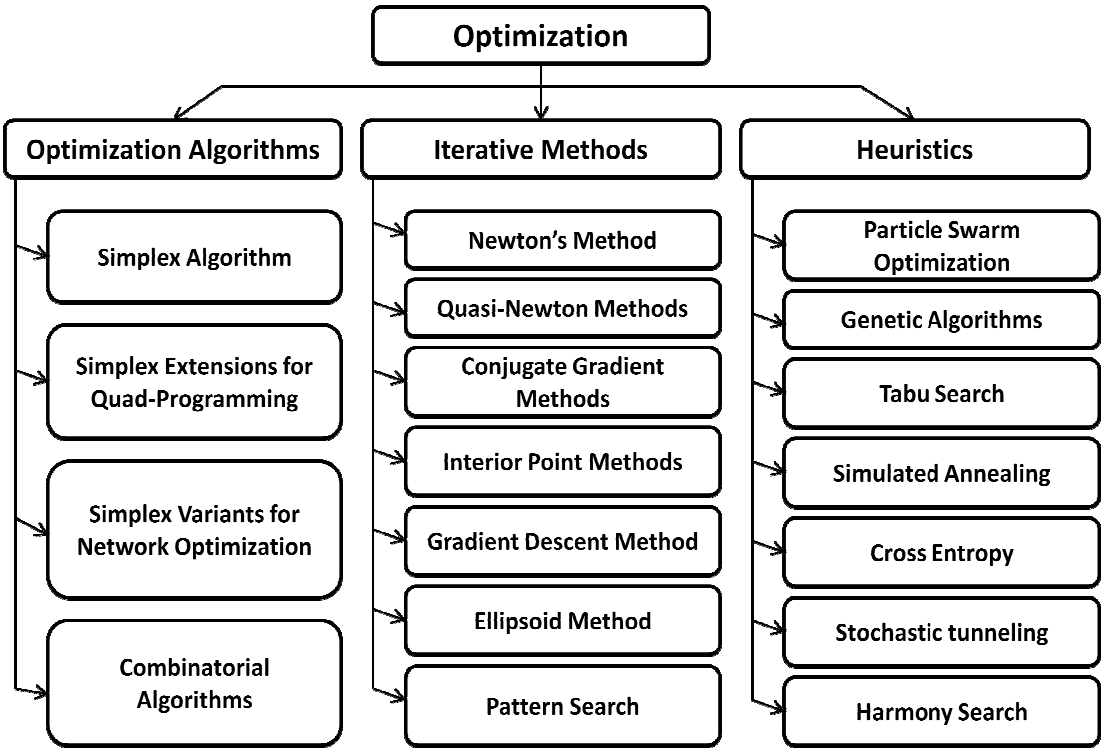

Figure 2.1: Taxonomy of optimization.

2) Tabu Search (TS): The goal of TS is to prevent the search procedure from falling in a local optimum by keeping track of the search paths that already visited by the search procedure. This can lead the algorithm to accept some inferior solutions to avoid revisiting previous paths to find the best solution through more globalized search. The visited search paths (forbidden solutions) are stored in a tabu list which is maintained by a forbidding strategy that decides which solutions are candidates and to be kept in the tabu list [17-20].

3) Simulated Annealing (SA): This method mimics the process of crystallization from a melt. The atoms of a melt are free to move at high temperatures

and when cooling the melt sample, these atoms start to crystallize into a solid. If the melt is cooled quickly, the melt becomes amorphous and if it is annealed (cooled) slowly the melts becomes a perfect crystal which is considered the global minimum energy configuration of the system. The goal is to find the best annealing schedule that converts the melt into a perfect crystal [21].

4) Harmony Search (HS): Another meta-heuristic optimization approach that mimics the musical improvisation. Each musician corresponds to a variable in the fitness function and the pitch range of each musical device corresponds to the range of values a variable can have. A candidate solution is represented by an improvised harmony. The more the musicians practice, the better harmonies created and correspondingly better solution vectors will be produced [22-25]. The algorithm works by randomly generating solution vectors (harmonies) and then disturbing these vectors according to HS algorithm to get the global minimum (best harmony).

5) Stochastic Tunneling (ST): a global optimization technique was originally proposed for minimizing the energy function in complex rugged Potential Energy Surfaces (PES). In this method, the dynamical process explores a transferred adaptively changing version of the PES not the original one. The idea of this algorithm is to flatten the energy surface in all areas that have a value for energy above a certain threshold [26]].

6) The Cross Entropy Method (CE): This is a new generic method used in combinatorial optimization and rare event simulation. The CE is an iterative method in which each iteration is comprised of two phases. The first phase is generating

random data samples using a specified mechanism followed by the second phase of updating the parameters of the mechanism to produce better results in the next iteration. The power of the CE method stems from the fact that it produces a framework to derive optimal learning rules obtained from the theory of simulation. In this approach, the deterministic problem is converted to a stochastic problem. After that, the CE method is used to solve the problem [27-29].

2.2.3 Standardization of PSO

In the original PSO, Euclidean neighborhood was used for information sharing between particles. This approach, also known as the *lbest* model, has high computational complexity which was the reason for replacing it with topological neighborhood. In this approach, also known as the *gbest* model, each particle has a global knowledge of all particles of the swarm [30]. Using one of these models is application dependent. In general, the local best approach has slow convergence rate unlike the *gbest* model which has high convergence rate that can trap the algorithm into a local minimum.

In the original PSO, the velocity and position update equations are [30]:

$$v_{id} = v_{id} + c_1 r_1 (p_{id} - x_{id}) + c_2 r_2 (p_{gd} - x_{id}) \quad (2.5)$$

Where r_1 and r_2 are random numbers uniformly distributed between 0 and ,1 c1,c2 are the cognitive and social parameters. p_{id} is the particle's best position and p_{gd} is the swarm's best position [30].

$v_{id} = V_{max}$ if $v_{id} > V_{max}$

$v_{id} = - V_{max}$ if $v_{id} < -V_{max}$

$$x_{id} = x_{id} + v_{id} \quad (2.6)$$

In order to avoid large values for the velocities which cause the swarm to explode after several iterations, the particles' velocities are clamped by setting a maximum velocity V_{max}. Having a fixed value for V_{max} is not applicable for all search spaces. Large search spaces require high values of V_{max} allowing for adequate exploration of the search space. This is not the case for small search spaces which require small values of V_{max} to avoid the swarm from exploding out of the feasible solution space.

Incorrect choice of V_{max} can lead to poor performance while there is no precise method for choosing a value for V_{max} beyond trial and error. For this reason, the inertia weight w is introduced to replace V_{max} and equation 2.5 becomes:

$$v_{id} = w.v_{id} + c_1.r_1.(p_{id} - x_{id}) + c_2.r_2(p_{gd} - x_{id}) \quad (2.7)$$

The inertia weight has an initial value greater than 1 to allow for more exploration of the search space. The inertia weigh then decreases reaching a value that allows for in detail exploration around the global optimum.

In [31], a study for the choice of the inertia weight value was carried out. The authors concluded that the use of the inertia weight for controlling the velocity induces high efficiency of the PSO. A carefully chosen value of the inertia weight allows for providing a balance between the local exploitation and global exploration of the search space. Setting the inertia weight to a random value between [0, 1] is

better than setting the inertia to a maximum value and linearly decrease this value until it reaches zero. The linear decrease of the inertia weight can trap the in local minima instead of a global one.

Another method for this adaptive search in the search space was the introduction of the Constriction Factor *X* which is defined by [31]:

$$X = \frac{2}{\left| 2 - \varphi - \sqrt{\varphi^2 - 4\varphi} \right|} \tag{2.8}$$

Where $\varphi = c_1 + c_2$

It was found that when $\varphi < 4$, the convergence around the best fitness value is slow but not guaranteed. However, when $\varphi > 4$ the convergence is fast and is guaranteed. A typical value for *X* is around *0.7* which results when $c_1 = c_2 = 2$, and the velocity equation becomes [31]:

$$V_{id} = X * [\, v_{id} + c_1 r_1 (p_{id} - x_{id}) + c_2 r_2 (p_{gd} - x_{id}) \,] \tag{2.9}$$

The initial number of particles has a great impact on the performance of the optimization technique. Small number of particles can lead to insufficient explorers resulting in local optima.

On the other hand, a high number of particles leads to unnecessary processing degrading the performance of the optimization algorithm. Empirical results show that a number of 50 particles provides good results with many objective functions. In general 20-100 particles are usually acceptable [30].

2.2.4 PSO in Continuous Space

Kennedy and Eberhart introduced in PSO 1995 [32]. They founded techniques to optimize continuous nonlinear mathematical functions. The ideas of this technique were obtained from social psychology, artificial intelligence, and swarming theory [32]. The algorithm simulates swarms of animals looking for foods like bird flocks and fish schools. This technique represents a problem by randomly created set of particles. These particles move in the search space observing the particle with best value for the objective function

The i^{th} particle in the swarm is represented by a *D*- dimensional vector $X_i = (x_{i1}, x_{i2}, \ldots, x_{iD})$, the particle' velocity $V_i = (v_{i1}, v_{i1},\ldots,v_{iD})$ and the particle's best position denoted as $P_i = (p_{i1}, p_{i2}, \ldots, p_{iD})$. The dynamics of the particle in the solution space is governed by the following equations [32]:

$$V_i^{k+1} = x\,[\, w\, V_i^k + c_1\, r_{i1}^k \left(P_i^k - X_i^k\right) + c_2\, r_{i2}^k \left(P_g^k - X_i^k\right)\,] \tag{2.10}$$

$$X_i^{k+1} = X_i^k + V_i^{k+1} \tag{2.11}$$

Where $i \in [1, N]$ and N is the size of the swarm.

The relative magnitude between $r_1.c_1$ and $r_2.c_2$ determines whether the particle travels in the direction of *gBest* or *pBest*. If the upper bound of $r_1.c_1$ is higher than the upper bound of $r_2.c_2$, the particle benefits from the neighborhood experience more than its own experience.

A. The PSO algorithm

The algorithm of the PSO can be written as follows [33]:

"For each particle
{
Do
{ Initialize particle } }
Do
{
For each particle
{
Calculate the corresponding fitness value
If the fitness value is better than the particle's best fitness value
Set the current P vector to the particle's current X vector
}
Choose the particle with the lowest fitness value and make it the global best position
For each particle
{
Calculate the particle's velocity according to Eq.11
Update the particle current position vector X
}
} while maximum iteration or minimum error criteria is not attained"

The algorithm starts by generating the swarm initialized particles. The position of each particle is calculated based on equations (2.10, 2.11). In each iteration, the particle compares between its current value for the position and its best position. If the current value is better than the previously calculated best position, the particle's position is assigned to the current value. The swarm's best particle is assigned to the particle with the best value for the objective function. The flow chart in Fig. 2.2 shows the PSO mode of operation.

The advantage of the PSO is that it does not require tuning many parameters in order to get acceptable performance [34]. Furthermore, the parameters of the PSO are problem dependent and it is not trivial to find the best values for parameters.

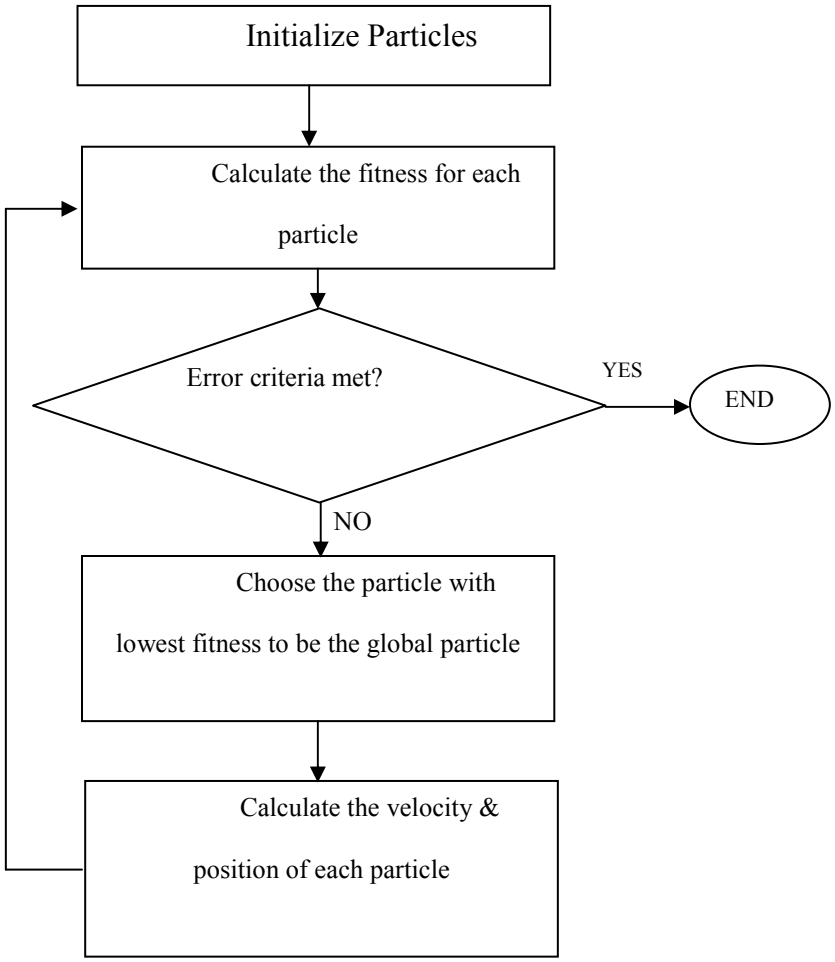

Figure 2.2: PSO mode of operation.

B. Modifications to PSO

In [30], a modification to the original PSO in [32] was presented. The modified PSO was applied in power system applications particularly in location and sizing of multiple STATCOM units in a power system. The goal was to find the best solution to improve the voltage profile of the power system at minimum cost. A comparison with the original PSO illustrated that the modification of velocity

equations allowed the search process to be more efficient in obtaining the best feasible solutions. The enhancement was in a form of basic logic added to the particles thus facilitating the search in the solution space. The logic was defined by the rules [30]:

"*If the particle is not yet in the feasible space then its velocity is defined as:*

$$v(t) = w_i v_i (t-1) + c \cdot r \cdot [\, p_g - x_i (t-1) \,] \qquad (2.12)$$

This means that the particle should rely on its neighborhood to get into the feasible space rather than on its current position. If none of the particles in the swarm are in the feasible space, the maximum velocity of each particle is set to a random number so that the particles move erratically in the search space trying to find one feasible particle. If the particle local best and the swarm's global best are both feasible, then the original canonical PSO is applied".

Simulation results illustrated that the enhanced PSO suggested in [30] shown significant gains in the performance when comparing with the canonical PSO, higher ability to pinpoint the feasible solutions and it found the optimal solution for the problem in hand.

Another modification to the PSO called New PSO (NPSO) was addressed in [35]. The goal was to guide each particle in the swarm to move away from its previous worst position and the group's worst position. The previous worst position of a particle is represented by $P=(p_{i1}, p_{i2}, \ldots, p_{iD})$, the swarm's worst position is g and the change in position is $\Delta X_i=(\Delta x_{i1}, \Delta x_{i2}, \ldots, \Delta x_{iD})$. The velocity equations are similar to those of the original PSO:

$$\Delta X_{id}=\Delta X_{id}+c_1r_1(x_{id}-p_{id})+c_2r_2(x_{id}-p_{gd}) \quad (2.13)$$

$$x_{id} = x_{id} + \Delta X_{id} \quad (2.14)$$

This modification to the PSO was tested on benchmark functions (i.e. The sphere, Griewank, Rastrigrin and Rosebrock) that are used by many researchers. The results for specific settings of these functions were better than that of the PSO. The results also illustrated that this technique does not always outperforms the PSO and can be considered as a variation of the PSO instead of a better PSO algorithm.

An efficient speedup strategy based on the introduction of an adaptive scaling term into the original PSO was presented in [36]. The parameter V_{max} was multiplied by the scaling term $(1 - (t/T)^h)$, where t is the current generation number, h is a constant obtained from trial and error and T is the maximum number of generations. The modified equations become:

$$v_{id} = (1 - (t/T)^h)\ V_{max} \quad (2.16)$$

if $v_{id} > (1 - (t/T)^h)\ V_{max}$

$$V_{id} = -(1 - (t/T)^h)\ V_{max} \quad (2.17)$$

if $v_{id} < -(1 - (t/T)^h)\ V_{max}$

This scaling term $((1 - (t/T)^h)$ allows the algorithm to evolve with a descending searching scale.

The modified PSO in [36] was tested on four well known benchmarks for PSO. The modification resulted in a considerably better convergence performance than PSO. The modified algorithm also added more control on the PSO which is

achieved by introducing the parameter *h*. This parameter is introduces to control the speed of the searching scale.

2.2.5 Binary PSO

Many optimization problems that require a discrete ordering of discrete elements can be represented in the discrete or binary space. Scheduling and routing are examples of these problems. In the continuous space, the trajectories of particles are adjusted based on information about the previous best performance. In the binary space, the trajectories are changed based on the probability that a certain coordinate will change to one or zero. In other words, a particle may be seen as moving nearer or father on the corners of a hypercube [37].

The velocity of the particle is defined as the hamming distance between the particle at time *t* and at time *t+1* in the next iteration. This is represented in terms of probability changes that a certain bit will be zero or one. In general, v_{id}, will represent the probability that a bit will be equal to 1.

The PSO equations remain the same as equations (10, 11) except that p_{id} and x_{id} are integers in {0, 1}. Since the velocity is a probability, it will be in the [0, 1]. To achieve that, a logistic transformation $S(v_{id},$ is used to limit the velocity to be in [0,1]. The transformation S can be the sigmoid function which is defined as:

$$S(v_{id}) = \frac{1}{1+e^{-v_{id}}} \tag{2.18}$$

The resulting position change is defined based on the rule

$$x_{id} = \begin{cases} 1 & R < S(vid) \\ 0 & R \geq S(vid) \end{cases} \quad (2.19)$$

Where *R* is uniformly distributed random number between [0, 1].

While a higher value of V_{max} in the continuous space allows for more exploration of the search space, a lower value of V_{max} is required initially to explore the search space. This value should be decreased gradually when the algorithm is about to converge to the optimal solution.

A. Runtime Analysis of Binary PSO

The analysis of the runtime for the evolutionary algorithm is getting more attention in the recent years. This kind of analysis is usually hard because the underlying probabilistic model of the swarm algorithm usually depends on a history of past solutions [38]. The first analysis for the Ant Colony optimization was carried out in [39, 40]. In 1997, Kennedy and Eberhart [37] introduced a binary version of the classical PSO.

The lower bounds analysis of v_{max} illustrated that setting v_{max} to a constant value leads to better results only for problems with bounded sizes. However, this leads to dramatic performance decrease for problems with variable size. The authors proved that the probability that PSO finds the global optimal value when having at most 2^K global optima is equal to 2^{-K} where *K* is a positive constant. The upper bound of the binary PSO was found by setting the cognitive constant $c_1 = 0$. This means that each particle follows the leader of the swarm and ignores its best found solution [38].

B. Modifications to Binary PSO

In [41], the authors studied the shortcomings of the original binary PSO proposed in [37]. Their work illustrated that no clear choice was made for the value of the inertia weight in the original binary PSO.

The proposed algorithm in [42] interpreted the velocity in a different way than in [41]. The velocity is defined as the rate of change in the bits of the particle. The direction of change to one or to zero is maintained through the introduction of new vectors of velocity $V_1^{\rightarrow}$ and $V_0^{\rightarrow}$ for each particle. After these vectors are updated, the velocity change is obtained as in equation 2.18. The previous direction and previous state of each particle is also taken into account and provided better solutions.

Another modification to the original binary PSO was introduced in [42]. In this work, the authors provided a better solution to the partner selection problem which is a critical issue in the research of the virtual enterprise. After presenting the optimization model, an improved version of the binary PSO was designed to decrease the probability of the algorithm falling into a local optimum and to enhance the search ability. The modification targeted the particle position equation such that the particle velocity is divided into three regions. The state of the particle being one, zero or unchanged is determined by the particle's current region. The ranges of these regions change adaptively as the algorithm iterates to achieve global convergence. The velocity equation remains the same as in the original binary PSO, but the position equations became [42]:

If $0.5 - \delta \leq s(v_{i,j}^{k+1}) < 0.5 + \delta$ then $x_{i,j}^{k+1} = x_{i,j}^{k}$ (2.20)

If $s(v_{i,j}^{k+1}) \leq 0.5 - \delta$ then $x_{i,j}^{k+1} = 0$ (2.21)

If $s(v_{i,j}^{k+1}) \leq 0.5 + \delta$ then $x_{i,j}^{k+1} = 1$ (2.22)

The value of δ is initially 0.5 and decrease gradually in each iteration. Equation 2.20 allows each particle to keep its own inertia and prevents particles from moving to the same position falling in a local optimum. The simulation results in [42] illustrated that the proposed algorithm provides better performance and has quick convergence abilities.

The work in [43] addressed a special type of optimization problems when the solution is a set of integers in the discrete space. The target problem was the Sudoku puzzle and the modified PSO is called Integer PSO (IPSO). This algorithm uses the same velocity update equation as in the original binary PSO. The only difference is that instead of having one velocity in the N-dimensional search space, a separate velocity value for each variable is utilized. This means that each particle has N-dimensional velocity vector and each of the variables' velocity vectors is updated separately.

The velocity vector is scaled by a modified version of the sigmoid function to a get a value between 0 and 1. The goal of this modification was to change the sigmoid function to get high probabilities for large positive and negative velocities. Hence, the sigmoid function was changed to [43] :

$$SI(x) = \frac{2}{1+e^{-|x|}} - 1 \quad (2.23)$$

The proposed IPSO in [43] was compared with two other algorithms, the $(\mu + \lambda)$ Evolutionary algorithm, and the original PSO algorithm. The IPSO was tested for 50 Sudoku puzzles of different levels of challenge. The algorithm outperformed the original PSO in all runs but was outperformed by the $(\mu + \lambda)$ Evolutionary algorithm in some cases.

Another modification to the original PSO was performed in [44]. The goal was to design an algorithm for gene selection and tumor classification. The modification was updating the next position such that 10% of the particles are forced to move away from the *gbest* to avoid falling in local optima. The new suggested position update equations were:

If($0 < v_{iD} \leq a$), then $x_{iD}(new) = x_{iD}(new)$ (2.24)

If($a < v_{iD} \leq \frac{1+a}{2}$), then $x_{iD}(new) = p_{iD}(new)$ (2.25)

If($\frac{1+a}{2} < v_{iD} \leq 1$), then $x_{iD}(new) = p_{gD}(new)$ (2.26)

The experimental results in [45] illustrated that this modification increased the effectiveness of the search algorithm and presented a tool mining high dimensional data.

The authors in [38] suggested an inertia weight equation that prevents the original PSO from getting immaturely trapped in a local optimum. The approach followed the evolutionary approaches to find a near optimal solution like Genatic Algorithms [14], Ant Colony Optimization [46] and Tapu Search [47]. The goal was to find a combined approach of the binary PSO with the K-nearest neighbor algorithm for the selection of features using logistic map. The suggested algorithm was called

the Chaotic Binary PSO (CBPSO). Chaos is a dynamic deterministic system that is sensitive to the assigned initial values. The suggested inertia weight equation that guarantees the global optimum results is [38]:

$$w(t+1) = 4.0*w(t)*(1-w(t)) \quad (2.27)$$

Where $w(t) \in (0,1)$

The experimental results in [38] illustrated that the new suggested inertia weight saves processing time compared to other methods in the literature. The results also revealed that the CBPSO with chaotic sequences achieves higher classification accuracy and reduces the number of features.

2.2.6 PSO Network Applications

PSO is applied in almost all disciplines of engineering. In this section we discuss the usage of PSO in computer network related applications. PSO was used in [48] to solve the problem of selecting the neighboring peers in a P2P network. In this kind of problems, performance and search efficiency are highly influenced by the topology of the network. Many P2P systems were built not only to share multimedia files between uses, but also for the public welfare like supplying a power for the process of fighting cancer. Any P2P system consists of peers and some connection between these peers. The key point for having efficient and high performance system is to define how these peers are connected.

Finding such a definition for these connections is challenging because of the dynamic membership of the peers in the network. Hence, a continuing reorganization

of the network topology is required all the time[48]. An efficient strategy for neighbor selection was proposed in [49]and was based on the Genetic Algorithm approach.

Another application of PSO in networks was performed in [50]. The goal was to study the usage of PSO in the design of a network infrastructure including decisions concerning the locations and sizes of the links. The optimization aimed to minimize the average packet delay in the network and the network layout cost [50].

The network design problem is considered NP-Hard so it was targeted using meta-heuristic techniques such as tabu search, simulated annealing and evolutionary computing [51, 52]. The solution to this problem is usually not optimal since it involves the optimization of several contradicting objectives such as network deployment, average delay or throughput subject to constraints like bandwidth and reliability [50].

A solution to the Shortest Path Problem (SPP) in networks was introduced in [53]. The shortest path calculation is one of the main problems in graph theory and finding a polynomial solution for such problem is known to be impossible. Finding a feasible solution for this problem is considered the basis for many applications like routing in communications networks, sequence alignment in molecular biology, robot motion planning, and many other applications [53].

The SPP was targeted by many approaches like Artificial Neural Networks (ANN) [54], Tabu Search [55] and Genetic Algorithms[13]. ANN weren't used on large scale for this kind of problem because the hardware complexity increases with the size of the network and ANNs do not provide sub-optimal path like the meta-

heuristic approaches. This was the reason for moving toward the evolutionary approaches like GA and TS. These two approaches gave better results than the ANN but when compared to PSO, the later outperformed them in terms of success rate, computation complexity and solution quality [53].

Along with PSO which was used in [53] to solve the SPP, additional noising mechanisms [56] were used to improve the local search quality around any local by using a diversification mechanism to discover near optimal points. The main issue in using PSO in SPP is the way a particle is encoded. A representation scheme called *cost-priority-based* was used in [54] to encode the particle based on node priorities. The goal was to compare the performance of PSO with two variants of GA based approaches. The results illustrated that PSO outperforms the GA for all configurations of the network in terms of speed and solution quality [53].

A modification on PSO called Trained PSO (TPSO) was presented in [57]. This approach distributed particles to reduce computational overhead and traffic in the optimization process and was applied in an Ad-Hoc network. The optimization goal was to find the node with the highest processing load in the network. To reduce the overhead of particles moving across the Ad-Hoc network, the original PSO was improved by changing the parameters of PSO (w and P) according to the system requirements using a training system [57]. The simulation results in [57] illustrated that the convergence time of TPSO is about to be constant.

PSO was used in [58] to design an algorithm for power consumption minimization in Ad-Hoc networks. In this type of networks, the power supply for the

mobile node is limited by the battery capacity. Hence it is necessary to develop a power minimization algorithm for enhancing the throughput to determine power level assignment and neighbor selection [58].

A solution to this problem is achieved by scheduling the node to enter into a sleep mode which will decrease the total power consumption. Another approach called power control [59] reduces the power consumption by adjusting the power level for each frame to be sent based on the perceived Ad-Hoc network status. The simulation results carried out in illustrated that the suggested power saving algorithm based on PS outperform all other existing algorithms.

2.3 Pedestrian Dead-Reckoning (PDR)

The fast advancement in manufacturing mobile devices that are equipped with built in positioning features stimulated the development of Location Based services (LBS) [1]. These services allowed various users to sopt their location within their environment. Many technologies are already available to develop these services like Wi-Fi [2, 3], UWB [5] and AoA based systems [4]. In our work we utilize low cost MEMS sensors along with TDoA based sensors to perform PDR.

The conventional approaches for PDR determine the position of a moving object by integrating the accelerometer and gyroscope signals [9, 10] . A heading sensor is used to determine the attitude of the object like a digital compass. Using signals from these technologies alone is susceptible to error due to inherent instrumental errors and environmental disturbances.

Experimental results in [60] illustrated that double integration of accelerometer signal produces fast growing position error overtime. To overcome this problem, many zero-velocity-update (ZUPT) based techniques were introduced in shoe-mounted systems in support of PDR. For example, both approaches in [9] and [10] proposed accelerometer calibration during the zero-phase velocity for each step.

A waist-worn IMU comprised of accelerometer, gyroscope and a magnetometer was proposed in [61] to perform PDR. A quaternion-based Extended Kalman Filter (EKF) was used to estimate the magnetic disturbances and correct them. The experimental results illustrated that a relative distance error of 3% to 8% can be achieved. However, such systems require sophisticated techniques for navigation. This is due to the difficulty to obtain zero-velocity points from the pelvis as in shoe-mounted DPR systems.

An alternative to using MEMS based accelerometers and gyroscopes is the system called monocular Simultaneous Localization and Mapping (SLAM) [62]. In SLAM, a local visual map of a region is captured by a mono-vision camera and is processed by applying image processing techniques [63]. Visual mono-SLAM provided precise estimation of the camera pose related to fixed features in the environment. The generated map is registered in the global virtual map if there exists an absolute positioning system (e.g., GPS). However, this makes SLAM work only in outdoor environments.

2.4 Data Fusion in Support of Indoor Localization

The majority of tracking applications is based on triangulation and literation techniques using light [64, 65], ultrasound [66, 67], or radio signals [2, 68-70]. Other techniques use inertial navigation to provide relative object location detection [71, 72]. Unfortunately, these techniques suffer from drifting and error accumulation resulting from noisy data integration over time, which requires continuous and periodic system calibration to reset the system state.

Available systems with different configurations and accuracies are currently being used worldwide.Microsoft Research's WaveLAN system [73] AT&T Cambridge Ultrasonic Bats [64], Active Badges [74], Radio tags, Computer vision systems [75], are examples of such systems. Data fusion for data observed from wireless LANs, Ultra Wide Band (UWB), Infrared (IR), Ultrasound, camera images and Inertial Measurement Units (IMUs) were fused by multiple data fusion techniques to provide more precise and robust localization systems [18-22].

Data fusion between IMU data and WLAN signal was proposed in [3]. Pedestrian tracking was achieved by the integration of sensor data from low-cost MicroElectroMechanical Systems (MEMS) accelerometer, WLAN and building map information. To avoid the error due to sensor noise, the zero-crossing algorithm was used to estimate the walking distance. The Sequential Importance Re-sampling Particle Filter (SIRPF) was used to perform data fusion. Simulation and real walking tests illustrated improved localization performance compared to Kalman Filtering in

terms of mean error and standard deviation. Using IMUs in localization requires periodic system calibration due to inherent drifting aspects of the accelerometers.

Sensor fusion between IMU and camera images using the Extended Kalman Filter (EKF) was proposed in [76, 77]. A wearable Augmented Reality (AR) system was mounted on a helmet. It is comprised of a real-time 3D visualization system comprised of a stereo see-through Head Mounted Display (HMD) and a real-time tracking system, composed of an Inertial Measurement Unit (IMU) and a camera. Sensor fusion used vision-based values in the correction step and inertial measurements in the prediction step.

A motion capture (MoCap) system fuses data obtained from gyroscopes, accelerometers and acoustic sensors by an EKF was presented in [78]. The ultrasonic system provides relative distances across sensors. However, this MoCap system does not obtain absolute positioning of users in the environment since it only provides relative distances. An improvement to this system was presented in [79] by introducing an UWB localization system.

Kalman filter is used in most of the data fusion techniques. Its main the linear model assumption can be hardly fulfilled in real systems. Unscented Kalman Filter (UKF) [80] and Extended Kalman Filter (EKF) [19, 20] were proposed to solve the non-linearity problem by linearizing all the non-linear models. These filters are reliable only for almost-linear systems. Distributed information is impossible to be included for tracking by UKF or EKF. An alternative to Kalman filter, Particle Filter (PF) is getting more attention recently [81].

In this work, we present a novel approach for performing data fusion between multiple sensor technologies including accelerometers, gyroscopes, Wi-Fi, and Time Difference of Arrival (TDoA) sensors to achieve precise localization in indoor environments. The proposed Inertial Navigation System (INS) is comprised of tri-axial accelerometer, gyroscope and an MIT Cricket system for gyroscope drift correction. The INS system provides the distance that the moving object travelled and the direction of that distance.

2.5 Data Fusion in Support of Outdoor Localization

Many surveillance and tracking applications rely completely on the knowledge of the position of objects in the environment. These applications provide an innovative automation layer called automatic object location detection. Real world applications using this automation process are numerous. Location detection of personnel or medical equipment inside a hospital, inventory tracking in warehouses, moving assets inside a store, location detection of firemen inside a burning building, intelligent guidance, location-aware multimedia services are examples of these applications [6-8].

The Extended Kalman Filter (EKF) was used in [79] to perform fusion between inertial motion MoCap system and an Ultra Wide Band (UWB) localization system. The MoCap registered the movements of the operator's limbs accurately but gave inaccurate results for the global object position. To obtain more precise position measurements, an UWB localization system is used along with MoCap [79].

Outdoor localization of robots was carried out in [82]. An enhanced 3-D navigation system was described using KF. The system integrates odometry obtained from Reduced Inertial Sensor System (RISS) comprised of three accelerometers and one Gyroscope, wheel encoders and GPS. The experimental results illustrated that during GPS outages, KF with velocity update obtained from the forward speed of wheel encoders is an acceptable method localization errors reduction. The results illustrated that the KF with RISS and velocity updates in GPS outages achieves huge enhancement over KF with full IMU without any updates.

Particle filter (PF) was used in [83] to combine sensor data from UWB with GPS to achieve localization both indoor and outdoor. The test was performed on a robot mounted with an UWB antenna and a GPS receiver. The location error was small as the robots moves inside a building were only UWB beacons were available. This error accumulates up to few meters as the robot moves away from the UWB beacons and GPS coverage only is available.

Finally, Covariance Intersection (CI) which is a generalization of the Kalman Filter is used in [84] to perform the fusion of data coming from a wireless pyroelectric sensory system embedded with traditional fire detector and wireless LAN signals. The maximum mean error of Received Signal Strength (RSS) was near 19% and the maximum mean distance error from the pyroelectric measurements was near 14%. Using the CI the distance mean error achieved was below 9%.

CHAPTER 3

PEDESTRIAN DEAD-RECKONING

3.1 Introduction

Precise and robust localization of humans is one of the most demanding challenges that researchers face these days. GPS provides a perfect solution in open areas with acceptable localization accuracy. However, in indoor environments where GPS service is almost denied, dead reckoning algorithms that utilize low cost sensors provide alternative solutions for such environments. In this work, we use low cost MEMS (accelerometer and gyroscope) sensors to achieve Pedestrian Dead Reckoning (PDR) based on the zero-velocity update (ZUPT) strategy. We also propose a novel drift correction mechanism using a TDoA technology; the MIT Cricket.

3.2 Step Detection

A Triaxial accelerometer mounted on the foot of the pedestrian can be used to detect the step length. Fig. 3.1 shows the total acceleration signal from a foot-mounted accelerometer. When the foot is in stable contact with the ground, the accelerometer measures around 1g ($9.8 m/s^2$). This means that during this time, the only force acting on the accelerometer is gravity force. This observation is used to detect the pedestrian step.

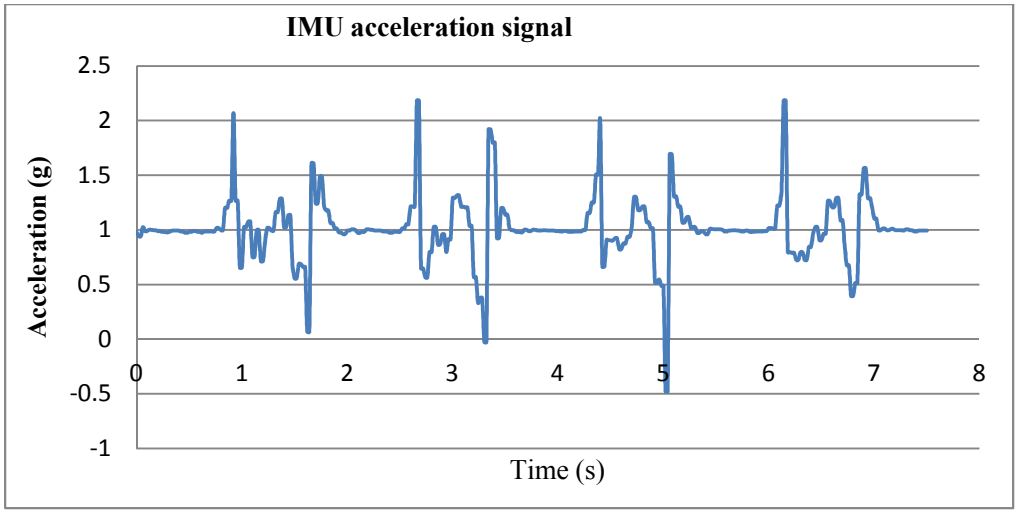

Figure 3.1:IMU acceleration signal.

3.2.1 Flip-Flop Filter

This filter consists of two Exponentially Weighted Moving Average (EWMA) filters. The first one is agile with a gain of 0.1 and the second one is stable with a gain of 0.9. The general equation of such filters is of the form:

$$E_t = \alpha E_{t-1} + (1-\alpha) O_t \quad (3.1)$$

Where E_t is the smoothed estimation at time t, E_{t-1} is the flip-flop filter output at time $t-1$, O_t is the observed signal value at time t and α is the filter gain which determines the reactivity to the input signal. If the gain is large, the EWMA filter will be less sensitive to the input signal and the old input values will dominate in calculating the current estimate. If the gain is small, the filter will be sensitive to the current observation and history will have less effect on the filter estimate.

3.2.2 Symbolic Representation of Accelerometer Signal

In our work, the output of the Flip-Flop filter (E_t) is represented as an infinite string of symbols. "*R*" means that the estimated filter output represents a "*Resting Foot*" with the corresponding E_t is between $0.9g$ and $1.1g$. All other values for E_t are represented by the symbol "*M*" meaning that he foot is in Moving state. For example, the signal in Fig. 3.1 has the symbolic representation shown in Fig. 3.2:

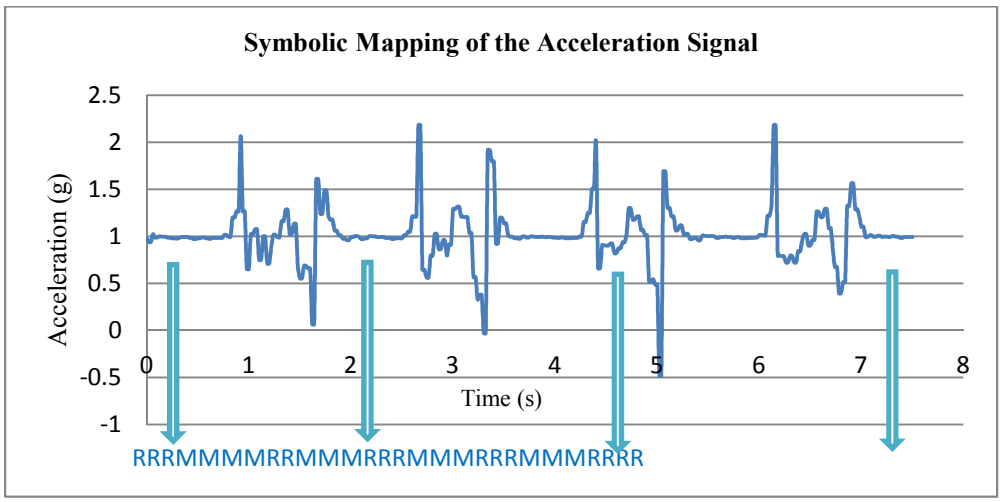

Figure 3.2: Mapping of the acceleration signal to symbolic string.

The sampling rate of the accelerometer is $50Hz$. This means that every 20 milliseconds a sample is acquired from the accelerometer. The step time can be determined by multiplying the number of *M* symbols in the signal string (before an *R* is observed) by 20 milliseconds. Fig. 3.3 shows our signal processing approach and the flow char of the flip-flop filter [85].

3.3 Pedestrian Dead-Reckoning (PDR)

Dead-Reckoning is the process of tracking an object's location based on a previously calculated position and updating this position based on speeds estimated over time. Our Inertial Navigation System (INS) is comprised of tri-axial accelerometer, gyroscope and an MIT Cricket system for gyroscope drift correction. In this section, we present our approach for PDR by starting from the theoretical background and then our approach for dead-reckoning.

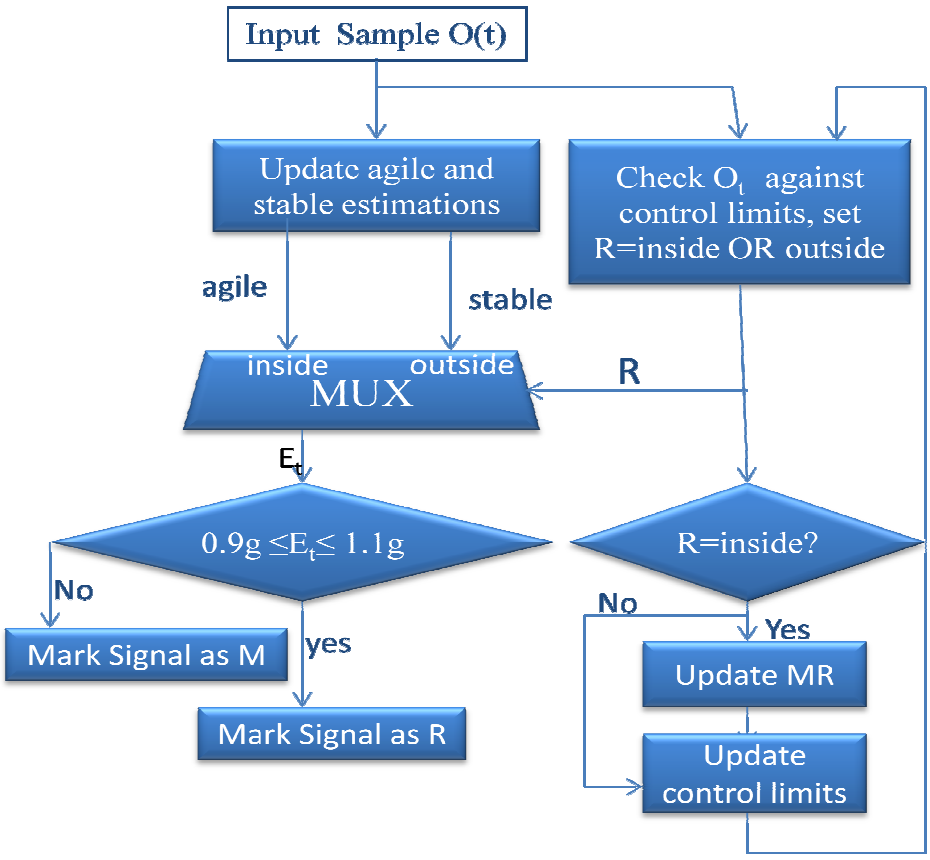

Figure 3.3: Accelerometer signal processing.

3.3.1 Acceleration, Velocity and Distance Relations

The estimated acceleration samples are double integrated over time to get the step size in meters. Only samples of type *M* are integrated since they symbolize real foot motion while samples of type *R* represent noise while the foot is resting and should be neglected in the integration process.

The attitude of the IMU relative to the global frame of reference can be tracked by integrating the angular velocity signal of the body frame $\boldsymbol{\omega_b}(t)=(\omega_{bx}(t), \omega_{by}(t), \boldsymbol{\omega_{bz}}(t))$ acquired from the gyroscope. Quaternion, Euler's Angles and Direction Cosines can be used to represent the orientation of the IMU. In this approach we use the direction cosines for orientation representation which is specified by a *3-by-3* rotation matrix *C*. Each column of *C* represents a unit vector in the body axis projected along the global frame.

$$C = \begin{Bmatrix} c\theta c\psi & c\theta s\psi & -s\theta \\ s\varphi s\theta c\psi - c\varphi s\psi & s\varphi s\theta s\psi + c\varphi c\psi & s\varphi c\theta \\ c\varphi s\theta c\psi + s\varphi s\psi & c\varphi s\theta s\psi - s\varphi s\psi & c\varphi c\theta \end{Bmatrix} \quad (3.2)$$

Where s* and c* are the sine and cosine respectively.

The element in the j^{th} column and i^{th} row is the cosine of the angle between the j^{th}-axis of the body frame and the i^{th}-axis of the reference frame [86] and (φ, θ, ψ) are the Euler angles (roll, pitch, yaw), respectively. A vector quantity $\mathbf{V_b}$ in the body frame is equivalent to:

$$\mathbf{V_g} = C\mathbf{V_b} \quad (3.3)$$

Where $\mathbf{V_g}$ is represented in the global frame of reference. To track the attitude of the IMU, the *C* matrix must be updated through time. The rate of change in *C* at time *t* is given by:

$$\dot{C(t)} = \lim_{\Delta t \to 0} \frac{C(t+\Delta t) - C(t)}{\Delta t} \tag{3.4}$$

From equation 3.4, the rotation matrix $C(t+\Delta t)$ can be written as:

$$C(t+\Delta t) = C(t) \left(\boldsymbol{I} + \frac{sin\sigma}{\sigma} \mathbf{B} + \frac{1-cos\sigma}{\sigma^2} B^2 \right) \tag{3.5}$$

Where:

$$\mathbf{B} = \begin{Bmatrix} 0 & -\omega_{bz}\Delta t & \omega_{by}\Delta t \\ \omega_{bz}\Delta t & 0 & -\omega_{bx}\Delta t \\ -\omega_{by}\Delta t & \omega_{bx}\Delta t & 0 \end{Bmatrix} \tag{3.6}$$

$$\boldsymbol{\omega_b} = (\omega_{bx}(t), \omega_{by}(t), \omega_{bz}(t))^T \tag{3.7}$$

$$\sigma = | \Delta t \omega_{\mathbf{b}} | \tag{3.8}$$

A complete derivation of equation 3.5 can be found in [87].

To track the IMU position, acceleration data is integrated to get the velocity vector which is then integrated to get the position of the IMU. Before integrating the 3-dimentional accelerometer vector $\mathbf{a_b}(t) = (a_{bx}(t), a_{by}(t), a_{bz}(t))$, it must be projected on the global frame of reference using the rotation matrix C [87]:

$$\mathbf{a_g}(t) = C(t)\, \mathbf{a_b}(t) \tag{3.9}$$

Fig. 3.4 shows how the gravity is compensated from the accelerometer signal using the gyroscope and the rotation matrix

.

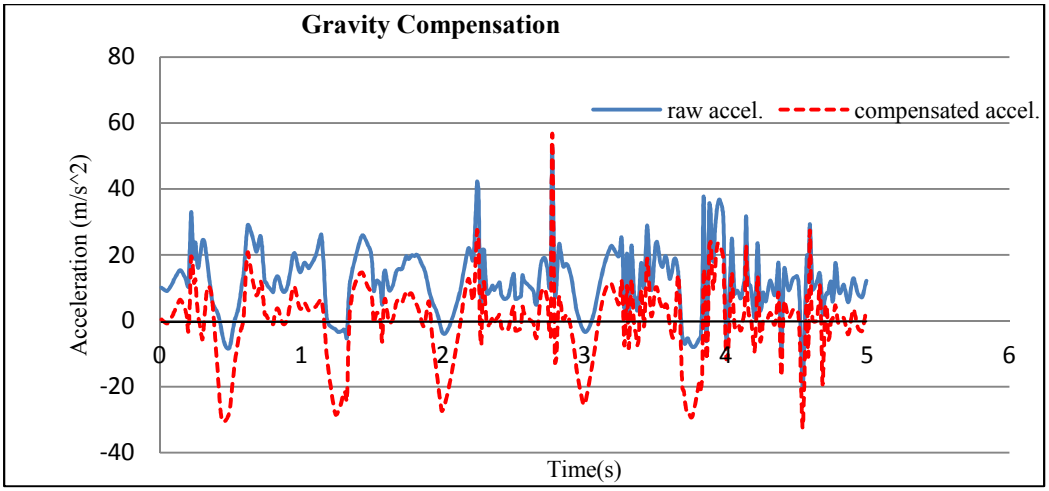

Figure 3.4: Gravity compensation from the accelerometer signal.

After projecting the acceleration signal from the body frame of reference to the global frame of reference, it is possible to compensate for the acceleration due to gravity by subtracting the gravity acceleration vector **G** = (0,0,g) from $\boldsymbol{a_g(t)}$. The resulting acceleration is then integrated to get the IMU velocity:

$$\mathbf{v_g}(t) = \mathbf{v_{g\_initital}} + \int_0^t \boldsymbol{a_g}(t) - \boldsymbol{G}\, dt \tag{3.10}$$

Where $\boldsymbol{v_{g\_initital}}$ is the initial velocity of the IMU. The new position of the IMU is computed by integrating the velocity vector $\boldsymbol{v_g(t)}$:

$$\mathbf{p_g}(t) = \mathbf{p_{g\_initial}} + \int_0^t \boldsymbol{v_g}(t)\, dt \tag{3.11}$$

Where $\boldsymbol{p_{g\_initital}}$ is the initial position of the IMU.

3.3.2 Accelerometer and Gyroscope Error Characteristics

Accelerometer and gyroscope signals suffer from various types of noise that should be filtered out before being utilized in position tracking. In [87], the authors summarize the various types of noise that affect both gyroscopes and accelerometers:

1) Constant bias: Despite of being stationary, gyroscope and accelerometer output a signal that has an average called the bias (b). If this bias is integrated over time, it will cause an angular error of $b*t$ and a position error of $\frac{bt^2}{2}$. This bias must be subtracted from all subsequent samples of the device.

2) Thermo-Mechanical White Noise: this noise fluctuates by a rate greater than sensor's sampling rate which causes a white noise affecting the samples obtained from the sensor. This noise causes an angle random walk with standard deviation of $\sigma\sqrt{t.\Delta t}$ and a position error with a standard deviation of $\sigma.t^{3/2}.\sqrt{\Delta t/3}$.

3) Other types of noise like bias instability, calibration, and temperature effects. These types minimally affect the position and angular estimation compared with the first two types of noise.

3.3.3 Time Difference of Arrival (TDoA) Background

To overcome the accelerometer and gyroscope error characteristics described above, the distance obtained from the accelerometer is corrected by using a Time Difference of Arrival (TDoA) technology like the MIT Cricket [88].

In its basic design form, Cricket is comprised of *beacons* attached to the ceiling of a building, and *listeners* attached to the tracked object. The beacons periodically broadcast their location information in an RF message and simultaneously transmit an ultrasonic (US) pulse. The listeners receive these signals and measure the distances from close by beacons based on the difference in the time of arrival between the RF and US signals and accordingly estimate their relative location. This architecture allows for scalable system and preserves user privacy [88].

In Cricket, beacon-to-listener distance computation is achieved using TDoA between the RF and US signals. Since the velocity of the RF signal (speed of light) is much higher than that of the US (speed of sound), the listener measures the time interval ΔT between the start of the RF signal and the arrival of the US signal. The distance between the beacon and the listener is then computed by:

$$\Delta T = \frac{D}{V_{US}} - \frac{D}{V_{RF}} \tag{3.12}$$

At normal room temperature, the speed of sound, $V_{US} = 344$ *m/s* and the speed of light, $V_{RF} = 3*10^8$ *m/s*. Since $V_{RF} \gg V_{US}$, equation 3.12 becomes:

$$D \approx \Delta T * V_{RF} \tag{3.13}$$

The major disadvantage of using Cricket in localization is that is requires a Line of Sight (LoS) between the beacon and the listener. We overcome this drawback in the Cricket by mounting the beacon and the listener on the feet of the pedestrian as shown in Fig. 3.5. This installation of the beacon and listener solves the LoS problem. It also provides more distance accuracy since the Cricket system has an accuracy of 1

cm if the distance between the beacon and the listener is less than 3.5*m* [89]. This is apparently always the case in our Cricket system installation as presented in Fig 3. 5.

Since the Cricket subsystem provides absolute distance information, this information will be used to correct the noisy distances provided by the accelerometer due to drift.

Figure 3.5: Installation of Cricket beacon and listener in support of PDR.

3.3.4 Our Approach for PDR

Accelerometer bias can be removed by calibrating the accelerometer while the IMU is momentarily stationary and averaging the output signal to get the bias. For gyroscope, the bias changes with time and the gyroscope needs to be calibrated whenever possible to remove the bias and other sources of noise. In our approach, we calibrate the gyroscope when the foot is in contact with the ground (i.e. while the accelerometer symbolic signal output is of type *R*).

The gyroscope bias $\mathbf{b_g}=(b_{gx},b_{gy},b_{gz})$ is estimated each time the foot is in contact with the ground by comparing the estimated distance resulting from the acceleration signal integration (E_{da})with the observed accurate distance from the Cricket motes (d_c).

Our approach for PDR is summarized by the following navigation steps:

1) Initialize the rotation matrix C(t) = I

i. The initial position of the IMU is $\mathbf{P}(0) = (x_0, y_0, z_0)$

ii. Start = the time at which first "*M*" is received

iii. End = the time at which first "*R*" is received

2) If the accelerometer signal stream is of type *R*, calibrate the gyroscope:

End = current time

ii. Use Particle Swarm Optimization (PSO) to solve the unconstrained optimization problem:

Minimize

$| [d_c^2 - [[P_x(End) - P_x(Start)]^2 + [P_y(End) - P_y(Start)]^2 + [P_z(End) - P_z(Start)]^2] |$

Where ***P(End)*** is the position of the IMU at time t = End and ***P(Start)*** is the position of the IMU at time t = Start. The output of this minimization problem is the gyroscope bias vector $\boldsymbol{b_g}$ that will be removed from all gyroscope samples.

3) If the accelerometer signal stream is of type *M*:

i. Start = Current time

ii. Update *C(t+Δt)* in equation 3.5 using the gyroscope signal $\boldsymbol{\omega_b}(t)=(\omega_{bx}(t), \omega_{by}(t), \omega_{bz}(t))$ and the gyro bias $\mathbf{b_g}=(b_{gx},b_{gy},b_{gz})$ found from the previous step by substituting $\mathbf{W}_b(t) = \boldsymbol{\omega_b}(t) - \mathbf{b_g}$ in equations 3.6-3.8.

iii. Find $\mathbf{a_g}(t)$, $\mathbf{v_g}(t)$, $\mathbf{p_g}(t)$ using equations 3.9-3.11

iv. From equation 3.11, the new position of the IMU is:

$$\mathbf{P}(t+\Delta t) = \mathbf{P}(t) + \frac{\Delta t^2 . \mathbf{a_g}(t)}{2}$$

3.4 Simulation Setup

In our work, we used IMUSim [90] as our simulation environment. IMUSim simulates sensor readings based on continuous trajectory models, and shows how suitable models can be generated from existing motion captures or other sampled data [18].

The input to IMUSim is a motion capture file [91] that describes the motion trajectories of a moving subject. The output can be chosen to be the accelerometer, gyroscope or magnetometer signals from the IMUs attached to the body parts of the moving subject. In our work, we utilized signals from an IMU attached to the right foot of the pedestrian. IMUSim has several types of IMUs . We used an ideal IMU and added white noise and constant gyroscope drift during each walking step of the moving pedestrian. We used MIT Cricket to correct this drift as described in the previous sections. The accuracy of the MIT Cricket is assumed to be 1 cm [88].

3.5 Results and Discussion

In this section, we illustrate how our novel drift correction approach works based on TDoA technology. Fig. 3.6 shows the position of an IMU attached to the right foot of a pedestrian starting at position (0, 0, 0). The figure represents the ideal position in x, y and z dimensions. This acts as a baseline of comparison with our proposed approach. This figure shows two steps of the pedestrian as apparent in the x-dimension from the figure.

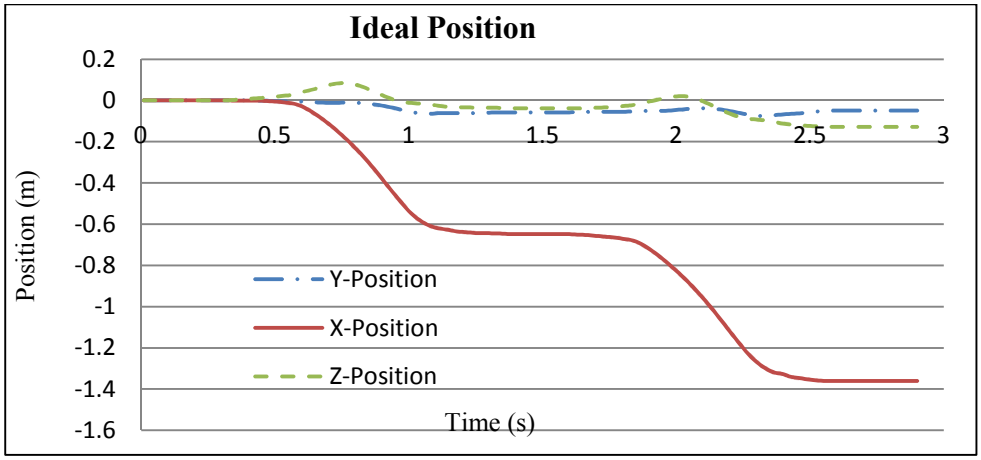

Figure 3.6: The ideal position obtained from double integrating acceleration signal from IMUSim.

A drift [d_x, d_y, d_z] is added to the filtered gyroscope signal and the resulting signal is used in equation 3.9 to rotate the accelerometer signal vector. Fig. 3.7 shows the resulting position after integrating the accelerometer signal using the drifty gyroscope signal.

Our TDoA drift correction approach was able to remove high percentage of the drift in the gyroscope signal. Fig. 3.8 shows the IMU position after drift removal which is very close to the ideal position shown in Fig. 3.6. Fig. 3.9 shows the x-axis position of the IMU for the ideal, drifty and corrected gyroscope signals. Fig. 3.10 shows the amount of drift added on the x, y, z axes and the remaining drift in the signal after correction. About 85% of the added drift was removed using our drift correction approach.

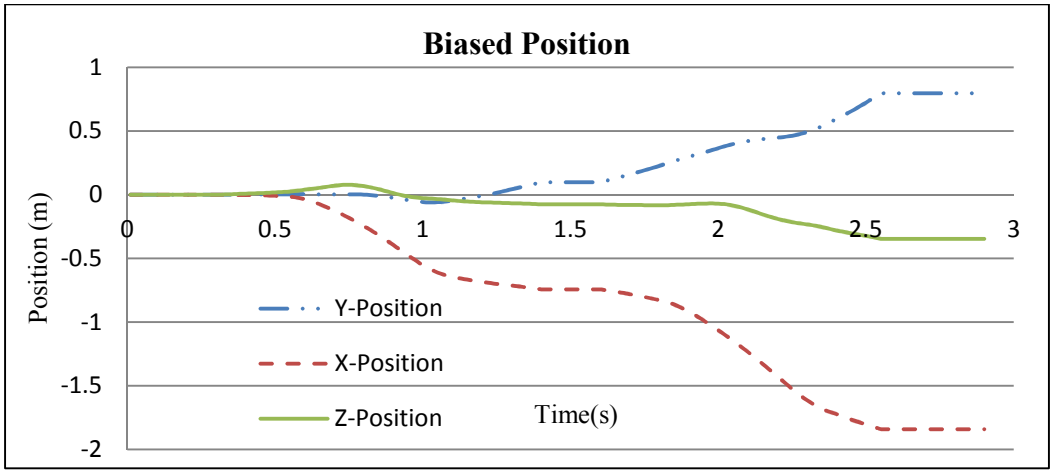

Figure 3.7: The position obtained from double integrating accelerometer and drifty gyroscope signals.

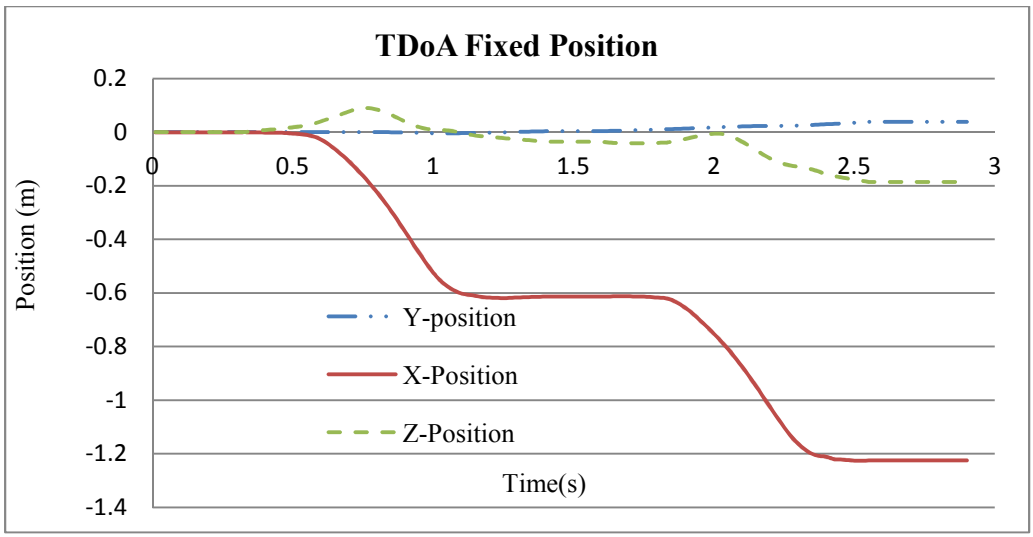

Figure 3.8: The position obtained from double integrating accelerometer and corrected gyroscope signals.

Another experiment that we conducted is for a pedestrian moving forward and going back to the initial position is shown in Fig. 3.11. The figure shows the position on the x-axis for ideal, drifty and corrected gyroscope signals. The amount of the drift added is shown in Fig. 3.12. The figure shows both the added and the remaining drift

in the gyroscope signal. On average about 95% of the added bias was removed using our drift correction approach.

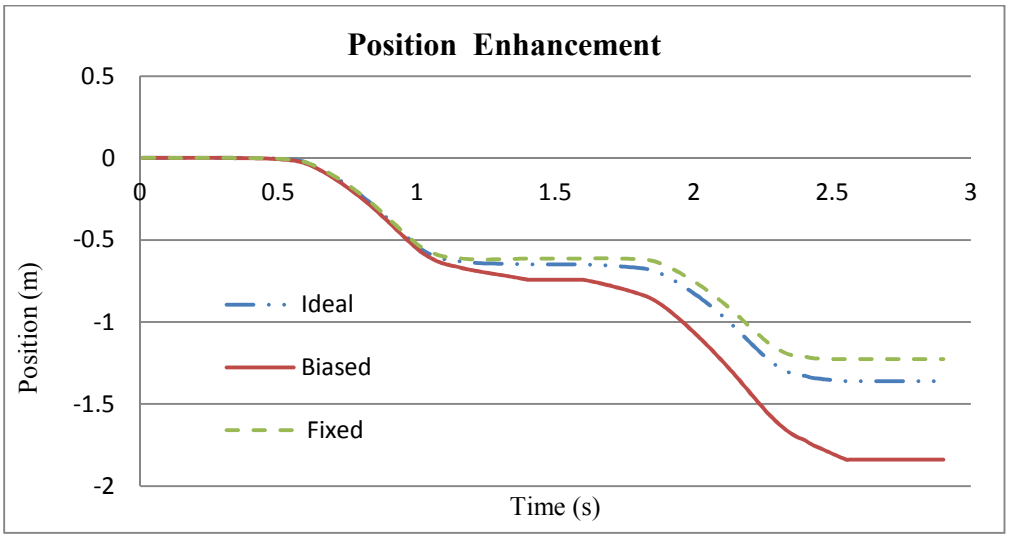

Figure 3.9: The x- axis position obtained from ideal, drifty and corrected gyroscope signals.

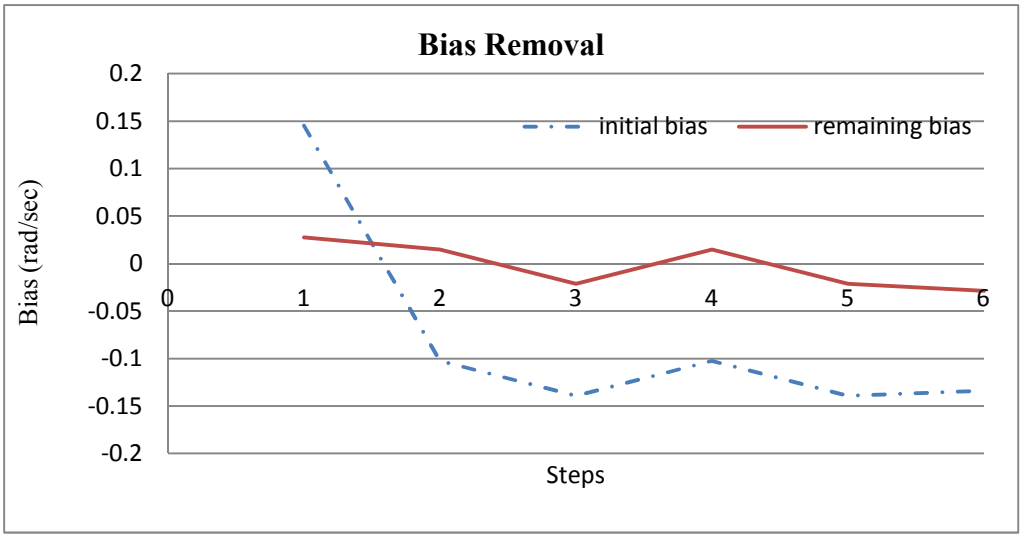

Figure 3.10: The added and remaining drift after correction of the gyroscope signal.

Figure 3.11: The x-axis position obtained from ideal, drifty and corrected gyroscope signals.

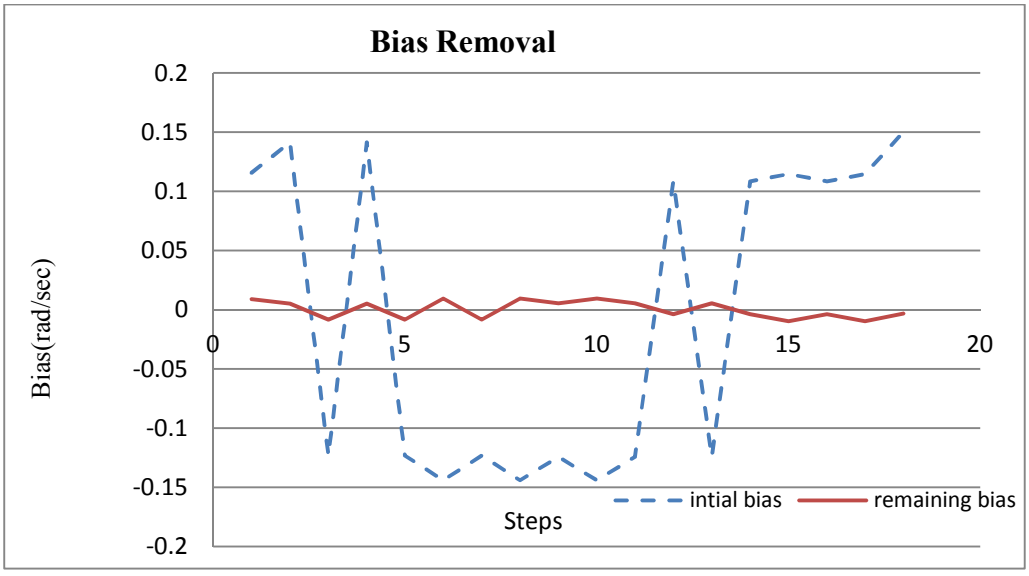

Figure 3.12: The added and remaining drift after correction of the gyroscope signal.

CHAPTER 4

DATA FUSION IN SUPPORT OF INDOOR LOCALIZATION

4.1 Introduction

Indoor localization with a significant degree of precision is extremely challenging. In this chapter, we present a precise indoor localization approach based on novel particle filter and dynamic exclusion techniques. The approach is compared with the Euclidian Distance probabilistic methods used for localization. The novelty of the proposed approach stems from its ability to fuse data collected from different sensor technologies to converge to more accurate distance estimation. Furthermore, the proposed approach is a pattern-based one that relies on empirical training data as opposed to closed-form mathematical models.

4.2 Background

Location fingerprints is an offline collection of features collected from the localization network. Location fingerprinting refers to the process of matching the fingerprint of a signal characteristic that is location dependent. This can be done in two stages [92]:

1) Offline: in this stage, a site survey is performed to collect a known location coordinates/tags with their respective signal strengths from nearby LPAM radio base stations. These measurements are stored in a location fingerprint database to be used in the online stage.

2) Online: during this stage, the localization technique uses the offline data stored in the location fingerprint database along with the online Received Signal Strength Indicator (RSSI) measurements to estimate the object position. The main challenge for this technique is that the signal strength is affected by diffraction, reflection and scattering in indoor environments [93].

The location fingerprints database contains *N* fingerprints with each location fingerprint representing the RSSI measurements observed from *M* nearby access points. The database structure is shown in Table 4.1. The table has two columns, the RSSI measurements vector and the corresponding location ID at which these RSSI measurements are observed. A location fingerprint FP_i at location i can be represented by: $FP_i = \{RSSI_1^{FP_i}, RSSI_2^{FP_i}, RSSI_3^{FP_i}, \dots, RSSI_M^{FP_i}\}$.

Table 4-1: The fingerprint database structure.

Fingerprint	Location
$FP_1 = \{RSSI_1^{FP_1}, RSSI_2^{FP_1}, RSSI_3^{FP_1}, \dots, RSSI_M^{FP_1}\}$	$L_1^{FP} = (X_1, Y_1)$
$FP_2 = \{RSSI_1^{FP_2}, RSSI_2^{FP_2}, RSSI_3^{FP_2}, \dots, RSSI_M^{FP_2}\}$	$L_2^{FP} = (X_2, Y_2)$
…	…
$FP_N = \{RSSI_1^{FP_N}, RSSI_2^{FP_N}, RSSI_3^{FP_N}, \dots, RSSI_M^{FP_N}\}$	$L_N^{FP} = (X_N, Y_N)$

By using the Euclidian Distance (EUC) technique, the object localization is treated as a classification problem. Given *N* location fingerprints and an online RSSI vector (*S*) for the moving target *T*, where $S = \{RSSI_1^T, RSSI_2^T, RSSI_3^T, \dots, RSSI_M^T\}$, the Euclidian distance between *S* and FP_i is given by [92]:

$$D_{FP_i}^{T} = \sqrt{\sum_{j=1}^{M} \left(RSSI_j^{FP_i} - RSSI_j^{T} \right)^2} \quad (4.1)$$

Where M is the number of access points. The probability that the object is near fingerprint FP_i given the measured RSSI vector S is given by:

$$P(FP_i|S) = \frac{1}{D_{FP_i}^{T}} \quad (4.2)$$

Then the location decision rule becomes [93]:

Choose if $P(FP_i|S) > P(FP_j|S)$, *where is the location of* FP_i, *for* $\forall$ $i, j = 1, 2, 3, \ldots, N, j \neq i$.

Assuming that the probability that the tracked object is at location L_i^{FP} is given by $P(L_i^{FP})$ and assuming that $P(L_i^{FP}) = P(L_j^{FP})$ for i, j = 1, 2, 3, . . . , N, $j \neq i$. Using Bayes' theorem we have the following formula that is based on the likelihood that $P(S|FP_i)$ is the probability that the signal strength is S provided that the tracked object is at location L_i^{FP}:

Choose L_i^{FP} *if* $P(S|FP_i) > P(S|FP_j)$, *for* $\forall$ $i, j = 1, 2, 3, \ldots, N; j \neq i$.

Based on the fact that the measuring units in the localization area are independent, the overall likelihood of the target location can be calculated from the collected signal strengths during the online phase. Hence, the tracked object location can be estimated using the previous rule. However, this can be applied only for discrete location candidates. In reality, the tracked object can be at any position in the network and not only at discrete locations. Thus, the location L_T of the tracked object

can be interpolated by a weighted average of the locations of all location fingerprints in the database using the equation:

$L_T(x,y) = \sum_{i=1}^{N} P(FP_i \mid S) L_i^{FP}(x,y)$ (4.3)

Where *x,y* are the coordinates of FP_i .

The following provides the Pseudo code for the EUC approach for indoor localization:

N : Number of location fingerprints
M : Number of access points

D: distance between a location fingerprint and the tracked object.

FPw : Weight assigned to a given fingerprint

W : Summation of all location fingerprint weights

F = Multiplication factor

S : The current RSSI vector for the tracked object

L_T*: Estimated location of the tracked object expressed by* L_{Tx}, L_{Ty}

L^{FP_i} *: Location of fingerprint i expressed by* $L_x^{FP_i}$ *and* $L_y^{FP_i}$

$L_{Tx}=0$ $L_{Ty}=0$ $W=0$

For i = 1 To N
    Compute D_i *using equation (4.1)*
    $FPw_i = \frac{1}{D_i}$
    $W=W+FPw_i$
End For
F = 1/ W
For i = 1 To N
$P(FP_i \mid S) = F * FPw_i$
$L_{Tx} = L_{Tx} + P(FP_i \mid S) * L_x^{FP_i}$
$L_{Ty} = L_{Ty} + P(FP_i \mid S) * L_y^{FP_i}$

End For

A static location fingerprint exclusion technique was demonstrated in RADAR [2] using the *K*-Nearest Neighbors (KNN) algorithm. The idea behind this approach is that only a specific set of location fingerprints are used in the location estimation. Only *K* nearby fingerprints contribute to location estimation based on their RSSI distance from the tracked object.

The results in [2] shown that a small value of *K* induces higher location error. The error becomes smaller as more nearby location fingerprints are added to the set of the contributing location fingerprints. Adding further location fingerprints increases the location error again. The drawback of this approach lies in the difficulty to determine the value of *K* that offers minimum location error for a certain localization environment.

In this research, the performance of the proposed localization technique using Budgeted Dynamic Exclusion (BDE) is compared with existing approaches using the Euclidian Distance methods. Further localization accuracy is achieved by fusing INS data with RSSI measurements using the particle filter.

4.3 Proposed Approach

In the EUC approach, the weight of a given location fingerprint in estimating the tracked object location completely depends on the RSSI differences between the location fingerprint and the tracked object. All access points are involved in computing the RSSI distance between a specific location fingerprint and the RSSI vector of the tracked object according to equation 4.1.

The proposed approach aims to identify the outlier access points and exclude them from the position estimation process to gain more localization accuracy. Further localization accuracy is gained by fusing the RSSI data with data from the INS using the Particle Filter. The following sections discuss the proposed indoor localization in detail.

4.3.1 Budgeted Dynamic Exclusion (BDE) Heuristic in Support of Indoor Localization

In the BDE heuristic, we aim to exclude the RSSI values from the outlier access points while computing the distance between an RSSI vector retrieved from the location fingerprint database and the RSSI vector measured by the tracked object. This is done by comparing the RSSI value from the j^{th} access point in the i^{th} location fingerprint with the corresponding RSSI value ($RSSI_j^{FP_i}$) from the jth access point in the RSSI vector of the tracked object. If the difference between these two RSSI values is less than certain budget, the corresponding access point (the j^{th} access point) will be excluded from the RSSI distance measurement process (equation 4.1).

If the tracked object is close to a certain location fingerprint, the corresponding RSSI values on both of them are most likely less than the budget. Hence, more access points will not be involved in equation 4.1 and consequently that location fingerprint will have more weight in the location estimation process according to equations 4.1 and 4.2. This means that only RSSI values from non-

outlier access points which conform to the budget constraints are involved in the location estimation process.

BDE starts with small value for the budget, (i.e 5%) and computes the Euclidian Distance from all access points in a location fingerprint. A new distance is calculated based on the criteria described above. If the change in the new calculated distance compared with the previously computed Euclidian Distance is also greater than the budget, we increase the budget to achieve more access point exclusion. Otherwise, the system returns the last distance as the distance between the current fingerprint's RSSI vector and the tracked object's RSSI vector. This process is further described in the following steps:

The following provides the Pseudo code for the BDE approach for indoor localization:

N : Number of fingerprints

M : Number of access points

D: distance between a fingerprint and the tracked object.

FPw : Weight assigned to a given fingerprint

W : Summation of all fingerprint weights

F = Multiplication factor

S : The current RSSI vector for the tracked object

L_T: Estimated location of the tracked object expressed by L_{Tx}, L_{Ty}

L^{FP_i} : Location of fingerprint i expressed by $L_x^{FP_i}$ and $L_y^{FP_i}$

$L_{Tx}=0$ $L_{Ty}=0$ $W=0$

For i = 1 To N
Compute D_i from equation (4.1)
For budget = 5% To 25% StepBy 5%
For j=1 To M
If ($RSSI_j^{FP_i}$ - $RSSI_j^T$)/ $RSSI_j^{FP_i}$ > budget
$D = D + (RSSI_j^{FP_i} - RSSI_j^T)^2$
End if
End For
If $(D-D_i)/D_i$ > budget
Budget = budget+5%
Else
Di = D
D=0
Break
End if
End For
$FPw_i = \frac{1}{D_i}$
$W=W+FPw_i$
End For
F = 1/ W
For i = 1 To N
$P(FPi\ |S) = F * FPw_i$
$L_{Tx}= L_{Tx} + P(FPi\ |S) * L_x^{FP_i}$
$L_{Ty}= L_{Ty} + P(FPi\ |S) * L_y^{FP_i}$
End For

4.3.2 Particle Filter Application in Support of Indoor Localization

In addition to the Sequential Monte Carlo (SMC) nature of estimation, the PF allows for flexible design and parallel implementation. The main advantage of the PF is its ability to combine measures from multiple sensors considering their probabilistic behavior. The key idea behind the PF is that the posterior Probability Density Function (PDF) of state *x(k)* is directly estimated conditioned on the set of measurements *Z(k)* according to the equations [84]:

$$P[x(k)|Z(k)] \approx \sum_{i=1}^{N} w_i(k)\delta[x(k) - x_i(k)] \quad (4.4)$$

$$\sum_{i=1}^{N} w_i(k) = 1 \quad (4.5)$$

Where N is the number of particles, $w_i(k)$ is the weight of the i^{th} particle $x_i(k)$ and $\delta(.)$ is the Dirac distribution.

The biggest advantage of using the PF instead of the EKF is its ability to solve non-Gaussian and non-linear estimation problems. Many versions of the PF are available in the literature; in the proposed approach we use the SIRPF. This algorithm comprises the following steps [84] :

1) Particle Generation: generate N $\{x_1(0),\ x_2(0),\ x_3(0), \ldots, x_N(0)\}$ initial particles according to the initial PDF *p(x(0))*.

2) Prediction Sampling : for each particle $x_i(k)$, propagate the $x_i(k+1)$ particle according to the transition PDF *p[x(k+1)|x(k)]*

3) Importance Sampling : for each particle $x_i(k+1)$, generate the $w_i(k+1) = p\,[Z(k+1)|\, x_i(k+1)]$.

4) Normalization and Rejection Sampling: The weights of the particles are normalized. Particles with low weight are deleted and particles with high weight are duplicated such that each particle has the same weight.

In this work, the dynamics of the particle filter are controlled by the data from the INS and the RSSI location fingerprints. If the tracked object's position was estimated at time *k*, the position estimation at time *k*+1 is guided by the INS to

generate particles in the direction given by the INS. For each particle $\boldsymbol{X_i}(k)$ = [$x_i(k)$ $y_i(k)$], the next particle $\boldsymbol{X_i}(k+1)$ can be obtained by:

$$x_i(k+1) = x_i(k) + d\cos(\Theta) + N(\mu,\sigma) \quad (4.6)$$

$$y_i(k+1) = y_i(k) + d\sin(\Theta) + N(\mu,\sigma) \quad (4.7)$$

Where *d* is the absolute distance traveled by the tracked object during the time interval [*k*, *k*+1] and Θ is the direction of *d*.

For each particle $\boldsymbol{X_i}(k)$ = [$x_i(k)$, $y_i(k)$], the RSSI distance between the closest location fingerprint to $\boldsymbol{X_i}(k)$ and the tracked object is calculated. The smaller this distance, the higher weight $\boldsymbol{X_i}(k)$ will have.

$$D^{T}_{FP_{closest}} = \sqrt{\sum_{j=1}^{M}\left(RSSI_j^{FP_{closest}} - RSSI_j^{T}\right)^2} \quad (4.8)$$

Where $D^{T}_{FP_{closest}}$ is the closest location fingerprint to particle $\boldsymbol{X_i}(k)$. Hence, the weight of $\boldsymbol{X_i}(k)$ is given by:

$$w_i(k) = \frac{1}{D^{T}_{FP_{closest}}} \quad (4.9)$$

Fig. 4.1 shows our overall localization system architecture which has three main components, the Inertial Navigation System (INS), the location fingerprints database and the particle filter for data fusion.

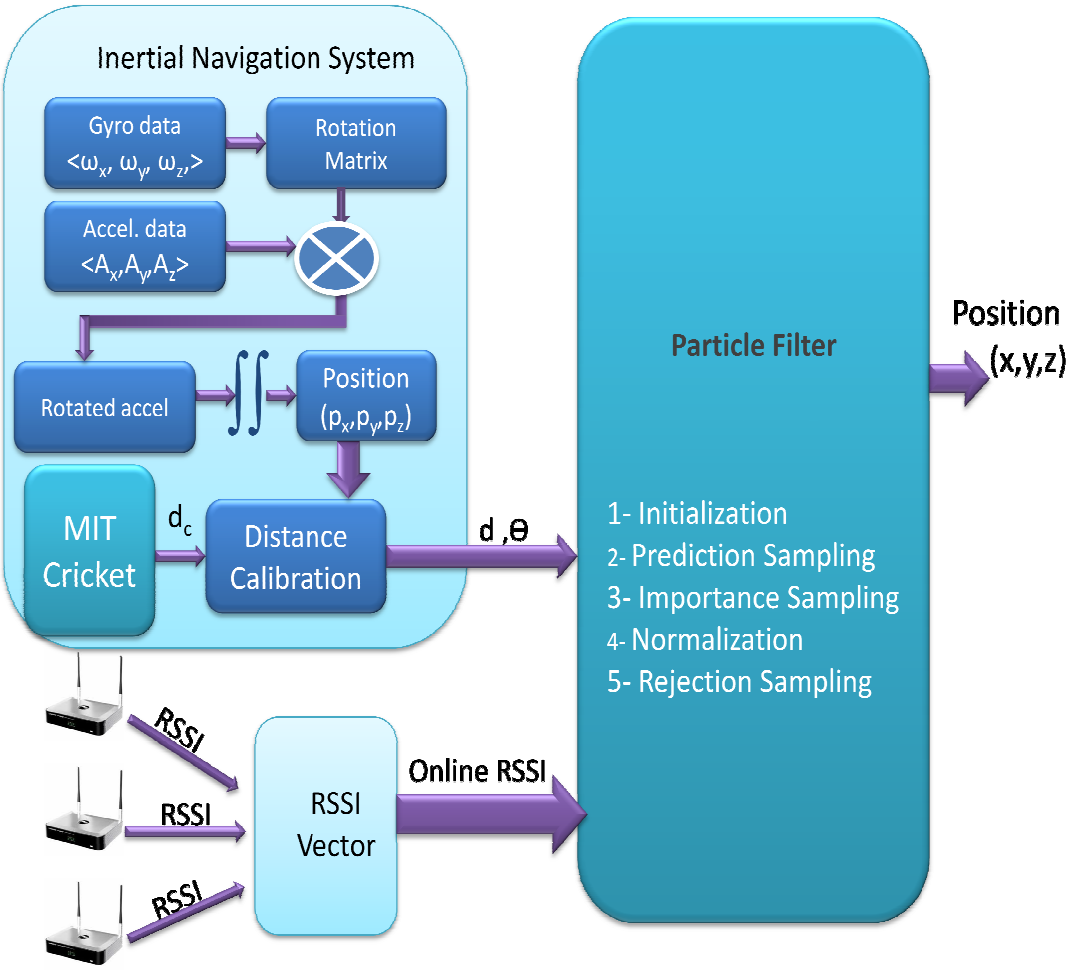

Figure 4.1:The overall localization system architecture.

4.3.3 Mathematical Error Analysis for Indoor Localization

In this section, the error analysis for indoor localization is addressed. Mathematical bound analysis on the position error in the proposed localization approach is also described in this section. The analysis is based on RSSI measurements observed at the tracked object's location, the RSSI measurements on

the Location Fingerprints (LF) around the localization estimate (P_E), and the INS data obtained from the proposed INS subsystem.

The localization accuracy is defined as the probability that the tracked object is within a certain distance (*D*) from its real location. Obviously, the closer the physical distance between the LF and the tracked object, the more weight that LF will have. The weight assigned to a given LF is composed of two parts:

1) Weight based on INS data.

2) Weight based on RSSI measurements.

The region of confidence around the location estimate is described as a circle. The probability that the tracked object is within this circle is calculated based on the weights of the LFs around the location estimate (P_E). For example, if the required localization accuracy equals 90%, we find the area of the circle such that the percentage of the LF weights that lies within this circle equals to 90% of the total weights of all LFs in the localization area. Fig. 4.2 shows a snapshot of the localization area at a given time.

The localization area shown in Fig. 4.2 has *M* LPAM radio base stations (T), *N* circles around the localization Position Estimate (P_E) and *K* location fingerprints on each circle. LF_n denotes the LF on the n^{th} circle. d_n^m denotes the distance between the LF on the n^{th} circle and the m^{th} LPAM radio base station. β_m denotes the angle between the m^{th} LPAM radio base station and the positive x-axis. The radius of the

inner circle is denoted by *R* which is also equal to the distance between any consecutive circles.

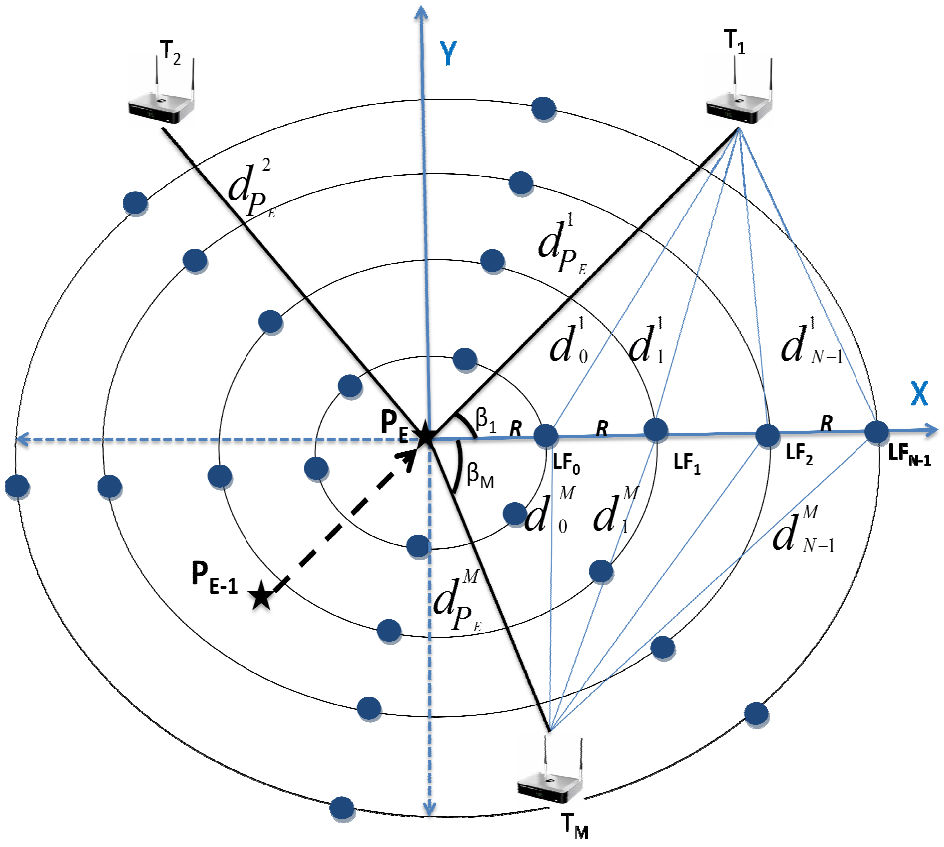

Figure 4.2: A snapshot of the localization area with the previous location estimate included.

A. RSSI-based Weight Analysis

The signal strength estimation between the tracked object and any access point is expressed in terms of the mean path loss and a log-normally distributed noise. The mean path loss at a given distance is given by:

$$\overline{PL}\,(d)[dBm] = PL(d0)[dBm] + 10 * v * \log10(\frac{d}{d0}) \qquad (4.10)$$

Where PL(d0)[dBm] is the path loss at a 1 meter distance in free space which is about -20 *dBm* [94], d0=1 *m*, and *v* is the mean path loss exponent and provides an indication of how fast path loss increases when distance increases. The RSSI at a given distance is then given by:

$$RSSI = \overline{PL}(d)[dBm] + X\sigma\ [dBm] \quad (4.11)$$

Where $X\sigma$ is a zero mean log-normally distributed random variable with standard deviation σ in decibels. The values of *v* and σ are suggested in [95] based on empirical experiments for various environments. In the proposed path loss model, we chose v=4.04 and σ = 4.3 for an office building. Fig. 4.3 shows the RSSI values against distance in meters.

Given *N* LFs and an online RSSI vector (*S*) for tracked object *T*, where $S = \{RSSI_1^T, RSSI_2^T, RSSI_3^T, \ldots, RSSI_M^T\}$, the Euclidian distance between *S* and LF_n RSSI vector is given by:

$$D_{LF_n}^{T} = \sqrt{\sum_{i=1}^{M}\left(RSSI_i^{LF_n} - RSSI_i^{T}\right)^2} \quad (4.12)$$

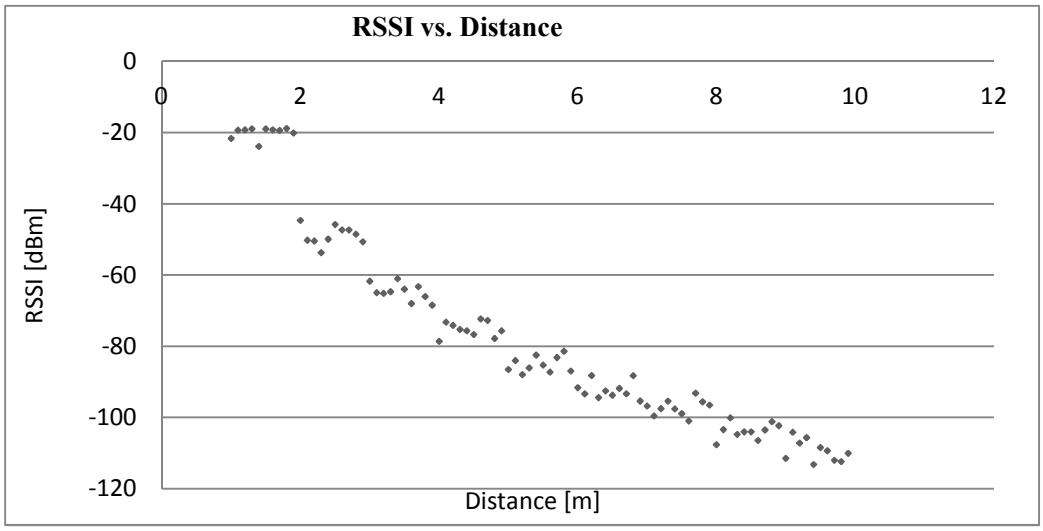

Figure 4.3: The RSSI values at various distances.

To show how the weight based on the RSSI of the n^{th} LF changes with the distance from the position estimate P_E, we represent this weight in terms of the radius of the n^{th} circle and the distance between P_E and the M access points in the localization area. A single RSSI measurement from access point i at the tracked object location is denoted by [95]:

$$RSSI_i^T = PL(d0)[dBm] + 10 * v * \log10(\frac{d_{P_E}^i}{d0}) + X\sigma \quad (4.13)$$

Where $d_{P_E}^i$ is the distance between the position estimate (P_E) and the i^{th} access point. As we showed in the previous section, the distance between the n^{th} location fingerprint and the i^{th} access point is denoted by:

$$d_n^i = \sqrt{(d_{P_E}^i)^2 + (nR)^2 - 2(nR)d_{P_E}^i\cos(\beta_i)} \quad (4.14)$$

Hence, the RSSI value on the n^{th} location fingerprint from measured from the i^{th} access point is denoted by:

$$RSSI_i^{LF_n} = PL\ (d0)\ [dBm] + 10 * v * \log_{10}\left(\left[(d_{P_E}^i)^2 + (nR)^2 - 2\,(nR)\,d_{P_E}^i \cos(\beta_i)\right] * d_0{}^{-1}\right) + X\sigma \quad (4.15)$$

The Euclidian distance between the online RSSI vector of the tracked object and the RSSI vector of the n^{th} location fingerprint is given by:

$$D_{FP_n}^T = \sqrt{\sum_{i=1}^{M}\left(RSSI_i^{LF_n} - RSSI_i^T\right)^2} \quad (4.16)$$

The non-normalized weight based on the RSSI measurements of the nth LF is then given by:

$$w = \frac{1}{D_{FP_n}^T} = \frac{1}{\sqrt{\sum_{i=1}^{M}\left(RSSI_i^{LF_n} - RSSI_i^T\right)^2}} \quad (4.17)$$

The normalized weight based on the RSSI measurements of the n^{th} LF is then given by:

$$w_n^{RSSI} = \frac{w}{\sum_{i=1}^{N} w_i} \quad (4.18)$$

B. INS- based weight analysis

Instead of finding the position estimate based on phase shifts or RSSI values only, the INS subsystem guides the future position estimate with certain distance and direction. We define the LF's normalized INS weight (w_n^{INS}) based on how close it is from the position estimate:

$$w_n^{INS} = \frac{1}{\sqrt{nR} * \sum_{j=1}^{N}\sqrt{jR}} \quad (4.19)$$

C. Combined-Weight Error Analysis

As a result of data fusion, the generic total weight of a given location fingerprint is calculated by combining the weights from the two approaches explained in the previous sections (w_n^{RSSI}, w_n^{INS}). The best combination approach is to multiply these normalized weights to get the total weight of LF:

$$w_n = w_n^{RSSI} * w_n^{INS} \quad (4.20)$$

Let *W* be the summation of the location fingerprints weights along the positive x-axis:

$$W = \sum_{i=0}^{N-1} w_n \quad (4.21)$$

Where *N* is the number of LFs on the positive x-axis.

Our goal is to find the value of *n* such that a certain percentage (*P*) of *W* lies within the n^{th} circle. In other words we want to find *n* such that:

$$\sum_{i=0}^{n-1} w_n = P * \sum_{i=0}^{N-1} w_n \quad (4.22)$$

Based on equations (4.18, 4.19, 4.20), it is difficult to find a deterministic solution for equation 4.22, we will use linear approximation to find a function En approximating w_n such that $E_n \geq w_n \; \forall \; n \in \{0,1, \ldots, N-1\}$.

Theorem 1:

The required localization accuracy defined as the probability (*P*) that the tracked object is within a certain distance *D* is guaranteed by the approximation function E_n.

Proof:

The value of w_n expressed in equation 4.20 decreases as the value of n increases which means that farther LFs have less weight than those who are close to the position estimate. This means that the function w_n is a decreasing function of n. The goal of linear approximation is to find a linear function that approximates the increasing function $q_n = 1/w_n$:

$$qn= \frac{1}{w_n^{INS} * w_n^{RSSI}} \quad (4.23)$$

The line (L_n) approximating q_n is required to have values such that $L_n \leq qn \; \forall \; n \in \{0,1, \ldots, N-1\}$. This guarantees that our approximation to w_n (E_n), will always be greater than $w_n \; \forall \; n \in \{0,1, \ldots, N-1\}$. Fig. 4.4 shows an example of q_n and L_n and Fig. 4.5 shows the corresponding $wn=1/q_n$ and $E_n=1/L_n$.The goal now is to find the linear approximation function (L_n) with the properties mentioned above. Since L_n is a line, it can be written as:

$$L_n = A.n+ B \quad (4.24)$$

Where A is the slope of L_n and B is the point where L_n intersects with the positive y-axis.

Since $L_n \leq q_n$, this means that B can be equal to q_0, which is the point at which q_n intersects with the positive y-axis (see Fig. 4.4). After finding B, the objective becomes finding the slope of L_n that guarantees no intersection between L_n and q_n $\forall n \in \{1,2, \ldots, N-1\}$. q_n is an increasing discrete function with values in $[w_0 : w_{N-1}]$.

Fig. 4.6 shows the slopes of all lines connecting pairs (w_n, w_{n+1}) $\forall\ n \in \{0,1, \ldots, N-1\}$. Choosing the slope with minimum value guarantees that *Ln* will not intersect with q_n $\forall\ n \in \{1,2, \ldots, N-1\}$.After finding the line $L_n = An+B$, the function E_n (Fig. 4.5) that will approximate w_n is of the form:

$$E_n = 1/L_n = \frac{1}{A.n+B} \qquad (4.25)$$

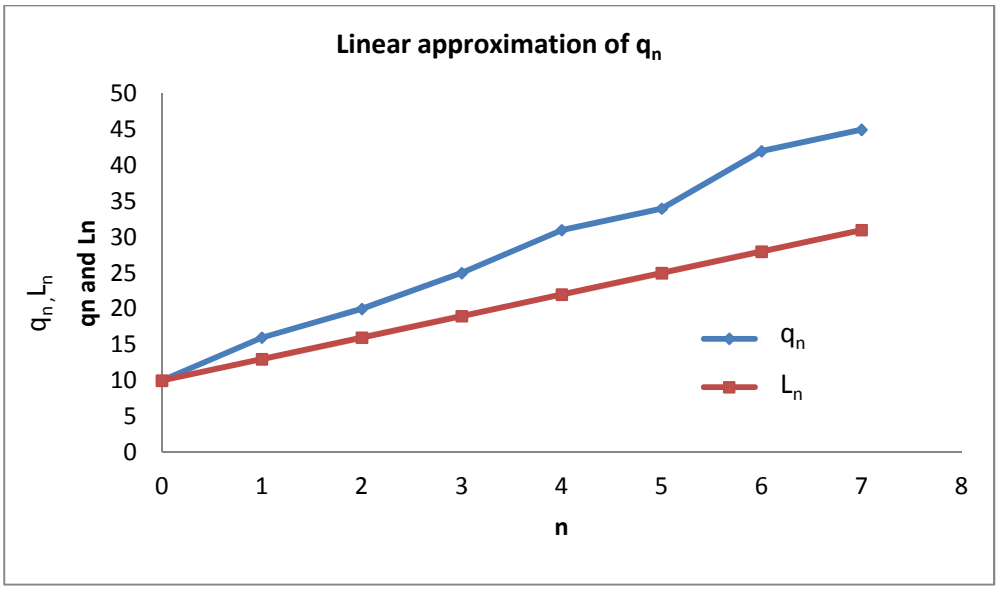

Figure 4.4: An example of q_n and L_n.

The analysis of this function is much easier than w_n in equation 4.20. To find the n^{th} circle at which lies in a percentage (P) of the total weight (W) is at least equivalent to finding P of the area under the curve E_n.

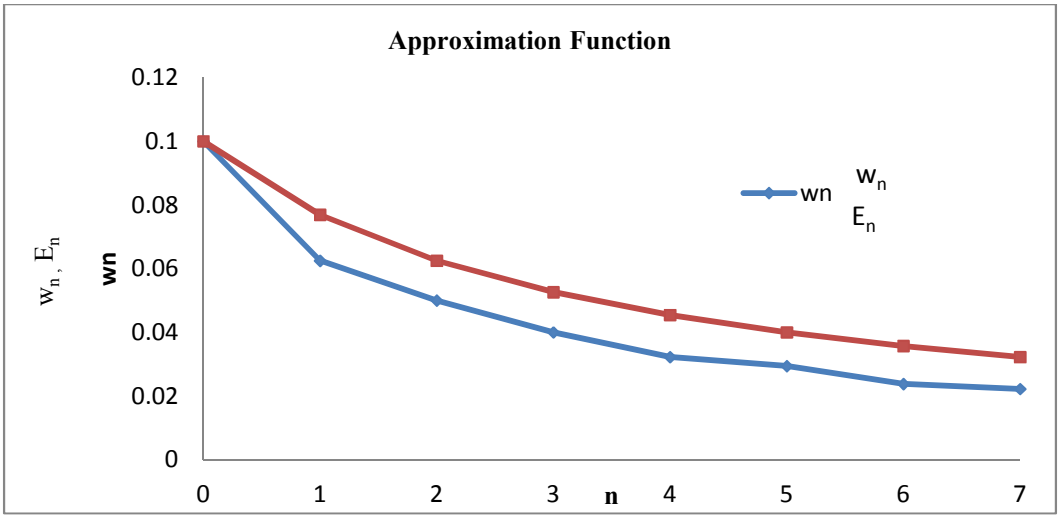

Figure 4.5: The required approximation function and the real w_n.

Since $E_n \geq w_n \ \forall \ n \in \{0,1, \ldots, N-1\}$, finding P of the area under the curve E_n implies finding P or more of the area under w_n. This guarantees that the localization accuracy will be more than P that the tracked object is within a certain distance.

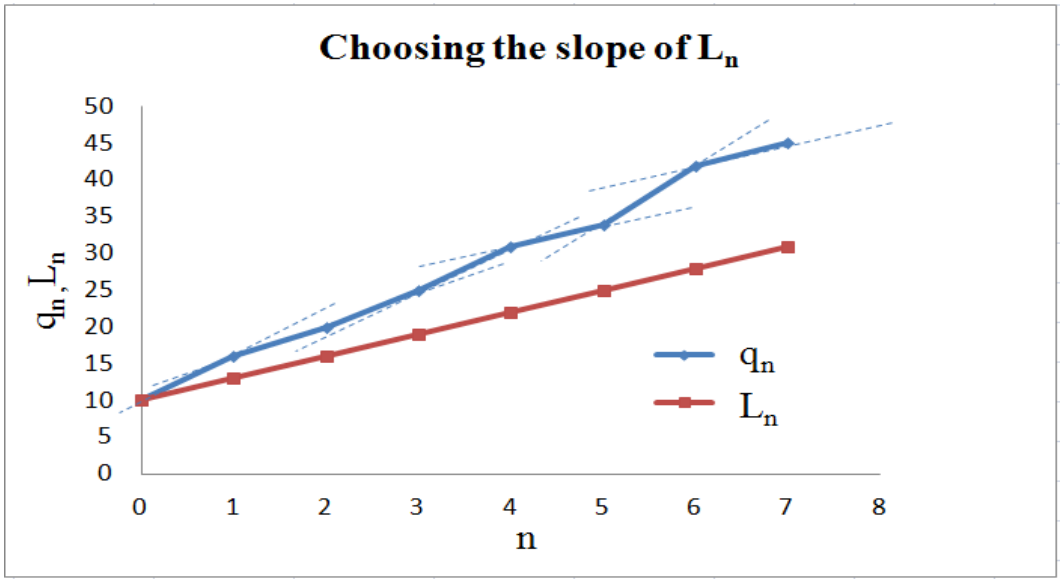

Figure 4.6: The slopes between all values of q_n.

D. Distance Estimation

To find the value of n at which lies P percentage of the area under E_n, compute the total area under E_n and then find the P percentage of that area. The area under E_n can be found by:

$$A_{En} = \int_0^{N-1} \frac{1}{A.n+B}\, dn$$

$$A_{En} = \frac{\ln A(N-1)+B}{A} - \frac{\ln B}{A}$$

$$A_{En} = \frac{\ln(A(N-1)+B)/B}{A} \tag{4.26}$$

Where N is the total number of LFs on the positive x-axis.

The value of n at which P^{th} percentage of the area under E_n is:

$$\int_0^{n-1} \frac{1}{A.n+B}\, dn = P * A_{En}$$

$$\frac{\ln(A(N-1)+B)/B}{A} = P * \frac{\ln(A(N-1)+B)/B}{A}$$

$$\ln \frac{(A(n-1)+B)}{B} = \ln\left[\frac{(A(N-1)+B)}{B}\right]^P$$

$$\frac{(A(n-1)+B)}{B} = \left[\frac{(A(N-1)+B)}{B}\right]^P$$

$$n = \frac{B * \left(([(A(N-1)+B)/B])^P - A - 1\right)}{A} \tag{4.27}$$

After finding the value of n, the distance (D) at which exists P percentage of the total weight W is equal to:

$$D = n * R \tag{4.28}$$

Where R is the radius of the inner circle around P_E .

Our approximation algorithm can be summarized in the following steps:

1- Find the values of w_n $\forall$ n$\in${0,1, …, N-1} using equation 4.20.

2- Set B = w_0.

3- SetA=$\underset{S}{\overset{\arg \text{MIN}}{}}$ [$S = (w_{i+1} - w_i,) \forall i \in \{0,1,\dots,N-1\}$]

4- Find n = $\frac{B * (([(A(N-1)+B)/B])^P - A - 1)}{A}$.

5 - Find D = n*R.

4.4 Simulation Setup

The simulation was performed in a 10x10 *m* room. The simulation starts by setting up the access points uniformly in the simulation area. The training data (location fingerprints) are then spread uniformly across the area. Fig. 4.7 shows the distribution of 2 access points and 35 location fingerprints in the simulation area.

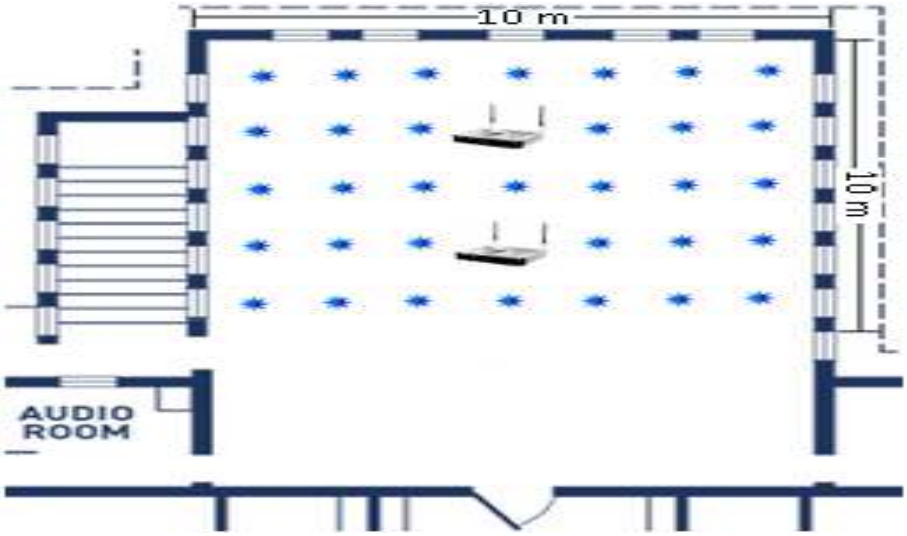

Figure 4.7: Distribution of access points and fingerprints in the simulation area.

The particles in the PF are divided into two groups: explorers and exploiters. About 20% of the particles are generated with high value for μ= 10 *m* to allow the particle filter converge quickly and prevent it from moving away from the optimal solution. The other 80% of particles are exploiters for the best possible solution.

As the tracked object moves in the simulation area, the position is calculated using the EUC, BDE and PF techniques. The position estimated by these techniques is then compared with the simulation position of the tracked object. The simulation was executed for three hours and then replicated for 10 times. The accuracy of the INS subsystem is assumed to have an average error of 5 *cm*.

4.5 Results

The main goal of this work is to fuse data from multiple sources to achieve minimum localization error. We compare the performance of the proposed EUC, BDE and PF in terms of the mean location error. System accuracy is defined as the probability that the tracked object is within a certain distance. We also show the enhancement achieved by using the PF for data fusion by comparing the performance of the PF with RSSI only and the PF with RSSI and INS data.

Fig. 4.8 shows the relation between the number of access points and the mean location error. As the number of access points increases, the mean location error decreases. The EUC error remains almost the same for 4 or more access points and no further enhancement can be achieved. The PF performance also remains unaffected by adding more than 7 access points to the simulation area. The performance of the

BDE approach becomes closer to that of the PF approach when increasing the number of access point to 20, which is impractical in real life.

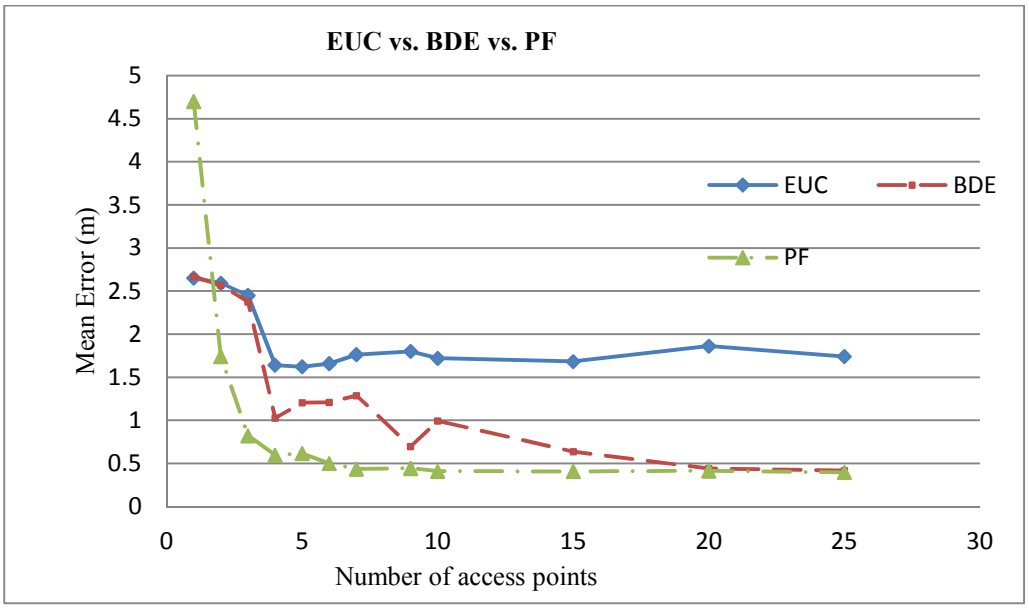

Figure 4.8: The mean location error vs. number of access points.

The location error Cumulative Density Function (CDF) shows that the PF outperforms both EUC and BDE in terms of location accuracy. For 3 access points, the probability that the tracked object is within 50 *cm* was about 50% and about 10% for both EUC and BDE. This probability increases rapidly to about 90% within the 80*cm* range for the PF and becomes about 15% for both BDE and EUC.

Fig. 4.9 shows that increasing the number of access points in the simulation area provide more accurate results. The figure shows the error CDF for 3 and 10 access points for the three localization techniques.

The data fusion between the RSSI and INS data provides better location accuracy than using the RSSI alone for location estimation.

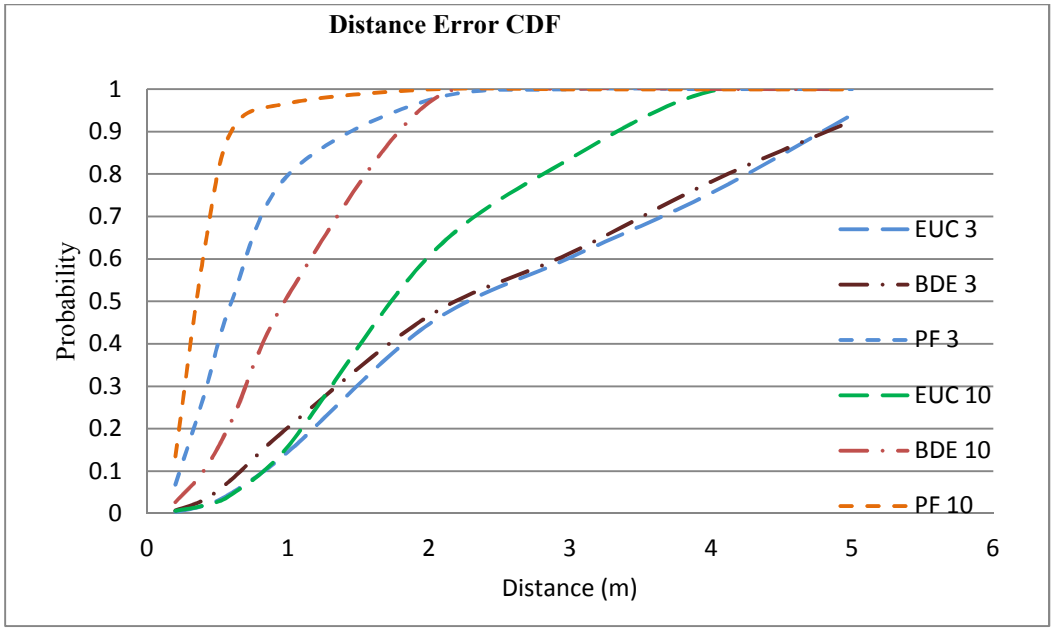

Figure 4.9: The CDF of the accuracy of the proposed localization system.

Fig. 4.10 shows the mean location error attained from the PF with RSSI only and PF with INS data. The data fusion achieves almost double the accuracy compared to RSSI-only location estimation.

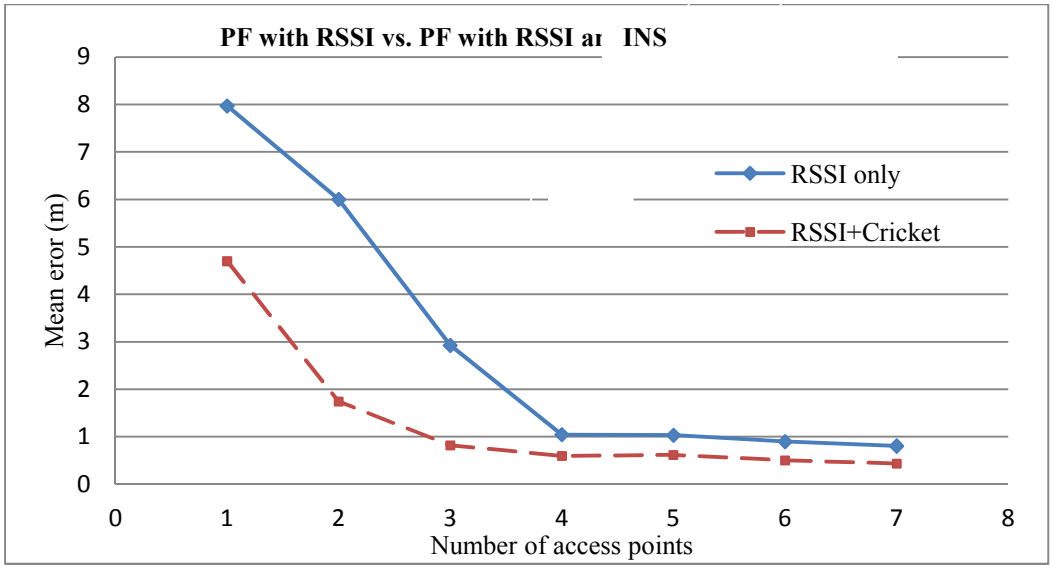

Figure 4.10: PF performance with RSSI measurements only and with RSSI and INS data.

CHAPTER 5

DATA FUSION IN SUPPORT OF OUTDOOR LOCALIZATION

5.1 Introduction

Outdoor localization is an important issue for many applications like autonomous mobile robotics and augmented reality. In this research, we propose a budgeted dynamic exclusion heuristic based on signal phase shifts from multiple base stations. We also propose an outdoor localization technique based on the particle filter for data fusion. The novelty of the proposed approach stems from its ability to fuse data collected from different sensor technologies to converge to more accurate distance estimation. The combination of multiple sensor data tends to overcome the drawbacks of using one sensor technology in the localization process.

In this section, we discuss the details of our proposed data fusion approach to combine data collected from multiple sensor technologies to achieve precise localization in outdoor environments. Our Inertial Navigation System (INS) is comprised of tri-axial accelerometer, gyroscope and an MIT Cricket system for gyroscope drift correction. The INS system provides the distance that the moving object travelled and the direction of that distance.

5.2 Proposed Approach

The substantial capabilities of the emerging Software Defined Radio (SDR) systems can be used to utilize AM radio signals for the purposes of indoor/outdoor localization. Phase shifts between the carriers of different AM radio signals can be used to calculate the distances between the receiver and the locally deployed Low Power AM (LPAM) radio base stations. Hence, the relative position of the receiver can be estimated within a Euclidian coordinate system.

Phase shift is any change that occurs in the phase of one signal, or in the phase of two different signals [96]. We refer to it by angle θ as it represents the shift from zero phase. For infinite sinusoids, the change in θ is the time-delay between two signals. If a sinusoidal signal *x(t)* is time-shifted (delayed) by ¼ of its cycle, it becomes :

$x(t - \frac{1}{4}T) = A \sin(2\pi f(t - \frac{1}{4}T) + \theta) = A \sin(2\pi f t - \pi/2 + \theta)$, which is equivalent to *x(t)* shifted by $\pi/2$ radians.

The SDR system on the tracked object surveys the phase shifts in signals received from multiple LPAM radio base stations and analyzes them to estimate the distances between the LPAM radio base stations and current location. In a configuration of three LPAM radio base stations $\{BS_1, BS_2, BS_3\}$ and one receiver $\{R\}$, the distances between R and the LPAM radio base stations can be estimated by:

$$D_R^{BS_1} = c * t_R^{BS_1} \quad (5.1)$$

$$D_R^{BS_2} = c * t_R^{BS_2} \quad (5.2)$$

$$D_R^{T_3} = c * t_R^{BS_3} \tag{5.3}$$

Where c is the Speed of Light, $D_R^{T_i}$ is the distance between R and T_i, and $t_R^{T_i}$ is the signal propagation time from T_i to R.

The receiver requires a minimum of three signals to estimate each $D_R^{T_i}$ based on the following equation

$$\frac{D_R^{BS_1} - D_R^{BS_2}}{c} = \theta_{BS_2}^{BS_1} \tag{5.4}$$

$$\frac{D_R^{BS_1} - D_R^{BS_3}}{c} = \theta_{BS_3}^{BS_1} \tag{5.5}$$

$$\frac{D_R^{BS_2} - D_R^{BS_3}}{c} = \theta_{BS_3}^{BS_2} \tag{5.6}$$

This is a system of three equations and three unknowns and can be solved by the Particle Swarm Optimization (PSO) [97] method described in the previous sections. $\theta_{BS_j}^{BS_i}$ is the measured phase shift between the signals from BS_i and BS_j and is expressed as:

$$\theta_{BS_j}^{BS_i} = t_R^{BS_i} - t_R^{BS_j} \tag{5.7}$$

A positive value for $\theta_{BS_j}^{BS_i}$ indicates that the signal from BS_i arrived before the signal from BS_j and vice versa. Our localization area is comprised of N LPAM radio base stations distributed over the coverage area. We utilize the phase shifts measured from the various LPAM radio base stations to estimate inter-base-station distances. The knowledge of these distances provides the relative position of each LPAM radio base station provided that we know the position of only one base station serves as a

reference of the coordinate system. We call this base station the Anchor Base Station (ABS). Fig. 5.1 shows our coordinate system. The ABS can be any base station in the area with known (*x, y*) coordinates.

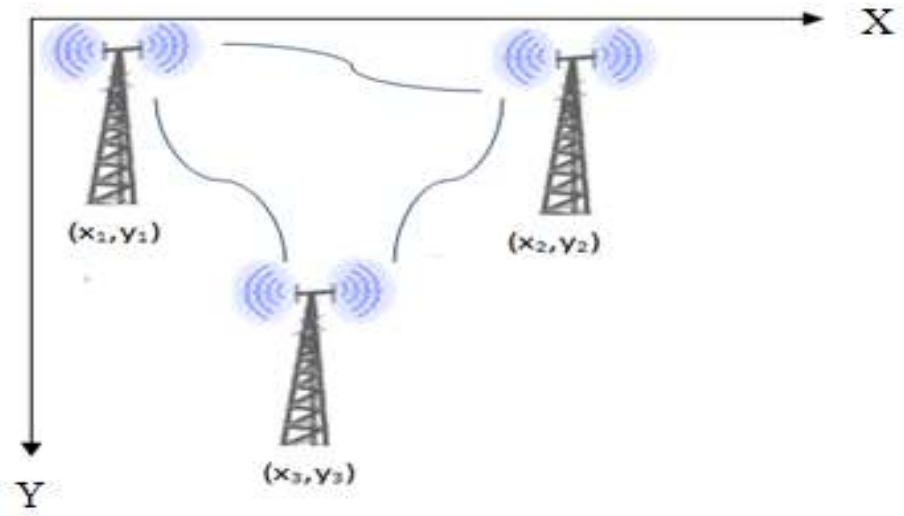

Figure 5.1: Our network coordinate system showing the LPAM radio base stations.

5.3 LPAM Radio Base Stations Position Estimation

For each BS_k, we use PSO to estimate its distance from other LPAM radio base stations. These distances are then used to estimate the positions of the remaining base stations in the localization area. The idea is to compute the distances between base stations from the measured phase shifts at each base station using the following constrained optimization problem:

Minimize:

$$\sum_{i}^{N-1}\sum_{j=i+1}^{N} \left| d_k^i - d_k^j - c * \theta_{BS_j}^{BS_i} \right|, \forall\ i < j,\ (i, j) \neq k,\ i \in \{1, \ldots, N\},\ k \notin \{1, \ldots, N\} \quad (5.8)$$

Such that:

$$d_k^i - d_k^j = c * \theta_{BS_j}^{BS_i} \qquad \forall \quad i< j,\ (i,\ j) \neq k,\ i\in\{1,\ \ldots,\ N\},\ k\notin\{1,\ \ldots,\ N\}$$

(5.9)

Where d_k^i is the estimated distance between BS and BS_k and is expressed by:

$$d_k^i = \sqrt[2]{(x_i - x_k)^2 + (y_i - y_k)^2} \tag{5.10}$$

Where (x_i, y_i) is the position of BS_i.

From graph theory, for *N* LPAM radio base stations, there are $\frac{N(N-1)}{2}$ estimated distances (edges) between the LPAM radio base stations. After finding these distances using equation 5.8, the positions of the *N* LPAM radio base stations are found by solving $\frac{N(N-1)}{2}$ equations of the form in equation 5.10. We also use the PSO method to solve for the *(x, y)* coordinates of the LPAM radio base stations using the following unconstrained optimization problem assuming that the ABS is at the known position (X_1, Y_1): **Minimize:**

$$\sum_{i=2}^{N} | d_i^1 - \sqrt[2]{(x_i - X_1)^2 + (y_i - Y_1)^2} | + \sum_{i=2}^{N} \sum_{j=i+1}^{N} [| d_j^i - \sqrt[2]{(x_i - x_j)^2 + (y_i - x_j)^2} |]$$

$\forall$ i<j (5.11)

The solution for this problem provides the coordinates of all LPAM radio base stations in the coverage area. This helps in estimating the location of the tracked object as explained in the next section.

5.4 **Euclidian Distance Heuristic (EUC) in Support of Outdoor Localization**

The knowledge of the locations of nearby LPAM radio base stations helps in finding the global location of any tracked object with an AM radio receiver. The same approach used to estimate the distances between the LPAM radio base stations which led to estimating the LPAM radio base stations' positions can be used to estimate the distances between the tracked object and the LPAM radio base stations.

For an area with *N* LPAM radio base stations and one receiver *R*, there are *N* distances between *R* and the LPAM radio base stations. The location estimation of the moving object requires that these distances be estimated first then used in the next step of location estimation. As noted before, it requires at least three LPAM radio base stations to triangulate the location of the tracked object as shown by equations 5.4-5.6.

Given the signal phase shifts observed at *R*, the distances between *R* and the LPAM radio base stations can be estimated by using the PSO method to solve the constrained optimization problem:

Minimize:

$$\sum_{i=1}^{N}\sum_{j=i+1}^{N} | d_R^i - d_R^j - c * \theta_{BS_j}^{BS_i} |, \forall \text{ i} < \text{j} \quad (5.12)$$

Such that :

$$d_R^i - d_R^j = c * \theta_{BS_j}^{BS_i}, \forall \text{ i} < \text{j}, \text{i} \in \{1, \ldots, \text{N}\} \quad (5.13)$$

Where d_R^i is the estimated distance between *R* and BS_i. To estimate the position of the tracked object, we substitute the estimated distances from equation

5.10 in the following unconstrained optimization problem and solve it using the PSO method to get the estimated position of the moving object:

Minimize:

$$\sum_{i=1}^{N} [\,|\, d_R^i - \sqrt[2]{(x_R - x_{BS_i})^2 + (y_R - y_{BS_i})^2}\,|\,] \quad (5.14)$$

Where (x_R, y_R) is the estimated position of the moving object and (x_{BS_i}, y_{BS_i}) is the position BS_i.

5.5 Budgeted Dynamic Exclusion (BDE) in Support of Outdoor Localization

The EUC approach utilizes the signals received from all LPAM radio base stations in location estimation. In practice, some signals can be noisy and harmfully contribute to the location estimation process. The BDE approach aims to identify these noisy LPAM radio base stations and remove them from the processing of location estimation process.

The BDE benefits from the estimated distances in equation 5.12 and the estimated position in equation 5.14 to calculate the total error of distance estimation. This is done by using the estimated position (x_R, y_R) of R and the positions of LPAM radio base stations to calculate the Euclidian distances between R and the LPAM radio base stations. These calculated distances are then compared with the estimated distances from equation 5.12. Estimated distances from noisy LPAM radio base stations have large difference between them and the calculated ones compared to ones

from non-noisy LPAM radio base stations. This helps identifying noisy LPAM radio base stations and excluding them from the location estimation process.

The BDE starts by removing a small number of LPAM radio base stations that have high distance differences. If the total error is decreased by more than the percentage of the removed LPAM radio base stations from the *N* LPAM radio base stations, BDE attempts to remove more noisy base stations to get a smaller location error. The BDE ends when removing more base stations does not reduce the total error by more than the percentage of the removed base stations.

The following provides the Pseudo code for the BDE approach for outdoor localization:

N: Total number of BSs.

N_{cur}: Current number of BSs.

N_{prev}: Previous number of BSs.

P_{init}: Initial percentage of removed LPAM radio base stations

$P_{inc:}$ The increment on initial percentage

P_{cur}: Current percentage

E_{cur}: Current total error

E_{prev}: Previous total error

E_{prev}= EUC total error

$P_{cur}=P_{init}$

$N_{cur}=N$

LOOP Remove $\lfloor (P_{cur} * N_{cur}) \rfloor$ *noisy LPAM radio base stations based on the differences between the calculated and estimates distances*

Perform EUC on the remaining LPAM radio base stations

$N_{cur} = P_{cur} * N_{cur}$

Compute E_{cur} *for the* N_{cur} *Ts*

If $\frac{E_{cur} - E_{prev}}{E_{prev}} > \frac{N_{cur} - N_{prev}}{N_{prev}}$

$N_{prev} = N_{cur}$

$E_{prev} = E_{cur}$

$P_{cur} = P_{cur} + P_{inc}$

goto LOOP

Else END

After applying this heuristic, the position estimated by the BDE is the result of applying the EUC computations described in the previous section on the set of LPAM radio base stations that were not removed by the BDE heuristic. The remaining base stations are the most likely non-noisy base stations in the area.

5.6 Particle Filter (PF) in Support of Outdoor Localization

In addition to the Sequential Monte Carlo (SMC) nature of estimation, the PF allows for flexible design and parallel implementation. The main advantage of the PF is its ability to combine measures from multiple sensors considering their probabilistic behavior. The key idea behind the PF is that the posterior Probability Density Function (PDF) of state *x(k)* is directly estimated conditioned on the set of measurements *Z(k)* according to the equations [84]:

$$p(x(k)|Z(k)) \approx \sum_{i=1}^{N} w_i(k)\delta[x(k) - x_i(k)] \quad (5.15)$$

$$\sum_{i=1}^{N} w_i(k) = 1 \quad (5.16)$$

Where N is the number of particles, $w_i(k)$ is the weight of the i^{th} particle $x_i(k)$ and $\delta(.)$ is the Dirac distribution.

The biggest advantage of using the PF instead of the EKF is its ability to solve non-Gaussian and non-linear estimation problems. Many versions of the PF are available in the literature; in our approach we use the SIRPF. This algorithm comprises the following steps [84] :

1) Particle Generation: generate N $\{x_1(0),\ x_2(0),\ x_3(0),\ \ldots,\ x_N(0)\}$ initial particles according to the initial PDF *p(x(0))*.

2) Prediction Sampling : for each particle $x_i(k)$, propagate the $x_i(k+1)$ particle according to the transition PDF *p[x(k+1)|x(k)]*

3) Importance Sampling : for each particle $x_i(k+1)$, generate the $w_i(k+1) = p\,[Z(k+1)|\ x_i(k+1)]$.

4) Normalization and Rejection Sampling: The weights of the particles are normalized. Particles with low weight are deleted and particles with high weight are duplicated such that each particle has the same weight.

In our work, the dynamics of the particle filter are controlled by the data from the INS and the RSSI location fingerprints. If the tracked object's position was estimated at time *k*, the position estimation at time *k*+1 is guided by the INS to generate particles in the direction given by the INS. For each particle $\boldsymbol{X_i}(k) = [x_i(k)\ y_i(k)]$, the next particle $\boldsymbol{X_i}(k+1)$ can be obtained by:

$$x_i(k+1) = x_i(k) + d\cos(\Theta) + N(\mu,\sigma) \quad (5.17)$$

$$y_i(k+1) = y_i(k) + d\sin(\Theta) + N(\mu,\sigma) \quad (5.18)$$

Where d is the absolute distance traveled by the tracked object during the time interval [k, k+1] and Θ is the direction of d.

The particle's weight is computed by comparing the phase shifts generated at the particle's position with the phase shifts received by the tracked object. The closer the particle's phase shifts to the tracked object's phase shifts, the more influence the particle will have in computing the tracked object's position. The following equation computes the weight of particle $\boldsymbol{X_i}(k)$ = [$x_i(k)$ $y_i(k)$] at time k:

$$w_i(k)=\frac{1}{\sum_{l=1}^{N-1}\sum_{j=l+1}^{N}|\,R_{\theta}{}_{\mathrm{BS}_j}^{\mathrm{BS}_l}-P_{\theta}{}_{\mathrm{BS}_j}^{\mathrm{BS}_l}\,|},\quad \forall\, l<j \tag{5.19}$$

Where $R_{\theta}{}_{\mathrm{BS}_j}^{\mathrm{BS}_l}$ is the received phase shift between signals from BS_l and BS_j on the receiver R and $P_{\theta}{}_{\mathrm{BS}_j}^{\mathrm{BS}_l}$ is the observed phase shift between signals from BS_l and BS_j for the particle $\boldsymbol{X_i}(k)$.

The PF approach can be applied to all LPAM radio base stations in the coverage area including the noisy LPAM radio base stations. It can also be applied after performing the BDE heuristic first to isolate the noisy LPAM radio base stations in the coverage area. Then, the PF approach is applied on the set of non-noisy LPAM radio base stations to get a better estimation of the tracked object location. Fig. 5.2 shows the overall architecture of our localization system which has three main components; namely the Inertial Navigation System (INS), the set of measured signal phase shifts by SDR and the particle filter for data fusion.

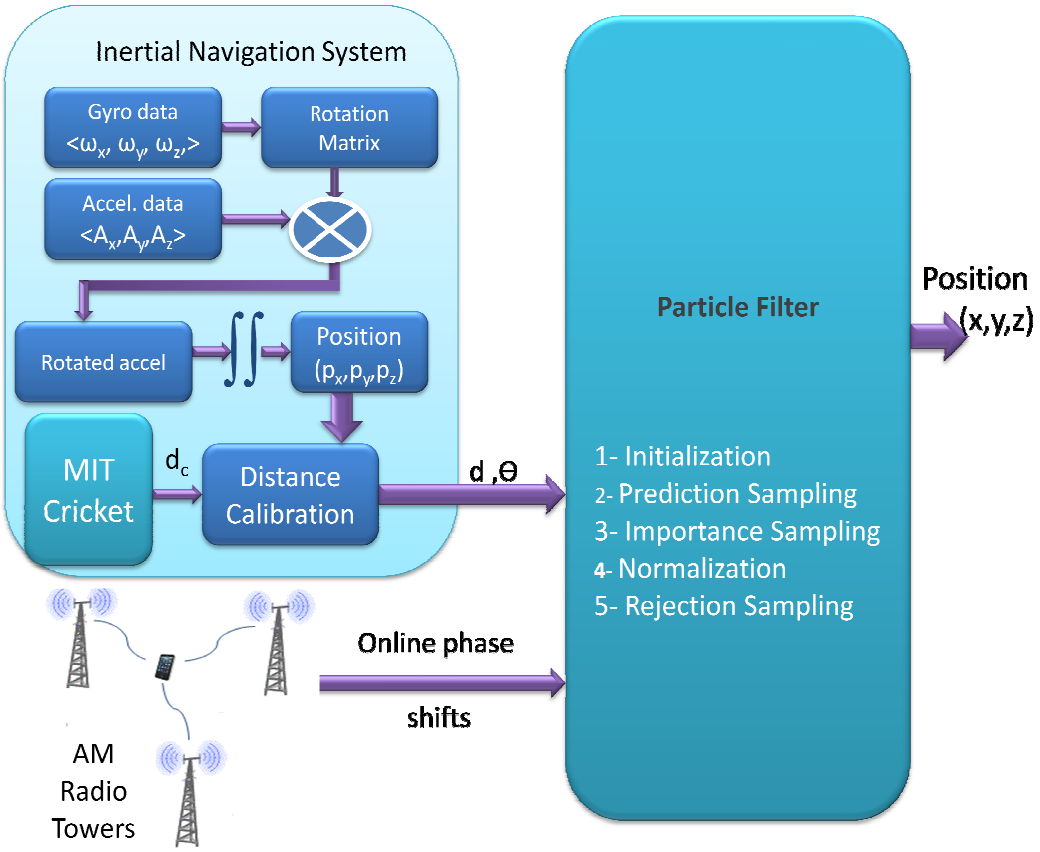

Figure 5.2: The overall localization system architecture.

5.7 Mathematical Error Analysis for Outdoor Localization

In this section, we address the error analysis for outdoor localization. For outdoor localization, the analysis is based on phase shifts observed at the tracked object's location, phase shifts at the LFs around the tracked object and INS data obtained from our INS subsystem.

Mathematical bound analysis on the position error in our localization approach is also described in this section. We define the localization accuracy as the probability that the tracked object is within a certain distance (D) from its real

location. Obviously, the closer the physical distance between the LF and the tracked object, the more weight that LF will have. The weight assigned to a given LF is composed of two parts:

1) Weight based on INS data (w_n^{INS}).

2) Weight based on signal phase shifts (w_n^{θ})

We describe the region of confidence around the location estimate as a circle. The probability that the tracked object is within this circle is calculated based on the weights of the LFs around the location estimate (P_E). For example, if the required localization accuracy equals 90%, we find the area of the circle such that the percentage of the LF weights that lies within this circle equals to 90% of the total weights of all LFs in the localization area. Fig. 5.3 shows a snapshot of the localization area at a given time.

The localization area shown in Fig. 5.3 has *M* LPAM radio base stations (T), *N* circles around the localization Position Estimate (P_E) and *K* location fingerprints on each circle. LF_n denotes the LF on the nth circle. d_n^m denotes the distance between the LF on the n^{th} circle and the m^{th} LPAM radio base station. βm denotes the angle between the m^{th} LPAM radio base station and the positive x-axis. The radius of the inner circle is denoted by *R* which is also equal to the distance between any consecutive circles

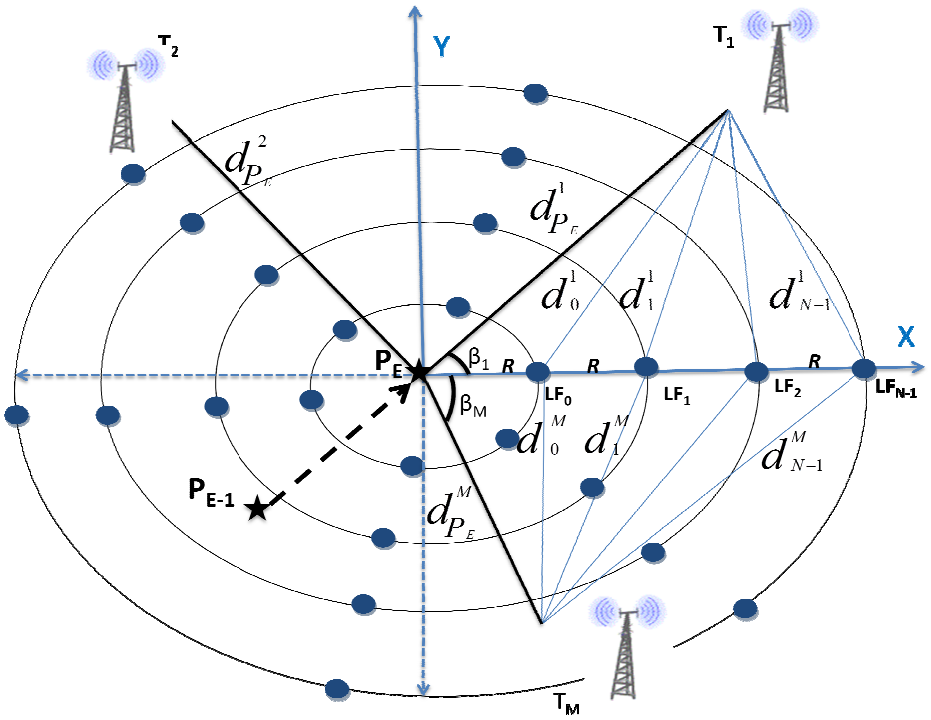

Figure 5.3: A snapshot of the localization area with the previous location estimate included.

5.7.1 Phase-Shift-Based Weight Analysis

As we mentioned before, the outdoor analysis depends on the phase shifts observed at the tracked object location, the phase shifts at the LFs and the data obtained from the INS subsystem. In this section, we will show how the weights of the LFs around the position estimate will change as we move farther from the position estimate.

The phase shift (θ) observed at P_E between the signal received from T_1 and T_2 is denoted by $\theta_{P_E}^{BS_1BS_2}$ and is equal to:

$$\theta_{P_E}^{BS_1BS_2} = \frac{d_{P_E}^1 - d_{P_E}^2}{C} \qquad (5.20)$$

Where *C= 2.99x108 m/s2* is the speed of light. Similarly, the phase shift (θ) observed at LF_1 between the signals received from BS_1 and BS_2 is denoted by $\theta_1^{BS_1BS_2}$ and is equal to:

$$\theta_1^{BS_1BS_2} = \frac{d_1^1 - d_1^2}{C} \quad (5.21)$$

As we mentioned early, the phase shifts weight assigned to a given LF depends on the closeness between the phase shifts (obtained from all the LPAM radio base stations) at LF and the phase shifts observed at the tracked object location P_E. The LF's weight also depends on INS data obtained from the INS subsystem.

We will derive the part of the weight depending on the phase shifts only, and then derive the part depending on the INS data. The final weight of the LF will be a multiplication of those two parts.

To simplify the error analysis, we calculate the weights for the location fingerprints residing on the positive x-access as shown in Fig. 5.3. The non-normalized weight for the n^{th} LF residing on the positive x-axis is denoted by w_n where $n \in \{0,1, \ldots, N-1\}$.

$$w = \frac{1}{\sum_{i=1}^{M}\sum_{j=i+1}^{M}\left|\theta_{P_E}^{BS_iBS_j} - \theta_n^{BS_iBST_j}\right|} \quad (5.22)$$

Where $\theta_n^{BS_iBS_j}$ is the phase shift between signals received from the i^{th} BS and the j^{th} BS observed at the n^{th} LF on the positive x-axis. $\theta_{P_E}^{BS_iBS_j}$ is the phase shift between signals received from the i^{th} BS and the j^{th} BS observed by the tracked object SDR system.

The goal is express w in terms of the distances between the P_E and all the LPAM radio base stations in the localization area and in terms of radius of the n^{th} circle around the estimate. This will show how w increases as we move farther from P_E. It is obvious from equation 5.21 that θ_n depends on the distance between LF_n and the entire LPAM radio base stations in the localization area (d_n^M). Hence, we will express these distances in terms of the distances between P_E and LPAM radio base stations and in terms of the radius of the n^{th} circle around P_E.

Using the law of cosines, the distance between the location fingerprints and the i^{th} LPAM radio base stations can be expressed as:

$$d_1^i = \sqrt{(d_{P_E}^i)^2 + R^2 - 2Rd_{P_E}^i \cos(\beta_i)} + Y_\sigma^i \tag{5.23}$$

$$d_2^i = \sqrt{(d_{P_E}^i)^2 + (2R)^2 - 2(2R)d_{P_E}^i \cos(\beta_i)} + Y_\sigma^i \tag{5.24}$$

$$\vdots$$

$$d_n^i = \sqrt{(d_{P_E}^i)^2 + (nR)^2 - 2(nR)d_{P_E}^i \cos(\beta_i)} + Y_\sigma^i \tag{5.25}$$

Where Y_σ^i, is a uniformly distributed noise on the distances observed from the LPAM radio base stations *i.* By substituting 5.25 into equations 5.21-5.22, the non-normalized weight of the n^{th} location fingerprint can be written as:

$$w = \left[\frac{1}{C}\sum_{i=1}^{M}\sum_{j=i+1}^{M}\left|\left[d_{P_E}^i - d_{P_E}^j\right] - \left[d_n^i - d_n^j\right]\right|\right]^{-1} \tag{5.26}$$

The normalized weight of the n^{th} LF is denoted by:

$$w_n^\theta = \frac{w}{\sum_{j=i}^{N} w_j} \tag{5.27}$$

5.7.2 INS-Based Weight Analysis

Instead of finding the position estimate based on phase shifts or RSSI values only, the INS subsystem guides the future position estimate with certain distance and direction. We define the LF's normalized INS weight (w_n^{INS}) based on how close it is from the position estimate:

$$w_n^{INS} = \frac{1}{\sqrt{nR} * \sum_{j=1}^{N} \sqrt{jR}} \quad (5.28)$$

5.7.3. Combined-Weight Error Analysis

As a result of data fusion, the generic total weight of a given location fingerprint is calculated by combining the weights from the two approaches explained in the previous sections (w_n^{θ}, w_n^{INS}). The best combination approach is to multiply these normalized weights to get the total weight of LF:

$$w_n = w_n^{\theta} * w_n^{INS} \quad (5.29)$$

Let *W* be the summation of the location fingerprints weights along the positive x-axis:

$$W = \sum_{i=0}^{N-1} w_n \quad (5.30)$$

Where *N* is the number of LFs on the positive x-axis.

Our goal is to find the value of *n* such that a certain percentage (*P*) of *W* lies within the n^{th} circle. In other words we want to find *n* such that:

$$\sum_{i=0}^{n-1} w_n = P * \sum_{i=0}^{N-1} w_n \quad (5.31)$$

Based on equations 5.27-5.29, it is difficult to find a deterministic solution for equation 5.31, we will use linear approximation to find a function En approximating w_n such that $E_n \geq w_n \; \forall \; n \in \{0,1, \ldots, N-1\}$.

Theorem 1:

The required localization accuracy defined as the probability (P) that the tracked object is within a certain distance D is guaranteed by the approximation function E_n.

Proof:

The value of w_n expressed in equation 5.29 decreases as the value of n increases which means that farther LFs have less weight than those who are close to the position estimate. This means that the function w_n is a decreasing function of n. The goal of linear approximation is to find a linear function that approximates the increasing function $q_n = 1/w_n$:

$$q_n = \frac{1}{w_n^{\theta} * w_n^{INS}} \tag{5.32}$$

The line (L_n) approximating q_n is required to have values such that $L_n \leq qn \; \forall \; n \in \{0,1, \ldots, N-1\}$. This guarantees that our approximation to w_n (E_n), will always be greater than $w_n \; \forall \; n \in \{0,1, \ldots, N-1\}$. Fig. 5.4 shows an example of q_n and L_n and Fig. 5.5 shows the corresponding $wn=1/q_n$ and $E_n=1/L_n$.

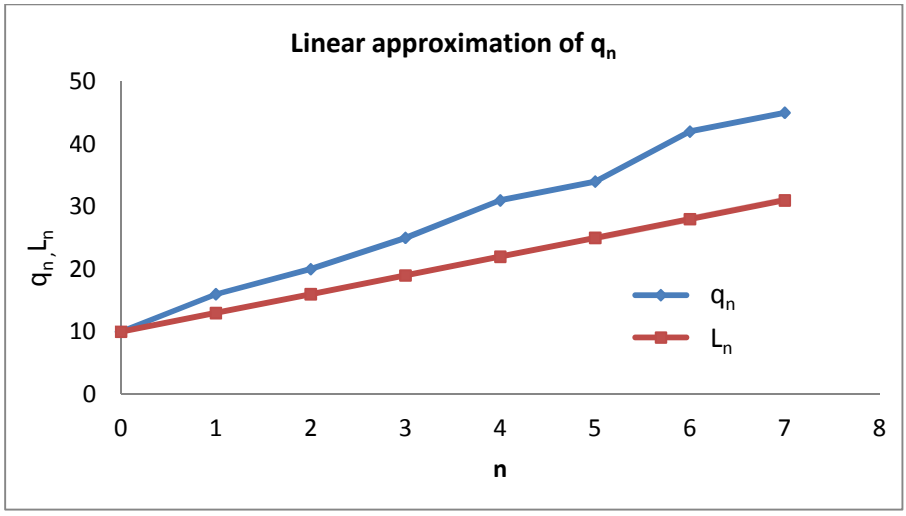

Figure 5.4: An example of q_n and L_n.

The goal now is to find the linear approximation function (L_n) with the properties mentioned above. Since L_n is a line, it can be written as:

$$L_n = A.n + B \tag{5.33}$$

Where A is the slope of L_n and B is the point where L_n intersects with the positive y-axis. Since $L_n \leq q_n$, this means that B can be equal to q_0, which is the point at which q_n intersects with the positive y-axis Fig. 5.4. After finding B, the objective becomes finding the slope of L_n that guarantees no intersection between L_n and q_n $\forall$ $n \in \{1,2, \ldots, N-1\}$. q_n is an increasing discrete function with values in $[w_0 : w_{N-1}]$.

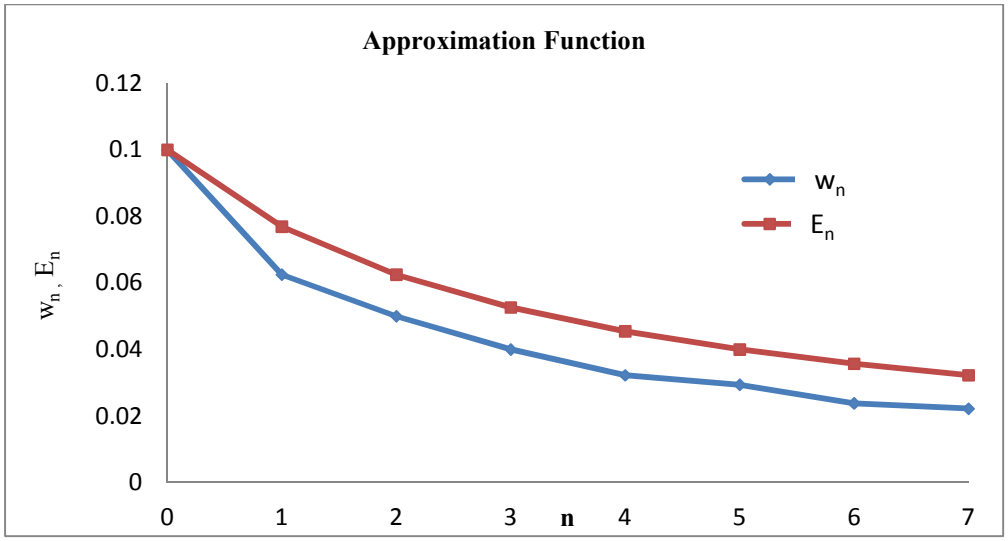

Figure 5.5: The required approximation function and the real wn.

Fig. 5.6 shows the slopes of all lines connecting pairs (w_n, w_{n+1}) $\forall\ n \in \{0,1, \ldots, N-1\}$. Choosing the slope with minimum value guarantees that *Ln* will not intersect with q_n $\forall\ n \in \{1,2, \ldots, N-1\}$.

After finding the line $L_n = An+B$, the function E_n (Fig. 5.5) that will approximate w_n is of the form:

$$E_n = 1/L_n = \frac{1}{A.n+B} \tag{5.34}$$

The analysis of this function is much easier than w_n in equation 5.29. To find the n^{th} circle at which lies in a percentage (P) of the total weight (W) is at least equivalent to finding P of the area under the curve E_n.

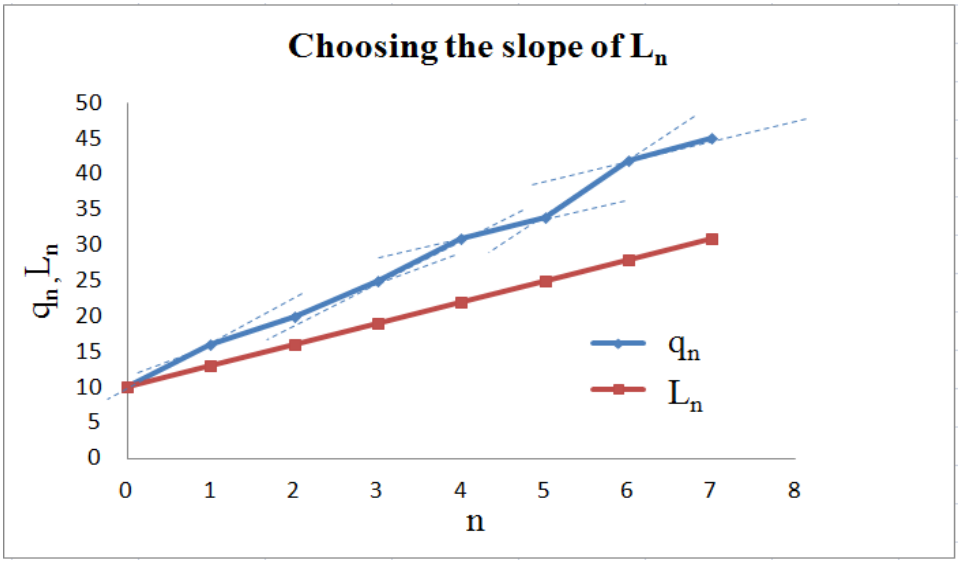

Figure 5.6: The slopes between all values of q_n.

Since $E_n \geq w_n \ \forall \ n \in \{0,1, \ldots, N-1\}$, finding P of the area under the curve E_n implies finding P or more of the area under w_n. This guarantees that the localization accuracy will be more than P that the tracked object is within a certain distance.

5.7.4. Distance Estimation

To find the value of n at which lies P percentage of the area under E_n, compute the total area under E_n and then find the P percentage of that area. The area under E_n can be found by:

$$AEn = \int_0^{N-1} \frac{1}{A.n+B} \, dn$$

$$AEn = \frac{\ln A(N-1)+B}{A} - \frac{\ln B}{A}$$

$$AEn = \frac{\ln(A(N-1)+B)/B}{A} \qquad (5.35)$$

Where N is the total number of LFs on the positive x-axis.

The value of n at which P^{th} percentage of the area under E_n is:

$$\int_0^{n-1} \frac{1}{A.n+B}\, dn = P^* AEn$$

$$\frac{\ln(A(N-1)+B)/B}{A} = P * \frac{\ln(A(N-1)+B)/B}{A}$$

$$\ln \frac{(A(n-1)+B)}{B} = \ln\left[\frac{(A(N-1)+B)}{B}\right]^P$$

$$\frac{(A(n-1)+B)}{B} = \left[\frac{(A(N-1)+B)}{B}\right]^P$$

$$n = \frac{B * \left(([(A(N-1)+B)/B])^P - A - 1\right)}{A} \quad (5.36)$$

After finding the value of n, the distance (D) at which exists P percentage of the total weight W is equal to:

$$D = n * R \quad (5.37)$$

Where R is the radius of the inner circle around P_E .

Our approximation algorithm can be summarized in the following steps:

1) Find the values of w_n $\forall$ n$\in\{0,1, \ldots, N-1\}$ using equation 5.29.

2) Set $B = w_0$.

3) Set $A = \underset{S}{\arg\,MIN} [S = (w_{i+1} - w_i,) \forall i \in \{0,1, \ldots, N-1\}]$

4) Find $n = \frac{B * \left(([(A(N-1)+B)/B])^P - A - 1\right)}{A}$.

5) Find $D = n*R$.

5.8 Simulation Setup

The simulation was performed in a 100x100m area. The simulation starts by setting up the LPAM radio base stations uniformly in the simulation area. The noise is expressed as the inaccuracy in the phase shifts received from a specific LPAM radio base station. Equation 5.7 is used to generate the phase shifts between signals from different LPAM radio base stations. The distance between *R* and a noisy LPAM radio base station is simulated to have a value equivalent to the simulation distance plus a noise factor between 1m and 5 m.

On the other hand, a distance from a non-noisy LPAM radio base station is simulated to have a value equivalent to the simulation distance plus a noise factor between 0m and 1m. This noise is due to inaccuracy in estimating the LPAM radio base stations' positions. It is also a result of other sources of noise in AM radio signals (i.e. nearby power lines, electric motors, TV sets, etc.). The accuracy of the INS subsystem is assumed to have an average error of 5cm.

5.9 Results

The main goal of this work is to fuse data from multiple sources to achieve minimum localization error. We compare the performance of our proposed EUC, BDE and PF approaches in terms of the mean location errors, and location accuracy defined as the probability that the target node is within a certain distance. We also show the enhancement achieved by using the PF for data fusion by comparing the

performance of the PF using only signal phase shifts and the PF with phase shifts and INS data.

The first experiment studies the effect of increasing the number of LPAM radio base stations in the area on the localization performance. Fig. 5.7 shows that the PF overcomes both EUC and BDE techniques in terms of the mean location error when having 20% noisy LPAM radio base stations in the area. Both BDE and EUC are highly sensitive to increasing the number of LPAM radio base stations in the area while the PF performance remains unaffected by increasing the number of LPAM radio base stations to more than 9 stations. It is also clear how the BDE approach reacts to increasing the number of noisy LPAM radio base stations compared to BDE. The EUC is badly affected when more noisy LPAM radio base stations are in the area while BDE was able to detect these noisy stations and isolate them from the location estimation process.

The percentage of noisy LPAM radio base stations in the coverage area has a big impact on the localization accuracy. Fig. 5.8 shows how the three approaches react to increasing this percentage. The figure also shows that BDE fails to detect the noisy LPAM radio base stations when the percentage of noisy LPAM radio base stations is more than 40%. When the percentage of noisy LPAM radio base stations is more than 40% the PSO solution is biased towards the noisy LPAM radio base stations in the solution space and incorrectly removes the non-noisy LPAM radio base stations instead of noisy LPAM radio base stations.

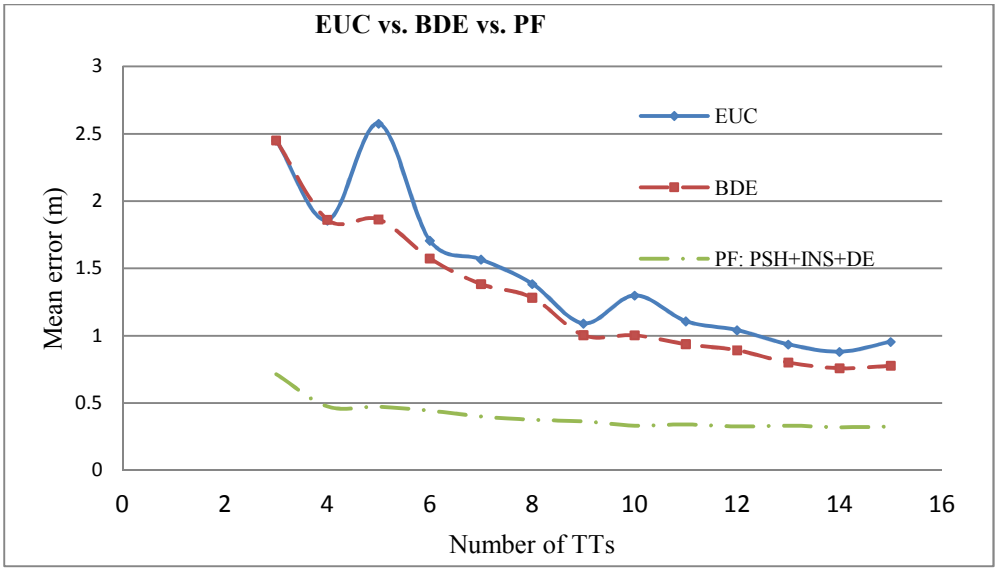

Figure 5.7: The effect of increasing the number of BSs on system performance.

The INS subsystem greatly enhances the localization accuracy. Fig. 5.9 shows the performance of the particle filter when working with phase shift data alone and when fusing it with INS data. The localization accuracy is almost 7 times better with INS data being fused with phase shift data. The figure also shows how the particle filter (with phase shifts only) is affected by increasing the number of noisy LPAM radio base stations in the area. The number of noisy LPAM radio base stations in this figure is 20% of the total number of LPAM radio base stations.

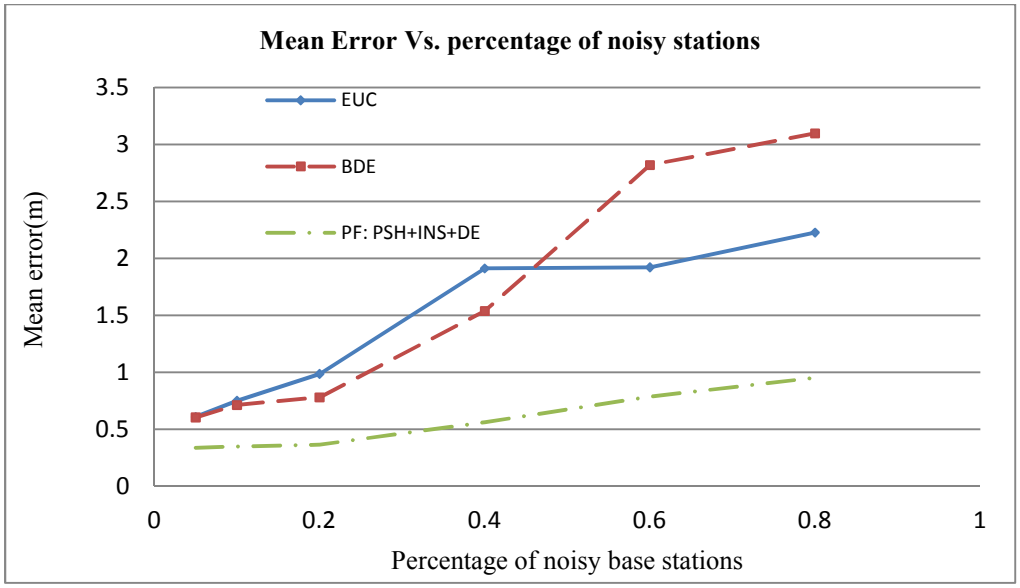

Figure 5.8: The effect of increasing the percentage of noisy BSs on system performance.

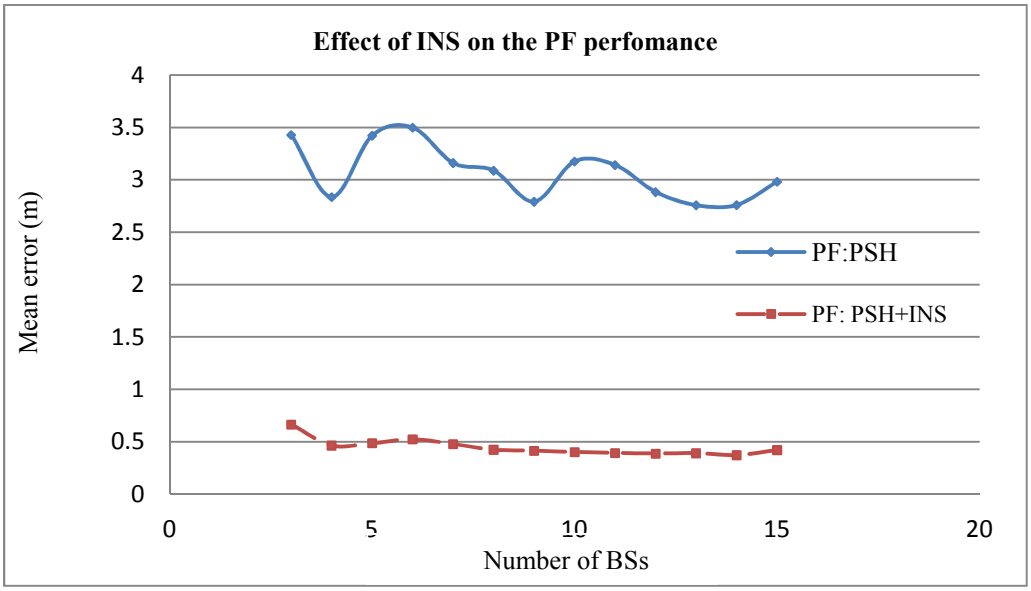

Figure 5.9: INS contribution to localization performance.

The BDE is also integrated with the PF to achieve further location accuracy by allowing the PF to use phase shift measurements collected from non-noisy LPAM

radio base stations when performing data fusion. Fig. 5.10 shows that the mean location error is enhanced by about 25% when integrating the BDE with the PF. The figure shows the localization performance when there are 20% noisy LPAM radio base stations in the coverage area.

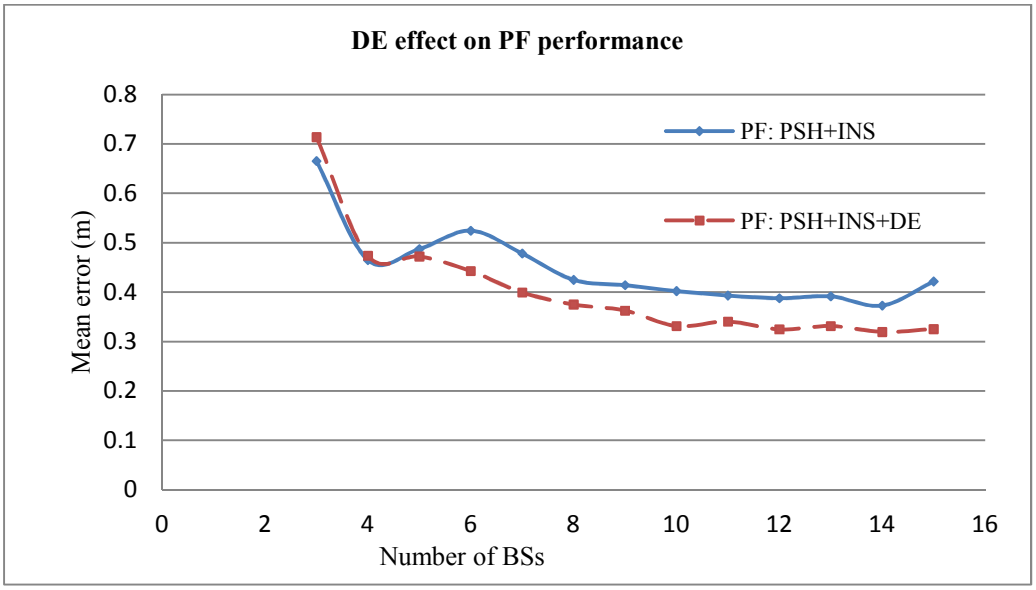

Figure 5.10: Integration of BDE with PF.

The location error CDF shown in Fig. 5.11 shows that the PF approach outperforms both EUC and BDE in terms of location accuracy. For 10 LPAM radio base stations, the probability that the tracked object is within 40cm was about 50% and about 5% for both EUC and BDE. This probability increases rapidly to about 90% within the 70cm range for the PF and becomes about 20% for both BDE and EUC.

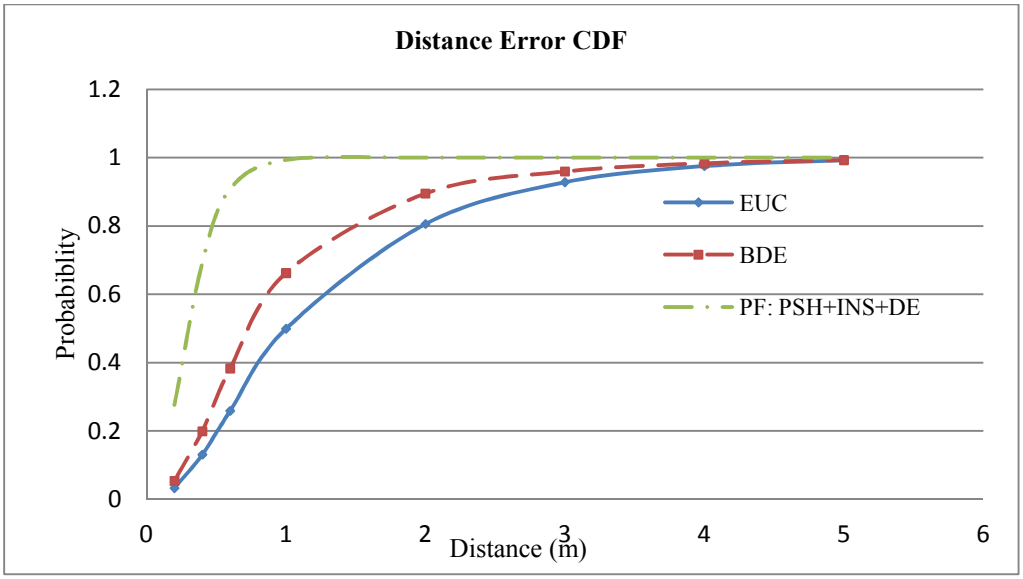

Figure 5.11: Localization accuracy CDF.

.

CHAPTER 6

SOCIAL NETWORK FOR BLIND AND VISUALLY IMPAIRED

6.1 Introduction and Background

The Blind and Visually Impaired (BVI) encounter various difficulties during their daily activities like path planning, navigation and obstacle avoidance. Many BVI still trust and rely on the white cane to explore their environments. The state-of-the-art white canes have audio systems to guide the BVI through their environment. In this chapter, we present a top-down design approach for a social collaboration service especially designed for the BVI.

The Blind and Visually Impaired (BVI) usually lack the information needed to bypass obstacles and hazards. They also have little knowledge about the landmarks in their surrounding environment and about appropriate routes that need to be followed from a source to a destination. In this work, we propose a social collaboration service that provides such information to make this important segment of our society more independent.

Research and development efforts that focus of the use of technology to help the BVI started not long time ago. These efforts focused on building tools to avoid obstacles. The regular white cane (which is used by most of the blind) is supplemented with laser and ultra sound to efficiently help the blind avoid obstacles. However, these tools are not enough to help the BVI navigate complex environments without the assistance of others.

In the last two decades, research and development efforts have focused mainly on the navigation component of assistive devices. Location identifiers were affixed on places with known locations to help the blind identify land marks in the environment. Those identifiers are then equipped with sensors and the blind can sense these identifiers using special equipments [98]. The disadvantage of this approach is that individuals have to scan the entire surrounding environment to search for these identifiers.

An example of systems that use location identifiers is the Talking Signs System [99]. In this system, infrared transmitters are spread throughout the environment and continuously send digital speech signals in a range of 15-40 meters depending on their battery power. The receiver held by the user picks up these signals and the user can listen to these voice commands. Transmitter localization can also be achieved using a hand-held receiver for maximum efficiency. The disadvantage of using such systems is their high installation cost and maintenance relative to the limited coverage they provide [98].

A group of researchers at the University of Rome developed a system called Project II [100] to help the blind navigate their environment. The system is comprised of a tag network, an RFID tag reader and a PDA with a system installed especially for this purpose. The RFID tags are spread all over the test floor to provide the system with all kinds of needed information. The reader is installed in a cane similar to those used by the blind and the antenna is installed in the extremity of the cane near to the ground. The antenna is directly connected to the reader which connects to the PDA

using Bluetooth. The reader continuously reads the signals from the tags and transmits them to the PDA which in turn converts them into speech signals that can be heard by the user through the use of headsets. The disadvantage of using such systems is the high cost of installation and it requires the blind to continuously wear a headset while moving around.

Most recent systems like iNAV [101] is based on a compass and consisted of many sensors to provide location and orientation. Using compass for providing a decentralized location service allowed iNAV to separate navigation and positioning problems. Cricket motes by MIT [89] utilize the speed difference between light and sound to compute the distance between a sender called "*Beacon*" and a receiver. In another work, a system called DOLPHIN [102] uses a distribution of ultra sound sensors to determine the absolute location of the user. SWAN [103] determines the location of the user using the signal strength of the radio signal observed at the smart phone to determine the location of the user. Finally, the Active Bat [104] system uses the reflections of ultrasonic pulses to determine the position and the orientation of the blind person.

All of the aforementioned systems provide position coordinates and/or orientation details needed as a foundation to provide a comprehensive social collaboration environment for the blind. What these systems do not provide is a collaboration environment that enables its users to easily associate and share data with location coordinates and orientations. In this work, we present a complete social collaboration environment for the blind and visually impaired and demonstrate how

the user experience can be collected over time and shared among the users; thus, leveraging the power of the crowd to quickly build and mine a database that associates data with location coordinates and orientations.

6.2 Social Networking for the Blind

A survey [105] conducted by researchers at the University of Birmingham in 2012 targeted a group of blind and visually impaired students at the age of 14 years and up. The goal of the survey was to understand how this segment of students access the Internet and whether they are interested in using known social networking sites or not. Furthermore, the work intended to know why the blind uses social networking sites and whether they use their phones to access these sites and whether accessibility to these sites is easier through cell phones or computers. [105].

Almost all of the participants in the survey access the internet at home from their own computers and about 69% access the internet through their mobile phones. The reasons for internet access vary between all the participants. The majority of the participants are familiar with the term "*Social Networking*" and all of the participants mentioned Facebook as an example of a social networking site while 80% mentioned Twitter [105].

About 15% of the participants were unable to register in the social networking website by their own because of their visual impairment. The website that most of the participants were unable to register on was Facebook which is expected because Facebook was the most popular site to be used. Participants with severe visual

impairment were those who face more difficulty in site registration. Only about fifth of the 70 participants said that the accessibility of the website determines whether or not to register in the website. Website accessibility is defined in [105] as:

> "*The ease with which you can use the site because it may be difficult to set the colours or size of things on the screen for you to see properly, or difficult for you to use a screen reader.*"

The survey in [105] states that about 60% of the participant access the social networking website on a daily basis spending about 20-40 minutes. The most popular functionality on the social network website is chatting which participants use to keep up with their friends, communicate with relatives living far away and even discussing homework. Six of the participants said that they were intimidated by using social networking website either for issues related to their eyesight or because of unpleasant comments [105].

Only one participant had no access to a mobile phone and about 75% of the participants have smart phones. Most of the phone features used by the participants are related to the accessibility of the mobile phone. The majority of the participants use mobile phone to mainly communicate with their families and friends. Other activities are listening to music, internet access, using phone applications and social networking [105].

6.3 Software Architecture of a Social Network for BVI

6.3.1 Overview

Our goal is to build a social network especially designed for BVI. In this section we present a software architecture for a social collaboration environment for the blind. This network provides assistance for the blind and visually impaired while they travel from a source Point of Interest (PoI) to a destination location PoI. The social collaboration environment provides route planning services, obstacle avoidance services (i.e. what kind of obstacles available on a specific route and how to navigate around them), route alternatives service and the ability to tag a certain PoI both by audio and text.

The user is given the ability to create PoIs and register them on the collaboration server. The user can tag each PoI by providing textual or audio metadata that provides details about the PoI. The PoI is then registered on the server with its location and orientation information. PoIs can be public or private based on the user preference. This gives the user the ability to tag places in his/her own private environment (home, office, etc.) and will allow the user to navigate between these PoIs. If the user creates and tags a public PoI, then the PoI will be accessible by other users using the collaboration network.

Keyhole Markup Language (KML), a product of Google Earth, is used to represent the PoI tags through XML. KML provides the ability to provide meta-data about a certain location using XML. The user can provide a description and exact

location information for the PoI. Fig. 6.1 shows an example of a tag created for the Waldo Library at Western Michigan University.

In order to navigate between various PoIs, the user is equipped with our indoor/outdoor localization system that provides precise position and orientation information and continuously sends it to the collaboration server. Our precise localization system allows the user to create PoIs with sub-meter accuracy. As the server receives this information, it provides textual or audio assistance to the user about the assigned route and the PoIs located on the route from the source to destination. Users are able to configure their preferences on whether they like to be provided with audio or tactile instructions to allow them to navigate through the details of the followed route. Furthermore, the system allows the user to interact with other users (using text or audio) within a certain distance to share their experiences.

```
<?xml version="1.0" encoding="UTF-8"?>
<kml xmlns="http://www.opengis.net/kml/2.2">
<Document>
<Placemark>
 <name>Waldo Library </name>
 <description> Waldo Library located at West Michigan Avenue  in Kalamazoo,
MI. The best way to arrive to this library is to follow Aracadia Rd and head north-
west as this route has minimum number of obstacles and has special road-way for
pedestrians along all the way. </description>
 <Point>
  <coordinates>42°16'56.21"N, 85°36'49.24"W</coordinates>
```

Figure 6.1: An example of a PoI tag using KML.

6.3.2 General Architecture

In this section we provide generic software architecture of the proposed social collaboration services. Fig. 6.2 shows the block diagram for the overall system while a detailed discussion is provided in section III-C.

The system components are hosted on the server which the users will connect with while using the collaboration services. The User Interface (UI) provides the user with the ability to tag locations and pass it to the Rule-Based Expert System (RBES) which is considered the "*Brain*" of the system. The RBES analyzes all the rules that are related to the user's current location and orientation and provides the proper assistance to the user.

All the rules and user accounts are stored in the system's database. The user must be authenticated to the system first before starting to use the collaboration services. Users can connect to the server from any device and uses Google maps augmented with audio/tactile instructions to navigate outdoor environments (floor plans augmented with audio/tactile instructions are utilized for indoor environments).

The system also provides some reporting services functionality through the reporting services system. This reporting service provides some statistical reports about all network services (i.e. the number of current online users, the number of visually impaired users logged in the system in the past three months, etc.). The location tagging sub-system interacts with the user through the UI provides the user with the ability to create public or private location tags in the form of text or audio.

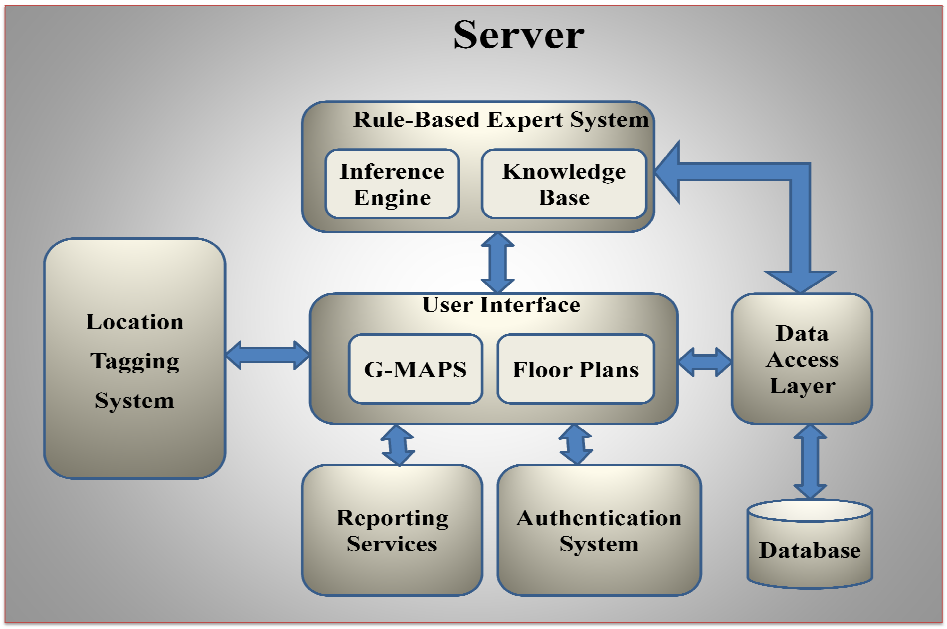

Figure 6.2: System block diagram.

The system also provides some reporting services functionality through the reporting services system. This reporting service provides some statistical reports about all network services (i.e. the number of current online users, the number of visually impaired users logged in the system in the past three months, etc.). The location tagging sub-system interacts with the user through the UI and provides the user with the ability to create public or private location tags in the form of text or audio.

6.3.3 System Objects

In this section we provide a description of the various system objects and describe their functionality. Fig. 6.4 shows the class diagram of the overall system and the relations between its objects.

1) The Server Object: This object will host our social collaboration services. All other system objects will be part of this object. Users connect to this object through the Internet and interact with it as they move between PoIs.

2) Point of Interest (PoI) Object: A PoI can be any place with known location. The user can create a PoI of interest, tag it with textual or audio information and register it on the system. As mentioned earlier, PoIs can have public visibility (i.e. a cafe on a route, restaurant, bookstore, mall, etc) or can have private visibility (i.e. parent's house, desk location in the living room, etc.).

3) User Object: This object is created once a user gets registered on the collaboration network.

4) Profile Object: Each user has a profile object that contains the user's preferences and details.

5) Tag Object: This object contains a description about a certain PoI and is created by the user and represented using KML. The Tag can contain textual or audio description about the PoI. For example, if the PoI is for a hotel, the tag can be a textual review about the hotel's customer reviews, etc.

6) Location Object: This object has 3D location information about a certain PoI.

7) Database Object: All user profiles, PoI tags, Rules, PoI locations and orientations are stored in the database object

8) Route Object: The route object contains detailed directions from a source PoI to a destination PoI. It also has information about the route details including its associated obstacles and PoIs.

9) Rule-Based Expert System: This system analyzes the users' tags and PoIs and provides consultation and advice for the user. The user interacts with the Rule-Based expert system through an interface especially designed for this purpose. The user experience is stored in the knowledge object in the form of a set of rules. Sample rules can be of the form:

"*Avoid routes having obstacles when navigating from point A to point B"*.

"Go from Point A to Point B and Pass by a pizzeria".

"Go to a hotel in downtown Chicago that has a review *score of 4 out of five or more"*.

Go to a coffee shop in downtown Kalamazoo that has at least 50 reviews about it".

"Follow the route with least traffic from point A to point B".

"Only use routes where traffic lights have a speech system installed on".

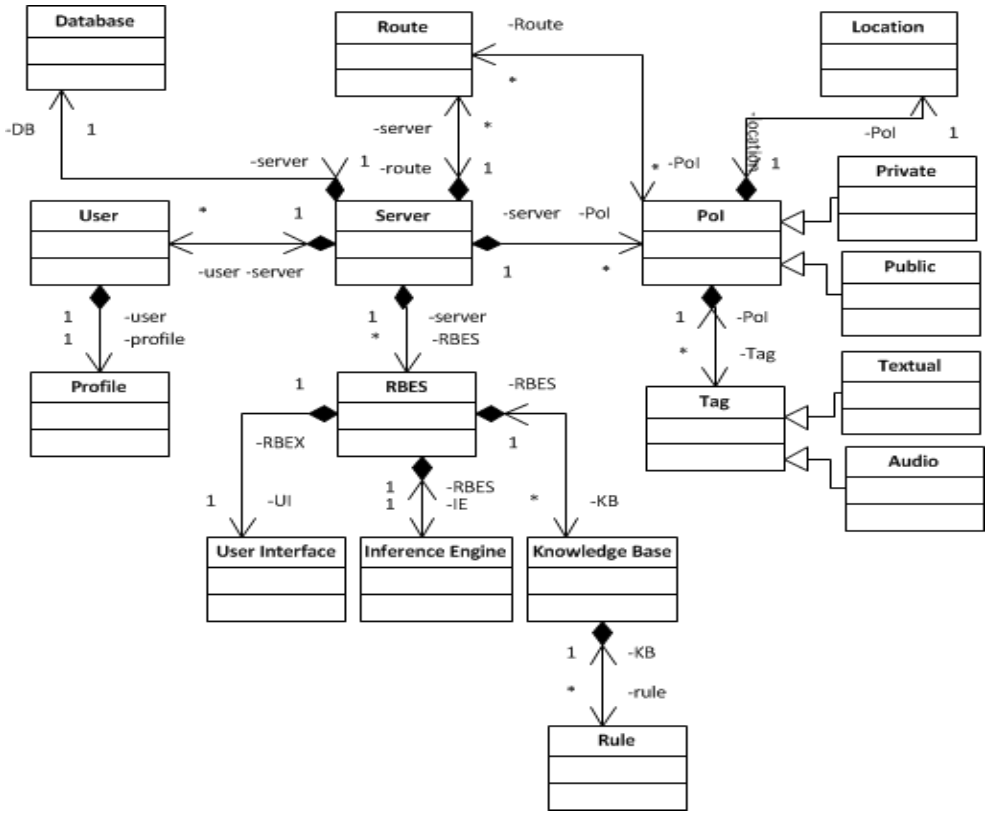

Figure 6.3: The social network class diagram.

6.3.4 Use Case Diagrams

This section describes the interactions between the system object in a form of use case diagrams.

1) Account Use Case: In this use case, the user attempts to create an account on the system by providing a unique username and a password. The systems sends a confirmation email to verify the user and when the user confirms his/her email address, the systems create a profile object for the user and the user can login to the system. The use case diagram in Fig. 6.4 illustrates this operation.

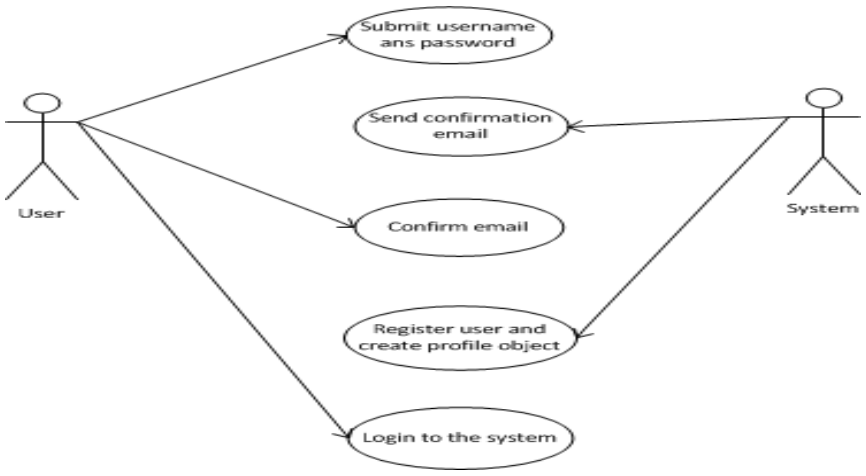

Figure 6.4: The account use case diagram.

2) Create PoI use case: This use case illustrates how PoI gets created bases on the user request. The user requests a use case creation from the system. The system then asks for the location information. Then, the user input all the information about the location and new PoI gets registered in the database. Fig. 6.5 shows an illustration of this process.

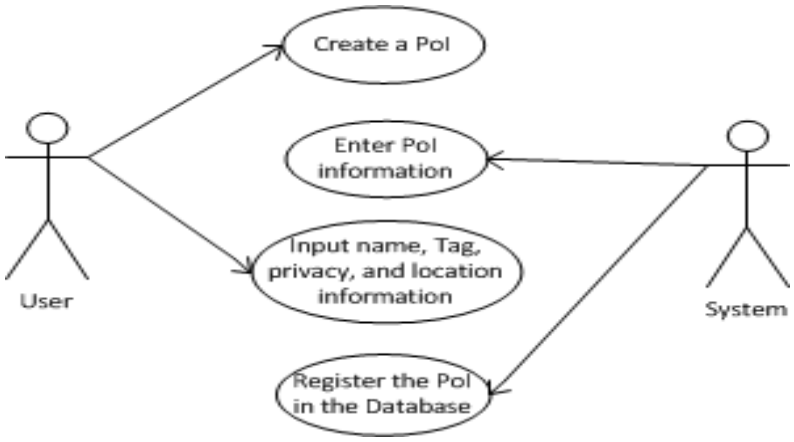

Figure 6.5: Tag creation use case.

3) Route use case: This use case illustrates how the system provides a route between a source and a destination provided by the user. The user first prompts the system to navigate from a source to a destination. The system then prompts the user to input the source and destination and any user requirements. The system analyzes the user input through its Rule-Based Expert System and provides the best route based on the user input. Fig. 6.6 shows an illustration of this use case.

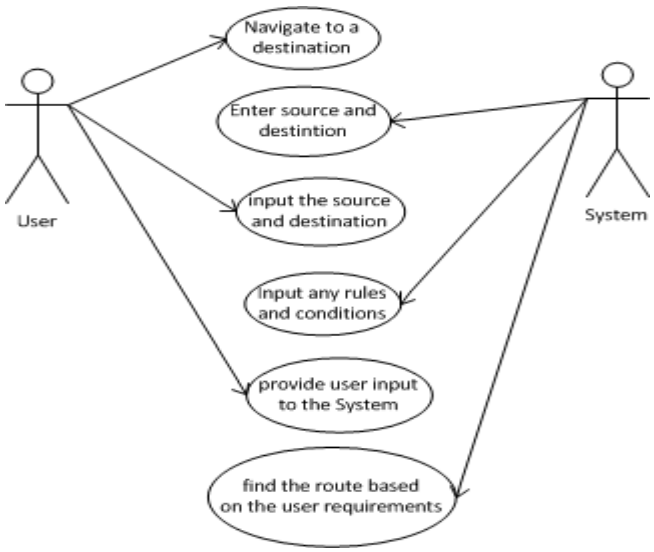

Figure 6.6: The route use case.

CHAPTER 7

SOFTWARE TESTING: PRINCIPLES AND TECHNIQUES

7.1 Introduction

Approximately 50% of software development time and cost is spent on testing [106]. The word test refers to the Latin word *testum* which is an earthen pot or vessel. This vessel was used to examine metal to check for the existence of various metals, thus the idiom "to put to the test". The concept of "*Software Testing*" arose concurrently with the first computer systems. Programs developed in the early stages of programming had to be tested and references to testing techniques return back to 1950s [107][108].

At the beginning time of software development, software testing was thought of as a follow on activity. After the completion of a computer program, the testing goal was not only discovering errors, but also correct them. In the earliest publications on testing, debugging was the main topic for these papers until 1957 where testing was clearly distinguished from debugging. In the last 1950s and 1960s testing became more and more important because a considerable amount of development budget was spent on correcting for program deficiencies.

7.2 Testing Definitions

Testing "*is the process of establishing confidence that a program or system does what it is supposed to"* [109].

Testing "*is the process of executing a program or system with the intent of finding errors"* [110] . This definition of testing is based on the belief that a computer program is initially assumed to have errors and the goal of testing is to find these errors. If the goal of testing was only to show that the program works, test data will be selected in a way that will have low probability of causing errors. On the other hand, if the intention was to prove that the program has errors, test data will be more sophisticated and test will be more successful in general.

Testing "*is any activity aimed at evaluating an attribute or capability of a program or system and determining that it meets its required results"* [111].

From the above definitions of software testing, the main goal of software testing is to find errors in the software and verify that the software is doing what it is intended to do.

7.3 Principles of Software Testing

Several basic testing principles can be extracted from the existing sufficient software testing experience. These principles help provide a foundation for testing methods and techniques that will be discussed later in this chapter. They also help understand what testing is all about and how it should be done to save time and money. In this section we summarize the most important principles in software testing and verification.

Principle 1: Complete testing is not possible

Even the simplest programs has practically and infinite number of test cases. It is impossible to try all these test cases and consequently verify program correctness. The tester selects test cases that hopefully will be representative to the test data space. The selected test cases should be big enough to measure the system's performance but it also should not be too big and make it difficult to manage the test process [111].

Principle 2: Testing must be done by different persons at different levels

Several factors decide who will perform system testing. These factors are the development methodology used, risks, the size and the context of the system and the skill and the experience of the testers. It's usually the responsibility of the component developer to perform component testing, but system/subsystem testing should be performed by independent person or team. Testers should be assisted by development staff before performing acceptance testing which is performed by the end user [112].

Principle 3: Testing is context dependant

The context determines what testing should be performed at different points in time. For example, an e-commerce website is testing differently from a safety-critical software application. Furthermore, systems developed using agile development approach is tested differently than systems developed using the waterfall approach for system development. The objectives of testing are also different at different points in time of the software development. For example, the objective of unit and component

testing it to verify that the code is working correctly. However, the objective of system testing is to verify that the system is doing what it is intended to do [112].

Principle 4: Testing Work Is Creative and Difficult

Anyone who had enough experience in testing work knows that performing software testing is not an easy task. Complete understanding of the system purpose is required before the testing process starts. Usually, senior business analysts are assigned to insure that the system is doing what it is intended to as they should understand the complex systems and their interactions within the enterprise. Asking untrained testers to perform system testing is exactly like asking parents to test their children on what they have learned in school. Hence, experienced and knowledgeable testers should be assigned to perform testing [111].

Principle 5: Testing is Risk-Based

This principle is concerned about how much testing should be done if the risk of system failure or finding a defect is not negligible? On the other hand, how much testing should be done is the risk of failure can cost a human life? The answer to these questions emphasizes that good testing is inherently risk-based. The amount of testing that should be done is dependent on the amount of risk involved in the system. If the system involves too much risk, many test well designed test cases should be designed for testing [111].

Principle 6: Design Effective Test cases

System requirements should be complete and precise to perform effective testing. User requirements should be well known before test case design and testing

should be performed against those requirements. Full understanding of system architecture and requirements is important. It allows designing test cases that will discover deficiencies in short amount of time. Test cases must provide a complete description of the input data and the expected output data [112].

Principle 7: Testing should be start early in the development

Finding errors at the early stages of system development saves huge amount of money and rework. According to [112], finding a problem after product delivery costs 10 – 100 times more than if it was detected at the requirement gathering process. Fig. 7.1 shows the cost of finding errors at different stages of development [112].

Principle 8: Testing must be planned

All software developers agree with this principle, but yet most of them do not discipline themselves to act on it. Appropriate testing requires building an overall approach, designing tests, and establishing expected results for the chosen test cases. Since a complete testing is impossible (Principle 1), a representative test data distinguishes between good and poor testing. A test plan document describes the testing objectives and the overall testing approach. A test design document describes which system components are to be tested and describes the expected test results. Test plans and designs can be developed for any stage of software development. Testing few inputs is not considered testing. The purpose of testing is to measure the software quality. Unplanned or Ad hoc testing can be harmful and may lead to a false sense of security [111].

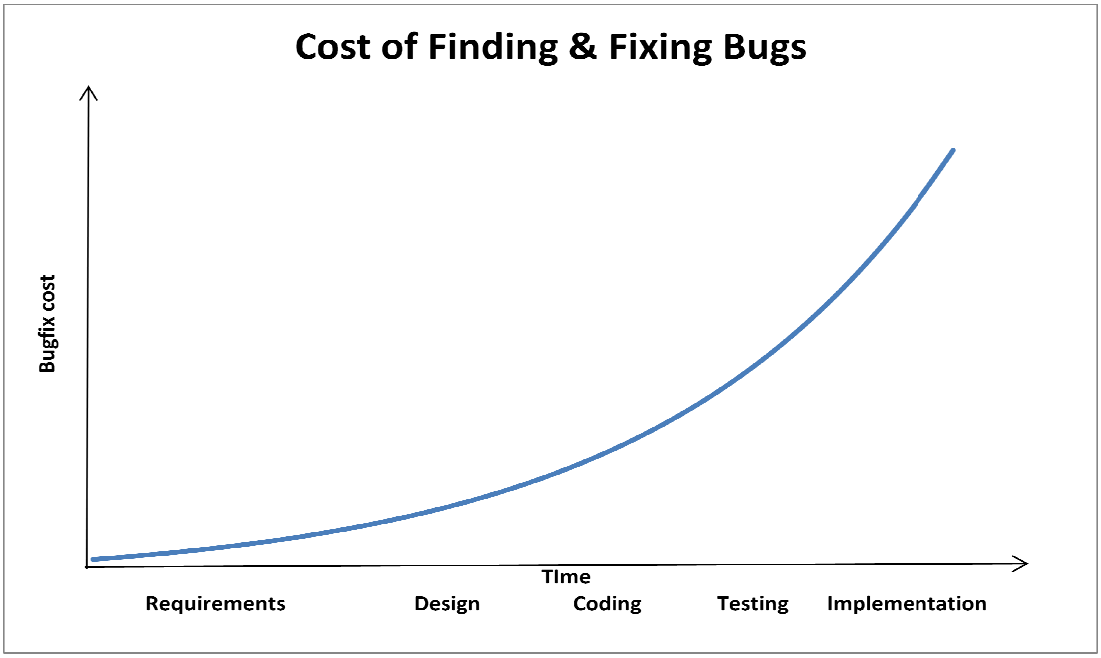

Figure 7.1: Cost of finding bugs at different stages of software development.

Principle 9: Testing Requires Independence

Unbiased measurements require unbiased person to perform the measurement process. Some organizations achieve independence by creating independent quality control units that report outside of the development function. Other organizations assign an outside "verification and validation" contractor that is responsible for system testing in achieve independence. Having a group of people from outside the development team to perform testing is not always necessary. Independence can also be achieved by team approaches and contentious reviews throughout the software development process [111].

Principle 10: End of testing

Testing is an ongoing process that has to be stopped at some point in time. Practically, testing is a trade-off between budget, time and quality. Effort spent on software testing is correlated with the consequences of software errors [113]. Testing can be stopped when the risk is under acceptable boundaries or for budgetary/scheduling limitations [112].

7.4 Software Testing Requirements

A successful testing process completely depends on building a complete and robust system requirements document. Software requirements are the conditions and constraints that the system must abide by and are usually broken into functional and non-functional requirements. The requirements document is basically a "problem definition". From testing point of view, the testers need to verify that this document is properly formulated before testing starts. So, in order to verify that the developed system works, it is important to test against these requirements to prove that the system meets its requirements contract. In this section we will deal with software requirements from a testing perspective [114].

The knowledge of how a requirement will be tested helps focusing on the real meaning of the requirement and makes it easy to correctly formulate the requirement. To achieve a good statement of the requirements, each requirement should be associated with a test solution along with it. Consequently, all problems associated

with misunderstanding the requirements disappear [111]. Furthermore, poor requirements document can cause the whole project to fail. This is a result of a constantly changing requirements because the stakeholder themselves they don't have a clear understanding of what they want the system to do [114].

Having a poor requirements document makes it difficult to prove that the developed system works or not when it's complete. Since establishing this proof is the responsibility of testers, they have to make sure that the requirements are good and clear. Otherwise, it will be very difficult to understand the expected results of the system or even develop a global understanding of what the system is doing [114]. Usually, requirements are not completely defined until they are implemented. Hence, the implementer's interpretation of the requirements becomes the final definition. This causes testers to periodically adjust their test cases at the last minute and stakeholders will more likely not accept the final product. Some aspects of "good requirements" are [114]:

1) Clear (Understandable): Everybody can understand them. Stakeholders, implementers and testers should have the same understanding of the requirements document. The clearness problem can avoided if all disciplines are involved in the requirements review process.

2) Complete and reasonably detailed: complete requirements help testers to make sure that everything that needs to be tested is being tested. Meanwhile, it is necessary to have enough but not too much detail to avoid misunderstanding of the

requirements. This can be achieved by referencing supporting details as external documents.

3) Verifiable (Testable): This means that a test case can be written against that can verify whether the requirement has been implemented correctly or not.

4) Unambiguous: A requirement is unambiguous if all of its readers understand it in the same way. On the other hand, a requirement should not be so non-ambiguous in a way that makes it a legal document.

Testers should be involved early in the requirement gathering stage. If this was not possible, they should at least get involved in the requirements review process. Once the tester team receives the requirements document, they start writing test cases for these requirements and it quickly becomes apparent to them whether the requirements are good or not. If it is possible for testers to ask for changes in the requirements, they immediately should ask managers for requirement revisions. This helps having clear vision of what the final shape of the requirement will be and it will be easier to write tests for the requirements [114].

On the other hand, if the requirements are base-lined and the tester is not allowed to ask for requirement changes, the tester can make informal changes and document them. The tester can write his own interpretation of the questionable requirements and discuss that with manager and stakeholders. Based on that, the tester can start writing test to these questionable requirements and present execution results to manager and stake holders. This will stir discussions about these requirements and make interpretations clear to everybody [114].

A Requirements Validation Matrix (RVM) is used to effectively organize requirements. It insures that tests are specified for all requirements in a form of requirement versus test cases. Each requirement is listed with test cases or situations that are created to test it. The requirement can have multiple test cases associated with it and it is very rare to have a single test case to verify a single requirement. Furthermore, a test case associated with more than one requirement is a common thing in the RVM. The RVM has many benefits and those include [111]:

1) Ensures that all requirements are listed.

2) Identifies the tests linked to each requirement.

3) Facilitates the requirements review process.

4) Provided a simple technique to track the status of the test design and review.

Another important strategy for testing the requirements is by using test models or prototypes. This includes building a model system with no intention to use it but to test and confirm that true understanding of requirement is available for all disciplines. Test models are more beneficial when there is little understanding about requirements that it is essential to gain more experience with a working model.

Modeling can be considered as a part of the incremental development approach. Incremental development means that gathering requirements, designing, building and testing the system is done in parts. Rather than solving a total problem, the problem is divided into sub problems. Sub problem-subsystem is built and tested independently of others. This has the advantage that requirements for later increments

do not have to be defined prematurely and can be changed based on the work experience.

7.5 Software Testing Techniques

Software testing is an art and cannot be considered as science [115]. Techniques invented 20-30 years ago are still being used in today's testing. Some of these techniques are crafted methods or even heuristics rather than well-engineered testing techniques. Software testing is not an easy process at all. A piece of software can never be considered as correct and the same applies on the software specification. There is no verification system available that can verify the correctness of every program and it is even impossible to verify the correctness of the verification program itself [115].

7.5.1 General Testing Techniques

There are various testing techniques serving multiple purposes during the software life cycle. Those techniques can be classified based on purpose in several categories:

1) **White Box Testing**: This is a complementary testing technique that relies on the knowledge the analyst has for the internal structure of the software. In white box testing, the analyst has complete knowledge of the structure of the system being tested. For example, if the analyst is designing a test case for a function that checks if a certain input is of integer type, the analyst is aware that the function will be of type IF-THEN-ELSE and can make sure that the appropriate logic is implemented. In

white box testing the analyst has full access to all software design documents and other sources of knowledge about the software being tested [116].

2) Black Box Testing: In this technique, the analyst does not have knowledge about the internal structure of the software. Example software can be an extension to legacy software where system documentation is lost or Commercial-Off-The-Shelf (COTS) software. In black box testing, test cases must be designed for the external behavior of the system. If the software requirements are available, those must be test against. Otherwise, user guides or any document that describes how the software should behave can be used in testing [116].

3) Correctness Testing: The minimum requirement of any software is to be correct. This technique requires an oracle to tell the wrong behavior of the system from the correct one. In correctness testing the tester may or may not know the internal details of the system to be tested. Hence, either black box or white box testing can be used in testing [115].

4) Performance Testing: This technique involves all the phases in the mainstream of the software lifecycle. Performance testing insures that the software under testing complies with the performance requirements of the system. The performance of any system is usually measured by throughput, resource usage and stimulus response time[117].

7.5.2 Functional Testing Techniques

The previous techniques are considered general techniques and are used in most software testing approaches. The authors in [116] classified testing techniques as functional and non-functional techniques. Functional techniques are used to verify that the Application Under Test (AUT) meets all functional requirements listed in the requirements document. The major functional testing techniques are:

1) Equivalence Partitioning: This technique divides the input data to a software module into partitions from which test cases can be derived. Each partition has at least one test case designed especially to uncover certain classes of error. This reduces the number of test cases that should be developed. Another advantage of using this technique is that it discovers the so-called "*Dirty*" test cases. An inexperienced tester may ignore the use of invalid data during testing which results in huge number on unnecessary test cases. It is necessary to mention that equivalence partitioning is not enough to derive test cases and should be supplemented by boundary value analysis which is the next functional testing technique [118].

2) Boundary Value Analysis: Since errors tend to occur near the extreme values of the input variables, test cases should be designed to test for these conditions. For example, loop conditions may test for a "<" when they should test for a "≤" and loop counters are usually off by 1. This technique works well when the UAT is a function of independent variables representing bounded physical quantities. The limitation of this techniques is that is does not consider the nature of the function or the semantic

meaning of the input variables. Test cases belonging to this technique tend to be elementary with no insight and imagination [119].

3) Intrusive Testing: Using this technique, the AUT is intentionally modified by the tester for the purposes of testing. For example, some variable values that are not visible during the execution of the program are made visible by adding statements to view these variables' values. Other example is that the tester deliberately sets variable values using symbolic debugger or intentionally triggers error conditions for testing purposes. The modifications made by the testers should not be delivered to the user which requires careful change control and management. The main objection on using this technique for testing is the argument that what you tested is not what you deliver to the user. Furthermore, some of systems tested using this techniques contained tester modifications that manifested themselves during normal system execution [116].

4) Random Testing: The purpose of this technique is to generate so many test cases to hopefully uncover as many faults or hit many coverage targets. For instance, the test cases are continuously sampled until certain number of feasible branches in the AUT is executed. Since test cases detect failures and do not uncover faults, from a mathematical point of view faults cannot be considered as targets. The target is to make specific system expectations fails, thus triggering visible failures [120].

5) State Transition Analysis: This technique is used where some parts of the AUT can be represented as Finite State Machine (FSM). This means that at any point in time the system is in a known state and the transitions between system states are

governed by machine rules. The state transition model of this technique has four main parts [121] :

1) System states.

2) State transitions.

3) Events that trigger transitions between states.

4) Actions that result from state transitions.

Evaluating what have been tested using this technique is a white box approach and developing test cases for the state transition model is a black box approach. It is possible to design test cases for every state transition in the model which is known as "0-Switch" coverage. It is also possible to develop test cases for every pair of state transitions in the so-called "1-Switch" coverage. Deriving test cases from the state transition only may result in neglecting the negative test which represents the invalid state transitions [121].

6) Static Testing: In this testing technique, the AUT is not actually used. This approach only checks the sanity of the code. It also involves syntax checking of the code and manually reviewing the code for errors. From a black box point of view, this approach involves reviewing the requirement and specification documents [11] .

7.5.3 Non-Functional Testing Techniques

The last category of testing techniques is the Non-functional testing techniques. The goal of these techniques is to insure that the application under testing meets all of it non-functional requirements. These techniques include:

1) Installation/Configuration Testing: The goal of this testing technique is not to find application errors but to find installation errors. Test cases designed using this approach to insure that [122]:

1) The compatible set of system options is selected.

2) All parts of the system exist before installation.

3) Interconnections between all programs are all set.

4) All files are created and have their content.

5) The hardware configuration is appropriate.

Installation test cases are formulated by analyzing the test plan after being designed by analyzing the system external specification. This testing approach should be performed by the application development team which is familiar with both the software and the available hardware specifications [122].

2) Performance Testing: This technique validates the scalability, speed and stability of the AUT. Performance related issues like testing and tuning are considered to achieve response time, throughput and resource localization requirements of the AUT. This approach to software testing is useful to uncover the performance bottlenecks in high use applications. Performance testing allows involves an automated test suits allowing for simulation of normal and exceptional workloads [123].

3) Stress Testing: The AUT is stressed by applying more than the maximum allowed loads, by offering a certain amount of loads in a very short period of time or by offering no loads to the AUT at all [124]. The main goal of stress testing is to verify

that when the system fails it recovers in a graceful way; this is called the recoverability of the system [116].

For example, if a system is designed to accept a maximum of thirty simultaneous users. A stress testing technique may check what happens when the thirty first user tries to login to the system? Or what happens if the thirty users login to the system at the same time? Simulation is the best tool to perform stress testing. In the previous example, it is hard to have thirty real users logging in the system at the same time from thirty different terminals. In general, developing test cases for stress testing is time and resource consuming but this helps taking the critical decision of whether to put the AUT live or not [116].

4) Recovery Testing: Operating systems and database management systems usually have system recovery objectives that states how the system should recover from programming errors and disk failures. The goal of this testing technique is to show that recovery functions do not work properly. Programming errors are intentionally injected in the application to show how it reacts to failures due to these errors. Simulation can be used to simulate hardware failures like parity errors, device I/O errors or even a noise in a communication link [110] .

The design goal of such systems is minimizing the Mean Time to Recovery (MTTR) which is the time the application takes to recover from failure. The objective of this testing technique is also showing that the system does not meet the service level agreement of the MTTR which always have lower and upper boundaries that should be considered in the design of test cases [110] .

5) Security Testing: The role of the AUT determines the level of security testing that should be performed. Some of the requirements specify the need to insure the integrity, availability and confidentiality of the application and data. The goal of security testing is to test whether the features implemented in the software provide the required level of protection. Security testing is often conducted by a dedicated team whose role is to make sure that the AUT conforms to all the security requirements of the AUT. This team may also verify that a standard or formal process was followed during the development of the application. It is possible to run security test during any phase of software testing but they are usually run in system testing and acceptance testing phases [116].

6) Volume Testing: In this technique the AUT is subjected to heavy volumes of data. For example, a compiler application can be tested by feeding it a huge program to compile. Also, a linker/loader can be tested by feeding it with a program having thousands of modules. An operating system job queue can be fed with full capacity number of jobs. The goal of volume testing is to prove that the AUT cannot handle the volumes of data specified in the objectives. Since volume testing is known to consume significant resources, time and workforce, it is not possible to go overboard. However, each application should be tested with volumes of data in a certain way [110].

7.6 The V-Model in Software Testing

The V-model is a software development process that was first found by Paul Rock [125]. The V-model can be considered as an extension to the water fall model of software engineering [11]. Instead of progressing in a learner way, the software process is bent upward forming a v-shape. This model reveals the relationships between the software development phases and their corresponding testing phases. The goal of this model is to improve the effectiveness and the efficiency of the software development process and to demonstrate the relationships between development and test activates as shown in Fig. 7.2 [11, 126].

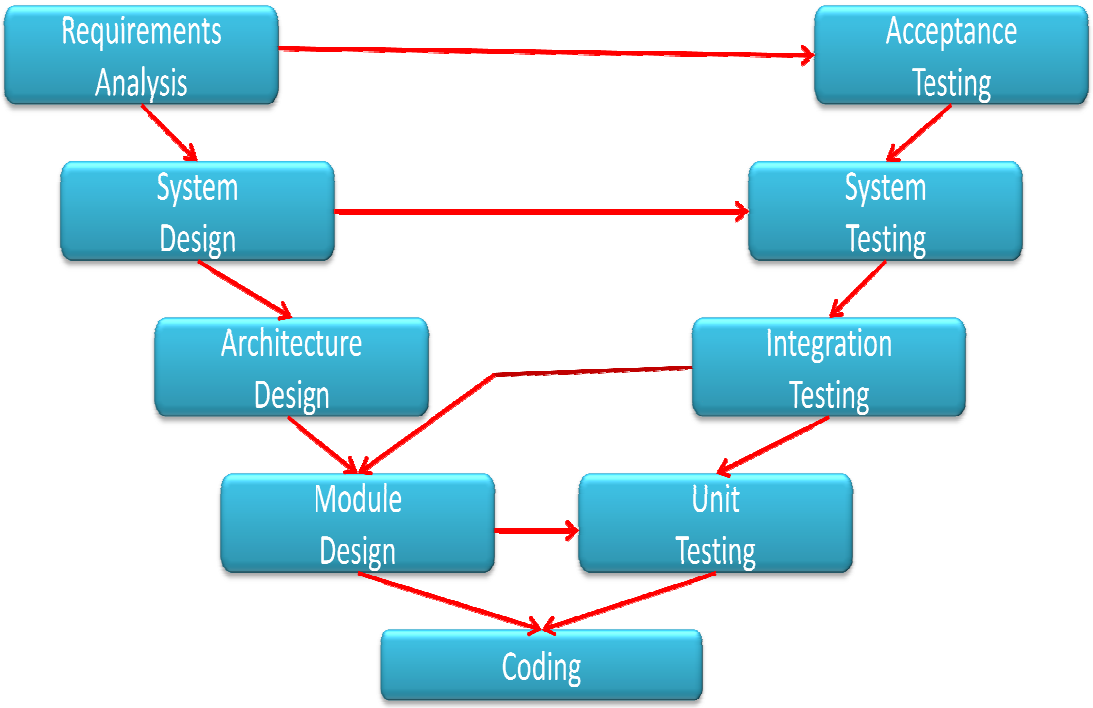

Figure 7.2: The V-model.

In the V-model, testing begins as early as possible in the development process. It is always important to involve testers in the early phases of the software life cycle. There are a lot of test activities that should be carried out before ending the coding phase. These activities run concurrently with the development activities to enable the testes produce some test deliverables. The model demonstrates that software verification and validation can be integrated in each phase of the product life cycle.

There are many variants of the V-Model in the literature [11, 125, 127-130] . In this section, we will discuss a common type of the V-model which has four main testing levels namely; Unit Testing, Integration Testing, System Testing and Acceptance Testing.

7.6.1 Unit Testing

Unit testing is the first level of testing and it refers to testing each program unit separately. Examples of system units are procedures, functions or methods. Classes in Object-Oriented Programming can be also considered as application units. Syntactically, a program unit is a piece of code that that is called from outside the unit and can also call or invoke other units. Furthermore, a unit is known to implement well-defined functionality at a low or high level of abstraction [131].

It is natural to test a program unit separately before integrating it with other program units. This makes it easy to attribute errors found during testing to a specific unit so that is can be easily fixed. This also allows for verifying that each distinct

execution of a program unit provides the expected results. All possible unit execution paths should be considered while performing unit testing. This requires cautious selection of input data for each execution of the program unit [131].

A programmer needs to verify that a code works or not when performing unit testing because unit testing has inherent limited scope. Intuitively, a program unit is tested in the following ways [131]:

1) Every line of code should be executed. The programmer should observe the behavior of the program unit when each line of code is executed.

2) Every predicate in the unit should be executed and evaluated as true or false.

3) The program unit should perform its intended function without any known errors.

Because of the limitations of testing in isolation, not all the expected functionalities of a program unit can be tested in isolation. Hence, there is no guarantee that the tested unit will work correctly from a system-wide perspective. This means that some of the errors will be found later when integrating the unit with other system units during integration and system testing stage. Although it is not possible to find all errors during unit testing, it is always necessary to make sure that a unit functions satisfactorily before integrating it with other units. The reason for that is it will be a waste of time and resources to perform testing in subsequent test stages and it will be harder to find the root cause of error [131].

Unit testing insures that all parts of the system works correctly before finally integrating them together. Although this is considered as a good idea by many researchers, unit testing still not widely and formally performed. One of the main reasons for this is the lack of effort and commitment to unit testing. Reusable test cases for units are hard to build and if the unit code changes, this might require changing the unit test cases as well. Another reason is the lack of software management support for unit testing. Unit testing is not considered as a real deliverable in a project. Hence, a program tester might divert his attention to more superficial deliverables of a project plan [132].

7.6.2 Integration Testing

Integration testing is the process of checking how system components work together especially at unit interfaces. The purpose of integration testing is to make sure that different program units interact and communicate data among each other and function consistently. Integration testing can be performed at different levels. The lowest level is to test that all program units work together with no errors. Higher levels of testing might be performed by the developer of the test team [113].

The integration test plan is considered as a road map for developers in most organizations. It allows the test team to test how individual parts works together and to make sure that what the group is testing does not overlap with what other groups are testing. Many companies deliver well tested systems without performing

integration testing. This is done by performing an extended unit testing or well performed system testing [113].

Integration testing can be very important when developing large and complex systems. Some bugs are impossible to discover during unit testing and are discovered by integration testing. The integration testing targets the interfaces between program units since they constitute the error-prone parts of the system. It is common to start the test plan as soon as possible. This means that the integration testing should begin as soon as the system design is beginning to stabilize [113].

System developers play an important role in performing integration testing. Developers have no way of knowing that the system they developed is viable until they participate in integration testing. Having the developers involved in integration testing speeds up the process of finding bugs and correcting for them. When performing integration testing, it is always necessary to creating a scaffolding code (drivers and stubs). The test team member may or may not have the required skill to make such a scaffold code [113].

The testing order in integration testing can be performed by four different strategies [133]:

1) Top Down: In this strategy, the interfaces located in the top layer of the system design hierarchy are tested first, followed by layers going downward in the design hierarchy. In this technique, the main programs servers are the driver for testing and in this technique a shell is quickly created. The disadvantage of this strategy is that it requires so many stubs to perform integration testing [133].

2) Bottom-Up: This strategy starts with lower interface in testing and higher components are replaced with drivers. This means that it might require too many drivers to perform testing. The advantage of this strategy is that it enables early integration with system hardware where this is relevant [133].

3) Functional Integration: In this strategy, the system is integrated by functionality area. This is a kind of a vertically divided top-down approach. This strategy makes it possible to have functional areas as quick as possible [133].

4) Big-Bang Integration: In this strategy, all parts of the system are integrated in one run. At first glance, this strategy seems to reduce the test effort but in reality it does not. It is very hard to find any defects in the interfaces and impossible to get proper coverage when testing the interface in a big bang approach. Both bottom-up and top-down end up in a big bang testing even if this was not the initial intention [133].

7.6.3 System Testing

System testing begins after the completion of integration testing. The goal of this test is finding defects in features of the system compared to how it was defined in the requirements document. Furthermore, this test aims to reach to a fully integrated functioning system. Efficient unit and integration system leads to more efficient system testing. System testing is usually impacted by poor or missing component in unit and integration testing [133].

System testing is still necessary after unit and integration testing because the later techniques test for techniques specifications; from the technical viewpoint of the

software developer. However, system testing is more biased to the perspective of the end user of the system. Furthermore, many system function result from the interaction of various system components and they are only visible at the global system level and can only be tested at this level [133, 134].

The specification of the system test is based on the requirements document specification. In this specification, both functional and non-functional system expectations should be expressed. The system functional requirements express what the system shall do and the non-functional requirements express how the system presents its functionality and behavior. The test techniques used are often among functional techniques and can be enhanced with experience-based techniques. However, it is recommended that experience-based techniques should never be the only techniques used in system testing [134].

The system test configuration should be designed to place the system under all of its operational inputs and environmental conditions. The sources of deciding which measurements are valuable should exist in system-level specifications. The main elements of system test configuration are [135]:

1) System Inputs and Environment: The system test configuration must include all the conditions that affect the system behavior like inputs and interactions with the environment. If any of these are impractical, simulation should be used to realistically represent their interactions with the system [135].

2) System Outputs and Test Points: All expected system output should be converted into measurable quantities and recorded during the test. These recordings

should also be performed for the inputs to enable the correlation between the variations in inputs and changes in outputs [135].

3) Test Conditions: The conditions under which the system will be placed during operational evaluation should be visualized and duplicated. This leads to successful operational evaluation by the end customer. Furthermore, some system parts maybe stresses intentionally to insure system robustness under severe conditions. This includes verifying the functionalities that guarantees system recovery to full system capabilities [135].

7.6.4 Acceptance Testing

Software acceptance is "*an accumulative process of approving or rejecting systems during development or maintenance stages according to how close the software is to predefined requirements"* [136]. The acceptance decisions verify that the documentation delivered is consistent with the system and that the final submitted system meets all the end user requirements. The final software acceptance must be done at the end of the software development process to insure that the delivered software meets predetermined functionality, quality and interface criteria. Security or safety criteria might be enforces legally or by the nature of the software system [136].

Software acceptance is described in a formal document during the early stages of the software development process. The plan lists the products for acceptance, the acceptance criteria, acceptance reviews, and the acceptance testing throughout the software development life cycle [136] . The software acceptance must state a certain

criteria that the product should meet in order to be accepted. The most important means to accept critical software like weapon systems or defense information network is a periodic review of temporary software documentation and other software products [137].

Final software acceptance testing is considered as the last chance for the end user to check the software for functional, safety, security, and quality requirements before finally accepting the software. At this stage, the software should include the delivered system with all of its documentation and versions. The final step in performing software acceptance testing is writing the review document which contains the software acceptance testing results [137].

The main goal of software testing is to pinpoint errors and find potential problems in the software. Initial analysis of the requirements document is important because requirements are the basis for acceptance testing criteria since they verify that the product will meet the user's needs. However, later acceptance testing activities do not focus on direct proof that the system will meet user expectations. These activities focus on the each successive products of development are consistent with each other and that all requirements will be met. If the earlier stages of acceptance testing complete successfully, then the final testing stage should be little more than a formality [137].

The acceptance criteria and a complete set of system requirements form the basis for the acceptance testing approach. The software system and the installation site affect how the acceptance testing will be performed. Some special arrangements

are required when the software system cannot be completely installed and tested live. Multiple configurations might need to be installed at different test sites and test cases might not be the same for all installation sites. Hence, configuration management of test documentation requires special attention. Planning in advance is required to make sure that the operations staff is well trained so that the acceptance testing of the software goes well without any interruptions [137].

The end users are involved in deciding how the acceptance testing will be performed, even if the acceptance testing is to be performed by a third party. The minimum requirement is that the staff of the end users is involved in the testing process. The end users should create scenarios of how they perform their functions and how the final software will be used. Furthermore, the end user must provide managerial time for the process of acceptance testing from its beginning. They also must allow tie for reviewing and developing the test documentation [137].

CHAPTER 8

SOFTWARE VERIFICATION OF RULE-BASED EXPERT SYSTEMS

8.1 Introduction

Problems like theorem proving, speech and pattern recognition, game playing (e.g. Backgammon and Chess) and many complex and stochastic problems were thought to only be solved by humans. The reason is that the formulations and solutions to these problems require human abilities like thinking, observing, memorizing and learning. However, intensive research in the last five decades or so shows that the majority of these problems can be formulated and solved by computers. The broad field which is referred to now as Artificial Intelligence (AI) deals with these kinds of problems which first seemed intractable and impossible to solve [138].

The Artificial Intelligence is defined by A, Barr and E. A. Feigenbaum [139] as:

> *"Artificial Intelligence is the part of computer science that is concerned with designing intelligent computer systems, that is, systems that exhibit the characteristics we associate with intelligence in human behavior- understanding language, learning, reasoning, solving problems and so on."*

Artificial Intelligence has several subareas including expert systems, automatic game playing, automatic theorem proving, pattern recognition, artificial

vision, robotic and neural networks. Although expert systems are treated as a subarea of AI, most of other areas in AI include an expert system part in their structure [138].

8.2 Expert System Definition

Expert systems have many definitions in the literature. For example, Stevens [140] defines an expert system as:

> *"Expert systems are machines that think and reason as an expert would in a particular domain. For example, a medical –diagnoses expert system would request as input the patients symptom's, test results, and other relevant facts; using these as pointers, it would search it database for information that might lead to the identification of the illness. […] A true expert system not only performs the traditional computer functions of handling large amounts of data, but it also manipulates that data so the output is a meaningful answer to a less fully specified question."*

The definition of expert systems evolved over the years due to the rapid technology development. M. Josephine and K. Sankara defined the expert system as [141]:

> *"An expert system is a computer program that simulates the judgment and behavior of a human or an organization that has expert knowledge and experience in a particular field. Typically, such a system contains a knowledge base containing accumulated*

experience and a set of rules for applying the knowledge base to each particular situation that is described to the program."

Based on these definitions, an expert system should have the ability to:

1) Process and memorize data.

2) Understand this data and reason it in both deterministic and un-deterministic situations.

3) Interact with humans and experts.

4) Make decisions based on this experience and also provide explanations on why these decisions have been made.

An expert system can be thought of as a consultant that can provide help to a human expert with certain degree of reliability [138].

8.3 Expert System Components

There are several taxonomies for the expert systems that are available in the literature. In general, an expert system has four main components:

1) Knowledge Base (KB): KB is considered as a storage place for the human expert knowledge to solve a certain problem in a certain area like engineering, finance medicine and so on. Knowledge can be in a form of heuristics which are based on experience and intuition. Therefore, it varies from one expert to another. Knowledge can be also described as "*deep knowledge*" which includes theories and principles obtained from textbooks [142].

2) Inference Engine: The inference engine is basically a program that includes an analytical framework for generating and explaining the advice to the user. The inference engine task is to combine user's answers to questions with the rules stored in the KB in order to provide an advice to the user or solve a certain problem. While doing that, the inference engine can also generate additional questions as required, and conclude based on the satisfied rules what are the final solutions and recommendations should be [143] .

3) User Interface (UI): The user interface is a software the allows the user interact with the computer by inputting data and accepting output from the computer. The UI provides a communication layer between the inference system and the user. Furthermore, the inference engine uses the UI to obtain facts from the users about the rules in the knowledge base [142, 143].

4) Working Memory: This component also referred to the as "*Blackboard*", represents the memory of the expert system. It includes the initial facts that are entered by the user and the fact inferred by the inference engine [144]. It also contains the plan of action for solving the problem in hand and the alternative actions that can be taken to solve the problem.

5) Explanation Module: The purpose of this module is to provide the user of justifications to the decision that are taken by the expert system. It also provides the user with the ability to request on how a certain conclusion was obtained. For example, the user of the expert system can request a rule trace or the set of rules what were executed until a decision or a conclusion is reached [144].

8.4 Rule Based Systems

A Rule-Based System (RBS) is a system that comprises knowledge base in a set of rules in the form of "IF-THEN-ELSE" and a simple inference mechanism or a rule interpreter. The interpreter examines the facts on the left hand side of the rule. A successful examination of these rules sets the facts to new values based on the right hand side of the rule. The database of facts is affected by the actions of the rules and rule firing depends on the fact stored in the database [145].

In a RBS knowledge is represented by a set of rules. That is, a set of conditional statements relating facts to one another. The primary rule of inference that systems use to add new facts to the knowledge base is called "*Modus Ponens*" [146]. This rule works as follows:

Suppose P and Q are both logical statements and assume that P and "P then Q" are both true. Hence, we can infer that B is also true. An example of inference using the modus ponens can be described by the following three sentences [147]:

- *If it is raining, then there are clouds in the sky (P → Q)*
- *It is raining (P)*
- *Therefore, there must be clouds in the sky (Q).*

OR

P

P→Q

_ _ _ _ _

Q

The Rule-Based system frame work is conceptually simple. The variations required to interact with the real word makes this framework interestingly complex. For instance, the rule $P \rightarrow Q$ is often interpreted as "*P suggests Q*". The fundamental assumption in a Rule-Based system is "*Knowledge is a power*". The general problem solving knowledge when merged with specific knowledge of task provides an expert level analysis of difficult situations [146].

Heuristics, or rules of thumb, have been used by Artificial Intelligence (AI) researchers in intelligent problem solving. The reason for that is mathematically precise and computationally feasible methods are knows for a relatively few problem classes. The most important information that should be available to the expert or Rule-Based system is the body of heuristics that experts use to solve hard problems. An expert system can also use informal knowledge reasoning in problem solving. The goal of the expert system is not to simulate a specialist problem solving behavior. However, the power of the expert system is derived from integrating the heuristic knowledge as specialists use with the same style of informal reasoning [146].

The pioneers of experts systems are DENDRAL [148] in organic chemistry and MACSYMA[149] in symbolic integration. These systems were built in 1660's and their emphasis was on system performance. They also were unique systems in AI at that time since they targeted real world problems on specialized knowledge. Serious work in expert systems started in 1970's especially in the field of medicine. The systems in [150-153] are examples of the first expert systems. Later systems [154, 155] added the explanation capabilities to the Rule-Based expert system making

the system provide explanations for the recommendations provided to the user. Furthermore, the flexibility in acquiring knowledge was added in newer expert systems like the systems in [146] [156, 157] .

The construction of a new expert systems was difficult and time consuming because each system was custom-crafted [146]. The major difficulty was in acquiring expert's knowledge and converting it into a form that can be consumed by the computer. This process was later known as knowledge engineering. The major contribution of the researchers work in 1980's was the development of "*knowledge engineering frameworks*". These frameworks were constructed to help build, debug, interpret and explain new expert systems. The process of engineering an expert knowledge into a useful computer program is a difficult task and thus, computer aid to the system builder is necessary. Hence systems like EMYCIN [158], ROSIE [159], EXPERT [160], OPS [161] and KAS [162] were built to help developers building robust expert systems [146].

8.4.1 Rule-Based Systems Verification and Validation

A Rule-Based system can be verified using two approaches [163]. The first approach considers the system as a one unit while the second approach deals with each system unit separately. Since the modern Rule-Based systems are modular in design, a combination of the two verification approaches can be used. The integrity of the Rule-Based system can be maintained by following the accepted software engineering techniques. These techniques include performing unit, module and

system level testing. The work in [164] identifies the tests that are applied to the expert system knowledge base to verify its correctness, consistency and completeness. During this phase of testing, online debugging tools can be involved in system testing [165].

The goal of the testing process if to verify that the functionality of the system developed meets the expectations of the end user. This work is often performed by the development personnel using the review document of the system specification and documentation. Beta-testing can be performed of the prototypes through the initial implementation of the Rule-Based expert system. The outcome of this process can cause reformulation of the objects and heuristics and can also result in redesigning the system knowledge structure [165].

The common way of testing a Rule-Based system is by providing a number of test cases with known results and measuring how the rule-base performs against these test cases. The drawback of using this method is that it is difficult to obtain an indication of how much the rule-base is actually exercised by test cases. Furthermore, it is possible that there are some rules never get fired by these test cases at all. Hence, the accuracy of the test applies only to part of the rule base [166].

The work in [166] implemented a Rule-Based verification and validation tool called TRUBAC. This tool proved to enhance the typical functional evaluation based on a series of base coverage measures. The work was motivated by the control flow analysis and data flow that have been applied to procedural programs in the past. TRUBAC uses a set of Rule-Based measures to evaluate the rule base.

O'Keefe and O'Leary [167] and Boehm [168] developed an approach that defines verification as" *building the system right and validation as building the right system"*. Hence, verification tests whether the system follows its formal specification. Their work focuses on validation issues and provided methodologies to prove whether the system provides its expected functionalities in the eyes of experts and users as well [169].

The Verification and Validation (V&V) of Rule-Based system was proposed in many approaches [170-173]. These approaches focus on detecting structural abnormalities between rules and on the completeness system to check whether all ranges of a condition are checked or not. None of the proposed approaches verify the accuracy of the system rules. On the other hand, verification methods are concerned with constructing suitable test cases for a given Rule-Based system [174].

8.4.2 Rule Accuracy Measurement

Rule accuracy is concerned with how a given rule accurately represents the expert knowledge encoded inside the rule. The absence of accurate methods to structure and formulate the rules may result in inaccurate expert knowledge representation by the rules. An accurately coded rule must represent its part of the domain knowledge and nothing else. If it is possible to verify the rule against the knowledge that is represents, then it is possible to evaluate whether the rule correctly models its intended part of knowledge [174].

Rules apply on objects in a finite universe. A rule selects objects from this universe based on the rule's conditions. A correctly coded rule will select the correct objects in its actions and conditions. Hence, if it is possible to verify that the rule selects the correct objects in both actions and conditions with respect to the knowledge that the rule represents, then the rule accuracy can be safely verified [174].

For example, a rule R in a rule-bases knowledge system can be defined as [174]:

- If there is a patient X, and
- X has dry cough, and
- X has high temperature, and
- X has headache,
- Then assert that X has influenza.

Assume the knowledge K that is to be represented is: *Patients having dry cough, high temperature, muscular paint and headache suffer from influenza*. It is clear that R is not accurate with respect to the knowledge K. If the patient universe was the one represented in table 1, then the rule R would select patients {X2, X3}. However, knowledge K implies selecting only {X1}. In this case, R violates the condition for accuracy which is whether the rule represents part or all of the intended knowledge [174].

Table 8-1: A sample patient universe.

Patient	Dry Cough	Headache	High Temperature	Muscular Pain
X1	FALSE	TRUE	TRUE	FALSE
X2	TRUE	TRUE	TRUE	TRUE
X3	TRUE	TRUE	TRUE	FALSE
X4	FALSE	FALSE	TRUE	TRUE
X5	TRUE	FALSE	TRUE	FALSE

8.4.3 Coverage Analysis and Rule-Based Systems

Coverage analysis is one of various techniques that are utilized in the validation and verification of software systems. Coverage analysis is a part of dynamic analysis which presents a structured and well-defined approach to exercise a software system. It requires the construction of test cases that are comprehensive to reveal the correctness of software. The development of these test cases is done using structural analysis (White Box Testing) or using functional analysis (Black Box Testing) [175].

Coverage analysis provides a measure to the completeness of the software testing process. It provides an indication to how deeply the software was tested and how effective the test cases have been. The general objective of software verification is to make sure that all statements in the software are executed at least once. An executed statement is considered as tested [175].

Test coverage analyzers or dynamic analyzers [111, 176, 177] are instrumented in the software during compilation or execution to monitor its

execution. These tools then use the information provided by the instrumented statements to record which software parts were exercised by the test cases. Test coverage analyzers also provide information about which parts of the software were not exercised and the percentage of coverage. This information is used by the testing team to determine the effectiveness of the set of test cases [175].

A demonstration prototype called SAVE (Suzanne and Abe's Validator for Expert Systems) [178] was developed to demonstrate the feasibility of constructing tools for the verification and validation of Rule-Based systems. This tool is equipped with a coverage analysis tool to verify and validate the knowledge base. The percentage of coverage of rules within the knowledge base and the effectiveness of each rule is analyzed using SAVE. Using the available commercial knowledge systems allows the Rule-Based system developer to focus on knowledge testing instead of inference engine and user interface testing [175].

8.4.4 Verification and Validation of Rule-Based Systems Challenges

The major difficulty in the verification and validation of expert systems is the lack of requirements specifications and acceptance criteria. The absence of a specification for the acceptance criteria makes the concept of acceptance unclear and subject to misinterpretation. Rule-Based system developers are usually reluctant to write a formal specification of user requirements. The reason for this is the ill-defined nature of the problems which the Rule-Based system usually tries to solve. Furthermore, the research environments at which experts system are developed and

the small size projects also cause the absence of acceptance criteria or requirements specification [178].

Unlike other process-oriented Artificial Intelligence systems, expert systems are problem oriented. Hence, expert systems are more amenable to the requirements specifications definition. Embedded Rule-Based systems are more accessible to requirements specification due to the exacting environment in which the operate in. categories of requirements are defined in current research to reflect the uniqueness of Rule-Based systems [178].

The type of results provided by the expert systems makes it even more difficult to verify and validate expert systems. It is easy to verify the correctness of conventional software because their results are exact and can only be correct or incorrect. However, it is hard to evaluate the results of an expert systems in the same exact manner. Validation of expert system is like grading an essay question as explained in [179]. The results of Rule-Based systems are usually complex and what makes system validation even more complex is that experts usually do not agree on the correctness evaluation of the produced results [178].

8.4.5 Recommended Activities in the Verification of Rule-Based Systems

In this section, we summarize a list of activities that can be done during the verification of Rule-Based System. These activities are illustrated in table 2 according to the linear model described in [178].

1) Planning Phase

A verification and validation plan that describes a comprehensive verification activity of the Rule-Based system being developed in is recommended in this phase. This plan is not completed, reviewed and base lined until the next phase starts [178].

Table 8-2: Recommended verification in linear model.

Planning	
Knowledge Definition	Source Identification & Selection
	Acquisition, Analysis and Extraction
Knowledge Design	Definition
	Detailed Design
Code & Checkout	
Knowledge Verification	Formal test
	Test Analysis
System Evaluation	

2) Knowledge Definition Phase

The output of this phase is a knowledge that has been verified, reviewed and base lined by the system experts. Since the knowledge is not encoded into rules until the next phase, the only verification that can be done at this phase is a manual examination of knowledge. This examination can be done using the techniques of walkthrough or inspection [178].

The test cases collection process is recommended to start during the knowledge definition phase. This process is coordinated and performed by the personnel responsible for the verification and validation of the Rule-Based system.

The tested cases can be collected from on-going transactions, real world data or from historical cases. End users and expert are the best sources for these test cases. The target usage for these test cases must be specified in advance and they should be maintained in a repository for this usage [178].

3) Knowledge Design Phase

The major activities in this phase are the generation of test plan and formal reviews of the design document. The verification and validation document, which is a product of the knowledge definition phase, is tested with respect to the design document. The design decisions during this phase can cause modifications and additions to this plan. Hence, the plan should be reviewed after these modifications [178].

4) Code Checkout

This phase includes the activities of documentation review produced in this phase, the formal review of test readiness, and code testing. The linear model does not specify the type of code testing that should be performed in this phase. However, the most common type of testing performed during this phase is the dynamic unit testing performed according to the test plan for unit testing [178].

5) Knowledge Verification

During this phase, the test functions listed in the test plan are implemented, the results of these tests are analyzed and documented, and the software under development is formally reviewed and base-lined. At this phase, knowledge base should be free from internal errors due to the previously performed testing activities.

Hence, the test objective is to verify that the system parts works well together as a whole or incrementally. Furthermore, knowledge implementation is examined in this phase (i.e. the expert system answers and the explanation module behavior). After the testing team is comfort that the system is working correctly, validation starts after that [178].

The goal of validation testing is to examine the level of expertise and the accuracy of the Rule-Based expert system. This provides an indication to the testing team whether the system is ready for validation by the end user or not. Using automated tool to compare the testing results is recommended during this phase. These tools provide statistical measures to the equivalency between the Rule-Based systems results and those obtained from human experts [178].

6) System Evaluation

The meaning of system evaluation in this phase is to determine the correctness of the Rule-Based system and validate it according to the user requirements. After performing all the testing activities in the previous phases, the testing team is satisfied with the functionality of the expert system and guaranteed that it meets the user expectations. Hence, the testing activities during this phase are mainly user acceptance testing activities [178].

The user acceptance validation uses the test cases maintained in the test repository to illustrate both the explicit and implicit requirements of the Rule-Based system. The user validation testing is commonly performed with the existence of the user or a representative of the user in the environment where the Rule-Based expert

system will be installed. The end-user should be given access to all the tools used for verification and validation during system development [178].

CHAPTER 9

CONCLUSION AND FUTURE WORK

9.1 Conclusion

In this work we presented a technique for gyroscope drift correction based on a TDoA technology; the MIT Cricket system. We used this technique to build a complete Inertial Navigation System (INS) comprised of a gyroscope, accelerometer and the MIT cricket motes. This INS was used in our indoor localization system at which we fuse data from RSSI measurements with the data received from our INS to perform precise location estimation. Furthermore, we used our INS to perform outdoor localization by fusing the data from our INS with signal phase shift measurements obtained from opportunistic AM radio signals.

The advantages of using TDoA over magnetometers for drift correction is that it is less susceptible to magnetic noise sources in the environment. The proposed approach was able to remove more than 85% of the inserted drift in the gyroscope signal. This enables the usage of the proposed drift correction approach along with the PDR techniques described in this paper in order to perform precise localization of pedestrians in both indoor and outdoor environments.

In this work, we proposed a novel dynamic access point exclusion technique for indoor localization. We also proposed a data fusion technique based on particle filter. The proposed approach fuses RSSI measurements received from nearby access points and data obtained from the INS subsystem.

The proposed approach takes advantage from the high distance accuracy provided by the MIT Cricket and avoids its LoS problems by proper installation of the beacon and listener. Significant location accuracy was achieved using PF which can be further enhanced if data from other sources like camera images or UWB were fused by the PF.

We also proposed a novel dynamic base station exclusion technique for outdoor localization. Furthermore, we proposed a data fusion technique based on the particle filter. The proposed approach fuses phase shifts from signals received from nearby base stations with data obtained from the INS subsystem.

9.2 Dissertation Contributions

Our proposed indoor/outdoor localization approach provides a solid framework to perform localization in both indoor and outdoor environments. While carrying out our proposed localization approach, the following contributions we successfully achieved:

1) We provided a novel gyroscope drift correction technique based on MIT Crickets instead of using a magnetometer for drift correction. Using a magnetometer for drift correction is very susceptible to noise from any device with magnetic field. We avoided this problem by using MIT Cricket motes with intelligent installation that avoids the LoS problems of MIT Crickets mores.

2) Our indoor localization approach provides the ability to locate any moving object in an indoor environment where there is no GPS coverage with sub-meter

location accuracy. The indoor localization approach makes use of the internet infrastructure that is available in any organization in these days and of low cost MEMS sensors to achieve precise indoor localization.

3) The usage of available signals of opportunity almost everywhere in outdoor environments allows our outdoor localization technique to precisely locate users in these environments where there is no Wi-Fi. The outdoor localization approach achieves high accuracy of less than one meter compared to the most popular outdoor localization tool which is GPS that has an accuracy of about ten meters.

4) All of the above contributions enabled the design of a comprehensive social network for blind and visually impaired. This group of people will be able to use our social network to navigate and share users' experience in their environments. Users will use our precise indoor/outdoor localization system to provide the network with their location. The Rule-Based Expert System installed on the network server provides the appropriate help for this group of people based on their specific requirements.

9.3 Future Work

The contributions mentioned in the previous section provide a solid basis for a significant and relevant future work. That includes:

1- Our outdoor localization approach makes use of the online available AM radio signals of opportunity to perform precise localization. While doing that, we run some optimizations online using PSO to solve for the distances and the position of the

user relative to the nearby LPAM radio base stations. This is done using Matlab and requires significant resources to be done in a real time system.

Our future plan for this involves introducing the concept of location fingerprints used in performing indoor localization into outdoor localization. We will create location fingerprints from the signal phase shifts obtained from multiple nearby LPAM radio base stations, record them in a database and use them in the same way as we did in indoor localization using the Particle Filter.

2) Our current implementation separates indoor and outdoor localization and both techniques works independently. We are planning to merge those two techniques into one approach that performs indoor/outdoor localization and make use of any available Wi-Fi or AM radio signals of opportunity or both. This new approach will still depend on our novel Inertial Navigation System (INS) and on the Particle Filter to perform data fusion between INS data, RSSI measurements and AM radio signals of opportunity all together.

3) There is some other senor data sources used in localization in the literature like RFIDs, heat sensors, Ultra Sound and camera images. We plan to fuse data from these sensor technologies using our adaptive particle filter to provide more localization accuracy and more awareness of the surrounding environment.

4) We designed a preliminary software architecture that provides a solid basis for our social network especially designed for the blind and visually impaired. This network will be developed based on the requirements of this group of people and will allow them to navigate through their environments with the least help from others.

The network will provide features like location tagging, obstacle avoidance routing and some other features based on requirements collected from the blind and visually impaired.

BIBLIOGRAPHY

[1] A. Klupper, *Location-Based Services: Fundamentals and Operation.* John Wiley, 2005.

[2] P. Bahl and V. N. Padmanabhan, "RADAR: An in-building RF-based user location and tracking system," in *INFOCOM 2000. Nineteenth Annual Joint Conference of the IEEE Computer and Communications Societies. Proceedings. IEEE,* 2000, pp. 775-784 vol.2.

[3] W. Hui, H. Lenz, A. Szabo, J. Bamberger and U. D. Hanebeck, "WLAN-based pedestrian tracking using particle filters and low-cost MEMS sensors," in *Positioning, Navigation and Communication, 2007. WPNC '07. 4th Workshop on,* 2007, pp. 1-7.

[4] C. Wong, R. Klukas and G. G. Messier, "Using WLAN infrastructure for angle-of-arrival indoor user location," in *Vehicular Technology Conference, 2008. VTC 2008-Fall. IEEE 68th,* 2008, pp. 1.

[5] S. J. Ingram, D. Harmer and M. Quinlan, "UltraWideBand indoor positioning systems and their use in emergencies," in *Position Location and Navigation Symposium, 2004. PLANS 2004,* 2004, pp. 706.

[6] J. Hightower and G. Borriello, "Location systems for ubiquitous computing," *Computer,* vol. 34, pp. 57-66, 2001.

[7] C. A. Patterson, R. R. Muntz and C. M. Pancake, "Challenges in location-aware computing," *Pervasive Computing, IEEE,* vol. 2, pp. 80-89, 2003.

[8] M. Weiser, "Hot topics-ubiquitous computing," *Computer,* vol. 26, pp. 71-72, 1993.

[9] E. Foxlin, "Pedestrian tracking with shoe-mounted inertial sensors," *Computer Graphics and Applications, IEEE,* vol. 25, pp. 38, nov.-dec., 2005.

[10] S. Beauregard, "Omnidirectional pedestrian navigation for first responders," in *Positioning, Navigation and Communication, 2007. WPNC '07. 4th Workshop on,* 2007, pp. 33.

[11] Wikipedia. Wikipedia the free encyclopedia. [http://www.wikipedia.com]. Available: http://www.wikipedia.com.

[12] K. E. Parsopoulos and M. N. Vrahatis, "Particle swarm optimization method for constrained optimization problems," in *In Proceedings of the Euro-International Symposium on Computational Intelligence 2002,* 2002, pp. 214-220.

[13] W. A. Chang and R. S. Ramakrishna, "A genetic algorithm for shortest path routing problem and the sizing of populations," *Evolutionary Computation, IEEE Transactions on,* vol. 6, pp. 566-579, 2002.

[14] J. Stender, "Introduction to genetic algorithms," in *Applications of Genetic Algorithms, IEE Colloquium on,* 1994, pp. 1/1-1/4.

[15] A. Gomez-Iglesias, M. A. Vega-Rodriguez, F. Castejon-Magana, M. Cardenas-Montes and E. Morales-Ramos, "Using a genetic algorithm and the grid to improve transport levels in the TJ-II stellarator," in *Parallel and Distributed Computing, 2008. ISPDC '08. International Symposium on,* 2008, pp. 81-88.

[16] G. Pengfei, W. Xuezhi and H. Yingshi, "The enhanced genetic algorithms for the optimization design," in *Biomedical Engineering and Informatics (BMEI), 2010 3rd International Conference on,* 2010, pp. 2990-2994.

[17] D. T. Pham and D. Karaboga, *Intelligent Optimisation Techniques: Genetic Algorithms, Tabu Search, Simulated Annealing and Neural Networks.* Springer, 2000.

[18] Q. Ke, T. Jiang and S. D. Ma, "A tabu search method for geometric primitive extraction," *Pattern Recog. Lett.,* vol. 18, pp. 1443-1451, 12, 1997.

[19] C. A. C. Coello, D. A. Van Veldhuizen and G. B. Lamont, *Evolutionary Algorithms for Solving Multi-Objective Problems.* Kluwer Academic, 2002.

[20] P. Siarry and G. Berthiau, "Fitting of tabu search to optimize functions of continuous variables," *Int J Numer Methods Eng,* vol. 40, pp. 2449-2457, 1997.

[21] S. Kirkpatrick, C. D. Gelatt and M. P. Vecchi, "Optimization by Simulated Annealing," *Science,* vol. 220, pp. 671-680, May 13, 1983.

[22] G. Zong Woo, K. Joong Hoon and G. V. Loganathan, "A New Heuristic Optimization Algorithm: Harmony Search," *Simulation,* vol. 76, pp. 60-68, February 01, 2001.

[23] K. S. Lee and Z. W. Geem, "A new meta-heuristic algorithm for continuous engineering optimization: harmony search theory and practice," *Comput. Methods Appl. Mech. Eng.,* vol. 194, pp. 3902-3933, sep, 2005.

[24] M. Mahdavi, M. Fesanghary and E. Damangir, "An improved harmony search algorithm for solving optimization problems," *Applied Mathematics and Computation,* vol. 188, pp. 1567-1579, may, 2007.

[25] X. Wang, "Hybrid nature-inspired computation methods for optimization," *Multiprint Oy,* 2009.

[26] A. Schug, T. Herges and W. Wenzel, "Reproducible Protein Folding with the Stochastic Tunneling Method," *Phys. Rev. Lett.,* vol. 91, pp. 158102, Oct, 2003.

[27] R. Y. Rubinstein and D. P. Kroese, *The Cross-Entropy Method: A Unified Approach to Combinatorial Optimization, Monte-Carlo Simulation, and Machine Learning*. Springer, 2004.

[28] P. E. Heegaard, O. Wittner, V. F. Incola and B. Helvik, "Distributed asynchronous algorithm for cross-entropy-based combinatorial optimization," in *International Workshop on Rare Event Simulation \& Combinatorial Optimization (RESIM 2004),* 2004, .

[29] P. T. D. Boer, D. P. Kroese, S. Mannor and R. Y. Rubinstein, "A Tutorial on the Cross-Entropy Method," *Annals of Operations Research,* vol. 134, 2002.

[30] Y. del Valle, M. Digman, A. Gray, J. Perkel, G. K. Venayagamoorthy and R. G. Harley, "Enhanced particle swarm optimizer for power system applications," in *Swarm Intelligence Symposium, 2008. SIS 2008. IEEE,* 2008, pp. 1-7.

[31] L. Zhang, H. Yu and S. Hu, "A new approach to improve particle swarm optimization," in *Proceedings of the 2003 International Conference on Genetic and Evolutionary Computation: PartI,* Chicago, IL, USA, 2003, pp. 134-139.

[32] J. Kennedy and R. Eberhart, "Particle swarm optimization," in *Neural Networks, 1995. Proceedings., IEEE International Conference on,* Perth, WA, Australia, 1995, pp. 1942-1948 o.4.

[33] X. Hu and R. Eberhart, "Solving constrained nonlinear optimization problems with particle swarm optimization," in *6th World Multiconference on Systemics, Cybernetics and Informatics (SCI 2002,* 2002, pp. 203-206.

[34] K. Y. Lee and Jong-Bae Park, "Application of particle swarm optimization to economic dispatch problem: Advantages and disadvantages," in *Power Systems Conference and Exposition, 2006. PSCE '06. 2006 IEEE PES,* 2006, pp. 188-192.

[35] C. Yang and D. Simon, "A new particle swarm optimization technique," in *Proceedings of the 18th International Conference on Systems Engineering,* 2005, pp. 164-169.

[36] H. Fan, "A modification to particle swarm optimization algorithm," *Engineering Computations,* vol. 19, pp. 970-989, 2002.

[37] J. Kennedy and R. C. Eberhart, "A discrete binary version of the particle swarm algorithm," in *Systems, Man, and Cybernetics, 1997. Computational Cybernetics and Simulation., 1997 IEEE International Conference on,* 1997, pp. 4104-4108 vol.5.

[38] D. Sudholt and C. Witt, "Runtime analysis of binary PSO," in *Proceedings of the 10th Annual Conference on Genetic and Evolutionary Computation,* Atlanta, GA, USA, 2008, pp. 135-142.

[39] W. J. Gutjahr, "First steps to the runtime complexity analysis of ant colony optimization," *Comput. Oper. Res.,* vol. 35, pp. 2711-2727, sep, 2008.

[40] F. Neumann and C. Witt, "Runtime Analysis of a Simple Ant Colony Optimization Algorithm," *Algorithmica,* vol. 54, pp. 243-255, apr, 2009.

[41] M. A. Khanesar, M. Teshnehlab and M. A. Shoorehdeli, "A novel binary particle swarm optimization," in *Control & Automation, 2007. MED '07. Mediterranean Conference on,* 2007, pp. 1-6.

[42] F. Gao, G. Cui, Q. Zhao and H. Liu, "Application of Improved Discrete Particle Swarm Algorithm in Partner Selection of Virtual Enterprise," *IJCSNS International Journal of 208 Computer Science and Network Security,* vol. 6, March 2006.

[43] J. M. Hereford and H. Gerlach, "Integer-valued particle swarm optimization applied to sudoku puzzles," in *Swarm Intelligence Symposium, 2008. SIS 2008. IEEE,* 2008, pp. 1-7.

[44] Q. Shen, W. Shi, W. Kong and B. Ye, "A combination of modified particle swarm optimization algorithm and support vector machine for gene selection and tumor classification," *Talanta,* vol. 71, pp. 1679-1683, 3/15, 2007.

[45] H. Yu, G. Gu, H. Liu, J. Shen and C. Zhu, "A novel discrete particle swarm optimization algorithm for microarray data-based tumor marker gene selection," in *Proceedings of the 2008 International Conference on Computer Science and Software Engineering - Volume 01,* 2008, pp. 1057-1060.

[46] J. Hsin, C. Yang, K. Huang and C. Yang, "An ant colony optimization approach for the protein side chain packing problem," in *Proceedings of the 6th Conference on Microelectronics, Nanoelectronics, Optoelectronics,* Istanbul, Turkey, 2007, pp. 44-49.

[47] W. Yang, "A tabu-search based algorithm for the multicast-streams distribution problem," *Comput.Netw.,* vol. 39, pp. 729-747, aug, 2002.

[48] S. Sun, A. Abraham, G. Zhang and H. Liu, "A particle swarm optimization algorithm for neighbor selection in peer-to-peer networks," in *Proceedings of the 6th International Conference on Computer Information Systems and Industrial Management Applications,* 2007, pp. 166-172.

[49] S. G. M. Koo, K. Kannan and C. S. G. Lee, "On neighbor-selection strategy in hybrid peer-to-peer networks," *Future Gener.Comput.Syst.,* vol. 22, pp. 732-741, aug, 2006.

[50] C. Papagianni, K. Papadopoulos, C. Pappas, N. D. Tselikas, D. T. Kaklamani and I. S. Venieris, "Communication network design using particle swarm optimization," in *Computer Science and Information Technology, 2008. IMCSIT 2008. International Multiconference on,* 2008, pp. 915-920.

[51] M. G. C. Resende and P. M. Pardalos, *Handbook of Optimization in Telecommunications.* Springer, 2006.

[52] M. Pioro and D. Medhi, *Routing Flow, and Capacity Design in Communication and Computer Networks.* Elsevier/Morgan Kaufmann, 2004.

[53] A. W. Mohemmed and N. C. Sahoo, "Efficient Computation of Shortest Paths in Networks Using Particle Swarm Optimization and Noising Metaheuristics," *Discrete Dynamics in Nature and Society,* vol. 2007, 2007.

[54] M. K. M. Ali and F. Kamoun, "Neural networks for shortest path computation and routing in computer networks," *Neural Networks, IEEE Transactions on,* vol. 4, pp. 941-954, 1993.

[55] J. Kuri, N. Puech, M. Gagnaire and E. Dotaro, "Routing foreseeable lightpath demands using a tabu search meta-heuristic," in *Global Telecommunications Conference, 2002. GLOBECOM '02. IEEE,* 2002, pp. 2803-2807 vol.3.

[56] I. Charon and O. Hudry, "The noising method: a new method for combinatorial optimization," *Oper. Res. Lett.,* vol. 14, pp. 133-137, 10, 1993.

[57] S. Gheitanchi, F. Ali and E. Stipidis, "Trained Particle Swarm Optimization for Ad-Hoc Collaborative Computing Networks," .

[58] A. Z.M., H. G.W., Y. A. M. and D. M. I., "Power Minimization Algorithm in Wireless Ad-hoc Networks Based on PSO," *Journal of Applied Sciences,* vol. 7, pp. 2523-2526, 2007.

[59] A. Muqattash and M. Krunz, "Power controlled dual channel (PCDC) medium access protocol for wireless ad hoc networks," in *INFOCOM 2003. Twenty-Second Annual Joint Conference of the IEEE Computer and Communications. IEEE Societies,* 2003, pp. 470-480 vol.1.

[60] Y. K. Thong, M. S. Woolfson and R. E. Challis, "Dependence of inertial measurements of distance on accelerometer noise," *Sci.Technol,* vol. 13, pp. 1163-1172, 2002.

[61] P. Goyal, V. J. Ribeiro, H. Saran and A. Kumar, "Strap-down pedestrian dead-reckoning system," in *Indoor Positioning and Indoor Navigation (IPIN), 2011 International Conference on,* 2011, pp. 1.

[62] M. Farley and M. Chapman, "Monocular SLAM," *GPS World,* pp. pp. 42-49, September, 2008.

[63] A. J. Davison, I. D. Reid, N. D. Molton and O. Stasse, "MonoSLAM: Real-Time Single Camera SLAM," *Pattern Analysis and Machine Intelligence, IEEE Transactions on,* vol. 29, pp. 1052, june, 2007.

[64] R. Want, A. Hopper, V. Falcao and J. Gibbons, "The active badge location system," *ACM Trans.Inf.Syst.,* vol. 10, pp. 91-102, January, 1992.

[65] R.Want, B.Schilit, N. Adams, R. Gold, D. Gold- berg, K.Petersen ,J.Ellis, M Weiser, "The parctab ubiquitous computing experiment," in *Mobile Computing*, Tomasz Imielinski, Ed. Kluwer Publishing, 1997, pp. pp 45-101.

[66] A. Ward, A. Jones and A. Hopper, "A new location technique for the active office," *Personal Communications, IEEE,* vol. 4, pp. 42-47, 1997.

[67] A. Harter, A. Hopper, P. Steggles, A. Ward and P. Webster, "The anatomy of a context-aware application," in *Proceedings of the 5th Annual ACM/IEEE International Conference on Mobile Computing and Networking,* Seattle, Washington, United States, 1999, pp. 59-68.

[68] J. Hightower, G. Borriello and R. Want, "SpotON: An indoor 3D location sensing technology based on RF signal strength," Tech. Rep. 2000-02-02, February 18, 2000.

[69] RFID, "RadioFrequencyIdentification (RFID) home page," .

[70] K. Lorincz and M. Welsh, "MoteTrack: A robust, decentralized approach to RF-based location tracking." in *LoCA,* 2005, pp. 63-82.

[71] S. Kennedy, J. Hamilton and H. Martell, "Architecture and system performance of SPAN -NovAtel's GPS/INS solution," in *Position, Location, and Navigation Symposium, 2006 IEEE/ION,* 2006, .

[72] J. Collin, O. Mezentsev, G. Lachapelle, "Indoor Positioning System Using Accelerometry and High Accuracy Heading Sensors," *Proceedings of the 16th International Technical Meeting of the Satellite Division of the Institute of Navigation (ION GPS/GNSS 2003),* pp. 1164 - 1170, September 2003.

[73] P. Bahl and V. N. Padmanabhan, "User location and tracking in an in-building radio network," Microsoft Research Technical Report: MSR-TR-99-12, 1999.

[74] A. Harter and A. Hopper, "A distributed location system for the active office," *Network, IEEE,* vol. 8, pp. 62-70, 1994.

[75] J. Krumm, S. Harris, B. Meyers, B. Brumitt, M. Hale and S. Shafer, "Multi-camera multi-person tracking for EasyLiving," in *Visual Surveillance, 2000. Proceedings. Third IEEE International Workshop on,* 2000, pp. 3-10.

[76] M. Ribo, M. Brandner and A. Pinz, "A Flexible Software Architecture for Hybrid Tracking," *J. Robot. Syst.,* vol. 21, pp. 53-62, 2004.

[77] M. Ribo, P. Lang, H. Ganster, M. Brandner, C. Stock and A. Pinz, "Hybrid tracking for outdoor augmented reality applications," *Computer Graphics and Applications, IEEE,* vol. 22, pp. 54-63, 2002.

[78] D. Vlasic, R. Adelsberger, G. Vannucci, J. Barnwell, M. Gross, W. Matusik and J. Popovic, "Practical motion capture in everyday surroundings," *ACM Trans.Graph.,* vol. 26, July, 2007.

[79] J. A. Corrales, F. A. Candelas and F. Torres, "Hybrid tracking of human operators using IMU/UWB data fusion by a kalman filter," in *Proceedings of the 3rd*

ACM/IEEE International Conference on Human Robot Interaction, Amsterdam, The Netherlands, 2008, pp. 193-200.

[80] J. J. LaViola and Jr., "A Comparison of Unscented and Extended Kalman Filtering for Estimating Quaternion Motion," 2003.

[81] R. C. Luo and O. Chen, "Indoor human dynamic localization and tracking based on sensory data fusion techniques," in *Intelligent Robots and Systems, 2009. IROS 2009. IEEE/RSJ International Conference on,* 2009, pp. 860-865.

[82] E. North, J. Georgy, M. Tarbouchi, U. Iqbal and A. Noureldin, "Enhanced mobile robot outdoor localization using INS/GPS integration," in *Computer Engineering & Systems, 2009. ICCES 2009. International Conference on,* 2009, pp. 127-132.

[83] J. Gonzalez, J. L. Blanco, C. Galindo, A. Ortiz-de-Galisteo, J. A. Fernandez-Madrigal, F. A. Moreno and J. L. Martinez, "Combination of UWB and GPS for indoor-outdoor vehicle localization," in *Intelligent Signal Processing, 2007. WISP 2007. IEEE International Symposium on,* 2007, pp. 1-6.

[84] B. Ristic, S. Arulampalam and N. Gordon, *Beyond the Kalman Filter: Particle Filters for Tracking Applications.* Artech House, 2004.

[85] M. Kim and B. Noble, "Mobile network estimation," in *Proceedings of the 7th Annual International Conference on Mobile Computing and Networking,* Rome, Italy, 2001, pp. 298-309.

[86] D. H. Titterton and J. L. Weston, *Strapdown Inertial Navigation Technology.* Stevenage, U.K. Reston, VA: Institution of Electrical Engineers American Institute of Aeronautics and Astronautics, 2004.

[87] O. J. Woodman, C. O. J. Woodman and O. J. Woodman, "An introduction to inertial navigation," 2007.

[88] N. B. Priyantha, "The cricket indoor location system," PhD Thesis MIT, 2005.

[89] MIT, "Cricket project: Cricket v2 user manual," MIT Computer Science and Arti_cial Intelligence Lab, Cambridge, MA 02139, MIT, July 2004, URL: http://cricket.csail.mit.edu/v2man.html.

[90] A. D. Young, M. J. Ling and D. K. Arvind, "IMUSim: A simulation environment for inertial sensing algorithm design and evaluation," in *Information Processing in Sensor Networks (IPSN), 2011 10th International Conference on,* 2011, pp. 199.

[91] CMU Graphics Lab, "CMU Graphics Lab motion capture database," *Http://mocap.Cs.Cmu.Edu/,* .

[92] R. Priwgharm and P. Chemtanomwong, "A comparative study on indoor localization based on RSSI measurement in wireless sensor network," in *Computer Science and Software Engineering (JCSSE), 2011 Eighth International Joint Conference on,* 2011, pp. 1.

[93] Hui Liu, H. Darabi, P. Banerjee and Jing Liu, "Survey of Wireless Indoor Positioning Techniques and Systems," *Systems, Man, and Cybernetics, Part C: Applications and Reviews, IEEE Transactions on,* vol. 37, pp. 1067-1080, 2007.

[94] F. D, "Modeling signal attenuation in IEEE 802.11 wireless LANs," Stanford University, Tech. Rep. TR-KP06-0118, July 2005.

[95] S. Y. Seidel and T. S. Rappaport, "914 MHz path loss prediction models for indoor wireless communications in multifloored buildings," *Antennas and Propagation, IEEE Transactions on,* vol. 40, pp. 207-217, 1992.

[96] Glen Ballou, *Handbook for Sound Engineers.* Focal Press, Gulf Professional Publishing, 2005.

[97] D. Bratton and J. Kennedy, "Defining a standard for particle swarm optimization," in *Swarm Intelligence Symposium, 2007. SIS 2007. IEEE,* 2007, pp. 120-127.

[98] J. M. Loomis, R. G. Golledge and R. L. Klatzky, "Navigation System for the Blind: Auditory Display Modes and Guidance," *Presence: Teleoper.Virtual Environ.,* vol. 7, pp. 193-203, apr, 1998.

[99] W. Crandall, W. Gerrey and A. Alden, "Remote signage and its implications to print-handicapped travelers," *Proceedings: Rehabilitation Engineering Society of North America (RESNA) Annual Conference,* pp. 251-253, 1993.

[100] E. D'Atri, C. M. Medaglia, A. Serbanati, U. B. Ceipidor, E. Panizzi and A. D'Atri, "A system to aid blind people in the mobility: A usability test and its results," in *Proceedings of the Second International Conference on Systems,* 2007, pp. 35.

[101] F. Kargl, S. Gessler and F. Flerlage, "The iNAV indoor navigation system," in *Proceedings of the 4th International Conference on Ubiquitous Computing Systems,* Tokyo, Japan, 2007, pp. 110-117.

[102] M. Minami, Y. Fukuju, K. Hirasawa, S. Yokoyama, M. Mizumachi, H. Morikawa and T. Aoyama, "DOLPHIN: A practical approach for implementing a fully distributed indoor ultrasonic positioning system," in *UbiComp 2004: Ubiquitous Computing*, N. Davies, E. Mynatt and I. Siio, Eds. Springer Berlin / Heidelberg, 2004, pp. 347-365.

[103] J. Wilson, B. N. Walker, J. Lindsay, C. Cambias and F. Dellaert, "SWAN: System for wearable audio navigation," in *Proceedings of the 11th International Syposium on Wearable Computers (ISWC 2007,* 2007, .

[104] M. Hazas and A. Hopper, "Broadband ultrasonic location systems for improved indoor positioning," *Mobile Computing, IEEE Transactions on,* vol. 5, pp. 536-547, 2006.

[105] R. Hewett, G. Douglas, A. Ramli and S. Keil, "A survey of the social activity and social networking of blind and partially sighted young people," University of Birmingham, February 2012.

[106] G. J. Myers, C. Sandler and T. Badgett. *Art of Software Testing* 2004.

[107] E. F. Miller, "Program testing -- an overview for managers," in *Computer Software and Applications Conference, 1978. COMPSAC '78. the IEEE Computer Society's Second International,* 1978, pp. 114-119.

[108] B. Hetzel, *The Complete Guide to Software Testing, Second Edition.* John Wiley & Sons (US), 1993.

[109] W. C. Hetzel, *Program Test Methods.* Prentice-Hall, 1973.

[110] G. J. Myers, C. Sandler and T. Badgett, *The Art of Software Testing.* John Wiley \& Sons, 2011.

[111] W. C. Hetzel, *The Complete Guide to Software Testing.* QED Information Sciences, 1988.

[112] S. M. K. Quadri and S. U. Farooq, "Article: Software Testing - Goals, Principles, and Limitations," *International Journal of Computer Applications,* vol. 6, pp. 7-11, September, 2010.

[113] R. D. Craig and S. P. Jaskiel, *Systematic Software Testing.* Artech House, 2002.

[114] G. Fournier, *Essential Software Testing: A use-Case Approach.* Auerbach Publications, 2008.

[115] D. Choudhary and V. Kumar, "Software Testing," *Journal of Computational Simulation and Modeling,* vol. 1, pp. 1-9, 2011.

[116] J. Watkins and S. Mills. *Testing IT : An Off-the-Shelf Software Testing Process (2nd Edition)* 2010.

[117] M. Khan, "Different Forms of Software Testing Techniques for Finding Errors," *IJCSI International Journal of Computer Science Issues,* vol. 7, May 2010.

[118] W. Foundation, *Software Testing.* eM Publications, 2011.

[119] P. Jorgensen, *Software Testing: A Craftsman's Approach, Third Edition.* CRC Press, 2008.

[120] A. Arcuri, M. Z. Iqbal and L. Briand, "Random Testing: Theoretical Results and Practical Implications," *Software Engineering, IEEE Transactions on,* vol. 38, pp. 258-277, 2012.

[121] AMIRG. State transition analysis testing. November, 2008. Available: http://www.testingexcellence.com/state-transition-testing/.

[122] E. Y. Li, "Software testing in a system development process: A life cycle perspective," *Journal of Systems Management,* pp. 23-31, 1990.

[123] J. D. Meier and e. al., *Performance Testing Guidance for Web Applications: Patterns & Practices.* Microsoft Press, 2007.

[124] L. Bernstein and C. Yuhas, *Trustworthy Systems through Quantitative Software Engineering* Wiley-IEEE Press, 2005.

[125] P. Rook, "Controlling software projects," *Software Engineering Journal,* vol. 1, pp. 7, 1986.

[126] S. Mathur and S. Malik, "Article: Advancements in the V-Model," *International Journal of Computer Applications,* vol. 1, pp. 29-34, February, 2010.

[127] B. Hambling, P. Morgan, A. Samaroo, G. Thompson and P. Williams, *Software Testing: An Istqb-Iseb Foundation Guide.* British Informatics Society Limited, 2010.

[128] D. Graham and E. Van Veenendaal, *Foundations of Software Testing: ISTQB Certification.* Cengage Learning, 2008.

[129] I. Burnstein, *Practical Software Testing: A Process-Oriented Approach.* Springer, 2003.

[130] M. M. Wang and Z. Sun, *Handbook of Research on Complex Dynamic Process Management: Techniques for Adaptability in Turbulent Environments.* Igi Global, 2009.

[131] S. Naik and P. Tripathy, *Software Testing and Quality Assurance: Theory and Practice.* John Wiley & Sons, 2011.

[132] I. Bashir and A. L. Goel, *Testing Object-Oriented Software: Life Cycle Solutions.* Springer, 1999.

[133] A. M. J. Hass, *Guide to Advanced Software Testing.* Artech House, 2008.

[134] A. Spillner, T. Linz and H. Schaefer, *Software Testing Foundations: A Study Guide for the Certified Tester Exam.* Rocky Nook, 2011.

[135] A. Kossiakoff, W. N. Sweet, S. Seymour and S. M. Biemer, *Systems Engineering Principles and Practice.* John Wiley & Sons, 2011.

[136] D. R. Wallace and J. C. Cherniavsky, *Guide to Software Acceptance.* U.S. Department of Commerce, National Institute of Standards and Technology, 1990.

[137] D. R. Wallace and R. U. Fujii, "Software verification and validation: an overview," *Software, IEEE,* vol. 6, pp. 10-17, 1989.

[138] E. Castillo, J. M. Gutierrez and A. S. Hadi, *Expert Systems and Probabilistic Network Models.* Springer, 1997.

[139] A. Barr, E. A. Feigenbaum and P. R. Cohen, *The Handbook of Artificial Intelligence.* HeurisTech Press, 1982.

[140] L. Stevens, *Artificial Intelligence, the Search for the Perfect Machine.* Hayden Book Co, 1985.

[141] M. Josephine and K. Subramanian, "Software Error Detection And Correction(Sedc) Using Layer Based Approach In Expert System," *International Journal of Reviews in Computing, IJRC,* vol. 3, pp. 34-43, June 2012.

[142] S. Shajahan. *Management Information Systems* 2004.

[143] D. W. Peter B. B. Turney and D. A. Watne, *Auditing EDP Systems.* Prentice-Hall International, 2002.

[144] R. Nakatsu, *Diagrammatic Reasoning in AI.* Wiley, 2009.

[145] R. J. K. Jacob and J. N. Froscher, "A software engineering methodology for rule-based systems," *Knowledge and Data Engineering, IEEE Transactions on,* vol. 2, pp. 173-189, 1990.

[146] B. Buchanan and R. Dude, "Principles of rule based systems," Department of Computer Science, Stanford University, Tech. Rep. STAN-CS-82-926, August 1982.

[147] J. L. Hein, *Discrete Mathematics.* Jones and Bartlett Publishers, 2003.

[148] R. K. Lindsay, *Applications of Artificial Intelligence for Organic Chemistry: The DENDRAL Project.* McGraw-Hill Book Co, 1980.

[149] J. Moses, "Symbolic integration the stormy decade," in *Proceedings of the Second ACM Symposium on Symbolic and Algebraic Manipulation,* Los Angeles, California, United States, 1971, pp. 427-440.

[150] H. E. Pople, "The formation of composite hypotheses in diagnostic problem solving: An exercise in synthetic reasoning," in *Proceedings of the 5th International Joint Conference on Artificial Intelligence - Volume 2,* Cambridge, USA, 1977, pp. 1030-1037.

[151] E. H. Shortliffe, A. C. Scott, M. B. Bischoff, A. B. Campbell, W. V. Melle and C. D. Jacobs, "Oncocin: An expert system for oncology protocol management," in 1981, pp. 876-881.

[152] P. Szolovits and S. G. Pauker, "Readings in uncertain reasoning," in , G. Shafer and J. Pearl, Eds. San Francisco, CA, USA: Morgan Kaufmann Publishers Inc, 1990, pp. 282-297.

[153] S. M. Weiss, C. A. Kulikowski, S. Amarel and A. Safir, "A model-based method for computer-aided medical decision-making," *Artif. Intell.,* vol. 11, pp. 145-172, 8, 1978.

[154] L. C. Tong, "An explanation facility for a grammar writing system," in *Proceedings of the 13th Conference on Computational Linguistics - Volume 2,* Helsinki, Finland, 1990, pp. 359-364.

[155] W. R. Swartout, "Explaining and justifying expert consulting programs," in *Proceedings of the 7th International Joint Conference on Artificial Intelligence - Volume 2,* Vancouver, BC, Canada, 1981, pp. 815-823.

[156] Mitchell and T. Michael, "Version spaces: An approach to concept learning," Stanford Univ Calif Dept Of Computer Science, Tech. Rep. ADA074462, December 1978.

[157] R. Davis, "Interactive transfer of expertise: Acquisition of new inference rules," *Artif. Intell.,* vol. 12, pp. 121-157, 8, 1979.

[158] W. J. Van Melle, "A domain-independent system that aids in constructing knowledge-based consultation programs," 1980.

[159] Fain, J., F. Hayes-Roth,H.Sowizral, and D. Waterman, "Programming Examples in ROSIE," 1981.

[160] S. M. Weiss and C. A. Kulikowski, "EXPERT: A system for developing consultation models," in *Proc. International Joint Conference on Artificial Intelligence (IJCAI-79),* 1979, pp. 942-947.

[161] C. Forgy and J. McDermott, "OPS: A domain-independent production system language," in *Proceedings of the 5th International Joint Conference on Artificial Intelligence - Volume 2,* Cambridge, USA, 1977, pp. 933-939.

[162] SRI International. Artificial Intelligence Center and R. Reboh, *Knowledge Engineering Techniques and Tools in the Prospector Environment.* SRI International, 1981.

[163] M. J. Liberatore and A. C. Stylianou, "The development manager's advisory system: A knowledg-based," *Decision Sciences,* vol. 24, pp. 953, Sep/Oct 1993, 1993.

[164] L. Medsker and J. Liebowitz, *Design and Development of Expert Systems and Neural Networks.* Macmillan, 1994.

[165] C. T. Leondes, *Knowledge-Based Systems: Systems and Applications*. Academic Press, 2000.

[166] V. Barr and D. V. Barr, "Rule-Based System Testing with Control and Data Flow Techniques," 1996.

[167] R. O'Keefe and D. O'Leary, "Expert system verification and validation: a survey and tutorial," *Artif. Intell. Rev.,* vol. 7, pp. 3-42, 1993.

[168] B. W. Boehm, "Verifying and Validating Software Requirements and Design Specifications," *Software, IEEE,* vol. 1, pp. 75-88, 1984.

[169] R. Knauf, A. J. Gonzalez and T. Abel, "A framework for validation of rule-based systems," *Systems, Man, and Cybernetics, Part B: Cybernetics, IEEE Transactions on,* vol. 32, pp. 281-295, 2002.

[170] T. Bench-Capon, F. Coenen, H. S. Nwana, R. Paton and M. Shave, "*Two Aspects of the Validation and Verification of Knowledge-Based Systems*," *IEEE Expert,* vol. 8, pp. 76, june, 1993.

[171] R. Evertsz, "The automated analysis of rule-based systems, based on their procedural semantics," in *Proceedings of the 12th International Joint Conference on Artificial Intelligence - Volume 1,* Sydney, New South Wales, Australia, 1991, pp. 22-27.

[172] R. Greiner and C. Elkan, "Measuring and improving the effectiveness of representations," in *Proceedings of the 12th International Joint Conference on Artificial Intelligence - Volume 1,* Sydney, New South Wales, Australia, 1991, pp. 518-524.

[173] G. Guida and G. Mauri, *Knowledge and Data Engineering, IEEE Transactions on Title={Evaluating Performance and Quality of Knowledge-Based Systems: Foundation and Methodology,* vol. 5, pp. 204, apr, 1993.

[174] C. Anantaram, G. Nagaraja and K. V. Nori, "Verification of accuracy of rules in a rule based system," *Data Knowl. Eng.,* vol. 27, pp. 115-138, sep, 1998.

[175] S. Smith and A. Kandel, "Verification and validation in fuzzy expert systems," in *Systems, Man and Cybernetics, 1995. Intelligent Systems for the 21st Century., IEEE International Conference on,* 1995, pp. 3647-3651 vol.4.

[176] P. B. Powell, "Software validation, verification, and testing technique and tool reference guide," pp. vi, 131 p. :, 1982.

[177] I. Sommerville, *Software Engineering.* Wokingham, England: Addison-Wesley Publishing Company, 1992.

[178] S. Smith, "Verification and validation of rule-based expert systems," *Doctoral Dissertation, University of Microfilms International,* 1991.

[179] C. J. R. Green, "An evolutionary approach to verification and validation of expert systems," in *Proceedings of the 1987 Fall Joint Computer Conference on Exploring Technology: Today and Tomorrow,* Dallas, Texas, United States, 1987, pp. 760-760.