ADVANCES IN INSTANTANEOUS AND DYNAMIC LOCALIZATION IN INDOOR ENVIRONMENTS

BY BEGÜMHAN TURGUT

A dissertation submitted to the

Graduate School—New Brunswick

Rutgers, The State University of New Jersey

in partial fulfillment of the requirements

for the degree of

Doctor of Philosophy

Graduate Program in Computer Science

Written under the direction of

Richard P. Martin

and approved by

_____

_____

_____

_____

New Brunswick, New Jersey

October, 2014

ABSTRACT OF THE DISSERTATION

Advances in instantaneous and dynamic localization in indoor environments

by Begümhan Turgut

Dissertation Director: Richard P. Martin

The knowledge of the exact location of an object or person is an important enabler for a wide variety of applications. Localization, itself, has a long and distinguished history from sextants to the geodesic tools of modern civil engineers. The recent general availability of global positioning systems has enabled the development of new applications from car navigation and emergency assistance systems to location targeted advertisements. However these systems are largely limited to outdoor spaces preventing the accurate determination of location inside of a building. In this dissertation, we consider the issues of both the instantaneous and dynamic localization of people in an indoor environment where GPS signals are not available.

We begin by studying instantaneous localization and the variety of new techniques that share the common approach of utilizing a number of fixed reference points with known locations in order to determine the location of the target. Since indoor environments usually prevent a direct line of sight from the target to a known point we need to infer the targets location from secondary measurements with varying reliability. We then outline a new method for instantaneous localization whose output not only provides the location of the target but also the confidence of the localization based on the environmental variability.

We then examine how these factors extend to the complex scenarios of dynamic localization. In these scenarios instantaneous localization steps would need to be performed at an unrealistically high frequency in order to extract the desired smooth trajectory of a moving object. We find that by passing the results of these measurements over time through a sampling-importance-resampling particle filter can reduce the influence of noise and allow interpolation to predict the current, and sometimes future, location of the target. This particle filter utilizes a probabilistic reasoning technique to extract continuous tracking information from the periodic instantaneous location input as well as to integrate knowledge about the environment and the targets capabilities in order to provide its location. We then present the experimental results from scenarios taken from multiple indoor locations demonstrating the accuracy and precision of the localization method.

Acknowledgements

Although a mere thank you as written in this page would seem to fall short of representing my gratitude, it is only fitted that I shall try.

I owe a great deal to my family for my upbringing where I was taught many lessons that have proven invaluable and also to be fortunate enough to know that their support was with me at all times. They always encouraged me to try things without fear of to breaking them (except maybe in the kitchen), to explore, to ask questions, to never be satisfied with a single answer but always search for more, to think for myself, to try figure things on my own first to learn and then ask for help when I needed. All these and more have shaped the road I took and I am truly grateful for it all.

My one and only sister, Dr. Damla Turgut, and my brother-in-law, Dr. Ladislau Boloni, both of who showed their support, whether to proof read a document or to brain storm an idea or simply to listen when I needed to complain. I often recall a friends comment saying that we all can use a big sister like yours which is as great summary as I could provide. I could also not pass on without mentioning my brother-in-law who has always been the best mentor one can wish for, be it a technical or non-technical discussion, staying focused after all others had resigned. To my lovely niece, Miss Deniz Boloni-Turgut, whom can do no wrong by just by being herself, has help me just by being her lovely witty self and for asking questions that often makes me stop and ponder. She brings such joy to our lives and I am ever so grateful for her being in mine.

I was also fortunate to have interacted with some of the nicest people during this journey including my fellow graduate students interacting with whom has made daily life less routine, to say the least, and allowed me to glimpse how ones peers approach research problems, from the definition phase, all the way to a proposed solution is one

of the greatest experiences of graduate school.

Finally I owe many thanks to my advisor, Prof. Richard P. Martin. His firm, yet kind, approach to teaching how to perform research, not just solving a single question at hand but understanding the bigger issues in the area, was very informative and valuable. From him I have learned that one can solve most anything if one knows how to define the problem and how to attack the problem from different angles without dwelling on what is not working but rather learn from it and very importantly, knowing how to apply ones knowledge from different disciplines in order to the find not just any solution or outcome but the best possible one. Thank you!

Dedication

To my family, for their support in every sense of the word.

Table of Contents

Chapter 1

Introduction

The knowledge of the exact location of an object, a person or a robot is an important enabler for a wide variety of applications. Localization has a long and dignified history from the sextants of the sailors to the geodesic tools of civil engineers. Most recently the wide availability of global navigation satellite systems such as GPS, Galileo and GLONASS have opened a range of new applications such as car navigation systems, location-based targeted advertising and the automatic localization of an emergency phone call. However, the use of GPS is limited to open spaces - for instance they can not be used to accurately determine locations inside buildings.

In recent years a variety of new techniques have been proposed for localization in environments where GPS or similar systems are not usable. These techniques are based on the same principles as all the other localization techniques used throughout history: they are determining the location of the target relative to a number of reference points with known locations. Unfortunately, in these environments we usually do not have a direct line of sight, nor the ability to extend a steel tape from the target to a known point. We need to infer the location from indirect, noisy and potentially corrupted measurements.

The work described in this thesis considers indoor localization based on wireless signal strength measurements. As most modern office buildings are pre-wired with wireless access points of known location this method can be applied in a wide range of scenarios.

The nature of the indoor environment, however, also creates new challenges. Wireless signals decay in empty space based on a well-known formula. Indoors, however, the received signal strength (RSS) is affected by the attenuation provided by the building

structure, such as walls, metallic frames and furniture, as well as by moving objects such as persons, opening doors, elevators and so on. Other factors such as temperature and air humidity can also affect the RSS. Estimating the correct location of the target under these conditions requires complex mathematical techniques to integrate the information from different measurements and estimate the error. In some cases, we need to identify and discard measurements which have been tampered with.

Even more complexity appears if we are considering dynamic localization (tracking). A naive approach to tracking would be to perform instantaneous localization steps at a high frequency. A better approach is to pass the results of the measurements at different timepoints through a filtering algorithm which can reduce the influence of noise, interpolate among the measurement points and predict the current location of the target even at some time after the last measurement. In recent years *particle filters* have been found especially effective for this purpose.

This thesis describes a series of contributions to the field of indoor localization using wireless signal strengths. We start by outlining a new method of instantaneous localization which simultaneously estimates the location of the target and the accuracy of the localization. Then, we use the output of this method to build a particle filter based dynamic localization system which can take advantage of both estimates. Finally we describe a number of contributions which improve the accuracy of dynamic localization under certain conditions. The performance improvements are quantified using extensive simulation studies.

1.1 The problem

The main question we tried to answer through our research is whether localization systems, especially in indoor environments, should rely on a single source of information, or should integrate information from a heterogeneous set of sources to obtain a better accuracy.

This is not a rhetorical question. GPS, the most successful localization system for outdoor environments, is a single source system: it only uses the satellite signals for

localization. A GPS device is not aware of its environment. In indoor environments, the balance of arguments is different. There is no single source of localization information of a quality comparable to the GPS signal, thus the integration of multiple, noisy information sources would make more sense. Still, even in indoor environments, there is a question whether the enrollment of additional information sources is worth the cost, or indeed, if it even produces any improvement at all. There are opinions in the literature which advocate that simpler systems produce more accurate results.

Our work proposed a number of approaches which combine information from a heterogeneous collection of sources. The main source of information is the received signal strength (RSS) from transmitters with known location. We found RSS to be a much more unreliable source of localization information than GPS: it is affected by attenuation, fading, reflections and can be tampered with by malicious opponents. We argue that using additional information sources - such as the map of reachable and unreachable locations in the environment, the statistical distribution of the locations of the targets, as well as the statistically likely behavior of the targets can significantly improve localization accuracy.

The arguments for a simpler, single source system have some merit. The use of additional information sources is a non-trivial enterprise both in terms of the acquisition of the information, and in terms of the handling of contradictory information. One component of our work centered around modalities to automatically restart the particle filter underlying the localization when the predictions and the observations diverge. If we could rely on a single, reliable source of localization information, this would not be necessary.

To anticipate our results, we found that the integration of multiple sources of information offers benefits which can not be matched by simpler systems. The noise and distortion affecting the RSS data has significant non-Gaussian components which can not be removed by uninformed filtering techniques. The careful use of additional sources of information improves our ability to estimate the location of the target, estimate the localization error, or create a discrete list of possible locations in cases when the measurements have ambiguous interpretations.

1.2 List of contributions

The scientific contributions of this thesis are as follows:

(1) We designed and implemented a method for instantaneous localization based on the *minimal intersection area* of the iso-RSS lines. This method is described in Chapter 3 with an experimental study in Chapter 4.

(2) We designed and implemented a particle filter based method for dynamic localization (tracking) which takes as input the minimal intersection area based instantaneous localization. This method is described in Chapter 5.

(3) We extended the aforementioned method to take advantage of a priori information about the internal structure of the building as well as the physical limitations of the target. These extensions are described in Chapter 5. In the experimental study in Chapter 7 we compare the baseline approach of sequence of instantaneous localizations with the approach of contribution (2), and the extensions provided by contribution (3).

(4) We designed and implemented a method through which the particle filter can automatically detect if the particle cloud has been "stranded" and does not track the target correctly. This method relies on the measurement of the Kullback-Leibling divergence between the observation and the particle cloud. By restarting the particle filter when the divergence exceeds a threshold, the dynamic localization can recover faster from incorrect predictions and thus improve the localization accuracy. The method is described and experimentally verified in Chapter 8.

(5) We designed and implemented a method which allows a particle filter to follow multiple hypotheses with regards to the location of the target. This allows the particle filter to make better decisions when the target is faced with divergent choices. In cases when the default algorithm is making the wrong choice, this method had been shown to reduce the localization error from 60ft to about 5ft, while matching the accuracy of the default algorithm in cases when that makes the right decision. The method is described and experimentally verified in Chapter 9.

1.3 Instantaneous localization with intersection areas

Our instantaneous localization technique is based on inferring the location of a target based on the signal strength of wireless signals, which are either sent by access points of known location and measured by the target, or, alternatively, transmitted by the target and measured by access points of known location.

Our proposed intersection area technique relies on the creation of a series of iso-RSS line maps during the preparation, offline phase. We start by collecting RSS readings at known locations in the indoor area (typically, the corridors and rooms of a large building). The frequency of the data collection depends on the number of landmarks present, the number of access points, and other characteristics of the environment. The collected data for each reading includes the current known location (the ground truth) and the measured RSS value for each access point.

These measurements serve the input to interpolation techniques which create an RSS map for every AP (access point), in a format which facilitates the fast extraction of the iso-RSS lines.

During the localization phase, the measured values are used to extract a number of iso-RSS lines from the maps. In a perfect world, these lines would intersect in a single point representing the location of the target. In practice, however, the iso-RSS lines but they might pass more or less close to each other. Our technique searches for the *minimum intersection area*: the smallest rectangular area which is intersected by all the iso-RSS lines. The center of the minimal intersection area is an estimate of the location of the target, while the area provides an estimate of the localization error.

This approach is described in detail in Chapter 3, while in Chapter 4 we describe an experimental study which validates our approach.

1.4 Particle filter based dynamic localization using the minimal intersection areas as input

If we have an approach to perform instantaneous localization, the simplest way to implement dynamic localization (tracking) is to perform a series of instantaneous localizations at time points t_1, t_2... This approach, which treats the readings as separate observations, discards the information which can be obtained by considering the correlations between the readings.

The alternative is to use the readings to build a *model* of the movement of the target, which can then be consulted to obtain a better estimate of the location of the target, including at timepoints in-between readings and after the last reading. An efficient method to create such a model is the particle filter (also called a sequential Monte Carlo method) which is a *simulation-based* method, which require a predictive model of the simulated phenomena's evolution.

The starting point for the remainder of our work is a well-known type of particle filter, the *sampling-importance-resampling (SIR) filter*. Although a particle filter can be described in a purely abstract language, without anchoring it to any specific application domain, we will make the assumption that the particle filter is targeted to localization, with particles representing potential locations in the 2D space. Each particle has an associated weight which represents the importance of its contribution to the estimate. This allows us to talk about the location of a particle, or the area covered by the particle cloud. Other than this purely linguistic distinction, which avoids many unnecessary qualifications, our implementation rigorously follows the general theory of particle filters.

SIR particles filters have been previously used for dynamic localization (especially for the self-localization of robots). Our contribution for this topic, was to adapt the particle filter to the particularities of the GRAIL system, in particular, the interpretation of the output of the instantaneous localization in probabilistic terms suitable for the update step of the particle filter.

Reviewing the operation of the particle filter based tracking represents our knowledge about the current location of the target in the form of a *cloud of particles*. At each timestep, we perform a prediction step which updates the particles based on a statistical model of the movement of the target. Whenever new information in form of a reading is available, it will be used to update the weight of the particles. As this step might lead to many particles having a low weight, periodically we perform a *resampling* step, replacing the particles with a new set of particles with an equally distributed weight. The most likely location of the target can be extracted from the current particle cloud through several estimation techniques.

In the specific case of our system, the weight update step takes its input from the minimum intersection area localization method. While a naive approach would simply feed in the most likely location returned by the instantaneous localization, the minimum intersection area method also offers an estimate of the localization error (by returning the minimum localization area). Our interpretation is that there is a certain probability that the target will be in the intersection area, and then gradually decreasing probabilities of it being in exponentially larger intersection areas centered on the original one (we call this the ziggurat probability model).

The advantage of this approach is that readings with a low precision (large intersection areas) would have a lower effect on the tracking output than readings with a high precision (small intersection areas).

The prediction step, which attempts to predict the state of the particles for the next step. The prediction needs to encompass all the information the system has about particles (although some systems, for efficiency consideration and others, have decided to ignore some information in favor of simpler models).

Chapter 5 describes this adaptation process of the particle filter model for our settings.

1.5 Using a priori information in particle filter based localization

Some research approaches have suggested that the best results in particle filter based localization can be obtained with simple, lightweight models.

For the purpose of the update step, for instance, this would involve a simple interpretation of the instantaneous localization information: we accept the output of the information provided by the localization step even if this is contradicted by our background knowledge (eg. it puts the target outside the building, or makes it cross walls).

For the purpose of the mobility step, this involves a simple model of the mobility, such as Gaussian location distribution, even if such a movement pattern is highly or even physically impossible for the target.

In contrast to these approaches, we developed an information rich approach where the update and the prediction steps are informed by our a priori knowledge about the environment and the target.

For the update step, we will consider a priori knowledge about the environment, in our case the indoor area of an office building. In such an environment, some movement patterns and locations (such as those which cross walls or enter locked rooms) are not feasible. A first step is to reduce the weight of the particles which violate these constraints to zero. However, as office buildings are highly constrained environments, this is a very inefficient use of particles, a wide majority of which will be discarded at every step. A better approach is to adjust the mobility model to account for the constraints and only generate those movement patterns which are compatible with the constraints.

Let us now consider how a priori information can be used in the prediction step. A simple Gaussian distribution would model a target which is randomly jumping around: a highly unlikely scenario for a human or robot. Our first step is to assume that the target is engaged in a *purposeful movement*. To model this, our mobility model will have a deterministic component which reflects the purposeful movement of the target, and a random component which reflects the uncertainties in the execution of the movement,

observation errors and uncertainties in the modeling of the persistent components.

Both of these components can be modulated by our knowledge of the target and the environment, and the way to formalize this knowledge will be a recurring theme of our research.

For this first phase of the work we have used an inertial estimation model of the deterministic component - which considers the physics and, in some sense, the psychology of the target but it does not consider the impact of the environment.

For the probabilistic component we considered two models. In the *Gaussian dispersion* model we superimpose Gaussian noise over the location values. This has desirable mathematical characteristics, but it leads to unrealistic and sometimes physically impossible trajectories (as the emergent trajectory might have an unrealistically large acceleration). The second model, *random wandering* introduces a random component in the speed of the movement, with a separate value for angle and magnitude.

These models are described in Chapter 6.

Chapter 7 describes a series of experimental studies with the algorithms developed. The question we are investigating is whether information rich approaches (the consideration of physical and psychological constraints of the target and the environmental constraints yield performance benefits or whether the simpler, information agnostic approaches are more performant.

Our experimental results show that:

(a) the use of physical and psychological models yields a major improvement in accuracy, especially for low number of particles (up to 75% improvement compared to Gaussian diffusion).

(b) the use of environmental information in the weight update step yields an accuracy benefit which cannot be matched by the information agnostic approaches even with a large number of particles (about 20% improvement in accuracy).

(c) the use of the environment information in the mobility model (prediction step) allows the system to reach the accuracy plateau with a significantly smaller number of particles (in our experiments, 30 particles instead of 70).

1.6 Restarting particle filters

Another direction of our work involved finding a solution to a well known challenge of the particle filter based tracking: the issue of recovering from estimation errors. Particle filters, in contrast to grid based approaches track only the probabilities of the area where there is a high probability of finding the target. If the estimate was incorrect, and none (or only a very few) particles are in the correct location, the system becomes *stranded*, and it can spend a long time in a very high estimation error and uncertainty state before recovering. In robot localization, this situation is the cause of the *wake-up robot* and *kidnapped robot* problems. For our case, similar situations can appear if the tracked target suddenly takes a sharp turn which was not correctly predicted by the mobility model. This can be solved by *restarting* the particle filter using the latest instantaneous observation as a starting point (and implicitly, discarding the history encoded in the particles).

We found that a human observer watching a visualization of the observations and particles can easily detect the point where the particles become stranded. The SIR particle filter, however, does not have an automatic mechanism for this. Our goal was to design an automatic solution for the restarting of the particle filter.

The solution is based on the measurement of the Kullback-Leibler divergence (KLD), a well-known measure of the (dis)similarity of two probability distributions. The particle filter will restart if the KLD between the distribution implied by the observation and the distribution implied by the particle cloud is larger than a given threshold.

In order to integrate the restart decision in our workflow, we need to solve three individual problems: (A) the efficient computation of the Kullback-Leibler divergence between two probability surfaces described with the ziggurat model, (B) the representation of the observation informed by the a priori information in the ziggurat model and (C) the representation of the probability surface induced by the particles in the ziggurat model.

We tested our automatic restart methodology on a number of scenarios. We found that, as expected, the performance increase strongly depends on the appropriate choice

of the restart threshold. If the restart threshold is too high, the restart will not happen. If, in contrast, the restart threshold is too low, the system will unnecessarily restart even in cases where the particles could have been retrieved by the default mechanisms, thus lowering performance. We found that by experimentally setting the restart threshold to an appropriate value, significant performance increases can be achieved.

The restart mechanism and the validation experiments are described in Chapter 8.

1.7 Multi-hypothesis particle filters

The structure of indoor environments frequently presents the target with divergent choices: for instance at a T-shaped corridor intersection the target might turn to left or right, with no intermediary values possible. Such situations present a major challenge to particle filters, as the single particle cloud can not track simultaneously the two different prediction models associated with the choices.

One possibility is to make two separate hypotheses covering the two possible choices, and track them separately using the classical methods. Often, after some future observations are recorded, one of the hypotheses can be discarded as inconsistent with the observations. In other cases, such as when avoiding an obstacle on the right or left side, the two hypotheses will be merged when their associated trajectories converge.

Our approach is based on "hypothesis modifiers", which encode the choices facing the targets in specific areas of the building. The hypothesis modifier might split the set of particles into multiple clouds. Each cloud is matched with a specific hypothesis which determines the prediction model of the particles (but *not* their weight update model). The different clouds will evolve separately under their specific prediction model, each cloud providing a separate, distinct estimate to the tracking.

An experimental study shows that an estimation method based on the strongest hypothesis can significantly outperform the traditional particle filters for scenarios with divergent choices.

The description of the multi-hypothesis mechanism and the experimental study is described in Chapter 9.

Chapter 2

Related work

2.1 Introduction

Outdoor localization is a mature and well established technology. The current state of the art is based on the measurement of the Time of Arrival (ToA) [13] of radio signals from at least three or four satellites with known orbital information. This approach is used in the US-based Global Positioning System (GPS) [13], as well as in the Russian GLONASS, in the French GEOSCOPE [33] and European Galileo systems.

Satellite based systems do not work in indoor environments, where the satellite-based radio signals can not be received. A significant body of work addresses the challenges of radio-signal based localization in indoor environments [3]. Many systems use the existing wireless infrastructure (such as WiFi access points), or build a dedicated infrastructure from off-the-shelf components. These systems do not have high precision clocks. The quality of the signals is diminished by environmental conditions such as metal frames, machine noise or thick walls.

The Time of Arrival (ToA) [13] technique, used in outdoor environments, is difficult to use in indoor systems. Thus, indoor localization frequently rely on measuring other properties of the incoming signal, such as Received Signal Strength (RSS) [3, 35], Time Difference of Arrival (TDoA) [31, 16] and Angle of Arrival (AoA) [25, 26, 10].

Another difference between outdoor and indoor localization is the security aspect. The GPS satellites have some security measures, and can use encryption for the authentication of their signals. In contrast, the wireless signals in indoor localization can be manipulated by attackers. In addition, indoor localization often rely on measured quantities, such as RSS, which can not be cryptographically signed. Thus, the security

aspects of indoor localization represent a separate research area.

2.1.1 Application scenarios

The challenge of indoor localization had been addressed by a number of different research communities separated by the application scenario and the amount of dedicated hardware infrastructure used for the localization framework. We can distinguish three application scenarios.

(1) Indoor localization using the networking infrastructure. This application scenario performs localization using the existing wireless infrastructure designed and deployed for communication. It assumes no dedicated localization infrastructure in the environment or in the mobile nodes. This is the framework used by RADAR [3]as well as more recent approaches using Bayesian reasoning or similar approaches. Both instantaneous and dynamic localization had been demonstrated in this framework.

The advantage of these kind of systems is that is does not require additional investment in infrastructure, and it can use existing, unmodified equipment. On the other hand, the lack of dedicated infrastructure limits the accuracy of localization. Thus, this approach is most appropriate for cases when the main purpose of the system is networking, with the benefits of the localization being an add-on value. Examples include location-based services, location-based social messaging, targeted advertising and so on.

(2) Indoor localization with additional sensing. These systems require an additional infrastructure (for instance in the form of dedicated beacons), as well as specialized hardware on the mobile devices. Examples include systems such as Cricket as well as Angle of Arrival (AoA) based approaches. These systems can frequently take advantage of the existing networking infrastructure as well, as a backup, or additional source of location information. This approach is justified when the localization is a critical to the mission of the overall system - for instance, for tracking the location of employees in a secure facility.

(3) Robot localization. A special case of indoor localization is the self-localization of autonomous robots. Typically, the robot has an extensive set of sensors (including visual, LIDAR, ultrasonic range finders and so on), but ideally it should not rely on infrastructure support from the environment.

Robotics applications often consider the dynamic localization (tracking) of the moving robot. Another important feature of many robotics applications is that the robot must build a map of its environment. Thus one of the important research challenges is to integrate the localization with the building of the map - these approaches are usually called Simultaneous Localization And Mapping (SLAM). Many SLAM applications use particle filters to integrate input from noisy sensor readings and the imperfect movement control of the robot.

Our work positions it into the first application scenario: our experiments assume that the localization is based on a WiFi network, which is normally used for networking. Nevertheless, the localization can be helped by positioning the network access points in such a way that they improve the localization performance and having the access points collaborate in the localization (this is the approach taken by the GRAIL system). Most of our contributions are in the field of dynamic localization (tracking). While our main assumption is the localization of a human user, rather than a robot, we have found that some of the techniques used in robot localization can be successfully adapted to this case. Furthermore, certain scenarios considered in robotics scenarios (such as the case of the kidnapped robot), have analogies to situations in human user localization (the case of a failed tracking, where the localization must be started from scratch).

2.2 Instantaneous localization in indoor environments

In the following we review a series of representative papers from the instantaneous indoor localization literature.

2.2.1 RSS-based approaches

One of the earliest radio-frequency signal based indoor localization systems is the RADAR system of Bahl and Padmanabhan [3]. In this system, a mobile wireless station periodically broadcasts packets (beacons) which are collected by base stations using wireless cards whose firmware allows the packet-by-packet reading of the received signal strength. The collection of RSS values can be used to determine the location of the transmitter, if we have a model of the wireless propagation inside the specific indoor environment. The authors propose two different methods to develop this model. In the *empirical method*, they collect a number of measurements at various locations and orientations inside the indoor area, creating a database of RSS values and associated locations, which can be consulted through nearest neighbor or related techniques. In the *radio propagation method*, they consider several mathematical models of radio signal propagation, which would allow the creation of the model without the need for extensive measurements. The authors choose the Floor Attenuation Factor (FAF) model, augmented, when necessary with Wall Attenuation Factors (WAF). The experimental studies had shown that the system is strongly dependent on the orientation of the transmitter, and in general, the empirical method gave more accurate results.

The RADAR system is historically important because it set up the general methodology for RSS-based indoor localization. Later systems, including the system used for the results in this dissertation, had retained the general methodology while improving various components.

The Ad hoc Positioning System (APS) proposed by Niculescu and Nath [25] provides a method for the estimation of the positions of the nodes in an ad hoc wireless network. The assumption is that a limited number of nodes have a known position, either by having a fixed, known location, or having access to a GPS unit. The location of the other nodes are estimated by using a distributed approach, where the nodes are propagating locally measured or inferred information to the other nodes in the network. This information can be hop-count (in which case, the system must make a network-wide estimate of the distance corresponding to one hop). Alternatively, this

information can be expressed in distance units, either as distance-vector distance inferred from signal strenght, or the true euclidian distance to a given landmark. The nodes use triangulation to infer their own position based on the received information.

2.2.2 Time difference of arrival (TDoA) based approaches

The CRICKET system [31] has been designed to perform self-localization in an indoor environment. Initially, the authors aspired to build a purely radio-frequency (RF) based system. This, however, turned out to be unsatisfactory, as it was found that the RF propagation inside buildings does not match well will the theoretical models. Thus, the authors had chosen to implement a system the *beacons* are distributed in the indoor area emit simultaneously an RF signal and a ultrasonic (US) pulse. The US signal travels significantly slower than the RF, thus the receiving node can infer its distance from the transmitter based on the time difference of arrival between the RF and US signal.

While the RF signal carries information which can be used to identify the sender, the US pulse does not carry such information. It is possible that the receiver misidentifies which US pulse is associated with which RF transmission. One solution would be for the the beacons need to coordinate among each other. The CRICKET system, however, had chosen to acti without a centralized control system, relying on system parameter tuning and inference to minimize the false correlations.

In the CRICKET system, the transmitting nodes are part of the infrastructure, while the receiver calculates its own position, which it might withold from the infrastructure. This maintains the privacy of the users, at it might be a more advantageous for certain applications. Overall, CRICKET is not a location tracking, but a location support system.

A system based on similar technical foundations has been implemented by Harter et al. [16]. This system also relies on the measurement of the time difference of arrival between an RF and an US signal. In contrast to the CRICKET system, the transmitter components are carried by the mobile users, in the form of small devices called *bats*. The

environment is instrumented with receivers which are connected to a centralized system. Bats newly entering the system register their presence with the infrastructure. The registered bats are actively polled by the infrastructure. The infrastructure unregisters the bats which had left the area (or are no longer active).

In contrast to the CRICKET systems, the system based on bats is centralized and allows the tracking of the location of all the bats without cooperation from the user. The user will loose its location privacy - on the other hand, allows applications where the system is main customer of the location information.

2.2.3 Angle of Arrival (AoA) based approaches

Angle of arrival (AoA) based approaches take advantage of the directionality of radio signals to determine the angle from which a given signal is broadcasted. This can be combined with other information, such as received signal strength. AoA approaches, in general, require specialized hardware, such as directional antennas. An advantage of the angle of arrival approaches is that they do not require a signal map of the indoor area.

Niculescu and Nath [26] describe an approach which is based on deploying specially configured base stations throughout the indoor environment. These stations rely on VHF omnidirectional ranging (VOR), are customized to measure both signal angles and ranges. The hardware of these stations relies on attaching a rotating omnidirectional antenna to a wireless access point. Using the information collected by these base stations, the authors describe several techniques for the calculation of transmitter locations. These technique differ in whether the system uses (a) the measured angles (in discrete, distribution or quantized form) and (b) the measured ranges. The system also allows the integration of the measurements coming from multiple stations, according to a simple rule which assigns less confidence to information arriving from more distance base stations which have a higher error.

A more complex method of integrating the information arriving from multiple stations is provided in Elnahrawy, Francisco and Martin [10]. The angle or arrival information acquired from multiple based stations is integrated using a Bayesian network, which considers separately the X and Y coordinates of the localization. The approach had been validated using both real world and synthetic data sets.

It does not have the requirement of having a signal map for the floor and with the aid of directional antenna it finds the strongest RRSI which is concluded to be the correct direction of the device's position. The accuracy is said to be comparable to other techiniques without the extra overhead of sampling.

2.2.4 Recent trends

Many recent papers in localization research are trying to improve on the accuracy of previous approaches, solve the problems of practical implementation and apply the localization models to emerging technologies, such as smartphones, shown in Roy et al. [34] for example.

An direction in localization research is the development of fully implemented architectures for practical localization. One such example is the Horus location determination system (Youssef and Agrawala [45]) which relies on 802.11 WiFi signals and a probability map of radio signal strength. To allow for RSS measurements, the authors have developed custom drivers for the Lucent Orinoco network interface cards both under Linux and Microsoft Windows. The approach is to generate a signal map in an offline step, and then gather signal strength readings in real time. Horus uses an autoregressive model to capture the correlation between different samples from the same AP.

There are certainly some research work that employ approaches not involving the use of RSS measurements such as PinLoc described in Sen et al. [36]. The idea behind PinLoc is to utilized information on the physical layer through the assumption that multipath signal elements arriving at a specific location would consists of phase and magnitude values that are unique to that location. Authors describe that the use

of CFR (channel frequency responses), designated by 30 complex numbers versus the single real number based RSS value, yeilds to more information for the system to deploy in the localization method.

As we have seen, many approaches, including ours, use precomputed radio signal strength maps as a basis for localization. The real time measurements will be evaluated in the context of these maps. One of the challenges of this approach is that many indoor environments go through configuration changes, which affect the signal propagation. For instance, different models must be used for an office floor during daytime, with many doors open and workers present, versus the same office floor at nighttime or on weekends. Yin, Yang and Ni [44] propose an algorithm called Location Estimation using Model Trees (LEMT) which is able to reconstruct the radio map at a specific point in time based on signal strength readings at reference points. This way, the system can adapt to various environment states without the need to rebuild the radio map for each state.

With the emerging popularity and ubiquity of smartphones a recent research direction covers the problem of distributed location estimation using cellphones. Most recent cellphones have a GPS module, but this usually does not work in indoor environments, where users spend the majority of their time. Banerjee et al. [4] describe a system called Virtual Compass. In this system, which does not rely on the infrastructure, cellphones rely on their nearby peers and the shared radio technologies (e.g. WiFi and Bluetooth) to build a neighbor graph, which describe their relative spatial relationship. The system had been implemented on Windows Mobile 6.0-based cellphones. The source of location information was radio signal strength measurements from multiple radios. It was found that the system provided sufficiently good relative positioning, with an average error of 0.9m. The authors conclude that the obtained accuracy is sufficient for most location aware and social network-type applications, which rely on relative rather than absolute position. However, it was found that the technique is expensive in terms of energy consumption and present significant security and privacy problems.

One of the works Sen et al. [36] describes.

2.3 Dynamic localization (tracking) in indoor environments

In the following we describe work which deals with dynamic localization (tracking) in indoor environments. Many of these approaches use particle filters to integrate the results of observations made in time.

One of the earliest papers which introduced the idea of particle filters (also known as Monte Carlo methods) in the context of indoor localization is Deallaert et. al [8]. The authors point out the advantages of particle filters versus Kalman filters (the ability to represent multimodal distributions) and versus grid-based Markov localization (efficiency and lower memory requirements). The paper describes the workflow for particle filter based indoor localization, and apply it for the specific setting of sensor based localization for mobile robots.

While different settings, error models and information sources had been used in later papers, many of the subsequent projects, including ours, used the workflow from this paper as the starting point.

One of the most exhaustive reviews of the various mathematical techniques which can be used for tracking in a high noise environment are Arumpalam et al. [1]. The authors survey the traditional techniques which give optimal resuts given certain assumptions about the noise (Kalman filters with their variations and grid based methods). Then, the authors survey the methods which can be used when the assumptions do not hold: extended Kalman filters, approximate grid based methods and particle filters. From the point of view of our work, it is especially relevant the discussion with regards to the various problems which can be encountered during the use of the particle filters, and the ways in which these can be augmented to solve the specific problems. For instance, the basic sequential importance sampling (SIS) particle filter can be augmented with a resampling step to avoid the degeneracy problem. A number of additional techniques are also surveyed, such as the resample-move algorithm and the regularization approach.

The paper considers the problem of tracking on a purely theoretical basis, with assumptions about the error distribution, but without considering concrete observation

metrics. Our work applies many of the techniques described in the paper. However, our experience with real world RSS data collected in the indoor environment shows that the complexity of the error function in an indoor environment exceeds even the most pessimistic assumptions of the theoretical papers. The error appears to have a multi-modal distribution, and in addition, have a significant fraction of outliers. Furthermore, the error does not appear to be correlated among the RSS measurements between the different access points. For a given set of four measurements, one can be an outlier, while the other three good approximations of the expected value. Most of our contributions in this thesis come in addition to the techniques described in [1].

Another paper, useful as a theoretical foundation for practicioners of particle filters is Crisan and Doucet [7]. The paper summarizes some of the convergence results from the probability theory literature for the benefits of the signal processing practitioners. One of the challenges in the theoretical analysis of particle filters is that the particles are not independent, thus classical limit theorems assuming statistically independent samples do not apply. The paper concentrates on the Sampling-Importance-Resampling particle filter, being the most widely used, and establishes results of asymptotical convergence, the convergence rate, the accumulation of error in time as well as considerations about large deviations.

One of the earliest papers which apply particle filters to the problem of indoor localization is Vlassis et al. [41]. In this paper the authors consider the problem of tracking the position of a robot from vision data. The authors start with the auxiliary particle filter of Pitt and Shephard [30], which moves the sampling to the predicted location of the particles, in effect providing a one-step lookahead. This approach had become the standard practice in particle filters for localization, our system implementing it as well. The techniques of restart, described in Chapter 8 are actually necessary to avoid the problems when the prediction of the auxiliary particle filter is incorrect.

Due to the fact that the system described in [41] operates in the circumstances of the information source being a robot vision system (being equated to a very high dimensional data) the observation model can not be expressed in an analythical form. Instead, the authors rely on a nearest neighbor-based conditional density estimation in

the inverted model. These considerations are not applicable for our case, as the RSS values are of low dimensionality (4 or 5 signals corresponding to the access points).

Milstein et. al [21] perform indoor localization of a mobile robot based on the laser range finder readings. They notice that in certain highly symmetric environments some positions of the robot might have equal probability, which can lead to a localization failure. The approach proposed is to assign the particles to a set of randomly created clusters, which are independently tracked. The location estimate corresponds to the most likely cluster. Particles cannot move from one cluster to another, but new clusters might be created if the observations do not match with the existing clusters. The experimental study shows that the method outperforms a traditional particle filter in a number of scenarios involving indoor environments with symmetric features.

Particle filters had been also found useful in one of the canonical problems of mobile robotics. Truly autonomous mobile robots must be able to operate in unknown environments. The robots need to know their own locations, but they also need to simultaneously build a map of the landmarks in their environment. This is the problem of Simultaneous Localization And Mapping (SLAM). Early SLAM algorithms used Extended Kalman Filter (EKF) for incrementally estimate the location of the robot and of the landmarks.

Unfortunately, these approaches turned out to be of quadratic complexity in the number of the landmarks in the map, which limited their scalability to complex indoor environments. The particle filter-based FastSLAM algorithm [22] had achieved a logarithmic complexity in terms of the landmarks in the environment, and it had been shown to scale up to 50,000 landmarks. The FastSLAM-2.0 algorithm [23] improved the accuracy of the mapping, and established convergence results for linear SLAM problems involving a single particle. Montermerlo et al. [24] applies conditional particle filters to the closely related problem of simultaneously localizing the robot and tracking people moving in the environment. The conditional particle filter approach is able to handle different interpretations of sensor readings based on robot's poses over the predefined map. The experimental study shows that the algorithm is able to accurately localize a mobile robot and track multiple people simultaneously, even in the presence of global

uncertanity about the robot poses.

Fox [11] describes an improvement over the baseline particle filter localization method, which relies on a variable number of particles. The application considered is mobile indoor robot localization. The approach uses the Kullback-Leibler distance (KLD) to decide on the minimum number of particles needed to achieve sufficient accuracy in a given moment. Experimental results show that the presented approach reduces the number of samples to 6% compared to fixed sampling and to 9% compared to likelihood sampling, while achieving higher accuracy.

Fox et al. [12] describe a Bayesian filter technique to estimate the location of multiple targets from wireless signal strengths in the presence of attenuation. This approach tracks each particle in the system as a separate hypothesis. The goal is to improve on approaches based on Kalman filters which are restricted to unimodal distributions. Experimental validation was performed in an indoor environment.

Gustafsson et al. [14] are considering a wider range of localization-type applications of particle filters. *Positioning* is the estimation of a moving target's own position. *Navigation*, besides determining the position, also requires the estimation of the velocity, attitude and heading of a vehicle or airplane. Finally, *target tracking* estimates another object's position with respect to the tracker's own position. Similar problems have been traditionally handled using linearized models, Gaussian noise assumptions and Kalman filters. The proposed particle filter based approach allows the consideration of scenarios where these assumptions do not hold, for instance when using a digital road map to restrict the vehicle's possible positions. The authors consider a variety of possible sources of input, including radio frequency measurements and the vehicle's sensors. The approach also allows to integrate GPS measurements whenever available.

While the scenarios described by Gustafsson et al. are mostly considering outdoor applications, the approach extensively considers external constraints, such as the road network or the terrain. Thus, many of the ideas described are applicable to the indoor scenarios considered in this thesis.

Hu et al. [19] describe a particle filter based SLAM algorithm. The considered application is the vision-based navigation of robots in an indoor environment, where colored

cylinders serve as landmarks. This leads to a different measurement and motion model compared to [24] which uses laser range scans. The experimental results show that the approach outperforms extended Kalman filter based approaches for the same robot, and it is robust with respect to the odometric noise and ambiguous data association.

Röfer et. al [32] describes self-localization methods for four legged Aibo robot dogs in a robotic soccer tournament. A vision system extracts features specific to the robotic soccer field such as beacons, goals, field lines and field wall. These features are integrated by a particle filter to determine the position of the players in the field.

Letchner et al. [20] describe a localization approach which represents the possible location of the target as a point on the edges of a 2D or 3D graph. The graphs are manually built using building information, with the edges representing corridors, paths or elevators. They propose a hierarchical Bayesian sensor model which they claim that combines the benefits of signal propagation and fingerprint based sensor models. The experimental results show that the proposed sensor model achieves a better localization than the sensor model introduced by Haeberlen et al. [15].

Seshadri et. al [37] present an indoor localization approach based on wireless received signal strength readings. The map of the indoor building is divided into cells, and in the training phase, the wireless signal strength is measured for each cell. This signal map of the floor is incorporated in the prediction step of a particle filter. Two experimental studies are presented, a walkthrough with simulated readings, and a real walkthrough. As expected, the error rate for the real walkthrough was higher due to environmental factors such as open and closed doors and people moving in the environment, which affect the signal strength readings. The experiments find that the accuracy improves with the number of particles, but the improvement levels off at around 3000 particles.

Seshadri et. al [37] present an indoor localization approach using a bayesian sampling with particle filters as the estimation method. The received signal strength values are gathered to form a signal map of the floor and incorporated in the prediction step to update the existing model in order to get the possible locations of a target. The real time experimental study using high number of particles in the orders of thousands

has shown more accurate localization results of tracking a target over the simulation study performed. There was also an increase seen in the average error caused by the environmental factors' reflection on the sensor data collected.

Howard [17] presents a method of extending particle filters to support simultaneous multi-robot localization with SLAM. The two main scenarios considered are based on whether the initial location of the robot is unknow or known a prior. In the former instance, the assumption made is that a pair will cross each other in their path, and the result of this impact would be to have their relative pose. This pose then gets assigned to the particle filter as the initial pose and all observations from both the robots is joined into a single map. Experimental study using four robots equipped with odometry and scanning laser range-finders has been done. The results show the localization performance in terms of overlay of the floor blue print and map produced by the algorithm which includes the encounters of multiple robot pairs.

Cao et. al [5] present an algorithm using omni-directional vision system with the aid of particle filter to perform localization. The mobile object uses the becon recognition and tracking as a guidence and deploys the particle filters for its nonlinear dynamic state estimation. The experimental study shows the respective pair of actual and the localization coordinates for a set of runs in which the two parameters seem to follow each other closely.

Widyawan et. al. [42] describe a modified particle filter for indoor localization which performs "backtracking", that is, it refines its own historical estimates by eliminating those particles which move in invalid trajectories later in the localization. The unfeasible trajectories are identified using "map filtering", by superimposing them to a floor plan of the building. An important component of this work is the ability to inherit the historical information of the particles through the resampling step.

Djuric, Vemula and Bugallo [9] consider a sensor network for target tracking. The sensors are *binary* in the sense that the sensors are sending a simple presence signal if a target is in their sensing range, otherwise they do not transmit. The control unit of the network uses a particle filter to track the target's trajectory, velocity and power. The authors use both the widely used auxiliary particle filter (APF) and the cost-reference

particle filter (CRPF). For the latter, the particles have a user defined cost associated with them, but they don't rely on probabilistic assumptions about the dynamic system. An experimental study shows that the localization performance of the CRFP in most scenarios closely matches the performance of the more complex APF. If fact, when the assumed distributions are inaccurate, which is a frequent occurrence in practical deployments, the CRFP algorithm performs more robustly than the APF.

A somewhat unusual approach to the problem of target tracking in WSNs is described in Ozdemir, Niu and Vershney [27]. Most particle filter based localization approaches use the filter to eliminate *sensing noise*. The authors argue that the WSNs operation in a difficult wireless transmission conditions, with imperfect communication channels. The authors propose a solution where the imperfections of the wireless communication channels are incorporated into the same particle filter which filters for the sensing errors.

While early papers on particle filter based localization have set up the general framework and provided proof of functionality, recent research had provided ongoing improvements in the accuracy of filtering, and the computational complexity of the algorithms. An example of this class of research is Baggio and Langendoen [2]. The authors are considering the problem of localization in a mobile wireless network. The assumption is the presence of *mobile anchors*, wireless nodes which are broadcasting their own accurate location. The approach improves on the localization technique of Hu and Evans [18], but it makes it more lightweight by relying primarily on the information from the one-hop and two-hop anchors, and restricting the area from which the node draws samples to a small box. The authors have found that the localization accuracy have improved with an average of 22%, while the processing time is reduced 93%.

Particle filter-based localization techniques have found application in approaches which attempt to mimic the localization techniques employed by the human brain. For instance, Siagian and Itti [38] implement a robot localization technique which is based on computer vision. The authors argue that humans self-localize in a scene by quickly extracting the *gist* of a scene, which provides an initial, coarse location hypothesis. Next,

humans refine this original estimate by locating salient landmarks. The paper describes an analogous model for a robot, then uses Monte Carlo localization, more concretely a sampling importance resampling particle filter to generate the robot's position.

Not all the localization approaches deploy particle filters. In the following we shall review several approaches which take a different path.

One research direction is *cooperative localization*, a system where multiple nodes are exchanging information to improve their own location information. Wymeersch, Lien and Win [43] describes such an approach which uses ultrawideband UWB ranging methods. The main challenge is that the system needs to schedule the exchange of information among the nodes, while taking into account the time-varying network topology and the movement pattern of the nodes. The approach uses the theory of factor graphs to perform statistical inference. The resulting algorithm called SPAWN (sum-product algorithm over a wireless network) can be considered as a generalization of previous algorithms to a network - for instance, reverting to Bayesian filtering in the case of a single agent.

Chapter 3

Instantaneous localization with intersection areas

3.1 Introduction

In this chapter we describe a new approach for instantaneous localization based on the minimal intersection areas of iso-RSS lines. This approach will also be used as the basis of the dynamic localization approaches described in the following chapters of this thesis.

Early localization systems using WLAN infrastructures were point-based approaches, where the output of the localization is a single point, representing the likely location of the device. Our approach, in contrast, returns an *area of the likely location*.

We will characterize the quality of the localization by two measures: its *accuracy* and its *precision*. We define accuracy as the distance of the point returned by the localization algorithm to the real location of the device. The precision of the measurement describes the level of confidence of the algorithm in the result.

In the *training phase* of our algorithm we create a database of iso-RSS lines. Let us assume that we want to perform localization in a building with n wireless access points. First we collect a series of measurements at a series of locations spanning the whole building. For each location we collect a vector of measurements $\{m_1, m_2 \dots m_n\}$. Using these measurements for each access point, we build a smoothed RSS surface (a radio map). We extract the iso-RSS lines of these surfaces. Note that due to the obstacles in the radio path, these lines are usually not simple circles around the wireless access points, but have a more complex shape, and frequently, they are composed of several disjoint lines (multi-lines).

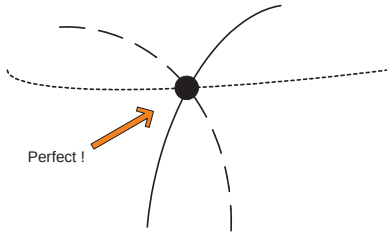


Figure 3.1: An example of localization where the three iso-lines corresponding to the three measured values intersect in a single point. This point will be returned as the location of the device.

During the *localization phase* we measure the signal strengths at an unknown location, and try to infer the location using the database of iso-RSS lines created during training. First we find the n iso-RSS lines corresponding to the measurements. If these lines intersect in a single point, this point would be the correct location, as shown in Figure 3.1. However, in practice the lines might not intersect in the same point, due to factors such as noise, measurement error, systematic error, reflections, or an imperfectly trained model. It is also possible that the data set contains one or more signals which were tempered with. Hence, the goal would be to find a measurement which is robust and not too reliant on accidental factors.

In practice, instead of a single point in which all n iso-RSS lines intersect, we find a series of points in which two or more lines intersect. One might be led to believe that the greatest density of such partial intersections would indicate the likely location of the target. Unfortunately, this is misleading, as the number of intersections depend on the smoothness of the iso-RSS lines. "Jagged" curves might intersect several times in close proximity, as shown in Figure 3.2. In the following we propose a more robust approach to finding the location of the device.

3.2 Minimum intersection area

We denote an area A through which all the iso-RSS lines pass as an *intersection area*. Naturally, a rectangle covering the whole building is such an intersection area. We are interested in finding the smallest of such areas, the minimum intersection area (MIA).

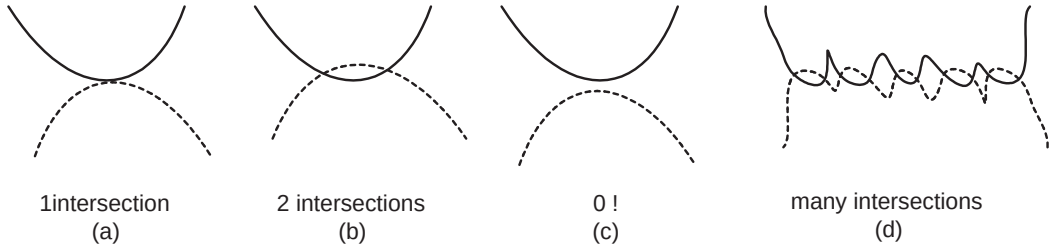


Figure 3.2: Intersections between iso-RSS lines. Two "jagged" iso-RSS lines can intersect several times along the curves (d).

In this context, an intersection is defined as an intersection between a rectangular area defined by upper right point, width and height, and a multi-line which is a series of interconnected segments.

The multi-lines are built as approximations of the iso-RSS lines corresponding to the different access points and measurement values. An iso-RSS line might be represented by multiple, distinct multi-lines.

A rectangle and an iso-RSS line is said to intersect is any of the multi-lines corresponding to the iso-RSS line has at least one point inside the rectangle.

Having n iso-RSS lines, we will call an intersection area of degree i an area which is intersected by at least i iso-RSS lines. Naturally, an intersection area of degree i is also an intersection area of degree $i-1$, but the reverse is not in general true. This observation is important in situations when one of the signals is very noisy, weak or has been tampered with. For instance, if we have a scenario with 5 signals and we find that the minimum intersection area of degree 5 is very large, but the minimum intersection area of degree 4 is much smaller, we can assume that the signal which does not participate in that area is unreliable, and we can exclude it from the measurements.

Another observation is that the minimum intersection area might not be unique, due to the complex shape of the iso-RSS lines. In general, the lower degree we consider, the larger number of minimum intersection areas we find.

Finding the exact minimum intersection area is a complex problem of geometry. We can obtain a good approximation by finding the smallest division of the measurement

area which would be the intersection area by repeatedly dividing the measurement area according to a *partitioning scheme*. The problem then becomes finding the optimal partitioning algorithm that will return this approximate minimum area.

3.3 Finding the minimum intersection area via partitioning

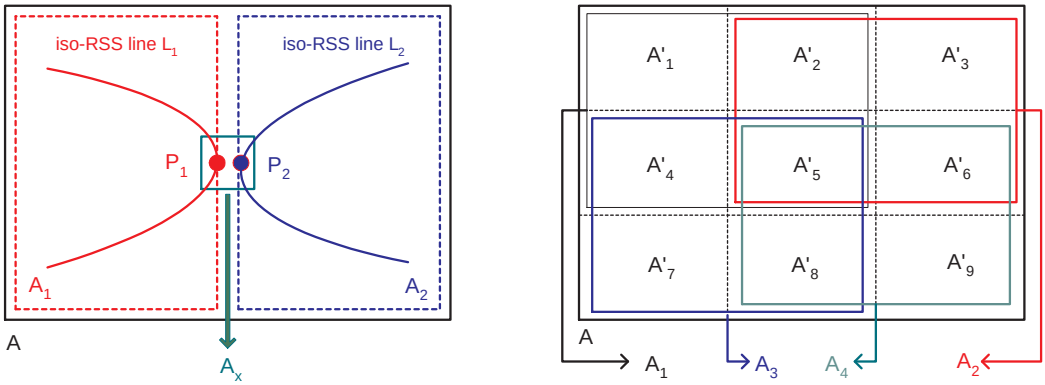
Our goal is to find the minimum intersection area (MIA) which we can define as the smallest rectangle intersected by a set of iso-RSS lines. However, the exhaustive search for the MIA is very expensive. Our approach will be to start with the largest possible intersection area, covering the complete measurement area, and gradually consider smaller and smaller intersection areas:

$$A_1, A_2, A_3 \ldots A_n$$

where $area(A_i) \geqslant area(A_{i+1})$, A_0 is the area covering the complete environment, and every new A_i is generated through a partitioning rule from the previous areas. The process stops when the partitioning rule is not able to generate a smaller area which is an intersection area. The expectation is that the partition rule will generate an intersection area which is a good approximation of the minimum intersection area. Not every partitioning scheme is able to deliver on this. Let us consider a partitioning scheme which partitions the rectangle in half along its longer dimension as in the case of Figure 3.3 (a).

As shown in Figure 3.3 (a), we have an intersection area A for the iso-RSS lines L_1, and L_2 as the starting space. This area is partitioned in areas A_1 and A_2 with neither of them being intersection areas, as each of them contains only one of the iso-RSS lines. Therefore, the algorithm will return A as the smallest intersection area found.

Let us consider the proximity points of the two iso-RSS lines, P_1, and P_2 (the points which minimize the distance dist(P_1,P_2) while P_1 element of L_1 and P_2 element of L_2). Note from the figure that P_1 and P_2 are very close, hence we can create an area A_x which is much smaller than A, which would still be an intersection area. In fact, we can easily create an area where the minimal iso-RSS area is arbitrarily small, and still, the



(a) Partitioning Rules

(b) The 2/3 partition rule

Figure 3.3: (a) An incorrect partitioning rule based on the division of the area in half at every step. Despite of the existence of a very small minimum interception area at the center of the figure, the smallest interception area found by this partitioning scheme is the whole measurement area.
(b) The 2/3 partition rule.

partition rule will insist that smallest intersection area it can find is the full rectangle A. Therefore, dividing the rectangle along the longer edge is a very bad choice for the partitioning rule.

We are looking for a partitioning rule that provides *formal guarantees* that the smallest intersection area found by the partitioning rule, will approximate the minimal intersection area within a constant percentage.

3.4 The 2/3 partition rule

As shown in Figure 3.3 (b), let us consider an area A, with a secondary partitioning into 9 identical, disjoint rectangles, by subdividing both the height and the width of the rectangle into three equal segments. Denote these nine segments as A'_1,A'_2 ... A'_9. We will consider the primary partition of the rectangle A as to do the four identical segments A_1,A_2,A_3 and A_4 with $width(A_i) = 2/3 \cdot width(A)$ and $height(A_i) = 2/3 \cdot height(A)$. It is worthwhile to point out that these are not disjoint rectangles. Also, note that the area of the segment is 4/9-th of the area of A. Each of these rectangles is composed of four rectangles of the secondary partition, for instance $A_1 = A'_1 \cup A'_2 \cup A'_4 \cup A'_5$.

The process of subdividing the space and searching for the minimal intersection areas is as follows. We start with a rectangle covering the complete measurement space. The subdivision rectangles will have their linear dimensions as 2/3 of the original rectangle (and in consequence, an area which is 4/9 of the original area) and are created according to the rules described above.

Let us now describe the process of searching for the minimal intersection area. We maintain a queue of the areas which are intersected by all iso-RSS lines. The queue is initialized with an area which covers the complete environment. At each step we choose one area from the beginning of the queue and create its four subdivisions. If none of these are intersected by all iso-RSS lines, the specific area is a minimal intersection area and is retained in a separate list. If one or more of the subdivisions are intersected by all areas, they are put at the end of the queue and the larger area is discarded. This process continues until the queue is empty.

At the end of the process the list of the minimal intersection areas contains the rectangles which are intersected by all iso-RSS lines but they cannot be partitioned further. The smallest rectangle in this list (which, due to the way in which the process works will be always the one which is inserted last) is the minimum intersection area.

During this search process, we will also keep track of which are the iso-RSS lines which had prevented the further partitioning of specific rectangles. If these lines are consistently associated with a specific access point, it can be interpreted as a sign that the signal from that access point is unreliable or has been tampered with.

The question which remains is how close is the approximate minimum intersection area computed with this algorithm to the theoretical optimum? We find that any area of the size equal to the areas $A'(1) \ldots A'(9)$, but shifted to an arbitrary position inside rectangle A will be contained fully by one of the areas $A(1) \ldots A(4)$. What this means, is that in the worst case, the partitioning scheme yields an approximate minimum intersection area two times the linear size of the absolute minimum intersection area.

3.5 Generalization: the (n-1)/n partition rule

The approximation provided by the 2/3 partition rule is still a relatively loose bound. The question then becomes whether we can devise a partitioning rule which obtains an arbitrarily tight bound.

We can generalize the 2/3-s partition rule into the (n-1)/n-partition rule in the following way. Let us consider a secondary partition into n^2 disjointed rectangles. The primary partition will be still in four rectangles, each of size $(\frac{n-1}{n}w, \frac{n-1}{n}h)$. We can show that by increasing the value of n, we can obtain an arbitrarily close approximation. Note, however, that the complexity of the algorithm also increases with n.

3.6 Using minimum intersection areas of variable degrees to identify unreliable signals

We have already mentioned that by comparing intersection areas of various degrees we can identify signals which are unreliable (for instance, because they are too weak or they have been tampered with). Let us consider the four iso-RSS lines in Figure 3.4. We note that the MIA of degree four is much larger than the MIA of degree three. The reason behind this is that iso-RSS line 4 is much farther than the other lines. We can speculate that "there is something wrong" with this line, for instance, it was tampered with. In this case, by removing this line and considering only the remaining, trusted signals we can have a much smaller MIA, corresponding to a higher precision and, if our hypothesis was correct, also a better accuracy.

We will be using this finding later in our experiment to show how localization results can be made more precise by eliminating unreliable signal vector readings. More generally, if

$$area(MIA_{r_1 \ldots r_n}) \gg area(MIA_{r_1 \ldots r_{i-1}, r_{i+1}, \ldots, r_n})$$

then the signal r_i should be considered unreliable and should be removed from the localization algorithm.

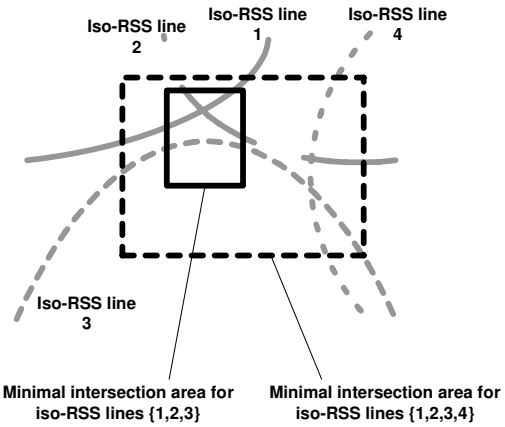


Figure 3.4: An example of finding minimum intersection areas for 4 iso-RSS lines. The iso-RSS line 4 is unreliable, thus the 3-rd degree minimum intersection area is much smaller than the 4-th degree minimum intersection area.

Chapter 4

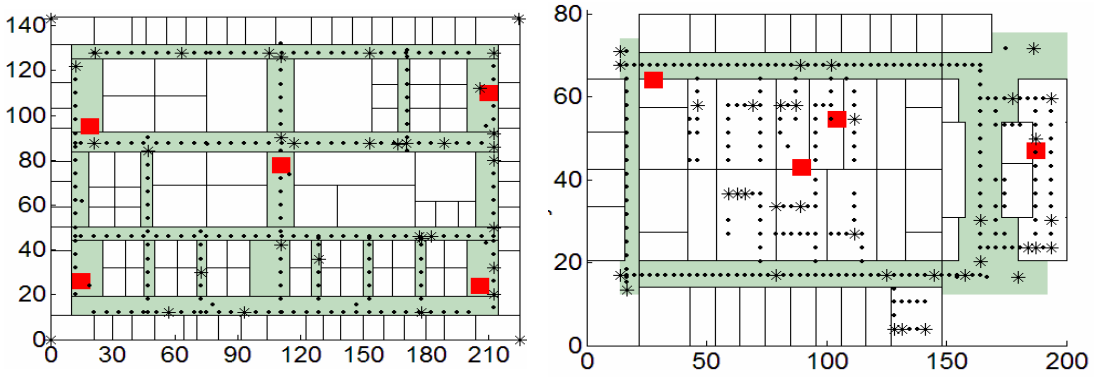
Experimental study: instantaneous localization with intersection areas

We tested our approach through an implementation which uses a combination of Matlab and Java code. In the training step we create a grid surface for the RSS field of each AP. The grid surfaces are created as an interpolation of the measured points which are used as the training data. This is done using a triangle-based cubic interpolation of the data points, through the `griddata()` function in Matlab 7. We found that the errors in the data set lead to a very "jagged" surface, therefore we performed a second set of smoothing using a bi-cubic interpolation through the `interp2()` function. This essentially involves a re-sampling of the surface at a lower resolution, and then re-scaling the resolution of the surface. We have found that a re-sampling at a resolution of 20ft yields the smoothest surfaces, but a less aggressive smoothing, at 5ft, had in fact yielded better localization. An automatic way to determine the right amount of smoothing is future work. The resulting surfaces are saved as the training data of the localization algorithm.

Whenever the algorithm needs to identify the location of a point, it will compute the five iso-RSS lines corresponding to the measurements from the five APs. The iso-RSS lines are the isolines on the previously saved grid surfaces.

4.1 Experimental data sets and an example localization

We have used two data sets collected from two different buildings. Our first data set contains five APs which were positioned in the four corners and the center of a research laboratory (see Figure 4.1-a). There were 254 readings from twenty rooms. The second data set was collected on the third floor of the CoRE building at Rutgers University



(a) Research Lab - (225 x 144 feet) (b) CoRE 3rd Floor - (200 x 80 feet)

Figure 4.1: Floor maps of (a) a research lab and (b) the third floor of the CoRE building, showing the position of access points (APs) and points where signal strengths were collected.

with four APs. There were 286 readings taken from hallways and rooms, following a largely uniform distribution.

Let us start with an example from the first data set. Recall that there were five APs in the first data set, so given an RSS-vector which consists of five signal strength values, each from a different AP, our goal is to find out which location these RSS readings belong to. Let us now see what we can expect on a successful localization. Taking the measurements at the location A=(153,120) and running the localization algorithm we obtain an estimate for the location a rectangle with the center at (154.296875, 120.703125), with an area of 9.1552734375 square ft for a five line intersection requirement. This is a very good approximation of the correct value of (154, 120). The output is represented with the iso-RSS lines and the intersection areas. The high quality of the approximation is due to the fact that this point is near the center of the area, where the signal from all the APs is comparatively strong.

Next, we examine how having factors such as noise, bias or systematic errors would affect our result. We have seen that if we can recognize that one or more of the AP readings are unreliable, we can exclude them from the algorithm, and thus obtain a better estimation. To mimic a systematic error, we have artificially changed the RSS values from one of the APs and run the algorithm on these values. To run this test, we

have again used node A from the previous example and changed AP3's reading by 3dB. On these values the program returned the estimate point as (148.4375, 117.1875), with an area of 149.484375 square ft for a five line intersection requirement. Clearly, this is not a very accurate result. Having such a large returned area indicates that there can be an error in the readings, that is throwing us off. Hence, we should perhaps eliminate this reading. We re-ran the algorithm with a four line intersection requirement. The program returned the estimate point as (154.296875, 120.703125), with an area of 9.1552734375 square ft for a four line intersection requirement with the returned area missing AP3's iso-line.

4.2 Errors and error estimation on the research lab dataset

We have run a series of experiments using both of our data sets. We have measured the absolute and the estimated error of the data. We define as the *most probable location* based on the five line intersection area as the center of the smallest of the candidate intersection areas. If the area has the height h and width w then the estimated error is equal to the semidiagonal of the intersection area:

$$\epsilon_{estimated} = \frac{\sqrt{h^2 + w^2}}{2} \tag{4.1}$$

The *absolute error* of the measurement is the Euclidian distance between the most probable location and the (known) real location of the point.

There are two objectives we need to keep in mind when performing localization: (a) minimize the absolute error and (b) improve the precision of the estimation, while ensuring that the estimation is higher than the absolute error. Note that in a real deployment scenario the absolute error is not measurable.

4.3 Experiments over a large set of points from the two data sets

In this experiment, we have run both the data sets with the five and the four line minimal intersection area algorithm for a large set of points in order to identify outliers, unreliable signal strength values and attempt to improve the cumulative distribution

Table 4.1: Improving the quality of the localization by identifying bad RSS readings.

	"poor" points	"poor" and "maybe poor" points
Number and percentage of bad localizations after the initial phase (5-th degree minimum intersection area)	11 (7.3%)	16 (10.6%)
Estimate of how many bad readings are due to one unreliable iso-RSS line (based on human visual observation)	8 (5.3%)	can not estimate
Number and percentage of points with previously bad localization which were improved by removing the unreliable iso-RSS line (thus using 4-th degree minimum intersection area)	8 (5.3%)	10 (6.6%)
Number and percentage of bad localizations, after removing the unreliable iso-RSS lines	3 (2%)	6 (4%)

function (CDF) of precision by means of returning a smaller area. Table 4.1 shows the statistics for the research laboratory data set.

Most of the localization CDFs look similar in shape as our CDF of precision, where there is a tail of 5–10% for most of these algorithms. The goal is to decrease the value of this tail as much as possible.

In this scenario, we ran the localization for 150 points out of the 254 total available points in the research laboratory data set. By looking at the maps of the five line intersection areas we performed a manual classification of the quality of data. We defined 11 points as "poor" when either the intersection area was of the size of the whole environment, or the point was outside and far away from the intersection area. In addition, we classified 5 points as "maybe poor" where the point was outside but very close to the intersection area.

The algorithm was ran with four line intersections. We have considered two cases. In the first case, the points initially marked as uncertain of being poor were in fact not

Figure 4.2: Example localization. Iso-RSS lines, 5-th and 4-th degree intersection areas for "best" localized points. Research lab data set. The minimum intersection area is basically a single point.

poor, so they were not included in the statistics. In the second case, the points marked as uncertain of being poor were actually poor, in return not included in the statistics.

Through visual observation of the iso-RSS lines we have concluded that for seven out of the eleven points the returned area can be made much smaller by removing one of the lines from the localization algorithm. This would decrease the percentage of the bad localizations from 7.3% to 2% which is a substantial improvement. Similarly, if we considered the points which are either "poor" or "maybe poor" the percentage of bad localizations was improved from 10.6% to 4%.

Figures 4.2-4.7 present one each of the best, medium, worst cases from the research lab (first of the 2-set figures) and the CoRE3 (second of the 2-set figures) data sets. The figures show all the minimum intersection areas for five iso-RSS lines (continuous line rectangles) and for four iso-RSS lines (dotted line rectangles). Even for the same number of intersection areas, there can be multiple areas with the same size.

In the best cases, since the returned area containing the real location is very small, it is hard to see either the five line or the four line intersection areas. In the median cases, we can visually identify both types of area and even though the real location is within the returned area, in some cases, the size of the box is too large, showing a

Figure 4.3: Example localization. Iso-RSS lines, 5-th and 4-th degree intersection areas for "best" localized points. CoRE3 data set. Although there are multiple minimum intersection areas, they are very small and close together.

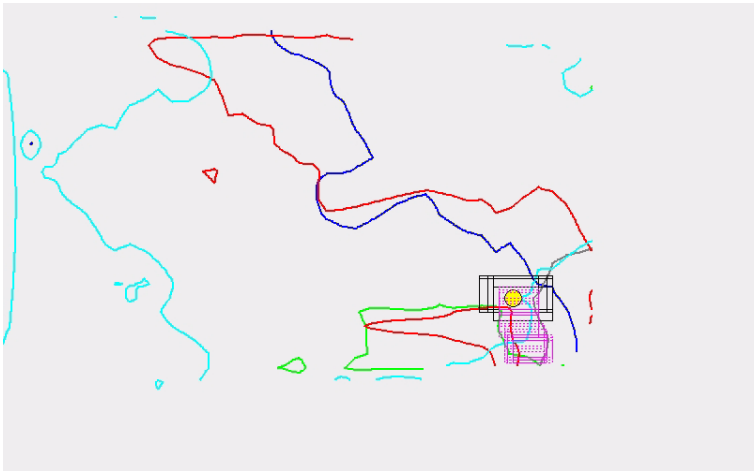


Figure 4.4: Example localization. Iso-RSS lines, 5-th and 4-th degree intersection areas for "medium" localized points. Research lab data set. Note how the 4-th degree intersection areas (dotted line rectangles) are smaller, but there are more of them than 5-th degree MIAs. The real location of the point is inside the rectangles.

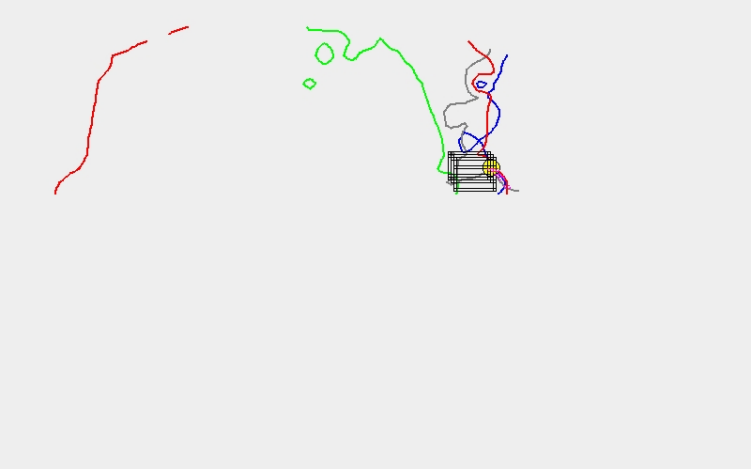


Figure 4.5: Example localization. Iso-RSS lines, 5-th and 4-th degree intersection areas for “medium” localized points. CoRE3 data set. There are multiple MIAs, but they all contain the real location of the point.

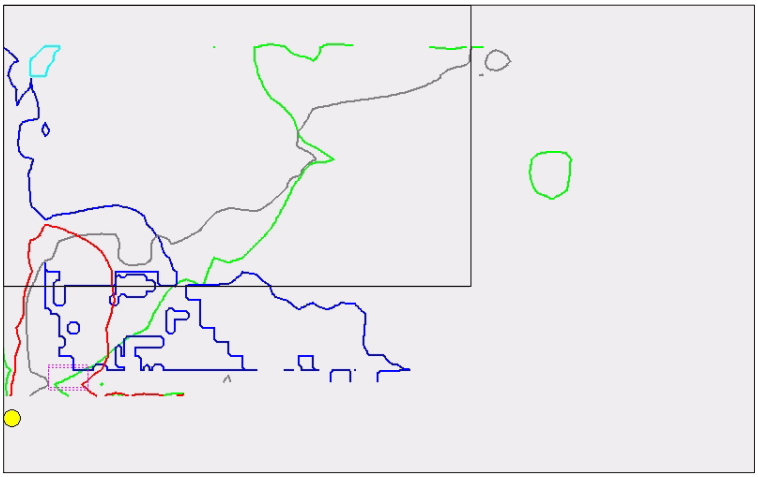


Figure 4.6: Example localization. Iso-RSS lines, 5-th and 4-th degree intersection areas for “worst” localized points. Research lab data set. The 5-th degree MIA covers 2/3 of the complete measurement area and it does not contain the point! The 4-th degree MIA, in the lower left corner is much smaller, and although it still does not contain the point, it shows its approximate location. The reason for the poor localization is the peripheral location of the point.

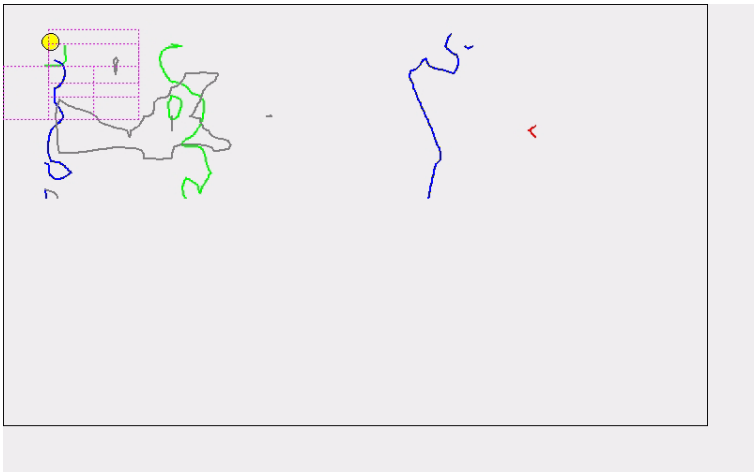


Figure 4.7: Example localization. Iso-RSS lines, 5-th and 4-th degree intersection areas for "worst" localized points. CoRE3 data set. The 4-th degree MIA contains virtually the whole area. There are multiple 3-th degree MIA's which provide a reasonably good approximation of the point, although not all of them contain it.

low precision. The worse cases clearly stood out as the size of the returned five line intersection area is close to the whole search space. However, in most cases, seven to eight out of eleven points in the research laboratory data set were improved by the four line intersection as the dotted line rectangle shown is considerably smaller in size and either contains the real location point or is very close to it.

4.4 Conclusions

This chapter described an experimental study of the area-based localization method for wireless networks described in Chapter 3. We use the existing wireless infrastructure to collect RSS readings to be later matched to an unknown location. We have quantified our method's performance through a trace-driven experimental study with accuracy and precision metrics. We have improved the precision metric from which was previously known to be 5-10% to 2–4%. We have achieved an accuracy of 3 to 5 feet (even though there are still outliers, due to bad readings in the data set). We were able to recognize bad RSS readings based on the returned area. Removing bad RSS readings improved the accuracy of the algorithm.

Chapter 5

Dynamic localization using particle filters

Dynamic localization (tracking) determines the trajectory of a target over a period of time. At its simplest, dynamic localization can be achieved using a discrete series of instantaneous localization steps. As the target is moving, its current location needs to be extrapolated from the latest localization, with the help of additional information including previous locations as well as knowledge about the environment and the target. Unfortunately, these pieces of information are frequently noisy or uncertain, and many of them are snapshots of dynamically changing phenomena. Particle filters, a probabilistic reasoning technique have been shown to be an efficient approach for integrating the various sources of information.

One area in which particle filters have been found useful is the self-localization of mobile robots. The approach proposed in this paper shares many common features with the approaches used in robotics. The most important difference is that the robots are localizing *themselves*. Although environmental factors, odometry errors and so on can introduce uncertainty in the location of the robot, there is no uncertainty about the intention of the robot, which forms the basis of the deterministic component of the prediction step of the particle filter. In our case, however, we do not know the intentions of the human targets. Although some assumptions can be made (for instance, of inertial movement), we simply do not know which direction will a human turn at an intersection, or whether it might decide to suddenly stop or turn around.

While the particle filter itself is a well tested theoretical tool, the practical expression of the available information in a format usable for the PF is a complex challenge, which led many researchers to advocate a "less is more" approach, claiming that by keeping the data model simple and the quantity of the processed information low they can

achieve a better performance than complex models which try to exploit every available information piece.

For indoor dynamic localization system we are using a sampling-importance-resampling particle filter which integrates observations received from a instantaneous localization technique based on wireless signal strengths. We are using the GRAIL system [6] to measure the signal strength and perform the instantaneous localization using the algorithm based on minimum intersection areas of iso-RSS lines described in Chapter 3. The advantage of the latter is that it provides both an estimate of location, as well as an estimate of the localization error, which allows us to convert the output into a probability distribution which represent external observations for the particle filter.

The Sampling-Importance-Resampling particle filter represents the knowledge about the current location of the target through a cloud of particles at specific locations with associated importance weights. The particles are evolved in time through a probabilistic prediction model, and their weights are adjusted depending on new observations. Occasionally, when too many particles reach a zero weight, a new set of equally weighted particles is generated through a resampling process. The estimate of the location can be calculated by considering the location of the particles in the particle cloud and their importance weights.

In this chapter we describe the general architecture of the particle filter applied to our problem domain. In the following chapters, we will enumerate three contributions (the utilization of a-priori information, the automatic restarting of the particle filter and multi-hypothesis particle filters) which improve on the default architecture.

5.1 The architecture of the system and the generic particle filter workflow

The general architecture of our system, together with the flow of control and flow of information is described in Figure 5.1. The central part of our solution is the particle filter loop composed of four steps a) prediction, b) weight update, c) estimation and d) resample and restart. We will discuss them one by one.

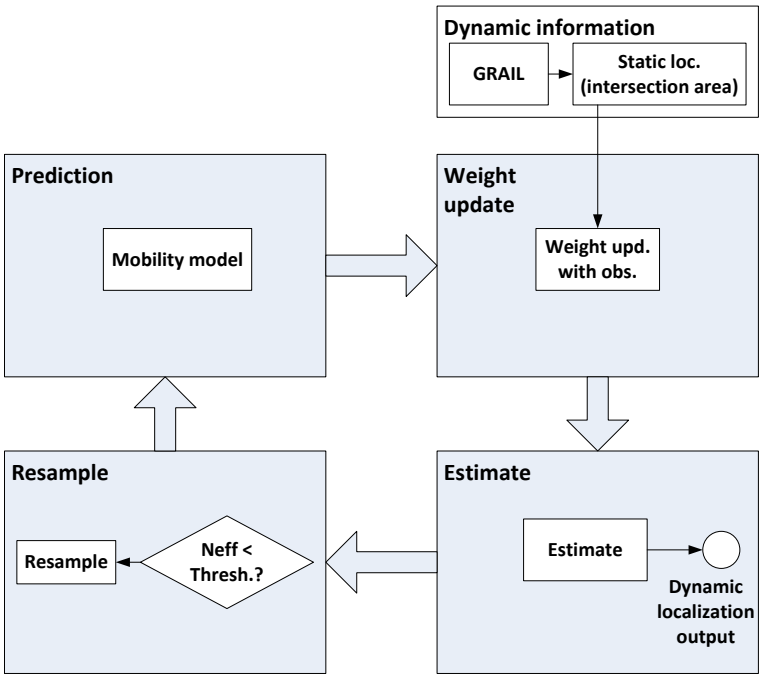


Figure 5.1: The workflow of dynamic localization using a particle filter.

Prediction. The prediction step applies a mobility model consisting of a deterministic and a random component to all the particles in the particle cloud. The mobility model depends on the a priori information about the target, and the particle history. However, as we will see in Section 6.4, the overall performance of the particle filter can be improved if we integrate environmental information in the prediction step as well.

Weight update. The weight update step is different depending on whether a new external observation is available or not. If a new observation is available, the corresponding probability is used to change the weight of the particles:

$$w_p^{'} = w_p \cdot prob(p|obs) \tag{5.1}$$

If no new observation is available, the weight update is done based on the a priori probability from the environmental information. In both cases, the resulting weights are normalized such that they sum to 1.0 for all particles.

Estimation. The estimation step determines the output of the particle filter. There

are several choices we can use here - the simplest one involves a weighted sum of the particles. Various robust estimation techniques can be applied as well.

Beyond extracting the output of the localization, the estimate is also used internally in the particle filter to update the deterministic component of the prediction model. If the prediction is inertial, as in our case, we need a list of previous locations to calculate the inertial speed. These locations are extracted from the global estimate.

Resample. During the operation of the particle filter, a natural occurrence is that the weight of certain particles becomes very small or zero. These particles will not contribute to the estimate of the location, thus they are excluded from the prediction and weight update step as well. Over time, the effective number of particles are reduced, and the performance of the particle filter suffers.

The resampling step replaces the existing particles with a new set which has equal weights. The resampling algorithm is designed such that the probability distribution implied by the new particle set approximates the one implied by the old one. The resampling technique we are using is multinomial resampling.

Unfortunately, resampling introduces its own errors, thus it needs to be deployed as rarely as possible. In our current system we resample the particles, when the effective number of particles becomes less than $\frac{2}{3}$ of the original number of particles.

The last step of the particle filter loop concerns the merging of the particle set. Under certain conditions, it is advantageous to replace the particles with a new set of particles.

The first such situation is the restart decision we mentioned above. If we notice that there is a major, irreconcilable difference between the current observation and the particle set, and we trust the observation more than our particles, we might decide to discard all the particles and replace them with a new set of particles created from the sampling of the probability distribution obtained from the most recent observation. Details about this technique are available in [39].

5.2 Adapting the weight update step to take advantage of the minimal intersection area method

In the weight update step, the particle filter updates the weight of the particles, based on external information of their likelihood. If an external observation is available at the given time point, the update step uses it together with any available a prior information. If no observation is available, the update is based exclusively on the a priori information.

The main challenge of the weight update step is the representation of the observations. The canonical representation of the observation for the particle filter is a $P(X|E)$ conditional probability, describing the probability that the target is the location X given observation E. For many instantaneous localization algorithms, however, the output is only a single "most likely" location X_{ML}.

Converting this output into a conditional probability involves some sensitive choices. Representing as a single, "stick" probability is a bad choice which would eliminate all or all-but-one particle. A better choice would be the assumption of a bivariate Gaussian distribution centered in X_{ML}. Gaussian distributions are frequently used to represent probability surfaces, because of *(a)* the theoretical reason that many real world error distributions are actually good approximations, and *(b)* the pragmatic reason, that this allows us to represent the probability surface with a small number of parameters, in the bivariate case the median point *M(x,y)* and the *2x2* covariance matrix.

In our case, however, the Gaussian representation is not as advantageous, as the a priori information contains hard transitions from feasible to unfeasible areas, which are difficult to represent in the form of Gaussians or mixture of Gaussians.

In the specific case of our system, the weight update step takes its input from the minimum intersection area localization method. While a naive approach would simply feed in the most likely location returned by the instantaneous localization, the minimum intersection area method also offers an estimate of the localization error (by returning the minimum localization area). Our interpretation is that there is a certain probability that the target will be in the intersection area, and then gradually decreasing probabilities of it being in exponentially larger intersection areas centered

on the original one (we call this the ziggurat probability model).

We choose to use a computationally efficient representation which can represent probabilities with sharp transitions accurately, and can approximate smooth transitions to an arbitrary accuracy. This representation, the *ziggurat model*, represents the probability distribution as a series of rectangular areas with specific probabilities. The probabilities multiplied with the area of the rectangles are summing to 1.0. Thus, the probability at a given point is equal with the sum of probabilities of the rectangles which contain the given point. For a description of the ziggurat model see the appendix.

5.3 Comparison versus the particle filters used in robotics

Particle filter based techniques have been frequently used in localization for mobile robot applications. Our system has many common points with those systems. The major difference is that when a robot performs self-localization, the deterministic component of the prediction is based on the known intentions of the robot. The uncertainties are only in the random factors, such as slippery floors, odometry errors, imperfections in the actuators and so on.

When localizing a human target, however, we do not have access to his or her intentions, thus the deterministic component will also have uncertainties. We can, for instance, assume an inertial motion in long corridors, however, in the intersections between corridors we have at best a statistics based guess in which direction will the target turn. Human targets might also surprise us by suddenly stopping or turning around. In conclusion, our model needs to operate under a much larger uncertainty in its prediction model, and we need to handle the occasional wrong guesses.

Chapter 6

Exploiting a priori information about the environment and the target

6.1 A priori information

Some research approaches have suggested that the best results in particle filter based localization can be obtained with simple, lightweight models. For the purpose of the update step, for instance, this would involve a simple interpretation of the instantaneous localization information: we accept the output of the information provided by the localization step even if this is contradicted by our background knowledge (eg. it puts the target outside the building, or makes it cross walls). For the purpose of the mobility step, this involves a simple model of the mobility, such as Gaussian location distribution, even if such a movement pattern is unnatural or even physically impossible for the target.

In contrast to these approaches, we developed an information rich approach where the update and the prediction steps are informed by our a priori knowledge about the environment and the target.

We are considering two types of a priori information: (a) environmental information and (b) target information. Environmental information includes the floor plan of the building, including walls, unreachable or forbidden areas. This information will be represented as a prior probability distribution (efficiently represented in a sum of rectangles probability model called the *ziggurat model*), and used in the weight update step of the particle filter. Target models contain information about the likely movement patterns of the target. In case of a human target this includes both physical constraints (humans have limitations on the speed and acceleration they can achieve) as well as social constraints (certain movement, although physically possible, are not likely to be

performed on the corridors of an office building). Technically, this information should also be presented in form of posterior probabilities in the form $p(X_t|X_{t-1})$, where X_t is the location of the target at time t. In practice, however, we are describing this in the form of a *mobility model*, which accounts for the uncertainty of the location through a built-in random component. As we will see, the nature of this randomness makes a significant difference in the performance of the overall system.

6.2 Changes in the particle filter workflow

In order to accommodate the a priori information in our system, the architecture and the particle filter workflow had to be augmented. The changes are described in Figure 6.1 with the modified components being colored dark gray. First, the a priori information is maintained in the separate component of the system which provides input to the mobility model. For instance, particles need to know the locations of the walls such that they can slide along them. Second, the a priori information allows us to update the weight of a particle even in the absence of an observation. For instance, if a mobility model lands a particle outside the building when modeling a higher floor, we know that the location is of a low probability even without the need to make a measurement.

6.3 Representation of the a priori information in the update step

Technically, a priori information is represented as a prior probability of a value. Gaussian distributions are frequently used to represent such values. In our case, however, the Gaussian representation is not as advantageous, as the a priori information contains hard transitions from feasible to unfeasible areas, which are difficult to represent in the form of Gaussians or mixture of Gaussians. We choose to use a computationally efficient representation which can represent probabilities with sharp transitions accurately, and can approximate smooth transitions to an arbitrary accuracy. This representation, the *ziggurat model*, represents the probability distribution as a series of rectangular areas with specific probabilities. The probabilities multiplied with the area of the rectangles are summing to 1.0. Thus, the probability at a given point is equal with the sum of

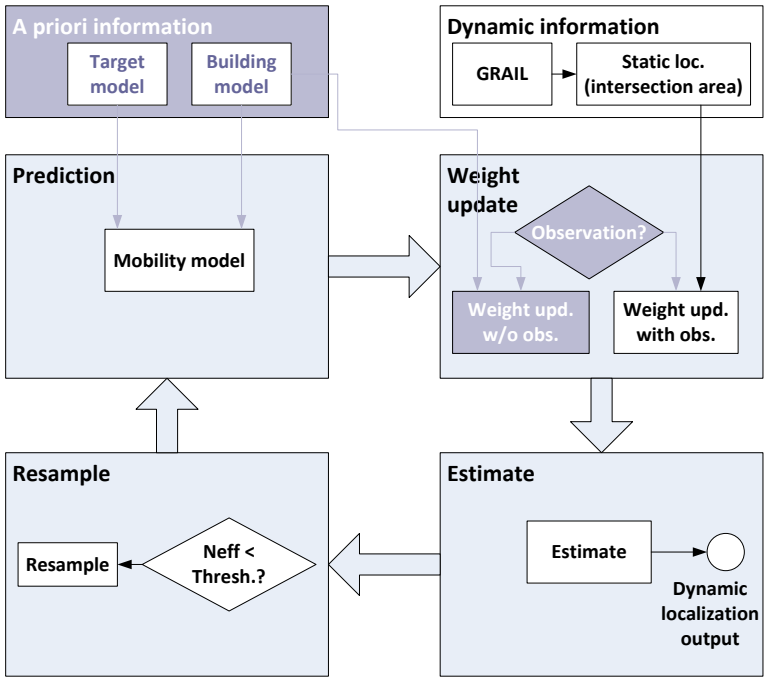


Figure 6.1: The workflow of the dynamic localization augmented to take advantage of a priori information.

probabilities of the rectangles which contain the given point.

The a priori location probability of a target is the probability, independent of any measurement, to be at a specific location at a specific moment in time. We can represent it as a 2D probability surface $p_{\text{apriori}}(target, t, x, y)$. An example of the feasible and unfeasible locations for the 2nd floor of the CoRE building is shown in Figure 6.2. This representation, determined from the building's floor plan, assumes that the feasible locations are only the corridors (that is, all the labs and offices are closed).

6.4 The prediction step and the mobility models

The mobility models of the particles describe the way in which the particles are evolving in time. The mobility model is a more convenient way to express the probability distribution of the location of the target conditional of the previous location $P(x_t|x_{t-1})$. Thus the mobility model needs to include the uncertainty which otherwise would be present in the probability distribution.

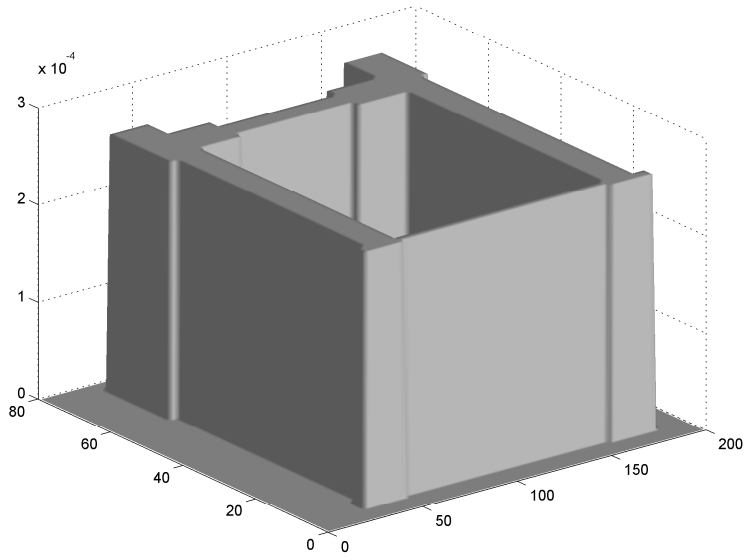


Figure 6.2: The a priori probabilities for possible and impossible locations on the 2nd floor of the CoRE building, using the ziggurat probability model.

A simple Gaussian distribution would model a target which is randomly jumping around: a highly unlikely scenario for a human or robot. Our first step is to assume that the target is engaged in a *purposeful movement*.

Thus the mobility model can be seen as having two components: a deterministic component which reflects the purposeful movement of the target, and a random component which reflects the uncertainties in the execution of the movement, observation errors and uncertainties in the modeling of the persistent component.

6.4.1 The deterministic component: inertial estimation

In particle filter based localization systems used in mobile robotics, the deterministic component of the movement is simply the desired movement pattern of the mobile robot. In our case, we do not have access to the desired movement pattern of the human observer, thus we need to estimate it based on the limitations of the target.

One of the relatively reliable approaches is based on the *inertial estimate*: we assume that the target, in absence of new decisions, will move on an inertial trajectory,

maintaining its current speed of movement. The decisions we need to make for the calculation of the inertial speed concern the time span over which the decisions are made and the source of locations we are considering. Over the source of the considered locations we have essentially three choices: we can consider the locations of the particles, the locations of the observation and the estimated location over the whole particle filter.

Using the historical estimate over the particles historical location would allow each particle to make a different inertial prediction. However, the problem is that this way inertial prediction will be completely independent from the observations. Remember that the observations do not affect the location of the particle, only its weight. Essentially, such a system would only introduce an inertial movement over the random component of the prediction model.

Using the location implied by the observations would allow us to use a common deterministic component for each particle (with significant performance advantages). A problem, however, is the quality of these observations. If we have frequent observations such that the space traversed by the target between two observations is smaller than the sum of localization errors, we can have a situation where the inferred inertial movement vector is pointing exactly in the opposite direction. We can, naturally improve on this by performing a filtering over the history of observations. However, this second filter, for speed values, would be in addition to the existing, location based filter which is our main goal.

Finally, the third choice, which we used in our system, uses the already filtered location values of the current particle filter to perform the inertial estimate. The velocity estimate will be based on the output of the particle filter and will be computed using the following formula.

$$v^{'} = \alpha v + (1 - \alpha)[L_{est}(t) - L_{est}(t - 1)] \tag{6.1}$$

where $t - 1$ is the value of the previous observation.

One of the problems with the inertial estimate is that in an indoor environment

inertial movement is limited by obstacles, the shapes of the corridors and the specific objective of the target. In a corridor running on the periphery at a building and turning at a corner, the inertial estimate is actually the worst choice, because it will throw all particles outside the building (this is actually the case for the floor map we are considering).

Naturally, a high level prediction model might be able to account for traffic models on the corridors, human decision models etc. We can, however, do a simple modification, which resolves most of these problems: if the predicted future location of the estimate is unfeasible, restore the inertial estimate to zero.

The Formula 6.1 performs an exponential smoothing over the estimates over a time. Unfortunately, the disadvantage of exponential smoothing is that it starts up slowly at the beginning of the simulation, or after a visualized reset. If the localization process is visualized, this phenomenon can be clearly seen as the estimate falls behind the observation at the beginning of the target, then catches up, only to fall behind again when the target changes direction in the corners.

To solve this problem, we introduce a slightly modified formula by setting:

$$\alpha^{'} = \min(\alpha, 1/n) \tag{6.2}$$

where n is the number of observations since the inertial speed estimate was restarted.

6.4.2 The random component of mobility

The random component of the mobility model accounts for the observation errors, uncertainties and imperfections in the mobility of the target, as well as uncertainties in the deterministic component of the movement model.

The random component is important because while the persistent component is responsible for the movement of the weight center of the posterior probability surface, the random component is responsible for its actual shape. The random component itself is significantly affected by environment and target constraints. Most human targets perform purposeful movement, for instance by moving from a room to another through

a corridor. While the target might randomly deviate from its purposeful movement, these deviations still need to conform to the laws of physics (the target cannot change direction instantaneously, nor pass through walls) and, in most cases, the target will also be limited by psychological and social constraints - for instance it will not repeatedly hit and bounce back from walls.

In the following we describe a series of models of the random component which we have developed, starting from the uninformed Gaussian dispersion to increasingly more informed models.

Gaussian dispersion

The simplest choice for the random component, frequently used in particle filters, is Gaussian dispersion. The new position of the particle is determined by the deterministic speed and Gaussian noise added to the x and y coordinates:

$$x(t + \Delta t) = x(t) + v_x \Delta t + \varepsilon_x \tag{6.3}$$

$$y(t + \Delta t) = y(t) + v_y \Delta t + \varepsilon_y \tag{6.4}$$

where ε_x and ε_y are drawn from a normal distribution $N(0, \Gamma)$.

The problem with the Gaussian dispersion model is that while it is an accurate representation of observation error, it is typically not a good representation of uncertainties and the imperfections in the mobility of the targets. Gaussian noise is *historyless*, it assumes that the divergence from the deterministic component can change in every iteration of the particle filter. Figure 6.3-a traces the movement of five independent particles using the Gaussian dispersion over the span of 1000 time steps of 1ms. The "jagged" trajectories are characteristic of the model.

Our knowledge about the nature of the target (a moving human), tells us that this model is unrealistic. A human target, or a robot with an equivalent weight can not physically enact the accelerations necessary for this trajectory. Note that this model would be feasible for the tracing of the red dot of a laser pointer.

Another problem is that the Gaussian model is dependent on the update rate of the

particle filter. The trace in the figure was obtained with a relatively fine tracing rate of 1ms. Using a larger, 100ms sampling rate, we would obtain an equally jagged, but significantly different trajectory in which the small "shake-like" movements are replaced by large "jumps". Neither of these are realistic models of human targets moving in an office environment.

Random wandering

Let us now try to develop a model which can generate trajectories which can be interpreted as a realistic hypothesis of the movement of a human target. To achieve such a model, we need to satisfy several conditions. We need to keep the movement of the individual particles within the bounds of the physical possibilities of a target with the weight of a human. This implies that the random component of the movement needs to be also subject to the laws of inertia. We also prefer a model which does not give significantly different trajectories at different sampling rates.

The proposed *random wandering* model assumes that the deterministic component of the partical movement is summed with a $\bar{v}_{random}$ random component, which is carried over between timesteps, but is gradually changing through two separate, small acceleration noise components: the *angle noise* ε_{angle} and the *magnitude noise* $\varepsilon_{magnitude}$. The separation of these two components is justified by the fact that the heading and the propulsion system is relatively well separated both in humans and in autonomous robots.

In conclusion, the random component of the prediction is described by the following formulas:

$$v_r(t + \Delta t) = v_r(t) + \varepsilon_\Theta \tag{6.5}$$

$$v_\Theta(t + \Delta t) = v_r(\Theta) + \varepsilon_\Theta \tag{6.6}$$

$$v_x = v_r + \sin(v_\Theta) \tag{6.7}$$

$$v_y = v_r + \cos(v_\Theta) \tag{6.8}$$

Figure 6.3-b traces the movement of five independent particles using the random wandering movement over the span of 1000 time steps of 1ms. We note that the model generates trajectories which are physically feasible for the human target.

Random wandering with sliding

The random wandering method considers the physical possibilities of a target moving in an empty space, but it does not consider the limitations imposed by the indoor space, nor the ways in which a target would be likely to respond to them. In our current setup, the building information is taken into account in the weight update step of the particle filter. The mobility model generates movement patterns without considering the priori probability of the destination location, thus particles might end up traversing walls, moving into forbidden rooms and so on. In the weight update step, however, the weights of these particles will be adjusted accordingly.

Ignoring the a priori information at the prediction step, however, can create problems even if it is subsequently considered at the weight update step. Particles which had an unfeasible prediction, will have their weights reduced to zero or near zero. These particles will be *lost*, eliminated in the resampling step. While they do not introduce errors by themselves, they reduce the effective number of the particles in the filter. In addition, every resampling step introduces resampling errors.

A more subtle effect of keeping a priori information only in the weight update step is that it introduces an *adverse selection* over the particles. For instance, when moving in narrow corridors, fast particles are more likely to "stray" into forbidden areas. These particles will be lost, thus the filter will predict a target which is moving slower than the one in reality.

It is desirable to adjust the mobility model in such a way as to avoid generating particles moving into unfeasible locations. In our particular case, the main problem relates to the map of the building: how should a particle behave if the mobility model generates a prediction which requires the particle to cross a wall?

There are several immediate possible solutions. The particle can stop, that is, predict the next position as the current position. The particle can bounce back from the

wall in an elastic collision. These models are physically feasible (as long as they do not involve large accelerations), but they are socially unfeasible: human targets normally do not bounce back from walls, or stop in the middle of movement. The alternative approach we propose is *sliding* - if a predicted movement would make the target cross a wall, instead it would loose the movement along the axis perpendicular to the wall, and retain the movement along the axis parallel to the wall. This corresponds to the socially feasible movement pattern of a target avoiding an obstacle, but continuing along the length of the corridor at a path closer to the wall.

Figure 6.3-c traces the movement of five independent particles using the random wandering with sliding, with the wall structure used being the map of the CoRE building at Rutgers University. The trace is very similar to trace (b), for the simple random wandering approach, as the same random generators were used with identical seeds. One of the five particles was about to enter the forbidden area, its movement was modified with the sliding approach.

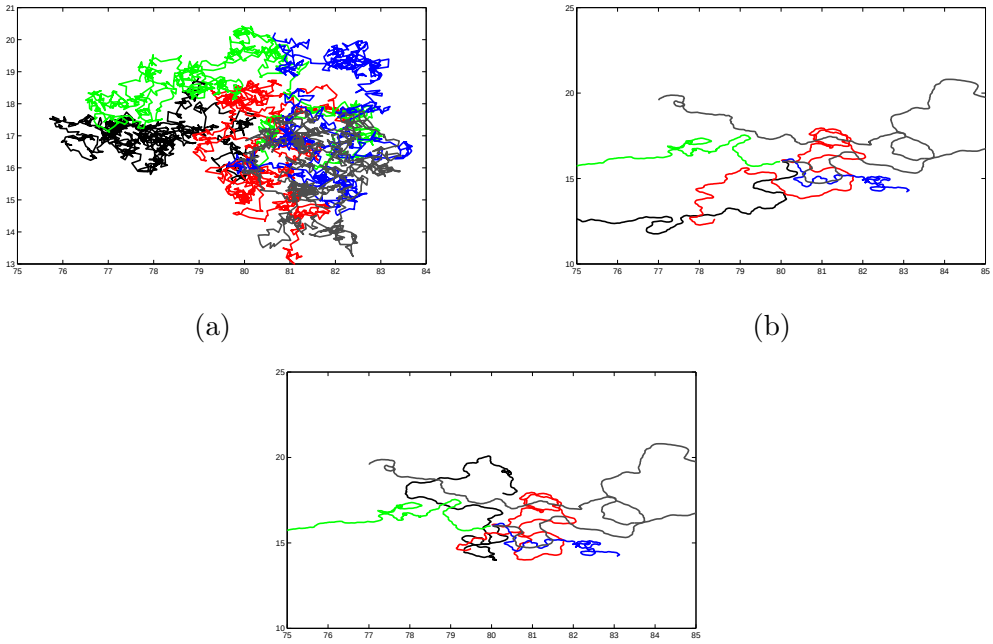


Figure 6.3: The traces of five particles over 1000 timesteps of 1ms each using (a) Gaussian diffusion (b) Random wandering and (c) Random wandering with sliding

Chapter 7

Experimental study: dynamic localization using particle filters and a priori information

In the following we describe the results of an experimental study designed to investigate the benefits of using a priori information in the particle filter localization.

For this study, we have considered a scenario involving a person moving at a normal walking pace around the corridors of the CoRE building at Rutgers University. Instantaneous localization using the intersection area algorithm is performed 24 times during this experiment. The results are fed into the particle filter workflow, which then provides the results at a much higher temporal resolution.

This scenario is a relatively good match for the particle filter model: the movement is relatively predictable, and the narrowness of the corridors used as a priori information constraints the possible errors. There are however three 90° turns in the trajectory where the inertial prediction of the movement would fail. These, together with the starting point are the cases where the particles might make a bad prediction stranding the particle filter. Whether this will indeed happen depends on the actual series of observations, their errors and precision. The particle filter can usually handle relatively large errors in the longer straight sequences, but one or more large errors in the turns might make the particle filter "miss the turn".

The original data was collected from a real time experiment using the GRAIL system. However, in order to have a larger dataset, we have generated artificial scenarios, in which we have generated the error of the instantaneous localization from a distribution which matches the cumulative distribution function of the actual instantaneous localization.

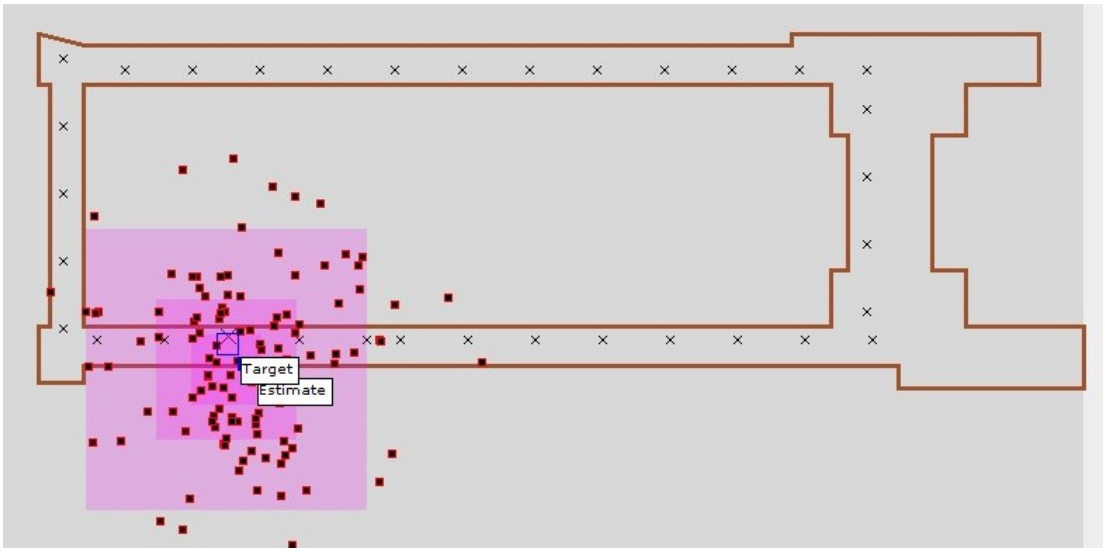


Figure 7.1: Screenshots of the distributions of the particles at the same time point t = 8000ms using Gaussian dispersion

7.1 Prediction and weight update models

For our experiments, we have considered four combinations of prediction models and weight update models, with various levels of use of a priori information. In the following, we succinctly describe these combinations. For a better understanding of the differences between these models, we have taken screenshots of the distribution of the particles at timepoint t=8000ms in the movement of the target (see Figure 7.1, 7.2, 7.3, 7.4). The visualization represents the relative weight of the particles with their colors, thus particles represented by white rectangles have zero or near zero weight, while particles represented by black rectangles have weights equal or higher than $1/n$, where n is the number of particles.

For the deterministic component, all the models are using the inertial model described in Section 6.4.1.

Gaussian dispersion with no a priori information. In this case the prediction model is based on the Gaussian dispersion model described in Section 6.4.2. Neither the prediction nor the weight update step uses any a priori information about the building or the target. The update step only uses information from the observations (the

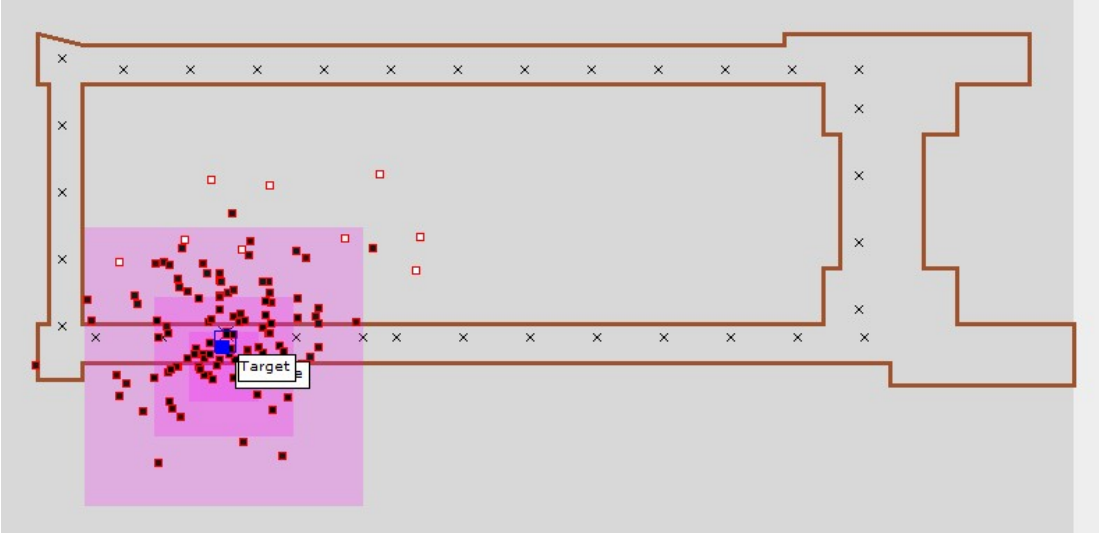


Figure 7.2: Screenshots of the distributions of the particles at the same time point t = 8000ms using random wandering without a priori information

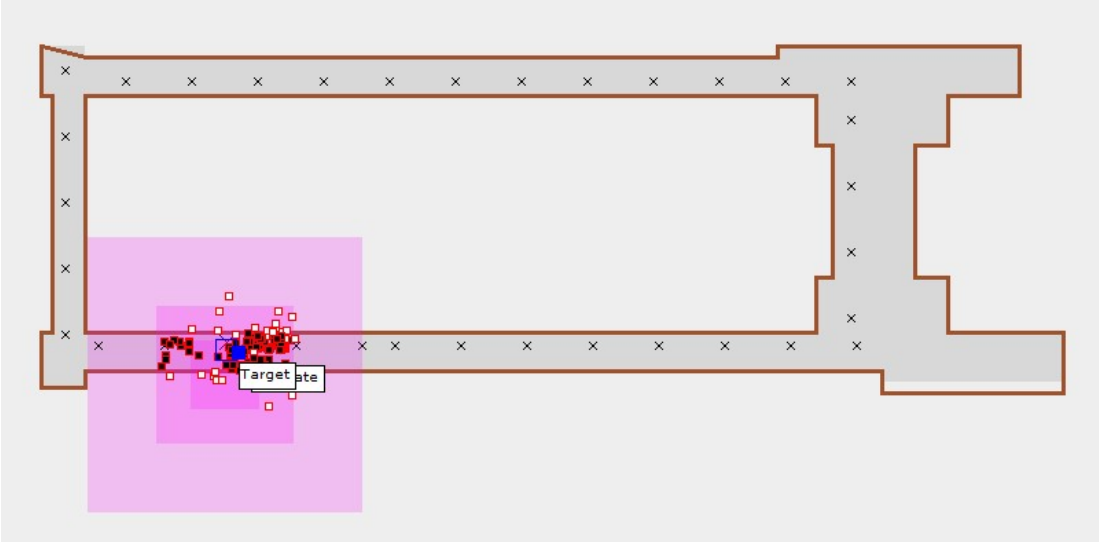


Figure 7.3: Screenshots of the distributions of the particles at the same time point t = 8000ms using random wandering with a priori information

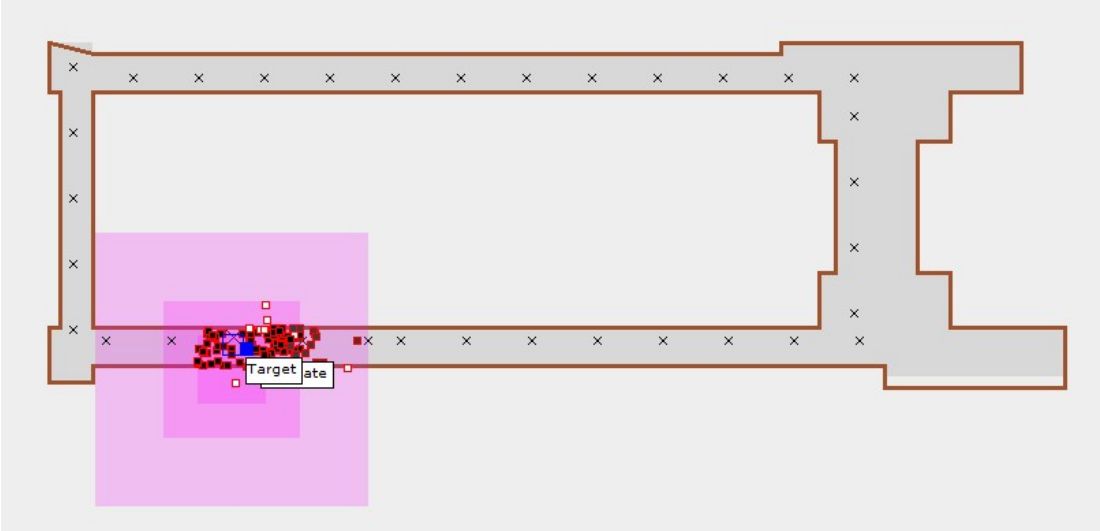


Figure 7.4: Screenshots of the distributions of the particles at the same time point t = 8000ms using random wandering with a priori information and slide

instantaneous localization steps). The screenshot in Figure 7.1 shows that the particles are distributed over a very large area, although the actual estimate is reasonably close to the correct ground truth.

Random wandering without a priori information. In this setup, the prediction model uses the random wandering model described in Section 6.4.2. Again, no a priori information was used in the prediction or the weight update step. The static snapshot of this setup in Figure 7.2 shows relatively little difference from the Gaussian diffusion setup, although from Figure 6.3 we know that the *dynamic* behavior is radically different.

Random wandering with a priori information. In this setup we are using a priori information in the weight update step, but not in the prediction step. Figure 7.3 shows a very different picture from the previous ones: particles are clustered more tightly around the estimate. Still, a relatively large number of particles are outside the feasible area, with very low or zero weight, these are the particles which moved outside the feasible area since the last resampling. In addition, we have a relatively large number of particles with zero weight inside the feasible area, these being mostly particles whose trajectory took them out of the feasible area, and then back again.

Random wandering with sliding. This setup uses a priori information both in the prediction model (by implementing the sliding mechanism described in Section 6.4.2, and in the weight update step. Naturally, the sliding method eliminates most cases of particles entering the unfeasible area (although such events can still happen, for instance if a particle heads straight to the wall, or in corners). A screenshot of this setup is shown in Figure 7.4. As expected, we see a more tight packing of the particles in the feasible area (the corridor), with only a small number of particles straying outside.

7.2 Experimental results

We have run the localization workflow over the 100 scenarios generated as described in the previous section. The number of particles had been varied from 2 to 80. The deterministic component was a smoothed inertial predictor based on the running average of estimated speed, with the weight of the new observation being $w = 0.2$. For the Gaussian dispersion model we used a dispersion speed of 0.03 m/s. For the random wandering models we used an angle noise $\epsilon_{angle} = 0.05$ and speed noise $\epsilon_{speed} = 0.05$.

Figure 7.5 shows the average distance from the ground through of the estimated location function of the number of particles for the four different prediction models. To put the performance of the particle filter in perspective, we draw the average accuracy of the extrapolated instantaneous localization (8.75ft in our experiments) as a thick grey line. If a particle filter setup does not improve on this baseline, its deployment is not justified.

As an overall assessment, all the four methods show a decrease in the localization error with the increase in the number of particles. The methods without a priori information are performing the worst. The random wandering method performs somewhat better performance for a low number of particles, but for larger number of particles the two approaches converge to the same values. The relative improvement in the Gaussian dispersion approach happens because the "jagged" movement of the particles becomes smoothed out if summed over a sufficiently large number of particles. Overall, however,

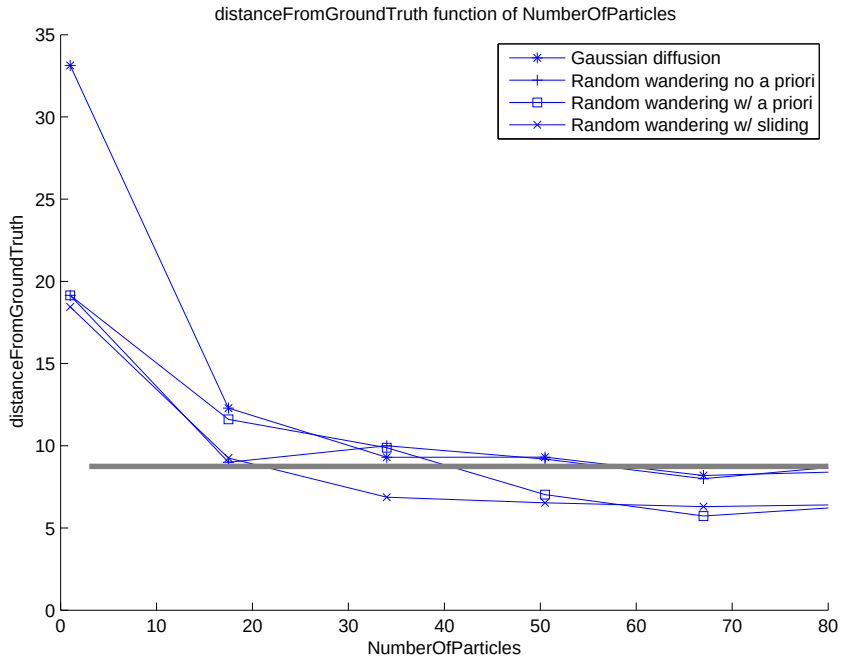


Figure 7.5: The average distance from ground truth of the estimated location function of the number of particles for the four considered setups. The horizontal gray line shows the accuracy of the extrapolated instantaneous localization.

the results show only a minor increase in accuracy over the extrapolation of the instantaneous localization, an increase which does not justify the computational cost of the particle filter. Note, though, that in our scenario we have assumed a very dense arrival time of instantaneous localization observations - when the observations are move rare, the improvements will be more significant.

The remaining two approaches, random wandering with a priori and sliding respectively, form a separate group, both of them achieving an average accuracy of about 5.5 ft, significantly better than the 8.75 ft accuracy of the extrapolated instantaneous localization. The difference between these two approaches is on the speed of the convergence: while the slide approach achieves this accuracy with about 30 particles, the approach not utilizing sliding requires 60 particles, with the associated memory and computational cost.

The reasons behind this is explained by the Figure 7.6 which shows the number

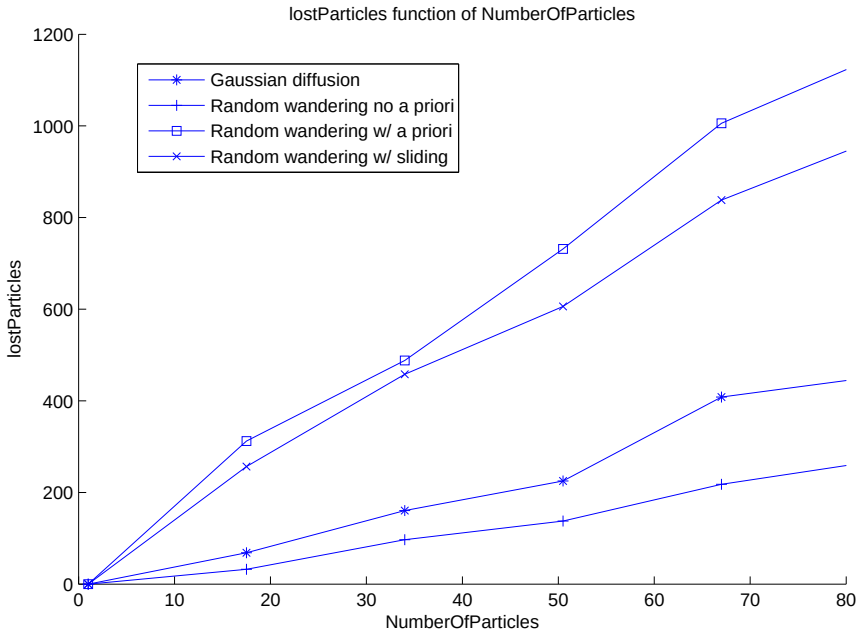


Figure 7.6: The number of lost particles function of the number of particles for the four considered setups.

of particles which were "lost" through the resampling. The sliding approach shows a smaller particle loss compared to random wandering with a priori information. This was expected, because a every sliding particle would have otherwise crossed into the unfeasible areas, and become lost. The smaller number of lost particles allows the sliding approach to achieve its best accuracy with a lower number of particles. However, lowering the number of lost particles can not be a goal in itself: the approaches without a priori information are having a much lower number of particle loss, because they only loose particles through the incompatibility with observations. Nevertheless, the smaller number of lost particles does not translate in higher accuracy.

Chapter 8

Restarting particle filters: automatic recovery from flow estimation errors

Particle filters integrate information from periodic observations, domain knowledge and predictions of the target movement. As opposed to grid-based probability estimation techniques, particle filters track only the area where there is a relatively high probability of the presence of the target. Each particle is a possible location as well as an estimate on the probability of that location; the filter provides the final estimate based on the weights of all the particles. The Sampling Importance Resampling (SIR) particle filter model periodically removes particles with low weight, and increases the density of the particles in the high probability areas.

Particle filters, however, encounter problems when the target can choose among multiple, quickly diverging movement patterns such as turns in different directions on the corridors. A particle filter can recover from estimation errors, if at least several particles are close to the correct value. If, however, all the particles are far from the correct location, the particle filter becomes *stranded* and the intrinsic methods of the particle filter (prediction, weight update and resampling) are not sufficient to recover the estimate. This situation can be easily observed in a visualized particle filter, as the particles move aimlessly in locations far from the location of the target. A manual restart of the particle filter can help in these situations, but this is, of course, not a realistic option for a deployed system.

The problem of the stranded particle filter is related to two well known problems in robot localization. The “wakeup robot” problem concerns the moment when a robot is awaken without any localization information. The “kidnapped robot” problem applies to the case when the robot is physically transported to a new location. In both cases we

have a situation where the current set of particles no longer provide a good estimate. The proposed solutions range from adding a certain number of randomly distributed particles (sensor resetting) to changing the number of particles as can be seen from Parsons's summary in [28].

In contrast, our problem is not concerned with physical transport of the target, but with consequences of complex environments and difficult-to-predict actions of the target. The main challenge in restarting a stranded particle filter is the *decision* to restart: a delicate balancing problem with a strong impact on the overall performance. In contrast, in the case of the kidnapped robot problem the decision is relatively straightforward, as there is a sudden break in the location of the robot rather than the slow accumulation of errors.

In this chapter we present an approach which allows us to automatically restart the particle filter when the particle cloud diverges too much from the observations. The method is based on the Kullback-Leibler divergence between the probability surfaces associated with the current observation and the particle cloud respectively. The restart operation replaces the current set of particles with a new particle cloud obtained by sampling the latest observation.

8.1 Changes in the particle filter workflow

To implement restart we have made a minor change in the workflow of the particle filter by adding a *restart test* before the resample test. Figure 8.1 shows the modified workflow with the dark gray shapes indicating the newly added components. The KLD-based restart test is performed after the estimate component. If the decision is to restart, the system re-initializes the particles based on the last observation. If the decision is not to restart, the control is handed over to the resample test. If the decision is to resample, the particles are recreated by resampling the current particle distribution. If there is no resampling, the current set of particles will be used.

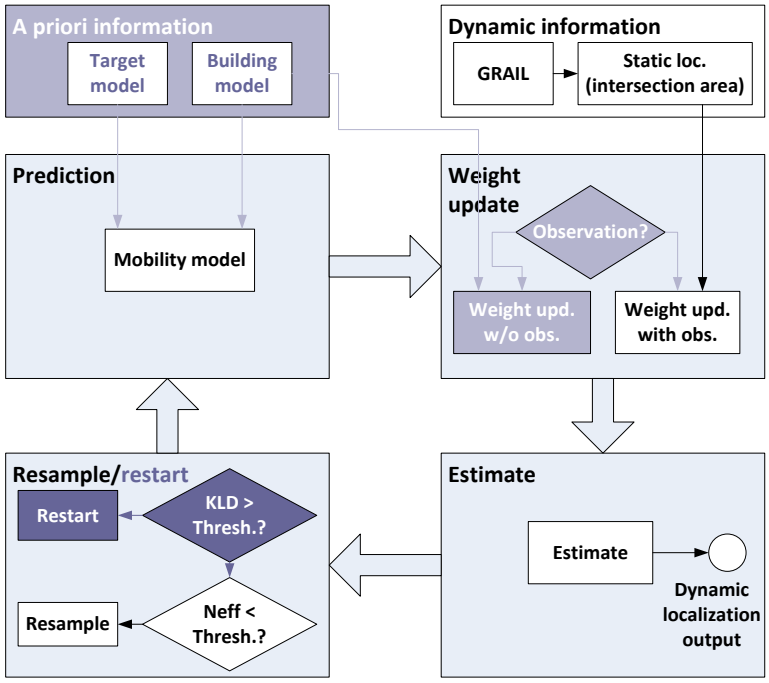


Figure 8.1: Augmenting the particle filter workflow to allow for restart.

8.2 The need for restarting a particle filter

During the operation of the particle filter, it is expected that the predictions will contain errors that are corrected periodically by the observations. Intuitively, we increase the weight of the particles which match well with the observation, and decrease the weight of particles which are unlikely given the observation. Overall, this process changes the shape of the probability distribution represented by the particle cloud, to match better the observation.

However, this technique is inefficient if the difference between the particle cloud and the observation is too large. For instance, if *all* the particles have the same very low probability in the observation, eg. $p = 10^{-10}$, their weight will be decreased in the update step, but after normalization they will have the same weight as before! In the end, it would appear as if the observation would have validated the incorrect prediction. Such a particle cloud can end up being stranded indefinitely in the low probability areas. There are several situations in a dynamic localization system which

can lead to a stranded particle cloud:

i At the beginning of the localization process, where we have a high uncertainty about the initial location and speed.

ii Whenever the target makes an unexpected change in velocity.

iii Whenever the target makes a discrete choice, for instance at the intersections of corridors.

iv Whenever the environment forces a change in the trajectory - for instance at corners.

Of course, some prediction algorithms are better than others and maintain some particles in the high probability areas even in some of the cases above. For example, a prediction model which is aware of the building structure might be able to predict correctly in case iv and a prediction model which colors particles with different hypotheses might be able to follow both choices in case iii.

A stranded particle cloud can be quickly recovered through the operation of *restart*: discard the history of the particle filter and start with a new set of particles, initialized by sampling the current observation.

8.3 The decision to restart

Formalizing the intuition behind a stranded particle cloud is nontrivial. Low probability can be caused by various reasons - for instance, if the particles are spread around in a wide open area, the individual particles will have a lower probability than particles gathered together in a tight corridor, without this being an indication of the incorrect prediction.

We propose a model to base the decision on the measurement of the Kullback-Leibler divergence KLD), a well-known used measure of the (dis)similarity of two probability distributions. The particle filter will restart if the KLD between the distribution implied

by the observation and the distribution implied by the particle cloud is larger than a given threshold[1].

In order to integrate the restart decision in our workflow, we need to solve three individual problems: (A) the efficient computation of the Kullback-Leibler divergence between two probability surfaces described with the ziggurat model, (B) the representation of the observation informed by the a priori information in the ziggurat model and (C) the representation of the probability surface induced by the particles in the ziggurat model.

8.4 Computation of the Kullback-Leibler divergence in the ziggurat model.

Let us now consider the practical computation of the Kullback-Leibler divergence in our system. The KLD between two probability distributions $P(x, y)$ and $Q(x, y)$ defined over the 2D plane is defined as:

$$D_{KL}(P \parallel Q) = \int_{-\infty}^{\infty} \int_{-\infty}^{\infty} P(x, y) \log \frac{P(x, y)}{Q(x, y)} dxdy \tag{8.1}$$

As we do not have an analytical form representation of the probability distributions, we need to approximate the KLD value by replacing the integrals with sums over the domains where both P and Q are non-zero.

In the following we describe our method for calculating the KLD between two 2D probability distributions represented as ziggurats. The first step is the calculation of the minimum non-zero rectangle of the distribution $R^{MNZ}(P)$. We define a non-zero rectangle $R^{NZ}(P)$ such that $P(x, y) \neq 0 \Rightarrow (x, y) \in R^{NZ}(P)$. A minimum non-zero rectangle $R^{MNZ}(P)$ is defined by $\forall R^{NZ}(P) \;\; R^{MNZ}(P) \subseteq R^{NZ}(P)$.

Once the non-zero rectangle of the individual distributions have been obtained, they are used to obtain the non-zero rectangle of the two distributions, $R^{NZ}(P, Q)$ which

[1]The Kullback-Leibler divergence have previously been used in particle filters for localization by Fox [11], where it was used to vary the number of particles. If a sufficiently good approximation can be achieved with a lower number of particles, then this can be used to speed up the simulation.

We are using the Kullback-Leibler divergence in a different way: if the divergence is larger than a threshold, than the prediction is considered incorrect, and we restart with a set of new particles.

is defined by $(P(x,y) \neq 0) \vee (Q(x,y) \neq 0) \Rightarrow (x,y) \in R^{NZ}(P,Q)$, and the minimum non-zero rectangle of the two distributions defined by $\forall R^{NZ}(P,Q)\ R^{MNZ}(P,Q) \subseteq R^{NZ}(P,Q)$.

Having the minimum non-zero rectangle $R^{MNZ}(P,Q)$ $\left(x_{min}^{MNZ}, y_{min}^{MNZ}, x_{max}^{MNZ}, y_{max}^{MNZ}\right)$ is obtained, we can approximate the Kullback-Leibler divergence with the following sum:

$$
\begin{aligned}
&D_{KL}(P \parallel Q) \approx \left(\frac{d_x \cdot d_y}{k^2}\right) \cdot \\
&\sum_{i=0}^{k}\sum_{j=0}^{k} P\left(x_{min}^{MNZ} + (i+0.5)\frac{d_x}{k}, y_{min}^{MNZ} + (j+0.5)\frac{d_y}{k}\right) \cdot \\
&\log\left(\frac{P\left(x_{min}^{MNZ} + (i+0.5)\frac{d_x}{k}, y_{min}^{MNZ} + (j+0.5)\frac{d_y}{k}\right)}{Q\left(x_{min}^{MNZ} + (i+0.5)\frac{d_x}{k}, y_{min}^{MNZ} + (j+0.5)\frac{d_y}{k}\right)}\right)
\end{aligned}
\tag{8.2}
$$

where $d_x = x_{max}^{MNZ} - x_{min}^{MNZ}$ and $d_y = y_{max}^{MNZ} - y_{min}^{MNZ}$. The k value is the *resolution* of the numerical integration. The sufficient value of k need to be determined experimentally. Our experiments show that for $k = 100$ the obtained value remains stable up to the third decimal, more than sufficient for our purposes.

8.5 The "informed" observation distribution

The probability distribution implied by the actual observation is an "uninformed" distribution, which does not take into account the *a priori* information we have about the building and target behavior. The particles, however, encode this information as part of the particle update step. In order for our divergence measurement to be meaningful, we need to apply this information to the observation distribution. As the a priori probability and the probability implied by the observation are independent, the informed probability distribution can be obtained from a simplified application of the Bayes's rule. For the case of a probability distribution represented as a ziggurat, we find that the posterior of two ziggurats is also a ziggurat.

Let us now consider an example. Figure 8.2-a shows the a priori distribution of the probability of the targets for the interior of the CoRE building at Rutgers University. The non-zero areas represent the corridors of the building which are accessible to the

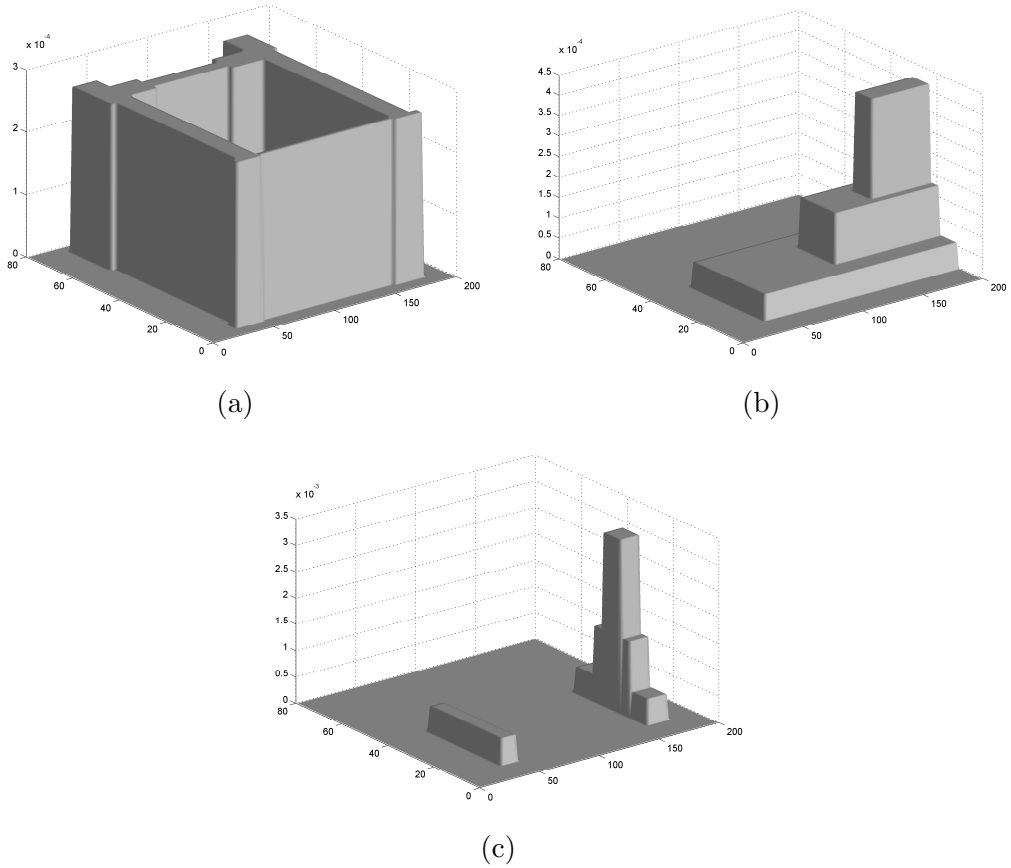


Figure 8.2: An example of the computation of the posterior probability of two independent distributions represented as ziggurats. (a) The a priori distribution of the probabilities. (b) The probability distribution corresponding to the intersection area of a reading. (c) The probability distribution obtained as the result of combining the a priori probability with the reading. Notice that the result is still a ziggurat, but it is a multi-modal distribution.

target. Figure 8.2-b shows the probability distribution implied by the observations. The highest step of the ziggurat denotes the intersection area, while the lower steps account for the lower probability of the areas around the intersection area. The ziggurat is truncated on the right side because parts of it fall outside the building. Finally, Figure 8.2-c shows the posterior distribution of these two ziggurats.

8.6 The distribution implied by the particles

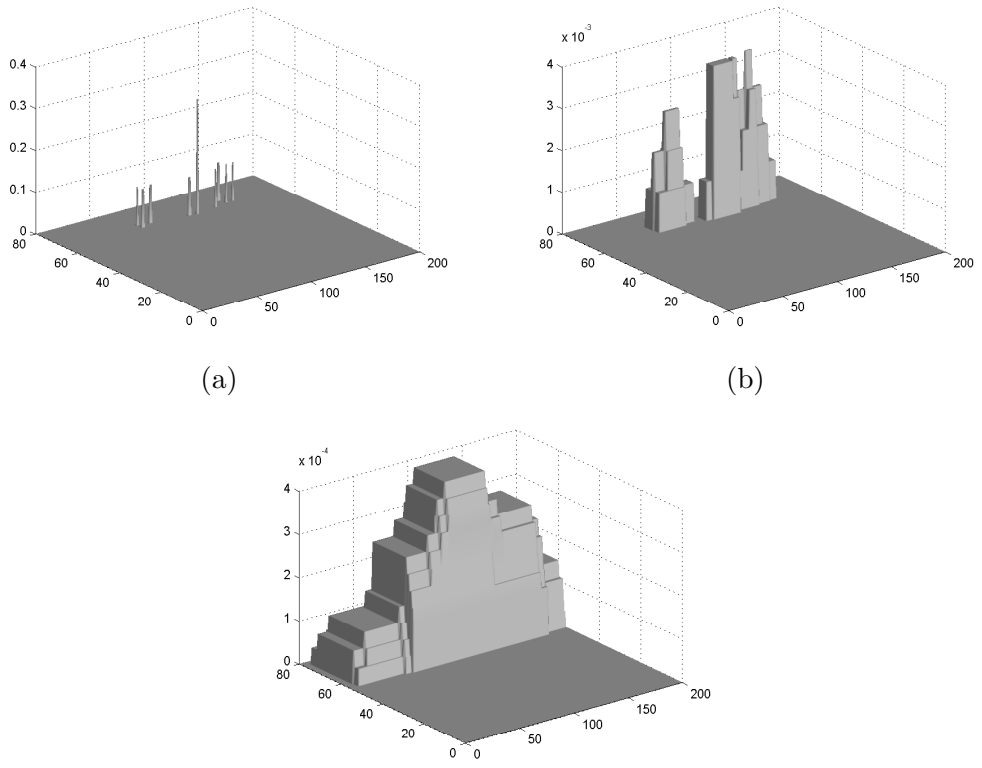
Although it is usually accepted that the particles "imply" a distribution, this is not strictly speaking true. The way in which the particle weights are defined are only about the *expectation* of the function f:

$$\int f(X_k) \cdot p(X_k|o_0, \ldots o_k) dX_k \approx \sum_{L=1}^{p} w^{(L)} f(X_k^{(L)}) \tag{8.3}$$

where the f function can be the moments of the distribution to some degree of approximation - in our case the expected value. Also, in our case, $X = (x, y)$. There are many different probability distributions which verify this result. Normally, the particle filter workflow operates without explicitly instantiating this distribution. For the purposes of the KLD calculation however, we need a concrete instantiation which matches our intuition and domain knowledge for what the shape of the distribution would be and it is also computationally convenient. The latter requirement means that we prefer the distribution to be expressed as a ziggurat.

The approach we propose is an adaptation of the Kernel Density Estimation (also known as Parzen windows) method [29], frequently used in statistics to estimate a distribution from a series of samples. We choose the kernel as a ziggurat with base b. Then we sum these probability distributions, each scaled with the weight of the corresponding particle. The resulting distribution is then normalized such that it integrates to 1. The distribution obtained depends on the size b of the kernel. Figure 8.3 shows the shape of the distribution from a typical population of 50 particles with $b = 1$, $b = 10$ and $b = 50$. The expected location is (120, 65) for all locations, but the probability distributions differ widely. For case (a), the probability distribution is a collection of disjoint probability peaks, the locations between these peaks are considered of zero probability. For (b) the probability locations are starting to coalesce, forming two disjoint groups separated by a zero-probability "valley". One of the groups also has two distinct "peaks". Finally, in (c) we have a single group, which however, has a very coarse outline.

The distributions with the individual peaks do not match well with our domain knowledge. There is nothing in the application domain which would force the target



(a) (b)

(c)

Figure 8.3: The probability distribution generated from a collection of 50 particles using the ziggurat summation algorithm with the size of the base rectangle being (a) 1, (b) 10, and (c) 50.

to only occupy certain fixed locations in space. On the other extreme, the distribution shown in Figure 8.3-c feels too coarse and it looses some of the information present in the particles. For instance, it shows an equal probability of the target on the y axis between 60 and 80. In reality, the target is restricted to a narrow corridor correctly represented in the particles and in the distributions in Figure 8.3-b.

In conclusion, the kernel size needs to be calibrated to conform to our domain knowledge. For the purpose of the experiments described in this paper we used a kernel size of 10.

8.7 Experimental study

8.7.1 Experimental scenario

In the following we present the results of an experimental study which investigates the effect of the proposed automatic restart on the accuracy of the dynamic localization workflow. The restart step erases the history of the particle filter, thus, when unnecessarily applied to a non-stranded, correctly operating particle filter, it will actually decrease the accuracy of the localization. The goal is to *calibrate* the restart step by the appropriate setting of the restart threshold T_{KLD} such that it restarts only when the particle filter is indeed stranded and cannot recover on its own.

For this study, we have considered a scenario involving a person moving at a normal walking pace around the corridors of the CoRE building at Rutgers University. During this walk approximately 24 observations are received from the minimum intersection area algorithm operating on data received from the GRAIL system. The observations are fed into the particle filter workflow, which then provides the results at a much higher temporal resolution.

This scenario is a relatively good match for the particle filter model: the movement is relatively predictable, and the narrowness of the corridors used as a priori information constraints the possible errors. There are however three 90° turns in the trajectory where the inertial prediction of the movement would fail. These together with the starting point are the cases where the particles might make a bad prediction stranding the particle filter. Whether this will indeed happen depends on the actual series of observations, their errors and precision. The particle filter can usually handle relatively large errors in the longer straight sequences, but one or more large errors in the turns might make the particle filter "miss the turn".

The original data was collected from a real time experiment using the GRAIL system with the ground truth collected from markings on the floor. In order to have a larger dataset, we also generated a series of artificial scenarios, by generating the error of the instantaneous localization from a distribution which matches the cumulative distribution function of the experimental data.

8.7.2 Experimental results

We have run the localization workflow over the 100 scenarios for the same movement pattern of the target, but different distributions of the instantaneous localization error (but with the same statistical properties). We used $n = 140$ particles. The deterministic component was a smoothed inertial predictor based on the running average of estimated speed, with the weight of the new observation being $w = 0.2$. The stochastic components was random wandering, with angle noise $\epsilon_{angle} = 0.05$ and speed noise $\epsilon_{speed} = 0.05$.

For each scenario we have repeated the localization workflow using several values of the KLD threshold (T_{KLD}). A value of $T_{KLD} = 0$ means that the particle filter restarts at every observation, while $T_{KLD} = \infty$ means it never restarts. For each experiment, we measured the distance between the current ground truth (the actual location of the target), and the target location estimated by the particle filter using the weighted mean of the particles.

Figure 8.4 shows the average localization error over the 100 scenarios for T_{KLD} values of 0, 10, 15 18, 22, and ∞. We find that, as expected, both too frequent or too rare restarts lower the performance. Without the restart mechanism the average localization error is 8.49 ft. The best performance has been obtained for $T_{KLD} = 18$ with the average error being 5.38 ft (a 36.6% improvement), followed by $T_{KLD} = 22$ with 5.96 ft.

However, the average values do not tell the complete story. Let us now investigate in detail what is happening during the dynamic localization. Figures 8.5, 8.6, 8.7, 8.8 show the time series of the localization error for four representative scenarios. To reduce the clutter on these graphs, we only retained the T_{KLD} values of 0, 18, 22, and ∞.

In Figure 8.5 (scenario 3), we find that the "always restart" case creates a very jagged estimate, essentially mirroring the observation errors. Of course, the frequent restart guarantees that the system will quickly recover from the errors, but the overall error will still be high.

In contrast, the "never restart" approach, in this particular case becomes stranded

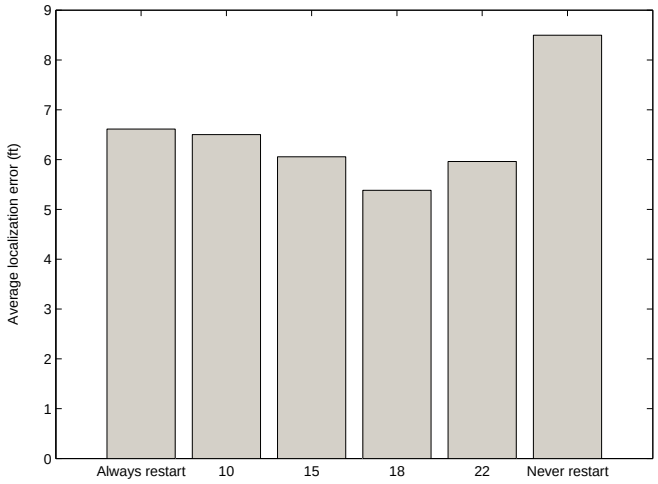


Figure 8.4: The average localization error over 100 scenarios for various levels of the restart threshold.

with a large error throughout most of the scenario. Figure 8.9 shows the position of the particles at time 2230, clearly showing that the stranded particles are not overlapping with the current observation. At some moment, through the random spreading of the particles, some of the particles have accidently strayed in the correct zone, and the particle filter recovered by resampling around those particles. After this recovery, the "never restart" approach had actually provided very good performance.

The remaining two cases, for $T_{KLD} = 18$ and $T_{KLD} = 22$ provide the best overall performance. We notice that these two cases start to diverge from the "never restart" around timepoint 500, and from each other at around timepoint 3000 (when $T_{KLD} = 18$ restarted but $T_{KLD} = 22$ did not). In the remainder of the scenario, these two settings alternate in providing the best approximation, without a clear winner emerging.

Figure 8.6 (scenario 17) shows similar results. The "never restart" approach gets stranded at the beginning, recovers around 7000, gets stranded again at 11000, recovers around 13000. The "always restart" scenario shows occasional large errors but quick recoveries. The interesting difference is between $T_{KLD} = 18$ and $T_{KLD} = 22$. For the previous scenario, these were evolving very similarly. Here $T_{KLD} = 22$ follows very closely the "never restart" scenario, getting stranded at the same points. $T_{KLD} = 18$,

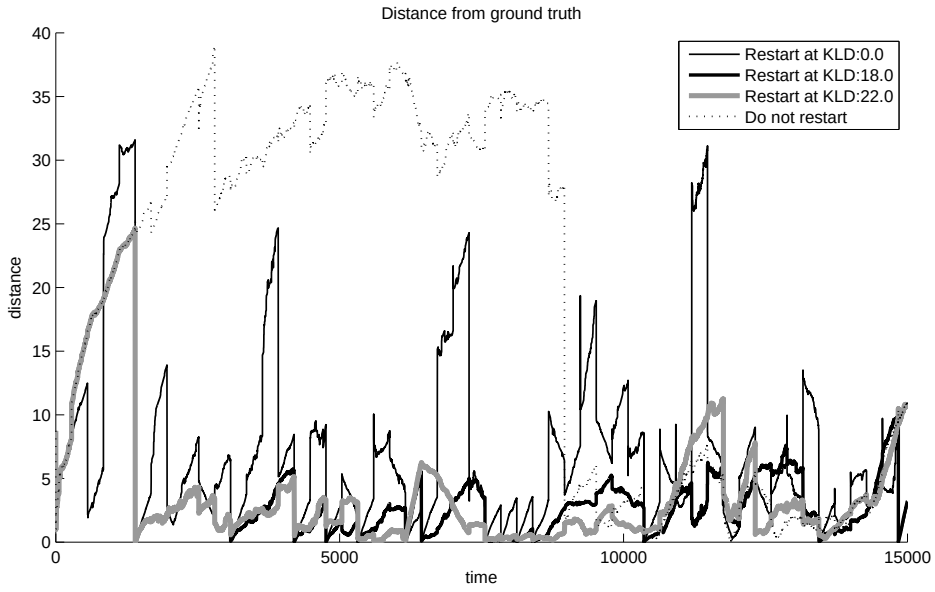


Figure 8.5: The accuracy of four representative runs for various levels of the restart threshold; given scenario 3.

although starts on the same trajectory, recovers much sooner through a restart and it yields the overall best performance.

Figure 8.7 (scenario 22) offers a similar scenario to Figure 8.5 (scenario 3), with both the $T_{KLD} = 18$ and $T_{KLD} = 22$ offering good performances (the latter does not become stranded in this scenario).

Finally, Figure 8.8 (scenario 59) shows a scenario where the distribution of the instantaneous localization errors happened in such a way that the particle filter did not get stranded. In this case the "never restart" approach gives the best results, being a canonical application of the particle filter model.

We conclude that the T_{KLD} threshold needs to be set in such a way as to balance the early detection of a stranded particle filter, with the performance cost of too frequent restarts. T_{KLD} values of around 18-22 were found to be the best performers - they restart relatively infrequently, but will restart in the case of a large discrepancy between particles and observation.

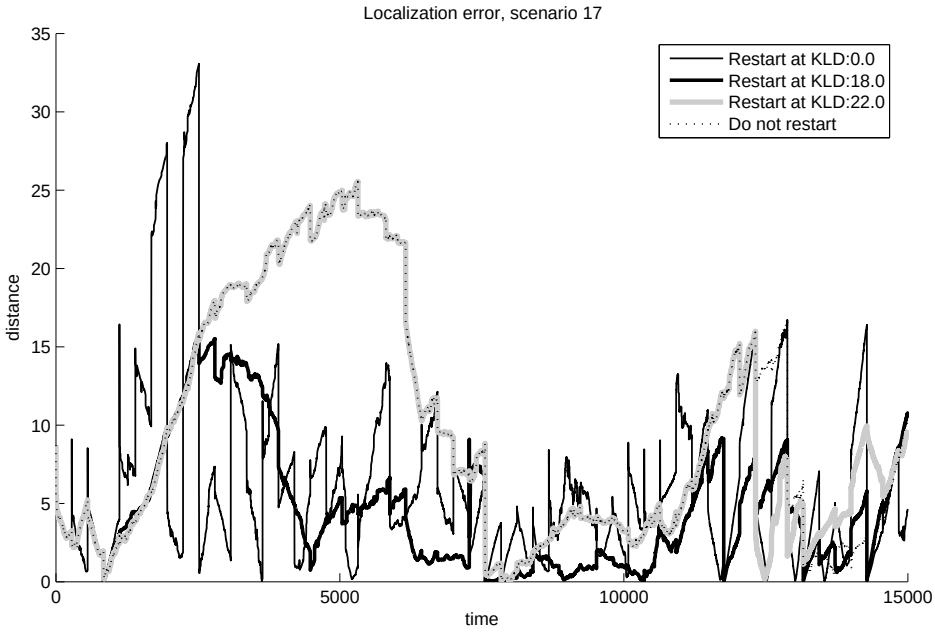


Figure 8.6: The accuracy of four representative runs for various levels of the restart threshold; given scenario 17.

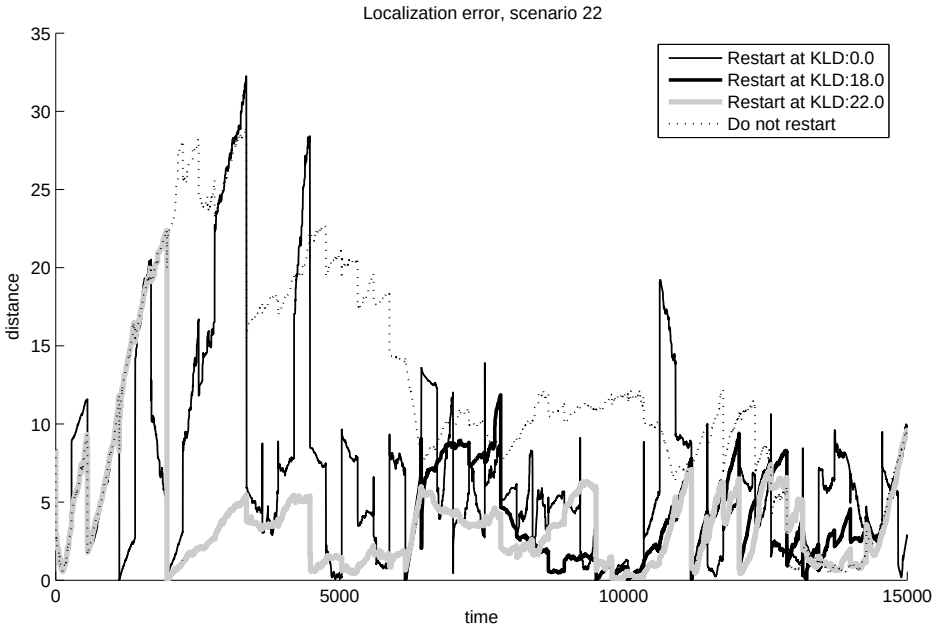


Figure 8.7: The accuracy of four representative runs for various levels of the restart threshold; given scenario 22.

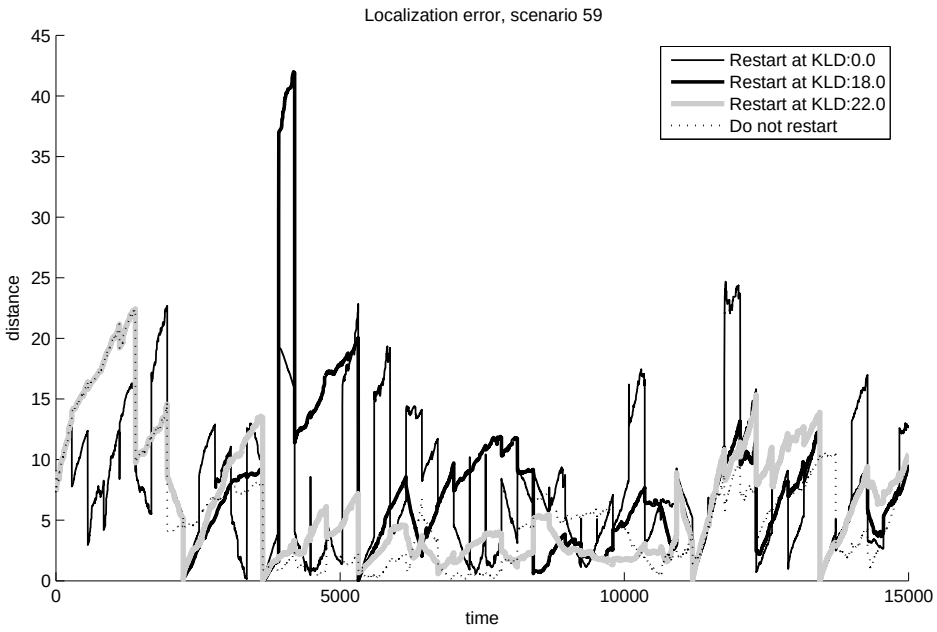


Figure 8.8: The accuracy of four representative runs for various levels of the restart threshold; given scenario 59.

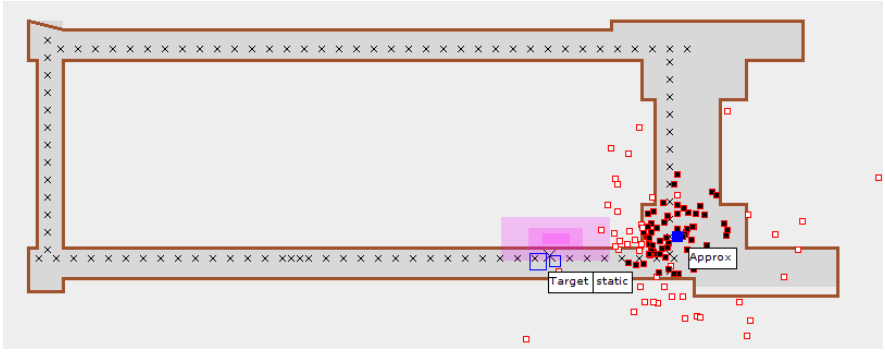


Figure 8.9: A screenshot of the evolution of the particles in scenario 3 at cycle 2230. The shaded area represents the ziggurat corresponding to the current observation. We can clearly identify that the particles are "stranded".

Chapter 9

A multi-hypothesis particle filter for dynamic localization

Dynamic localization in indoor environments is complicated by the fact that the environment periodically presents decisions to the target. For instance at a T-shaped corridor intersection, the target might turn to the left or to the right; intermediate values are not possible. The mechanisms used for target tracking (such as particle filters) do not fit well with such divergent choices. Accurately determining the choice made by the target can be done only later, when sufficient observation data is accumulated. Making a guess is risky, and it can lead to large errors.

One possibility is to make two separate hypotheses covering the two possible choices, and track them separately using the classical methods. Often, after some future observations are recorded, one of the hypotheses can be discarded as inconsistent with the observations. In other cases, for instance, when the choice is between avoiding an obstacle on the right or left side, the two hypotheses will be merged when their associated trajectories converge.

In this chapter, we describe the design and implementation of such a multi-hypothesis approach. The approach is based on "hypothesis modifiers", which encode the choices facing the targets in specific areas of the building. The hypothesis modifier might split the set of particles into multiple clouds. Each cloud is matched with a specific hypothesis which determines the prediction model of the particles (but *not* their weight update model). The different clouds will evolve separately under their specific prediction model, each cloud providing a separate, distinct estimate to the *tracking*.

The particles in the different clouds are facing weight updates according to the same observations and the same rules, thus the sum of the weights of the individual clouds can change in time.

Furthermore, they are also resampled on an overall basis (rather than cloud-by-cloud). Thus, the number of particles allocated to the individual clouds can also change. When a cloud will remain with 0 particles, it is considered that the associated hypothesis has been *refuted by the observations*, and the cloud is discarded.

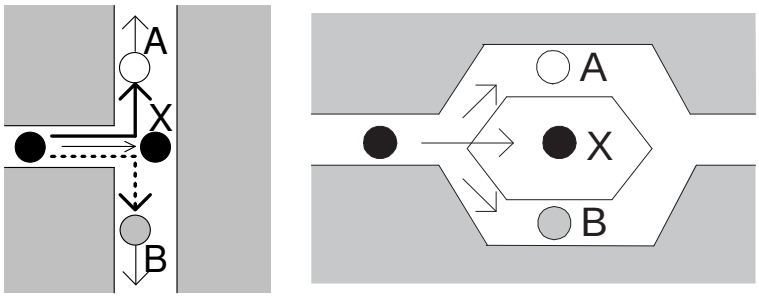
Alternatively, the hypothesis associated with two clouds might converge to the same value. In this case, we can merge the two hypotheses into a single one.

9.1 Divergent choices in indoor environments

The movement of targets in indoor environments is constrained by walls, obstacles and social conventions. The structure of the indoor environments occasionally forces the targets to make discrete, either-or choices with far-reaching results. Let us consider a human target walking towards a T-shaped intersection. The human, arriving to the intersection is forced to turn left or right (Figure 9.1-a). Without additional information, we can make the assumption that the probability of both choices is $p_A = p_B = 0.5$. In an uninformed particle filter the particles will spread evenly in the two directions. The estimate of the location would be in the weighted average of the particles' locations, that is, in the center at location X. This result can only be interpreted as the target stopping at the T intersection, which is almost surely incorrect.

In Figure 9.1-b the divergent choice is between avoiding an obstacle to the left or to the right. In this case, the weighted average estimate X is not only unlikely, but also *unfeasible*, falling in the middle of the obstacle.

The problem in these examples stems from the loose definition of the word "estimate". As a point which minimizes the statistical errors, the X estimates in these examples are correct. Many practical applications, however, require a *feasible estimate* point or even a small set of estimates. If we want to find John, saying "I estimate John to be either in room A or room B", is more useful than saying "I estimate John to be in the wall between room A and B". The incorrect estimates in these examples resulted from a continuous integration over discrete choices. Our solution is to create a new, explicit hypothesis for each possible choice, and estimate based on continuous



(a) (b)

Figure 9.1: Discrete choices in an indoor environment: (a) Turning left or right at a T-shape corridor (b) Avoiding an obstacle to the left or to the right.

integration only inside the individual hypothesis.

9.2 Particle hypothesis

A particle hypothesis encompasses the prediction model of the particle, represented by the deterministic prediction function $f(p_t) \rightarrow p_{t+1}$ and a stochastic component. In our system, the stochastic component model is the random dispersion model in [40]. We assume that the stochastic model is shared across all the hypotheses, which differ only in the deterministic component.

It is assumed that all the environmental information is already encoded in the function, thus the only parameter of deterministic prediction function is the current particle p_t. The prediction function can take different forms depending of the level of detail and sophistication we use in modeling the target. To illustrate the design space let us consider some representative examples:

Stay in place: $f_I(p_t) = p_t$ assumes that the particle does not move.

Per/particle inertial movement: assumes that the particle moves with a constant speed. Naturally, this hypothesis assumes that the speed of the particle $\bar{v} = (v_x, v_y)$ is part of the particle representation $p_t = (x, y, \bar{v})$. Under this representation

$$f(x, y, \bar{v}) = ((x + \Delta t \cdot y_x, y + \Delta t \cdot v_y, (v_x, v_y), (v_x, v_y)) \tag{9.1}$$

Per/cloud inertial movement: assumes that the *estimate* of the cloud has an inertial movement, with a speed $\bar{v}^{cloud} = (v_x^{cloud}, v_y^{cloud})$.

$$f(x, y) = (x + v_x^{cloud}, y + v_y^{cloud}) \tag{9.2}$$

with $\bar{v}^{cloud}$ updated after every step as follows:

$$\bar{v}^{cloud} = \frac{\bar{E}_{t+\Delta t}^{cloud} - \bar{E}_t^{cloud}}{\Delta t} \tag{9.3}$$

9.3 Hypothesis modifiers

The nature of the indoor environment might occasionally make some hypotheses untenable. For instance, when a particle cloud is approaching a bend in a corridor, or a T-shape intersection, the hypothesis of inertial movement would predict that the target would crash through the wall.

A single-hypothesis particle filter would make the prediction anyhow, end up with a large error, and then gradually recover through the mechanism of weight updates and resampling.

With sufficient environment knowledge we can anticipate the changes necessary to maintain the hypothesis valid after a certain choice is made. For instance, in a L-bend, we can make the assumption that the target turns left and continues in the new direction with its previous speed.

We define a *hypothesis modifier* as a triplet $< M, G, C >$ where:

- $M(H) \rightarrow \{\texttt{true}, \texttt{false}\}$ is the *match function*, which decides whether the old hypothesis is eligible for modification. Frequently, the match function is location-based, that is, if the old hypothesis provides an estimate in a certain geographical area, it is eligible for modification. Another example of match function is one which takes into account both the location and the speed of the old hypothesis's estimate.

- $G(H) \rightarrow H_{new}$ is the *hypothesis generator function*, which generates a new hypothesis based on the old one.

- C is the *cloud splitting function* which decides what fraction of the particles of the original hypothesis will become part of the new hypothesis. If all the particles will become part of the new hypothesis, the old particle cloud will be replaced by a new cloud (with the same particles but a different hypothesis). Alternatively only a fraction of the particles become part of a new hypothesis, carving out a smaller particle cloud from the existing cloud. A hypothesis modifier can also distribute the particles of the old cloud among a number of newly created hypotheses.

Figure 9.2 shows several examples of hypothesis modifiers.

9.4 The life cycle of particle hypotheses

Particle hypotheses are created at the beginning of the localization process (usually by having a single starting hypothesis) and subsequently by the hypothesis modifiers triggered by the particles.

The initial number of the particles in a cloud associated by the hypothesis is given by the modifier. The size of the cloud can increase or decrease through the weight update and the resampling steps of the particle filter. In contrast to the prediction step (determined by the hypothesis), the weight update and resampling steps are performed over the union of all particles in all the clouds. A cloud which has particles with large weights, might emerge with a larger number of particles from the resampling step. A cloud with particles with low weights might emerge with a lower number or even zero particles from resampling.

When the cloud associated with a particle hypothesis has either no particles, or the sum of the weights of the particles is zero, we say that the particle hypothesis is *refuted*. As there is no mechanism through which such a hypothesis can be restored, these hypotheses can be removed from the particle filter.

For instance, in Figure 9.2-c, the particle cloud was split into a cloud corresponding to the hypothesis “Turning” and one corresponding to the hypothesis “Moving

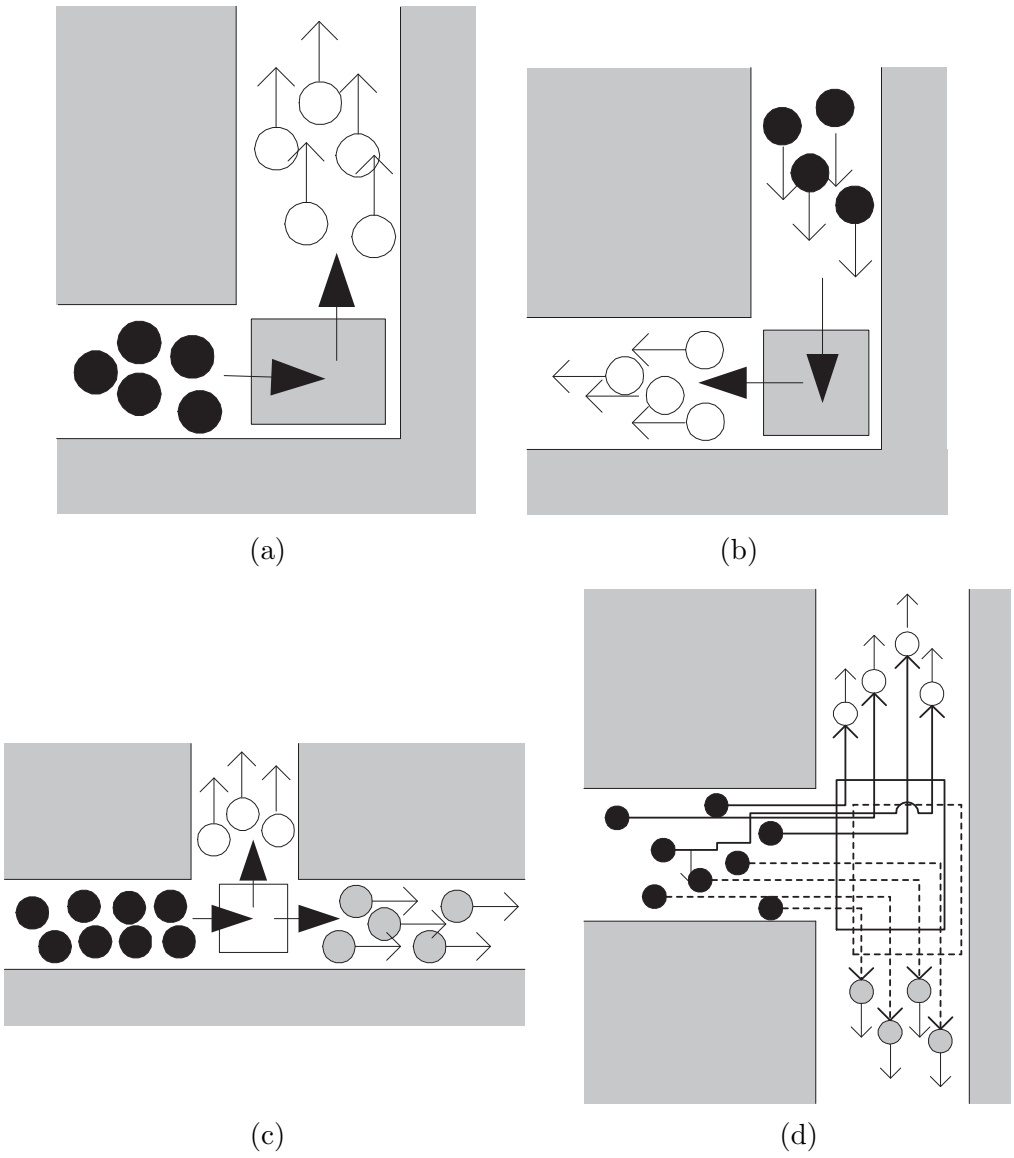


Figure 9.2: Examples of hypothesis modifiers: (a) turning right in an L-bend (b) turning left in an L-bend (c) choice between staying on course or turning and (d) choice between a left turn and a right turn.

Table 9.1: Simulation parameters

Particle filter	
Number of particles	100
Prediction model	
random component	random diffusion
deterministic component	inertial prediction
Instantaneous localization	
A priori information	Building map with inaccessible areas
Real time information	Intersection area of iso-RSS lines
Observation frequency	3 sec

straight". If the incoming observations are more in line with the assumption of "Moving straight", this cloud will gradually become larger, until the "Turning" cloud disappears completely.

A different situation occurs when clouds merge. This is typical for cases when obstacles are avoided - after the obstacle is avoided to the left or right, the clouds merge, both in terms of location and in terms of the predictions. The merging of the clouds happens automatically, but the removal of the associated particle hypothesis structure does not. Such merged clouds can live for a long time, because the weight updates and resamplings will affect them equally. While this does not have a negative effect on the accuracy of the localization, it can have an impact of the performance, especially if these clouds are further split by particle modifiers encountered down the line. The solution is to periodically check the existing hypotheses and explicitly merge those whose pair-to-pair distance is very small.

9.5 Experimental results

In the following we describe a series of experiments in which we compare the results given by the multi-hypothesis particle filter with a single hypothesis particle filter. To keep the comparison fair, the approaches share the same prediction model, environmental model, resample and restart parameters.

The simulation parameters are summarized in Table 9.1.

During the simulation, we will extract the various estimates provided by the particle

filters. There are several of these estimates, based on the type of the particle filter.

For the single hypothesis case we extract the E_H *single hypothesis estimate* based on the weighted sum of the particles.

For the multi-hypothesis case we extract two possible estimates, both of which can serve as the estimate in real-world deployed systems. The *average estimate* E_{avg} is obtained by finding the average of the estimates of the currently active clouds. The *strongest cloud estimate* E_{str} is obtained by retaining the estimate based on the cloud with the maximum sum of the particle weights.

Finally, we will retain the estimates based on the *best* and *worst hypothesis* where the best hypothesis is the one whose estimate E_{best} is the closest to the ground truth, while the worst hypothesis is the one whose estimate E_{worst} is the farthest. Naturally, in a real-world system, these values can only be calculated during the calibration phase when the ground truth is available. Once the system is calibrated, we can retain these values which would help us in the interpretation of the data. The knowledge of average values of these errors, however, can help us to interpret the real time data.

9.5.1 Scenario 1: L-bend

The first scenario involves an *L-bend* in a corridor. The typical problem for a particle filter in this scenario is that the inertial prediction breaks down in the bend, as the targets normally turn to follow the bend in the corridor.

To correct this, we use the following two hypothesis modifiers (see Figure 9.3):

Modifier 1: matches targets coming from the left in the horizontal corridor. Creates a new hypothesis which assumes that the particles turn 90° to the left but maintain their existing velocity. All the particles of the original hypothesis are transferred to the modified hypothesis.

Modifier 2: matches targets coming from the top in the vertical corridor. Creates a new hypothesis which assumes that the particles turn 90° to the right but maintain their existing velocity. All the particles of the original hypothesis are transferred to the modified hypothesis.

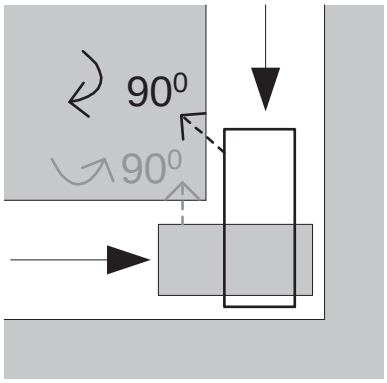


Figure 9.3: Examples of hypothesis modifiers in an L bend corridor.

Note that although this scenario contains hypothesis modifiers, it will never "split a cloud", because the modifiers always replace *all* the particles in the cloud.

For the experiments we have generated 100 scenarios covering targets moving in both directions, with speeds between 2.4-3.6 ft/s. Table 9.2 shows the estimation error for the single and multi-hypothesis cases. As there is no cloud splitting, all the multi-hypothesis estimates are identical. The multi-hypothesis estimates are significantly more accurate than the single hypothesis estimate. As a note, as the estimation error is averaged over the complete scenario, including the relatively long time before the turn, the average estimation error difference appears smaller than it is in the period after the turn.

This is illustrated in Figure 9.4 which describes the evolution of the estimation error in time for a particular scenario of a node traveling from the left. The two estimates are identical until the time of 20 seconds, where the node reaches the bend. At this moment the estimates diverge sharply: the estimates with the modified hypothesis having a significantly lower error of 4-6 ft, while the single hypothesis estimate quickly diverging an error of 20 ft at the end of the simulation. The screen shot of the visualization of the particles in Figure 9.5 shows the particles and the estimate after the turn in the L-bend.

Table 9.2: Estimation error for the single and multi-hypothesis estimates for the L-bend scenario

E_H	E_{avg}	E_{str}	E_{best}	E_{worst}
9.4096	8.2562	8.2562	8.2562	8.2562

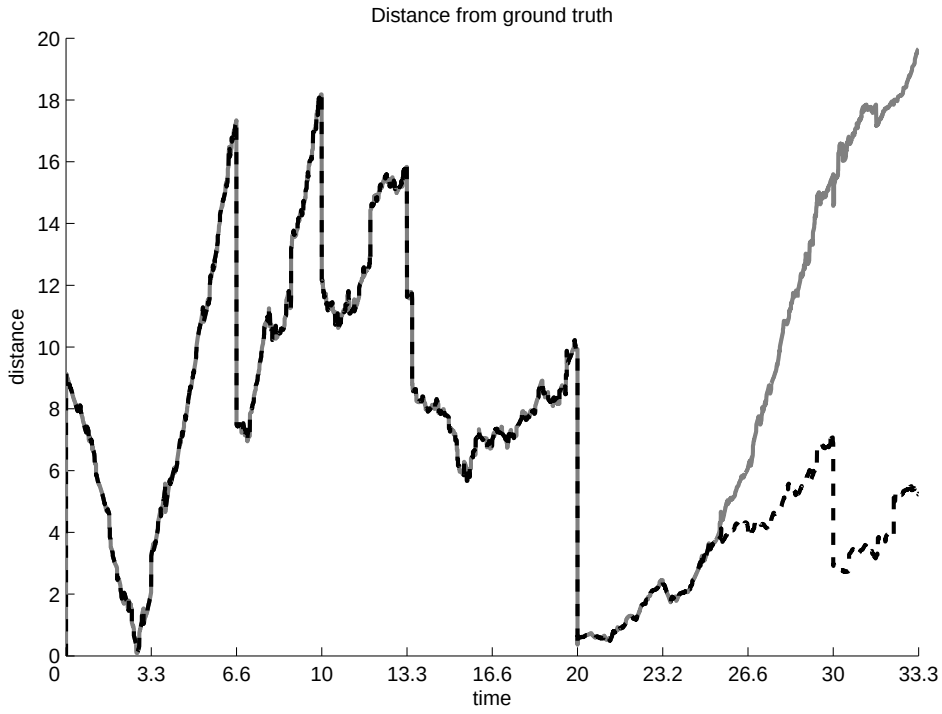


Figure 9.4: Time series of the estimation errors for the L-bend scenario

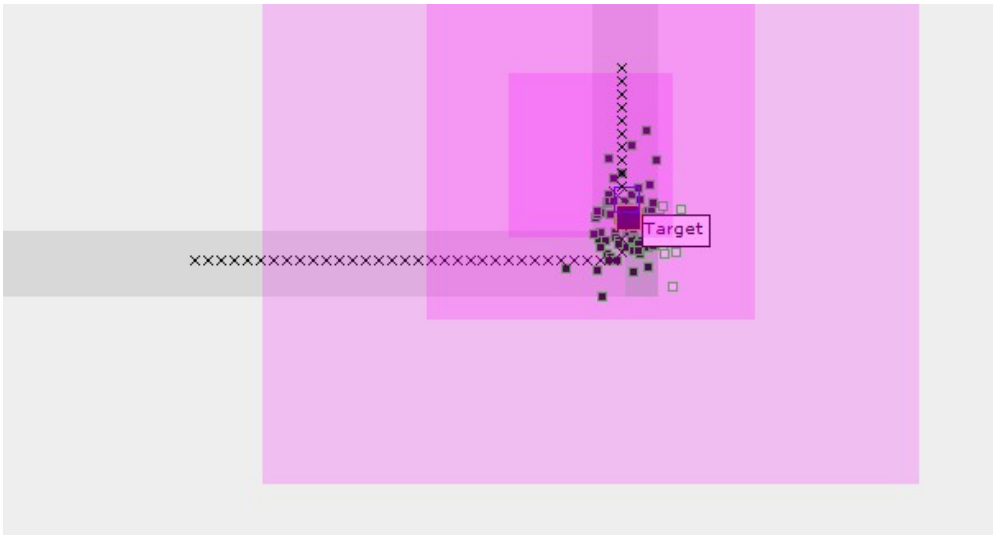


Figure 9.5: Screen shot of the L-bend scenario

9.5.2 Scenario 2: T intersection

The second scenario involves a *T intersection* in an indoor corridor. The target can come from three different possible directions in this scenario (if we consider the orientation of a T letter, these directions would be from left, from right and from bottom). In each case, it faces a binary choice. In our implementation we assume that the probability of each choice is 50%. These choices can be represented through the following 6 modifiers:

Modifier 1+2: match targets coming from the left. They each create a new hypothesis which assumes that 50% of the particles turn 90° to the right with their existing velocity magnitude (Modifier 1) or continue on the corridor (Modifier 2).

Modifier 3+4: match targets coming from the right. They each create a new hypothesis which assumes that 50% of the particles turn 90° to the left with their existing velocity magnitude (Modifier 3) or continue on the corridor (Modifier 4).

Modifier 5+6: match targets coming from the bottom. They each create a new hypothesis which assumes that 50% of the particles turn 90° to the left (Modifier 5) or 90° to the right (Modifier 6) with their existing velocity magnitude.

The experimental setup is similar to the L-bend scenario. The 100 experimental scenarios were evenly distributed among the six possible choices and with the speed of

the target being randomly chosen between 2.4-3.6 ft/s. Table 9.3 shows the estimation error for the single and multi-hypothesis cases. As opposed to the L-bend scenario, the modifiers for T-intersection split the particle cloud, thus the various estimates for the multi-hypothesis case are different. We see that the average and the strongest cloud estimates for the multi-hypothesis case are both significantly better than the single hypothesis estimate. This is due to the fact that the particle filter operates more efficiently when the intermediate value is not present. The best cloud, as expected shows the smallest error, while the worst cloud shows the worst.

As an interesting observation, the estimate of the E_{worst} cloud is only marginally worse than the single-hypothesis estimate E_H. What is happening here, is that the worst cloud is determined dynamically. Immediately after the choice point, the worst cloud is the one which did not guess right the target's choice. However, this cloud will be quickly refuted by the observations. When the cloud disappears, the remaining single cloud will be both the best and the worst, with an estimation better than the corresponding single hypothesis cloud.

Figure 9.6 shows how the estimation error progresses in time for the scenario with the node coming from left. The single and multi-hypothesis estimates remain the same until the time of 20 seconds, where the target reaches the T intersection. In this initial phase, the estimation error gradually decreasing as the particle hypothesis, using the sequence of observations, develops a more and more exact model of the target's movement. After the choice point, however, the estimates diverge radically. The best and the strongest hypothesis (which are the same in this case) continue the low estimation error of the previous couple of seconds. The single hypothesis estimate and the worst cloud estimate show a high and increasing estimation error as their respective predictions (that the target continues to the right side of the T) are incorrect.

The screen shot of the visualization of the particles in Figure 9.7 shows the particles and the estimate after the turn at the T intersection.

Table 9.3: Estimation error for the single and multi-hypothesis estimates for the T-intersection scenario

E_H	E_{avg}	E_{str}	E_{best}	E_{worst}
11.3858	9.4864	9.6271	8.27629	11.5182

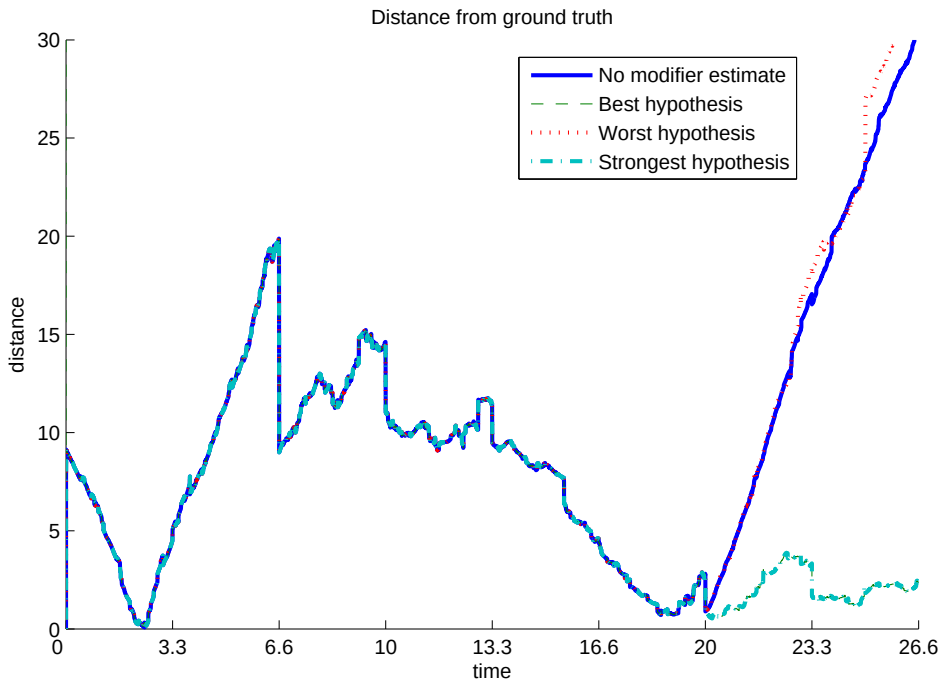


Figure 9.6: Time series of the estimation errors for the T-intersection scenario

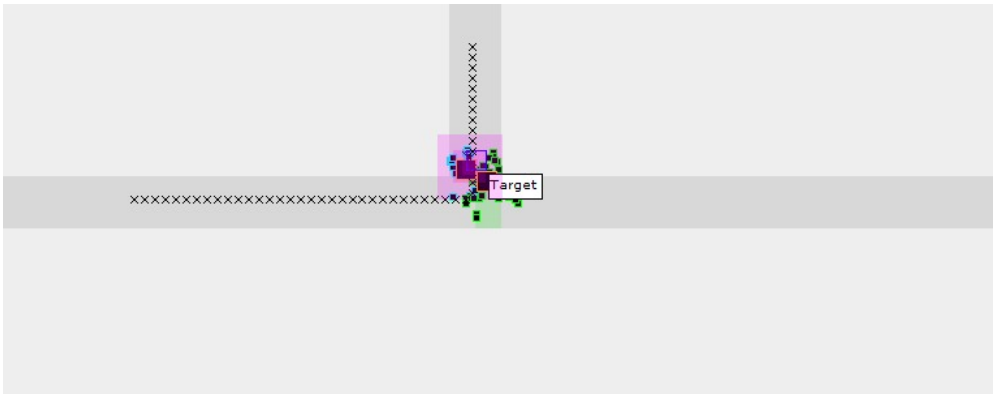


Figure 9.7: Screen shot of the T-intersection scenario

9.5.3 Scenario 3: Obstacle avoidance

The third scenario we are considering involves having an obstacle in the middle of the corridor. The target might choose to avoid the obstacle to the left or to the right. Although the choice diverges for a while, after the obstacle is avoided, the target will return to its original trajectory, unaffected by the choice.

To model this case we will use eight hypothesis modifiers, four each for the two directions of movement on the corridor. At the beginning of the obstacle, two modifiers are splitting the cloud evenly for the choices to avoid to the left or to the right. At the end of the obstacle, two modifiers are guiding all the particles back to the original trajectory on the corridor.

In the experimental study, we used 100 scenarios, 25 each for all the combinations of moving in the two possible directions in the corridor and the choices of avoiding the obstacle. Again, the speed of the target was randomly chosen between 2.4-3.6 ft/s.

Table 9.4 shows the estimation error for the single and multi-hypothesis cases. We note that the accuracy for all the multi-hypothesis estimates is better than the single hypothesis estimate - even for the *worst* hypothesis! The reason for this is that in the single hypothesis case the default hypothesis is the inertial one, which assumes that the target moves straight into the obstacle. Furthermore, even if the single hypothesis eventually finds its way into one of the avoidance choices, it will face another challenge at the end of the obstacle in moving back to the normal path.

Figure 9.8 illustrates the time series of an interesting scenario where the multi-hypothesis approach fails to make the correct hypothesis the strongest one and still manages to obtain a better performance that the single hypothesis approach. Let us see what is happening step by step. All the estimates are overlapped until about the 30 second point, where the hypotheses are split in the multi-hypothesis case. Due to a combination of observation errors, however, the strongest hypothesis remains the one which goes around the obstacle in the wrong way. The “best” hypothesis, of course, have chosen the correct path - but in a real world scenario, we would not know which one is the best!

Table 9.4: Estimation error for the single and multi-hypothesis estimates for the obstacle avoidance alternatives scenario

E_H	E_{avg}	E_{str}	E_{best}	E_{worst}
20.0961	18.2461	18.5660	17.6995	19.4947

At time 50 seconds, however, something peculiar happens: a new observation comes in which effectively refutes the incorrect hypothesis. The multi-hypothesis approach suddenly decreases to a very small error - and continues like that to the end of the scenario. For the single hypothesis approach however, although a temporary reduction of error happens at the observation at time 50 seconds, the error will continue to increase until the end of the scenario.

The reason why the multi-hypothesis approach achieved a better accuracy in this scenario is not because it "guessed right" – in fact it guessed wrong: its strongest hypothesis was the incorrect one. But the fact that it continued to track the weaker hypothesis as well allowed the multi-hypothesis particle filter to quickly switch to the backup hypothesis when the preferred one was refuted by the observations. In contrast, the single hypothesis particle filter did not have particles in the correct region, thus it could not recover. (As a note, the single hypothesis particle filter would eventually recover through the restart mechanism, but this did not happen until the end of this scenario). The screenshot of the visualization of the particles in Figure 9.9 shows the particles and the estimate after the turn at the intersection where the obstacle is placed.

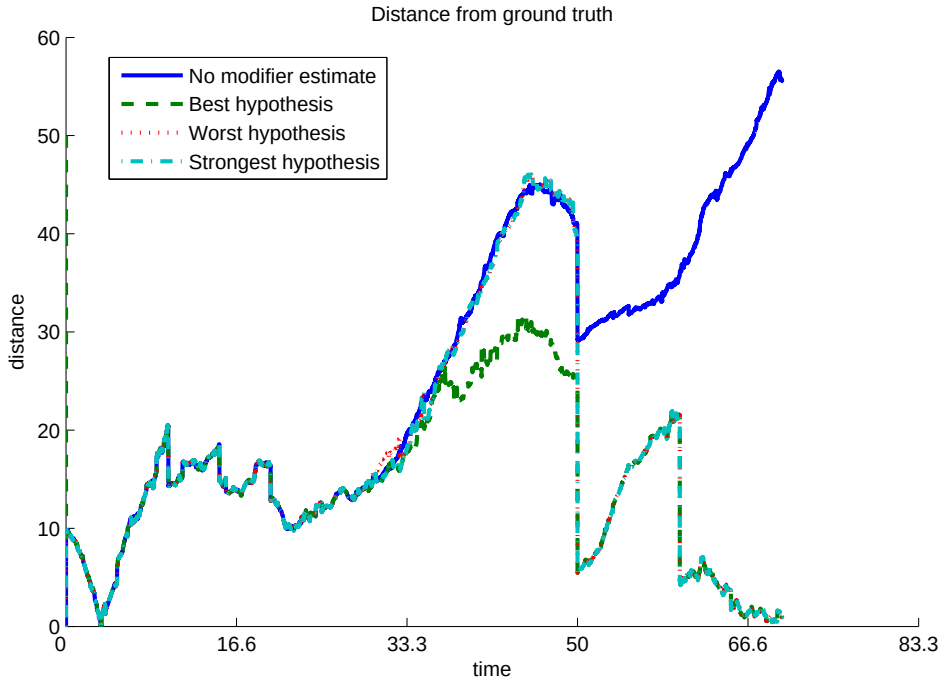


Figure 9.8: Time series of the estimation error for the obstacle avoidance alternatives scenario

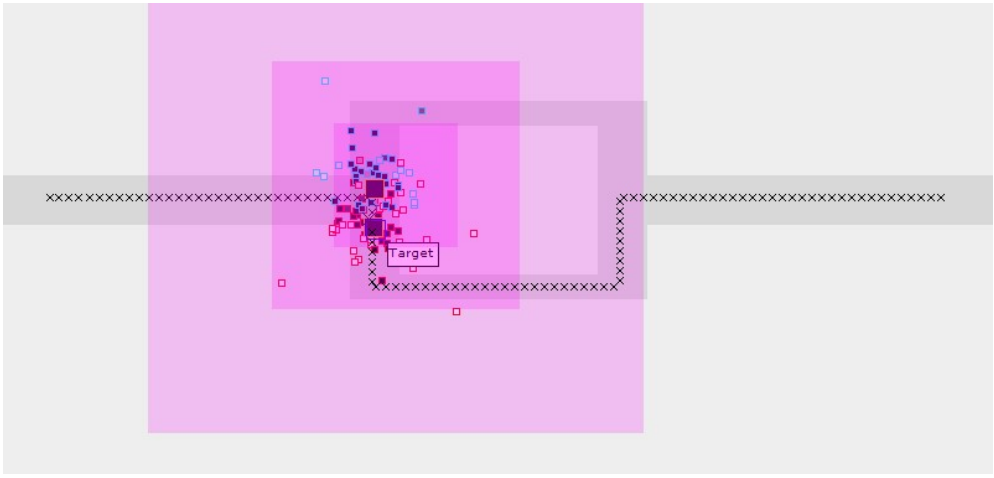


Figure 9.9: Screen shot of the obstacle avoidance alternatives scenario

Chapter 10

Conclusions and future work

In this thesis we have presented a series of contributions to instantaneous and dynamic indoor localization using wireless signal strengths. For the instantaneous localization, we proposed a method based on the minimum intersection area of the iso-RSS lines. This approach allows us to estimate both the location and the localization error. This output is then used by a dynamic localization technique based on particle filters. We have proposed three improvements to the particle filter localization approach: (a) a way to exploit a priori knowledge about the environment and the targets, (b) an approach to automatically restart the particle filter in situations when particles are stranded, that is the observations and the particle cloud are too far from each other for the particle filter to operate effectively and (c) an approach for the particle filter to track multiple hypotheses at the same time, which is important in cases where the targets need to make discrete decisions, which lead to a rapid divergence in the locations.

Experimental studies have found that each of the proposed augmentations of the original particle filter model provide clear improvements in the localization accuracy. Furthermore, these techniques are largely orthogonal to each other, and they can be deployed together in the same system. The best experimental results were obtained in cases where all the components have been deployed.

All the three components can benefit of future work. In the following we outline some ideas which might be used to improve the proposed techniques.

Exploiting a priori knowledge. Our method outlines the *use* of the a priori knowledge. However, the *acquisition* of the knowledge is still a challenge. In our system we have used the type of structural building information which can be acquired from the blueprint of the interior floor. Future systems might exploit other types of information,

for instance open and closed doors, area which are accessible only to some class of personnel, the typical movement patterns of the targets, both at a collective as well as individual level and so on. In general it is desirable that a system learns this information "online", both to reduce the initial deployment cost, and to allow it to adapt to changing patterns of movement, access rights and so on.

Restart. Our approach proposed that the particle filter restarts when the Kullback-Leibler divergence between the probability distribution implied by the observation and the particle cloud exceeds a certain fixed threshold. The correct choice of this threshold has a significant impact on the benefit of the restart - both restarting too often or delaying the restart too much can reduce the accuracy. Our current approach, to calibrate this threshold for a certain environment is functional, but requires a significant number of samples. A possible future work direction would be how to infer this threshold from first principles, without experimental tuning. Another possible avenue would be for a particle filter to self-tune, by learning from its own mistakes. Such a particle filter would be able to recognize later in the process that it should have restarted earlier – or, possibly, that the restart was not necessary. Such a system could adapt its own threshold dynamically and store it for later use in similar circumstances.

Multi-hypothesis approach. Our proposed approach relies on *hypothesis modifiers*, which modify the prediction model of a subset or all the particles. Currently, these modifiers need to be hand-crafted, a laborious process of knowledge engineering. Future work needs to include methods to make the generation of the hypothesis modifiers automatic or semi-automatic. Approaches might include identifying locations where the target trajectories diverge from the predictions, learning the modifications necessary for a correct prediction, learning the fraction of the particles which needs to be affected by the hypothesis modifier and so on.

The overall conclusion of our work is that localization in indoor environments is a complex problem which can not ignore the influence of the complex and dynamic environment, nor the targets which are usually humans or robots with complex behavior which defy simple prediction approaches. We might *wish* to reduce our localization

problem to an idealized mathematical model - but the chaotic reality of indoor environments does not make this possible. The best localization systems of the future need to accept the realities of the environment and integrate all sources of knowledge, including real-time measurements, knowledge of the physical environment, signal propagation and behavior patterns of the human and robotic targets. We hope that the contributions described in this thesis represent steps in this direction.

References

[1] M. Arulampalam, S. Maskell, N. Gordon, and T. Clapp. A tutorial on particle filters for online nonlinear/non-gaussian bayesian tracking. *IEEE Transactions on Signal Processing*, 50:174–188, 2002.

[2] A. Baggio and K. Langendoen. Monte Carlo localization for mobile wireless sensor networks. *Ad Hoc Networks*, 6(5):718–733, 2008.

[3] P. Bahl and V. Padmanabhan. RADAR: An in-building RF-based user location and tracking system. In *Proceedings of the 19th IEEE International Conference on Computer Communications (InfoCom)*, volume 2, pages 775–784, March 2000.

[4] N. Banerjee, S. Agarwal, P. Bahl, R. Chandra, A. Wolman, and M. Corner. Virtual compass: relative positioning to sense mobile social interactions. *Pervasive Computing*, pages 1–21, 2010.

[5] Z. Cao, S. Liu, and J. Roning. Omni-directional vision localization based on particle filter. In *Fourth International Conference on Image and Graphics*, pages 478–483, August 2007.

[6] Y. Chen., G. Chandrasekaran, E. Elnahrawy, J.-A. Francisco, K. Kleisouris, X. Li, R. P. Martin, R. S. Moore, and B. Turgut. GRAIL: A general purpose localization system. *Sensor Review, special edition, Localization Systems*, 28:115–124, 2008.

[7] D. Crisan and A. Doucet. A survey of convergence results on particle filtering methods for practitioners. *IEEE Transactions on Signal Processing*, 50:736–746, 2002.

[8] F. Dellaert, D. Fox, W. Burgard, and S. Thrun. Monte Carlo localization for mobile robots. In *IEEE International Conference on Robotics and Automation*, volume 2, pages 1322–1328, 1999.

[9] P. Djuric, M. Vemula, and M. Bugallo. Target tracking by particle filtering in binary sensor networks. *Signal Processing, IEEE Transactions on*, 56(6):2229–2238, 2008.

[10] E. Elnahrawy, J.-A. Francisco, and R. P. Martin. Bayesian localization in wireless networks using angle of arrival. In *Proceedings of ACM SenSys*, pages 272–273, November 2005.

[11] D. Fox. Adapting the sample size in particle filters through KLD-sampling. *I. J. Robotic Res.*, 22(12):985–1004, 2003.

[12] V. Fox, J. Hightower, L. Liao, D. Schulz, and G. Borriello. Bayesian filtering for location estimation. *IEEE Pervasive Computing*, 2:24–33, 2003.

[13] I. Getting. The global positioning system. *IEEE Spectrum*, 30(12):36–47, December 1993.

[14] F. Gustafsson, F. Gunnarsson, N. Bergman, U. Forssell, J. Jansson, R. Karlsson, and P.-J. Nordlund. Particle filters for positioning, navigation, and tracking. *IEEE Transactions on Signal Processing*, 50:425–437, 2002.

[15] A. Haeberlen, E. Flannery, A. Ladd, A. Rudys, D. Wallach, and L. Kavraki. Practical robust localization over large-scale 802.11 wireless networks. In *Proceedings of the Tenth ACM International Conference on Mobile Computing and Networking (MobiCom)*, pages 70–84, September 2004.

[16] A. Harter, A. Hopper, P. Steggles, A. Ward, and P. Webster. The anatomy of a context-aware application. In *Proceedings of 5th Annual International Conference on Mobile Computing and Networking*, pages 59–68, August 1999.

[17] A. Howard. Multi-robot simultaneous localization and mapping using particle filters. *The International Journal of Robotics Research*, 25(12):1243–1256, December 2006. Jet Propulsion Laboratory, California Institute of Technology, Pasadena, California 91109, U.S.A.,.

[18] L. Hu and D. Evans. Localization for mobile sensor networks. In *Proceedings of the 10th annual international conference on Mobile computing and networking*, pages 45–57. ACM, 2004.

[19] W. Hu, T. Downs, G. Wyeth, M. Milford, and D. Prasser. A modified particle filter for simultaneous robot localization and landmark tracking in an indoor environment. In *Australasian Conference on Robotics and Automation (ACRA)*, December 2004.

[20] J. Letchner, D. Fox, and A. LaMarca. Large-scale localization from wireless signal strength. In *Proceedings of the National Conference on Artificial Intelligence (AAAI-05)*, pages 15–20, July 2005.

[21] A. Milstein, J. N. Sánchez, and E. Williamson. Robust global localization using clustered particle filtering. In *Eighteenth National Conference on Artificial Intelligence*, pages 581–586, 2002.

[22] M. Montemerlo, S. Thrun, D. Koller, and B. Wegbreit. FastSLAM: A factored solution to the simultaneous localization and mapping problem. In *Proceedings of the National conference on Artificial Intelligence*, pages 593–598, 2002.

[23] M. Montemerlo, S. Thrun, D. Koller, and B. Wegbreit. FastSLAM 2.0: An improved particle filtering algorithm for simultaneous localization and mapping that provably converges. In *Proceedings of the Sixteenth International Joint Conference on Artificial Intelligence (IJCAI)*, pages 1151–1156, August 2003.

[24] M. Montemerlo, S. Thrun, and W. Whittaker. Conditional particle filters for simultaneous mobile robot localization and people-tracking. In *IEEE Conference on Robotics and Automation*, volume 1, pages 695–701, May 2002.

[25] D. Niculescu and B. Nath. Ad hoc positioning system (APS). In *Proceedings of Globecom*, pages 2926–2931, November 2001.

[26] D. Niculescu and B. Nath. VOR base stations for indoor 802.11 positioning. In *Proceedings of the 10th Annual ACM International Conference on Mobile Computing and Networking (MobiCom)*, pages 2926–2931, September 2004.

[27] O. Ozdemir, R. Niu, and P. Varshney. Tracking in wireless sensor networks using particle filtering: Physical layer considerations. *Signal Processing, IEEE Transactions on*, 57(5):1987 –1999, may 2009.

[28] S. Parsons. A note on robot localization. Technical report, Dept. of Computer and Information Science. Brooklyn College, City University of New York, 2005.

[29] E. Parzen. On estimation of a probability density function and mode. *Journal of Annals Mathematical Statistics*, 33(3):1065–1076, 1962.

[30] M. Pitt and N. Shephard. Filtering via simulation: Auxiliary particle filters. *Journal of the American Statistical Association*, pages 590–599, 1999.

[31] N. Priyantha, A. Chakraborty, and H. Balakrishnan. The CRICKET location-support system. In *Proceedings of the 6th Annual International Conference on Mobile Computing and Networking*, pages 32–43, August 2000.

[32] T. Röfer, T. Laue, and D. Thomas. Particle-filter-based self-localization using landmarks and directed lines. In *Proceedings of RoboCup 2005 Symposium*, pages 608–615, July 2005.

[33] B. Romanowicz, M. Cara, J. F. Fel, and D. Rouland. Geoscope: A french initiative in long-period three-component global seismic networks. *Eos, Transactions American Geophysical Union*, 65(42):753 –753, 1984.

[34] N. Roy, H. Wang, and R. R. Choudhury. I am a smartphone and i can tell my user's walking direction. In *Proceedings of the 12th Annual International Conference on Mobile Systems, Applications, and Services*, pages 329–342, June 2014.

[35] S. Saha, K. Chaudhuri, D. Sanghi, and P. Bhagwat. Location determination of a mobile device using IEEE 802.11 access point signals. In *Proceedings of IEEE Wireless Communications and Networking Conference (WCNC)*, volume 3, pages 1987–1992, March 2003.

[36] S. Sen, R. R. Choudhury, B. Radunovic, and T. Minka. Precise indoor localization using phy layer information. In *Proceedings of the 10th ACM Workshop on Hot Topics in Networks*, pages 18:1–18:6, November 2011.

[37] V. Seshadri, G. Zaruba, and M. Huber. A bayesian sampling approach to in-door localization of wireless devices using received signal strength indication. In *Pervasive Computing and Communications (PerCom 2005)*, pages 75–84, 2005.

[38] C. Siagian and L. Itti. Biologically inspired mobile robot vision localization. *Robotics, IEEE Transactions on*, 25(4):861 –873, Aug 2009.

[39] B. Turgut and R. P. Martin. Restarting particle filters: an approach to improve the performance of dynamic indoor localization. In *submitted to IEEE Global Communications Conference (Globecom)*, 2009.

[40] B. Turgut and R. P. Martin. Using a-priori information to improve the accuracy of indoor dynamic localization. In *submitted to The 12-th ACM International Conference on Modeling, Analysis and Simulation of Wireless and Mobile Systems (MSWIM)*, 2009.

[41] N. Vlassis, B. Terwijn, and B. Krose. Auxiliary particle filter robot localization from high-dimensional sensor observations. In *IEEE International Conference on Robotics and Automation (ICRA)*, volume 1, pages 7–12, May 2002.

[42] Widyawan, M. Klepal, and S. Beauregard. A novel backtracking particle filter for pattern matching indoor localization. In *MELT '08: Proceedings of the first ACM international workshop on Mobile entity localization and tracking in GPS-less environments*, pages 79–84, September 2008.

[43] H. Wymeersch, J. Lien, and M. Win. Cooperative localization in wireless networks. *Proceedings of the IEEE*, 97(2):427 –450, Feb. 2009.

[44] J. Yin, Q. Yang, and L. Ni. Learning adaptive temporal radio maps for signal-strength-based location estimation. *Mobile Computing, IEEE Transactions on*, 7(7):869–883, July 2008.

[45] M. Youssef and A. Agrawala. The Horus location determination system. *Wireless Networks*, 14(3):357–374, 2008.