UNIVERSITY OF CALGARY

Positioning Robots with Dead Reckoning and Received Signal Strength

by

Ahmed Arafa

A THESIS

SUBMITTED TO THE FACULTY OF GRADUATE STUDIES

IN PARTIAL FULFILLMENT OF THE REQUIREMENTS FOR THE

DEGREE OF MASTER OF SCIENCE

DEPARTMENT OF ELECTRICAL AND COMPUTER ENGINEERING

CALGARY, ALBERTA

MAY, 2010

UNIVERSITY OF CALGARY

FACULTY OF GRADUATE STUDIES

The undersigned certify that they have read, and recommend to the Faculty of Graduate Studies for acceptance, a thesis entitled "Positioning Robots with Dead Reckoning and Received Signal Strength" submitted by Ahmed Arafa in partial fulfillment of the requirements for the degree of Master of Science.

Supervisor, Dr. Geoffrey Messier
Department of Electrical and Computer Engineering

Dr. Sebastian Magierowski
Department of Electrical and Computer Engineering

Dr. John Nielsen
Department of Electrical and Computer Engineering

Dr. Thomas Williams
Department of Geomatics Engineering

May 19/10

Date

Abstract

Location information is of utmost importance for research and applications in both wireless sensor networks (WSNs) and mobile robotics. This work demonstrates the feasibility of combining two different location estimation techniques to improve the overall location and tracking capabilities of low complexity devices such as mobile sensor nodes in a WSN.

The system model is made up of stationary wireless devices known as *beacons* whose locations are known, and the low complexity device trying to know its position. The low complexity device localizes using the robot's dead reckoning (DR) position estimate and the received signal strength (RSS) range estimates made between itself and the beacons.

A Gaussian DR positioning error model is developed, and verified based on empirical measurements collected with a robot chassis. A Kalman filter, utilizing the Gaussian model is implemented to track the robot. The accuracy of the filter is compared with the Cramer-Rao lower bound (CRLB).

Acknowledgements

I would like to begin by thanking Dr. Geoffrey Messier for being an excellent supervisor. I learned alot under his supervision.

I would like also to thank my committe members, Dr. Sebastian Magierowski, Dr. John Nielsen and Dr. Thomas Williams for their time and their valuable suggestions.

I would like to thank Junjie liu, Jean François Bousquet and Zhixing Zhao for their friendship and the fun time we had together. I would also like to thank Mostafa for making my stay in Calgary comfortable.

Finally, I offer a special thanks to my parents: thank you for your support, encouragement, and for making this possible.

Dedication

To my parents

Table of Contents

List of Figures

List of Symbols

$s(t)$	:Transmitted signal
$r_n(t)$	:Received signal at robot n
a_k	:Amplitude of the received signal
T_k	:Time delay of signal
φ_k	:Phase of signal
τ	:Time offset between the clocks of two nodes
X_D	:Desired robot X position
Y_D	:Desired robot Y position
X	:Actual robot X position
Y	:Actual robot Y position
A_i	:Random variable representing the angle turned at segment i of the robot path
R_i	:Random variable representing the distance moved at segment i of the robot path
α_i	:Desired angle turned at segment i
β_i	:Desired distance moved at segment i
Δ_A	:Uniformly distributed deviation from the desired angle
Δ_R	:Uniformly distributed deviation from the desired distance
$f_{(RA)}(r, a)$	:Joint probability density function of random variables R and A
X_i	:Random variable representing robot X position after moving segment i
Y_i	:Random variable representing robot Y position after moving segment i
$\mu_{X,i}$	:Mean of the random variable X position after moving segment i
$\mu_{Y,i}$	:Mean of the random variable Y position after moving segment i
$\sigma^2_{X,i}$	:Variance of the random variable X position after moving segment i
$\sigma^2_{Y,i}$	:Variance of the random variable Y position after moving segment i
U	:Uniform distribution
$\mathcal{N}$	:Normal distribution
$X_{b,k}$	:Beacon k X position
$Y_{b,k}$	:Beacon k Y position
PL	:Path loss
η	:Path loss exponent
λ	:Wavelength
$\sigma^2_X[n]$	:DR variance in X direction
$\sigma^2_Y[n]$	:DR variance in Y direction
σ^2_R	:Shadowing variance

Z	:Log-normal shadow process with mean zero and variance σ_R^2
d_k	:Represents the distance between the robot position (X, Y) and the beacon position $(X_{b,k}, Y_{b,k})$
$\boldsymbol{\theta}$	:Vector containing the X and Y robot position
X_{DR}	:DR X position
Y_{DR}	:DR Y position
R_k	:Received power at robot's (X, Y) position from beacon $(X_{b,k}, Y_{b,k})$
$\mathbf{z}[n]$	:Observation vector
$\boldsymbol{\mu}(\boldsymbol{\theta})$	:Mean of the observation vector
$\mathbf{C}(\boldsymbol{\theta})$	:Covariance matrix of the observation noise
$\widehat{\boldsymbol{\theta}}$	:Represents an estimate of the robot Cartesian position $\boldsymbol{\theta}$
$\mathbf{I}(\theta)$	:FIM for estimating the parameter θ
$\mathbf{s}[n]$	:State vector at time n
$\mathbf{A}[n]$	:Matrix defining the robot kinematics
$\mathbf{u}[n]$	:Process noise
$\mathbf{Q}[n]$	:Covariance matrix of the process noise at time n
$\mathbf{z}[n]$	:Observation vector at time n
$\mathbf{H}[n]$	:Observation matrix at time n
$\mathbf{w}[n]$	:Observation noise at time n
$\mathbf{C}[n]$	:Covariance matrix of the observation noise at time n
$\mathbf{I}(\mathbf{s}[n])$	:FIM for estimating the vector $\mathbf{s}[n]$
$\mathcal{P}(\mathbf{z}[n]\|\mathbf{s}[n])$	:Probability of $\mathbf{z}[n]$ given s[n]
$\nabla_{\mathbf{s}[n]}$	:First-order partial derivative operator with respect to s[n]
$\mathbf{S}[n]$	:Sequence of states from time 1 up to time n
$\mathbf{Z}[n]$	:Sequence of observations from time 1 up to time n
$\mathcal{P}(\mathbf{S}[n], \mathbf{Z}[n])$	:Joint probability of $\mathbf{S}[n]$ and $\mathbf{Z}$[n]
ν_n	:FIM for estimating $\mathbf{S}[n-1]$
ξ_n	:FIM showing the correlation between $\mathbf{S}[n-1]$ and $\mathbf{s}[n]$
γ_n	:FIM for estimating $\mathbf{s}[n]$
$\mathbf{a}(\mathbf{s}[n])$	:Non linear function of $\mathbf{s}[n]$ defining robot kinematics matrix
$\mathbf{h}(\mathbf{s}[n])$	:Non linear function of $\mathbf{s}[n]$ defining observation matrix
$\widetilde{\mathbf{A}}[n]$	:Jacobian of $\mathbf{a}(\mathbf{s}[n])$ evaluated at the true value of $\mathbf{s}[n]$
$\widetilde{\mathbf{H}}[n]$	:Jacobian of $\mathbf{h}(\mathbf{s}[n])$ evaluated at the true value of $\mathbf{s}[n]$

$v_X[n]$	:Robot's X velocity component
$v_Y[n]$	:Robot's Y velocity component
Δt	:Interval between observations
$u_X[n]$	:Maximum change in velocity component in the X direction from time sample n to $n-1$
$u_Y[n]$	:Maximum change in velocity component in the Y direction from time sample n to $n-1$
σ^2_{Vx}	:Variance of $u_X[n]$
σ^2_{Vy}	:Variance of $u_Y[n]$
Δ_X	:Spacing between the beacons in the X direction
Δ_Y	:Spacing between the beacons in the Y direction
N_c	:Number of columns
N_r	:Number of rows
$\hat{\mathbf{s}}[n-1\|n-1]$	:State estimate at time $n-1$
$\hat{\mathbf{s}}[n\|n-1]$	:Predicted state $\mathbf{s}[n]$ at time n given the previous state $\mathbf{s}[n-1]$
$\mathbf{M}[n-1\|n-1]$	:MSE estimate at time $n-1$
$\mathbf{M}[n\|n-1]$	:Predicted MSE estimate at time n given the previous MSE at time $n-1$
$\mathbf{K}[n]$	:Kalman gain at time n
$\hat{\mathbf{s}}[n\|n]$	:State estimate at time n
$\mathbf{M}[n\|n]$	:MSE estimate at time n
$\breve{\mathbf{A}}[n-1]$	:Partial derivative of $\mathbf{a}(\mathbf{s}[n-1])$ with respect to $\mathbf{s}[n-1]$, evaluated at the previous state estimate $\hat{\mathbf{s}}[n-1\|n-1]$
$\breve{\mathbf{H}}[n]$	:Partial derivative of $\mathbf{h}(\mathbf{s}[n])$ with respect to $\mathbf{s}[n]$ evaluated at the predicted state estimate $\hat{\mathbf{s}}[n\|n-1]$
ν_{n+1}	:FIM for estimating $\mathbf{S}[n-1]$
γ_{n+1}	:FIM for estimating $\mathbf{s}[n]$
κ_{n+1}	:FIM for estimating $\mathbf{s}[n+1]$
ξ_{n+1}	:FIM defining the correlation between $\mathbf{S}[n-1]$ and $\mathbf{s}[n]$
χ_{n+1}	:FIM defining the correlation between $\mathbf{S}[n-1]$ and $\mathbf{s}[n+1]$
ω_{n+1}	:FIM defining the correlation between $\mathbf{s}[n]$ and $\mathbf{s}[n+1]$

List of Abbreviations

DR	:Dead reckoning
RSS	:Received signal strength
TOA	:Time of arrival
TDOA	:Time difference of arrival
AOA	:Angle of arrival
CRLB	:Cramer Rao lower bound
WSN	:Wireless sensor network
MSN	:Mobile sensor network
LAN	:Local area network
RF	:Radio frequency
GPS	:Global positioning system
SLAM	:Simultaneous localization and mapping
IR	:Infra red
CI	:Covariance intersection
MSE	:Mean squared error
CW	:Clockwise
CCW	:Counter clockwise
QQ plot	:quantile quantile plot
dB	:decibel
FIM	:Fisher information matrix
tr	:Trace of a matrix
det	:Determinant of a matrix
ln	:Natural log
erfc	:Complementary error function

Chapter 1

Introduction

This thesis studies the feasibility of combining two low complexity localization methods to infer the position of a robot. A robot in this thesis represents any low complexity device such as a mobile sensor node in a wireless sensor network. The location techniques used are the robot's dead reckoning (DR) position estimate and the received signal strength (RSS) range measurements collected by the robot's wireless module. These location techniques are ideal for low complexity networks, such as mobile sensor networks, due to their simplicity. A Gaussian DR error model characterizing the error in the robot motion is presented and is verified based on empirical measurements on a real robot chassis. A Cramer Rao lower bound (CRLB) and a Kalman filter are implemented to assess the performance of the combined DR/RSS system. The DR error model serves to simplify the analysis of the Cramer Rao lower bound (CRLB) and the Kalman filter simulations.

Chapter 1 is organized as follows. Section 1.1 introduces the motivation behind low complexity localization. Wireless localization techniques are discussed in Section 1.2. The proposed hybrid dead reckoning and received signal strength localization scheme is presented in Section 1.3. Section 1.4 discusses the techniques used for evaluating the performance of the proposed hybrid positioning scheme. Section 1.5 discusses the Gaussian model for DR error. The thesis contributions are listed in Section 1.6. Finally, Section 1.7 presents the thesis outline.

1.1 Low Complexity Localization

Recent research has focused on the deployment of very low complexity wireless devices, such as wireless sensor nodes. Wireless sensor networks have applications in various fields and have been proposed for applications such as smart homes, habitat/temperature monitoring, animal tracking and traffic monitoring.

In [2], a wireless sensor network is proposed to monitor the position of coal miners and the coal mine environment for harmful gases. In [3], a new wireless sensor network topology is proposed to enhance traffic monitoring. Wireless sensor networks are proposed to monitor the health of bridges against erosion [4]. In [5], wireless sensor nodes are used in precision agriculture to monitor soil characteristics such as pH, temperature and chemical concentrations.

Sensor nodes operate under severe resource constraints because they run on low battery power and have limited memory storage. This results in limited computation and communication capabilities. Because of this, individual sensor nodes can only sense a limited portion of the environment. Hence, a typical WSN consists of hundreds or thousands of sensor nodes deployed either randomly or according to some predefined distribution.

In order to prolong the lifetime of the sensor network, the sensor nodes have to consume a minimum amount of energy to perform its tasks. Thus, any localization scheme used by a sensor node must be very low complexity.

In traditional sensor networks, the sensor nodes are static. Recent research has focused on using mobile sensor nodes to enhance network performance [6, 7, 8, 9, 10]. This is done by mounting the sensor node on a mobile platform such as a mobile robot

[6]. A mobile sensor network (MSN) has a number of advantages.

One of the fundamental performance criteria of a sensor network is coverage. Coverage is the quality of surveillance that a network can provide, or how effectively the sensor nodes deployed in a certain region can detect intruders or targets. After initial deployment, the areas of interest that need to be covered may change. In this case mobile sensors offer a flexible means to move to the area of interest and monitor it. Also, an intruder in a MSN is more likely to be detected as the mobile nodes patrol the field [6].

Sensors are unreliable units. If a sensor node fails, some area of the network may be uncovered. In this case, mobile sensors can move and replace the node that has died. This also improves the connectivity of the sensor network [8].

In situations where manual deployment is not possible, such as battle field monitoring, toxic regions and disaster areas, mobile sensor nodes are an asset since they can deploy in the field and provide the appropriate coverage [6].

Position information is critical for a mobile sensor network and since the mobile nodes move, they must be able to estimate their position. Numerous sensor network applications require location information, such as monitoring patients in hospitals [9], search and rescue missions [11], target tracking [11], precision agriculture [12], water quality monitoring [12],. indoor air quality monitoring [12], intrusion detection [12], traffic monitoring [7] and surveillance [7]. Location information is also essential for communication protocols that use geographical routing [13, 10], and to successfully guide a mobile sensor in a region of interest by avoiding obstacles.

1.2 Localization Techniques

This section presents techniques that can be used to estimate the position of low complexity devices or robots. Assuming a system with N robots in a multipath channel, the signal captured by the receive antenna of robot n can be mathematically modeled as

$$r_n(t) = \sum_{k=1}^{K} a_k s(t - T_k) \exp(\varphi_k) \tag{1.1}$$

where K is the number of multipath signals, a_k is the amplitude of the received signal, $s(t)$ is the waveform transmitted by the source, T_k is the time delay of the signal, and φ_k represents the phase of the signal arriving at the receive antenna.

Positioning techniques based on received signal strength measurements, time of arrival/ time difference of arrival measurements, angle of arrival measurements and dead reckoning positioning scheme are presented in Sections 1.2.1, 1.2.2, 1.2.3 and 1.2.4 respectively.

1.2.1 Received Signal Strength

Received signal strength (RSS) range measurements are determined by first measuring the signal power of the incident signal, and then determining the distance from the source emitting the signal using the radio channel model.

RSS measurements do not require additional hardware since most radio modules are able to measure RSS power values as part of their normal operation. Inspite of the fact that RSS is a simple and inexpensive approach for localization, its sources of error need to be carefully studied owing to the unpredictable nature of radio channel attenuation.

RSS range measurements have been studied in [9, 14, 15, 16]. In [12] and [16]experimental

results showed that RSS location estimation reported accuracies of approximately 1 meter in indoor environments. In [17] a system known as RADAR estimates position based on the 802.11 wireless local area network (LAN). Three base stations were deployed in a floor area of dimension 43.5 m by 22.5 m. The base stations would then measure the RSS from a mobile station and estimate its position. An accuracy in the range of 2-3 meters was achieved.

Although RSS is seen as a coarse source of localization, in dense networks where the distance between nodes is small, RSS range estimation is more accurate [15] and can perform as well as TOA estimation [18].

1.2.2 Time Of Arrival/ Time Difference Of Arrival

Time of arrival (TOA) localization is based on measuring the time taken for the first signal to travel from the transmitter to the receiver. The distance from the transmitter is calculated by multiplying the speed of the signal by the TOA value. The signal used can be of any frequency e.g. radio frequency (RF) or acoustic signals.

A common location estimation technique based on RF TOA is the Global Positioning System (GPS), which has been suggested in [19] to obtain location information in ad-hoc networks. A receiver on the ground estimates its 3D position via GPS by measuring the TOA value from four satellites in space. Although GPS can be accurate to a few meters, it is often not practical in sensor network applications because it drains considerable power out of the nodes [20], and does not work in environments such as indoors where the view of the satellites is blocked [21, 16, 22, 23, 24].

A number of indoor positioning systems based on ultrasonic range measurements are discussed in [25]. The majority of these schemes employ TOA/TDOA to infer distance

from the source emitting the signal. The reason for choosing ultrasound frequency instead of radio frequency (RF) is because the RF signals need very high resolution clocks to detect small distances. This is due to the very high propagation speed of the RF signal compared to the ultrasound signal.

Ultrasound localization measurements have been investigated in [26, 27, 28]. It was shown that it provides high accuracy in dense networks. However, ultrasound ranging is limited to a few meters. In [27, 28], the maximum range of ultrasound localization is 3-5 meters.

The Cricket positioning system [25] developed by MIT uses ultrasound ranging. It is low cost and has an accuracy of 1-2 cm. However, its maximum range is only 10 m. Another drawback of ultrasound is that its speed is highly correlated to temperature. In order to get accurate distance measurements, the mean ambient temperature must be correctly measured [29].

TOA localization measurements have been studied in [12, 18]. Although TOA typically has a better performance than received signal strength (RSS) range measurements, it is impractical for low complexity devices due to hardware complexity and high cost.

TOA requires knowledge of when the signal was transmitted in order to determine position. Synchronization is required in order to calculate TOA. Each node would have its own clock. Due to inaccuracies in the clock, the clocks of each node would have a time offset relative to one another. Before two nodes determine their position based on TOA, the relative offset between the two clocks need to be determined. This is done as shown in Figure 1.1.

Node 1 would send a packet at time t_1 relative to its local clock.

Node 2 would receive this packet at time t_2 relative to its local clock. Time t_2 is equal to

time t_1 plus the propagation delay, T, between nodes 1 and 2, plus the offset τ between the clocks of nodes 1 and 2.

Node 2 would then transmit a packet at time t_3 relative to its local clock, to node 1. This packet contains the times t_1 and t_2.

Node 1 would then receive this packet at time t_4 relative to its local clock.

Time t_4 is equal to time t_3 plus the propagation delay, T, between nodes 1 and 2, minus the offset τ between the clocks of nodes 1 and 2.

The offset τ is then calculated by subtracting t_4 from t_2, such that $t_2 - t_4 = (t_1 + T + \tau) - (t_3 + T - \tau) = t_1 - t_3 + 2\tau$, therefore $\tau = \frac{1}{2}(t_2 - t_4 - t_1 + t_3)$. The

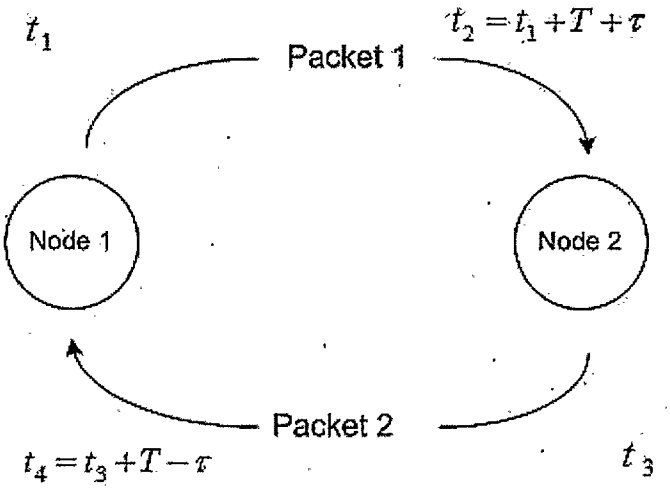

Figure 1.1: Two nodes synchronizing.

result of the time offset, τ is then sent to Node 2 and the two nodes are synchronized. Another way of node synchronization assumes all nodes update their clocks relative to a reference clock in the network [30].

Attaining the desired ranging accuracy requires the use of complex devices to achieve more precise clock synchronization which adds to the complexity and cost of the low

complex nodes [17]. Moreover, in an indoor environment, the signal suffers from multiple reflections from the walls and objects in between the transmitter and receiver, adding ambiguity in measurements based on TOA.

Time difference of arrival (TDOA) or hyperbolic positioning works by using the time difference of the signal propagation between the robot with unknown position and the beacons. However, the beacons must still be synchronized together.

Once the TDOA is found between two beacons and the robot with unknown position, the distance between the beacons is calculated and a nonlinear hyperbolic algorithm is implemented to define a hyperbola on the possible position of the robot. This is repeated with the signals from the other beacons and the intersection of these hyperbolas is the location of the robot in a 2D plane [31, 32].

Similar to the TOA positioning, TDOA still has synchronization requirements which adds to the cost and complexity of the network.

1.2.3 Angle Of Arrival

Angle of arrival (AOA) localization measurements have been proposed in [33] to enhance position estimates in ad-hoc networks. AOA works by providing information on the direction of the signal. This is done by mounting two or more antennas on the robot with known fixed locations relative to one another. By measuring the phase of the signal received by each antenna, the direction of the signal is calculated. The drawback of using AOA positioning is that it requires an antenna array which adds to the robot's cost and size [18, 17]. Hence it would not be practical for small, low complexity devices.

1.2.4 Dead Reckoning

Dead Reckoning (DR) position estimation is based on using odometers, gyroscopes and accelerometers to determine the position of a robot that departs from a known starting point.

The errors in the distance and angle measurements collected by the odometers accumulate over time and cause the robot to lose track of its position. These errors are caused by many factors including: wheel slippage, precision of the wheel encoders, differences in the diameter of the wheels, and surface roughness.

In [34], the authors used dead reckoning position estimation to track pedestrians given an initial starting point. Sports pedometers and accelerometers were used to track the distance moved. Two and three-axis compasses and gyroscopes were used to track orientation. Case studies were performed on city streets and on a tourist trail. In each place, a different combination of distance and orientation sensors were used.

In [35], the paper proposes combining dead reckoning position estimation, using two wheel encoders with eight range sensors mounted on a mobile robot. The location information from both techniques is combined using a least squares algorithm in order to improve overall accuracy.

1.3 Hybrid Dead Reckoning And Received Signal Strength Localization

In this thesis, the positioning technique that is investigated is based on combining dead reckoning (DR) position estimates with received signal strength (RSS) range measurements.

Although the TOA and AOA techniques discussed in the previous section can be

used to determine position, they require additional hardware that will add to the cost and size of the low complexity devices being considered in this thesis. Also, this increased hardware will drain the battery more quickly, which will reduce the lifetime of the whole network.

DR and RSS position estimates are the best candidates for the localization of low complexity devices for a number of reasons. DR hardware consists of odometers and compasses which are widely available and are low cost. Most low complexity wireless devices also have the capability to measure RSS, without additional hardware and cost.

DR and RSS position estimates are two schemes that complement each other. Moving from an initial starting point along the robot path, DR accuracy can be quite good. As DR accuracy degrades over time, the RSS measurements are necessary to place the robot back on track.

In order to improve DR positioning accuracy, research has been done into combining DR with other positioning schemes, such as signal time of arrival measurements [23], ultrasonic ranging [26, 29], angle of arrival measurements [33], received signal strength maps [36], radar [37] and range measurements to landmarks [35, 38].

In [26] the authors discuss how to detect and isolate faults due to DR errors when the robot collides with undetected obstacles. The robot's DR position estimates are combined with ultrasonic range measurements between the robots in a maximum likelihood estimator to detect and avoid incorrect DR position estimates.

In [23], the authors talk about combining DR position estimates with time of arrival (TOA) range measurements from beacons to determine the robot position. The DR position estimates and TOA range measurements are combined in a Kalman filter algorithm to track the robot. The Kalman filter is also used for simultaneous localization

and mapping (SLAM).

Position tracking of mobile terminals is discussed in [36]. The paper proposes using a recursive Bayesian filter that uses DR estimates to track mobile devices.

Although all the above localization techniques help improve the DR position error estimate, using the RSS information available in the nodes transceiver to correct DR position estimation has received less attention.

DR position estimates and RSS range measurements have been studied in [39]. The paper uses a robot platform with a Crossbow MicaZ sensor node. The robot is equipped with encoders and a compass to determine the distance and direction moved by the robot. Infra red (IR) sensors are placed on the sides of the robot to avoid hitting obstacles. The encoder and compass readings are connected to a microcontroller that calibrates the distance and angle moved by the robot.

The paper uses a maximum likelihood approach to find a propagation model for the fading channel. A covariance intersection (CI) algorithm is proposed that combines the convex combination of the mean and covariance of the RSS and DR measurements. Results show that the CI algorithm improves the mean distance error.

While there is some similarity between their work and the work in this thesis, there are also important differences. First, this thesis presents a Gaussian DR error model describing the robot motion and is verified based on empirical measurements. Second, a theoretical bound, the Cramer Rao lower bound (CRLB), for estimating the robot position is derived and implemented for both static and dynamic robots. The performance of this bound is further compared to a practical filter known as the Kalman filter. Our work also studies how the Kalman filter and the CRLB behave under different system parameters such as the shadowing standard deviation.

1.4 Evaluation Of DR/RSS Localization Performance

The proposed DR/RSS system is evaluated using two schemes, namely the Cramer Rao lower bound (CRLB) and a Kalman filter simulation. The CRLB technique is chosen because it shows the best performance that the hybrid DR/RSS system can ever achieve. Therefore it is used as a benchmark to assess the performance of practical estimators. The Kalman filter is the practical estimator used in this thesis to track the robot using the DR/RSS location estimates.

The CRLB is a fundamental estimation bound since it computes the minimum variance that an unbiased estimator could ever reach. The CRLB has been used in [40] to study the performance of the RSS/TOA and RSS/TDOA hybrid schemes. In [18], the CRLB is calculated for each of the RSS, TOA and AOA location schemes. The CRLB is a function of many parameters such as the number of robots with known and unknown positions, the geometry and density of the robots, and the channel characteristics.

The Kalman filter is a type of estimation technique used in many applications, such as tracking ground vehicles [23, 41, 29], and airborne targets in radar applications [42]. It is also used in estimating the radio channel [43], and restoring images contaminated by noise [44]. In this thesis, it is used to track and estimate the position of a robot. The state equations describe the motion model of the robot, and the observation equations describe the measurements made by the robot. The Kalman filter is a recursive filter that finds the minimum mean squared error (MMSE) of the parameter to be estimated. The filter equations also assume that the system to be corrupted by white Gaussian noise. This simplifies analysis and gives a closed form solution.

1.5 Gaussian Model For DR Error

The CRLB and Kalman filter require statistical error models for the DR position estimates and the RSS range measurements. The statistical error of the average power RSS measurements is well understood to be log-normally distributed [45, 18]. However, this thesis will present a new model for DR error.

In Chapter 2, a new Gaussian DR error model is presented and verified based on empirical measurements conducted on a real robot chassis. A number of papers assume that DR error is Gaussian [29, 39, 26, 23, 36], but provide no proof of that assumption. The model shows that the final position of a robot based on dead reckoning measurements has a Gaussian distribution even after moving over a relatively short path. The benefit of this model is that it simplifies the derivation of the CRLB and Kalman filter, and is easily incorporated into localization simulations.

1.6 Thesis Contributions

The contributions of this thesis are as follows:

a) A Gaussian model is developed for dead reckoning (DR) positioning error and is verified using empirical measurements on a real robot chassis. This model will help assist in evaluating the accuracy of localization schemes that combine DR with some other localization schemes.

b) As an application of the Gaussian DR error model, a Cramer-Rao lower bound (CRLB) is derived that evaluates the accuracy of the combined DR/RSS localization scheme.

c) A Kalman filter is developed to track a robot using DR and RSS localization in-

formation in an indoor environment. The performance of this filter is simulated and compared to the CRLB. The close agreement between the CRLB and the Kalman filter further verifies the accuracy of the Gaussian model.

1.7 Thesis Outline

This thesis is organized as follows. Chapter 2 presents the robot chassis DR measurement errors on carpet and linoleum, and the derivation of the Gaussian DR error model. Chapter 3 presents the derivation of the RSS/DR CRLB from both a static and dynamic perspective. The Kalman filter that operates using combined DR and RSS information is developed and evaluated in Chapter 4. Concluding remarks are presented in Chapter 5.

Chapter 2

Dead Reckoning Error Model

This chapter presents a Gaussian model for DR error that is verified using measurements collected with a robot platform. Section 2.1 discusses the robot platform and the experiments conducted to model the DR errors. Section 2.2 presents the derivation and the verification of the Gaussian DR error model.

2.1 iRobot Create

The robot platform used in this thesis is an iRobot Create [1], shown in Figure 2.1. It is used in this thesis due to its popularity as a development kit and because it is based on the iRobot Roomba platform, which is used as a vacuum cleaner in many households.

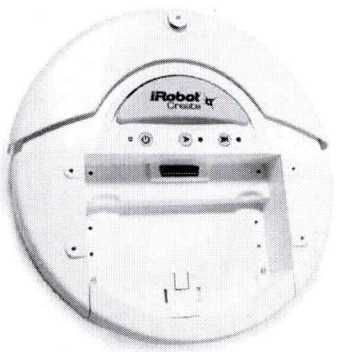

Figure 2.1: iRobot Create [1].

A simplified model of the robot control system is shown in Figure 2.2, where the input represents the commands/instructions that tell the robot where to go. The output

represents the actual distance moved by the robot. The sensor part represents the robot's wheel encoders, which sample the distance moved by the robot and provide feedback to the controller. When the distance reported by the sensors is equal to the desired distance, the robot stops. Therefore, the DR position estimate is the same as the desired position programmed into the robot controller.

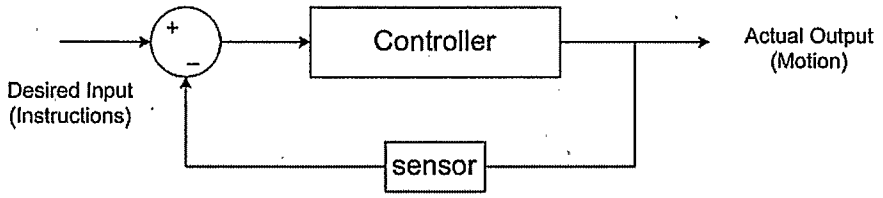

Figure 2.2: Simplified robot motion control system.

In real life, however, the actual position of the robot will not be the same as the desired position. This is due to many factors such as wheel slippage, precision of the wheel encoders and surface roughness. All of these factors will create inaccuracies in the information reported by the DR sensors.

All of this means that the definition of DR error used in this thesis is the difference between the desired distance programmed into the robot and the actual distance moved.

The iRobot Create moves by going in a straight line. In order to change its direction, the robot stops, rotates in place until it faces the desired direction and then moves in a straight line. This means, there are two types of DR errors, namely angle or rotation error and linear distance error.

Section 2.1.1 gives a brief overview of the tools used to program the robot. Sections 2.1.2 and 2.1.3 present the measurements done with the robot on carpet in order to characterize the DR angle and distance errors respectively. Similarly, Sections 2.1.4 and 2.1.5 present angle and distance measurements, respectively, done with

the robot on linoleum.

2.1.1 Programming The Robot

In order to instruct the robot to move to a certain position, a C/C++ program is written using a software tool called winAVR [46]. The program is then loaded from the computer into the iRobot command module using a USB cable.

The iRobot command module is an Atmel AVR ATMEGA168 8 bit microcontroller that is connected to the iRobot Create chassis. The iRobot Create speed ranges from 0-500 mm/s. A speed of 200 mm/s was chosen for all of the experiments as a suitable indoor speed that would not be alarming for the occupants of a household.

2.1.2 Angle Dead Reckoning Carpet Measurements

Angle experiments involved programming the robot to turn to angles of 45^0, 90^0 and 135^0 at a speed of 200 mm/s on carpet. A protractor is used to record the actual angle turned by the robot. The total number of angle measurements recorded is 180. The number of measurements recorded for each angle turned is 60, half of which is in the clockwise (CW) direction and the other half is in the counter clockwise (CCW) direction.

The difference between the desired and the actual angle turned by the robot for each measurement is recorded. The histograms in Figures 2.3, 2.4 and 2.5 show the distribution of the angle DR error for turning at an angle of 45^0, 90^0 and 135^0 respectively. The histogram in Figure 2.6 shows the distribution of the angle DR error for all of the three angles that were measured. A single distribution is used to group the angle DR error as a simplification so that the resultant histogram represents an approximation

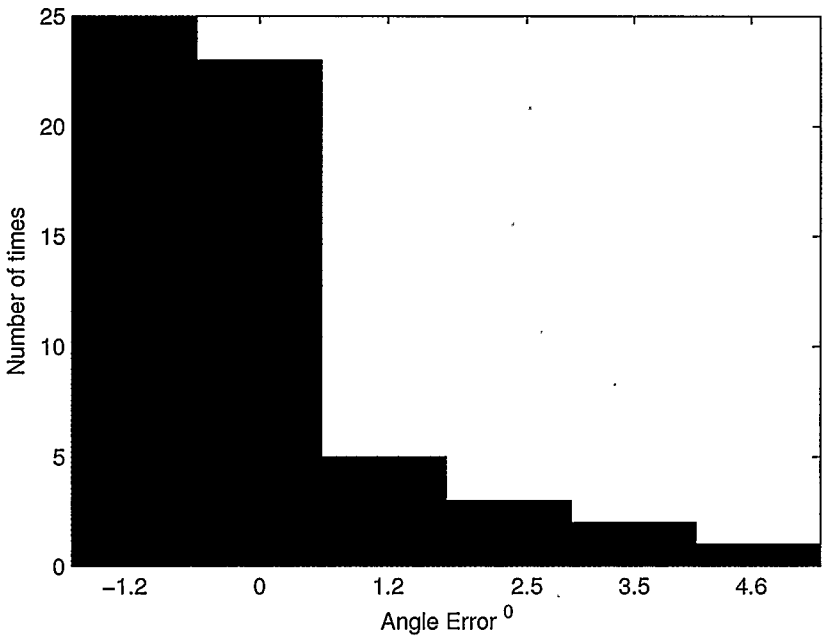

Figure 2.3: Angle DR error distribution for turning at an angle of 45^0.

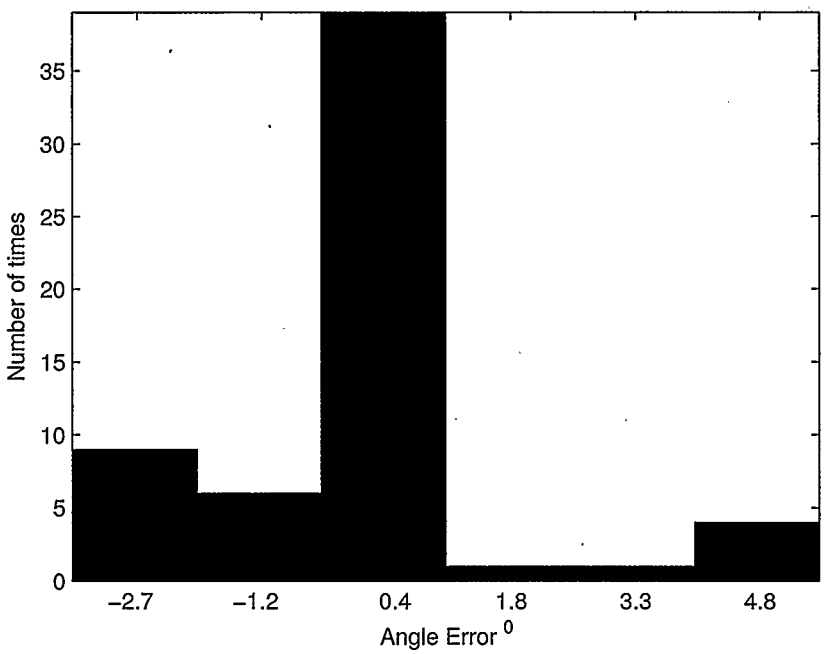

Figure 2.4: Angle DR error distribution for turning at an angle of 90^0.

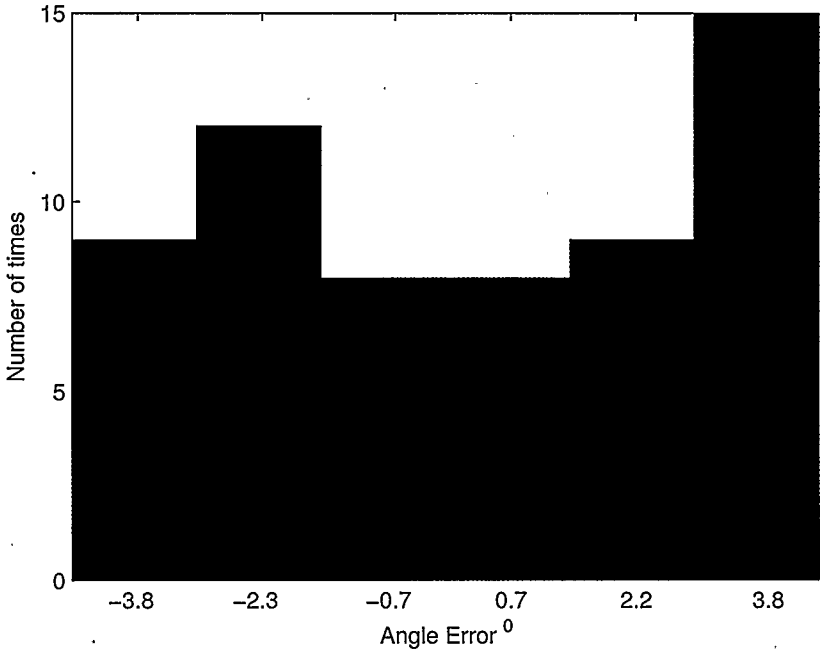

Figure 2.5: Angle DR error distribution for turning at an angle of 135^0.

for any angle turned.

2.1.3 Distance Dead Reckoning Carpet Measurements

Distance experiments involve programming the robot to move 1 m in a straight line at a speed of 200 mm/s. A total of 50 measurements were recorded. Similar to the angle DR measurements, the distance DR error distribution is found by noting the difference between the desired distance and the actual distance moved by the robot. These values are shown in Figure 2.7. Note that the number of measurements done to characterize the distance DR error histogram is far less than the number for the angle DR error histogram. This is because angle DR error has a larger effect on the accuracy of the robot position. A small angle error at the start of the robot motion will become

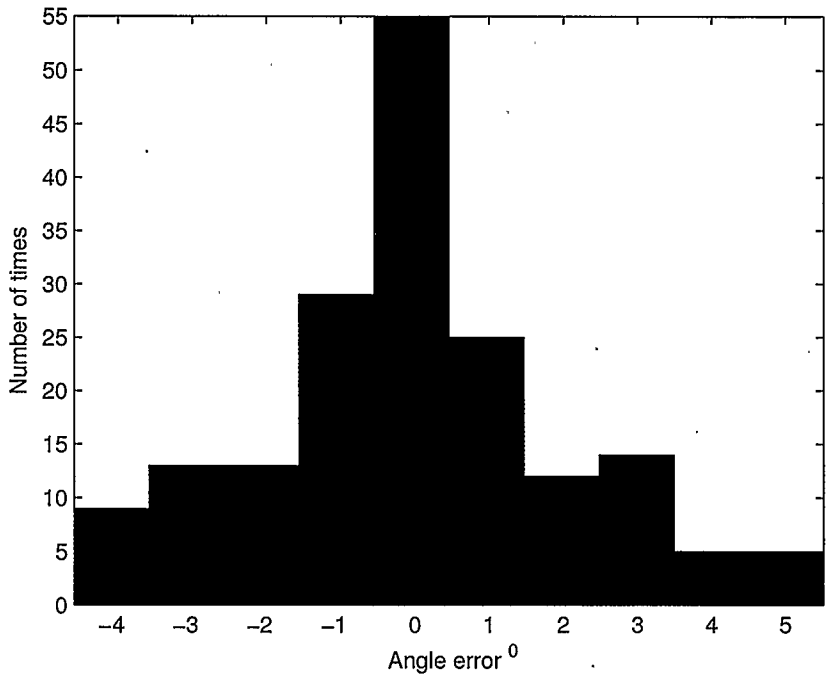

Figure 2.6: Angle DR error distribution for carpet measurements.

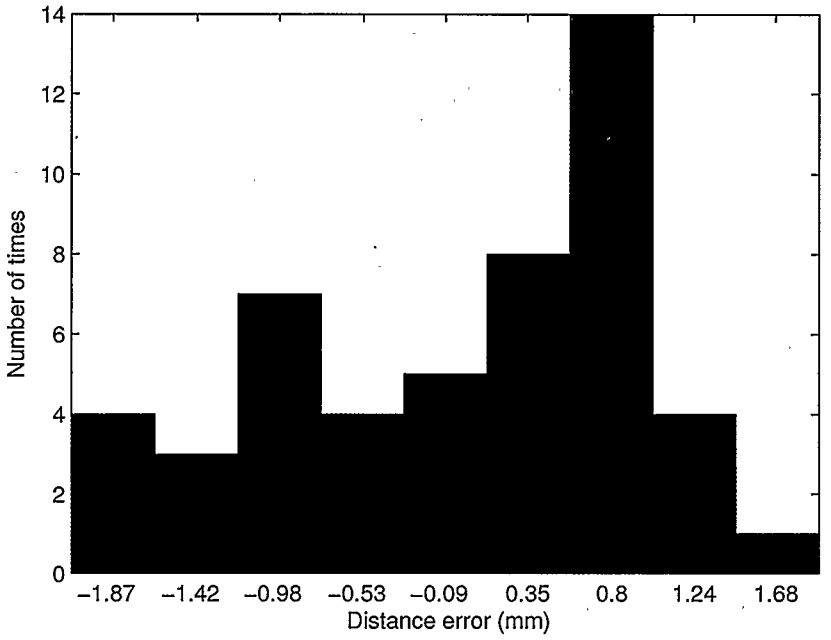

Figure 2.7: Distance DR error distribution for carpet measurements.

larger as the robot moves a longer distance. The distance error shown in Figure 2.7 is relatively minor by comparison.

2.1.4 Angle Dead Reckoning Linoleum Measurements

Similar to Section 2.1.2, angle experiments involve programming the robot to turn at angles of 45^0, 90^0 and 135^0 at a speed of 200 mm/s on linoleum. A protractor is used to measure the actual angle turned by the robot. The number of measurements recorded for each angle is 60, half of which is in the clockwise direction and the other half in the counter clockwise direction. The difference between the desired and the actual angle turned by the robot for each measurement is recorded. Similar to Section 2.1.2 a histogram showing the angle DR error for all three angles is shown in Figure 2.8.

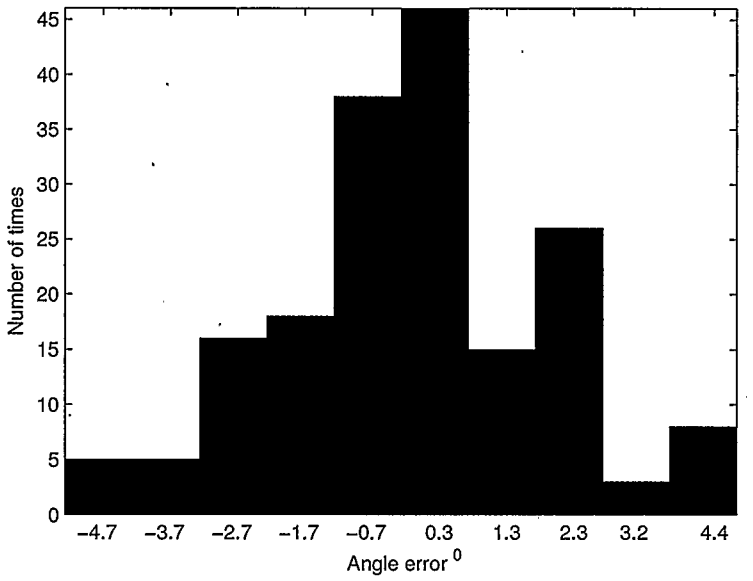

Figure 2.8: Angle DR error distribution for linoleum measurements.

2.1.5 Distance Dead Reckoning Linoleum Measurements

Similar to Section 2.1.3, distance experiments involve programming the robot to move 1 m in a straight line at a speed of 200 mm/s. A total of 50 measurements were recorded. The distance DR error distribution is found by noting the difference between the desired distance and the actual distance moved by the robot. These values are shown in Figure 2.9.

The histograms representing the distance and angle error distributions for carpet and linoleum show that robot platform can be used on different surfaces, and thus the error distribution introduced in Section 2.2 can be generalized to different terrains.

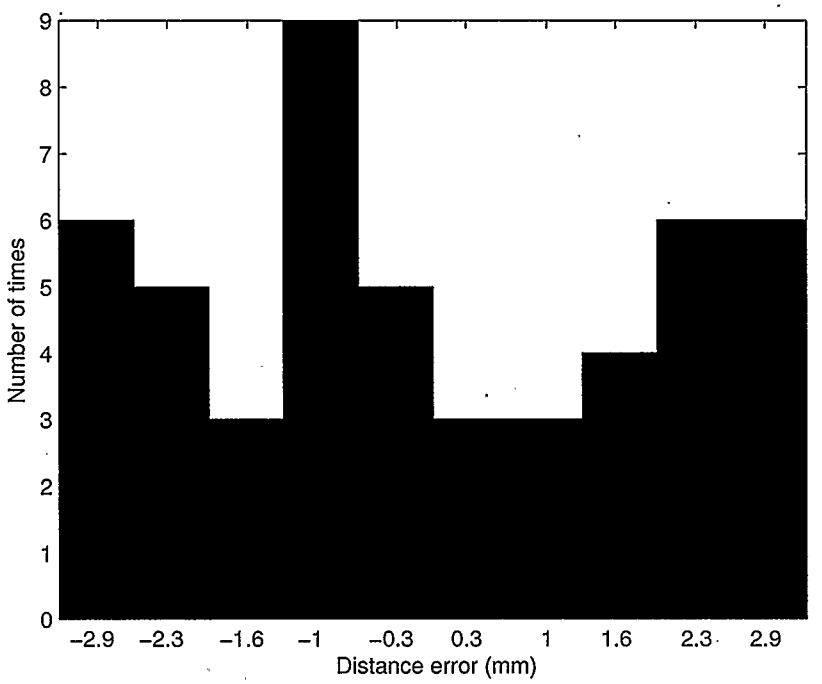

Figure 2.9: Distance DR error distribution for linoleum measurements.

2.2 Gaussian DR Error Model

Based on an assumption of uniformly distributed angle and distance error, a theoretical Gaussian DR error model is derived in Section 2.2.1. This is a position model in the Cartesian plane for a robot moving under the control of the DR-based control system described in Section 2.1. The measurements presented in Section 2.1 are used to illustrate how the Gaussian DR model can be applied to the iRobot platform. However, the Gaussian DR error model derived in Section 2.2.1 can be applied to any platform that rotates in place and then moves in a straight line. Section 2.2.2 verifies the accuracy of the Gaussian DR error model through Monte Carlo simulations that incorporate the empirical angle and distance error distributions for the carpet measurements in Figures 2.6 and 2.7 respectively.

2.2.1 Derivation

Assume the robot path is broken down into N segments, where the i^{th} segment corresponds to a rotation at a specific angle succeeded by motion in a straight line. Every time the robot changes its orientation and moves in a straight line, random error values from the angle and distance error distributions are added that increase the difference between the desired robot position (X_D, Y_D) and its actual position (X, Y). The final position of the robot after traveling the entire path is $X = \sum_{i=1}^{N} X_i + X_0$ and $Y = \sum_{i=1}^{N} Y_i + Y_0$, where (X_0, Y_0) is the initial robot position.

Assuming the angle and distance errors to be uniformly distributed random variables, an expression for the mean and variance of the robot's actual (X, Y) position can be derived as a function of the number of segments traveled. Note that Figures 2.6 and 2.7 show that this uniformly distributed angle and distance error assump-

tion is very approximate. However, it will be shown that this does not significantly affect the final accuracy of the model.

Denote the rotational angle for segment i as a random variable, A_i, such that $A_i \sim U(\alpha_i - \Delta_A, \alpha_i + \Delta_A)$ and denote the distance moved along segment i as a random variable, R_i, such that $R_i \sim U(\beta_i - \Delta_R, \beta_i + \Delta_R)$, where α_i and β_i are the desired angle and distance values, respectively, and Δ_A and Δ_R define uniformly distributed deviation from the desired angle and distance values, respectively.

Note that the random variables, A_i and R_i are assumed to be zero mean. This is a reasonable assumptions since the angle and distance histograms in Figures 2.6 and 2.7 are approximately symmetric. The derivation can be extended to non-zero mean error distributions by assuming a non-symmetric uniform distribution.

Assuming the random variables A_i and R_i to be independent, the joint probability density function $f_{(RA)}(r, a)$ is defined as

$$f_{RA}(r,a) = \begin{cases} \frac{1}{4\Delta_R\Delta_A}, & \beta_i - \Delta_R \leq r \leq \beta_i + \Delta_R, \alpha_i - \Delta_A \leq a \leq \alpha_i + \Delta_A \\ 0, & \text{otherwise} \end{cases} \tag{2.1}$$

From geometry, the Cartesian offset of the robot position after moving along segment i is defined by the random variables $X_i = R_i \cos(A_i)$ and $Y_i = R_i \sin(A_i)$. From [47], the mean of the random variable X_i, $\mu_{X,i}$ is given by

$$\begin{aligned} \mu_{X,i} &= \int_{\alpha_i-\Delta_A}^{\alpha_i+\Delta_A} \int_{\beta_i-\Delta_R}^{\beta_i+\Delta_R} X f_{RA}(r,a) dr da \\ &= \frac{1}{4\Delta_R\Delta_A} \int_{\alpha_i-\Delta_A}^{\alpha_i+\Delta_A} \int_{\beta_i-\Delta_R}^{\beta_i+\Delta_R} r\cos(a) dr da \\ &= \frac{\beta_i}{2\Delta_A} [\sin(\alpha_i + \Delta_A) - \sin(\alpha_i - \Delta_A)] \end{aligned} \tag{2.2}$$

Similarly, the mean of the random variable Y_i, $\mu_{Y,i}$ is given by

$$\begin{aligned}\mu_{Y,i} &= \int_{\alpha_i-\Delta_A}^{\alpha_i+\Delta_A}\int_{\beta_i-\Delta_R}^{\beta_i+\Delta_R} Y f_{RA}(r,a)drda \\ &= \frac{1}{4\Delta_R\Delta_A}\int_{\alpha_i-\Delta_A}^{\alpha_i+\Delta_A}\int_{\beta_i-\Delta_R}^{\beta_i+\Delta_R} r\sin(a)drda \\ &= \frac{\beta_i}{2\Delta_A}\left[\cos(\alpha_i-\Delta_A)-\cos(\alpha_i+\Delta_A)\right]\end{aligned} \tag{2.3}$$

In a similar fashion, $E(X_i^2)$ is given by

$$\begin{aligned}E(X_i^2) &= \int_{\alpha_i-\Delta_A}^{\alpha_i+\Delta_A}\int_{\beta_i-\Delta_R}^{\beta_i+\Delta_R} X^2 f_{RA}(r,a)drda \\ &= \frac{1}{4\Delta_R\Delta_A}\int_{\alpha_i-\Delta_A}^{\alpha_i+\Delta_A}\int_{\beta_i-\Delta_R}^{\beta_i+\Delta_R} r^2\cos^2(a)drda \\ &= \frac{(3\beta_i^2+\Delta_R^2)}{6\Delta_A}\left[\Delta_A+\frac{1}{4}\left(\sin(2\alpha_i+2\Delta_A)-\sin(2\alpha_i-2\Delta_A)\right)\right]\end{aligned} \tag{2.4}$$

and

$$\begin{aligned}E(Y_i^2) &= \int_{\alpha_i-\Delta_A}^{\alpha_i+\Delta_A}\int_{\beta_i-\Delta_R}^{\beta_i+\Delta_R} Y^2 f_{RA}(r,a)drda \\ &= \frac{1}{4\Delta_R\Delta_A}\int_{\alpha_i-\Delta_A}^{\alpha_i+\Delta_A}\int_{\beta_i-\Delta_R}^{\beta_i+\Delta_R} r^2\sin^2(a)drda \\ &= \frac{(3\beta_i^2+\Delta_R^2)}{6\Delta_A}\left[\Delta_A+\frac{1}{4}\left(\sin(2\alpha_i-2\Delta_A)-\sin(2\alpha_i+2\Delta_A)\right)\right]\end{aligned} \tag{2.5}$$

Knowing equations (2.2), (2.3), (2.4) and (2.5) the variances of X_i and Y_i are given by

$$\begin{aligned}\sigma_{X,i}^2 &= E(X_i^2)-\mu_{X,i}^2 \\ \sigma_{Y,i}^2 &= E(Y_i^2)-\mu_{Y,i}^2\end{aligned} \tag{2.6}$$

As the robot moves over a path with a total of N segments, the central limit theorem states that the distribution of the robot's actual (X,Y) position will be Gaussian, such that $X \sim \mathcal{N}(\mu_X,\sigma_X^2)$ and $Y \sim \mathcal{N}(\mu_Y,\sigma_Y^2)$ where the mean of the final robot position

μ_X and μ_Y is given by

$$\begin{aligned} \mu_X &= \sum_{i=1}^{N} \mu_{X,i} \\ \mu_Y &= \sum_{i=1}^{N} \mu_{Y,i} \end{aligned} \tag{2.7}$$

And the variance of the final robot position σ_X^2 and σ_Y^2 is given by

$$\begin{aligned} \sigma_X^2 &= \sum_{i=1}^{N} \sigma_{X,i}^2 \\ \sigma_Y^2 &= \sum_{i=1}^{N} \sigma_{Y,i}^2 \end{aligned} \tag{2.8}$$

In case the robot does not move in straight lines, i.e it moves in a curvature to its destination. Our model still holds if we approximate the robot curvature with small straight line segments.

2.2.2 Verification

The first step in verifying the Gaussian DR model is to confirm that the mean expressions in equation (2.7) and the variance expressions in equation (2.8) are accurate over a single segment (i.e. a single rotation followed by motion along a straight line). This is done by plotting the mean and variance of X_i and Y_i as a function of angle and distance moved and then verifying the curves using the measurements collected on the robot chassis.

The second step is to investigate the accuracy of the Gaussian assumption for variation in the final robot position using Quantile Quantile (QQ) plot analysis. This is done by simulating the motion of the robot along a path where the segments form a staircase pattern. The final histogram of the robot location and a QQ-plot indicating how well

this distribution matches the Gaussian assumption are shown at different stages along the staircase path.

Mean and Variance Analysis

The mean and variance of the robot position is calculated using the analytical expressions for the mean derived in equations (2.2) and (2.3) and variance derived in equations (2.4) and (2.5). The value of Δ_A is chosen such that $f_A(a)$ has the same mean and variance as the empirical histogram in Figure 2.6. Similarly, the value of Δ_R is chosen such that the distribution $f_R(r)$ has the same mean and variance as the empirical histogram found in Figure 2.7. This value of Δ_A corresponds to 3.58^0 and Δ_R to 1.91 mm.

The values for the mean and variance of the robot position are then calculated for robot rotation angles that vary from 0^0 to 90^0 followed by a linear motion of 5, 15 or 20 m.

These analytical mean and variance values are then compared with the mean and variance values generated utilizing Monte Carlo simulations of the robot rotating and then moving over a straight line. The angle and distance errors in this simulation are generated using the empirical distributions for the angle and distance DR error presented in Figures 2.6 and 2.7, respectively.

These empirical error values are calculated in the Monte Carlo simulation using the transformation method [48]. This method allows random numbers to be generated from an empirical distribution. If $y = F(x)$ is the cumulative distribution function (CDF) of the empirical PDF $f(x)$ and the sequence $\{u(1), ..., u(n)\}$ are random numbers from the uniform distribution $U(0, 1)$, then the sequence

$\{F^{-1}(u(1)), F^{-1}(u(2)), ..., F^{-1}(u(n))\}$

represents random samples drawn from the empirical distribution.

For a given desired angle of rotation and distance moved, 10,000 random angle and distance errors are generated using the transformation method. Those values are added to the desired angle and distance and the mean and variance of the robot's final position in the X and Y directions are determined. These are the values that are compared to the analytical expressions for mean and variance.

Figure 2.10 show a plot of the mean X position as a function of the angle turned and the distance moved for a single segment. Similarly, Figure 2.11 show a plot of the mean Y position as a function of the angle turned and the distance moved for a single segment. Since 0^0 corresponds to motion in the positive X direction and 90^0 corresponds to motion in the positive Y direction, the mean X position in Figure 2.10 will decrease as the robot rotates to higher angles, while the mean Y position in Figure 2.11 will increase as the robot rotates to higher angles.

The perfect agreement for the mean distance using the proposed Gaussian DR error approach and the Monte Carlo simulations verifies the zero mean uniform distribution assumption.

Figure 2.12 shows a plot of the variance in the X position as a function of the angle turned and the distance moved for a single segment. Similarly, Figure 2.13 show a plot of the variance in the Y position as a function of the angle turned and the distance moved for a single segment. From the plots, it can be seen that the analytical expressions overestimate the Monte Carlo simulations. The reason is that the angle and distance empirical histograms are not uniformly distributed, while the analytical expressions assume a uniform distribution. The uniform distribution assumption

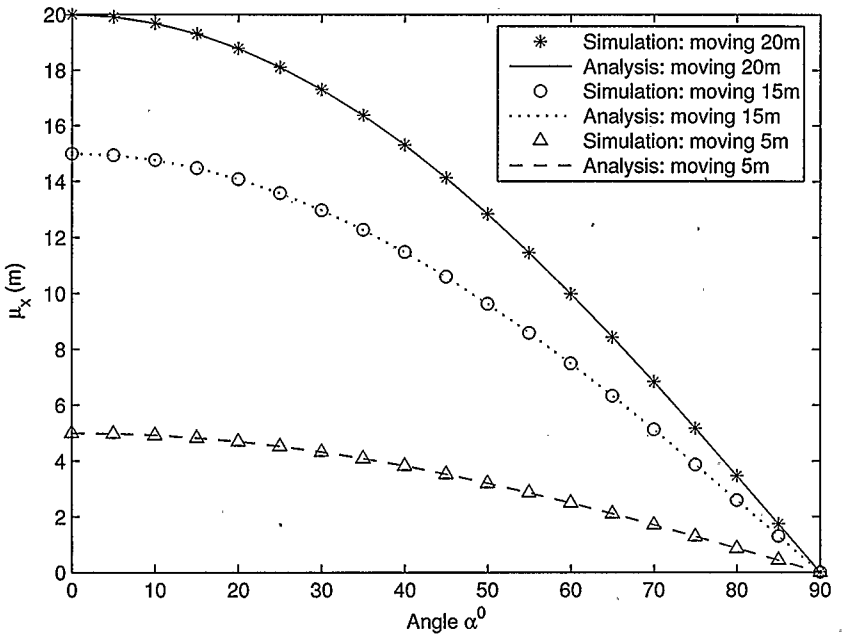

Figure 2.10: Mean actual mobile position in the x-direction for a single path segment.

places more emphasis on the larger angle error values which causes the expressions to overestimate variance. However, the accuracy of the expressions are still reasonable.

Note that when the robot is moving solely in the X direction (0^0) in Figure 2.12, the variance in the X direction is small because the only source of error is the distance error. The variance then increases as the robot turns at larger angles and is proportional to the distance moved. Similar reasoning can be used to explain the behavior of the variance in the Y direction in Figure 2.13.

QQ-plot Analysis

The next step in verifying the Gaussian DR error model is to show whether the variation in the final robot position is well approximated as Gaussian. This is done by performing a goodness-of-fit test. This is a test which shows whether two samples are drawn from

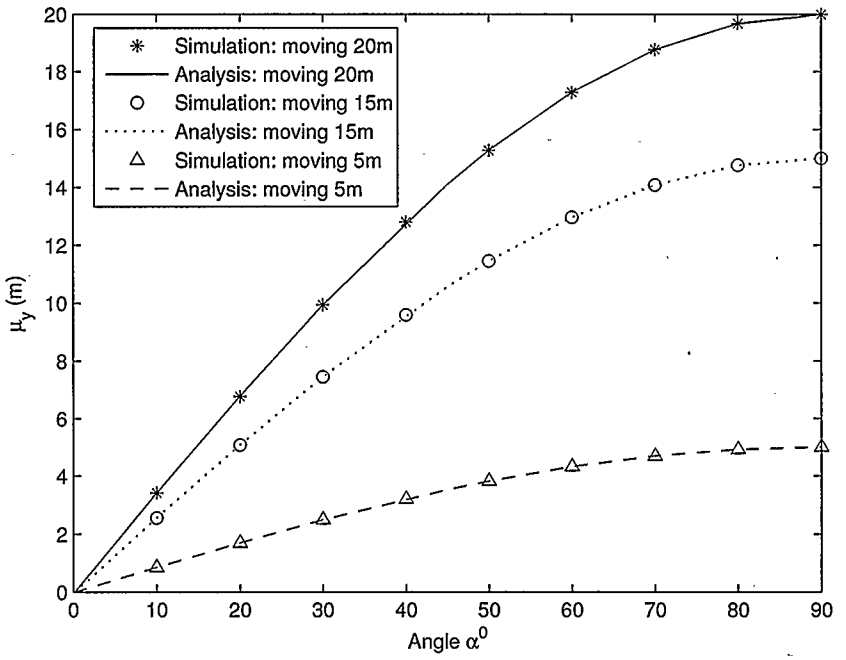

Figure 2.11: Mean actual mobile position in the y-direction for a single path segment.

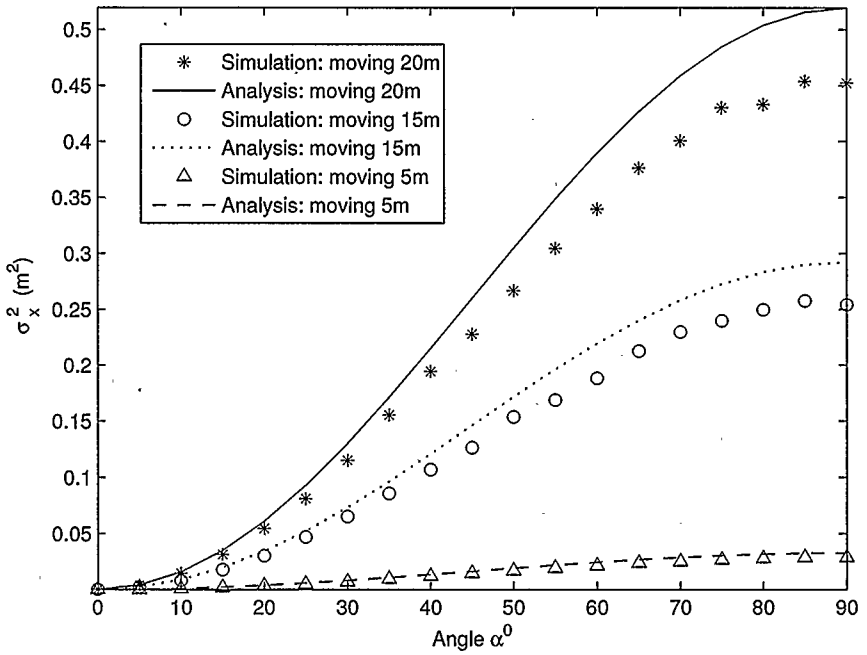

Figure 2.12: Variance of actual mobile position in the X-direction for a single path segment.

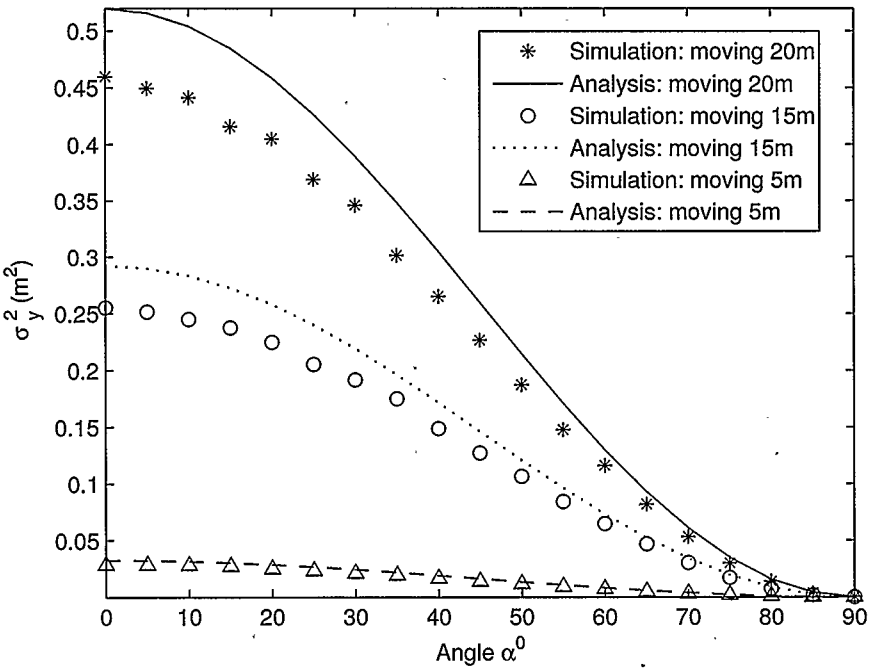

Figure 2.13: Variance of actual mobile position in the Y-direction for a single path segment.

the same distribution.

There exists various goodness-of-fit tests such as the Kolmogorov-Smirnov test and probability plots [47]. A QQ-plot, or quantile quantile plot, is a form of probability plot. It is chosen as the goodness-of-fit test in this thesis because it provides a nice graphical representation of how well the Gaussian distribution represents the actual variation in robot position.

The Gaussian DR model is verified by comparing it to simulated robot positions generated using the empirical distributions in Figures 2.6 and 2.7. This is done by doing a series of Monte Carlo simulations. In each iteration, random values from the empirical DR error distributions are added to the angle and distance moved by the robot using the transformation method. A histogram of the final robot position is then computed and compared with the Gaussian PDF calculated using the analytical mean and variance expressions in equations (2.2), (2.3), (2.4), (2.5) and (2.6).

A QQ-plot shows ordered samples of the actual error values versus samples that are generated using the Gaussian model. The samples generated using the Gaussian model are calculated by finding the mean, μ, and variance, σ^2, of the actual error values. The samples of the Gaussian model are given by [47] $q = \mu + \sqrt{2\sigma^2}\,\mathrm{erfc}(1-2j)$, where erfc is the erf complementary function and j are the percentile values. The percentile values are increasing, evenly spaced values from 0-1 assigned for each ordered sample of the actual error values. If b is a vector from 1 to the size of the sample of the actual error values, then a possible value of j would be
$j = (b - 0.5)/(size\ of\ sample)$. If the empirical data agrees with the Gaussian model, then the points will lie on a straight line with a slope of 1.

From the plot in Figure 2.12 it is evident that the variance in estimating the robot

X position is largest when the angle turned by the robot is 90^0, and from Figure 2.13, the variance in estimating the robot Y position is largest when the angle turned by the robot is 0^0.

As a worst case scenario, the distribution of actual final robot position is determined using Monte Carlo simulation where the robot moves along a staircase path and error is added for each segment using the transformation method.

Assume the robot starts at the origin and is pointing towards the X direction. Let the X axis be the 0^0 reference. The robot starts by turning 90^0 counter-clockwise and then moves 2 m, this is followed by turning 90^0 clockwise and then moving 2 m and so on until it reaches the end of the staircase. Figure 2.14 shows the staircase path. The number of segments in the staircase is varied in order to evaluate the effect of an increasing amount of error.

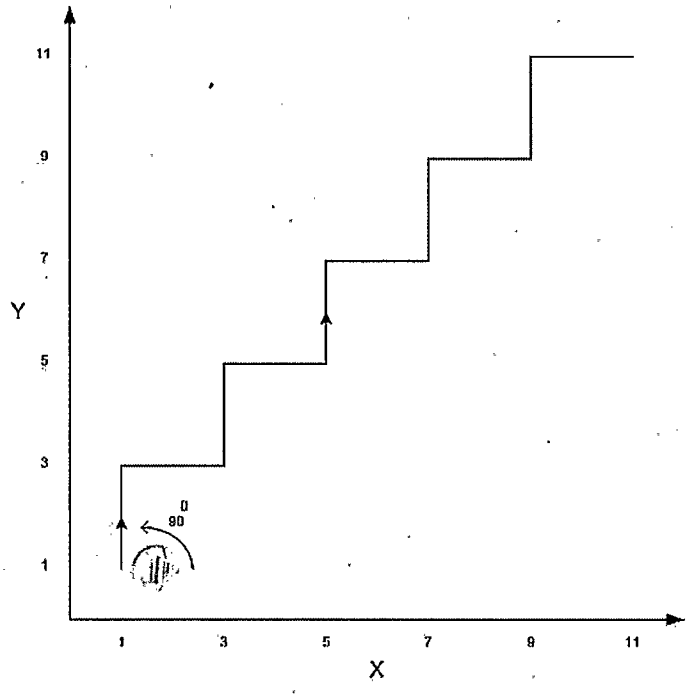

Figure 2.14: Staircase path of the robot.

A histogram of the actual final position of the robot is then calculated. Next the Gaussian approximation is used to approximate the actual final position distribution of the robot. This is done by calculating the mean and the variance expressions in equations (2.2), (2.3), (2.4), (2.5) and (2.6) along each segment of the staircase path. The total mean and variance at the end of the staircase path, of the Gaussian approximation is then calculated by the expressions in equations (2.7) and (2.8).

Figure 2.15a shows the histogram of the actual robot position in the X direction for 1 segment along the staircase, generated using the simulation as well as the Gaussian approximation. The solid line represents the actual distribution and the dashed line represents the Gaussian model. Figure 2.15b shows the QQ-plot that indicates the fit between these distributions. Similarly, the plot for the Y coordinate is shown in Figure 2.16. As expected, these plots show that the Gaussian assumption is a poor approximation after moving only one segment.

Investigating the distribution of the final robot position after 10 steps along the staircase increases the accuracy of the central limit theorem assumption due to the increased number of accumulated angle and distance errors. This improves the fit between the final position histogram and the Gaussian assumption. This is illustrated in Figures 2.17 and 2.18 for the X and Y coordinates respectively.

In conclusion, the Gaussian DR model proposed in this section accurately defines the distribution of the final robot position after going only ten steps. The Gaussian DR model still holds for other paths other than a staircase. This is because the central limit theorem assures that the DR error distribution will eventually become Gaussian irrespective of the path traveled.

To evaluate the accuracy of the central limit theorem (CLT), one could use the Chi-

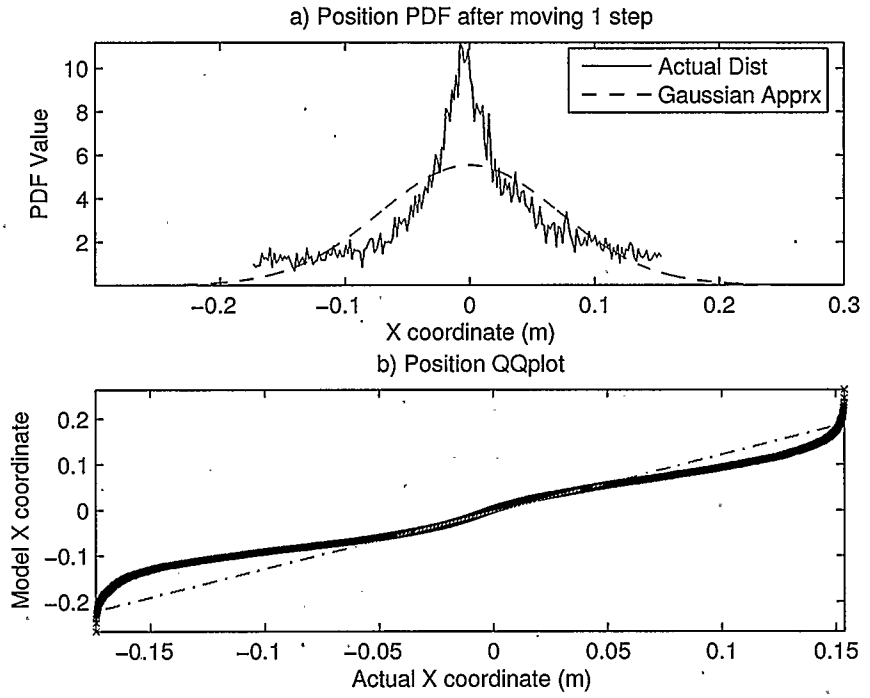

Figure 2.15: Comparison of the actual final DR position distribution to the Gaussian model in the X-direction after going 1 segment.

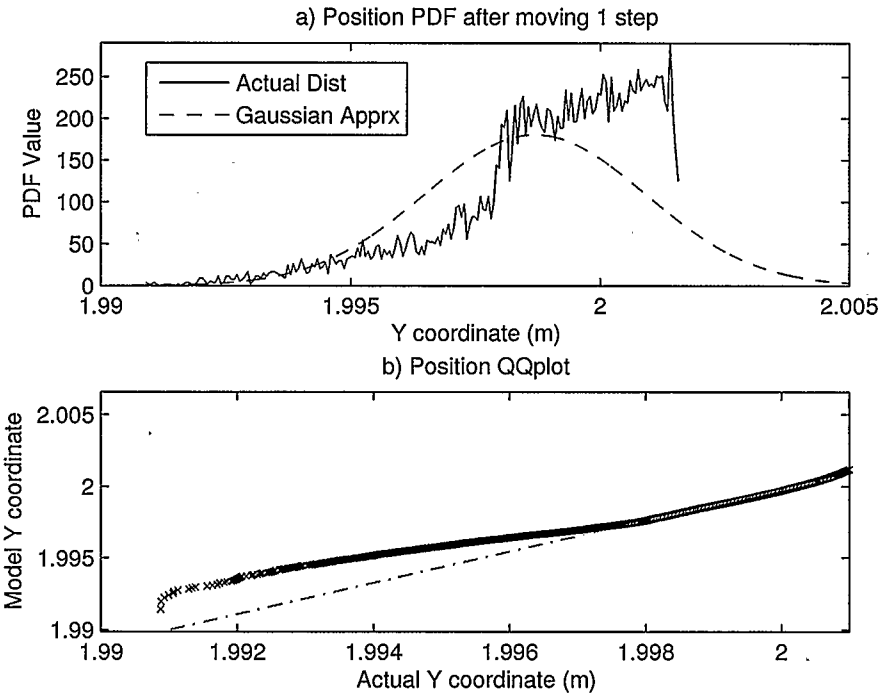

Figure 2.16: Comparison of the actual final DR position distribution to the Gaussian model in the Y-direction after going 1 segment.

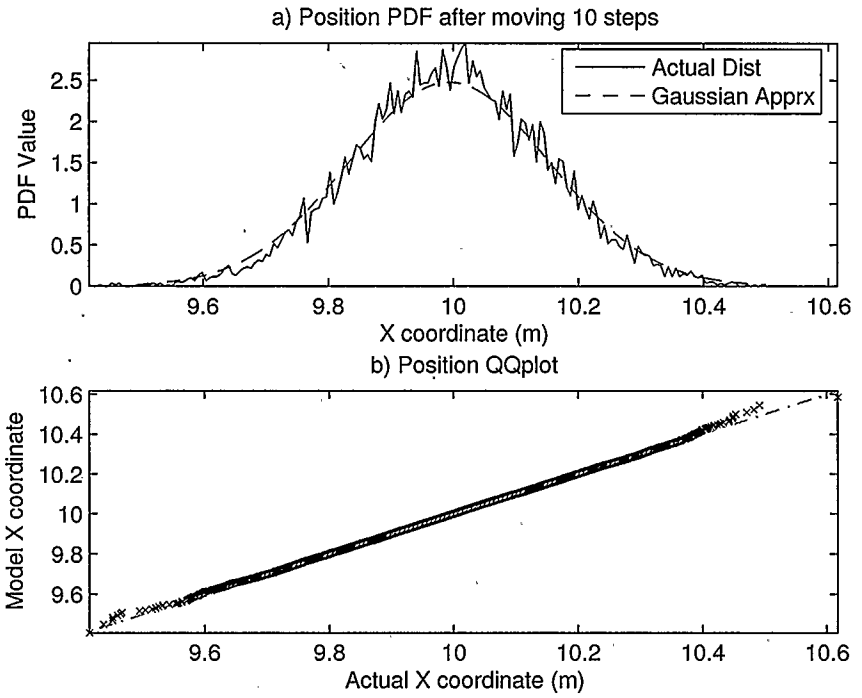

Figure 2.17: Comparison of the actual final DR position distribution to the Gaussian model in the X-direction after going 10 segments.

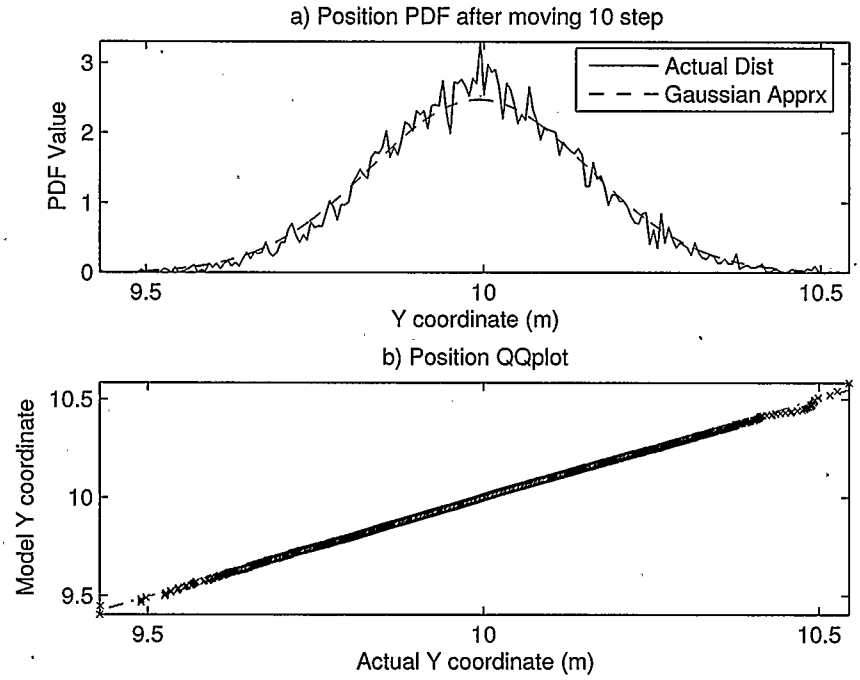

Figure 2.18: Comparison of the actual final DR position distribution to the Gaussian model in the Y-direction after going 10 segments.

square test [47] or the mean squared error between the empirical and Gaussian data to determine the number of segments required before the CLT attains a desired accuracy.

Chapter 3

Cramer Rao Lower Bound Analysis For A DR/RSS System

The first method used to evaluate the performance of DR/RSS localization is the Cramer Rao lower bound (CRLB). The CRLB works by finding a theoretical lower bound on the variance of an unbiased estimator.

The CRLB is commonly applied to static statistical models where the parameter being estimated does not change [49]. The CRLB for dynamic systems where the parameter being estimated is not constant was derived in [50]. An extension to the work in [50] includes a recursive calculation of the CRLB for deterministic [51] and random [52] system models that are used to represent the parameter being estimated.

The difference between the static and the dynamic notion of the CRLB for a hybrid DR/RSS system is presented in Section 3.1, along with the general overview of the problem to be investigated in this thesis. Section 3.2 presents an overview of the assumed uncertainty in the RSS measurements. Section 3.3 derives the static CRLB for the proposed DR/RSS positioning scheme and Section 3.4 discusses the CRLB derivation for the dynamic case. The static and the dynamic CRLB is used in Section 3.5 to evaluate system performance for a number of scenarios. In Chapter 4, the performance of the dynamic CRLB is further compared with a Kalman filter implementation.

3.1 Static And Dynamic CRLB

The CRLB described in Kay's book [49] applies to static statistical models. In this case, an estimate of a parameter or set of parameters is being sought from a set of noisy observations. These observations represent a stationary process where the parameters being estimated and statistics of the noise corrupting those parameters do not change with time. Therefore the CRLB represents estimation for a static system.

Sometimes the parameter being estimated changes with time. In the case where a model exists that describes these time variations, a CRLB can be determined. The main characteristic of the dynamic CRLB is that it has memory. The bound on the current estimate assumes the estimator has the benefit of knowledge of previous estimations.

In this thesis, a robot starting from known Cartesian coordinates is programmed to go over a path. As described in Chapter 2 the DR position estimate represents the desired position programmed into the robot control algorithm. The DR error due to the inaccuracies in the angle turned and distance moved will cause the desired position to be different from the actual robot position. A large building indoor scenario is assumed, where the robot path is constrained within a 100 m by 100 m area. Figure 3.1 shows an illustration of the area and one possible beacon configuration. The black dots in the figure represent the beacons which are wireless devices with fixed positions. These beacons could represent wireless access points in a conventional WiFi network. An estimator is then used to combine the DR position estimates along with the RSS information from the beacons to estimate the robot position.

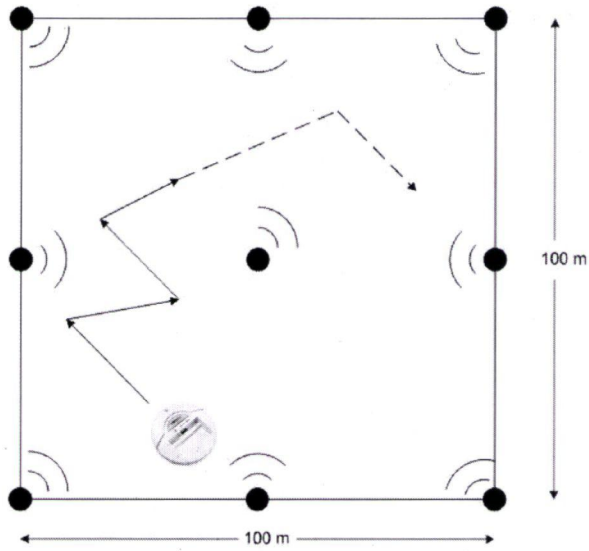

Figure 3.1: Area configuration.

Note that the static and dynamic CRLB systems represent two separate position estimation techniques. In this thesis, a static system describes the case where a robot estimates its position based only on DR information and RSS measurements available at a specific time instant. The robot could then move to a different position, and re-estimate its position after it reaches its new location. In this static scenario, the estimator does not use knowledge of previous robot positions when estimating its current position. Thus the static CRLB can be considered as isolated snap shots of the robot position estimate. This is called a static scenario since the robot position can be considered constant at estimation time.

In a dynamic system, the robot motion can be represented using a motion model. The robot estimates its position at specific time intervals along its path utilizing the DR and RSS information available at estimation time plus knowledge of previous positions. Therefore we would expect that the dynamic CRLB to have a higher accuracy in estimating the robot position compared to the static CRLB.

3.2 Radio Channel

Consider a robot with unknown position (X, Y) whose location is being estimated. There are m reference beacons with fixed positions $(X_{b,k}, Y_{b,k})$, $k = 1, 2, ..., m$ distributed in a square area of length 100 m $\times$ 100 m. The robot is able to measure the average received signal strength (RSS) of these beacon signals. Our goal is to combine the DR position estimate, which was found from Section 2.2 to have a Gaussian distribution, with the RSS positioning measurements to determine the accuracy of hybrid DR/RSS localization for indoor scenarios.

In order to understand the uncertainty in RSS measurements, it is necessary to understand the characteristics of the wireless channel. The Path loss, $PL(d)$, is a deterministic quantity that represents average signal attenuation as a function of the transmitter receiver separation, d. The path loss value expressed in dB is given by, $PL(d)_{dB} = PL(d_0)_{dB} + 10\eta \log(\frac{d}{d_0})$ where d_0 is a reference distance and η is the path loss exponent. Knowing the path loss is an advantage since the RSS range measurements rely on the above path loss equation to determine the distance d between the transmitter and receiver.

The value of the path loss exponent, η, depends on the environment in which the transmitter and receiver exist. In a factory or a building with a non-line of sight between the transmitter and receiver, typical values of η range between 3 and 6 [45].

RSS measurements are affected by two major sources of environment dependent errors, namely small scale fading and large scale fading. Both can potentially serve to corrupt the RSS range estimate.

Small scale fading or multipath fading happens when multiple signals arrive at the

receiver with different amplitudes and phases. These signals add constructively or destructively causing rapid variations in the signal amplitudes over small distances, typically $\lambda/4$ and λ is the wavelength of the signal. Small scale fading effects can be reduced by averaging the received signal power over a wide range of frequencies or by calculating a running average of the signal with respect to time. As a result, small scale fading is not accounted for in this thesis.

Large scale fading or shadowing is a random quantity that represents the attenuation of the signal power due to objects coming between the transmitter and receiver. Large scale fading is usually constant over at least a few meters [45]. For both indoor and outdoor channels, theoretical and empirical evidence suggest that the received power, in dB, can be written as [45]

$$R = 10\eta \log(\frac{d}{d_0}) + Z \tag{3.1}$$

where Z is Gaussian, with zero mean and variance σ_R^2. In [53], the shadowing standard deviation was measured in two different indoor locations. One location was in an office building with long narrow corridors, while the other location was in a large open area of dimension 15 m $\times$ 30 m $\times$ 8 m. Their results showed that the shadowing standard deviation was in the range between 1.8 - 2.8 dB.

Assuming a network of m beacons, let d_0 be at a distance of 1 m, the received power, R_k, from beacon k, with coordinates $(X_{b,k}, Y_{b,k})$ is

$$R_k = 10\eta \log(d_k) + Z \qquad k = 1, 2, ...m \tag{3.2}$$

where $d_k = \sqrt{(X - X_{b,k})^2 + (Y - Y_{b,k})^2}$

3.3 CRLB Derivation For A Static Network

Let the vector $\boldsymbol{\theta} = [X \;\; Y]$ denote a vector of the parameters to be estimated, where X and Y are the X and Y robot positions respectively to be estimated. The measurement or observation vector consists of the DR X position, X_{DR}, the DR Y position, Y_{DR}, and the RSS measurements, R_k, such that $X_{DR} \sim \mathcal{N}(X, \sigma_X^2[n])$, $Y_{DR} \sim \mathcal{N}(Y, \sigma_Y^2[n])$, and $R_k \sim \mathcal{N}(10\eta \log(d_k), \sigma_R^2) \;\; k = 1, 2, ...m$.

Let $\mathbf{z}[n]$ denote the observation vector such that

$$\mathbf{z}[n] = [\; X_{DR}[n] \; Y_{DR}[n] \; R_1[n] \; ... \; R_m[n]]^T_{(m+2)\times 1} \tag{3.3}$$

The mean of the observation vector, $\boldsymbol{\mu}(\boldsymbol{\theta})$ is given by

$$\boldsymbol{\mu}(\boldsymbol{\theta}) = [X \; Y \; 10\eta \log(d_1) \; ... \; 10\eta \log(d_m)]^T_{(m+2)\times 1} \tag{3.4}$$

and the covariance matrix of the observation noise, $\mathbf{C}(\boldsymbol{\theta})$, is given by

$$\mathbf{C}(\boldsymbol{\theta}) = \begin{bmatrix} \sigma_X^2[n] & 0 & 0 & 0 & \dots & 0 \\ 0 & \sigma_Y^2[n] & 0 & 0 & \ddots & 0 \\ 0 & 0 & \sigma_R^2 & 0 & \dots & 0 \\ 0 & 0 & 0 & \ddots & \dots & \vdots \\ \vdots & \vdots & \vdots & \ddots & \ddots & 0 \\ 0 & 0 & 0 & \dots & 0 & \sigma_R^2 \end{bmatrix}_{(m+2)\times(m+2)} \tag{3.5}$$

where σ_R^2 represents the shadowing variance. Assuming the beacons are well scattered across the 100 m by 100 m area, as shown in Figure 3.1, the shadowing variances are uncorrelated. The values of σ_X^2 and σ_Y^2 represent the DR error variances.

In general, the DR error variances are a function of time n, since it was established in Chapter 2 that DR variance increases with distance travelled. However, the static CRLB derivation assumes that estimation is performed using only information available at a single timestep. As a result, the DR error variances $\sigma_X^2[n]$ and $\sigma_Y^2[n]$ are assumed to be constant. The DR error variances σ_X^2 and σ_Y^2 are calculated using equation (2.6) in Chapter 2.

Since the DR error variances $\sigma_X^2[n]$ and $\sigma_Y^2[n]$, and the shadowing variances σ_R^2 from the m beacons are uncorrelated, the covariance matrix $\mathbf{C}(\boldsymbol{\theta})$ is a diagonal matrix as shown in equation (3.5). However if the shadow and the DR error values are correlated there would be non-diagonal elements signifying the correlation between the parameters.

The CRLB sets a lower bound on the covariance of any unbiased estimator. Let the vector $\widehat{\boldsymbol{\theta}}$ represent an estimate of the robot Cartesian position $\boldsymbol{\theta}$, such that $\widehat{\boldsymbol{\theta}} = [\widehat{X} \;\; \widehat{Y}]$. By definition, from [49], the CRLB of the estimation of the robot's position $\widehat{\boldsymbol{\theta}}$ is given by

$$cov\left\{\widehat{\boldsymbol{\theta}}\right\} \geq \mathbf{I}^{-1}(\boldsymbol{\theta}) \tag{3.6}$$

where $cov(\widehat{\boldsymbol{\theta}})$ represents the covariance of the error estimate, such that $cov\left\{\widehat{\boldsymbol{\theta}}\right\} = E\left\{(\boldsymbol{\theta} - \widehat{\boldsymbol{\theta}})(\boldsymbol{\theta} - \widehat{\boldsymbol{\theta}})^T\right\}$ and $\mathbf{I}(\boldsymbol{\theta})$ is the Fisher information matrix (FIM).

Since we are trying to estimate the vector $\boldsymbol{\theta} = [X \;\; Y]$ the matrix $\mathbf{I}(\boldsymbol{\theta})$ is 2×2 and has the general form

$$\mathbf{I}(\boldsymbol{\theta}) = \begin{bmatrix} [I(\theta)]_{11} & [I(\theta)]_{12} \\ [I(\theta)]_{21} & [I(\theta)]_{22} \end{bmatrix} \tag{3.7}$$

Since the probability density functions (PDFs) representing the DR and RSS mea-

surements are Gaussian, the FIM components are given by [49]

$$[I(\theta)]_{ij} = \left[\frac{\partial \boldsymbol{\mu}(\boldsymbol{\theta})}{\partial(\theta)_i}\right]^T \mathbf{C}^{-1}(\boldsymbol{\theta}) \left[\frac{\partial \boldsymbol{\mu}(\boldsymbol{\theta})}{\partial(\theta)_j}\right] + \frac{1}{2} tr\left[\mathbf{C}^{-1}(\boldsymbol{\theta}) \frac{\partial \mathbf{C}(\boldsymbol{\theta})}{\partial(\theta)_i} \mathbf{C}^{-1}(\boldsymbol{\theta}) \frac{\partial \mathbf{C}(\boldsymbol{\theta})}{\partial(\theta)_j}\right] \tag{3.8}$$

The inverse covariance matrix of the observation noise, $\mathbf{C}^{-1}(\boldsymbol{\theta})$ is given by

$$\mathbf{C}^{-1}(\boldsymbol{\theta}) = \begin{bmatrix} \frac{1}{\sigma_X^2} & 0 & 0 & 0 & \dots & 0 \\ 0 & \frac{1}{\sigma_Y^2} & 0 & 0 & \dots & 0 \\ 0 & 0 & \frac{1}{\sigma_R^2} & 0 & \dots & 0 \\ 0 & 0 & 0 & \ddots & \dots & \vdots \\ \vdots & \vdots & \vdots & \ddots & \ddots & 0 \\ 0 & 0 & 0 & \dots & 0 & \frac{1}{\sigma_R^2} \end{bmatrix}_{(m+2)\times(m+2)} \tag{3.9}$$

Since $\mathbf{C}(\boldsymbol{\theta})$ is assumed constant, the values of $\frac{\partial \mathbf{C}(\boldsymbol{\theta})}{\partial(\theta)_i}$ and $\frac{\partial \mathbf{C}(\boldsymbol{\theta})}{\partial(\theta)_j}$ in the second term of equation (3.8) will go to zero. Given equation (3.4), evaluating the expressions $\frac{\partial \boldsymbol{\mu}(\boldsymbol{\theta})}{\partial(\theta)_1}$ and $\frac{\partial \boldsymbol{\mu}(\boldsymbol{\theta})}{\partial(\theta)_2}$ yields

$$\frac{\partial \boldsymbol{\mu}(\boldsymbol{\theta})}{\partial(\theta)_1} \equiv \frac{\partial \boldsymbol{\mu}(\boldsymbol{\theta})}{\partial X} = \left[1 \;\; 0 \;\; \frac{-10\eta(X - X_{b,1})}{\ln 10 \; d_1^2} \;\; \frac{-10\eta(X - X_{b,2})}{\ln 10 \; d_2^2} \;\; \dots \;\; \frac{-10\eta(X - X_{b,m})}{\ln 10 \; d_m^2}\right]^T_{(m+2)\times 1} \tag{3.10}$$

$$\frac{\partial \boldsymbol{\mu}(\boldsymbol{\theta})}{\partial(\theta)_2} \equiv \frac{\partial \boldsymbol{\mu}(\boldsymbol{\theta})}{\partial Y} = \left[0 \;\; 1 \;\; \frac{-10\eta(Y - Y_{b,1})}{\ln 10 \; d_1^2} \;\; \frac{-10\eta(Y - Y_{b,2})}{\ln 10 \; d_2^2} \;\; \dots \;\; \frac{-10\eta(Y - Y_{b,m})}{\ln 10 \; d_m^2}\right]^T_{(m+2)\times 1} \tag{3.11}$$

Substituting equations (3.9), (3.10) and (3.11) into equation (3.8), results in

$$[I(\theta)]_{11} = \frac{1}{\sigma_X^2} + \left(\frac{10\eta}{\ln 10}\right)^2 \frac{1}{\sigma_R^2} \sum_{k=1}^{m} \frac{(X - X_{b,k})^2}{d_k^4} \tag{3.12}$$

$$[I(\theta)]_{12} = \left(\frac{10\eta}{\ln 10}\right)^2 \frac{1}{\sigma_R^2} \sum_{k=1}^{m} \frac{(X - X_{b,k})(Y - Y_{b,k})}{d_k^4} \tag{3.13}$$

$$[I(\theta)]_{21} = \left(\frac{10\eta}{\ln 10}\right)^2 \frac{1}{\sigma_R^2} \sum_{k=1}^{m} \frac{(X - X_{b,k})(Y - Y_{b,k})}{d_k^4} \tag{3.14}$$

$$[I(\theta)]_{22} = \frac{1}{\sigma_Y^2} + \left(\frac{10\eta}{\ln 10}\right)^2 \frac{1}{\sigma_R^2} \sum_{k=1}^{m} \frac{(Y - Y_{b,k})^2}{d_k^4} \tag{3.15}$$

Note that $[I(\theta)]_{12}$ is equal to $[I(\theta)]_{21}$. After deriving an expression for the elements of matrix $\mathbf{I}(\boldsymbol{\theta})$ in equation (3.7), the CRLB of the estimation of σ_X^2 and σ_Y^2 is given respectively by $\sigma_X^2 \geq [I^{-1}(\theta)]_{11}$ and $\sigma_Y^2 \geq [I^{-1}(\theta)]_{22}$. The determinant of matrix $\mathbf{I}(\boldsymbol{\theta})$ is given by

$$\begin{aligned}
\det(\mathbf{I}(\boldsymbol{\theta})) &= [I(\theta)]_{11}[I(\theta)]_{22} - [I(\theta)]_{12}[I(\theta)]_{21} \\
&= \frac{1}{\sigma_X^2\sigma_Y^2} + \frac{1}{\sigma_X^2}\left(\frac{10\eta}{\ln 10}\right)^2 \frac{1}{\sigma_R^2} \sum_{k=1}^{m} \frac{(Y - Y_{b,k})^2}{d_k^4} \\
&+ \frac{1}{\sigma_Y^2}\left(\frac{10\eta}{\ln 10}\right)^2 \frac{1}{\sigma_R^2} \sum_{k=1}^{m} \frac{(X - X_{b,k})^2}{d_k^4} \\
&+ \left(\frac{10\eta}{\ln 10}\right)^4 \frac{1}{\sigma_R^4} \sum_{k=1}^{m} \frac{(X - X_{b,k})^2}{d_k^4} \sum_{k=1}^{m} \frac{(Y - Y_{b,k})^2}{d_k^4} \\
&- \left(\left(\frac{10\eta}{\ln 10}\right)^2 \frac{1}{\sigma_R^2} \sum_{k=1}^{m} \frac{(X - X_{b,k})(Y - Y_{b,k})}{d_k^4}\right)^2
\end{aligned} \tag{3.16}$$

Therefore σ_X^2 and σ_Y^2 are given by

$$\begin{aligned}
\sigma_X^2 &\geq \left[I^{-1}(\theta)\right]_{11} \\
&\geq \frac{[I(\theta)]_{22}}{\det(\mathbf{I}(\boldsymbol{\theta}))} \\
&\geq \frac{\frac{1}{\sigma_Y^2} + \left(\frac{10\eta}{\ln 10}\right)^2 \frac{1}{\sigma_R^2} \sum_{k=1}^{m} \frac{(Y - Y_{b,k})^2}{d_k^4}}{\det(\mathbf{I}(\boldsymbol{\theta}))}
\end{aligned} \tag{3.17}$$

$$\begin{aligned}\sigma_Y^2 &\geq \left[I^{-1}(\theta)\right]_{22} \\ &\geq \frac{[I(\theta)]_{11}}{\det(\mathbf{I}(\boldsymbol{\theta}))} \\ &\geq \frac{\frac{1}{\sigma_X^2} + \left(\frac{10\eta}{\ln 10}\right)^2 \frac{1}{\sigma_R^2} \sum_{k=1}^{m} \frac{(X-X_{b,k})^2}{d_k^4}}{\det(\mathbf{I}(\boldsymbol{\theta}))}\end{aligned} \tag{3.18}$$

3.4 CRLB Derivation For Dynamic Systems

Consider a system with a linear state model

$$\mathbf{s}[n+1] = \mathbf{A}[n]\mathbf{s}[n] + \mathbf{u}[n] \tag{3.19}$$

where $\mathbf{s}[n]$ is an $r \times 1$ state vector, $\mathbf{A}[n]$ is a matrix of dimension $r \times r$ defining the robot kinematics, and $\mathbf{u}[n]$ is the process noise. It is assumed to be zero mean white Gaussian noise process with covariance matrix $\mathbf{Q}[n]$. It is assumed the estimator observations can be represented with the following equation

$$\mathbf{z}[n] = \mathbf{H}[n]\mathbf{s}[n] + \mathbf{w}[n] \tag{3.20}$$

where $\mathbf{z}[n]$ is the $p \times 1$ observation vector, $\mathbf{H}[n]$ is the observation matrix, and $\mathbf{w}[n]$ is the observation noise. It is assumed to be zero mean white Gaussian noise process with covariance matrix $\mathbf{C}[n]$.

Dynamic systems are divided into stochastic models and deterministic models. Stochastic models are characterized with a process noise, $\mathbf{u}[n]$, in the state model equation (3.19). On the other hand, deterministic models assume no process noise and thus no randomness in the state equations. In this thesis, a constant velocity motion model is assumed where the robot is allowed to change direction. In order to do that, the process noise $\mathbf{u}[n]$ must be present to model this velocity change, hence our system model is stochastic.

The CRLB derivation for stochastic system models is provided in Section 3.4.1, while the state and observation model initialization is discussed in Section 3.4.2.

3.4.1 Stochastic System Model

This section presents a recursive CRLB for stochastic system models that was illustrated in [52]. The FIM for estimating the parameters of the state vector $\mathbf{s}[n]$ given the observation vector, $\mathbf{z}[n]$ at a specific time, n was given by equation (3.8). In equation (3.8), $\boldsymbol{\theta}$ denotes the state vector $\mathbf{s}[n]$, and $\boldsymbol{\mu}(\boldsymbol{\theta})$ denotes the mean of the observation vector $\mathbf{z}[n]$. This equation describes the static CRLB case when the observations are corrupted by Gaussian noise. A more general definition of the static CRLB is [49]

$$\mathbf{I}(\mathbf{s}[n]) = -E\left\{\nabla_{\mathbf{s}[n]}\left(\nabla_{\mathbf{s}[n]}\ln\mathcal{P}(\mathbf{z}[n]|\mathbf{s}[n])\right)^T\right\} \tag{3.21}$$

where $\nabla_{\mathbf{s}[n]}$ is the first-order partial derivative operator with respect to the state vector $\mathbf{s}[n]$. The term $\ln\mathcal{P}(\mathbf{z}[n]|\mathbf{s}[n])$ is known as the likelihood function. It describes the PDF of the measurement, $\mathbf{z}[n]$, conditioned on the parameter to be estimated, $\mathbf{s}[n]$. The dimension of the FIM, $\mathbf{I}(\mathbf{s}[n])$, is $r \times r$, where r is the size of the state vector $\mathbf{s}[n]$ to be estimated.

Let the sequence $\mathbf{S}[n]$ represent the trajectory of the robot motion, such that $\mathbf{S}[n] = \{\mathbf{s}[1], \mathbf{s}[2], ..., \mathbf{s}[n]\}$, where $\mathbf{s}[n]$ is defined in equation (3.19).
Let $\mathbf{Z}[n] = \{\mathbf{z}[1], \mathbf{z}[2], ..., \mathbf{z}[n]\}$ represent the sequence of observations, where $\mathbf{z}[n]$ is defined in equation (3.20). The FIM for estimating the sequence of states $\mathbf{S}[n]$ is given by [50] as

$$\mathbf{I}(\mathbf{S}[n]) = -E\left\{\nabla_{\mathbf{S}[n]}\left(\nabla_{\mathbf{S}[n]}\ln\mathcal{P}(\mathbf{S}[n], \mathbf{Z}[n])\right)^T\right\} \tag{3.22}$$

and the dimension of $\mathbf{I}(\mathbf{S}[n])$ is $nr \times nr$. Note that the expectation is with respect to

both $\mathbf{S}[n]$ and $\mathbf{Z}[n]$, since both are stochastic.

The goal is to estimate the parameters of the state vector $\mathbf{s}[n]$, of dimension $r \times 1$, given the earlier observations $\mathbf{Z}[n-1]$, of dimension $(n-1)p \times 1$, and earlier states $\mathbf{S}[n-1]$, of dimension $(n-1)r \times 1$. The sequence of states $\mathbf{S}[n]$ can be decomposed into $\mathbf{S}[n] = \left[\mathbf{S}[n-1]^T \;\; \mathbf{s}[n]^T\right]^T_{nr\times 1}$. Since $\mathbf{S}[n]$ is decomposed into 2 vectors: $\mathbf{S}[n-1]$ and $\mathbf{s}[n]$, then the FIM, $\mathbf{I}(\mathbf{S}[n])$ is expressed in matrix form as:

$$\mathbf{I}(\mathbf{S}[n]) = \begin{bmatrix} \nu_n & \xi_n \\ \xi_n^T & \gamma_n \end{bmatrix}_{nr\times nr} \tag{3.23}$$

where ν_n, ξ_n and γ_n represent submatrix blocks and are defined as

$$\begin{aligned}
\nu_n &= -E\left\{\nabla_{\mathbf{S}[n-1]}\left(\nabla_{\mathbf{S}[n-1]}\ln\mathcal{P}(\mathbf{S}[n],\mathbf{Z}[n])\right)^T\right\}_{(n-1)r\times(n-1)r} \\
\xi_n &= -E\left\{\nabla_{\mathbf{S}[n-1]}\left(\nabla_{\mathbf{s}[n]}\ln\mathcal{P}(\mathbf{S}[n],\mathbf{Z}[n])\right)^T\right\}_{(n-1)r\times r} \\
\xi_n^T &= -E\left\{\nabla_{\mathbf{s}[n]}\left(\nabla_{\mathbf{S}[n-1]}\ln\mathcal{P}(\mathbf{S}[n],\mathbf{Z}[n])\right)^T\right\}_{r\times(n-1)r} \\
\gamma_n &= -E\left\{\nabla_{\mathbf{s}[n]}\left(\nabla_{\mathbf{s}[n]}\ln\mathcal{P}(\mathbf{S}[n],\mathbf{Z}[n])\right)^T\right\}_{r\times r}
\end{aligned} \tag{3.24}$$

The value ν_n is the FIM for estimating $\mathbf{S}[n-1]$, γ_n is the FIM for estimating $\mathbf{s}[n]$, and ξ_n and ξ_n^T are the FIM showing the correlation between $\mathbf{S}[n-1]$ and $\mathbf{s}[n]$.

Assume n is equal to 2, such that $\mathbf{S}[2] = \left[\mathbf{s}[1]^T \;\; \mathbf{s}[2]^T\right]^T_{2r\times 1}$, the CRLB on the estimation of $\mathbf{s}[2]$ is defined as the inverse of the FIM, $\mathbf{I}(\mathbf{S}[2])^{-1}$, such that [49]

$$\mathbf{I}(\mathbf{S}[2])^{-1} = \frac{1}{det(\mathbf{I}(\mathbf{S}[2]))}\begin{bmatrix} \gamma_2 & -\xi_2 \\ -\xi_2^T & \nu_2 \end{bmatrix} \tag{3.25}$$

and since the $det(\mathbf{I}(\mathbf{S}[2]))$ is equal to $\gamma_2\nu_2 - \xi_2^T\xi_2$, therefore, the CRLB or the inverse of the FIM, $\mathbf{I}(\mathbf{S}[2])^{-1}$ becomes

$$\mathbf{I}(\mathbf{S}[2])^{-1} = \frac{1}{\gamma_2\nu_2 - \xi_2^T\xi_2}\begin{bmatrix} \gamma_2 & -\xi_2 \\ -\xi_2^T & \nu_2 \end{bmatrix} \tag{3.26}$$

Since the state vector, $\mathbf{s}[2]$ represents the last element in vector $\mathbf{S}[2]$, therefore the CRLB on the estimation of the state vector $\mathbf{s}[2]$ is the inverse of the right lower submatrix of the FIM $\mathbf{I}(\mathbf{S}[2])$ such that

$$\mathbf{I}(\mathbf{s}[2])^{-1} = \nu_2(\gamma_2\nu_2 - \xi_2^T\xi_2)^{-1} \tag{3.27}$$

The FIM, $\mathbf{I}(\mathbf{s}[2])$ is simply the inverse of equation (3.27), such that

$$\begin{aligned}\mathbf{I}(\mathbf{s}[2]) &= \nu_2^{-1}(\nu_2\gamma_2 - \xi_2^T\xi_2) \\ &= \gamma_2 - \xi_2^T\nu_2^{-1}\xi_2\end{aligned} \tag{3.28}$$

Extending the result in equation (3.28) when the number of states n is greater than 2 the FIM, $\mathbf{I}(\mathbf{s}[n])$ becomes

$$\mathbf{I}(\mathbf{s}[n]) = \gamma_n - \xi_n^T\nu_n^{-1}\xi_n \tag{3.29}$$

In [52], a recursive form for the FIM is found. This form is more efficient than finding the FIM using equation (3.29) since we do not have to compute the inverse of the large matrix ν_n which has dimensions $(n-1)r \times (n-1)r$ as shown in equation (3.24). The recursive FIM equation is given by [52]

$$\mathbf{I}(\mathbf{s}[n+1]) = D_n^{22} - D_n^{21}(\mathbf{I}(\mathbf{s}[n]) + D_n^{11})^{-1}D_n^{12} \quad n \geq 0 \tag{3.30}$$

where

$$\begin{aligned}D_n^{11} &= -E\left\{\nabla_{\mathbf{s}[n]}\left(\nabla_{\mathbf{s}[n]}\ln\mathcal{P}(\mathbf{s}[n+1]|\mathbf{s}[n])\right)^T\right\} \\ D_n^{21} &= -E\left\{\nabla_{\mathbf{s}[n]}\left(\nabla_{\mathbf{s}[n+1]}\ln\mathcal{P}(\mathbf{s}[n+1]|\mathbf{s}[n])\right)^T\right\} \\ D_n^{12} &= [D_n^{21}]^T \\ D_n^{22} &= -E\left\{\nabla_{\mathbf{s}[n+1]}\left(\nabla_{\mathbf{s}[n+1]}\ln\mathcal{P}(\mathbf{s}[n+1]|\mathbf{s}[n])\right)^T\right\} \\ &\quad - E\left\{\nabla_{\mathbf{s}[n+1]}\left(\nabla_{\mathbf{s}[n+1]}\ln\mathcal{P}(\mathbf{z}[n+1]|\mathbf{s}[n+1])\right)^T\right\}\end{aligned} \tag{3.31}$$

The derivation of equation (3.30) is shown in Appendix A.1.

The coefficients D_n^{11}, D_n^{12}, D_n^{21} and D_n^{22} defined in equations (3.31) are simplified if the system state model and observation model are linear as in equations (3.19) and (3.20), and the noise is Gaussian. Recall from equation (3.19) that the system process noise, $\mathbf{u}[n]$ is a zero mean white Gaussian noise process with covariance matrix $\mathbf{Q}[n]$, and from equation (3.20) that the observation noise is a zero mean white Gaussian noise process with covariance matrix $\mathbf{C}[n]$.

Therefore the $\mathcal{P}(\mathbf{s}[n+1]|\mathbf{s}[n])$ in equation (3.31) can be expressed as

$$\begin{aligned}\mathcal{P}(\mathbf{s}[n+1]|\mathbf{s}[n]) = \frac{1}{\sqrt{2\pi|\mathbf{Q}[n]|}} \exp\left\{-\frac{1}{2}(\mathbf{s}[n+1]-\mathbf{A}[n]\mathbf{s}[n])^T\right\} \\ \exp\left\{\mathbf{Q}[n]^{-1}(\mathbf{s}[n+1]-\mathbf{A}[n]\mathbf{s}[n])\right\}\end{aligned} \tag{3.32}$$

and the $\mathcal{P}(\mathbf{z}[n+1]|\mathbf{s}[n+1])$ in equation (3.31) can be expressed as

$$\begin{aligned}\mathcal{P}(\mathbf{z}[n+1]|\mathbf{s}[n+1]) = \frac{1}{\sqrt{2\pi|\mathbf{C}[n+1]|}} \exp\left\{-\frac{1}{2}(\mathbf{z}[n+1]-\mathbf{H}[n+1]\mathbf{s}[n+1])^T\right\} \\ \exp\left\{\mathbf{C}[n+1]^{-1}(\mathbf{z}[n+1]-\mathbf{H}[n+1]\mathbf{s}[n+1])\right\}\end{aligned} \tag{3.33}$$

Using equation (3.32), the coefficient D_n^{11} defined in equation (3.31) is calculated as follows. First we take the natural log of equation (3.32).

$$\begin{aligned}\ln\mathcal{P}(\mathbf{s}[n+1]|\mathbf{s}[n]) = \ln\frac{1}{\sqrt{2\pi|\mathbf{Q}[n]|}} \\ -\frac{1}{2}(\mathbf{s}[n+1]-\mathbf{A}[n]\mathbf{s}[n])^T\mathbf{Q}[n]^{-1}(\mathbf{s}[n+1]-\mathbf{A}[n]\mathbf{s}[n])\end{aligned} \tag{3.34}$$

Taking the derivative of equation (3.34) with respect to the state vector $\mathbf{s}[n]$, yields

$$\begin{aligned}\nabla_{\mathbf{s}[n]} \ln \mathcal{P}(\mathbf{s}[n+1]|\mathbf{s}[n]) &= (\mathbf{s}[n+1] - \mathbf{A}[n]\mathbf{s}[n])^T \mathbf{Q}[n]^{-1} \frac{1}{2}\mathbf{A}[n] \\ &\quad + \frac{1}{2}(\mathbf{s}[n+1] - \mathbf{A}[n]\mathbf{s}[n])^T \mathbf{Q}[n]^{-1}\mathbf{A}[n] \\ &= (\mathbf{s}[n+1] - \mathbf{A}[n]\mathbf{s}[n])^T \mathbf{Q}[n]^{-1}\mathbf{A}[n]\end{aligned} \tag{3.35}$$

Substituting the value of equation (3.35) into the coefficient D_n^{11} defined in equation (3.31) gives

$$\begin{aligned}D_n^{11} &= -E\left\{\nabla_{\mathbf{s}[n]}\left(\nabla_{\mathbf{s}[n]} \ln \mathcal{P}(\mathbf{s}[n+1]|\mathbf{s}[n])\right)\right\} \\ &= -E\left\{\nabla_{\mathbf{s}[n]}\left((\mathbf{s}[n+1] - \mathbf{A}[n]\mathbf{s}[n])^T \mathbf{Q}[n]^{-1}\mathbf{A}[n]\right)\right\} \\ &= \mathbf{A}[n]^T \mathbf{Q}[n]^{-1}\mathbf{A}[n]\end{aligned} \tag{3.36}$$

Similarly, the coefficient D_n^{21} defined in equation (3.31) is calculated as follows.

Taking the derivative of equation (3.34) with respect to the state vector $\mathbf{s}[n+1]$, yields

$$\begin{aligned}\nabla_{\mathbf{s}[n+1]} \ln \mathcal{P}(\mathbf{s}[n+1]|\mathbf{s}[n]) &= -\frac{1}{2}(\mathbf{s}[n+1] - \mathbf{A}[n]\mathbf{s}[n])^T \mathbf{Q}[n]^{-1} \\ &\quad - \frac{1}{2}(\mathbf{s}[n+1] - \mathbf{A}[n]\mathbf{s}[n])^T \mathbf{Q}[n]^{-1} \\ &= -(\mathbf{s}[n+1] - \mathbf{A}[n]\mathbf{s}[n])^T \mathbf{Q}[n]^{-1}\end{aligned} \tag{3.37}$$

Substitute the value of equation (3.37) into the coefficient D_n^{21} defined in equation (3.31) gives

$$\begin{aligned}D_n^{21} &= -E\left\{\nabla_{\mathbf{s}[n]}\left(\nabla_{\mathbf{s}[n+1]} \ln \mathcal{P}(\mathbf{s}[n+1]|\mathbf{s}[n])\right)\right\} \\ &= -E\left\{\nabla_{\mathbf{s}[n]}\left(-(\mathbf{s}[n+1] - \mathbf{A}[n]\mathbf{s}[n])^T \mathbf{Q}[n]^{-1}\right)\right\} \\ &= -\mathbf{Q}[n]^{-1}\mathbf{A}[n]\end{aligned} \tag{3.38}$$

Since the coefficient D_n^{12} is defined as the transpose of coefficient D_n^{21} in equation (3.31), then

$$\begin{aligned}D_n^{12} &= [D_n^{21}]^T \\ &= -\mathbf{A}[n]^T \mathbf{Q}[n]^{-1}\end{aligned} \tag{3.39}$$

The calculation of coefficient D_n^{22} defined in equation (3.31) is as follows
The first part of the coefficient D_n^{22} is $-E\left\{\nabla_{\mathbf{s}[n+1]}\left(\nabla_{\mathbf{s}[n+1]}\ln\mathcal{P}(\mathbf{s}[n+1]|\mathbf{s}[n])\right)^T\right\}$. Substitute the value $\nabla_{\mathbf{s}[n+1]}\ln\mathcal{P}(\mathbf{s}[n+1]|\mathbf{s}[n])$ calculated in equation (3.37) into the first part of coefficient D_n^{22} yields

$$-E\left\{\nabla_{\mathbf{s}[n+1]}\left(\nabla_{\mathbf{s}[n+1]}\ln\mathcal{P}(\mathbf{s}[n+1]|\mathbf{s}[n])\right)^T\right\} = -E\left\{\nabla_{\mathbf{s}[n+1]}\left((\mathbf{s}[n+1]-\mathbf{A}[n]\mathbf{s}[n])^T\mathbf{Q}[n]^{-1}\right)^T\right\}$$
$$= \mathbf{Q}[n]^{-1} \tag{3.40}$$

The second part of the coefficient D_n^{22} in equation (3.31)
$-E\left\{\nabla_{\mathbf{s}[n+1]}\left(\nabla_{\mathbf{s}[n+1]}\ln\mathcal{P}(\mathbf{z}[n+1]|\mathbf{s}[n+1])\right)^T\right\}$ is calculated as follows. The probability $\mathcal{P}(\mathbf{z}[n+1]|\mathbf{s}[n+1])$ is defined in equation (3.33). Taking the natural log of equation (3.33) yields

$$\ln\mathcal{P}(\mathbf{z}[n+1]|\mathbf{s}[n+1]) = \ln\frac{1}{\sqrt{2\pi|\mathbf{C}[n+1]|}} - \frac{1}{2}(\mathbf{z}[n+1]-\mathbf{H}[n+1]\mathbf{s}[n+1])^T$$
$$\mathbf{C}[n+1]^{-1}(\mathbf{z}[n+1]-\mathbf{H}[n+1]\mathbf{s}[n+1]) \tag{3.41}$$

Evaluating the partial derivative $\nabla_{\mathbf{s}[n+1]}\ln\mathcal{P}(\mathbf{z}[n+1]|\mathbf{s}[n+1])$ gives

$$\nabla_{\mathbf{s}[n+1]}\ln\mathcal{P}(\mathbf{z}[n+1]|\mathbf{s}[n+1]) =$$
$$= \nabla_{\mathbf{s}[n+1]}(c_1 - \frac{1}{2}(\mathbf{z}[n+1]-\mathbf{H}[n+1]\mathbf{s}[n+1])^T$$
$$\mathbf{C}[n+1]^{-1}(\mathbf{z}[n+1]-\mathbf{H}[n+1]\mathbf{s}[n+1]))$$
$$= (\mathbf{z}[n+1]-\mathbf{H}[n+1]\mathbf{s}[n+1])^T\mathbf{C}[n+1]^{-1}\mathbf{H}[n+1] \tag{3.42}$$

where c_1 is a constant.

Evaluating the expression $-E\left\{\nabla_{\mathbf{s}[n+1]}\left(\nabla_{\mathbf{s}[n+1]}\ln\mathcal{P}(\mathbf{z}[n+1]|\mathbf{s}[n+1])\right)^T\right\}$ by using

the value of $\nabla_{\mathbf{s}[n+1]} \ln \mathcal{P}(\mathbf{z}[n+1]|\mathbf{s}[n+1])$ in equation (3.42), we get

$$-E\left\{\nabla_{\mathbf{s}[n+1]}\left(\nabla_{\mathbf{s}[n+1]} \ln \mathcal{P}(\dot{\mathbf{z}}[n+1]|\mathbf{s}[n+1])\right)^T\right\} = \mathbf{H}[n+1]^T\mathbf{C}[n+1]^{-1}\mathbf{H}[n+1] \tag{3.43}$$

Therefore D_n^{22} becomes the summation of the first part in equation (3.40) and the second part in equation (3.43).

$$D_n^{22} = \mathbf{Q}[n]^{-1} + \mathbf{H}[n+1]^T\mathbf{C}[n+1]^{-1}\mathbf{H}[n+1] \tag{3.44}$$

Substituting the values of D_n^{11}, D_n^{12}, D_n^{21} and D_n^{22} into equation (3.30), the FIM becomes

$$\begin{aligned}\mathbf{I}(\mathbf{s}[n+1]) = \mathbf{Q}[n]^{-1} + \mathbf{H}[n+1]^T\mathbf{C}[n+1]^{-1}\mathbf{H}[n+1] \\ - \mathbf{Q}[n]^{-1}\mathbf{A}[n]\left(\mathbf{I}(\mathbf{s}[n]) + \mathbf{A}[n]^T\mathbf{Q}[n]^{-1}\mathbf{A}[n]\right)^{-1}\mathbf{A}[n]^T\mathbf{Q}[n]^{-1}\end{aligned} \tag{3.45}$$

Applying the matrix inversion lemma [54] to equation (3.45), a more compact form follows, such that

$$\mathbf{I}(\mathbf{s}[n+1]) = (\mathbf{Q}[n] + \mathbf{A}[n]\ \mathbf{I}(\mathbf{s}[n])^{-1}\mathbf{A}[n]^T)^{-1} + \mathbf{H}[n+1]^T\mathbf{C}[n+1]^{-1}\mathbf{H}[n+1] \tag{3.46}$$

Note that equation (3.46) is only valid for the linear case, where $\mathbf{A}[n]$ and $\mathbf{H}[n]$ are independent of the state vector $\mathbf{s}[n]$. In the case where either $\mathbf{A}[n]$ or $\mathbf{H}[n]$ are nonlinear, then $\widetilde{\mathbf{A}}[n]$ and $\widetilde{\mathbf{H}}[n]$ are defined as the Jacobian's of $\mathbf{a}(\mathbf{s}[n])$ and $\mathbf{h}(\mathbf{s}[n])$ respectively, evaluated at the true value of $\mathbf{s}[n]$ such that $\widetilde{\mathbf{A}}[n] = (\nabla_{\mathbf{s}[n]}\mathbf{a}^T(\mathbf{s}[n]))^T$ and $\widetilde{\mathbf{H}}[n] = (\nabla_{\mathbf{s}[n]}\mathbf{h}^T(\mathbf{s}[n]))^T$.

Next the parameters in equation (3.46) are studied in Section 3.4.2 when calculating the CRLB variance using the DR/RSS measurements.

3.4.2 State And Observation Model Initialization

Let the state vector be $\mathbf{s}[n] = [\ X[n]\ Y[n]\ v_X[n]\ v_Y[n]]^T$, where $X[n]$ and $Y[n]$ are the X and Y position coordinates of the robot, and $v_X[n]$ and $v_Y[n]$ are the X and Y velocity

components. Matrix $\mathbf{A}[n]$ in equation (3.46) defines the robot kinematics with respect to the state vector $\mathbf{s}[n]$ such that

$$\mathbf{A} = \begin{bmatrix} 1 & 0 & \Delta t & 0 \\ 0 & 1 & 0 & \Delta t \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \tag{3.47}$$

where Δt is the interval at which observation vectors are captured. Since $\mathbf{A}$ is time independent, the time dependence n has been dropped. Assuming the robot speed is fixed at 0.2 m/s and the sampling distance is equal to 4 m, $\Delta t = 4/0.2$=20 sec. A sample distance greater than 2 meters is necessary to ensure the RSS measurements in the observation vector are uncorrelated from one time increment to the next [53].

The process noise $\mathbf{u}[n]$ in equation (3.19) is given by

$$\mathbf{u}[n] = [0 \;\; 0 \;\; u_X[n] \; u_Y[n]]^T \tag{3.48}$$

where $u_X[n] = v_X[n] - v_X[n-1]$, and $u_Y[n] = v_Y[n] - v_Y[n-1]$. The values of $u_X[n]$ and $u_Y[n]$ is equal to the maximum possible change in velocity in the X and Y directions respectively, and this is equal to 0.4 m/s. This would occur if the robot stops and makes a 180^0 turn. The covariance matrix $\mathbf{Q}$ of the process noise $\mathbf{u}[n]$ is given by

$$\mathbf{Q} = \begin{bmatrix} 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & \sigma_{Vx}^2 & 0 \\ 0 & 0 & 0 & \sigma_{Vy}^2 \end{bmatrix} \tag{3.49}$$

where the value of σ_{Vx}^2 and σ_{Vy}^2 equal 0.4^2 m^2/s^2. Assuming that the error corrupting the velocity components is equally likely to occur in any direction, then the variances in the X and Y directions may be assumed uncorrelated.

The observation model for the system is given by

$$\mathbf{z}[n] = \mathbf{h}(\mathbf{s}[n]) + \mathbf{w}[n] \tag{3.50}$$

Unlike equation (3.20), equation (3.50) describes the general case where $\mathbf{h}(\mathbf{s}[n])$ is a non-linear function of the state vector.

The covariance matrix of the observation noise, $\mathbf{C}[n]$ is given by

$$\mathbf{C}[n] = \begin{bmatrix} \sigma_X^2[n] & 0 & 0 & 0 & \ddots & 0 \\ 0 & \sigma_Y^2[n] & 0 & 0 & \dots & 0 \\ 0 & 0 & \sigma_R^2 & 0 & \dots & 0 \\ 0 & 0 & 0 & \ddots & \dots & \vdots \\ \vdots & \vdots & \vdots & \ddots & \ddots & 0 \\ 0 & 0 & 0 & \dots & 0 & \sigma_R^2 \end{bmatrix}_{(m+2)\times(m+2)} \tag{3.51}$$

where σ_R^2 represents the shadowing variance. Note that the values of $\sigma_X^2[n]$ and $\sigma_Y^2[n]$ are a function of time n since they increase as the mobile moves. These values are calculated using equation (2.6) in Chapter 2.

The measurement vector, $\mathbf{h}(\mathbf{s}[n])$ in equation (3.50) represents the DR and RSS measurements performed by the robot such that

$$\mathbf{h}(\mathbf{s}[n]) = \begin{bmatrix} X[n] \\ Y[n] \\ 10\eta \log\left(\sqrt{(X[n]-X_{b,1})^2 + (Y[n]-Y_{b,1})^2}\right) \\ \vdots \\ 10\eta \log\left(\sqrt{(X[n]-X_{b,m})^2 + (Y[n]-Y_{b,m})^2}\right) \end{bmatrix}_{(m+2)\times 1} \tag{3.52}$$

and η is the path loss exponent discussed in Section 3.3. The DR measurements are

equal to the desired robot position and the RSS measurements are given by equation (3.2).

The matrix $\mathbf{H}[n]$ is defined as $\mathbf{H}[n] = \frac{\delta \mathbf{h}}{\delta \mathbf{s}[n]}$ when $\mathbf{h}$ is a linear function of $\mathbf{s}[n]$. However, since $\mathbf{h}(\mathbf{s}[n])$ is a non-linear function of the elements of the state vector $\mathbf{s}[n]$, then $\mathbf{H}[n]$ is approximated by a first order partial derivatives. Let $\widetilde{\mathbf{H}}[n]$ be defined as the Jacobian of $\mathbf{h}(\mathbf{s}[n])$, evaluated at the true value of $\mathbf{s}[n]$ such that $\widetilde{\mathbf{H}}[n] = (\nabla_{\mathbf{s}[n]} \mathbf{h}^T(\mathbf{s}[n]))^T$.

The observation matrix, $\widetilde{\mathbf{H}}[n]$ is given by

$$\widetilde{\mathbf{H}}[n] = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ \frac{10\eta}{\ln 10} \frac{(X[n]-X_{b,1})}{(X[n]-X_{b,1})^2+(Y[n]-Y_{b,1})^2} & \frac{10\eta}{\ln 10} \frac{(Y[n]-Y_{b,1})}{(X[n]-X_{b,1})^2+(Y[n]-Y_{b,1})^2} & 0 & 0 \\ \vdots & \vdots & \vdots & \vdots \\ \frac{10\eta}{\ln 10} \frac{(X[n]-X_{b,m})}{(X[n]-X_{b,m})^2+(Y[n]-Y_{b,m})^2} & \frac{10\eta}{\ln 10} \frac{(Y[n]-Y_{b,m})}{(X[n]-X_{b,m})^2+(Y[n]-Y_{b,m})^2} & 0 & 0 \end{bmatrix}_{(m+2)\times 4} \tag{3.53}$$

3.5 Analyzing The Static And The Dynamic CRLB For Different Scenarios

In this section, the performance bounds for hybrid RSS/DR localization will be compared using the static and the dynamic CRLB expressions. The dynamic CRLB is calculated based on the recursive FIM equation (3.46), while the static CRLB is calculated based on equations (3.17) and (3.18).

It is important to note that the static and dynamic CRLBs describe two different systems for estimating the robot position. The dynamic CRLB considers both the current and previous position estimates to estimate the robot position, and thus the

dynamic CRLB is recursive i.e. the CRLB at time $n+1$ depends on the previous CRLB value at time n. On the other hand, the static CRLB relies only on the current position estimate and thus it can be considered as isolated snapshots of the robot position estimate. One would expect that prior knowledge of the previous position estimate would give the dynamic CRLB a higher positional accuracy than the static CRLB case.

Section 3.5.1 describes the simulation parameters that are used in the remaining sections. Section 3.5.2 examines the static and dynamic CRLB for varying the shadowing standard deviation. Section 3.5.3 examines the static and dynamic CRLB for varying the number of beacons on a grid. Section 3.5.4 examines the static and dynamic CRLB for varying the number of random beacons. Section 3.5.5 examines the static and dynamic CRLB for varying the segment length traveled by the robot.

3.5.1 Simulation Description

The robot path always starts from the point with coordinates $(1, 1)$. Assume the initial position of the robot is known, therefore the initial desired state vector at time 0, is equal to the actual state vector $\mathbf{s}[0]$ such that

$$\mathbf{s}[0] = [1 \quad 1 \quad 0.1414 \; 0.1414]^T \tag{3.54}$$

The first two elements of the vector $\mathbf{s}[0]$ are the initial robot coordinates $(X[n], Y[n])$, initialized to (1,1). The third and fourth elements are the X and Y components of the robot's velocity. The simulated robot speed is assumed to be 0.2 m/s and since the robot is assumed to start moving with a heading of 45^o with respect to the X axis, the X and Y components of the robot velocity are equal, and are given by $0.2 \cos 45^o$ and $0.2 \sin 45^o$.

The desired path of the robot is generated as follows. The path consists of segments, where each segment length varies uniformly from 5 to 40 meters, the angle varies uniformly from 0 to 2π, and the robot path is constrained within a 100 m by 100 m area.

The maximum battery life of the alkaline batteries that powers iRobot is 1.5 hours, and since the robot takes 20 sec to go over a segment of length 4 m, the maximum number of samples along the robot path is calculated to be 270.

The CRLB values on the graphs are calculated as follows. For each sample point along the robot path the dynamic CRLB is calculated using equation (3.46), and the static CRLB is calculated using equations (3.17) and (3.18). At the end of one random path, we have 270 CRLB sample values for each of the dynamic and static CRLB. The average CRLB for those 270 values are then computed for that path. Each point on the graph represents the average CRLB for 1000 random paths.

The matrix, $\mathbf{A}$, in equation (3.47) defines a constant velocity motion model with a sampling time, Δt, equal to 20 sec. The covariance matrix $\mathbf{Q}$ of the process noise $\mathbf{u}[n]$ is given by equation (3.49), where σ_{Vx}^2 and σ_{Vy}^2 are equal to $0.4^2 m^2/s^2$.

The CRLB variance $\mathbf{I}(\mathbf{s}[n])^{-1}$ in equation (3.46) is initialized to a value of $10,000\mathbf{I}$, where $\mathbf{I}$ is the identity matrix. This large initial variance value is chosen so as not to bias the estimator.

The observation matrix $\widetilde{\mathbf{H}}[n]$ in equation (3.53) defines the observations made by the robot. These are the DR and RSS measurements. These measurements are corrupted by the observation noise, with covariance matrix $\mathbf{C}[n]$ defined in equation (3.51).

Every 4 m along the path, the estimator performs DR and RSS measurements into the filter from m beacons. A path loss exponent typical to indoor scenarios of $\eta = 3$ [45]

and a shadowing standard deviation of $\sigma_R = 2$ dB [53] are assumed.

3.5.2 CRLB Analysis For Varying The Shadowing Standard Deviation

In this section, the CRLB for the static and dynamic CRLB are studied when varying the shadowing standard deviation. Assume there are 9 beacons distributed as shown in Figure 3.1. Figures 3.2 and 3.3 show the CRLB variance for the static and the dynamic case in the X and Y directions respectively versus the shadow standard deviation.

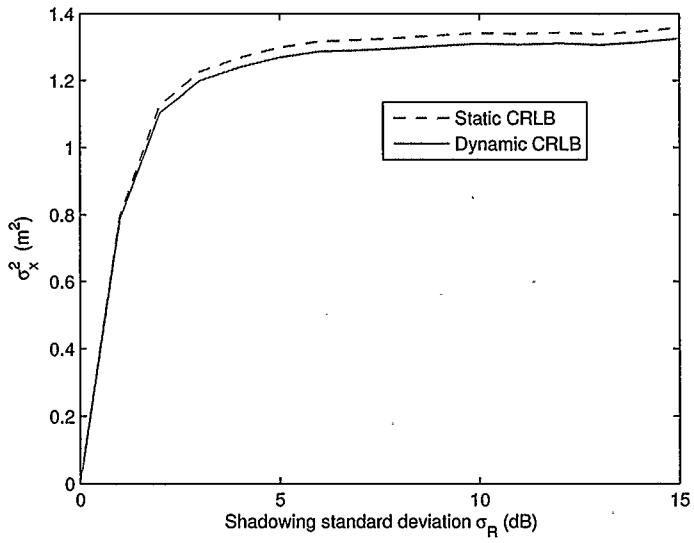

Figure 3.2: CRLB versus the shadowing standard deviation in the X direction.

From the plots, it is evident that the dynamic CRLB has a higher positional accuracy in the X and Y direction compared to the static CRLB. This is because the dynamic CRLB uses past observations to predict the present states, whereas the static CRLB relies only on the current observations. Note that the graph flattens out as the shadowing standard deviation increases. At these high values of σ_R^2, the RSS values no longer add any useful information to the estimate of the robot position. The esti-

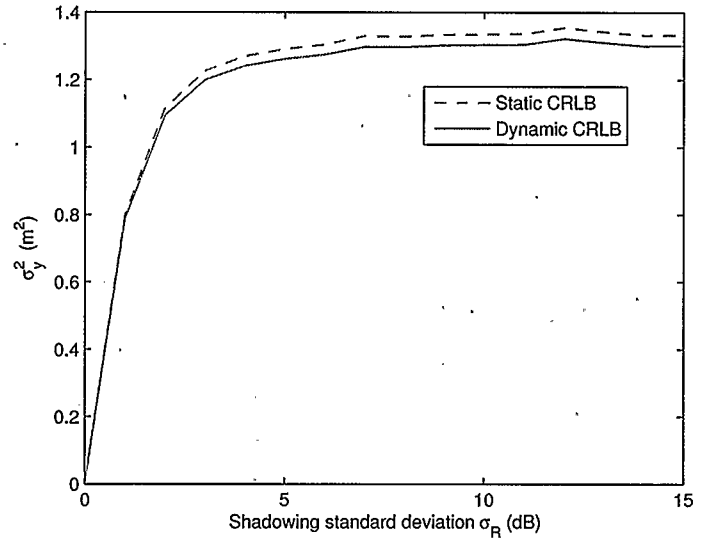

Figure 3.3: CRLB versus the shadowing standard deviation in the Y direction.

mator therefore ignores the RSS value and the overall estimator accuracy converges to that achieved using DR information only. The graphs show that a shadowing standard deviation less than 3 dB is required for the RSS measurement to show a significant improvement over the DR information.

3.5.3 CRLB Analysis For Varying The Number Of Beacons On A Grid

In this section, the CRLB is studied when the beacons are distributed uniformly in a grid like shape in the 100 m by 100 m area. The beacons are distributed as follows.

Let the initial X and Y beacon coordinates be equal such that $X_{b,1} = Y_{b,1}$, and assume they are randomly chosen from 0-30 m. For a given number of beacons, m, the number of rows, N_r, and columns, N_c, are defined such that the number of rows $N_r = \sqrt{m}$. If N_r is not an integer, then N_r is rounded up to the nearest integer. The number of columns, $N_c = m/N_r$. If the number of beacons is odd, then the number of

columns is rounded up to the nearest integer.

Once the number of rows and columns are defined, the X coordinate spacing Δ_X between the beacons is calculated as $(100 - 2X_{b,1})/(N_c - 1)$, and the Y coordinate spacing Δ_Y between the beacons is calculated as $(100 - 2Y_{b,1})/(N_r - 1)$.

For a given row, the X position, $X_{b,i+1} = X_{b,i} + \Delta_X$ and $i = 1, ..., N_c - 1$

For a given column, the Y position, $Y_{b,j+1} = Y_{b,j} + \Delta_Y$ and $j = 1, ..., N_r - 1$

As an example, suppose we have 6 beacons, the number of rows would be $N_r = \sqrt{6} = 2.45$, rounding the result up we get $N_r = 3$, and $N_c = m/N_r = 6/3 = 2$. Assuming that the first beacon position is $(20, 20)$, then $\Delta_X = (100 - 2(20))/1 = 60$, and $\Delta_Y = (100 - 2(20))/2 = 30$. The beacons positions are shown in Figure 3.4.

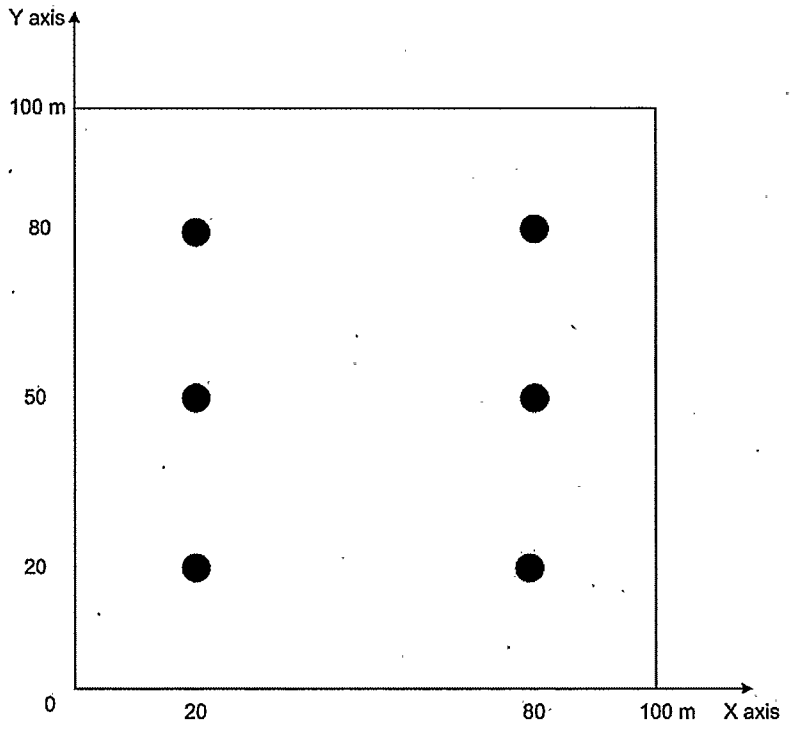

Figure 3.4: Six uniform beacon configuration.

This scenario would be typical to a WiFi network were the beacons are distributed

evenly within a large area. Figures 3.5 and 3.6 show the CRLB for the static and the dynamic case in the X and Y directions respectively versus the number of beacons.

As expected, the dynamic CRLB outperforms the static CRLB. The dynamic CRLB estimator seems to benefit more from an investment of beacons. This is because the dynamic CRLB takes into account the previous CRLB estimate as well as the motion model of the robot to find the CRLB estimate. At a high number of beacons, the CRLB for both the static and dynamic cases improve because there are more RSS information available. However, for a given sample point along the robot path, the dynamic CRLB makes use of the previous "accurate" estimate when finding the CRLB estimate the next sample point.

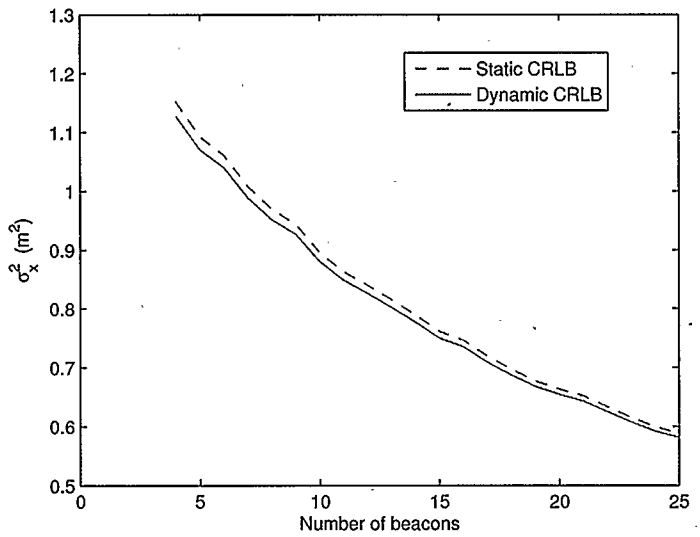

Figure 3.5: CRLB versus the number of beacons on a grid in the X direction.

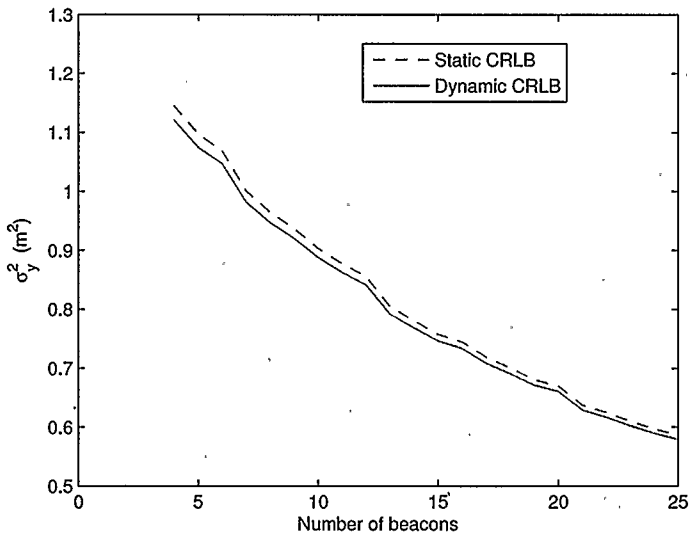

Figure 3.6: CRLB versus the number of beacons on a grid in the Y direction.

3.5.4 CRLB Analysis For Varying The Number Of Random Beacons

In this section, the CRLB is examined when varying the number of beacons from 4 to 50 beacons. The beacons' $(X_{b,k}, Y_{b,k})$ position are randomly selected from a uniform distribution that ranges from 0 to 100 m. This scenario would be typical to an Ad hoc wireless network were the beacons would be deployed randomly in an unknown or hostile environment.

Figures 3.7 and 3.8 show the CRLB for the static and the dynamic case in the X and Y directions respectively versus the number of random beacons.

Similar to Section 3.5.3, the dynamic CRLB has a better positional accuracy than the static CRLB. This is because the static CRLB does not make use of the a priori knowledge of the CRLB values into its formula.

Comparing the performance of the CRLB for the random beacon configuration and the CRLB when the beacons are distributed in a grid in Section 3.5.3. Figure 3.9

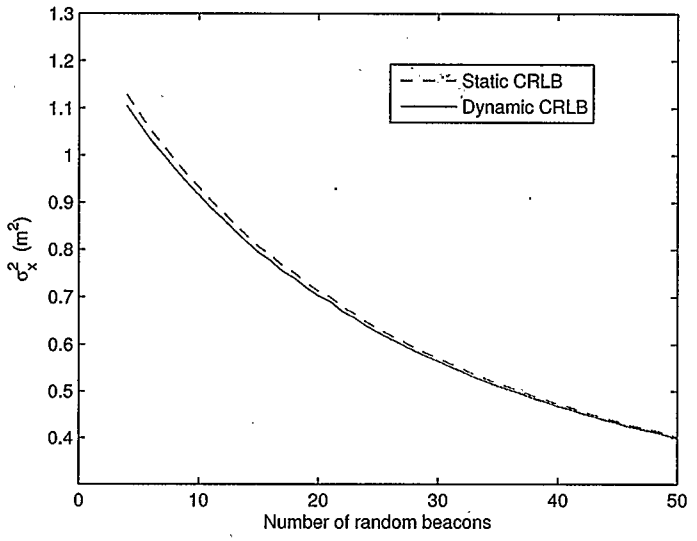

Figure 3.7: CRLB versus the number of random beacons in the X direction.

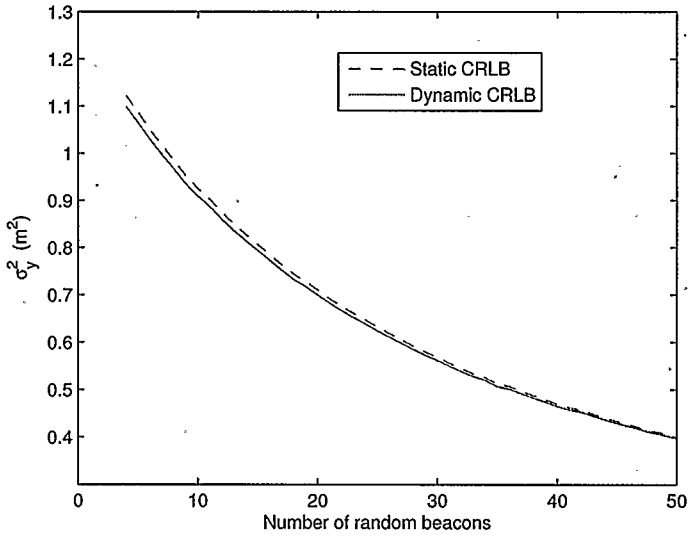

Figure 3.8: CRLB versus the number of random beacons in the Y direction.

compares the CRLB for the random and uniform beacon configuration in the X direction. Similarly, Figure 3.10 compares the CRLB for the random and uniform beacon configuration in the Y direction.

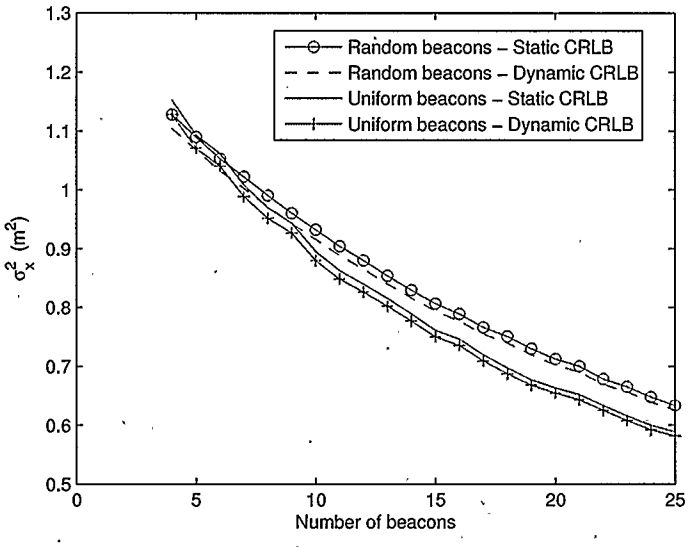

Figure 3.9: Comparing the CRLB for the uniform and random beacon configuration in the X direction.

Note that the uniform beacon configuration has a better performance than the random beacon configuration. This is possibly because in the uniform beacon scenario the beacons are equally distributed across the whole area. Whereas, in a random beacon scenario there are more chance that the beacons would be distributed in an unfavorable geometry, such as being concentrated at one corner. This situation is an unfavorable situation since the Jacobian matrix of the CRLB would be badly conditioned.

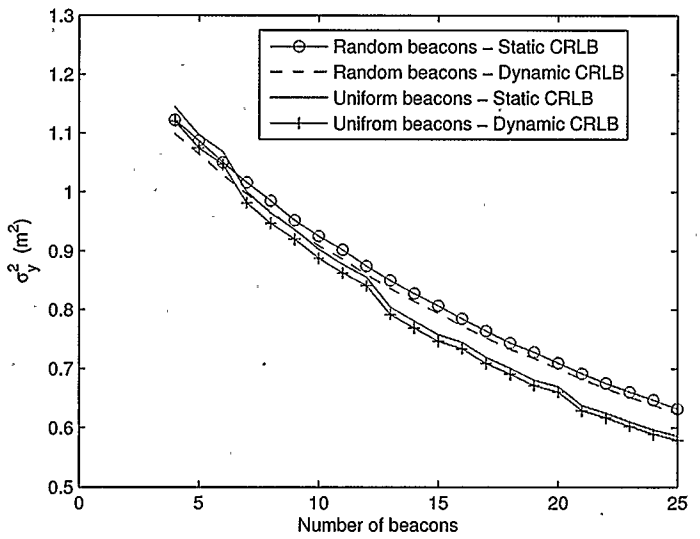

Figure 3.10: Comparing the CRLB for the uniform and random beacon configuration in the Y direction.

3.5.5 CRLB Analysis When Varying The Segment Length Traveled By The Robot

In the simulation description in Section 3.5.1 the segment length is assumed to vary uniformly from 5-40 m. In this section, the CRLB is examined when the segment length varies uniformly from 5-10 m, 10-20 m, 20-30 m and 30-40 m. The number of beacons is assumed to be 9. The beacons are distributed as in Figure 3.1.

The figures show that as the segment length increases the error in position increases, until it reaches a point were further increase in segment length does not produce a proportional increase in error.

A possible reason for this non-proportional increase in positioning error is as follows. The greater the segment length moved, the greater the distance moved by the robot and the greater the distance DR error. However by moving longer segment length, we have less angle turns, and thus less angle DR error, and it was found that angle error

contributes more DR error than distance error. And thus at higher segment length, there are more distance DR error but less angle DR error, and therefore we see a smaller increase in the error variance. Also, since the robot path is constrained in the 100 m by 100 m area, more of the really long segments will be discarded.

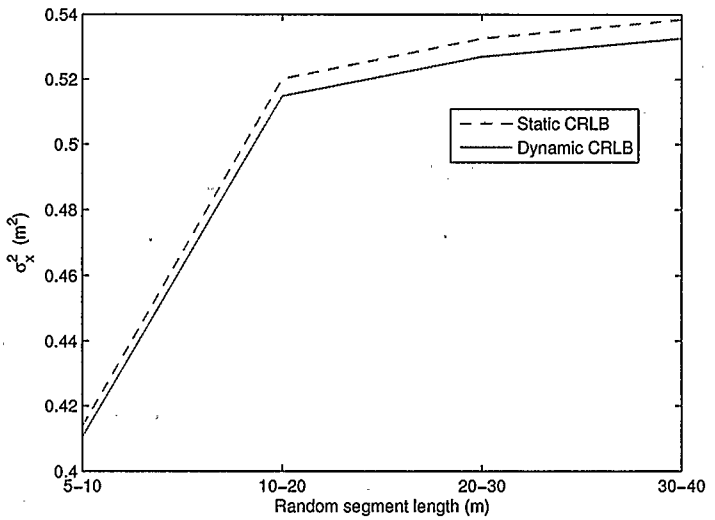

Figure 3.11: CRLB versus the segment length in the X direction.

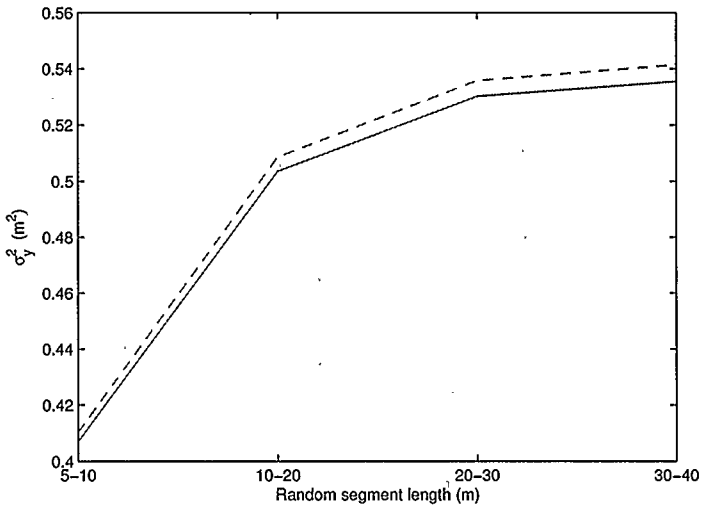

Figure 3.12: CRLB versus the segment length in the Y direction.

Chapter 4

Tracking A Moving Target

The second method used to evaluate the performance of the DR/RSS system is the Kalman filter. The Kalman filter is considered in order to show the practical positioning accuracy possible for a DR/RSS system. It is a recursive estimation technique that estimates the robot state. Robot state may include the target position, velocity and acceleration. The state is calculated using observations that are related to the robot state.

Section 4.1 describes the linear Kalman filter and the non-linear Kalman filter known as the extended Kalman filter (EKF). The EKF deals with situations where the motion model of the robot and/or the observation model are a non-linear function of the robot states.

Section 4.2 shows the extended Kalman filter performance for tracking a robot over different paths using RSS/DR positioning.

4.1 Kalman Filter Theory

The Kalman filter is an estimation technique that has applications in tracking moving targets such as in [23], estimating the radio channel [43] and restoring images contaminated by noise [44].

Under a linear system model, where the system and measurement noises are white and Gaussian, the Kalman filter is an optimal estimator with respect to minimizing the

mean squared error (MSE) of the parameters to be estimated.

The Kalman filter is composed of two phases: a prediction phase and an update phase. In the prediction phase, the parameters to be estimated e.g. the target states are estimated and the accuracy of that estimate is given by a covariance matrix. In the update phase, the observations are incorporated and an updated state and updated covariance matrix are computed. These updated state and covariance matrix are then used to calculate the predicted state and covariance matrix for the next iteration.

Section 4.1.1 examines the linear Kalman filter when the state and the observation models are linear functions of the state variables. Section 4.1.2 examines the extended Kalman filter, where the state and/or the observation models are nonlinear functions of the state variables.

4.1.1 Linear Kalman Filter

Following Kay's notation [49], let the vector $\mathbf{s}[n]$ be the vector of state parameters at time sample n. The relationship between the state vector at time n and $n-1$ is given by the first-order Gauss-Markov process such that

$$\mathbf{s}[n] = \mathbf{A}[n]\mathbf{s}[n-1] + \mathbf{u}[n] \quad n \geq 0 \tag{4.1}$$

where $\mathbf{A}[n]$ is a matrix of appropriate dimension, $\mathbf{u}[n]$ is the process noise, having a zero mean white Gaussian noise (WGN) with covariance matrix $\mathbf{Q}$.

Let $\mathbf{z}[n]$ denote a vector of observation parameters at time sample n, such that

$$\mathbf{z}[n] = \mathbf{H}[n]\mathbf{s}[n] + \mathbf{w}[n] \quad n \geq 0 \tag{4.2}$$

where $\mathbf{H}[n]$ is the observation matrix and $\mathbf{w}[n]$ is the observation noise vector having a zero mean (WGN) distribution with covariance matrix $\mathbf{C}[n]$.

Note that the Kalman filter state equation (4.1) and observation equation (4.2) are identical to the dynamic CRLB state and observation equations in Section 3.4. This is because both the dynamic CRLB and Kalman filter describe the same system. The Kalman filter, however, shows the practical positioning estimation, whereas the CRLB shows the theoretical bound on the variance.

The iterative Kalman filter equations are given by:

Prediction vector:

$$\hat{\mathbf{s}}[n|n-1] = \mathbf{A}[n]\hat{\mathbf{s}}[n-1|n-1] \tag{4.3}$$

The predicted state $\hat{\mathbf{s}}[n|n-1]$, designated with the ^ on top of $\mathbf{s}$ denotes an estimate of the state vector $\mathbf{s}[n]$ at time n given knowledge of the previous measurements. The state $\hat{\mathbf{s}}[n|n-1]$ is predicted using knowledge of how the system evolves, presented by matrix $\mathbf{A}[n]$, from the previous state, $\hat{\mathbf{s}}[n-1|n-1]$.

Minimum Prediction MSE Matrix:

$$\mathbf{M}[n|n-1] = \mathbf{A}[n]\mathbf{M}[n-1|n-1]\mathbf{A}[n]^T + \mathbf{Q} \tag{4.4}$$

The matrix $\mathbf{M}[n|n-1]$ denotes the predicted MSE at time n based on the MSE at time $n-1$, $\mathbf{M}[n-1|n-1]$. In other words, it estimates the uncertainty of the predicted state $\hat{\mathbf{s}}[n|n-1]$. The predicted MSE $\mathbf{M}[n|n-1]$ is a function of the matrix $\mathbf{A}[n]$, $\mathbf{M}[n-1|n-1]$ and the covariance noise of the system $\mathbf{Q}$.

Kalman Gain Matrix:

$$\mathbf{K}[n] = \mathbf{M}[n|n-1]\mathbf{H}^T[n](\mathbf{C}[n] + \mathbf{H}[n]\mathbf{M}[n|n-1]\mathbf{H}^T[n])^{-1} \tag{4.5}$$

The Kalman gain, $\mathbf{K}[n]$ represents a weighting function of the predicted values and the measured values. The higher the certainty of a given value, the more weight is given to

that value. The matrix $\mathbf{H}[n]$ represents the measurements/observation, and the matrix $\mathbf{C}[n]$ is the covariance of the observation noise. Note that if the measurement $\mathbf{C}[n]$ is very noisy, the Kalman gain $\mathbf{K}[n]$ weight is less. In the limit $\mathbf{C}[n]$ goes to infinity, $\mathbf{K}[n]$ approaches zero. However, when the predicted system noise, represented by $\mathbf{M}[n|n-1]$, is very noisy, the Kalman gain approaches unity.

Update vector:

$$\hat{\mathbf{s}}[n|n] = \hat{\mathbf{s}}[n|n-1] + \mathbf{K}[n](\mathbf{z}[n] - \mathbf{H}[n]\hat{\mathbf{s}}[n|n-1]) \tag{4.6}$$

The updated state $\hat{\mathbf{s}}[n|n]$ represents the predicted state vector estimate in equation (4.3), plus the Kalman gain, $\mathbf{K}[n]$ times the difference between the measurement $\mathbf{z}[n]$ and the best prediction of $\mathbf{z}[n]$ at time $n-1$, represented by $\mathbf{H}[n]\hat{\mathbf{s}}[n|n-1]$. Note that when the measurement is very noisy, the Kalman gain approaches zero, and the corrected state $\hat{\mathbf{s}}[n|n]$ is equal to the predicted state $\hat{\mathbf{s}}[n|n-1]$. Also, when the system MSE approaches zero in equation (4.5), the Kalman gain approaches zero as well. That is the filter is absolutely sure of the estimate regardless of the measurement. However, when the system noise is very noisy, the Kalman gain becomes equal to one, and the measurement is weighted heavily.

Minimum MSE Matrix:

$$\mathbf{M}[n|n] = (\mathbf{I} - \mathbf{K}[n]\mathbf{H}[n])\mathbf{M}[n|n-1] \tag{4.7}$$

The matrix $\mathbf{M}[n|n]$ represents the updated MSE. Note that when the observation noise is very high the Kalman gain $\mathbf{K}[n]$ is zero and the updated MSE equals the predicted MSE $\mathbf{M}[n|n-1]$ produced from the system.

4.1.2 Extended Kalman Filter

So far, both the state and the observation equations were a linear function of the state. And since both the state and the observations were corrupted by white Gaussian noise, the Kalman filter is optimal in minimizing the MSE. However, in situations where the state equations and/or the observation equations are nonlinear functions of the state vector, the Kalman filter MSE becomes intractable, and one possible solution is to linearize the state and the observation equations with respect to the state vector estimate.

The Kalman filter based on linearizing the state and or the observation equations about the current state estimate is known as the extended Kalman filter (EKF). Suppose we have

$$\mathbf{s}[n] = \mathbf{a}(\mathbf{s}[n-1]) + \mathbf{u}[n] \qquad n \geq 0 \tag{4.8}$$

as the state equation where $\mathbf{a}$ is a function of $\mathbf{s}[n-1]$ and

$$\mathbf{z}[n] = \mathbf{h}(\mathbf{s}[n]) + \mathbf{w}[n] \qquad n \geq 0 \tag{4.9}$$

as the observation equation, where $\mathbf{h}$ is a function of $\mathbf{s}[n]$.

Linearizing $\mathbf{a}(\mathbf{s}[n-1])$ and $\mathbf{h}(\mathbf{s}[n])$ about the state vector estimate using the first order Taylor series results in

$$\begin{aligned}\mathbf{a}(\mathbf{s}[n-1]) \approx\ & \mathbf{a}(\hat{\mathbf{s}}[n-1|n-1]) \\ & + \frac{\delta \mathbf{a}}{\delta \mathbf{s}[n-1]}\bigg|_{\mathbf{s}[n-1]=\hat{\mathbf{s}}[n-1|n-1]} (\mathbf{s}[n-1] - \hat{\mathbf{s}}[n-1|n-1])\end{aligned} \tag{4.10}$$

$$\mathbf{h}(\mathbf{s}[n]) \approx \mathbf{h}(\hat{\mathbf{s}}[n|n-1]) + \frac{\delta \mathbf{h}}{\delta \mathbf{s}[n]}\bigg|_{\mathbf{s}[n]=\hat{\mathbf{s}}[n|n-1]} (\mathbf{s}[n] - \hat{\mathbf{s}}[n|n-1]) \tag{4.11}$$

Let

$$\breve{\mathbf{A}}[n-1] = \left.\frac{\delta \mathbf{a}}{\delta \mathbf{s}[n-1]}\right|_{\mathbf{s}[n-1]=\hat{\mathbf{s}}[n-1|n-1]} \tag{4.12}$$

$$\breve{\mathbf{H}}[n] = \left.\frac{\delta \mathbf{h}}{\delta \mathbf{s}[n]}\right|_{\mathbf{s}[n]=\hat{\mathbf{s}}[n|n-1]} \tag{4.13}$$

Substituting equations (4.10) and (4.12) into (4.8) and equations (4.11) and (4.13) into (4.9) we get the linearized state and observation equations respectively.

$$\mathbf{s}[n] = \breve{\mathbf{A}}[n-1]\mathbf{s}[n-1] + \mathbf{u}[n] + (\mathbf{a}(\hat{\mathbf{s}}[n-1|n-1]) - \breve{\mathbf{A}}[n-1]\hat{\mathbf{s}}[n-1|n-1]) \tag{4.14}$$

$$\mathbf{z}[n] = \breve{\mathbf{H}}[n]\mathbf{s}[n] + \mathbf{w}[n] + (\mathbf{h}(\hat{\mathbf{s}}[n|n-1]) - \breve{\mathbf{H}}[n]\hat{\mathbf{s}}[n|n-1]) \tag{4.15}$$

The Extended Kalman filter equations become:

Prediction vector:

$$\hat{\mathbf{s}}[n|n-1] = \mathbf{a}(\hat{\mathbf{s}}[n-1|n-1]) \tag{4.16}$$

Note that the prediction equation (4.16) is different from that in the linear case in equation (4.3). The explanation is as follows. The state in equation (4.14) represents the first order Taylor series approximation of the nonlinear state $\mathbf{a}(\mathbf{s}[n-1])$. In equation (4.16), the prediction state, $\hat{\mathbf{s}}[n|n-1]$ is defined as $E(\mathbf{s}[n]|\mathbf{Z}[n-1])$, where $E(.)$ denotes the expected value and Z[n-1] is a vector containing all the observations from time 0 up to time $n-1$. Using the state equation $\mathbf{s}[n]$ in equation (4.8), the $E(\mathbf{s}[n]|\mathbf{Z}[n-1])$ becomes

$$\begin{aligned} E(\mathbf{s}[n]|\mathbf{Z}[n-1]) &= E(\mathbf{a}(\mathbf{s}[n-1]|\mathbf{Z}[n-1]) + E(\mathbf{u}[n]|\mathbf{Z}[n-1]) \\ &= \mathbf{a}(\mathbf{s}[n-1|n-1]) \end{aligned} \tag{4.17}$$

The second term $E(\mathbf{u}[n]|\mathbf{Z}[n-1])$ in equation (4.17) is equal to zero since the process noise $\mathbf{u}[n]$ is assumed to be uncorrelated with all the previous observations $\mathbf{Z}[n-1]$.

Minimum Prediction MSE:

$$\mathbf{M}[n|n-1] = \breve{\mathbf{A}}[n]\mathbf{M}[n-1|n-1]\breve{\mathbf{A}}[n]^T + \mathbf{Q} \tag{4.18}$$

Kalman Gain Matrix:

$$\mathbf{K}[n] = \mathbf{M}[n|n-1]\breve{\mathbf{H}}^T[n](\mathbf{C}[n] + \breve{\mathbf{H}}[n]\mathbf{M}[n|n-1]\breve{\mathbf{H}}^T[n])^{-1} \tag{4.19}$$

Update vector:

$$\hat{\mathbf{s}}[n|n] = \hat{\mathbf{s}}[n|n-1] + \mathbf{K}[n](\mathbf{z}[n] - \mathbf{h}(\hat{\mathbf{s}}[n|n-1])) \tag{4.20}$$

Minimum MSE Matrix:

$$\mathbf{M}[n|n] = (\mathbf{I} - \mathbf{K}[n]\breve{\mathbf{H}}[n])\mathbf{M}[n|n-1] \tag{4.21}$$

Note that equations (4.18), (4.19), (4.20) and (4.21) are exactly the same as in the linear Kalman filter case, the only equation that changed is (4.16) describing the predicted state.

Even though equation (4.20) describing the updated state looks different from its linear counterpart in equation (4.6), they are actually identical.

For the linear case, substituting the observation equation $\mathbf{z}[n]$ in equation (4.2) into the second term in equation (4.6), we get $\mathbf{K}[n](\mathbf{H}[n]\mathbf{s}[n] + \mathbf{w}[n] - \mathbf{H}[n]\hat{\mathbf{s}}[n|n-1]) = \mathbf{H}[n](\mathbf{s}[n] - \hat{\mathbf{s}}[n|n-1]) + \mathbf{w}[n]$. For the EKF case, substituting the observation equation

$\mathbf{z}[n]$ in equation (4.15) into the second term in equation (4.20), we get $\mathbf{K}[n](\breve{\mathbf{H}}[n]\mathbf{s}[n] + \mathbf{w}[n] + (\mathbf{h}(\hat{\mathbf{s}}[n|n-1]) - \breve{\mathbf{H}}[n]\hat{\mathbf{s}}[n|n-1]) - \mathbf{h}(\hat{\mathbf{s}}[n|n-1]) = \breve{\mathbf{H}}[n](\mathbf{s}[n] - \hat{\mathbf{s}}[n|n-1]) + \mathbf{w}[n]$. Which proves that both the update equations for the linear and extended Kalman filter are the same.

4.2 Tracking Along Different Paths

In this section, the performance of the extended Kalman filter is evaluated for tracking a robot over different paths. Section 4.2.1 discusses the initialization of the extended Kalman filter parameters in equations (4.16), (4.18), (4.19), (4.20) and (4.21).

The analysis of the filter for the case where the robot moves over a deterministic path is shown in Section 4.2.2 and the extended Kalman filter analysis for tracking a robot over a random path is shown in Section 4.2.3. In both sections, the extended Kalman filter performance is compared with that of the dynamic CRLB in equation (3.46). It is assumed that the robot is constrained within a 100 m by 100 m square area. This size is typical to a warehouse or a factory where the robot would be deployed.

4.2.1 Simulation Initialization

Note that the extended Kalman filter initialization is similar to the dynamic CRLB initialization in Section 3.5.1.

Let the state vector be $\mathbf{s}[n] = [\, X[n]\ Y[n]\ v_X[n]\ v_Y[n]]^T$, where $X[n]$ and $Y[n]$ are X and Y position coordinates of the robot, and $v_X[n]$ and $v_Y[n]$ are the X and Y velocity components.

The matrix $\mathbf{A}$ defines the robot kinematics with respect to the state vector $\mathbf{s}[n]$, as

shown in equation (4.22). It is a constant velocity motion model.

$$\mathbf{A} = \begin{bmatrix} 1 & 0 & \Delta t & 0 \\ 0 & 1 & 0 & \Delta t \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \tag{4.22}$$

where Δt is the interval at which observation vectors are captured. Assuming the robot speed is fixed at 0.2 m/s and the sampling distance is equal to 4 m, $\Delta t = 4/0.2$=20 sec.

The process noise $\mathbf{u}[n]$ in equation (4.1) is given by

$$\mathbf{u}[n] = [0 \;\; 0 \;\; u_X[n] \; u_Y[n]]^T \tag{4.23}$$

where $u_X[n] = v_X[n] - v_X[n-1]$, and $u_Y[n] = v_Y[n] - v_Y[n-1]$. The values of $u_X[n]$ and $u_Y[n]$ is equal to the maximum possible change in velocity in the X and Y directions respectively, and this is equal to 0.4 m/s. This would occur if the robot stops and makes a 180^0 turn. The covariance matrix $\mathbf{Q}$ of the process noise $\mathbf{u}[n]$ is given by

$$\mathbf{Q} = \begin{bmatrix} 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & \sigma_{Vx}^2 & 0 \\ 0 & 0 & 0 & \sigma_{Vy}^2 \end{bmatrix} \tag{4.24}$$

where the value of σ_{Vx}^2 and σ_{Vy}^2 equal 0.4^2 m^2/s^2. Similar to Section 3.4.2, the X and Y variances are assumed independent since the error distorting the velocity is equally probable to happen in any direction.

Let the number of beacons be m. The beacons coordinates are

$(X_{b,k}, Y_{b,k})$ $k = 1, .., m.$

The measurement vector, $\mathbf{h}(\mathbf{s}[n])$ in equation (4.9) represents the DR and RSS measurements performed by the robot such that

$$\mathbf{h}(\mathbf{s}[n]) = \begin{bmatrix} X[n] \\ Y[n] \\ 10\eta \log\left(\sqrt{(X[n]-X_{b,1})^2+(Y[n]-Y_{b,1})^2}\right) \\ \vdots \\ 10\eta \log\left(\sqrt{(X[n]-X_{b,m})^2+(Y[n]-Y_{b,m})^2}\right) \end{bmatrix}_{(m+2)\times 1} \tag{4.25}$$

and η is the path loss exponent discussed in Section 3.3. The DR measurements are equal to the desired robot position and the RSS measurements are given by equation (3.2).

The matrix $\breve{\mathbf{H}}[n]$ represents the Jacobian of $\mathbf{h}(\mathbf{s}[n])$, evaluated at the predicted value of the state vector $\hat{\mathbf{s}}[n|n-1]$, such that $\breve{\mathbf{H}}[n] = (\nabla_{\mathbf{s}[n]}\mathbf{h}^T(\mathbf{s}[n]))^T$. And since the predicted value $\hat{\mathbf{s}}[n|n-1] = [\,\widehat{X}[n]\ \widehat{Y}[n]\ \widehat{v_X}[n]\ \widehat{v_Y}[n]]^T$ is a 4×1 vector, and the number of measurements is m, the observation matrix, $\breve{\mathbf{H}}[n]$ is such that

$$\breve{\mathbf{H}}[n] = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ \frac{10\eta}{\ln 10}\frac{(\widehat{X}[n]-X_{b,1})}{(\widehat{X}[n]-X_{b,1})^2+(\widehat{Y}[n]-Y_{b,1})^2} & \frac{10\eta}{\ln 10}\frac{(\widehat{Y}[n]-Y_{b,1})}{(\widehat{X}[n]-X_{b,1})^2+(\widehat{Y}[n]-Y_{b,1})^2} & 0 & 0 \\ \vdots & \vdots & \vdots & \vdots \\ \frac{10\eta}{\ln 10}\frac{(\widehat{X}[n]-X_{b,m})}{(\widehat{X}[n]-X_{b,m})^2+(\widehat{Y}[n]-Y_{b,m})^2} & \frac{10\eta}{\ln 10}\frac{(\widehat{Y}[n]-Y_{b,m})}{(\widehat{X}[n]-X_{b,m})^2+(\widehat{Y}[n]-Y_{b,m})^2} & 0 & 0 \end{bmatrix}_{(m+2)\times 4} \tag{4.26}$$

The covariance matrix of the observation noise, $\mathbf{C}[n]$ is given by

$$\mathbf{C}[n] = \begin{bmatrix} \sigma_X^2[n] & 0 & 0 & 0 & \dots & 0 \\ 0 & \sigma_Y^2[n] & 0 & 0 & \dots & 0 \\ 0 & 0 & \sigma_R^2 & 0 & \dots & 0 \\ 0 & 0 & 0 & \ddots & \dots & \vdots \\ \vdots & \vdots & \vdots & \ddots & \ddots & 0 \\ 0 & 0 & 0 & \dots & 0 & \sigma_R^2 \end{bmatrix}_{(m+2)\times(m+2)} \tag{4.27}$$

where σ_R^2 represents the shadowing variance. Note that the values of $\sigma_X^2[n]$ and $\sigma_Y^2[n]$ are a function of time n since they increase as the mobile moves. These values are calculated using equation (2.6) in Chapter 2.

Let the robot start moving from the point with coordinates $(1, 1)$. Assume the initial position of the robot is known, therefore the initial desired state vector at time 0, is equal to the actual state vector $\mathbf{s}[0]$. The extended Kalman filter state estimate $\widehat{\mathbf{s}}[0]$ is initialized to the actual state vector such that

$$\mathbf{s}[0] = \widehat{\mathbf{s}}[0] = [1 \quad 1 \quad 0.1414 \; 0.1414]^T \tag{4.28}$$

The first two elements of the vector $\mathbf{s}[0]$ are the initial robot coordinates $(X[n], Y[n])$, initialized to (1,1). The third and fourth elements are the X and Y components of the robot's velocity. Since they are the equal, the robot will start off with a heading of 45^o with respect to the X axis.

The actual path of the robot takes into account the DR error corrupting the X and Y position. At the start of every segment length of the desired path, random angle and distance errors are added to the actual position using the transformation error [48].

In a deterministic system there is no process noise, and since a constant velocity state

model is assumed, the deterministic system will only work for a straight line trajectory. In order to overcome this problem, an equivalent system that allows tracking over paths which have different directions, would require a process noise covariance matrix, $\mathbf{Q}$ as given in equation (4.24). The values of σ_{Vx}^2 and σ_{Vy}^2 in $\mathbf{Q}$ is still 0.4 m^2/s^2. In essence, the problem of tracking the deterministic path is a stochastic problem.

For both the deterministic and random paths the extended Kalman filter then incorporates the DR measurements and RSS measurements into the filter from the 9 beacons every 4 m. The beacon configuration is shown in Figure 3.1. The initial MSE matrix $\mathbf{M}[0|0]$ for both the deterministic and random paths is initialized to $10{,}000\mathbf{I}$, where $\mathbf{I}$ is the identity matrix. A large initial MSE value is chosen so as not to bias the extended Kalman filter. In a practical system the extended Kalman filter would be initialized to the process noise covariance matrix, $\mathbf{Q}$ in equation (4.24), since the process noise models the error in the robot motion. Assume a path loss exponent $\eta = 3$ [45] and a shadowing standard deviation typical of indoor scenario of 2 dB [53].

4.2.2 Deterministic Path

In this section, the performance of the extended Kalman filter is evaluated based on tracking the robot over a known deterministic path made up of 20 segments as shown in Figure 4.1.

In this thesis, the desired path denotes the path that is programmed into the robot as described in Section 4.2.1. The actual path represents the desired path corrupted by the DR errors, and the extended Kalman filter is estimating the actual path.

The plot showing the extended Kalman filter tracking performance for a single iteration using the DR/RSS system is shown in Figure 4.2.

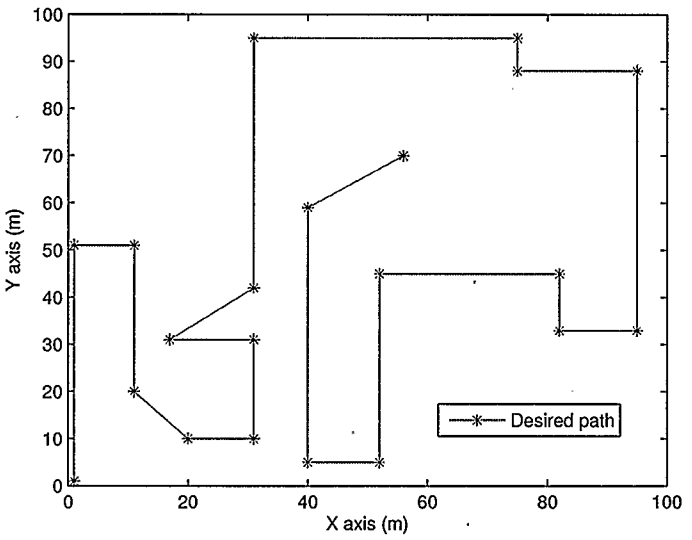

Figure 4.1: Desired path.

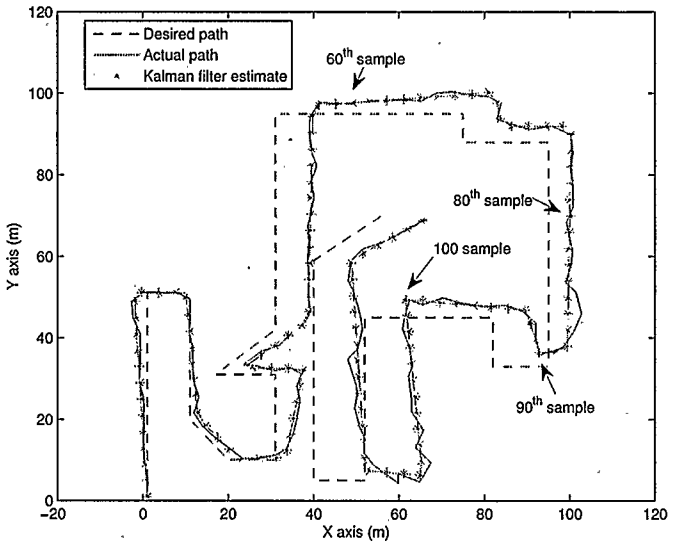

Figure 4.2: Kalman estimate versus the desired position.

Figures 4.3 and 4.4 show the MSE in the X and Y directions respectively versus the number of segments moved. The MSE plots is the average of 1000 iterations. The extended Kalman filter MSE is calculated in two different ways. The first way is by using the square of the difference between the actual path and the extended Kalman filter estimate. For a single random path, the squared error is calculated for each sample point. The total length of the path is 528 m and since samples are taken in increments of 4 m, we have 132 squared error sample values for one path. The squared error for each sample point is averaged over 1000 paths. The second way is by using the extended Kalman filter MSE $\mathbf{M}[n|n]$ in equation (4.21). The value of $\mathbf{M}[n|n]$ is also calculated for each sample point along the path, and then the MSE is averaged over 1000 paths. Note that the MSE in the X and Y direction is always increasing since

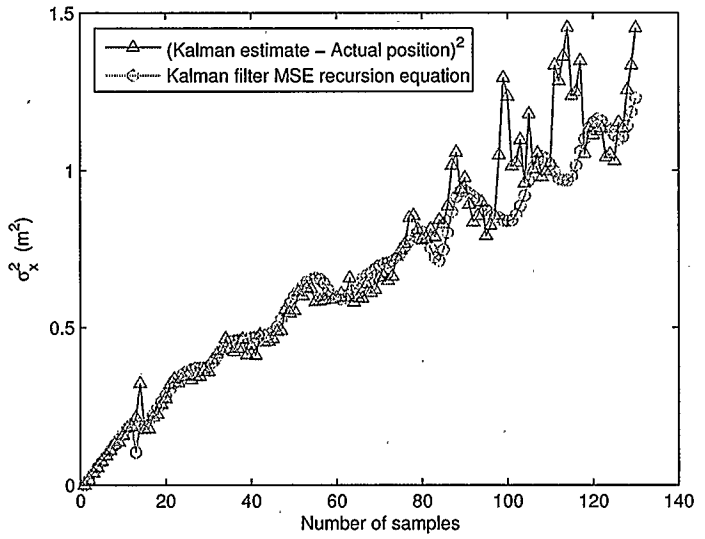

Figure 4.3: Extended Kalman filter MSE in the X-direction versus the number of segments moved.

the uncertainty of the robot increases as the robot travels more segments. Note also that the MSE curve calculated using the extended Kalman filter recursive equation is

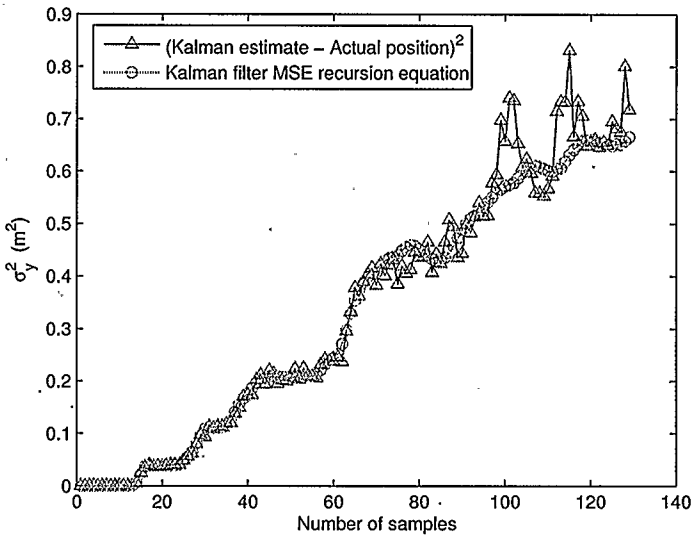

Figure 4.4: Extended Kalman filter MSE in the Y-direction versus the number of segments moved.

more smooth than that calculated using the squared difference between the actual and extended Kalman estimate. This is because the extended Kalman filter equations are designed to average the error. Averaging the squared difference between the actual and extended Kalman estimate over a large number of iterations shall eventually make the estimate close to that predicted by the extended Kalman filter equations.

Note that the MSE in the Y direction for the first 15 sample points in Figure 4.4 is almost zero. This is because the robot is moving in the Y direction, as shown in Figure 4.2 when going from point (1,1) to (1,51). In this case only the distance DR error is corrupting the robot position. However, the MSE in the X direction in Figure 4.3 is increasing steadily, since the robot starts off pointing in the positive X direction at coordinates (1,1), it makes a 90^0 turn counter clock-wise and then moves in the Y direction. Therefore the DR error corrupting the X axis is due to distance and angle error.

Note that the MSE in the Y direction in Figure 4.4 rises steadily at around sample number 60, this area denotes the region in Figure 4.2 where the robot makes a 90^0 turn and moves solely in the X direction. The MSE in the Y direction then decreases from sample number 80 to 90, this is because it is the region where the robot moves solely in the Y direction.

Note that the ylabel on Figures 4.3 and 4.4 denote the variance σ_x^2 and σ_y^2 respectively. The word variance and MSE are used interchangeably, since the variance is equivalent to the MSE of an unbiased estimator [49]. The dynamic CRLB expression in equation (3.46) is shown to be equivalent to the linear Kalman filter MSE recursive equation (4.7) in Appendix A.2.

Next the MSE is plotted versus the shadow standard deviation. Figures 4.5 and 4.6 show a comparison between the extended Kalman filter MSE and the CRLB in the X and Y direction respectively. Each point on the graphs is the average of 1000 iterations. The extended Kalman filter MSE is calculated as the squared difference between the desired position and the actual position of the extended Kalman filter. The CRLB is calculated using equation (3.46).

From the plots in Figures 4.5 and 4.6, note that the variance increases steadily when the shadow standard deviation is less than 4 dB. This is because at a low shadow standard deviation, the channel is good, and the extended Kalman filter weights the RSS values highly compared to the DR measurements. However as the channel gets worse, the shadowing standard deviation increases and the RSS measurements do not add any useful information of the robot estimate. Thus, the estimator ignores the erroneous RSS values, and places more emphasis on the DR measurement, making the estimate levels off converging to the DR position estimate.

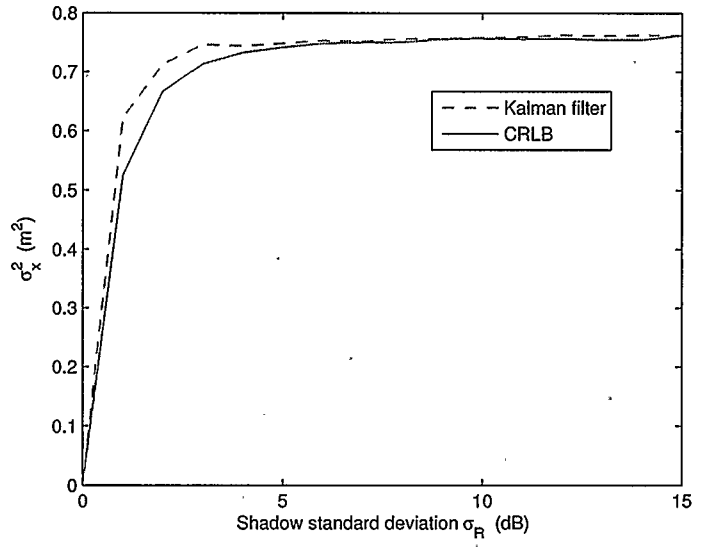

Figure 4.5: Positioning accuracy in the X-direction versus the shadowing standard deviation.

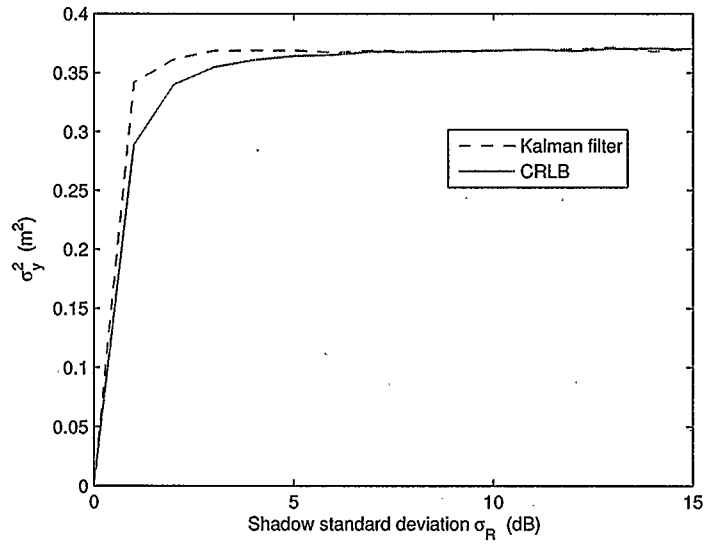

Figure 4.6: Positioning accuracy in the Y-direction versus the shadowing standard deviation.

Note also that the extended Kalman filter converges to the CRLB at high shadow, since the DR measurements are linear and are modeled as white Gaussian.

Next, the performance of the extended Kalman filter is studied as a function of the sampling distance. Figures 4.7 and 4.8 show the average MSE of the extended Kalman filter in the X and Y directions respectively, versus the sampling distance, after traveling over the deterministic path shown in Figure 4.1. Since the total length of the path

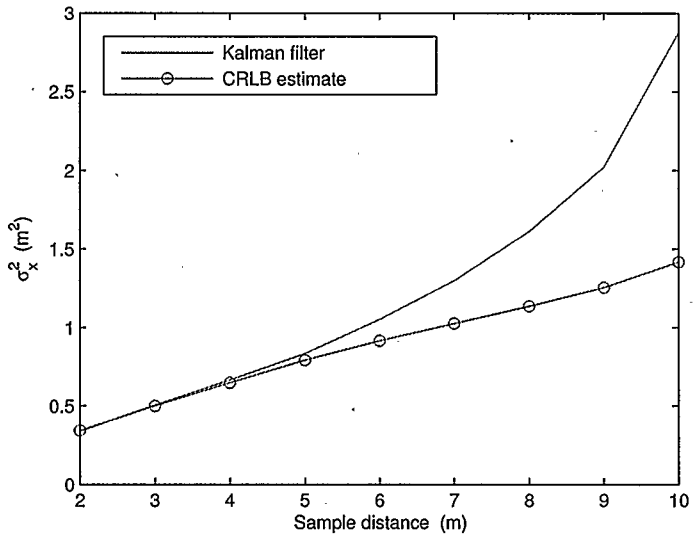

Figure 4.7: Positioning accuracy in the X directions versus the sample distance.

is 528 m and the sample distance varies from 2-10 m, then the number of samples is calculated as the length of the path/sample distance.

Note that when taking samples at a distance less than 5 m, the performance of the extended Kalman filter and the CRLB are almost the same. However, when sampling at larger distances the extended Kalman filter seems to diverge from the CRLB. This is possibly because at smaller distances the DR error is tolerable, and the extended Kalman filter can minimize the MSE efficiently. However, at larger sample distances

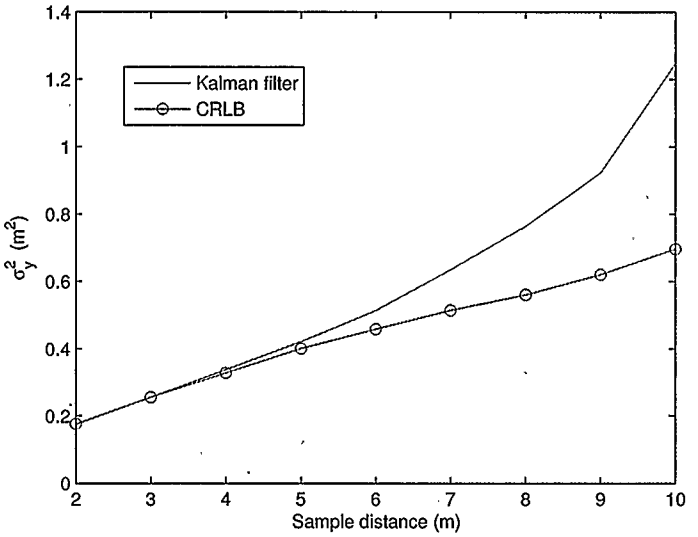

Figure 4.8: Positioning accuracy in the Y directions versus the sample distance.

more DR error is incurred and thus the extended Kalman filter estimate worsens.

4.2.3 Random Path

In this section, the performance of the extended Kalman filter is studied when tracking a robot over a random path. The random path consists of segments, where each segment length varies uniformly from 5 to 40 meters, the angle varies uniformly from 0 to 2π, and the whole path is constrained within a 100 m by 100 m area.

As in Section 3.5.1 since the maximum battery of iRobot is 1.5 hours, and the robot takes 20 sec to go over a path of length 4 m moving at 0.2 m/s, then the maximum number of samples along one random path is 1.5 hours / 20 sec = 270 samples.

Figure 4.9 shows a portion of a realization of the desired path, actual path and extended Kalman filter estimate. Figures 4.10 and 4.11 show the extended Kalman filter

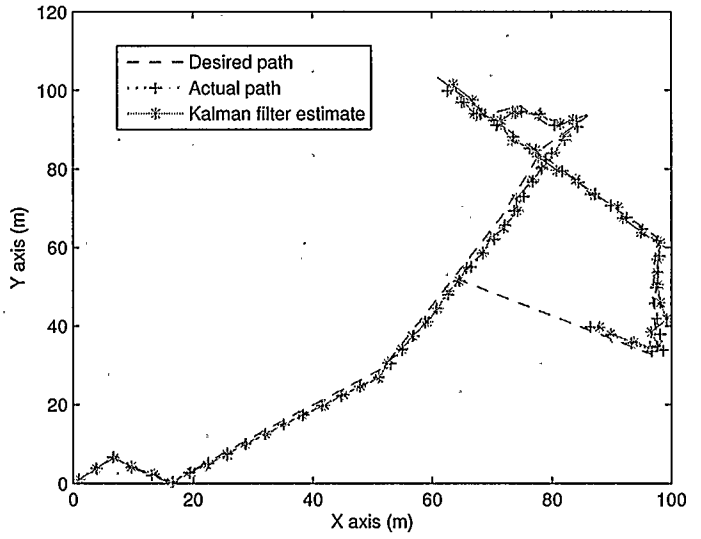

Figure 4.9: Kalman estimate versus the desired position.

MSE plot for the estimate of the X and Y coordinates respectively, versus the number of samples taken. Samples are taken in increments of 4 m. Similar to Section 4.2.2, the extended Kalman filter MSE is calculated in two different ways. The first way is by

using the square of the difference between the actual path and the extended Kalman filter estimate. For a single random path, the MSE is calculated for each sample point along the path, so at the end of one path we would have 270 MSE values. The MSE for each sample point is then averaged over 1000 random paths. The second way is by using the extended Kalman filter MSE $\mathbf{M}[n|n]$ in equation (4.21), where the value of $\mathbf{M}[n|n]$ is calculated for each sample point along the path, and then the MSE is averaged over 1000 random paths.

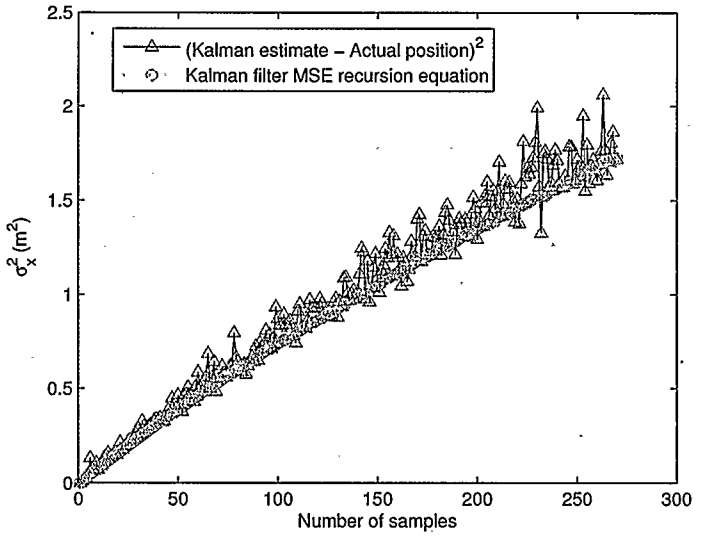

Figure 4.10: Extended Kalman filter MSE in the X direction versus the number of segments moved.

As expected, the MSE is increasing in the X and Y, since the more samples moved, the more DR error corrupting the robot position and thus the higher the MSE estimate. Averaging the squared error over a larger number of random path, will make the MSE calculated using the squared difference of the actual and Kalman estimate be equal to that of the extended Kalman filter equations.

Next, the extended Kalman filter MSE is studied as a function of the shadowing

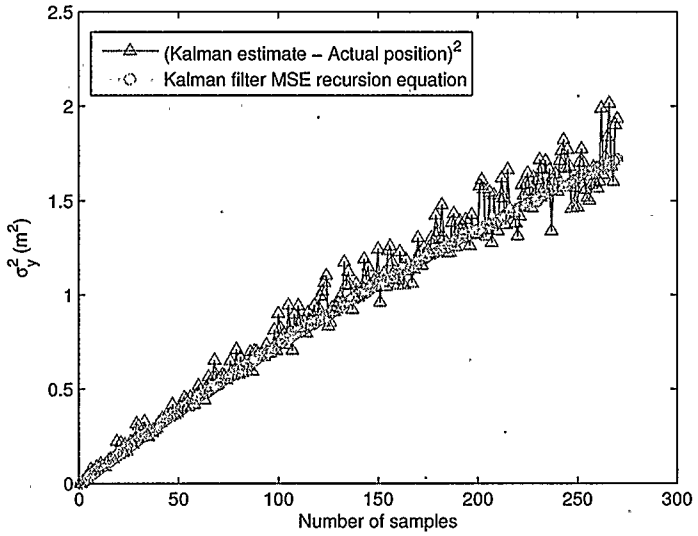

Figure 4.11: Extended Kalman filter MSE in the Y direction versus the number of segments moved.

standard deviation. Figures 4.12 and 4.13 show the average MSE of the extended Kalman filter and the average CRLB in the X and Y direction respectively, versus the shadowing standard deviation.

The extended Kalman filter MSE for Figures 4.12 and 4.13 is calculated as the square of the difference between the actual and the desired position. The CRLB is calculated using the recursion equation (3.46) where the initial FIM is initialized to the inverse of the initial MSE matrix of the extended Kalman filter, **M**. Each point on the graphs is the average of 1000 random paths.

Note that the CRLB is lower bounding the extended Kalman filter MSE. This shows that the extended Kalman filter simulation is unbiased. In this case, the MSE of the estimate is equal to the variance of that estimate, and thus the Y label in Figures 4.12 and 4.13 are labeled σ_x^2 and σ_y^2. At a low shadowing standard deviation, the extended Kalman filter gain acts like a weighting function, it ignores (weights less)

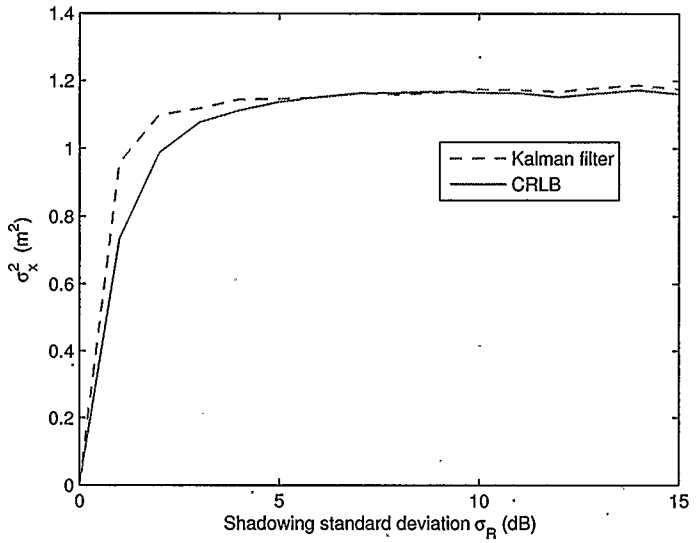

Figure 4.12: Positioning accuracy in the X-direction versus the shadowing standard deviation.

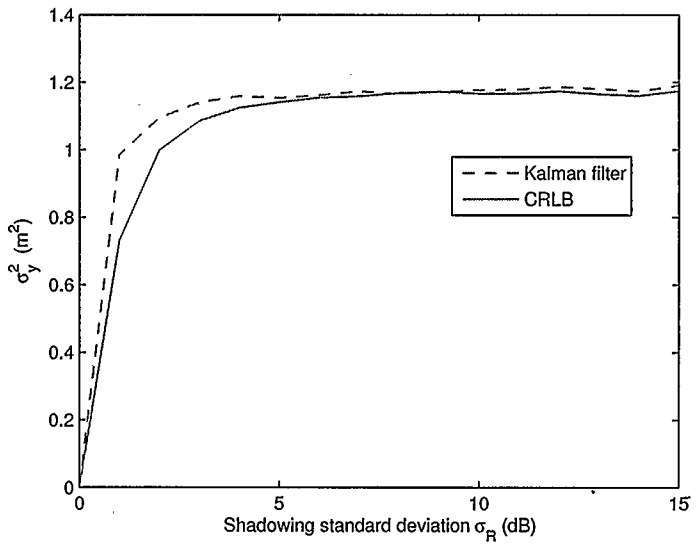

Figure 4.13: Positioning accuracy in the Y-direction versus the shadowing standard deviation.

the erroneous DR values and weights in favor of the RSS values. As the shadowing standard deviation increases, the channel becomes bad, the extended Kalman filter ignores the erroneous RSS measurements and accepts the DR measurements into the filter. The variance flattens out at high shadow since the RSS information do not add any improvement over the DR estimates. The close agreement between the extended Kalman filter and the CRLB plot at high shadowing standard deviation, is thus a verification of the Gaussian DR error model.

So far, the robot performs RSS measurements at a fixed sampling distance of 4 m. The extended Kalman filter performance is studied as the sampling distance varies. Figure 4.14 shows the average MSE of the extended Kalman filter in the X direction, versus the sampling distance. Similarly, Figure 4.15 shows the average MSE of the extended Kalman filter in the Y direction, versus the sampling distance.

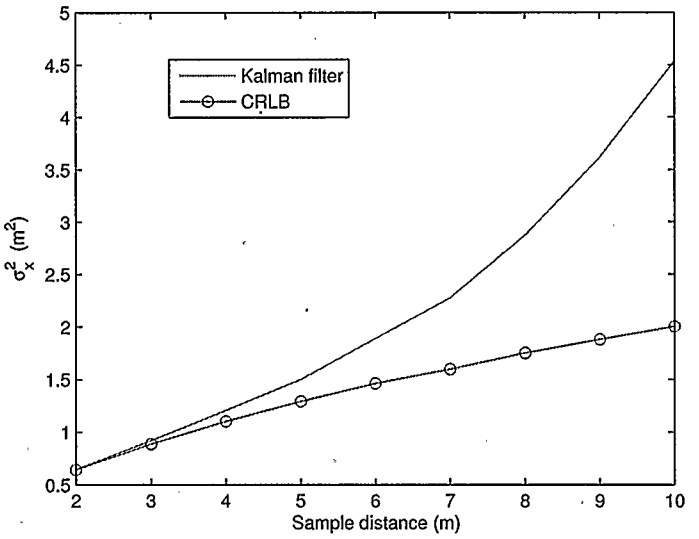

Figure 4.14: Positioning accuracy in the X-directions versus the sample distance.

Knowing that the maximum number of samples, when taking samples every 4 m is

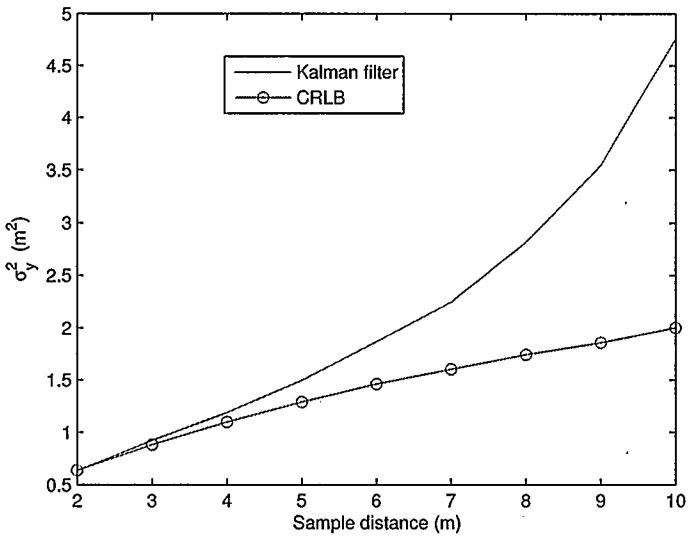

Figure 4.15: Positioning accuracy in the Y-directions versus the sample distance.

270, the total length to be traveled by the robot is 270 $\times$ 4= 1080 m. Therefore when the sample distance varies from 2-10 m, the number of samples is given as the total length of the path/sample distance.

Similar to Section 4.2.2, the extended Kalman filter MSE increases as the DR error builds up due to moving at larger segment length.

Chapter 5

Conclusion

Concluding remarks are made in this section. Section 5.1 provides a summary of the results of the thesis and Section 5.2 talks about the future work.

5.1 Summary

In this thesis, a Gaussian dead reckoning (DR) error model is developed and then verified based on empirical measurements with iRobot Create. The procedure for coming up with the model is illustrated in Chapter 2 in Section 2.2.1. This procedure can be generalized to any robot platform that rotates in place and then moves in a straight line. The error model developed is valuable since it validates the Gaussian error assumption found in the literature. The error model can also be extended to non-zero mean error distributions.

When coming up with the Gaussian model assumption, I learned some useful tools. I learned how to generate random numbers from the empirical distribution characterizing the robot's angle and distance error measurements, so that I can incorporate the empirical measurements into the simulations. I also learned how to use the QQ-plot as a means to verify the Gaussian DR error assumption.

A theoretical bound, known as the Cramer Rao lower bound (CRLB), is presented in Chapter 3 for static and dynamic systems.

I learned that a recursive CRLB for dynamic systems exist. This result is powerful

as it can be used against other estimators to assess their performance.

The performance of the static and the dynamic CRLB were studied for different scenarios. The performance of the dynamic CRLB always outperforms the static case, due to the fact that the dynamic CRLB uses previous observations to predict the current position estimate.

A practical filter, known as the extended Kalman filter, commonly used for tracking, is presented in Chapter 4 to track the robot over random and deterministic paths, while incorporating both DR and RSS range measurements. The dynamic CRLB derived in Appendix A is compared to the extended Kalman filter simulations against different system parameters. The plots comparing the CRLB and the extended Kalman filter MSE show that the extended Kalman filter simulation is unbiased, since the CRLB curve is lower bounding the extended Kalman filter MSE curve. The close agreement between the CRLB and the extended Kalman filter MSE curve, at a high shadow standard deviation, is a further verification of the Gaussian DR error model.

The extended Kalman filter MSE is studied at different sample intervals to determine the optimal sampling interval distance. Sampling at the lowest distance results in the lowest MSE since the robot would use RSS measurements more often to correct its position, compared to sampling at larger distances. The optimal sampling distance depends on the application in use. It is a trade-off between the required MSE accuracy and the robot resources.

5.2 Future Work

In this thesis, we came up with a novel DR error model. This DR model could be incorporated with other localization schemes such as TOA and AOA measurements.

We have assumed that we have only one robot that is trying to figure out its position using the dead reckoning information from the wheel encoders and the RSS information from the stationary beacons. An interesting extension of this work would be co-operative localization, where a number of robots would cooperate together to estimate their position. That is the robots would estimate their position relative to one another using the RSS range measurements.

One of the challenges of this problem is that the positions of the robots are relative to one another and thus a problem we face would be how to convert this relative position into absolute position coordinates given all the uncertainties and complexity of the problem.

Appendix A

Dynamic CRLB

A.1 Dynamic CRLB Derivation

The following is a summary of the derivation of equation (3.30) as presented in [55]. Assuming a first order Markov process, the $\mathcal{P}(\mathbf{S}[n+1], \mathbf{Z}[n+1])$ can be written in terms of the previous time sample as

$$\mathcal{P}(\mathbf{S}[n+1], \mathbf{Z}[n+1]) = \mathcal{P}(\mathbf{S}[n], \mathbf{Z}[n])\mathcal{P}(\mathbf{s}[n+1]|\mathbf{s}[n])\mathcal{P}(\mathbf{z}[n+1]|\mathbf{s}[n+1]) \quad \text{(A.1)}$$

Since equation (3.30), shows a recursive FIM of the state vector $\mathbf{s}[n+1]$ based on the previous state $\mathbf{s}[n]$, let the vector $\mathbf{S}[n+1]$ denote the trajectory of the state such that $\mathbf{S}[n+1] = [\mathbf{S}[n-1]^T \;\; \mathbf{s}[n]^T \;\; \mathbf{s}[n+1]^T]^T_{(n+1)r\times 1}$

where $\mathbf{S}[n-1]$ denotes the previous states up to time $n-1$, $\mathbf{s}[n]$ is the current state at time n, and $\mathbf{s}[n+1]$ is the future state at time $n+1$.

The FIM, $\mathbf{I}(\mathbf{S}[n+1])$ is written in matrix form as

$$\mathbf{I}(\mathbf{S}[n+1]) = \begin{bmatrix} \nu_{n+1} & \xi_{n+1} & \chi_{n+1} \\ \xi^T_{n+1} & \gamma_{n+1} & \omega_{n+1} \\ \chi^T_{n+1} & \omega^T_{n+1} & \kappa_{n+1} \end{bmatrix}_{(n+1)r\times(n+1)r} \quad \text{(A.2)}$$

where ν_{n+1}, ξ_{n+1}, χ_{n+1}, γ_{n+1}, ω_{n+1} and κ_{n+1} represent submatrix blocks and are

defined as

$$\begin{aligned}
\nu_{n+1} &= -E\left\{\nabla_{\mathbf{S}[n-1]}\left(\nabla_{\mathbf{S}[n-1]}\ln\mathcal{P}(\mathbf{S}[n+1],\mathbf{Z}[n+1])\right)^T\right\}_{(n-1)r\times(n-1)r} \\
\xi_{n+1} &= -E\left\{\nabla_{\mathbf{s}[n]}\left(\nabla_{\mathbf{S}[n-1]}\ln\mathcal{P}(\mathbf{S}[n+1],\mathbf{Z}[n+1])\right)^T\right\}_{(n-1)r\times r} \\
\chi_{n+1} &= -E\left\{\nabla_{\mathbf{s}[n+1]}\left(\nabla_{\mathbf{S}[n-1]}\ln\mathcal{P}(\mathbf{S}[n+1],\mathbf{Z}[n+1])\right)^T\right\}_{(n-1)r\times(n+1)r} \\
\gamma_{n+1} &= -E\left\{\nabla_{\mathbf{s}[n]}\left(\nabla_{\mathbf{s}[n]}\ln\mathcal{P}(\mathbf{S}[n+1],\mathbf{Z}[n+1])\right)^T\right\}_{r\times r} \\
\omega_{n+1} &= -E\left\{\nabla_{\mathbf{s}[n+1]}\left(\nabla_{\mathbf{s}[n]}\ln\mathcal{P}(\mathbf{S}[n+1],\mathbf{Z}[n+1])\right)^T\right\}_{r\times(n+1)r} \\
\kappa_{n+1} &= -E\left\{\nabla_{\mathbf{s}[n+1]}\left(\nabla_{\mathbf{s}[n+1]}\ln\mathcal{P}(\mathbf{S}[n+1],\mathbf{Z}[n+1])\right)^T\right\}_{(n+1)r\times(n+1)r}
\end{aligned} \tag{A.3}$$

Similar to equation (3.24) in Chapter 3, ν_{n+1} defines the FIM for estimating $\mathbf{S}[n-1]$, γ_{n+1} is the FIM for estimating $\mathbf{s}[n]$, κ_{n+1} is the FIM for estimating $\mathbf{s}[n+1]$, ξ_{n+1} and ξ_{n+1}^T are the FIM defining the correlation between $\mathbf{S}[n-1]$ and $\mathbf{s}[n]$, χ_{n+1} and χ_{n+1}^T are the FIM defining the correlation between $\mathbf{S}[n-1]$ and $\mathbf{s}[n+1]$, and ω_{n+1} and ω_{n+1}^T are the FIM defining the correlation between $\mathbf{s}[n]$ and $\mathbf{s}[n+1]$.

The goal is to find the FIM of the estimate of $\mathbf{s}[n+1]$. Since $\mathbf{s}[n+1]$ is the last element in the vector $\mathbf{S}[n+1]$, therefore the CRLB is the inverse of the right lower submatrix $\mathbf{I}(\mathbf{S}[n+1])$ in equation (A.2). Once the CRLB is found, the FIM is simply the inverse of that CRLB value.

Next the parameters of matrix $\mathbf{I}(\mathbf{S}[n+1])$ in equation (A.2) are evaluated. Starting with ν_{n+1} given by

$$\nu_{n+1} = -E\left\{\nabla_{\mathbf{S}[n-1]}\left(\nabla_{\mathbf{S}[n-1]}\ln\mathcal{P}(\mathbf{S}[n+1],\mathbf{Z}[n+1])\right)^T\right\} \tag{A.4}$$

Expanding $\mathcal{P}(\mathbf{S}[n+1],\mathbf{Z}[n+1])$ using equation (A.1) ν_{n+1} becomes

$$\begin{aligned}\nu_{n+1} &= -E\left\{\nabla_{\mathbf{S}[n-1]}\left(\nabla_{\mathbf{S}[n-1]}(\ln\mathcal{P}(\mathbf{S}[n],\mathbf{Z}[n])+\ln\mathcal{P}(\mathbf{s}[n+1]|\mathbf{s}[n]))\right)^T\right\}\\ &\quad - E\left\{\nabla_{\mathbf{S}[n-1]}\left(\nabla_{\mathbf{S}[n-1]}\ln\mathcal{P}(\mathbf{z}[n+1]|\mathbf{s}[n+1])\right)^T\right\}\\ &= -E\left\{\nabla_{\mathbf{S}[n-1]}\left(\nabla_{\mathbf{S}[n-1]}\ln\mathcal{P}(\mathbf{S}[n],\mathbf{Z}[n])\right)^T\right\}+0+0\\ &= \nu_n\end{aligned} \tag{A.5}$$

The expression ν_{n+1} is equal to ν_n in equation (3.24) in Chapter 3. The last two terms in equation (A.5) are zero since $\ln\mathcal{P}(\mathbf{s}[n+1]|\mathbf{s}[n])$ and $\ln\mathcal{P}(\mathbf{z}[n+1]|\mathbf{s}[n+1])$ are independent of the previous trajectory of the state vector $\mathbf{S}[n-1]$.

The submatrix ξ_{n+1} is given by

$$\xi_{n+1} = -E\left\{\nabla_{\mathbf{s}[n]}\left(\nabla_{\mathbf{S}[n-1]}\ln\mathcal{P}(\mathbf{S}[n+1],\mathbf{Z}[n+1])\right)^T\right\} \tag{A.6}$$

Expanding $\mathcal{P}(\mathbf{S}[n+1],\mathbf{Z}[n+1])$ using equation (A.1) ξ_{n+1} becomes

$$\begin{aligned}\xi_{n+1} &= -E\left\{\nabla_{\mathbf{s}[n]}\left(\nabla_{\mathbf{S}[n-1]}(\ln\mathcal{P}(\mathbf{S}[n],\mathbf{Z}[n])+\ln\mathcal{P}(\mathbf{s}[n+1]|\mathbf{s}[n]))\right)^T\right\}\\ &\quad - E\left\{\nabla_{\mathbf{s}[n]}\left(\nabla_{\mathbf{S}[n-1]}\ln\mathcal{P}(\mathbf{z}[n+1]|\mathbf{s}[n+1])\right)^T\right\}\\ &= -E\left\{\nabla_{\mathbf{s}[n]}\left(\nabla_{\mathbf{S}[n-1]}\ln\mathcal{P}(\mathbf{S}[n],\mathbf{Z}[n])\right)^T\right\}+0+0\\ &= \xi_n\end{aligned} \tag{A.7}$$

The expression ξ_{n+1} is equal to ξ_n in equation (3.24) in Chapter 3. The last two terms in equation (A.7) are zero since $\ln\mathcal{P}(\mathbf{s}[n+1]|\mathbf{s}[n])$ and $\ln\mathcal{P}(\mathbf{z}[n+1]|\mathbf{s}[n+1])$ are independent of the previous trajectory of the state vector $\mathbf{S}[n-1]$.

The submatrix χ_{n+1} is given by

$$\chi_{n+1} = -E\left\{\nabla_{\mathbf{s}[n+1]}\left(\nabla_{\mathbf{S}[n-1]}\ln\mathcal{P}(\mathbf{S}[n+1],\mathbf{Z}[n+1])\right)^T\right\} \tag{A.8}$$

Expanding $\mathcal{P}(\mathbf{S}[n+1], \mathbf{Z}[n+1])$ using equation (A.1) χ_{n+1} becomes

$$\begin{aligned}\chi_{n+1} &= -E\left\{\nabla_{\mathbf{s}[n+1]}\left(\nabla_{\mathbf{S}[n-1]}(\ln \mathcal{P}(\mathbf{S}[n], \mathbf{Z}[n]) + \ln \mathcal{P}(\mathbf{s}[n+1]|\mathbf{s}[n]))\right)^T\right\} \\ &\quad - E\left\{\nabla_{\mathbf{s}[n+1]}\left(\nabla_{\mathbf{S}[n-1]} \ln \mathcal{P}(\mathbf{z}[n+1]|\mathbf{s}[n+1])\right)^T\right\} \\ &= 0\end{aligned} \tag{A.9}$$

The first term in equation (A.9) $\ln \mathcal{P}(\mathbf{S}[n], \mathbf{Z}[n])$ is zero since it is independent of $\mathbf{s}[n+1]$. The last two terms are zero since $\ln \mathcal{P}(\mathbf{s}[n+1]|\mathbf{s}[n])$ and $\ln \mathcal{P}(\mathbf{z}[n+1]|\mathbf{s}[n+1])$ are independent of the previous trajectory of the state vector $\mathbf{S}[n-1]$.

The submatrix γ_{n+1} is given by

$$\gamma_{n+1} = -E\left\{\nabla_{\mathbf{s}[n]}\left(\nabla_{\mathbf{s}[n]} \ln \mathcal{P}(\mathbf{S}[n+1], \mathbf{Z}[n+1])\right)^T\right\} \tag{A.10}$$

Expanding $\mathcal{P}(\mathbf{S}[n+1], \mathbf{Z}[n+1])$ using equation (A.1) γ_{n+1} becomes

$$\begin{aligned}\gamma_{n+1} &= -E\left\{\nabla_{\mathbf{s}[n]}\left(\nabla_{\mathbf{s}[n]}(\ln \mathcal{P}(\mathbf{S}[n], \mathbf{Z}[n]) + \ln \mathcal{P}(\mathbf{s}[n+1]|\mathbf{s}[n]))\right)^T\right\} \\ &\quad - E\left\{\nabla_{\mathbf{s}[n]}\left(\nabla_{\mathbf{s}[n]} \ln \mathcal{P}(\mathbf{z}[n+1]|\mathbf{s}[n+1])\right)^T\right\} \\ &= -E\left\{\nabla_{\mathbf{s}[n]}\left(\nabla_{\mathbf{s}[n]} \ln \mathcal{P}(\mathbf{S}[n], \mathbf{Z}[n])\right)^T\right\} - E\left\{\nabla_{\mathbf{s}[n]}\left(\nabla_{\mathbf{s}[n]} \ln \mathcal{P}(\mathbf{s}[n+1]|\mathbf{s}[n])\right)^T\right\} \\ &= \gamma_n + D_n^{11}\end{aligned} \tag{A.11}$$

The expression γ_{n+1} is equal to γ_n in equation (3.24) plus D_n^{11} in equation (3.31) in Chapter 3. The term $-E\left\{\nabla_{\mathbf{s}[n]}\left(\nabla_{\mathbf{s}[n]} \ln \mathcal{P}(\mathbf{z}[n+1]|\mathbf{s}[n+1])\right)^T\right\}$ in equation (A.11) is zero since $\ln \mathcal{P}(\mathbf{z}[n+1]|\mathbf{s}[n+1])$ is independent of the state vector $\mathbf{s}[n]$.

The submatrix ω_{n+1} is given by

$$\omega_{n+1} = -E\left\{\nabla_{\mathbf{s}[n+1]}\left(\nabla_{\mathbf{s}[n]} \ln \mathcal{P}(\mathbf{S}[n+1], \mathbf{Z}[n+1])\right)^T\right\} \tag{A.12}$$

Expanding $\mathcal{P}(\mathbf{S}[n+1], \mathbf{Z}[n+1])$ using equation (A.1) ω_{n+1} becomes

$$\begin{aligned}
\omega_{n+1} &= -E\left\{\nabla_{\mathbf{s}[n+1]}\left(\nabla_{\mathbf{s}[n]}(\ln \mathcal{P}(\mathbf{S}[n], \mathbf{Z}[n]) + \ln \mathcal{P}(\mathbf{s}[n+1]|\mathbf{s}[n]))\right)^T\right\} \\
&\quad - E\left\{\nabla_{\mathbf{s}[n+1]}\left(\nabla_{\mathbf{s}[n]} \ln \mathcal{P}(\mathbf{z}[n+1]|\mathbf{s}[n+1])\right)^T\right\} \\
&= 0 - E\left\{\nabla_{\mathbf{s}[n+1]}\left(\nabla_{\mathbf{s}[n]} \ln \mathcal{P}(\mathbf{s}[n+1]|\mathbf{s}[n])\right)^T\right\} + 0 \\
&= D_n^{12}
\end{aligned} \tag{A.13}$$

The term D_n^{12} is defined in equation (3.31) in Chapter 3. The first term in equation (A.13) $\ln \mathcal{P}(\mathbf{S}[n], \mathbf{Z}[n])$ goes to zero since it is independent of $\mathbf{s}[n+1]$. The last term is zero since $\ln \mathcal{P}(\mathbf{z}[n+1]|\mathbf{s}[n+1])$ is independent of the state vector $\mathbf{s}[n]$.

The submatrix κ_{n+1} is given by

$$\kappa_{n+1} = -E\left\{\nabla_{\mathbf{s}[n+1]}\left(\nabla_{\mathbf{s}[n+1]} \ln \mathcal{P}(\mathbf{S}[n+1], \mathbf{Z}[n+1])\right)^T\right\} \tag{A.14}$$

Expanding $\mathcal{P}(\mathbf{S}[n+1], \mathbf{Z}[n+1])$ using equation (A.1) κ_{n+1} becomes

$$\begin{aligned}
\kappa_{n+1} &= -E\left\{\nabla_{\mathbf{s}[n+1]}\left(\nabla_{\mathbf{s}[n+1]}(\ln \mathcal{P}(\mathbf{S}[n], \mathbf{Z}[n]) + \ln \mathcal{P}(\mathbf{s}[n+1]|\mathbf{s}[n]))\right)^T\right\} \\
&\quad - E\left\{\nabla_{\mathbf{s}[n+1]}\left(\nabla_{\mathbf{s}[n+1]} \ln \mathcal{P}(\mathbf{z}[n+1]|\mathbf{s}[n+1])\right)^T\right\} \\
&= -E\left\{\nabla_{\mathbf{s}[n+1]}\left(\nabla_{\mathbf{s}[n+1]} \ln \mathcal{P}(\mathbf{s}[n+1]|\mathbf{s}[n])\right)^T\right\} \\
&\quad - E\left\{\nabla_{\mathbf{s}[n+1]}\left(\nabla_{\mathbf{s}[n+1]} \ln \mathcal{P}(\mathbf{z}[n+1]|\mathbf{s}[n+1])\right)^T\right\} \\
&= D_n^{22}
\end{aligned} \tag{A.15}$$

Note that κ_{n+1} is equal to the expression D_n^{22} in equation (3.31) in Chapter 3. The first term in equation (A.15) $\ln \mathcal{P}(\mathbf{S}[n], \mathbf{Z}[n])$ goes to zero since it is independent of $\mathbf{s}[n+1]$.

Substituting the appropriate values of equations (A.5 - A.15) into matrix

$\mathbf{I}(\mathbf{S}[n+1])$ in equation (A.2), the FIM becomes

$$\mathbf{I}(\mathbf{S}[n+1]) = \begin{bmatrix} \nu_n & \xi_n & 0 \\ \xi_n^T & \gamma_n + D_n^{11} & D_n^{12} \\ 0 & D_n^{21} & D_n^{22} \end{bmatrix}_{(n+1)r \times (n+1)r} \tag{A.16}$$

Where each of the coefficients represent a submatrix block. As in Section 3.4.1, we want to find an expression for the FIM of the vector $\mathbf{s}[n+1]$ from the big vector $\mathbf{S}[n+1]$. This is done by finding the CRLB of $\mathbf{s}[n+1]$, and then finding the inverse of it to find that FIM expression.

Given the FIM, $\mathbf{I}(\mathbf{S}[n+1])$ in equation (A.16), The CRLB for estimating the vector $\mathbf{s}[n+1]$ denoted by, $\mathbf{I}(\mathbf{s}[n+1])^{-1}$ is found from the inverse of the lower right submatrix of the FIM, $\mathbf{I}(\mathbf{S}[n+1])$. From [56], the inverse of the right lower submatrix, can be written as follows

$$\begin{aligned} \mathbf{I}(\mathbf{s}[n+1])^{-1} &= \left(D_n^{22} - [0 \quad D_n^{21}] \begin{bmatrix} \nu_n & \xi_n \\ \xi_n^T & \gamma_n + D_n^{11} \end{bmatrix}^{-1} \begin{bmatrix} 0 \\ D_n^{12} \end{bmatrix} \right)^{-1} \\ &= \left(D_n^{22} - D_n^{21} [\gamma_n + D_n^{11} - \xi_n^T \nu_n^{-1} \xi_n]^{-1} D_n^{12} \right)^{-1} \end{aligned} \tag{A.17}$$

From equation (3.29) $\mathbf{I}(\mathbf{s}[n]) = \gamma_n - \xi_n^T \nu_n^{-1} \xi_n$. Therefore substituting the value $\mathbf{I}(\mathbf{s}[n])$ into equation (A.17), and inverting the whole equation, we get the recursive FIM, $\mathbf{I}(\mathbf{s}[n+1])$, such that

$$\mathbf{I}(\mathbf{s}[n+1]) = D_n^{22} - D_n^{21} (\mathbf{I}(\mathbf{s}[n]) + D_n^{11})^{-1} D_n^{12} \quad n \geq 0 \tag{A.18}$$

A.2 Dynamic CRLB And Linear Kalman Filter MSE

Starting from the FIM, $\mathbf{I}(\mathbf{s}[n+1])$ in equation (3.46), which states that

$$\mathbf{I}(\mathbf{s}[n]) = (\mathbf{Q}[n-1] + \mathbf{A}[n-1]\ \mathbf{I}(\mathbf{s}[n-1])^{-1} \mathbf{A}[n-1]^T)^{-1} + \mathbf{H}[n]^T \mathbf{C}[n]^{-1} \mathbf{H}[n] \tag{A.19}$$

where the first term denotes the state estimate from the state model and the second term denotes the state estimate from the observation model. Note that the first term inside the bracket is simply the prediction MSE matrix in equation (4.4) given by

$$\mathbf{M}[n|n-1] = \mathbf{A}[n-1]\mathbf{M}[n-1|n-1]\mathbf{A}[n-1]^T + \mathbf{Q}[n-1] \tag{A.20}$$

Therefore equation (A.19) can be expressed as

$$\mathbf{I}(\mathbf{s}[n]) = \mathbf{M}[n|n-1]^{-1} + \mathbf{H}[n]^T\mathbf{C}[n]^{-1}\mathbf{H}[n] \tag{A.21}$$

and we know that the variance or the MSE is the inverse of the FIM $\mathbf{I}(\mathbf{s}[n])$. Therefore $\mathbf{M}[n|n] = \mathbf{I}(\mathbf{s}[n])^{-1}$ such that

$$\mathbf{M}[n|n] = (\mathbf{M}[n|n-1]^{-1} + \mathbf{H}[n]^T\mathbf{C}[n]^{-1}\mathbf{H}[n])^{-1} \tag{A.22}$$

Applying the matrix inversion lemma [54] to equation (A.22), we get

$$\mathbf{M}[n|n] = \mathbf{M}[n|n-1] - \mathbf{M}[n|n-1]\mathbf{H}[n]^T(\mathbf{C}[n] + \mathbf{H}[n]\mathbf{M}[n|n-1]\mathbf{H}[n]^T)^{-1}\mathbf{H}[n]\mathbf{M}[n|n-1] \tag{A.23}$$

Note that the term $\mathbf{M}[n|n-1]\mathbf{H}[n]^T(\mathbf{C}[n] + \mathbf{H}[n]\mathbf{M}[n|n-1]\mathbf{H}[n]^T)^{-1}$ is equal to the Kalman gain in equation (4.5). Thus the MSE $\mathbf{M}[n|n]$ is given by

$$\begin{aligned}\mathbf{M}[n|n] &= \mathbf{M}[n|n-1] - \mathbf{K}[n]\mathbf{H}[n]\mathbf{M}[n|n-1] \\ &= (\mathbf{I} - \mathbf{K}[n]\mathbf{H}[n])\mathbf{M}[n|n-1]\end{aligned} \tag{A.24}$$

Bibliography

[1] iRobot Corporation, "irobot create open interface specification," 2006. www.irobot.com.

[2] X. Wang, X. Zhao, Z. Liang, and M. Tan, "Deploying a wireless sensor network on the coal mines," in *Networking, Sensing and Control, 2007 IEEE International Conference on*, pp. 324–328, April 2007.

[3] W. Xie, X. Zhang, and H. Chen, "Wireless sensor network topology used for road traffic," in *Wireless, Mobile and Sensor Networks, 2007. (CCWMSN07). IET Conference on*, pp. 285–288, Dec. 2007.

[4] S. Kundu, S. Roy, and A. Pal, "A power-aware wireless sensor network based bridge monitoring system," in *Networks, 2008. ICON 2008. 16th IEEE International Conference on*, pp. 1–7, Dec. 2008.

[5] J.-S. Lin and C.-Z. Liu, "A monitoring system based on wireless sensor network and an soc platform in precision agriculture," in *Communication Technology, 2008. ICCT 2008. 11th IEEE International Conference on*, pp. 101–104, Nov. 2008.

[6] B. Liu, P. Brass, O. Dousse, P. Nain, and D. Towsley, "Mobility improves coverage of sensor networks," in *MobiHoc '05: Proceedings of the 6th ACM international symposium on Mobile ad hoc networking and computing*, (New York, NY, USA), pp. 300–308, ACM, 2005.

[7] Y.-C. Wang, C.-C. Hu, and Y.-C. Tseng, "Efficient placement and dispatch of sensors in a wireless sensor network," *IEEE Transactions on Mobile Computing*,

vol. 7, pp. 262–274, Feb. 2008.

[8] X. Du and F. Lin, "Improving sensor network performance by deploying mobile sensors," in *Performance, Computing, and Communications Conference, 2005. IPCCC 2005. 24th IEEE International*, pp. 67–71, April 2005.

[9] W. Wang and Q. Zhu, "Rss-based monte carlo localisation for mobile sensor networks," *IET Communications*, vol. 2, pp. 673–681, May 2008.

[10] A. Verma, H. Sawant, and J. Tan, "Selection and navigation of mobile sensor nodes using a sensor network," in *Pervasive Computing and Communications, 2005. PerCom 2005. Third IEEE International Conference on*, pp. 41–50, March 2005.

[11] T. He, C. Huang, B. M. Blum, J. A. Stankovic, and T. F. Abdelzaher, "Range-free localization and its impact on large scale sensor networks," *ACM Trans. Embed. Comput. Syst.*, vol. 4, no. 4, pp. 877–906, 2005.

[12] N. Patwari, Hero, M. Perkins, N. S. Correal, and R. J. O'Dea, "Relative location estimation in wireless sensor networks," *Signal Processing, IEEE Transactions on*, vol. 51, no. 8, pp. 2137–2148, 2003.

[13] S. Shenoy and J. Tan, "Simultaneous localization and mobile robot navigation in a hybrid sensor network," in *Intelligent Robots and Systems, 2005. (IROS 2005). 2005 IEEE/RSJ International Conference on*, pp. 1636–1641, Aug. 2005.

[14] Y. Ohta, M. Sugano, and M. Murata, "Autonomous localization method in wireless sensor networks," in *Pervasive Computing and Communications Workshops, 2005.*

PerCom 2005 Workshops. Third IEEE International Conference on, pp. 379–384, March 2005.

[15] N. Patwari, R. O'Dea, and Y. Wang, "Relative location in wireless networks," in *Vehicular Technology Conference, 2001. VTC 2001 Spring. IEEE VTS 53rd*, vol. 2, pp. 1149–1153 vol.2, 2001.

[16] M. Saxena, P. Gupta, and B. Jain, "Experimental analysis of rssi-based location estimation in wireless sensor networks," in *Communication Systems Software and Middleware and Workshops, 2008. COMSWARE 2008. 3rd International Conference on*, pp. 503–510, Jan. 2008.

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

[18] N. Patwari, J. Ash, S. Kyperountas, I. Hero, A.O., R. Moses, and N. Correal, "Locating the nodes: cooperative localization in wireless sensor networks," *IEEE Signal Processing Magazine*, vol. 22, pp. 54–69, July 2005.

[19] Y.-B. Ko and N. H. Vaidya, "Location-aided routing (lar) in mobile ad hoc networks," *Wirel. Netw.*, vol. 6, no. 4, pp. 307–321, 2000.

[20] C. D. Whitehouse, "The design of calamari: an ad-hoc localization system for sensor networks," 2002. Maters Thesis.

[21] Y. Gwon, R. Jain, and T. Kawahara, "Robust indoor location estimation of stationary and mobile users," in *INFOCOM 2004. Twenty-third AnnualJoint Conference*

of the IEEE Computer and Communications Societies, vol. 2, pp. 1032–1043 vol.2, March 2004.

[22] H. Akcan, V. Kriakov, H. Brönnimann, and A. Delis, "Gps-free node localization in mobile wireless sensor networks," in *MobiDE '06: Proceedings of the 5th ACM international workshop on Data engineering for wireless and mobile access*, (New York, NY, USA), pp. 35–42, ACM, 2006.

[23] D. Kurth, G. Kantor, and S. Singh, "Experimental results in range-only localization with radio," in *2003 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS 2003)*, vol. 1, pp. 974–979 vol.1, Oct. 2003.

[24] D. Niculescu and B. Nath, "Ad hoc positioning system (aps) using aoa," in *INFOCOM 2003. Twenty-Second Annual Joint Conference of the IEEE Computer and Communications Societies. IEEE*, vol. 3, pp. 1734–1743 vol.3, March-3 April 2003.

[25] R. Mautz, "The challenges of indoor environments and specification on some alternative positioning systems," in *Positioning, Navigation and Communication, 2009. WPNC 2009. 6th Workshop on*, pp. 29–36, March 2009.

[26] R. Tinos, L. Navarro-Serment, and C. Paredis, "Fault tolerant localization for teams of distributed robots," in *IEEE/RSJ International Conference on Intelligent Robots and Systems, 2001.*, vol. 2, pp. 1061–1066 vol.2, 2001.

[27] A. Savvides, C.-C. Han, and M. B. Strivastava, "Dynamic fine-grained localization in ad-hoc networks of sensors," in *MobiCom '01: Proceedings of the 7th annual international conference on Mobile computing and networking*, (New York, NY, USA), pp. 166–179, ACM, 2001.

[28] A. Savvides, H. Park, and M. B. Srivastava, "The bits and flops of the n-hop multilateration primitive for node localization problems," in *WSNA '02: Proceedings of the 1st ACM international workshop on Wireless sensor networks and applications*, (New York, NY, USA), pp. 112–121, ACM, 2002.

[29] C.-C. Tsai, "A localization system of a mobile robot by fusing dead-reckoning and ultrasonic measurements," in *Instrumentation and Measurement Technology Conference, 1998. IMTC/98. Conference Proceedings. IEEE*, vol. 1, pp. 144–149 vol.1, May 1998.

[30] F. Sivrikaya and B. Yener, "Time synchronization in sensor networks: a survey," *Network, IEEE*, vol. 18, pp. 45 – 50, july-aug. 2004.

[31] M. Deffenbaugh, J. Bellingham, and H. Schmidt, "The relationship between spherical and hyperbolic positioning," in *OCEANS '96. MTS/IEEE. 'Prospects for the 21st Century'. Conference Proceedings*, vol. 2, pp. 590–595 vol.2, Sep 1996.

[32] D. Akbarov, S. Kim, Y. Park, and G. Han, "An improved tdoa positioning method using pattern matching for cdma networks," in *Mobility '07: Proceedings of the 4th international conference on mobile technology, applications, and systems and the 1st international symposium on Computer human interaction in mobile technology*, (New York, NY, USA), pp. 268–273, ACM, 2007.

[33] O. Hernandez, F. Bouchereau, and D. Munoz, "Maximum likelihood position estimation in ad-hoc networks using a dead reckoning approach," *Wireless Communications, IEEE Transactions on*, vol. 7, pp. 1572–1584, May 2008.

[34] C. Randell, C. Djiallis, and H. Muller, "Personal position measurement using dead

reckoning," in *Wearable Computers, 2003. Proceedings. Seventh IEEE International Symposium on*, pp. 166–173, Oct. 2003.

[35] D. Kang, H. Hashimoto, and F. Harashima, "Position estimation for mobile robot using sensor fusion," in *1995 IEEE/RSJ International Conference on Intelligent Robots and Systems 95.*, vol. 1, pp. 271–276 vol.1, Aug 1995.

[36] M. Khalaf-Allah and K. Kyamakya, "Position tracking and global localization of mobile terminals in cellular networks," in *IEEE 8th Workshop on Signal Processing Advances in Wireless Communications. SPAWC 2007.*, pp. 1–5, June 2007.

[37] M. Vossiek, A. Urban, S. Max, and P. Gulden, "Inverse synthetic aperture secondary radar concept for precise wireless positioning," *IEEE Transactions on Microwave Theory and Techniques*, vol. 55, pp. 2447–2453, Nov. 2007.

[38] R. Luo, J.-T. Huang, and O. Chen, "A triangular selection path planning method with dead reckoning system for wireless mobile sensor mote," in *IEEE International Conference on Systems, Man and Cybernetics, 2006. SMC '06.*, vol. 1, pp. 162–168, Oct. 2006.

[39] R. Luo, S.-S. Huang, and W.-L. Hsu, "Mobile sensor node localization based on sensor fusion using covariance intersection approach," in *Advanced Robotics and Its Social Impacts, 2007. ARSO 2007. IEEE Workshop on*, pp. 1–6, Dec. 2007.

[40] A. Catovic and Z. Sahinoglu, "The cramer-rao bounds of hybrid toa/rss and tdoa/rss location estimation schemes," *Communications Letters, IEEE*, vol. 8, pp. 626–628, Oct. 2004.

[41] J. Leonard and H. Durrant-Whyte, "Mobile robot localization by tracking geometric beacons," *Robotics and Automation, IEEE Transactions on*, vol. 7, pp. 376–382, Jun 1991.

[42] V. T. Dang, "An adaptive kalman filter for radar tracking application," pp. 261 –264, sept. 2008.

[43] L. Li, B. Liu, H. Yu, and H. Hu, "A new channel tracking algorithm based on kalman filter," pp. 1 –4, sept. 2006.

[44] K. Deergha Rao, M. Swamy, and E. Plotkin, "Adaptive filtering approaches for colour image and video restoration," *Vision, Image and Signal Processing, IEE Proceedings* -, vol. 150, pp. 168 – 177, june 2003.

[45] T. S. Rappaport, *Wireless Communications, Principles and Practice.* Prentice Hall, 2 ed., 2002.

[46] winAVR, "http://winavr.sourceforge.net/."

[47] A. Papoulis and S. U. Pillai, *Probability, Random Variables, and Stochastic Processes.* McGraw-Hill, 4 ed., 2002.

[48] W. Press, T. S, V. W, and F. B, *Numerical Recipes in C.* Cambridge University Press, second ed., 1996.

[49] S. M. Kay, *Fundamentals of Statistical Signal Processing: Estimation Theory.* Prentice Hall, 1993.

[50] H. L. van Trees, *Detection, Estimation and Modulation Theory.* New York: Wiley, 1968.

[51] J. Taylor, "The cramer-rao estimation error lower bound computation for deterministic nonlinear systems," *IEEE Transactions on Automatic Control*, vol. 24, pp. 343–344, April 1979.

[52] P. Tichavsky, C. Muravchik, and A. Nehorai, "Posterior cramer-rao bounds for discrete-time nonlinear filtering," *Signal Processing, IEEE Transactions on*, vol. 46, pp. 1386–1396, May 1998.

[53] N. Jalden, P. Zetterberg, B. Ottersten, A. Hong, and R. Thoma, "Correlation properties of large scale fading based on indoor measurements," in *Proceedings of the 2007 IEEE Wireless Communications and Networking Conference (WCNC '07)*, pp. 1894–1899, March 2007.

[54] S. Haykin, *Adaptive Filter Theory.* Prentice-Hall, 3 ed., 1996.

[55] B. Ristic, S. Arulampalam, and N. Gordon, *Beyond the Kalman Filter.* Arctec House, 2004.

[56] M. Inversion, "http://en.wikipedia.org/wiki/invertible_matrix."