A STOCHASTIC LOCALIZATION STRATEGY FOR

WIRELESS SENSOR NETWORKS

By

YUNTAO ZHU

A thesis submitted in partial fulfillment of
the requirements for the degree of

MASTER OF SCIENCE

WASHINGTON STATE UNIVERSITY
School of Electrical Engineering and Computer Science

DECEMBER 2006
To the Faculty of Washington State University:

The members of the Committee appointed to examine the thesis of YUNTAO ZHU find it satisfactory and recommend that it be accepted.

________________________
Chair

________________________

________________________
ACKNOWLEDGEMENT

I would like to express my deepest gratitude to my advisor Dr. Murali Medidi for his constant help, guidance and support. Without his valuable advice and continual motivation and encouragement, I would not have reached this milestone. I have recognized him as an invaluable mentor and friend. I would also like to thank Dr. Sirisha Medidi and Dr. Min Sik Kim for agreeing to serve as committee members. I am also grateful to the entire faculty and staff of the Computer Science Department at Washington State University for their help during my graduate study here. I would also like to thank all the members of Washington State Security and Wireless Ad-hoc Networking (SWAN) Group and especially Christopher J. Mallery and Roger A. Slaaen for their valuable input and help.

Finally, no words can express my indebtedness to my parents, my wife Huajun Zhou, and my son DanDan and my daughter TianTian for their love, patience and support during my work on this thesis.
A STOCHASTIC LOCALIZATION STRATEGY FOR
WIRELESS SENSOR NETWORKS

Abstract

by Yuntao Zhu, M.S.
Washington State University
December 2006

Chair: Murali Medidi

Localization is one of the most active research area in wireless sensor networks (WSN). Lo-
calization usually refers to the process of determining the position(s) of one or more node(s) in
a large network. The challenge of localization lies in efficiently providing “acceptable” accu-

racy while conforming to the many constraints of WSNs. In this thesis we introduce a stochastic
strategy for estimating unknown node positions in a wireless sensor network based exclusively on
connectivity-induced constraints. Known peer-to-peer communication in the network is modeled
as a set of geometric constraints on the node positions. The solution of a feasibility problem for
these constraints yields estimates for the unknown positions of the nodes in the network. One
of the major characteristics of our iterative method is that the location information we obtain in
each iteration is utilized in our one-hop distance estimation for the next iteration, which makes our
method more accurate and adaptive.
# TABLE OF CONTENTS

<table>
<thead>
<tr>
<th>Section</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>ACKNOWLEDGEMENTS</td>
<td>iii</td>
</tr>
<tr>
<td>ABSTRACT</td>
<td>iv</td>
</tr>
<tr>
<td>LIST OF TABLES</td>
<td>vii</td>
</tr>
<tr>
<td>LIST OF FIGURES</td>
<td>viii</td>
</tr>
<tr>
<td>CHAPTER</td>
<td></td>
</tr>
<tr>
<td>1. INTRODUCTION</td>
<td>1</td>
</tr>
<tr>
<td>2. BACKGROUND AND RELATED WORK</td>
<td>5</td>
</tr>
<tr>
<td>2.1 Wireless sensor networks</td>
<td>5</td>
</tr>
<tr>
<td>2.2 Localization</td>
<td>6</td>
</tr>
<tr>
<td>2.3 Related work</td>
<td>8</td>
</tr>
<tr>
<td>2.3.1 Overview</td>
<td>8</td>
</tr>
<tr>
<td>2.3.2 Range aware</td>
<td>10</td>
</tr>
<tr>
<td>2.3.3 Probabilistic approach</td>
<td>15</td>
</tr>
<tr>
<td>2.3.4 Range free</td>
<td>17</td>
</tr>
<tr>
<td>3. STOCHASTIC LOCALIZATION FOR WSNS</td>
<td>23</td>
</tr>
<tr>
<td>3.1 Neighborhood Discovery</td>
<td>25</td>
</tr>
<tr>
<td>3.2 Constraints Construction</td>
<td>26</td>
</tr>
<tr>
<td>3.3 Localization Estimation</td>
<td>28</td>
</tr>
<tr>
<td>3.4 One-hop Distance Estimation</td>
<td>29</td>
</tr>
<tr>
<td>Table</td>
<td>Description</td>
</tr>
<tr>
<td>-------</td>
<td>---------------------------</td>
</tr>
<tr>
<td>4.1</td>
<td>Simulation parameters</td>
</tr>
</tbody>
</table>
# LIST OF FIGURES

<table>
<thead>
<tr>
<th>Figure</th>
<th>Description</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>3.1</td>
<td>Node neighborhoods</td>
<td>25</td>
</tr>
<tr>
<td>3.2</td>
<td>Communication constraints</td>
<td>27</td>
</tr>
<tr>
<td>3.3</td>
<td>Distribution of distances for a certain received signal strength. See [1].</td>
<td>31</td>
</tr>
<tr>
<td>3.4</td>
<td>Distribution of one-hop distances</td>
<td>31</td>
</tr>
<tr>
<td>4.1</td>
<td>Topology example. Adapted from [2]</td>
<td>36</td>
</tr>
<tr>
<td>4.2</td>
<td>One-hop distance modeled as a fixed value</td>
<td>36</td>
</tr>
<tr>
<td>4.3</td>
<td>One-hop distance modeled as a random variable with fixed parameters</td>
<td>37</td>
</tr>
<tr>
<td>4.4</td>
<td>One-hop distance modeled as a random variable with updated parameters</td>
<td>38</td>
</tr>
<tr>
<td>4.5</td>
<td>Example of a C-shaped topology. Adapted from [2]</td>
<td>39</td>
</tr>
<tr>
<td>4.6</td>
<td>Accuracy for C-shaped topologies</td>
<td>40</td>
</tr>
<tr>
<td>4.7</td>
<td>Accuracy for topologies with irregular node densities</td>
<td>41</td>
</tr>
<tr>
<td>4.8</td>
<td>Connectivity in topology with four obstacles. Adapted from [2]</td>
<td>42</td>
</tr>
<tr>
<td>4.9</td>
<td>Connectivity in topology with H-shaped obstacle. Adapted from [2]</td>
<td>42</td>
</tr>
<tr>
<td>4.10</td>
<td>Accuracy for topologies with four obstacles</td>
<td>43</td>
</tr>
<tr>
<td>4.11</td>
<td>Accuracy for topologies with one H-shaped obstacle</td>
<td>43</td>
</tr>
<tr>
<td>4.12</td>
<td>Convergence time</td>
<td>44</td>
</tr>
<tr>
<td>4.13</td>
<td>Communication overhead</td>
<td>44</td>
</tr>
<tr>
<td>4.14</td>
<td>Accuracy in large networks</td>
<td>45</td>
</tr>
</tbody>
</table>
CHAPTER ONE

INTRODUCTION

Wireless sensor network is a network consisting of spatially distributed autonomous devices using sensors to cooperatively monitor physical or environmental conditions, such as temperature, sound, vibration, pressure, motion or pollutants, at different locations. Each node in a sensor network is typically equipped with a radio transceiver or other wireless communications device, a small micro-controller, and an energy source, usually a battery. Size and cost constraints on sensor nodes result in corresponding constraints on resources such as energy, memory, computational speed and bandwidth.

The development of wireless sensor networks was originally motivated by military applications such as battlefield surveillance. However, wireless sensor networks are now used in many civilian application areas, including environment and habitat monitoring, healthcare applications, home automation, and traffic control. Wireless sensor networks operate in the absence of a pre-deployed infrastructure, are self-configurable, low cost and can be rapidly deployed. Many of the applications proposed for wireless sensor network require knowledge of the origin of the sensing information, which gives rise to the problem of “localization”.

Localization can usually be described as the process of determining the position, or location, of someone or something, relative to someone or something else. For sensor networks this simply means locating a sensor node in a network.

Extensive research has been done on localization for wireless sensor networks. The approaches taken to achieve localization in sensor networks differ in their assumptions about the network deployment and the hardware capabilities. Centralized localization techniques depend on sensor nodes transmitting data to a central location, where computation is performed to determine the location of each node. Distributed localization methods do not require centralized computation, and rely on each node determining its location with only limited communication with nearby nodes.
These methods can be classified as range-aware and range-free or hop-count based. Range-aware techniques use distance estimates or angle estimates in location calculations, while a hop-count based solution depends only on the contents of received messages. Range-based approaches have exploited time of arrival, received signal strength, time difference of arrival of two different signals, and angle of arrival. Though they can reach fine resolution, either the required hardware is expensive or the results depend on other unrealistic assumptions about signal propagation (for example, the actual received signal strengths of radio signals can vary when the surrounding environment changes). Because of the hardware limitations of sensor devices, hop-count based localization algorithms are a cost effective alternative to more expensive range-based approaches. There are two main types of hop-count based localization algorithms that have been proposed for sensor networks: local techniques that rely on a high density of anchors so that every node can hear several anchors, and hop-counting techniques that rely on flooding a network. In the centroid method, each node estimates its location by calculating the center of the locations of all anchors it hears. If anchors are well positioned, location error can be reduced. Some approach isolates the environment into triangular regions between beacon nodes, and uses a grid algorithm to calculate the maximum area in which a node will likely reside. This method typically assumes a larger radio range for anchor nodes and hence has high anchor density. To provide localization in networks where anchor density is low, hop-counting techniques propagate location announcements throughout the network. Each node maintains a counter denoting the minimum number of hops to each anchor, and updates that counter based on messages received. Anchor location announcements propagate through the network. When a node receives a new anchor announcement, if its hop count is lower than the stored hop count for that anchor, the recipient updates it hop count to the new value and retransmits the announcement with an incremented hop count value. The coordinates of anchors are flooded throughout the network so each node can maintain a hop-count to that anchor. Nodes calculate their position based on the received anchor locations and corresponding hop count.
We are interested in performing localization in a more general network environment where no special hardware for ranging is available, and no anchor nodes are required. In our approach, first we derive feasible solutions to the position estimation problem using optimization techniques. If one node can communicate with another, a proximity constraint exists between them. As a physical example, if a particular RF system can transmit 20m and two nodes are in communication, their separation must be less than 20m. These constraints restrict the feasible region of unknown node positions. Once such a region is obtained, we use least-square method to find the best-fit point within the feasible region. In our approach we strive to find a good estimation of the one-hop distance. We model the one-hop distance as a random variable $d$. Based on the current position information we calculate the distances from each node to its one-hop neighbors and use these distances as samples of the random variable $d$. Then we use these samples to estimate the parameters of the random variable $d$ and feed this information back to our algorithm. We believe this approach is more realistic than many others, in which one-hop distance is prescribed as a constant. In each iteration $d$ gets updated and it allows us to make more and more aggressive improvement on position estimations.

In a summary, we consider the localization problem for wireless sensor networks under the following assumptions

- The sensor networks is self-organizing which implies that there is no fine control over the placement of the sensor nodes when the network is installed.

- We assume that nodes are randomly distributed across the environment.

- Nodes are deployed on a planar terrain.

- Nodes are static.

- All the nodes are equipped with the same wireless transceivers and are capable of communicating with other nodes within transmission range.
• Initially, none of the nodes in the network knows its location (i.e., no anchor nodes).

• No ranging information is available.

• Antenna is symmetric.

Under these assumptions, our algorithm is fully distributed and only uses local broadcast for communication with immediate neighbors. Since we assume that the nodes in the network do not have the capability to measure the distance between directly connected nodes, we need to estimate the one-hop distance. We model it as a random variable \( d \) in our approach. Initially, we use the transmission range as an estimation of \( d \), and as our algorithm proceed and more information becomes available, we adjust our estimation on \( d \) accordingly. In this way our algorithms is expected to be more accurate than other hop-count based localization algorithms. Since the one-hop distance is models as a random variable in our algorithm rather than being assigned to a constant value. And hence the randomness in the localization problem is better captured and handled by our algorithm.

Clearly, in absence of any anchors, nodes are clueless about their real locations. So our algorithm only determines the relative or virtue positions of the wireless nodes. The underlying motivation for this paradigm is the observation that for many applications, it is not necessary to have real locations; often it is sufficient to have virtual locations only.

The remainder of this thesis is organized as follows. Chapter 2 presents background and related work in the area of wireless sensor network localization. Our stochastic localization technique is described in Chapter 3. Chapter 4 contains performance evaluation, and Chapter 5 provides concluding remarks and potential future work directions.
CHAPTER TWO

BACKGROUND AND RELATED WORK

2.1 Wireless sensor networks

A wireless sensor network (WSN) is a network consisting of spatially distributed autonomous devices using sensors to cooperatively monitor physical or environmental conditions, such as temperature, sound, vibration, pressure, motion or pollutants, at different locations [3]. The development of wireless sensor networks was originally motivated by military applications such as battlefield surveillance. However, wireless sensor networks are now used in many civilian application areas, including environment and habitat monitoring, healthcare applications, home automation, and traffic control [3]. In addition to one or more sensors, each node in a sensor network is typically equipped with a radio transceiver or other wireless communications device, a small microcontroller, and an energy source, usually a battery. The size a single sensor node can vary from shoebox-sized nodes down to devices the size of grain of dust. The cost of sensor nodes is similarly variable, ranging from hundreds of dollars to a few cents, depending on the size of the sensor network and the complexity required of individual sensor nodes [3]. Size and cost constraints on sensor nodes result in corresponding constraints on resources such as energy, memory, computational speed and bandwidth [3].

Wireless sensor networking is an active research area due to its applicability. The applications for WSNs are many and varied. They are used in commercial and industrial applications to monitor data that would be difficult or expensive to monitor using wired sensors. They could be deployed in wilderness areas, where they would remain for many years (monitoring some environmental variable) without the need to recharge/replace their power supplies. They could form a perimeter about a property and monitor the progression of intruders (passing information from one node to the next).
Typical applications of WSNs include monitoring, tracking, and controlling. Some of the specific applications are habitat monitoring, object tracking, nuclear reactor controlling, fire detection, traffic monitoring, etc. In a typical application, a WSN is scattered in a region where it is meant to collect data through its sensor nodes.

- Environmental monitoring
- Habitat monitoring
- Acoustic detection
- Seismic Detection
- Military surveillance
- Inventory tracking
- Medical monitoring
- Smart spaces
- Process Monitoring

2.2 Localization

Location awareness is important for wireless sensor networks since many applications such as environment monitoring, vehicle tracking and mapping depend on knowing the locations of sensor nodes. In addition, location-based routing protocols can save significant energy by eliminating the need for route discovery and improve caching behavior for applications where requests may be location dependent. Security can also be enhanced by location awareness (for example, preventing wormhole attacks).
Given a wireless sensor network consisting of low-power devices, localization is the task of discovering the positions of the sensor nodes.

The Global Positioning System (GPS) solves the problem of localization in outdoor environments for PC-class nodes. However, for large networks of very small, cheap, low-power devices, practical considerations such as size, form factor, cost, and power constraints of the nodes preclude the use of GPS on all nodes. Each node in the network needs to figure out its location based on some type of calculation instead of relying on GPS system. So when we address the problem of localization for such devices, we consider the following issues:

- **RF-based**: We focus on small nodes that have some kind of short-range radio frequency (RF) transceiver. Our primary goal is to leverage this radio for localization, thereby eliminating the cost, power, and size requirements of a GPS receiver.

- **Receiver-based**: In order to scale well to large distributed networks, the responsibility for localization must lie with the receiver node that needs to be localized and not with the reference points.

- **Ad hoc**: In order to ease deployment, we desire a solution that does not require preplanning or extensive infrastructure.

- **Responsiveness**: We need to be able to localize within a fairly low response time.

- **Low energy**: Small untethered nodes have modest processing capabilities and limited energy resources. If a device uses all its energy localizing itself, it will have none left to perform its task. Therefore, we desire to minimize computation and message costs to reduce power consumption.
2.3 Related work

2.3.1 Overview

Localization in wireless sensor network is an active research area. This research can be divided into two categories; range-aware and range-free. Both range-aware and range-free approaches employ traditional mathematical methods, such as triangulation or optimization techniques, in order to calculate node positions.

Range-aware localization techniques are typically dependent on some form of distance estimates, where these estimates generally are derived inter-node distances. There are several ways of obtaining an estimate for the distance between two nodes: received signal strength, time of arrival, angle of arrival, etc. The accuracy of the estimate will be highly dependent on the environment and quality of the transmission equipment. Errors caused by multi-path fading, echo from obstacles, interference, etc. can produce highly inaccurate estimates. Some range-aware localization techniques also depend on a number of specialized nodes. These can be anchor nodes, beacon nodes or mobile robots. An anchor node is usually a GPS enabled sensor node, but can also be a node that is more capable than a “normal” sensor node. A beacon node is a node that normally has no “sensing” functions, and the task of a beacon node is just to help the localization progress. A mobile robot can combine tasks of anchor nodes and beacon nodes and it has mobility. These robots often also function as data collectors.

Range-free localization, often also referred to as hop-based or connectivity-based localization, aims to overcome the inherent difficulty of accurately determining exact inter-node distances in sensor networks. While hop-based techniques do not require inter-node distance information, many hop-based techniques have the ability to take advantage of such information, if available, to provide more accurate results. The connectivity of a sensor node is used as an indication of how close this node is to other nodes, and all nodes within its transmission range is said to be one hop away. The measure of hops is simply a very crude approximation of distance.
In [4], Hightower and Borriello present a survey and taxonomy of location systems for mobile computing applications. Comparing different schemes and their performance can be very difficult because different approaches solve many different problems, and each technique can differ from others in many ways. The physical media used for the location, the context where the position is needed, power requirements, infrastructure and resolution of results will all change from application to application. Hightower and Borriello attempt to provide the means to make an easier comparison of implementations and applications. Another survey is presented in [5], where the authors provide a survey of positioning design for wireless communication technology. The authors look at localization both in cellular technology and in wireless networks like WSN. The survey provides a categorization of the different technologies ranging from GPS to sensor network schemes.

Practical results concerning two key issues in the deployment of localization service in wireless sensor networks are presented in [6]. Anlauf and Sunbul discuss both the selection of appropriate sensors for acquiring the data needed to perform localization and the actual localization algorithms. The authors look at the use of Motes or motes-like technologies and point out that for most applications there will not be any map of the monitored area. Motes are tiny devices equipped with sensors, an onboard computer and wireless communication technology. Motes with both long and short distance antennas were tested to determine the range of the transmissions. The authors claim that not only signal strength and distance determines how much of the information is received, but also the angle between the sender and receiver and the environment in which the signal is sent. Only at the distance of 40cm, the percentage of information received is unpredictable. Trilateration is shown to work well when the sensor reading is accurate, but the accuracy of the localization drops when the accuracy of the readings drops. The Gaussian distribution based algorithm is more robust and almost insensitive to deviation in the accuracy of the sensor readings, but is also much
more complex in terms of space and time requirements. Theoretical foundations for network localization in wireless sensor networks are also presented in [7]. The authors point out that the problem of localization shares a number of features with other active fields of study, rigidity and global rigidity in frameworks, the coordination formations of autonomous agents, and geometric constraints in CAD. The authors use techniques and results from other techniques to lay a coherent solid foundation for the underlying problem of when a network is uniquely localizable. The computational complexity of localization is also studied. In [8] the authors present studies of the Cramér-Rao lower bound (CRB) of localization. The authors look at two kinds of sensor network localization based on noisy range measurements, both with and without anchor nodes. CRB is used to evaluate the hardness of an estimation problem and a CRB-like bound for the estimation variance is presented. In [9] a Bayesian method to analyse the lower bounds of localization uncertainty in sensor networks is presented and compared it to the Cramér-Rao bound. The authors attempts to analyze the dependency of localization on sensor network topologies.

2.3.2 Range aware

In [10] a technique that uses range measurements between pairs of nodes and the known coordinates of the anchor nodes in wireless ad hoc networks, to estimate the position of every node is proposed. The method will first establish the position of the nodes close to one of the anchor nodes, and then use this information to estimate positions of nodes further away from the anchor nodes. As in most range aware technique, inaccuracies in the range measurements will severely affect this algorithm, and errors may propagate fast throughout the network. The distribution and number of anchor nodes in a network will also affect the performance of the algorithm. Another similar distributed algorithm for localization in wireless sensor network is described in [11]. This algorithm is based on the iterative propagation of information through the network, and measurements, with limited accuracy, of the distances between pairs of nodes used to derive position estimates.
The authors mention the standard methods for acquiring these measurements, but the technique described is independent of the method used for distance measurement. The authors also mention that different techniques offer different tradeoffs between accuracy, complexity cost, etc. Errors in the measurements can come from multipath interference, line-of-sight obstructions and more. It is assumed that there is a large error in the distance measurements used, and mentions that this should be representative of realistic measurements. The algorithm described is divided into two phases; start-up and refinement. The authors describes a Hop-TERRAIN algorithm that is used in the start-up phase and the result from this phase is iteratively improved during the refinement phase. The HOP-TERRAIN algorithm is similar to the DV-Hop algorithm. All tests were implemented in C++ using OMNeT++ discrete event simulator. All nodes were either randomly placed with a uniform distribution or placed according to a grid, but the refinement algorithm does not show any better results in the later case. As the average connectivity in the network decreases both algorithms tested fails to derive positions for parts of the network. The Hop-TERRAIN algorithm is shown to be insensitive to ranger errors, while the refinement algorithm needs a ranger error below 40% to show average performance.

In [12], Partial Range Information (PRI) and a partial-range-aware localization technique is described. Partial range information (PRI) is defined as any type of measurement which is monotonically increasing or decreasing and has an unknown or environment-dependent one-to-one relationship with the range measurement. The described scheme is distributed and uses received signal strength measurements in the localization process. The algorithm is developed as a module that can be plugged in to any range free hop-based localization algorithm to improve the result. The technique described is called RangeQ. It is a distributed range quantization technique that will associate range values with all 1-hop connections with unknown distances. The quantization is similar to quantization in image processing. The authors considers three different models, simple linear, min-max neighbor linear and area proportional model, for use in the quantization to obtain the cluster sizes and range values of each node. The scheme presented is shown to reduce position
errors with up to 50% from previous range-free-algorithms and it preforms better than both range
free and range aware methods when the range errors is between 15% and 35%.

In [13] the Approximate Maximum-likelihood (AML) method for source localization and Di-
rection of Arrival (DOA) estimation [14] is reviewed, and the authors also considerers a least
squares method applied to the DOA bearing crossings to preform the source localization. A virtual
array model applicable for the AML-DOA method for use in reverberant scenarios is proposed.
The work illustrates that the proper usage of novel array and signal processing algorithms imple-
mented on very low-cost Commercial-Off-The-Shelf (COTS) platforms is capable of achieving
quite sophisticated real-time acoustical beamforming operation for source localization.

In [15] localization based on a the use of a single mobile beacon is presented. The technique
uses received signal strength to estimate the distances between nodes and the beacon. This gives
the method an accuracy of a few meters. The beacon will broadcast its own position and any node
receiving this messages can infer that it is positioned in a certain area with a certain probability.
The received signal strength is measured for every received beacon message and will place a con-
straint on the possible position of a node. Each node will used Bayesian inference to compute the
position estimate. This technique can be implemented both centralized and distributed, and scales
well, both in terms of number of nodes and density of nodes. The trajectory of the mobile beacon
will affect the results and must be considered. A localization technique that uses the signal strength
from two static beacons is presented in [16]. Interference and fading in the signals are handled by
using the average over several measurements within a certain time-frame. The two beacons are
placed in two not-opposite corners of a rectangle that encloses all devices. Problems occurring
both with static and mobile nodes are considered and the best performance in both scenarios is
obtained in a fast fading environment. The minimum error is in both cases where the variance in
error as a function of the size of the time-frame is said to be very small. Beacons are also used
in [17], where an efficient localization algorithm using four mobile beacons is described. Four
beacon nodes will create a rectangle. The rectangle will have the sensor node in the center and the
four beacons in the corners, by moving towards the node. The node can then estimate its own position by position information from the four beacons. Received signal strength is used to estimate the distance between a sensor and a beacon. In this scheme the beacon nodes will perform the distance measurements in several iterations and provide the node with this information. All nodes will be assigned an unique ID to separate the localized and non-localized nodes. The authors also argues that the proposed technique presents a good tradeoff between scalability, robustness, energy saving and accuracy. In [18] motivations for the need for empirical adaptive beacon placement is presented. The authors describe the design and evaluate three adaptive beacon placement algorithms based on RF-proximity. The authors claim the density of beacon nodes has greater impact on the performance of placement algorithms than noise levels. In [19] a formulation to determine optimal beacon placement in wireless sensor networks is described. The authors note that any placements algorithm must consider both beacon coverage area and network lifetime. The authors formulate an optimization problem which is solved by Integer Linear Program.

The work described in [20] is motivated by the Smart Dust project that is aiming to scale sensing communication platforms down to cubic milliliter. The authors proposes a technique for estimating node positions in wireless sensor networks based on connectivity induce constraints. Known peer-to-peer communications is modeled as a set of geometric constraints on the node positions. The algorithm proposed, provides results close to the actual positions of node, give tight enough constraints. The paper also provides an additional method for placing rectangular bounds around the possible positions of all unknown nodes in the network. The methodology for formulating the position estimation problem as a linear or semidefinite program is presented. Results from simulations with on average 194 nodes, with average connectivity 5.7 is presented. The performance metric is defined as the mean error between the actual position and the estimated position. With a low number of anchor nodes, the algorithm will produce very inaccurate results, and only when the number of anchor nodes is very high is the accuracy acceptable. When using angular constraints, 26 anchor nodes gives a mean square error of $\frac{2}{3}R$ in a $10R \times 10R$ network. The
results can be improved by running the algorithm several times and used the intersection between the different results. Additional improvements can be achieved by placing the anchor nodes at the perimeter of the network. In [21] a distributed localization technique where geometric constraints, placed by both connectivity and sensing, is used to increase the accuracy of position estimates is proposed. Sensing constraints is caused by mobile objects that are sensed by several nodes. These constraints are said to usually be tighter than connectivity induced constraints. The authors assumes a fraction of the nodes will know its own position and the paper also assumes that the transmission range of each sensor is modeled as a disk with radius $r$ and the node in the center and that all nodes will have the same transmission range. The authors argues that it makes sense to integrate the localization process with the application and exploit additional information gathered over the course of running the application. The technique described suggest that instead of using triangulation, a bounding box for each node can be established. Negative information, i.e. when a target is not sensed, can also be used as constraint on a position. The efficiency of proposed technique is compared to the efficiency of a centralized solution. Using the distance bound sensing model and negative constraints the proposed algorithm outperforms the centralized one. The paper shows that the technique described can be used to increase the accuracy in localization. Simulations is done using TinyOS-Nido platform.

In [22] a distributed localization algorithm is proposed. The algorithm is based on the estimation-comparison-correction paradigm. Multidimensional scaling (MDS) [23] is used to merge maps from neighboring sensors. This is done along a path between a sensor and an anchor node to estimate the sensor position. Errors in the estimates caused by anisotropic network topologies is reduced by iterative estimation, comparison and correction. Received signal strength measurements are used to estimate the distances between the sensors. In [24] much of the same ideas as our own is adopted. The technique presented is Simple Hybrid Absolute Relative Positioning (SHARP). SHARP will attempt to preform localization on a subset of the nodes in a network using ranging measurements. The subset will be localized using the MDS-MAP algorithm [25], and this
subset will then be used as anchor-nodes when the remaining nodes is localized using APS [26]. The test are as in [25] done using MATLAB, but their metric for performance is different in that it also considers the work needed to achieve the accuracy. The authors present a new metric for localization performance called Performance-Cost Metric (PCM). PCM reflects both localization accuracy and localization cost in terms of delay and/or energy consumption.

2.3.3 Probabilistic approach

Many localization schemes take into account the range measurement inaccuracy. In [27] several distributed scalable and probabilistic localization algorithms and one mobile beacon driven communication protocol are compared. In the proposed scheme every unknown node will localize itself based on received beacon packets. The position is estimates using both parametric and non-parametric probabilistic estimation techniques. This technique will make use of a beacon node in a network, and Time of Arrival (TOA) measurements is used to estimate the positions of other nodes. The mobile beacon will traverse the network after initialization and broadcast packets containing the position of the beacon. When an node picks up a packet from the beacon it can infer with a certain probability that it is in a certain proximity to the beacon. The trajectory of the beacon in the simulations is helix, but this might not be optimal and determining the correct trajectory is a difficult problem that is not discussed in this paper. The authors presents simulation results for different distributed localization algorithms. The Non-parametric probabilistic method is shown to be more robust and more accurate that other methods.

In [1] authors consider a probabilistic approach to the problem of localization in wireless sensor networks and propose a distributed algorithm that helps unknown nodes to determine confident position estimates. The proposed algorithm is RF based, robust to range measurement inaccuracies and can be tailored to varying environmental conditions. The proposed position estimation algorithm considers the errors and inaccuracies usually found in RF signal strength measurements.
They also evaluate and validate the algorithm with an experimental testbed. The test bed results indicate that the actual position of nodes are well bounded by the position estimates obtained despite ranging inaccuracies. Their extensive outdoor field measurements and calibration indicate that a received signal strength is inaccurate and the proposed approach uses this information directly for ranging. The algorithm they propose is independent of the type of distribution as long as there exists a feasible and practical method to store and compute the distributions. The proposed algorithm can be tailored to varying environmental conditions by changing the probability distribution parameter that accounts for deviation from the mean position estimate. Their experimental testbed analysis has also shown positive results with the nodes being well bounded by the estimates obtained.

A probabilistic, constraint-based localization approach which is robust to range measurement inaccuracies is proposed in [28]. The proposed approach proceeds in three phases: the first phase involves modeling the uncertainties of range measurements; in the second phase, a set of probabilistic constraints are computed and combined to produce initial position estimates; in the final phase, negative constraints are used to refine the initial estimates. Authors evaluated the proposed approach through simulations based on real-world measurements; the results are compared with two other localization schemes and the Cramer-Rao lower bound. The results show that, for inaccurate range measurements, the proposed probabilistic approach performs the best and close to the optimal bound.

Location information is of paramount importance for Wireless Sensor Networks (WSN). The accuracy of collected data can significantly be affected by an imprecise positioning of the event of interest. Despite the importance of location information, real system implementations, that do not use specialized hardware for localization purposes, have not been successful. In [29], authors propose a location estimation scheme that uses a probabilistic approach for estimating the location of a node in a sensor network. Their localization scheme makes use of additional knowledge of topology deployment. They assume a sensor network is deployed in a controlled manner, where
the goal of the deployment is to form a grid topology. We evaluate our localization scheme through simulations, showing localization errors as low as 3% of radio range. This scheme outperforms similar localization schemes by obtaining 50% less error in localization, when compared to them. They also evaluate their localization solution and the DV-Hop scheme in a real system deployment, obtaining an average error in location of 79% of radio range, outperforming DV-Hop by approximately 40%. They analyze the significant differences in performance between simulations and the real system deployment and stress the importance of further evaluations of real implementations. The result is an effective and realistic protocol that works in an actual system, under certain assumptions, because it exploits deployment information. Their scheme was inspired by a similar solution, called DV-Hop, from Rutgers, which uses hop count, from anchors to sensor nodes, as a measure of distance to known locations. The DV-Hop scheme is more general than their solution, since it does not use the knowledge about deployment topology. However, this generality does not help, since the errors are high in practical deployments. Probability Grid is more resistant than DV-Hop to situations when the hop count is not a very accurate measure of the distance between two points. This is due to the fact that in real deployments, the radio range is not circular. Previous research has shown the presence of long links, backward links or stragglers and a significant deviation from a circular radio pattern. Using a probabilistic approach, the Probability Grid considers the hop count for a particular distance to be a discrete random variable, that has a Poisson distribution. Their solution is completely distributed and does not require special infrastructure.

2.3.4 Range free

In [26] APS, a distributed hop by hop positioning algorithm is proposed. The algorithm works as an extension of both distance vector routing and GPS positioning, and will provide a position estimate for all unlocalized nodes in a sensor network. The technique assumes a limited fraction of the nodes has the ability to determine its own actual position, this can be done with for
example GPS. The algorithm will derive the absolute position because this will incur lower signaling cost if the topology changes, and it enables the use of a unique namespace. One aim of the proposed technique is to enhance the accuracy of the position estimates as the fraction of anchor nodes increases. The paper describes three different hop-by-hop distance propagation techniques. The DV-Hop propagation method uses classical distance vector exchange. The anchor nodes will estimate the average hop size based on the number of hops between different anchor nodes. This average is distributed to the network by controlled flooding, and used to correct the estimates in the network. In large networks a “Time To Live” would be used in the propagation messages to limit the number of anchor nodes that each node acquires. This methods does not depend on measurement error, but this method will not work well in anisotropic networks. The authors presents simulations done in ns-2. Test are done on two different topologies, nodes placed in random uniform manner, and the shape of the letter “C”. The DV-based algorithms are shown to preform better when the ratio of anchor nodes is low. The message complexity is also show to be better in DV-hop in the uniform topology. The errors in position estimates range from around 2% to almost 200% of the hop distance depending on the topology and ratio of anchor nodes.

In [30] an algorithm for estimating the positions of nodes in wireless sensor networks using connectivity information is presented. The information concerning one node is assumed very likely to already be available to that node. This technique can also use additional information such as estimated distances between neighbors or known positions of nodes in a network. The position estimates is derived using Multidimensional Scaling (MDS) [23]. This technique shows especially good performance when the nodes are distributed relatively uniformly, and improves over other techniques when the network has few anchor nodes. The accuracy results from MDS method is dependent on the accuracy of the distance estimates, so MDS can potentially yield a perfect map. Test are done in MATLAB with randomly with uniform distribution, and in both a square and hexagonal grid with different placement errors. The experiments show that the algorithm works much better where the nodes are placed on one of the grids.
In [25] an improvement to the techniques presented in [30] is proposed. The authors describes a distributed algorithm MDS-MAP(P), that uses patches of relative maps. These maps can be computed in parallel at different nodes, to generate estimate for the positions of subsets of nodes. The algorithm is shown to improve over previous when networks are irregularly shaped, and preforms as well as others on uniformly shaped networks. This technique is best applied to networks where the shortest distance between two nodes does not correspond well to their Euclidian distance. The main idea for this method is to compute the local map for each node using MDS, as described in [30]. These local map can then be merged to form the complete map. This approach can reduce the affect that, when estimating the position of one node, a far away node will have on this estimate. An additional refinement step is also employed to improve the accuracy of each local map. The technique describe uses least square minimization, but different techniques, like collaborative multiilateration, can also be used. The method is said to often find the right general layout of a network, but not necessarily the precise location of the nodes. The test are done using two different scenarios; when only connectivity information is available and when both connectivity information and distance estimates are available. The authors present simulation results for four types of topologies, a uniform random placement, uniform grid placement, and random and grid placement in a “C” shape. When using connectivity information only the proposed algorithm preforms better than earlier methods. The average error when using only connectivity information on uniform networks is show to be 27%. Results from test using distance measurements are also shown, but as earlier, the error used is only 5%. The algorithm is show to always derive an estimate for all nodes in the network. The proposed algorithm is also shown to preform better the “C” shaped networks. The authors also presents comparisons between the MDS based algorithms and both DV-hop and DV-distance algorithms. In general the proposed algorithm is shown to preform better. Another distributed variation of the before mentioned MDS-MAP technique is presented in [31]. In this technique estimates the relative positions of nodes in wireless sensor network is derived. The algorithm utilizes a given communication path between a starting and a remote node.
This path can be discovered using techniques like constraint-based routing or limited broadcasting. The nodes on the path will compute their local map, as in [30], based on local distance estimates. As in previously mentioned technique, [25], the number of hops may be changed, and varying this may give different accuracy in different network topologies. The computation of the local maps will, as long as the network stays static, only have to be done once in each node. The relative maps of nodes along the path will then be aligned according to common nodes in the map before the optimal linear transform between the two is computed. The maps of two nodes will be aligned two at the time as long as the two maps has three or more common nodes. These common nodes can not be placed on one line as this would not provide enough information for accurate alignment. The optimal linear transform can be computed in the least-squares sense. The relative position of the remote node in the coordinate-system of the starting node, is computed by applying the sequence of linear transformations. The authors document the test done on proposed algorithm with changes in the chosen hop size in building the local maps and how many common nodes exists when aligning the local maps of neighbors. The test shows that the accuracy of the algorithm is depended on the network connectivity, the errors in local distance measurements, the length of the chosen path and the number of common nodes between two adjacent nodes along the path. The technique performs well on both regular and irregular networks, as long as the network has sufficient connectivity and small enough errors in the distance measurements. Test done on both randomly uniform, and grid based topologies is presented. As before [25] the assumed average range error is 5%. The test show that the method does not preform well when only 1-hop neighbors is considered when establishing the local maps. With the connectivity above 10 and 2-hop neighbors being considered, the algorithm preforms well, but has a higher message complexity. The paper also describes test done using different sized local maps and with different range errors.

In [32], Multilateration based localization like APS [26] and MDS-MAP [25] are compared. Key issues that affect their performance are studied. The authors studies the affect changing the
number of anchor nodes and their distribution has on the performance of the localization algorithms.

In [33] a range free localization (ROCRSSI) is described. In ROCRSSI each node uses a series of overlapping rings to narrow down the possible area in which it resides. The technique is said to perform well, have a low overhead and to be robust when employed in irregularly shaped networks. In [34], a differential Ad-Hoc Positioning System is described. A differential error correction scheme designed to reduce the cumulative distances and positioning error over multiple hops is used. A HopCount algorithm and a HopDistance algorithm is presented and both algorithm uses beacon nodes to estimate other nodes positions. The authors present simulations where 70% of the nodes is estimated with error less than 50% of the transmission range.

In [35] a localization algorithm based of growing local map is presented. The algorithm starts with a local map computed from three non-collinear neighboring nodes, and will continue until all localizable nodes are covered in the map. When the first local map is computed the position of these nodes are broadcast to neighboring nodes, and these nodes will, when they have received the position of two or three different nodes, start to localize itself and then broadcast its own position. Simulations show that compared to APS DV-distance [26] the proposed technique is about two times more accurate for C-shaped grid or hexagon networks with the same coverage as the APS DV-distance algorithm, when the range error is less than 10%.

[36] is one of several recent papers that attempts to provide a more scalable and feasible localization techniques. The authors proposes a localization scheme that employs clustering information and a small number of “anchor nodes”. The clustering is used to provide a regular pattern in the network and helps reducing the communication overhead since only cluster heads will be involved in the initial phase of the localization process. Anchor nodes are assumed to be able to derive the position of all adjacent cluster-heads, this set of localized nodes will be used by other cluster-heads to localize themselves. All cluster-heads will recompute their estimate as more information becomes available. The hop-distance to localized cluster-heads and the regularity in
the edge-length between clusters is used to produce the position estimates. In [37] a technique for energy efficient distributed clustering in wireless sensor networks is proposed. The technique is shown to have low overhead and in simulations the method outperforms weight-based clustering.

In the above approaches, the measure of hops is used as a very crude approximation of distance. And one-hop distance is modeled as a constant, either preassigned or derived from anchor nodes. This modeling method is very restrictive and does not adapt well to node density changes. So far no work shows any attempt to handle the error in one-hop distance estimation and our work reported in the next chapter tries to fill this blank with a novel stochastic localization algorithm.
CHAPTER THREE
STOCHASTIC LOCALIZATION FOR WSNS

In this chapter we propose a localization scheme for wireless sensor networks. Our scheme is distributed, computation efficient and can provide fairly accurate location estimation.

We consider a wireless sensor network with \( n \) nodes. These sensor nodes are deployed on a certain planar region and they stay static. We assume there were no fine control when sensors were distributed over the region so the location of each node is unknown. Each node is equipped with a transceiver and transmission range is limited to \( r \). A node can exchange messages (e.g., ID, location etc.) with other nodes that lie in its transmission range. If one node is within the transmission range of another node, we say they are one-hop neighbors. So each node in the network is aware of its one-hop neighbors and through exchanging information it may become aware of their neighbors’ current (estimated) locations. And through its one-hop neighbors, a node can also have the information about its two-hop neighboring nodes, for example, their currently (estimated) locations. Based on all the information gathered, a node tries to calculate (estimate) its own location. In our algorithm we only consider one-hop neighbors and two-hop neighbors due to a few reasons. First, in our algorithm no ranging information is available. We need to estimate the distance among neighboring nodes and distance estimation errors tend to get cumulated. This is the reason that we do not want to go beyond two-hop neighbors. Second reason is the communication and computational efficiency. Third, we have found that two-hop information is very important. If only one-hop neighbors are considered, it will introduce undesirable inaccuracy. So it is reasonable to consider one-hop and two-hop neighbors in our algorithm.

Our algorithm is distributed. It does not rely on any central computation facility. Instead, each node in the network is responsible for computing its own location. Our algorithm is iterative. Initially none of the sensor nodes knows its own location. So each node picks a random location within the network region as its current calculated location to get started. After communication
with its neighbors, each node recalculates its location based on information gathered from its neighbors and update its current calculated location. We call one cycle of these operations an iteration for a node. We expect the location estimation get refined after each iteration so that the calculated locations of each node will form a sequence that eventually converges to the actual position of the node.

Each node in the network calculates its own location independently. So the calculation among all the nodes in the network is not synchronized. For any node, each iteration of our algorithm contains the following phases:

- Neighborhood Discovery
- Constraints Construction
- Location Estimation
- One-hop Distance Estimation

In the first phase a node broadcasts a HELLO message and this message contains node’s unique ID and its current (estimated) location. This message will be picked up by one-hop neighbors. After a node receives a HELLO message from its one-hop neighbor, it records this message in its memory. In the second phase, all the messages received from one-hop neighbors are utilized to derive a set of quadratic constraints and these convex constraints are linearized to obtain a set of linear constraints, which will provide us a feasible region that the node could lie in. Then in the location estimation phase, we apply the least-square technique (or possibly other techniques) to calculate an estimation of the node's location. Since no ranging information is assumed available, we need to estimate one-hop distance in each iteration of our algorithm and it is done in the last phase.
3.1 Neighborhood Discovery

Two nodes in the network are called neighbors if they are adjacent, i.e. within each other’s transmission range. In Figure 3.1 we show an example of node neighborhood, where the circle represents the transmission range \( r \) of node \( a \) in the center and black dots represent the one-hop neighborhood of \( a \). We use density as a term for the number of nodes within one transmission range. In figure 3.1 the node-density in area enclosed by the range of \( a \) is 8.

![Node neighborhoods](image)

Figure 3.1: Node neighborhoods

In this phase of our algorithm each node \( i \) will broadcast its own unique ID and its current estimated location \((x_i, y_i)\) in what is called a HELLO message. This message will be picked up by any receiver within the transmission range of node \( i \), and nodes that get this broadcast will record this message in their memory. Each node in the network builds a list (table) of its adjacent nodes with their ID’s and their current locations respectively in its memory. After a node receives a HELLO message from its neighbor, it creates one new entry and adds the entry to its adjacency list. None of the broadcasts will be forwarded so that when this phase is over all nodes will know only its one-hop neighborhoods. And through its one-hop neighbors, it can also have the knowledge of its two-hop neighbors if each node in the network broadcasts its adjacency table in the HELLO message.
3.2 Constraints Construction

In this phase, known peer-to-peer communication information in the network is modeled as a set of geometric constraints on the node positions.

If one node can communicate with another, a proximity constraint exists between them. As a physical example, if a particular RF system can transmit 20m and two nodes are in communication, their separation must be less than 20m. These constraints restrict the feasible set of unknown node positions.

Based on the available neighborhood information acquired in the previous phase, a node will be able to derive these constraints. Formally, the network is a graph with $n$ nodes at the vertices (each node having a Cartesian position) and with bidirectional communication constraints as the edges. Positions of the neighboring nodes of node $k$ are known as $(x_1, y_1), \ldots, (x_m, y_m)$ and we want to find out the position of node $k$, say $(x_k, y_k)$. Then node $k$ must lie in the intersection of the following circles:

$$C_i := \{(x, y) : (x - x_i)^2 + (y - y_i)^2 \leq d^2\}, \quad i = 1, 2, \ldots, m,$$

where $d$ is the transmission range. So each neighboring node introduces a quadratic constraint on the position of node $k$.

Figure 3.2 gives an example of combination of geometric constraints. The shaded region represents the feasible position for the light node, constrained by the dark node positions. From (a)-(c), the intersected constraints yield progressively smaller feasible positions.

Each node in the network is aware of its one-hop neighbors and their current estimated locations. And through its one-hop neighbors, a node can also have the information about its two-hop neighboring nodes, for example, their currently estimated locations. These two-hop neighbors also introduce quadratic constraints. Suppose that positions of the two-hop neighboring nodes of node
$k$ are known as $(X_1, Y_1), \ldots, (X_t, Y_t)$ and we want to find out the position for node $k$, say $(x_k, y_k)$. Then node $k$ must lie inside each of the following circles:

$$T_i := \{(x, y) : (x - X_i)^2 + (y - Y_i)^2 \leq 4d^2\}, \quad i = 1, 2, \ldots, t,$$

where $d$ is the transmission range.

At the same time node $k$ must lie outside each of the following circles:

$$T_i' := \{(x, y) : (x - X_i)^2 + (y - Y_i)^2 \leq d^2\}, \quad i = 1, 2, \ldots, t,$$

where $d$ is the transmission range. Since otherwise they will become one-hop neighbors.

So each two-hop neighboring node introduces two quadratic constraint on the position of node $k$.

In a summary, if a node $k$ knows the locations of its one-hop neighbors and its two-hop neighbors as $(x_1, y_1), \ldots, (x_m, y_m)$ and $(X_1, Y_1), \ldots, (X_t, Y_t)$ respectively, then we have the following constraints on the position of node $K$, say $(x_k, y_k)$:

$$\begin{align*}
(x_k - x_i)^2 + (y_k - y_i)^2 &\leq d^2, \quad i = 1, 2, \ldots, m, \quad (3.1) \\
(x_k - X_i)^2 + (y_k - Y_i)^2 &\leq 4d^2, \quad i = 1, 2, \ldots, t, \quad (3.2)
\end{align*}$$
and
\[(x_k - X_i)^2 + (y_k - Y_i)^2 \geq d^2, \quad i = 1, 2, \ldots, t. \tag{3.3}\]

Here note that all the circles \(C_i, \ i = 1, 2, \ldots, m\) and \(T_i', \ i = 1, 2, \ldots, t\) have the same radius \(d\), which is the transmission range. It is because that we assume no ranging information is available. So we use \(d\) the transmission range as an upper bound of the one-hop distance which is very conservative. Actually the one-hop distance is much less than that in practice especially when node density is high. One novelty of our algorithm lies in the fact that later on in the forth phase, we will use location information available so far to derive a more reasonable one-hop distance estimation and use the estimation in the next iteration of our algorithm. So each \(d\) in (3.1, 3.2, 3.3) will be replace by this new estimation.

### 3.3 Localization Estimation

Once we have (3.1, 3.2, 3.3), we are ready to estimate the location of node \(k\). Note that (3.1, 3.2, 3.3) are quadratic constraints and in our computation we prefer linear constraints. So the first step we take in this phase is to linearize the constraints (3.1, 3.2). Currently in our algorithm we do not consider constraints (3.3) since we still do not have a way to linearize them efficiently. This is one of the disadvantages of our approach. We anticipate that incorporating constraints (3.3) into our algorithm in the future will further improve the accuracy of our algorithm.

The process of linearization is inspired by the following fact.

Consider two circles

\[C_1 := \{ (x, y) : (x - x_1)^2 + (y - y_1)^2 \leq d_1^2 \} \]

and

\[C_2 := \{ (x, y) : (x - x_2)^2 + (y - y_2)^2 \leq d_2^2 \}. \]
We assume $C_1$ and $C_2$ intercept at two points and these two points determine a straight line. The line equation can be derived as follows:

- Let $(x - x_1)^2 + (y - y_1)^2 = d_1^2$;
- Let $(x - x_2)^2 + (y - y_2)^2 = d_2^2$;
- Then $(x - x_1)^2 + (y - y_1)^2 - (x - x_2)^2 - (y - y_2)^2 = d_1^2 - d_2^2$.
- Simplifying the above equation gives us the equation of the straight line:
\[
(2y_1 - 2y_2)y + (2x_1 - 2x_2)x = x_1^2 - x_2^2 + y_1^2 - y_2^2 + d_2^2 - d_1^2.
\]

Knowing that a node has multiple one-hop and two-hop neighbors, any two of these one-hop neighbors will determine a straight line as we have just shown above. Similarly, any two of the two-hop neighbors will also determine a straight line. Now suppose the current node have $l$ one-hop neighbors and $t$ two-hop neighbors, then these one-hop neighbors determine $(\frac{l}{2})$ straight lines and two-hop neighbors determine $(\frac{t}{2})$ straight lines. Then these lines surround a polyhedron which is the feasible region for the node position. Note that the linearizing process reduces the quadratic constraints to linear constraints, and it also reduces the number of constraints to make our approach more computationally efficient.

Next we use the least square technique to obtain a best-fit point within these region as the estimation of node location. We update the estimation of the current node location and then move on to the next iteration.

### 3.4 One-hop Distance Estimation

As we have seen in the previous phase, when we derive convex constraints based on connectivity information, we use the transmission range as our estimation of one-hop distance. It is necessary at
the initial stage of our algorithm, even though it is very conservative. As our algorithm proceeds, more and more information becomes available. For example, after we get the estimation of the current node location, we can calculate the distances between current node and its neighboring nodes. If we consider the one-hop distance as a random variable \( d(\omega) \) whose realizations depend on a underlying outcome \( \omega \) in an event space \( \Omega \) with a known probability function \( P \), then these distances are realizations (instances) of \( d(\omega) \). The question is what distribution function \( P \) we should choose.

Recall that in a range-aware localization technique, one tries to use the signal strength to estimate inter-node distance. Extensive field measurements and calibrations are carried out to derive and test this estimation. For example, in [1], authors used two HP Compaq H3870 iPAQs equipped with Lucent Orinoco cards to measure the signal strength as a function of distance. One of the iPAQs was configured to send beacon packets continuously while the other was measuring the signal strength of each received packet. The two iPAQs were placed in an outdoor field and remotely controlled from a laptop. They measured the signal strength and noise in intervals of 2.5m up to 50m. For each distance, they measured the data at 16 different positions (the sender was rotated by 90 degrees, for each position of the sender, the receiver was rotated by 90 degrees). They took 200 measurements at each position for a total of 3200 measurements at every distance. Then they merged all of the data collected and calculated the probability distribution of each signal strength as a function of distance. Interestingly, the probability distribution followed a normal distribution for most of the signal strengths. See Figure 3.3.

No similar work has been done for range-free techniques. So we conduct our experimental studies. We deploy nodes uniformly within a network space and then we calculate the distances between all neighboring nodes. Not surprisingly, we obtain similar results: the one-hop distance fits the normal distribution very well, which is expected. See figure 3.4. The position estimation algorithm that we discuss in this thesis is independent of the type of distribution as long as there exists a feasible way to represent it in a compact fashion. But the work by others and our experiment
studies suggest that a normal distribution for $P$ might be a good choice and our later simulation results prove our intuition.

So in our algorithm, we model the one-hop distance $d(\omega)$ as a random variable that satisfies the normal distribution function $N(\mu, \delta^2)$ and we try to estimate $\mu$ and $\delta$. Suppose current node have $l$ neighbors and we consider $l$ distances from current node to its neighbors as $l$ samples (instances) of $d(\omega)$. We use these samples to estimate the parameters $\mu$ and $\omega$ of the normal distribution function $P$.

Let $d_1, d_2, \ldots, d_l$ be the distances we calculate. Then
\[ \tilde{\mu} := \frac{\sum_{i=1}^l d_i}{l} \]  

is a unbiased estimation for \( \mu \), and

\[ \tilde{\delta} := \sqrt{\frac{\sum_{i=1}^l (d_i - \tilde{\mu})^2}{l}} \]  

is a unbiased estimation of \( \delta \).

Next we draw a random sample \( \tilde{d} \) from the normal distribution \( N(\tilde{\mu}, \tilde{\delta}^2) \) and we use it as our updated one-hop distance estimation in the next iteration of our algorithm.

3.5 Our Algorithm

Each node \( i \) in the network runs a number of iterations of our algorithm to find its location, and in order to make it work we need to consider the following issues:

- **Initial conditions:** Initially, node \( i \) is assigned location \((x_i, y_i)\) randomly within the network space.

- **Location update:** The recalculation of location for node \( i \) is performed periodically. The duration of each iteration is controlled by a timer \( \tau_i \) for node \( i \). At the beginning of each iteration, timer \( \tau_i \) is set. Within the timer period, each node sends out the HELLO message to inform its neighbors its ID and its current estimated location and receives HELLO messages from neighbors containing their ID’s and locations. When the timer \( \tau_i \) expired, each node recalculate its location and sends out updated HELLO message to start a fresh iteration.

- **Stop criteria:** Calculation terminates when either of the following criteria is met:
  - The number of iteration \( COUNT \) reaches a predetermined threshold \( MAX\_COUNT \).
In each iteration, we calculate the difference $DIFF$ between the current calculated location and the calculated location in the previous iteration. If this difference is less than a predetermined threshold $\epsilon$, then we stop.

In a summary, our stochastic localization algorithm is:

**Algorithm 1** Stochastic Localization for WSNs

**Require:** Initial location of each node $(x_i, y_i)$. Timer $\tau_i$ is set.

for each node $i$ in the network do

while $COUNT \leq MAX\_COUNT$ and $DIFF \leq \epsilon$ do

    Calculate (update) location.
    Update one-hop distance estimation.
    Send HELLO message to neighbors with updated location.
    Receive HELLO messages from neighbors.
    Reset timer.

end while

end for
CHAPTER FOUR

PERFORMANCE EVALUATION

We implemented the proposed technique in ns-2 [38] as a routing-agent. Even though our technique has nothing to do with any form of routing, a routing-agent is used so that no additional information will be exchanged and the communication and computation will be kept to a minimum. The implementation of our algorithm utilizes the MAC and physical layers from 802.11, as there are no specific WSN implementations available.

Simulations were performed on several different topologies. We used two main categories: square-shapes topologies and C-shapes topologies. In addition to these two categories we augment the square-shapes topologies with obstacles and varying node-densities to further test our algorithm. We fixed the transmission range of nodes and varied the number of nodes and the enclosing network area. By doing this we can effectively control the average number of nodes within one node’s transmission range. We refer to this number as the node density. Table 4.1 summarizes the simulation parameters.

| ns-2 version | 2.29 |
| area         | 500x500, 1000x1000 |
| nodes        | 80, 100, 200, …, 600 |
| transmission range | 250 |

Table 4.1: Simulation parameters

The topologies used for the simulations are randomly generated for each experiment, so no two tests will be performed on the same topology. Our main metric for the simulations is accuracy. Similar to what is done in [25] the localization accuracy is represented by the average difference between the estimated positions and the corresponding actual positions as a percentage of the radio range \( r \). Equations 4.1 and 4.2 show how the accuracy \( \alpha \) is calculated. In 4.1, the Euclidean distance between the estimated position and the corresponding actual position of each node is calculated, where \( n \) is the number of nodes. In 4.2, we calculate the average distance and divide it
by the transmission range \( r \).

\[
d_{i'j'} = \sqrt{(x_i - x_{i'})^2 + (y_i - y_{i'})^2}, \quad i = 1, 2, \ldots, n. \tag{4.1}
\]

\[
\alpha = \frac{\sum d_{i'j'}}{n} \times 100. \tag{4.2}
\]

The actual and estimated positions are compared by finding the best linear transformation of the estimates onto the actual coordinates. We then calculate the average Euclidean distance between corresponding points in the actual map and the transformed estimated map and divide it by the transmission range \( r \).

The one-hop distance estimation technique we deployed directly affects the quality of localization as well as the computation overhead. To evaluate the localization quality, we present the accuracy of localization as a function of the network density and compare the accuracy of localization based on different one-hop distance estimation strategies.

4.1 Square-shapes networks

For square-shaped topologies we place the nodes randomly within the network space, which will yield a fairly uniform distribution of wireless sensor nodes. Figure 4.1 is an example of this topology (adapted from [21]).

We conduct our simulation in three stages. In the first stage, we model the one-hop distance as a deterministic value, while in the second stage, we model it as a random variable with fixed parameters. In the third stage, we fully apply our algorithm to model the one-hop distance as a random variable and we update the parameters on the fly; in another words, we model the one-hop distance as a random variable with parameters updated in each iteration of our algorithm.
Many authors model the one-hop distance as a deterministic value, for example, the transmission range is a very conservative estimation of the one-hop distance. First we run our algorithm with the one-hop distance \( d \) fixed to the transmission range \( r \). From figure 4.2 we can see that this estimation constantly yield very bad accuracy with inaccuracy over 50 percent. We also present the result in the same figure with \( d \) fixed to other deterministic values, say 125, and 70. We see that \( d = 125 \) yields much better results compared to \( d = 250 \) and it produces slightly better results compared to \( d = 70 \).

![Graph showing one-hop distance modeled as a fixed value.](image)

Figure 4.2: One-hop distance modeled as a fixed value

One interesting observation is that the curves for \( d = 125 \) and \( d = 70 \) have the “bowl” shapes and \( d = 125 \) reaches the bottom of the “bowl” earlier than \( d = 70 \) as node density increases. This is because we model the one-hop distance as a fixed value. So when the node density is such that the average distance between neighbors is close to this value, the algorithm produces best
results. When the node density is moving away from that point, the accuracy decreases since the average distances between neighbors becomes further away from the fixed value. This is one of the drawbacks of this modeling method, and we will see that our proposed algorithm is more adaptive to node density and yields better results constantly. We also include a reference line where \( d \) equals the true distance. Overall our observation is that if we fixed the one-hop distance to a deterministic value, then the accuracy will not be satisfactory. It is reasonable since in practice if nodes are placed randomly in the network space, the one-hop distance between two neighboring nodes will be quite uncertain.

Now we model the one-hop distance as a random variable with fixed parameters. Figure 4.3 shows the accuracy of this approach. We vary the distribution functions and the parameters for these distribution function to see how our algorithm works with these settings.

![Figure 4.3: One-hop distance modeled as a random variable with fixed parameters](image)

It can be observed that it yields the best results if the one-hop distance is modeled as a random variable satisfying the Normal distribution with mean value 125 and standard deviation 5. The uniform distribution produces the worst results as expected since nodes are evenly distributed in the network space. If we compare Figure 4.2 and Figure 4.3 we can see that modeling the one-hop distance as a random variable does not improve the accuracy significantly. But we have found that when the node density is really high in which case the one-hop distance is highly overestimated, the
modeling the one-hop distance as a random variable gives much better accuracy than modeling the one-hop distance as a fixed value since the previous one allows more flexibility. Even though in our final algorithm we model the one-hop distance as a random variable with updated parameters, these first two stage simulation indeed provide us further clue on our choice of distribution function for the one-hop distance estimation in our final algorithm.

In the third stage of the simulation, we apply our algorithm completely, say we model the one-hop distance as a random variable with parameters updated in each iteration. Again we vary the distribution functions trying to find the best one.

Figure 4.4: One-hop distance modeled as a random variable with updated parameters

Figure 4.4 illustrates the accuracy results. Not to our surprise, the Normal distribution works best among all the distribution functions and our algorithm yields fairly low inaccuracy even compared to other techniques that rely on anchor nodes information. We include two reference lines. One is based on the performance of using true distance at the same densities as our experiments and the other one is based on the performance of DV-hop [26] respectively in our accuracy figure. Here we should note that the data for DV-hop is adapted from other source and the comparison between our algorithm and DV-hop is not an “head-on” comparison since DV-hop utilizes anchor nodes and ours does not. The figure shows that both DV-hop and true distance achieve better accuracy than our algorithm. However our algorithm still obtains an acceptable localization accuracy
considering that it does not utilize any anchor nodes or ranging information.

4.2 C-shapes networks

In this section we also evaluate the performance of our algorithm in C-shaped networks, which is similar to what is done in [25, 26, 2]. The C-shaped network provides a good test for how localization techniques handle irregular topologies, where the hop distances between nodes can be very different from the actual Euclidean distances. This generally is a problem for any technique using any form of distance estimate based on connectivity, whether its range-free or not. In irregular topologies distance estimates might be severely overestimated compared to the actual Euclidean distance and it is important to overcome this problem. In Figure 4.5 we show an example of a C-shaped topology (adapted from [2]).

![C-shaped topology](image)

Figure 4.5: Example of a C-shaped topology. Adapted from [2]

To create this topology we start with the same topologies as in the previous section and remove a small portion of the nodes to create a network with the shape of a “C” and roughly 90% of the nodes remain in the network. By doing this we maintain roughly the same node-densities as in the previous section.

In Figure 4.6, we show the resulting accuracy of our algorithm applied to C-shaped networks and comparison of it with accuracies of other techniques.

As we can observe that our algorithm provides much better results than other deterministic approaches and our algorithm can also provide a better accuracy than DV-hop for these irregular
topologies. This is because the overestimation of distance affects the trilateration used in DV-hop and our localization algorithm is more adapted to the topology due to our updating one-hop distance estimation in each iteration.

4.3 Irregular node densities

It is highly likely that in any realistic scenario node densities will vary at different parts in a network. To test this we created an irregular topology. We use a random topology enclosed in network space of size $s$ by $s$ with $n$ nodes. We then add $\frac{2n}{10}$ new nodes in an $\frac{4}{5}$ by $\frac{4}{5}$ area within the original network space. By doing so we create a network in which a portion of the network has a significantly higher density than the rest of the network.

We can see from Figure 4.7 that our algorithm is not greatly affected by the irregularity in node density. Again this because the information gathered online is fed back to our algorithm and hence makes it more adaptive. Other deterministic techniques perform very poorly as expected. For example, the accuracy is about 40 percent worse than our algorithm if we model the one-hop distance as a fixed value and it gets even worse when the node density is higher. It is because one-hop distance is not the same at different parts of the network.
4.4 Obstacles

In any localization technique, obstacle is an important issue to address. Since just as irregular node-densities and irregular topologies, obstacles naturally present in realistic scenarios. However obstacles usually creates greater difficulties for any localization technique as one can no longer make the assumption that if two nodes are not adjacent or connected they will not positioned close to each-other. To test the performance of our algorithm under these difficult conditions, we preformed simulations on square-shaped topologies with two different types of obstacles: four line-shaped obstacles and one H-shaped obstacle.

In the case with four line-shaped obstacles, which is shown in Figure 4.8, we use four statically placed lines. Each line will hinder two nodes from communicating if the line between the two nodes intersect with the obstacle. Each obstacle has an average length of \( r \), where \( r \) is the radio range.

For the H-shaped obstacle, as shown in Figure 4.9, we placed the obstacle roughly in the center of the random topology, and hence provides a worse node-connectivity than the previous case.

As can be seen in Figure 4.9 the nodes that are most affected by the H-shaped obstacle is the nodes close to the horizontal line of the obstacle. These nodes will derive severely overestimated inter-node distances between themselves and nodes on the other side of the horizontal line. By
using the H-shape obstacle we are effectively creating an artificial perimeter and dividing the network into two parts with a fairly low connectivity in between them. Here we can clearly see that this obstacle will lead to distance overestimation.

As with the irregular density case, obstacles does not affect our algorithm very much. The use of one-hop distance estimation updating produces an algorithm that is less sensitive to obstacles and by utilizing only local information a node can in effect figure out its location effectively. Although inter-obstacle distances at times might be overestimated the overall accuracy is not significantly affected.

As can be seen from Figures 4.10 and 4.11 the resulting accuracy shows very litter changes from the networks without obstacles.
4.5 Complexity

Due to the energy and computational constraints of sensor nodes, any localization technique must be light-weighted. Our localization algorithm is deliberately designed to achieve fairly good accuracy while still to be computational efficient. In this section we show the convergence time and message overhead of our algorithm, which are two important metrics that reflect its efficiency.

Here the convergence time is proportional to the number of iterations before the algorithm terminates. From Figures 4.12 we can see that when the one-hop distance is modeled as a random variable, the computation time is always less then the computation time if the one-hop distance
is modeled as a fixed value. And when the node density is high, our algorithm takes much less iterations to converge. After a certain point, deterministic approach can never reach the accuracy tolerance so it will always reach the upper bound of number of iteration and the curve becomes flat after that point. But for our algorithm, as the node density increases, the convergence time decreases accordingly since each node in the network has more neighbors and hence it receives and process more information in each iteration of the algorithm to make quicker progress towards the convergence.

Figure 4.13: Communication overhead
Figure 4.13 shows the communication complexity. It can be observed that the number of messages exchanged among all the nodes in our algorithm is much less than the deterministic approach. Our algorithm is more efficient since it converges faster than other approaches and hence produces less communication overhead.

4.6 Scalability

The scalability of any localization technique is crucial. It is because that in a meaningful application of sensor networks, there will be a large number of sensors in the network. And when the size of the network is large, the distance estimation error, either this estimation relies on ranging information or not, will get propagated through the network and sometimes get cumulated for some techniques.

The inaccuracies in these estimations will affect the performance increasingly as the scale of the network is increasing. Our algorithm only count on the information provided by the one-hop and two hop neighbor. By doing so we can effectively control the error propagation since we only consider the local information. We illustrate the scalability of our algorithm in Figure 4.14.

![Graph](image)

Figure 4.14: Accuracy in large networks

We increase coverage area and the number of nodes while maintaining the same node density.
We have included a curve for DV-hop as a reference. Again the data for DV-hop is adapted from other source and for all DV-hop simulations 4 anchor nodes are used. For both our algorithm and DV-hop we see a significant degradation in the accuracy as the node numbers and network sizes increases, but as opposed to DV-hop, our algorithm does not depend on anchor-nodes. Our algorithm is more adaptive to node density due to the one-hop distance estimation scheme. In that sense, our algorithm is more adaptive than other techniques in which the one-hop distance is modeled as a fixed value.
CHAPTER FIVE
CONCLUSIONS AND FUTURE WORK

5.1 Concluding remarks
Localization is one of the most important challenge in wireless sensor networks (WSNs). Many techniques are introduced to estimate the sensor location in the network and some of these techniques do not rely on the ranging information, which gives rise to the problem of one-hop distance estimation. All these work model the one-hop distance as a deterministic value. While in this thesis we provide a feasible, accurate and efficient location discovery scheme for wireless sensor networks. Our scheme provides decent localization accuracy, and at the same time it does not rely on any ranging information. In our approach, we model the one-hop distance as a random variable and we keep updating the parameters of the random variable based on currently obtained information. Our simulation results show that our approach can achieve much better accuracy than other deterministic approaches. Our algorithm is computational efficient. The updating of random parameters is very light-weighted, but yet improves the final results significantly.

Our algorithm does not rely on any kind of specialized, extra capable nodes, or predetermined information about distributions, size etc. Our algorithm can be easily modified to facilitate other localization techniques in the first three phases of the algorithm. Our algorithm suggests that any localization technique that does not rely on ranging information can apply our one-hop distance estimation approach to improve its accuracy without increasing the total computational and time complexity significantly.

5.2 Future work
It would be interesting to further extend our algorithm if we utilize other localization techniques that do not rely on ranging information in the first three phases of our algorithm and to see how our technique may improve the accuracy of those approaches.
In our localization algorithm, we assumed that wireless sensor nodes are deployed on a planar (2-D) space. It would be also interesting to investigate our algorithm for localization in 3 dimensional space.
BIBLIOGRAPHY


