This is to certify that the
dissertation entitled

ADAPTING WIRELESS SENSOR NETWORKS TO
OBSTRUCTED AND CONCAVE ENVIRONMENTS

presented by

CHEN WANG

has been accepted towards fulfillment
of the requirements for the

PH.D. degree in Computer Science and Engineering

Major Professor's Signature

9/28/2007

Date

ADAPTING WIRELESS SENSOR NETWORKS TO OBSTRUCTED AND CONCAVE ENVIRONMENTS

By

Chen Wang

A DISSERTATION

Submitted to
Michigan State University
in partial fulfillment of the requirements
for the degree of

DOCTOR OF PHILOSOPHY

Department of Computer Science and Engineering

2007

ABSTRACT

ADAPTING WIRELESS SENSOR NETWORKS TO OBSTRUCTED AND CONCAVE ENVIRONMENTS

By

Chen Wang

The advances of electronic circuits make it possible to achieve intelligent computation in small devices at low cost, which enables the development of wireless sensor networks in recent years. Wireless sensor networks consist of large volume of sensors deployed in a field and have broad applications in collecting data from the physical world. In practice, wireless sensor networks are often deployed in complicated environments including obstructed or concave areas, where radio signals are interfered or blocked by obstructions, which leads to irregular communication patterns or concave network topologies. In this dissertation, we systematically investigate how system performance are affected by complicated environmental factors, and develop a suite of solutions in sensor localization and data communications that can be applied to sensor networks deployed in obstructed and concave environments.

Sensor Localization provides fundamental support for sensor network protocols and applications. However, localization results can be severely distorted by complicated environments, where distance measurements may have large errors because radio or ultrasound signals are blocked, reflected, or distracted by obstructions. To address these problems, we propose the virtual ruler approach to filter out the incorrect distances in the measurement step. We further propose the upper bound approach in

localization algorithms to eliminate the impact of incorrect distance measurements.

It is a challenging task to realize packet routing in a wireless sensor network, because only small routing states can be maintained in a sensor's limited memory space. Location aware routing has been proposed to realize scalable packet routing by greedy forwarding based on a small set of neighbors' positions. However, location aware routing has low routing success rate when sensors are deployed into obstructed and concave environments. To improve the routing success rate of the greedy forwarding, we propose the topology aware routing that uses the graph embedding to efficiently encode a network topology into small size routing states. We further improve the end to end routing performance by encoding the expected number of transmissions to small routing states.

Radio packets may be lost in transmission because radio signals are susceptible to environmental interference. We propose the receiver-centric protocol to realize reliable data transmission for sensor networks with high throughput and low overhead. The receiver-centric protocol utilizes the broadcast nature of radio signals and multihop forwarding of sensor networks. Since the communication throughput can also be reduced by radio interference among neighboring sensors, we further enforce channel access scheduling to CSMA protocol to mitigate channel interference. The channel access scheduling proposed in this dissertation is specially designed based on the tree structure that is naturally formed in data collection of a sensor network, and therefore outperforms other MAC protocols designed for general purposes.

ACKNOWLEDGMENTS

I would like to gratefully acknowledge the patient supervision of Prof. Li Xiao during this work. I thank Prof. Matt Mutka, prof. Abdol-Hossein Esfahanian, and Prof. Farhad Jaberi to serve as my committee. I also thank Prof. Rong Jin for his theoretical advisory. Particularly, I am grateful to Prof. Xiaodong Zhang, whose encouragement and guidance deeply impact my research philosophy as well as my career development.

I would also like to acknowledge all my collaborators of Yong Ding, Guokai Zeng, Yunhao Liu, Pei Zheng, and Kanthakumar Pongaliur for their help in technical discussion and experimental measurement.

Finally, I am forever indebted to my parents. Their expectation always encourages me to make progress.

TABLE OF CONTENTS

LIST OF TABLES

LIST OF FIGURES

CHAPTER 1

Introduction

The advances of electronic circuits make it possible to achieve intelligent computation in small devices at low cost, which enables the development of wireless sensor networks in recent years. A wireless sensor network consists of small sensors that are equipped with sensing devices, processors, and memory. Sensors are typically deployed in an environment in large volume and form an ad hoc network where neighbors communicate with each other through wireless channels. The primary function of a wireless sensor network is to collect data from a deployed environment to a central base station. Wireless sensor networks have the following unique characteristics: i) sensors are small and therefore can be deeply embedded into an environment; ii) sensors are cheap and therefore can be deployed in large volume; iii) sensors are self-organized and can automatically form a network after the deployment; iv) while each individual sensor has limited processor and memory resources, they can cooperate with each other to achieve substantial processing capability and storage capacity. These characteristics enable wireless sensor networks to be powerful tools that can closely observe the physical world with high fidelity.

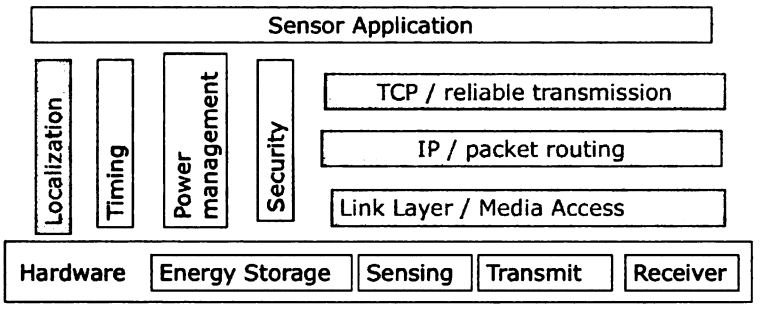


Figure 1.1 Architecture of a wireless sensor

Equipped with different sensing devices, wireless sensor networks have broad applications in collecting data from the physical world including battlefield surveillance, environmental and habitat monitoring, disaster recovery, structural monitoring, medical diagnostics, healthcare, asset tracking, etc. For example, sensors can be densely deployed in large volume to a battle field to monitor enemy activities, where numerous redundant sensors help the sensor network to be resilient to an enemy's attack. Sensors can also be deeply embedded into glacial layers to monitor the movement of glaciers, which is helpful to the investigation of the global warming phenomenon. In fire disaster, sensors can be rapidly deployed to the field and quickly collect temperature data for rescue teams. Because a sensor network can be self-operated for a long time without the attention of human being, sensors can be implanted into civil infrastructures like bridges to detect their structural defects. Equipped with cameras, sensors can also be deployed along streets to monitor traffic.

All these applications are built on the basic infrastructure of sensor networks, which have the functionalities of collecting, processing, storing, and transferring data. Sensors are equipped with sensing devices, processors/memory, and radio transceivers

to achieve these functionalities. Figure 1.1 shows the general design architecture of a wireless sensor, which includes the basic services of localization, timing, power management, security, and data communication. Many sensor network applications require sensed data to be labeled with location and time information, which necessitates the localization and timing services. Many applications demand the long life time of wireless sensor networks with limited energy resources, which can be achieved by power management that schedules sensors' activities for energy saving. Sensors may be deployed in a hostile environment, where security mechanisms are necessary to protect sensors against malicious attacks. Finally, sensors communicate with each other with low power transceivers through wireless channels, which necessitates a suite of energy-efficient networking protocols in MAC, network, and transport layers.

It is fundamental important to design energy-efficient sensor networks because sensors are powered by batteries and need to operate for a long time with the limited energy supply. Such a design principle, together with the constraints of low cost and small dimension, raises challenging problems to develop practical wireless sensor networks. Due to the limited available resources, in-network distance measurements based on radio or ultrasound signals are either error-prone or short-range, which makes it difficult to achieve accurate localization results for sensor networks. It is also a challenging task to achieve accurate and synchronized timing service for large scale sensor networks equipped with low cost clocks. Sensors need to be in sleeping status for most of the time to prolong their operation time and periodically activate themselves to achieve necessary sensing coverage and to maintain network connectivity. It is not a trivial task to optimally schedule sensors' activities such

that sensing coverage and network connectivity can be realized with minimal energy consumption. Sensors use low power wireless transceivers for communication, which incurs several problems: i) weak radio signals are susceptible to environmental interference and lead to unreliable data transmission; ii) sensors can only communicate with neighbors within a short range, and the long distance communication has to be realized through multihop forwarding, which requires the support of routing mechanism; iii) since sensors share wireless channels with neighbors, channel collision can easily happen in a densely deployed sensor network.

More challenging problems arise when sensors are deployed in complicated environments including obstructed or concave areas, which is often necessary due to the broad applications of sensor networks. For example, when sensors are deployed in forest for habitat monitoring, the communication between sensors can be blocked by trees or heavy bush. Neighboring sensors can also be blocked by buildings when sensors are deployed in streets for traffic monitoring. It is possible to deploy sensors along rivers or valleys for environmental monitoring, which have concave shapes. Complicated environments can significantly affect system performance of wireless sensor networks, which raises challenging problems in many aspects of their system design including sensor localization, timing service, sensing coverage, network connectivity, power management, and data communication. We aim to adapt wireless sensor networks to complicated environments. In this dissertation, we focus on addressing the following problems and developing solutions for sensor localization and data communication. First, in-network distance measurements will have large errors in complicated environments where radio or ultrasound signals are reflected or scattered by obstructions,

which can severely corrupt the localization results. Second, it is a challenging problem to realize packet routing in concave network topology where previously proposed location aware routing often fails to deliver a packet in the local minimum. Third, radio signals are susceptible to the interference of complicated environments, which leads to unreliable wireless channels and lost data packets. Fourth, sensors are often densely deployed and communicate with each other through short radio links, which are more resilient to interference of complicated environments. However, channel interference can easily happen among neighboring sensors with the dense distribution. In the following discussion, we detail how the system performance of localization and data communication are affected by complicated environments and introduce our proposed approaches that achieve accurate localization results and reliable data communications for sensor networks deployed in obstructed and concave areas.

1.1 Sensor localization

Most applications of sensor networks label sensed data with location information, which requires sensors to be aware of their own positions. Many approaches have been proposed to locate sensors with low cost [1]. The basic idea is to select a few sensors as beacons that can determine their own positions with the attached GPS receivers. The rest sensors can determine their positions by referring to nearby beacons. Sensor localization usually consists of two steps: distance measurement and geometric computing. Least squares fitting is commonly used in the second step to minimize the impact of distance measurement errors on the final positioning results.

Least squares fitting assumes that distance measurements in sensor networks are close to their true values, and sensors' positions can be accurately located by minimizing the difference between the measured distances and the distances computed from the assumed locations of the sensors. However, in a obstructed or concave environment, some distance measurements may have large errors, which cannot be simply reduced by least squares fitting, such that the final localization results are severely corrupted. This often happens in a realistic sensor network. We list possible cases as follows.

To reduce the cost and dimension of sensors, many localization approaches suggest reusing radio signals for measuring the relative distance between adjacent sensors. This is based on the observation that radio signals attenuate during their transmission such that their transmission distance can be inferred from received signal strength (RSS). However, since the radio signals can be strongly affected by environments, distances inferred from radio signals tend to be inaccurate and unreliable. Particularly, the distance measurement between a pair of sensors may have large errors if the line-of-sight path between the sender and the receiver is blocked by obstructions. In this case, radio signals will be weakened by the obstructions, which leads to erroneous distance measurements estimated by received signal strength. When ultrasound is used to measure distances between pairs of sensors, the measurements may also have large errors if line-of-sight paths are blocked between transceivers. In such a case, ultrasound signals are transmitted along reflected paths from a sender to a receiver, which is much longer than the distance of the straight line connecting the pairwise sensors.

Due to limited resources available from sensor networks, distance measurements in

sensor networks are often short-range. In order to obtain sufficient distance measurements, multihop based approaches were proposed to infer distances between any pair of sensors (including beacons) by approximating the lengths of the shortest paths to the Euclidean distances. The localization accuracy of multihop based approaches are built on the basis that the Euclidean distances between pairwise sensors can be well approximated by the lengths of the shortest paths. Such an approximation is achievable only when the shortest paths are close to straight lines, which requires sensor nodes are uniformly and densely distributed in a convex area. Although the uniform and dense distribution can be achieved through controlled deployment, it cannot be guaranteed that sensors are deployed in a convex area. A typical example is in habitat monitoring, sensors are deployed to complex areas such as valleys or rivers which may have concave shapes. The other scenario is that sensors are deployed in streets of urban areas where sensors may be separated from each other by buildings which results in concave network topologies. In such cases, the lengths of the shortest paths may not reflect the Euclidean distances correctly, because the shortest paths between some pairwise sensors have to detour along the concave areas and cannot be close to a straight line no matter how densely sensors are deployed.

Our performance evaluation shows that the large errors of incorrect distance measurements (defined as outliers) cannot be reduced by least squares fitting, which is the basis of most localization algorithms. In order to accurately locate sensors, it is necessary to exclude those outliers incurred by obstructed environments. However, it is a challenging task for sensors to identify those outliers because the resource constrained sensors do not have visualization capabilities to recognize obstructions. In

our research, we propose a suite of solutions to locate sensors in obstructed environments and concave areas. We first propose to exclude the outliers in the distance measurement step. We use mobile beacons to walk around obstructions and provide distance measurement service to pairwise sensors. Based on the multiple values obtained to the same distance, we exclude incorrect ones through statistical approaches [2]. We further propose to use the upper bound approach to exclude outliers in the localization algorithm [3][4][5]. Our upper bound approach is based on the observation that outliers, incurred by obstructions or concave structures, are always larger than their true values. The intensive performance evaluation shows that the upper bound approach can accurately locate sensors in both obstructed environments and concave areas.

1.2 Packet routing

The development of in-network storage and in-network process inspires intensive coordination among intelligent sensors. This necessitates point-to-point communication between any pair of nodes. Point-to-point routing, however, is a challenging problem in a wireless sensor network because the network may consist of thousands of nodes with limited resources to maintain routing states. Location aware routing (LAR) shows its potential in that packets can be delivered based on small constant size routing states. LAR proposes to forward packets in a wireless network according to nodes' geographic positions [6][7]. In LAR, a packet will be greedily forwarded to the next neighbor which is geographically closer to the destination, and finally delivered to the

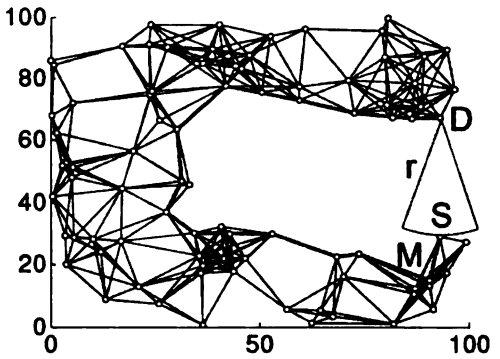


Figure 1.2 Inconsistence between the topological structure and geographic structure

destination after consecutive hop by hop forwarding. LAR is promising in that packet routing is realized through a *localized* algorithm that solely relies on the positions of the destination, the current node, and its immediate neighbors. The positions of a small set of neighbors compose a sensor's routing states, which can be easily fit to the sensor's limited memory.

The greedy forwarding of LAR, however, cannot guarantee packet delivery due to the problem of the *local minimum*, where a packet cannot find any neighbor which is geographically closer to the destination. This always happens when sensors are deployed in concave environments that contains voids. An example of local minimum is shown in Figure 1.2, where node S cannot find any neighbor that is geographically closer to destination D.

Many recovery solutions have been proposed to route a packet from the local minimum. For example, the face walking proposes to uses the *left (right) hand rule* to forward a packet toward the clockwise (counterclockwise) direction along the edge of a void whenever a packet is trapped in a local minimum [6][7]. The small scope

flooding has also been proposed to find the right routing path from the local minimum to the destination [8]. The recovery solutions often incur higher computation or communication costs than the simple greedy forwarding.

In our study, we seek to improve the routing performance of greedy forwarding without reliance on costly recovery strategies. we reveal the inherent tradeoff between the quality of routing performance and the size of routing states in wireless sensor networks. We suggest that the key approach to an optimal routing design is how to *efficiently* encode a network topology into small dimensional coordinates from which hop count distances between pairwise sensors can be *accurately* recovered. Based on the precisely hop count distance comparison, our proposed topology aware routing can assist the greedy forwarding to find the right neighbor that is one hop closer to the destination, and therefore achieve high success rate of packet delivery [9]. The intensive performance evaluation shows that the topology aware routing can achieve routing performance comparable to the shortest path routing while preserving the routing states as small as location aware routing. Since high routing success rate can be achieved in concave environments where voids are presented, topology aware routing provide a viable solution to realize point to point in a large scale sensor network deployed in a concave environment.

Not only the network topology, but also the radio communication channels of a wireless sensor network can be affected by the deployed environment. Since radio signals are susceptible to environmental interference, a wireless channel often demonstrates complex spatial characteristics in a obstructed environment. This spatial complexity is oversimplified by the disc model that are widely used in location aware

routing and the shortest path routing. The disc model uses a simple connectivity status to describe the wireless channels between pairwise nodes, i.e. pairwise nodes have the perfect reception channel if they are within the maximum transmission range of radio signals. In a realistic wireless network, neighboring nodes are often connected through unreliable wireless channels where packets may be lost due to the transmission error of radio signals. It is normal that packet loss rate is increased with the transmission range because the radio signals attenuate during their transmission. Both the location aware routing and the shortest path routing try to select a routing path with the least number of hops, such that each individual hop has a long transmission distance and high packet loss rate, which degrades the routing performance between the source and the destination.

We aim to improve the end-to-end routing performance of the greedy forwarding in complicated environments where radio signals are susceptible to interference. Instead of encoding hop count distances into virtual coordinates as in topology aware routing, we encode the number of expected transmissions for a packet to be successfully delivered between the source and the destination. Because virtual distances inferred from nodes' coordinates directly reflect the end-to-end communication channel quality, the greedy forwarding can guide a packet along the optimal routing path which uses the least number of transmissions to successfully deliver a packet from the source to the destination [10].

1.3 Reliable data transmission

Because radio signals are susceptible to environmental interference, packets may be corrupted or lost when transmitted through radio channels. Radio packets have high loss rate in obstructed environments because radio signals are interfered and blocked by obstructions. It is necessary to retransmit lost packet to achieve reliable data transmission in wireless sensor networks. Two basic mechanisms have been widely used for lost packet retransmission: the sequence-based mechanism and the tim-out mechanism. The sequence-based mechanism is mainly used by end-to-end recovery, in which the source labeled packets with continuous sequence numbers, and lost packets can be detected by the destination based on the discontinued sequence numbers of received packets. The sequence-based end-to-end recovery is not suitable for wireless sensor networks because it always recovers lost packets from source and therefore is not energy efficient. In contrast, the time-out mechanism is adopted by the hop-by-hop recovery, in which an intermediate node (sender) forwards a packet to the next hop (receiver) and waits for the acknowledgement (ACK) from the receiver for a certain period. The sender will retransmit the packet if the acknowledgement is not received. The hop-by-hop recovery is more energy efficient than the end-to-end recovery because it retransmits a packet just in the place where it is lost instead of from the beginning of the forwarding path.

The time-out based hop-by-hop recovery, however, incurs high overhead and reduces transmission throughput. First, data packets in wireless sensor networks usually have small size. In time-out mechanism, the receiver sends back the acknowledgement

for each small size data packet, which incurs high overhead that cannot be ignored in resource-constrained sensor networks. Second, the ACK itself my be lost due to the unreliable nature of wireless channels, which incurs unnecessary packet retransmission and consumes bandwidth resources. Third, packet may be lost because of channel congestion, in which packets are corrupted in collisions or dropped by the overflowed buffer in the receiver. However, the time-out mechanism can not distinguish channel loss from channel congestion and will blindly retransmit packets when ACKs are not received. This will intense channel congestion if all senders keep retransmitting lost packets.

We aim to improve the hop-by-hop recovery and seek an optimal design of retransmission mechanism with the following properties. i) It incurs small overhead to retransmit lost packets. ii) it only transmits lost packet when necessary and does not create duplicated packets. We propose the receiver-centric protocol to realize these two properties. First, the receiver-centric protocol uses the sequence based mechanism to detect lost packets. All the packets are labeled with continuously increased sequence numbers by the sender, and the lost packets are detected by the receiver based on the missing sequence numbers. The missing sequence numbers are sent back from the receiver to the sender through the virtual back channel, which is created by utilizing the broadcast nature of radio signals and the multihop forwarding of wireless sensor networks. When an intermediate node forwards a packet to the next node, the packet can be overheard by the previous node. Therefore, we can piggyback the missing sequence numbers to data packets. When a receiver continues to forward packets to the next hop, the missing sequence numbers can be sent back to

the previous sender through overhearing. Because we do not use extra packets to send missing sequences, the overhead of packet retransmission is minimized. Second, the receiver-centric protocol uses the negative ACk mechanism, i.e. the receiver notifies the sender only when it detects packet loss, and the sender retransmits packets only when the receiver requires. In this process, duplicated packets are avoided.

1.4 Channel access scheduling

In order to resist interference of complicated environments, sensor are usually densely deployed such that i) neighboring nodes can communicate with each other through short links with strong radio signals; and ii) short radio links have less chance to be blocked by obstructions. However, wireless channels can be easily interfered with each other in a densely deployed sensor network, which reduces channel utilization and transmission throughput. In order to reduce the channel interference, a suite of media access control (MAC) protocols have been proposed to schedule channel access among neighboring sensors. MAC protocols can be divided into two categories: TDMA protocols and CSMA protocols.

TDMA protocols assign time slots to neighboring sensors that share the same wireless channels. Since sensors only access the wireless channel in their own time slots, channel interference can be completely avoided by TDMA protocols. However, it is difficult to apply TDMA protocols in wireless sensor networks because i) they either require a global view of an entire network topology to assign time slots [11] or incur extra message exchange within two-hop scope [12]; and ii) sensors can only access

the channel in fixed time slots, which leads to idling time slots and reduces channel utilization. In contrast, CSMA protocols use the contention mechanism, in which each sensor listens to the channel first, and sends out packets whenever the channel is idle. Channel collision may happen when two sensors detect the channel idling and send out packets simultaneously. CSMA protocols solve the channel collision with the random back off mechanism, in which each sensor wait for a random time period before it attempts to send out a radio packets. Channel collision wastes bandwidth resources and reduces channel utilization. This can become a serous problem in a densely deployed sensor networks, where multiple sensors may send out radio packets through the same wireless channel.

To reduce channel collisions of CSMA protocols, we enforce channel access scheduling among neighboring sensors with our proposed receiver-centric protocol. The receiver-centric protocol operates as an overlay of CSMA MAC layer, and utilizes the tree-based topology, the unique data transmission pattern presented in sensor networks, to assist channel scheduling. The tree-based topology, naturally formed in sensed data collection, has the hierarchical structure and can be readily reused to schedule channel access. In the receiver-centric protocol, each intermediate node in the tree topology is viewed as a parent, which receives data from multiple sources of children. The parent manages channel access of its children. By utilizing the tree-based structure of data collection that is unique in sensor networks, the receiver-centric protocol improve the performance of CSMA that is originally designed fore general media access control.

1.5 Structure of the content

We present the dissertation as follows. We summarize previous work in sensor location, packet routing, data transmission, and channel access scheduling in Chapter 2. In Chapter 3, we demonstrate how to exclude outliers at the stage of distance measurement, the first step of localization. We further present how to use the upper bound approach to accurately locate sensors in obstructed and concave environments in Chapter 4. In Chapter 5, we detail how to realize optimal end to end packet routing with small routing states. We discuss the reliable data transmission in Chapter 6 and channel access scheduling in Chapter 7. We conclude the dissertation and discuss future study in Chapter 8.

CHAPTER 2

Background

It is often necessary to deploy sensors into complicated environments including obstructed or concave areas, where radio signals may be interfered or blocked by obstructions, which leads to irregular radio transmission patterns or concave network topologies. The complicated environments have severe impact on system performance and therefore should be carefully considered in system design of wireless sensor networks. In this dissertation, we develop a suite of solutions in sensor localization, packet routing, reliable data transmission, and channel access scheduling to adapt wireless sensor network to obstructed and concave environments. Before we describe our approaches, we summarize related work as below.

2.1 Sensor localization

Most applications of sensor networks require sensed data to be labeled with position information, which necessitates sensors to be aware of their own positions. The positions of sensors, however, cannot be directly measured by manual methods or simply acquired by GPS receivers because both approaches are costly when applied

to numerous disposable sensors. Moreover, sensors may be deployed to inaccessible areas such as volcanoes, which makes the manual measurement impractical. Sensors may also be deployed to indoor environments, wild habitats with heavy vegetation, or urban areas surrounded by skyscrapers, where GPS receivers may inaccurately or even impossibly locate sensors due to the bad signal reception. Considering the factors of cost, accuracy, and accessibility, it is necessary to seek a low cost solution to accurately locate sensors even in an extremely harsh environment. In the following discussion, we introduce sensor localization and its possible applications.

A simple approach is to infer a sensor's position through GPS, which measures distances from a sensor to multiple reference points in Satellites and calculates the sensor's position through triangulation computation. However, due to the low cost design constraint, it is prohibitive to equip GPS receivers in all sensors. A compromised solution is to deploy GPS receivers to a few sensors which are defined as beacons. The rest of sensors infer their positions based on their relative distances to those beacons. Based on this model, the sensor localization problem can be formalized as follows. Given a network graph $G = (V_m \bigcup V_n, E)$, the vertex set V_m defines the beacons set, the vertex set V_n defines the sensor set whose coordinates are unknown, and the edge set E defines all the measurable distances between pairs of vertex (i, j) where $i, j \in V_m \bigcup V_n$. The sensor localization is to recover coordinates of the vertex in the set V_n under the constraints of edge set E and beacon set V_m.

Despite its simple description in mathematics, sensor localization is a challenging task in engineering which imposes tight design constraints on sensor nodes with low costs, power saving and small dimensions. Under such constraints, current avail-

able distance measurement techniques based on ultrasound can only achieve accurate results in short-range, such that beacons are not globally accessible to all sensors especially when beacons are sparsely distributed. Consequently, the simple triangulation algorithm cannot be directly applied to locate all sensors because some sensors may not have sufficient beacons available as their immediate neighbors. Numerous solutions have been proposed for sensor localization. In the following discussion, we introduce the distance measurement techniques currently available for sensor networks in Section 2.1.1. In Section 2.1.2, we discuss why the inaccurate and short-range distance measurements raise challenges to sensor localization. We summarize representative sensor localization approaches that have been proposed before in Section 2.1.3.

2.1.1 Distance measurement methods

Following the sensor design principles of low costs and small dimensions, radio and sound signals are widely used to measure distances in sensor networks.

Radio based distance measurements

Radio signals are related to distances in that their signal strength attenuates during their propagation. Consequently, the transmission distance between a pair of sensors can be inferred from received signal strength (RSS) by the ideal radio propagation model $RSS \propto PS/d^n$, where d is the distance between the transceivers. Distances estimated from RSS, however, may have large errors because radio signals are susceptible to environmental interference. First, radio signals can be reflected, diffracted, and scattered by obstacles, which creates different transmission paths be-

tween transceivers. The actual received signals are the vector sum of all the radio signals received along different paths. The signal strength may be weakened when the waves of multipath signals are out of phase, or reinforced when the waves of multipath signals have the same phase. Second, radio signals will be attenuated by obstructions during their transmission. This phenomenon is called shadowing that also incurs measurement errors in the RSS approach.

We can also obtain the distance between a pair of sensors by measuring the time of flight (ToF) of radio signals from a sender to a receiver. Since the flight speed of radio signals is constant, the distance between a pair of sensors can be computed by multiplying the ToF of radio signals by their speed. Because radio signals transmit at an extremely fast speed, meaningful measurements can only be achieved by highly precise clocks that are synchronized between the sender and the receiver. The clock synchronization can be avoided by measuring the ToF of the round-trip of radio signals in the sender side. In the round-trip distance measurement, the signal process delay incurred by the receiver circuit needs to be carefully filtered. The clock synchronization can also be avoided by measuring the differences of ToF from a transceiver to multiple beacons. For example, sensors with GPS receivers can estimate their own positions by referring to the beacons in satellites. Here only beacons in satellites need to be equipped with synchronized and precise atomic clocks. The major sources of the measurement errors of the ToF approach is the multipath effect, the clock drift and clock resolution. In order to achieve accurate measurement, it is critical for the ToF approach to precisely identify the first arrival of radio signals that is received from the line-of-sight path between a pair of sensors. However, radio signals of the

line-of-sight path may be interfered by other radio signals transmitted along reflected paths.

Sound based distance measurements

Sound or ultrasound signals are suitable for distance measurement in sensor networks because accurate measurement can be achieved at relatively low costs. Since the speed of ultrasound signals is relatively slow (approximately 331.4m/s), their transmission delay are measurable by inexpensive clocks, which makes it possible to apply the ultrasound based ToF approach to low-cost sensors. The ultrasound based ToF approach, however, has three limitations in its distance measurements. First, the ultrasound based ToF approach can achieve accurate distance measurements only when a line-of-sight path exists between pairs of sensors. Similar to radio signals, ultrasound signals also have the multipath effects during their transmission, i.e. the receiver may receive ultrasound signals along multiple reflected paths besides the line-of-sight path. Since ultrasound signals spend more transmission time along the reflected paths than the "line-of-sight" path, the multipath effect can be filtered out by reading the earliest arrival signals. However, if the line-of-sight path is blocked between the transceivers, the distance measured by the ultrasound based ToF approach may have large errors. Secondly, ultrasound signals have unidirectional transmission. In order for all receivers around a transmitter to receive its ultrasound signals, ether multiple microphones are installed as in Figure 2.1 or a cone reflector is used as in Figure 2.2. Thirdly, the ultrasound ToF approach has short measurable range; because ultrasound signals attenuate fast in air and cannot propagate to a long distance.

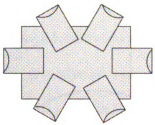


Figure 2.1 Multiple microphones installed in a sensor mote to achieve omni-directional ultrasound signal transmission

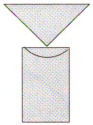


Figure 2.2 A cone reflector helps a sensor mote to reflect ultrasound signals to multiple directions

2.1.2 Challenges of sensor localization

The difficulty of sensor localization is to obtain sufficient positioning accuracy based on inaccurate and short-range distance measurements. A successful localization approach needs to address three challenges: 1) how to tolerate distance measurement errors; 2) how to overcome the limit of short-range distance measurements; 3) how to implement a localization algorithm at low communication/computation costs in a distributed fashion.

Tolerate measurement errors

In the triangulation algorithm, the position of a sensor is recovered from the constraints of distance measurements. As shown in Figure 2.3, when the measured distance between sensor S and beacon B_i is available, the position of sensor S is determined as the intersection point of the three circles with radius d_i. However, the simple triangulation algorithm may fail to find a feasible position for a sensor when distance measurements have errors. As shown in Figure 2.4, due to the measurement error of distance, the three circles will not intersect into one point, which means we

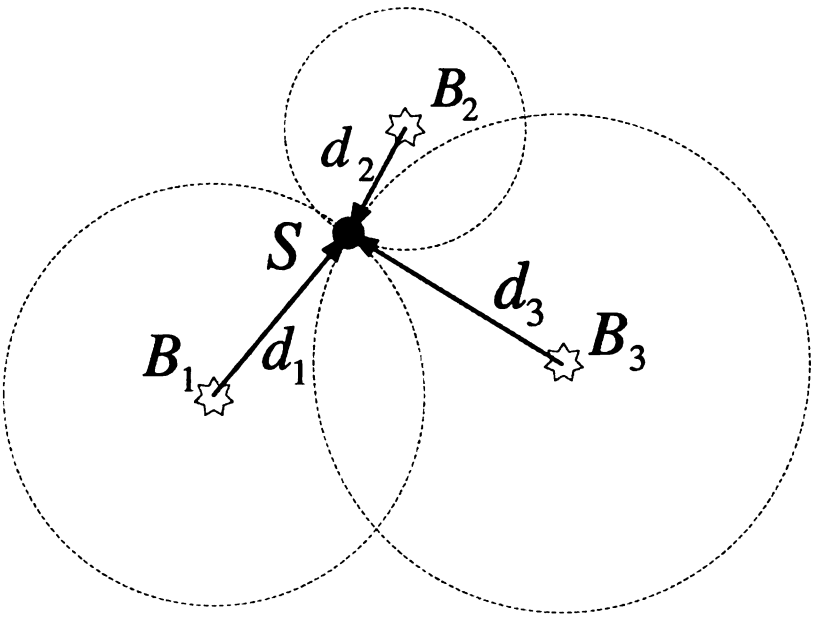


Figure 2.3 Circular constraints can intersect into one point with accurate distance measurements

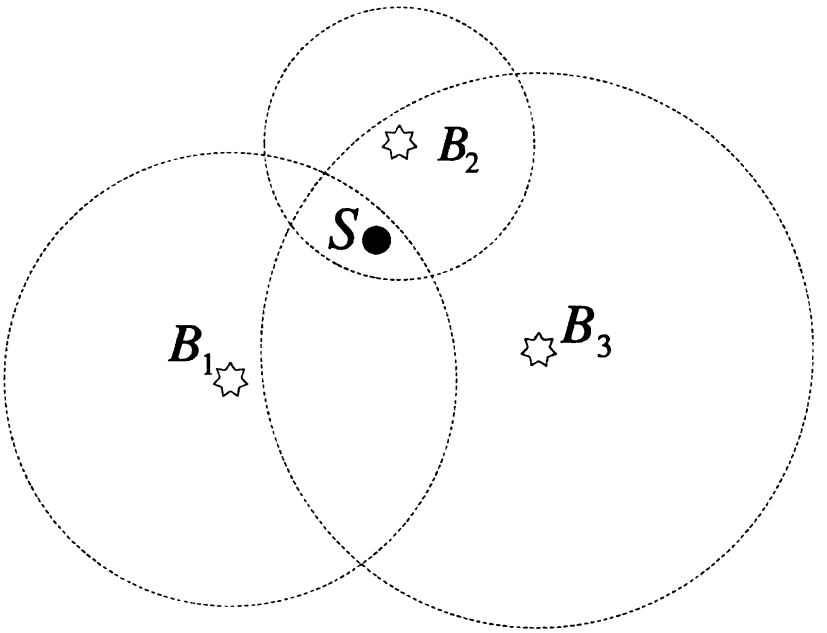


Figure 2.4 Circular constraints cannot intersect into one point because of measurement errors

cannot find a feasible position for sensor S whose distances to beacons can simultaneously fit all the measured distances. Since distance measurement errors are inevitable in practice, the objective of a localization algorithm is to minimize the impact of measurement errors and maximize the possibility that the estimated position of a sensor is close to its true position. Here the distance between the estimated position and the true position of sensor S is defined as the positioning error that is a critical metric to evaluate a localization algorithm. When sensors are deployed in a complex area abundant with obstructions, line-of-sight paths may be blocked between pairs of sensors. In such a case, distances measured either by RSS or ToF of ultrasound approach may deviate significantly from their true values. These distance measurements with large errors are regarded as outliers that could severely corrupt the final localization results. How to identify and exclude outliers is a challenging task because sensors usually have no visualization capability to detect obstructions.

Distance measurement range limitations

Because distance measurements in sensor network are short range (due to design constrains on energy consumption and size), there may be insufficient measurements to uniquely determine sensor positions. A simple example is given in Figure 2.5, which shows that the relative structure of the network cannot be known without sufficient distance measurements. In Figure 2.5, only four measurements are known, so the network structure could be any of an infinite set of non-equivalent quadrangles (two are shown). In Figure 2.6, all distance measurements between pairs of sensors are known, so the relative structure can be uniquely determined. But note that

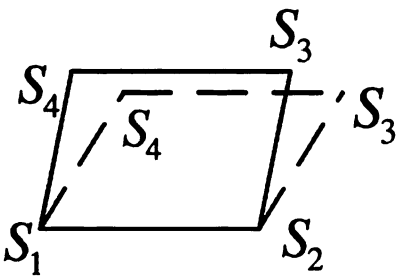


Figure 2.5 Network structure can be deformed

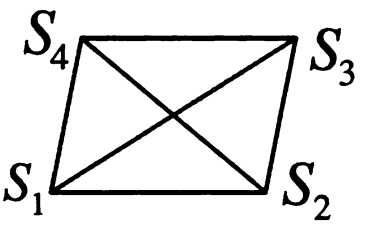


Figure 2.6 Network structure can be determined

at least one of the added cross-measurements will necessarily be larger than the side measurements, so it may not be available given sensor distance measurement limitations.

Communication and computation costs

Sensors have to be manufactured at low cost because they are disposable and intended to be deployed in large volume. This design constraint of low cost leads to limited hardware resources to support computation and communication. First, current MICA2 sensors use Atmega128 8-bit microcontrollers that have maximum 16 MIPS throughput at 16 MHz. The programmable flash memory of MICA2 sensors is only 128K bytes. Because of such limited hardware resources, a localization algorithm should avoid intensive computation before it can be successfully implemented in a sensor board. Second, because sensors with low power radio transmitters can only communicate with nearby neighbors, multihop forwarding is widely used for message transmission between pairs of sensors, which is the primary source of energy consumption for a sensor network. In order to save energy and extend the lifetime

of a sensor network, a good localization algorithm should avoid broadcasting massive messages to an entire network. Considering all the restrictions above, a practical localization algorithm should tolerate distance measurement errors, deal with short-range measurements, and incur low communication/computation costs. In the following sections, we will detail how current sensor localization algorithms provide solutions to those challenges.

2.1.3 Sensor localization algorithms

Due to the tight design constraints of low cost and small dimensions, sensors often lack sufficient hardware resources to achieve accurate and long-range distance measurements. To compensate for the limited measurement capability, numerous sensor localization algorithms have been proposed that can be categorized as area-based algorithms, distance-based algorithms, and mobile sensor based algorithms.

Area-based algorithms

In area-based algorithms, actual distance values are not involved in the localization algorithms, and the position of a sensor is estimated by picking up a point within an area. A typical example is the Centroid approach[13], which estimates a sensor's position as the centroid of the polygon formed by beacons that are within the radio transmission range of the sensor.

APIT[14] approach further improves the localization accuracy by utilizing the redundancy of available beacons, from which 3 beacons are selected out to form a triangle. The APIT approach can determine if a sensor is within the triangle by com-

paring the beacons' RSS with immediate neighbors. It is possible to obtain multiple triangles from different beacon combinations, and a set of triangles containing the sensor can be selected out. The sensor's position can be pinpointed to the intersection of all the containing triangles, which can be a small area when multiple triangles are involved in the intersection. As a result, the APIT approach can achieve better localization result than the basic Centroid approach.

Distance-based algorithms

Distance-based algorithms estimate sensors' positions from pairwise node distances, which are obtained through the media of radio signals or ultrasound with different measurement accuracy. All the distance-based algorithms recover coordinates of sensor nodes from distance constraints, such that pairwise node distances calculated from recovered coordinates are consistent with correspondent measured distances.

The basic distance-based algorithm is the multilateration, which is applicable when distances from a sensor to multiple beacons are available. To tolerate distance measurement errors, least squares fitting is proposed to minimize the difference between calculated distances and measured distances from the sensor to all available beacons:

$$\widehat{\mathbf{p}} = \arg\min_{\mathbf{p}} \sum (|\mathbf{p} - \mathbf{p}_i| - \widehat{d}_i)^2 \tag{2.1}$$

where $\mathbf{p}$ is the sensor's position to be estimated, $\mathbf{p}_i$ are beacons' positions, $|\mathbf{p} - \mathbf{p}_i|$ are calculated distances and $\widehat{d}_i$ are measured distances.

As we mentioned before, in-network distance measurements between pairwise sensors are often short-range. The basic multilateration algorithm may fail for some

sensors due to insufficient beacons available as their immediate neighbors. Two possible solutions to overcome the limitation of short-range measurement are recursive approaches and mutihop based approaches.

Recursive approaches[15][16][17] repeatedly apply the basic multilateration algorithm by converting sensors to beacons after their positions are determined. In recursive approaches, "converted" beacons can be propagated from an area which is close to the "starting" beacons to an area where the "starting" beacons are inaccessible. This makes it possible for sensors faraway from "starting" beacons to locate themselves with the aid of "converted" beacons. One problem of recursive approaches is that the localization error may accumulate and the final result may be severely distorted. To minimize the jeopardy of accumulated errors, recursive approaches are usually built on the basis of accurate distance measurements such as time of flight(ToF) of ultrasound. It is also pointed out in [17] to avoid large errors such as flip over when beacons are distributed closely to a straight line.

By approximating the length of the shortest path to the Euclidean distance between pairwise nodes, multihop based approaches[13][18][19][20] can infer distances between any pair of nodes in a connected network, hence all beacons are accessible to each sensor. Consequently, each sensor can locate itself through the basic multilateration algorithm. Multihop based approaches can only provide coarse localization result due to their approximate distance estimation. However, multihop based approaches are low-cost solutions because they suggest to reuse the communication radio signals to infer pairwise node distances. The low-cost character makes multihop based approaches ideal candidates for applications which have tight restriction on cost and

dimension while less demand on localization accuracy.

When a global view of all pairwise node distances is available, coordinate assignment can be achieved through centralized localization algorithms such as Mulitdimensional Scaling (MDS) [21][22][23] and Semidefinate Programming (SDP)[24][25]. Both approaches use optimization algorithms to search for coordinate assignment such that the distance constraints can be best fit. For instance, the MDS is to solve the following optimization problem which minimize the difference between all calculated distances and measured distances:

$$\widehat{P} = \arg\min_{P} \sum_{i,j \in V} (|\mathbf{p}_i - \mathbf{p}_j| - \widehat{d}_{ij})^2.$$

Here, P is the position matrix which contains all the sensors' positions to be estimated. $|\mathbf{p}_i - \mathbf{p}_j|$ is the calculated distance, $\widehat{d}_{ij}$ is the measured distance between sensor i and j and V is the vertex set representing all sensors. It is notable that the multihop approaches are also suggested in MDS[22] to infer distances between any pair of sensors since the classic MDS requires distance knowledge of all pairs of nodes.

Both the multilateration and MDS are built on the basis of least squares fitting, which fits the calculated distances to measured distances. The least squares fitting is based on the belief that all the distance measurements are close to their true values and have equal error distributions. However, when sensors are deployed in a concave environment, some of the distances estimated by the multihop based approach may deviate far away from their true values because the shortest paths have to detour along the concave shapes and deviate from straight lines. If we still use the least squares fitting algorithm to equally fit all the measurements, the final positioning

results will be deteriorated by those "bad" distance measurements. To address this issue in multihop algorithms, two approaches have been proposed before.

It is suggested in [26] to use the four nearest beacons instead of all of them in the multilateration localization. This is based on the observation that the distances from a sensor to the nearest beacons will be less affected by the concave shapes. However, it is still possible that the shortest path to the nearest beacon is affected by the concave shapes.

Proximity-distance map(PDM) is proposed in [27] to approximate lengths of the shortest paths to Euclidean distances correctly in anisotropic networks. In PDM, each sensor is assigned a coordinates in M-dimensional embedding space defined by the lengths of the shortest paths from sensors to all M beacons:

$$\mathbf{p}_i = [p_{i1}, \ldots p_{iM}]^T,$$

where p_{ij} is the length of the shortest path from the ith sensor to jth beacon. The objective of PDM is to find out sensors' coordinates in M-dimensional embedding space defined by Euclidean distances from sensors to all M beacons:

$$\mathbf{l}_i = [l_{i1}, \ldots l_{iM}]^T,$$

where l_{ij} is the Euclidean distance from the ith sensor to jth beacon. When the Euclidean distances from a sensor to all the beacons are available, the multilateration such as least squares fitting can be used to calculate sensor's location based on those Euclidean distances. PDM assumes there exists a linear transform between $\mathbf{p}_i$ and $\mathbf{l}_i$ such that $\mathbf{l}_i = T\mathbf{p}_i$. The linear transform T can be learned from beacons, where beacons' coordinates of both $\mathbf{p}_i$ and $\mathbf{l}_i$ are known.

The intuition of PDM is that the topology character of the entire network can be well represented by the beacons, since they are uniformly distributed in the network. Therefore, the linear transform T learned from beacons can be also applied to other sensors to transform their coordinates $\mathbf{p}_i$ to coordinates $\mathbf{l}_i$ defined by Euclidean distances.

Mobile sensor based approaches

Using mobile nodes in ad hoc wireless networks has been suggested by previous work [28][29]. Especially, using a mobile beacon to locate sensors has been proposed in [30], which assumes that the mobile beacon is equipped with GPS and know its absolute coordinate. When the mobile beacon moves around a sensor, the sensor can estimate the distances to various positions of the mobile beacon. Based on its relative distances to the mobile beacon, the sensor's absolute coordinate can be determined.

The mobile-assisted approach proposed in [31] suggest to measure distances between pairwise nodes without reliance on GPS signals. The mobile-assisted approach uses a single mobile beacon to obtain distances between pairwise sensors. In order to have sufficient constraints to calculate the distance between pairwise sensors, the mobile-assisted approach requires the beacon to move along a certain track. For example, the mobile beacon must move along a straight line for two consecutive steps in the two dimensional localization. Moreover, the mobile-assisted approach pays less attention to the incorrect distance measurement incurred by obstructions. Based on distances acquired by mobile beacons, the iterative least squares fitting is proposed in [32] to exclude an incorrect distance measurement in each iteration when a measured

distance significantly differs from the estimated distance.

2.2 Packet routing protocols

Due to its fundamental importance, packet routing in wireless networks has been studied for decades. In order to adapt to dynamic changing network topologies due to node mobility, DSDV[33], TORA[34], DSR[35], and AODV[36] have been proposed which either periodically broadcast routing updates or learn routing paths on demand by message flooding. On-demand routing protocols is further improved by utilizing multiple input multiple output technology in [37]. Dominating-Set Routing has been developed to limit the routing broadcast message within a subset of network nodes[38]. To further reduce or avoid the communication overhead of message flooding, Location aware routing (LAR) has been suggested to utilize nodes' geographical positions to discover routing paths[39][7][40][41][42] [43][44][45][46][47][48]. LAR is based on the observation that a wireless network's topological structure is similar to its geographical structure. Consequently, a packet can be forwarded one hop closer to its destination if it is greedily forwarded to a neighbor which is geographically closer to the destination. Nevertheless, the LAR may fail to deliver a packet in the local minimum when voids are presented, which is often the case when nodes are deployed in a concave area.

Numerous recovery schemes have been proposed to assist LAR to go through the local minimum. When local minimum happens due to the presentation of voids, the GPSR suggests to follow the right-hand rule to traverse around the edge circle of

the voids: a packet is always forwarded along the counterclockwise perimeter of a void[7]. BOUNDHOLE proposes to find all the points of local minimum by finding out the boundaries of holes(voids)[49]. The local minimum can be recovered by routing packets through paths connecting the boundaries of voids. Instead of locating and bypassing voids, fall-back mode has been proposed to route a packet to the beacon which is nearest to the destination whenever the local minimum is met[8]. The beacon thereafter initiates a small scope flooding to find the final destination. Despite their effectiveness, recovery schemes are more costly than simple greedy forwarding because either sophisticated algorithms are used to exploit geometric characteristics of a network or flooding is involved to search for destinations.

Due to the tight design constraints imposed in sensors, nodes's locations are often inaccurate or even unavailable. In order to be independent on the location information, several algorithms have been proposed recently to generate nodes' virtual coordinates from network topologies. GEM maps a mesh network into a tree structure such that a packet can be first forwarded up the tree until it reaches an ancestor of the destination. The ancestor thereafter relays the packet to the final destination. To shorten routing paths, short cuts are created in each layer of the tree such that a packet may be relayed to a destination before the common ancestor of the source and destination is reached[50]. Geographic routing without location information (NoGeo) in [51] proposes to use virtual coordinates to support greedy forwarding. This idea is further developed by a serial of virtual coordinates based approaches [52][53][54][8][55][56]. All these approaches proposed to construct nodes' coordinates from hop count distances instead of geographic distances to support the greedy for-

warding. However, local minimum still exists in these approaches because hop count distances cannot be accurately recovered from constructed coordinates.

Moreover, the hop count distance based greedy forwarding suffers the same problem as the geographic routing in that the quality of underlying wireless channels are not reflected in the routing decision. Both approaches tend to select a route with fewer intermediate forwarding nodes and therefore longer distance hops.

Several approaches[57][43][58][59] have been proposed to balance the forwarding distance and radio link quality, which either defines a threshold to exclude low quality radio links; or defines a new metric which can be maximized under the constraints of both forwarding distance and radio link quality. Nevertheless, the greedy forwarding based on defined local metrics may fail to find the optimal end-to-end routing path because the local metrics combined by the forwarding advance and the link quality between immediate neighbors can not reflect the global communication channel qualities.

The optimal routing metrics in a wireless network have been investigated in [60][61][62][63][64], which propose the ETX and RTT instead of the shortest path as the routing metric. The proposed metrics are incorporated into the on-demand routing such as DSR to discover the optimal routing path.

We propose to use embedding techniques to assign sensors with optimal virtual coordinates such that routing path quality can be accurately inferred from assigned coordinates. The embedding techniques have proved their success in Internet overlay networks[65][66][67]. All the work is motivated to create an overlay network which can exploit network proximity in the underlying Internet. Network embedding was

also suggested to discover sensor nodes' geographical positions from a network topology[22][21].

2.3 Reliable data transmission

To reliably transmit packets over lossy wireless channels, a group of transport protocols have been proposed to recovery lost packets based on the retransmission mechanism. Among the proposed approaches, the RMST[68] and RBC[69] approaches use time-out mechanism to detect lost packets. As we discussed in Section 1.3, the stop-and-wait nature of the time-out mechanism reduces channel throughput. On the other hand, the sequence-based mechanism is used by the PSFQ[70], GARUDA[71], and lazy loss detection [72] to detect lost packets and recover lost packets in a hop-by-hop fashion. The sequence-based mechanism uses the strict in-order sequence numbers to detect lost packets, which may incur packet delay and reduce channel throughput. This happens because packets cannot be forwarded by an intermediate node if any packet with the lower sequence number has not been recovered. Otherwise, unnecessary packets retransmission requests will be falsely invoked. The unnecessary retransmission requests can be partially reduced by the availability map proposed in GARUDA[71], in which the retransmission requests can only be initiated when the packets with missing sequences are available in an upstream core node. The above recovery approaches ensure high data transmission reliability through complex mechanisms to detect and retransmit lost packets, which is critical to applications where packet loss is intolerable. For example, the PSFQ and GARUDA approaches are

specially designed for download stream transmission from the sink to sensors. The control messages, such as reprogramming packets, have to be delivered from the sink to sensors with high reliability.

2.4 Channel access scheduling

Media access control (MAC) is critical to sensor networks where multiple sensors share the same communication channel for data transmission. To reduce channel collisions, a group of TDMA based protocols have been proposed to assign time slots among adjacent sensors [11][12][73][74]. However, TDMA protocol may either require a global view of the entire network topology, or incur massive message exchange between neighboring sensors. Moreover, TDMA is not flexible and scalable, which makes it impractical to be deployed in a changing sensor network comprising thousands of nodes. In contrast, the major MAC protocols that have been implemented in sensor networks are CSMA based protocols [75] [76] [77]. To reduce channel collisions while maintain the flexility of CSMA, hybrid solutions of CSMA and TDMA have been proposed. Z-MAC suggests to use CSMA when data transmission is low and switch to TDMA when data is transferred at high rate [73]. This approach helps a sensor network to achieve high channel utilization under both high and low channel contention. Funneling-MAC [78] proposes to use the TDMA in the region that is close to the sink while use CSMA for the rest part of a sensor network. This approach applies the TDMA to a small area that has highest channel contention in a sensor network, and keep the CSMA in rest sensors. Unlike these approaches, the receiver-centric

approach proposed in this dissertation is neither replace the CSMA, nor complement CSMA. Instead, we enforce a channel access scheduling overlay on the CSMA protocol, which retains the flexibility and scalability of CSMA while effectively reduce channel collisions.

CHAPTER 3

Mobile Beacons Based Distance Measurement for Sensor Localization

3.1 Motivation

Sensor localization involves two steps: distance measurement and localization algorithm. When distance measurements have large errors, it is difficult to use localization algorithm to achieve accurate results. In this chapter, we aim to filter out incorrect distance measurements in the first step, which can provide a good basis for localization algorithms. Particularly, we propose to use mobile beacons equipped with ultrasound transceivers to find correct distance measurements of pairwise sensors in an obstructed environment.

Despite its accurate results in the line-of-sight condition, the ultrasound ToF approach must address two challenges before it can be readily applied to a sensor network deployed in a complicated environment, especially in an indoor environment where ultrasound signals are reflected along multiple paths. The first challenge is that the ultrasound ToF approach has short measurable range due to the power constraint

of sensor nodes. The second challenge is that distance measurements will have large errors in an obstructed environment. As shown in Fig. 3.1, when the line-of-sight path is blocked between sensor S_1 and S_2, the distance has to be estimated from a reflected path which is much longer than the true distance between S_1 and S_2. In this chapter, we try to address these two challenges by using mobile beacons to measure distances between pairwise sensors deployed in an indoor environment.

To measure distance between pairwise sensors, we propose to use a pair of beacons attached to a small vehicle which randomly moves around in the deployed area. In our proposed approach, only beacons are equipped with ultrasound senders, and sensors are equipped with receivers that passively receive ultrasound signals broadcast by beacons. The distance between pairwise sensors can be determined if both sensors are within the beacons' ultrasound transmission range. When mobile beacons move around in the deployed area, it is possible to obtain sufficient distance constraints through which sensors' relative positions can be uniquely determined. Here, mobile beacons behave as a *virtual ruler* that wanders in the deployed area to provide distance measurement service to pairwise sensors as shown in Fig. 3.2. Compared with previous ultrasound based distance measurement approaches where a sensor acts as both a sender and a receiver, the virtual ruler approach can achieve longer distance measurement range such that more distance constraints are available to form a rigid network to uniquely determine sensors' positions. Such a long range distance measurement can be achieved without violation of the energy constraints, because only beacons are equipped with high power ultrasound senders and sensors are equipped with receivers consuming less energy. As a result, the virtual ruler approach addresses

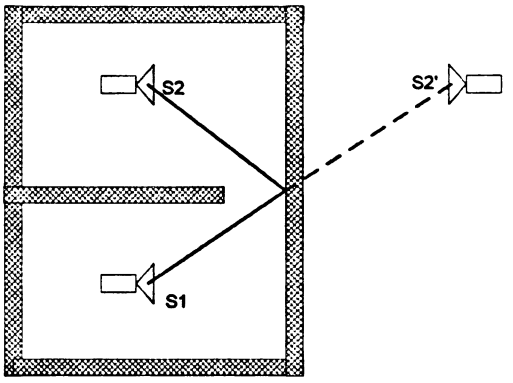


Figure 3.1 Ultrasound multipath effect in an indoor environment

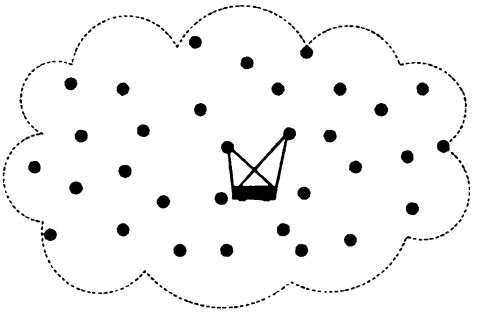


Figure 3.2 The virtual ruler moves around in the deployed area

the first challenge of short distance measurement range.

To address the second challenge that distances estimated along the reflected paths have large errors, we conduct intensive experiments to test ultrasound distance measurement in an indoor environment. We observe that a sensor's incorrect position, estimated from a reflected path instead of a straight line, is always mirrored to its true position. As shown in Fig. 3.1, the distance estimated between sensor S_1 and S_2 along the reflected path is equal to the distance between S_1 and S_2', the mirrored position of S_2. Based on this phenomenon, we further observed that incorrect distance measurements incurred by multipath effect have finite values that are virtual distances between sensors and their mirrors. This makes it possible to identify incorrect distance measurements through a statistical approach. When the virtual ruler moves around a pair of sensors, the distance between the pair of sensors can be measured by the virtual ruler multiple times from different perspectives. We observe two phenomena: i) a distance between the same pairwise sensors can be measured by

the virtual ruler more frequently if it is less affected by obstructions; ii) among all the measured values to the same distance, the correct measurement value is observed more frequently than incorrect ones. Therefore, we can identify correct distance measurements based on the distribution of measured values. Moreover, the confidence to a distance measurement can be quantified according to the distribution of its measured values. Based on the measurement confidence, the mobile distance measurement can be further combined with the recursive approach such that the distance measurement with higher confidence will have higher priority to be applied in the recursive approach.

The rest of the chapter is organized as follows. Section 3.2 describes the virtual ruler approach. Section 3.3 evaluates our proposed virtual ruler approach by comparing it with previous work. We summarize this chapter in Section 3.4.

3.2 Virtual ruler distance measurement approach

In this section, we first evaluate the measurement accuracy of the ultrasound ToF approach in the line-of-sight condition. After that, we test the ultrasound ToF approach in an obstructed environment. Based on our experimental observations, we propose the virtual ruler approach that can measure distances between pairwise sensors from multiple perspectives and filter incorrect distances statistically. How to combine the virtual ruler approach with the recursive approach is included at the end of this section.

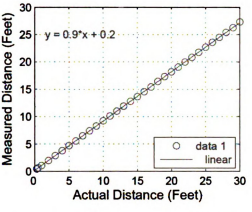


Figure 3.3 Ultrasound calibration experiment

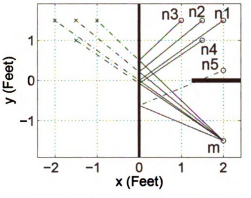


Figure 3.4 Experiment of multipath effect

3.2.1 Ultrasound distance measurement in the line-of-sight condition

Using ultrasound to measure distances in a sensor network has been extensively studied in previous work [79][80]. In this chapter, we repeat the experiment of distance measurement in a line-of-sight condition and compare the result with that in an obstructed environment. We use the same hardware as the Cricket system [81], which attaches two ultrasound transceivers to the MICA2 nodes developed by UC Berkeley [82]. The Cricket sensor nodes broadcast the radio signals and ultrasonic signals at the same time. The ultrasonic signals will reach a receiving node later than the radio signals due to their speed difference. By measuring the flight delay of the ultrasonic signals, the distance between a sender and a receiver can be estimated based on the constant speed of ultrasonic signals.

We measure the distances between a pair of Cricket sensor nodes in the line-of-sight condition. The pair of sensors are placed on platforms 2 inches above a hallway's floor with ultrasound transceivers facing towards each other. We vary the distances

between pairwise sensors from 0.3 feet to 30 feet. The experimental result is shown in Fig. 3.3, where x-axis represents the actual distances and y-axis represents the estimated distances. Fig. 3.3 shows that a clear linear relationship exists between the actual distances and estimated distances. The actual distances are slightly larger than the measured distances because sensors cannot be manufactured exactly the same as each other such that each sensor may use a slightly different time to process radio and ultrasound signals. However, the measurement errors caused by manufactory difference are a constant and can be easily calibrated through a fitting function between the actual distances and estimated distances. The fitting function for the pair of sensors used in this experiment is $y = 0.9 \times x + 0.2$. After calibration, the ultrasound approach can achieve high distance measurement accuracy. Methods for calibrating sensors in large volume has been studied in [80].

3.2.2 Ultrasound distance measurement in an obstructed environment

Despite its high measurement accuracy in the line-of-sight condition, the ultrasound approach may have large measurement errors in an indoor obstructed environment where multipath effects cannot be avoided. In an indoor environment, the ultrasonic signals may travel along multiple paths and arrive in a receiver at different time, though they start at the sender simultaneously. Nevertheless, if the line-of-sight path exists between pairwise sensors, the receiver can always identify the flight time along the shortest straight line by selecting the earliest arrival time of the ultrasonic signals. The situation becomes worse if a line-of-sight path does not exist between the sender and the receiver. As shown in Fig. 3.4, because the straight line between a sender

and a receiver is blocked by an obstruction, the ultrasonic signals have to travel a longer distance along reflected paths. Consequently, the estimated distance is much longer than the Euclidean distance between the pairwise sensors.

Because the ultrasonic signals along the reflected paths have the same format as those along the straight lines, it is impossible for a single receiver to identify incorrect distance measurements affected by obstructions. However, our experiments show that a sensor's false position estimated by the reflected path is always mirrored to the sensor's true position. In other words, the distance estimated along the reflected path between pairwise sensors is exactly equal to the distance between the sender and the mirror of the receiver, which is shown in Fig. 3.4, where the true positions of the receivers are represented as circles and the mirrored positions are represented as crosses. We verify this phenomena through a series of experiments. As shown in Fig. 3.4, we put the sender in the fixed position m while varying the receiver's position from n_1 to n_5. Because the line-of-sight path between the sender and the receiver is blocked by the obstruction, the ultrasonic signals have to travel along the path reflected by the wall. The distances of the reflected paths between the sender and various positions of the receivers are measured by the ultrasound, and we compare the measured distances with the actual distances between the sender and the mirrored positions of the receiver.

These experiments are demonstrated in Fig. 3.4, where the solid lines are the reflected paths from the sender m to the receiver's positions n_1 to n_4, while the dotted lines are the distances between the sender m to the mirrored positions of n_1 to n_4. The receiver at position n_5 cannot receive any ultrasound signals from m, because

Table 3.1 Comparison between the reflected distances and the distances from the sender to the receiver's mirrored positions

positions	measured distance	calibrated distance	distance to mirrors	Euclidean distance
$m - n_1$	4.95	5.28	5.00	3.00
$m - n_2$	4.59	4.88	4.61	3.04
$m - n_3$	4.17	4.41	4.24	3.16
$m - n_4$	4.36	4.62	4.30	2.55
$m - n_5$	N/A	N/A	N/A	1.75

even the signals reflected by the wall are blocked by the obstruction. Table 3.1 lists the comparison results between the measured distances along the reflected paths and the calculated distances between the sender and mirrored positions of the receiver. Table 3.1 shows that the measured distances of the reflected paths are much larger than the Euclidean distances between the sender and the receiver but close to the distances between the sender and mirrored positions of the receiver.

Because the incorrect distance measurement between pairwise sensors is always equal to the virtual distance from one sensor to the mirrored position of the other, we can conclude the incorrect distance measurements incurred by obstructions have finite values since the mirrored positions of a sensor are finite. This observation motivates us to measure the distance between pairwise sensors from multiple perspectives and filter out finite number of incorrect distance measurements through a statistical approach.

3.2.3 Measure distances through the virtual ruler

In order to measure distances between pairwise sensors from multiple perspectives, we attach two beacons to a small vehicle that randomly moves around in the deployed area. Because the distance between the attached beacons is fixed, we can easily infer

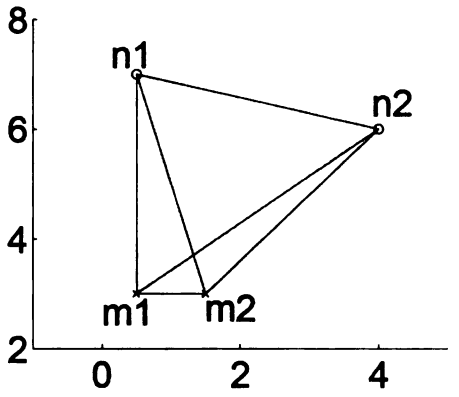


Figure 3.5 Using mobile beacons to measure distances between pairwise sensors

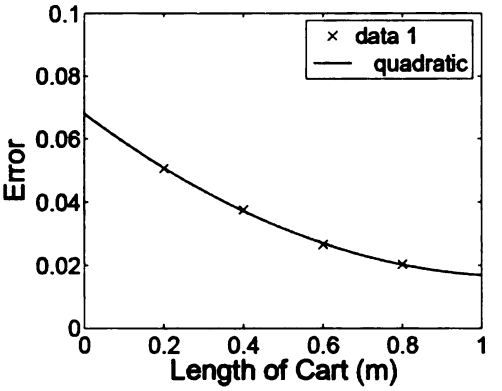


Figure 3.6 Impact of the fixed distance between mobile beacons on the measurement errors

the distance between a pair of sensors if the distances from the sensors to both beacons are known. Here, the mobile beacons behave as a virtual ruler that moves around to measure pairwise distances between sensors. Moreover, the distance between the same pair of sensors can be measured multiple times when mobile beacons move to different locations. Using mobile beacons to measure the distance between pairwise sensors is shown in Fig. 3.5, where node m_1 and m_2 are mobile beacons and node n_1 and n_2 are sensors. Because beacons' relative positions are fixed to each other, relative coordinates can be assigned to all the beacons. Here, we assign m_1's coordinate as $(0,0)$ and m_2's coordinate as $(0,L)$, where L is the fixed length between m_1 and m_2. By measuring distances d_{ij} from node n_i to beacon m_j, the relative position of sensor n_i can be calculated as below.

$$\widehat{\mathbf{n}_i} = \arg\min_{\mathbf{n}_i} \sum (|\mathbf{n}_i - \mathbf{m}_j| - d_{ij})^2 \tag{3.1}$$

Based on the relative position of n_1 and n_2, the Euclidean distance between the

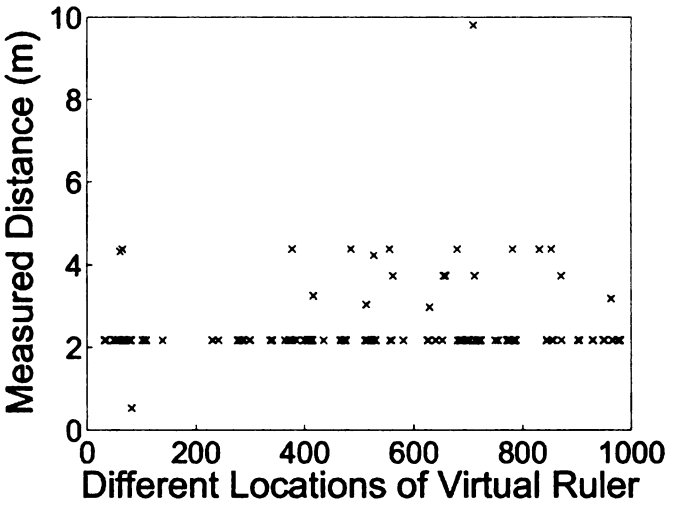


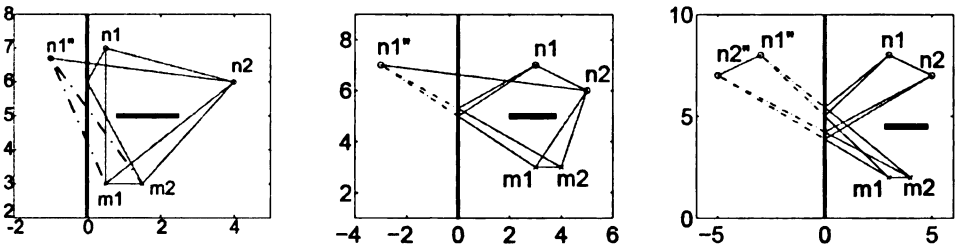
Figure 3.7 Distance measurement value distribution obtained by mobile beacons

pairwise sensors can be simply calculated as $|\mathbf{n}_1 - \mathbf{n}_2|$. We utilize the ultrasound's mono-directional transmission to avoid flip over error when two beacons are used. More robust results can be achieved when three beacons are laid out as a triangle. Intuitively, the error of the distance estimated by the virtual ruler is related to the fixed length L between pairwise beacons. Higher distance measurement accuracy can be achieved by the longer length L. To evaluate the relationship between the distance measurement error and the length L, we conduct a series of experiments by varying the length L. The measurement error under different value of L is shown in Fig. 3.6, which illustrates that sufficient measurement accuracy can be achieved with the relatively small length L. Consequently, we can implement the virtual ruler by attaching the beacons to a small vehicle.

Although high accuracy can be achieved by the virtual ruler when line-of-sight

paths exist from pairwise sensors to both beacons, the distance measurement may have large errors in an obstructed environment where some of the distances from pairwise sensors to beacons are measured along reflected paths. However, as we discussed in Section 3.2.2, the incorrect distance measurements caused by obstructions have finite values. We use the virtual ruler to measure the distance between a pair of sensors in a floor environment. The distribution of the measured value is shown in Fig. 3.7, where y-axis represents the measured values and x-axis represents the location from which the value is measured. Fig. 3.7 illustrates that only several values are obtained by the virtual ruler. The distance measurement to the same pair of sensors has finite values because incorrect distances are equal to the virtual distances between one sensor to the finite mirrored positions of the other sensor. To further explain this phenomena, we list three examples in Fig. 3.8, where m_1 and m_2 represent the mobile beacons and n_1 and n_2 represent sensors, and the distance between n_1'' and n_2 represents the incorrect distance estimation due to the multipath effects. Fig. 3.8(a), Fig. 3.8(b), and Fig. 3.8(c) show that 1, 2, and 4 line-of-sight paths are blocked by the obstruction respectively. An interesting observation is that when 4 line-of-sight paths are blocked by the obstruction in Fig. 3.8(c), both sensors are mirrored to the other side of the wall. As a result, the distance between the mirrored positions is equal to the distance between the original pairwise sensors. Therefore, the distance is correctly estimated by the mobile beacons, though they measure distances along the reflected paths to the sensors.

When the virtual ruler moves around in the deployed area, it will measure distances between pairwise sensors after each moving step. During the movement, the



(a) example a (b) example b (c) example c

Figure 3.8 Examples of distances estimated by mobile beacons in an obstructed environment

virtual ruler can measure the distance between a pair of sensors from different perspectives to obtain different values, among which the correct distance measurement is mixed with the incorrect ones incurred by obstructions. In the following discussion, we show how to identify a correct distance measurement by assigning a confidence value to the measured distance.

3.2.4 Evaluate the distances measured by the virtual ruler

In order to identify correct distance measurements, we conduct intensive simulations in various obstructed environments. When the virtual ruler moves around a pair of sensors, it can observe different distance measurement values $d_1, d_2, \ldots d_n$ between the same pair of sensors from different perspectives. Moreover, the same value d_i can be measured by the virtual ruler multiple times from different locations. Let k_i be the number of measurements of value d_i. Let N be the total number of measurements to the same pair of sensors. We have $N = \sum_{i=1}^{n} k_i$. Based on the distribution of the measured values, we observe two phenomena that are helpful in identifying correct distance measurements. First, a distance between a pair of sensors tends to

have a larger N if it is less affected by obstructions. This is because fewer distance measurements are observed by beacons when the pair of sensors are surrounded by obstructions and therefore are difficult for the virtual ruler to access. Second, among all the distance values measured to the same pair of sensors, the value with the largest number of measurements k_{max} has the highest probability to be the correct distance measurement. Because a typical indoor environment contains more open spaces than obstructions, pairwise sensors have higher probability of being observed by mobile beacons through line-of-sight paths than through reflected paths. Based on the observations above, we choose the value with the largest number of measurements k_{max} as the correct distance measurement between a pair of sensors. Moreover, we assign confidence C to a distance measurement according to N and k_{max} as $C = N + \lambda \times k_{max}$, where λ is the weighting coefficient. Based on the confidence of distance measurements, we combine the virtual ruler distance measurement with the recursive approach, in which the distance measurements with higher confidence will have higher priorities to be applied.

3.2.5 Combine the virtual ruler distance measurement with the recursive approach

In the recursive approach, a few sensors are pointed as beacons whose positions are determined through out-of-band approaches such as manual measurements. In order to distinguish the beacons used by recursive approaches from the mobile beacons used by distance measurement, we define the former as the static beacons. Recursive approaches first locate sensors that are close to static beacons and iteratively convert

sensors to beacons after their positions are determined. Consequently, the static beacons can be propagated from the initial beacons to an entire deployed area such that all sensors can be localized. In the process of the recursive approaches, multiple candidates are often available to be converted to new beacons. Distance measurements between pairwise sensors are also redundant in a densely deployed sensor network such that sensors can be localized by using only partial distance measurements. The localization result will not be affected by the sequence of converting sensors to beacons and the distance measurement subset if all the distance measurements have the same error distribution. However, in an obstructed indoor environment, where correct distances are mixed together with incorrect distances, it is critical to choose the optimal sequence of the recursive process such that the incorrect distance measurements can be excluded from the distance measurement subset used in the recursive process. Based on the confidence assigned to distance measurements by our mobile beacons, the optimal recursive sequence can be approached as follows. In each step of the recursive approach, the candidate is selected such that we can maximize the confidence of all the distance measurements that are used to locate the candidate.

3.2.6 Moving strategy

We assume that the virtual ruler moves around as below. It follows a step by step movement pattern. For each moving step, the virtual ruler randomly selects a moving direction. However, our test shows that such a random moving strategy leads to a non-uniform distribution of the moving tracks. As shown in Fig. 3.9, the virtual ruler only visits the right part of the floor while ignoring the left part. To improve the

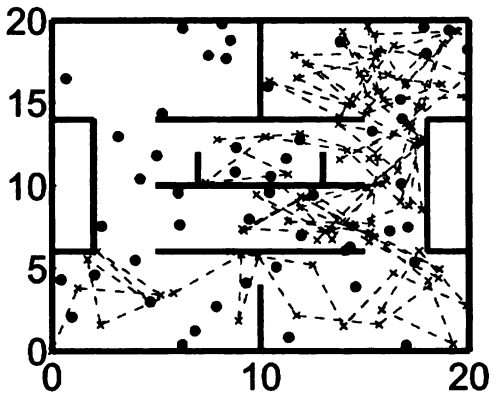


Figure 3.9 Beacons' moving tracks with random moving strategy

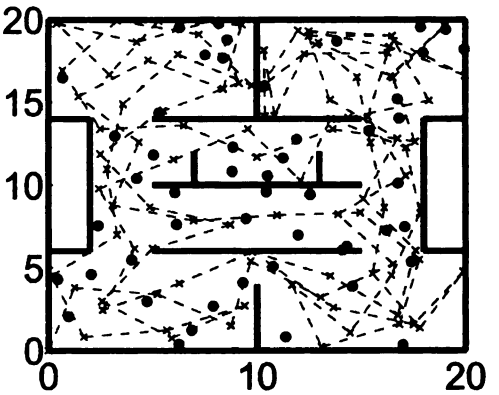


Figure 3.10 Beacons' moving tracks with enhanced moving strategy

coverage of the virtual ruler, we assume that the virtual ruler has certain intelligence such that they can communicate with nearby sensors to decide the moving direction. During the process of measurement, each sensor keeps recording how many times it has been measured. The virtual ruler queries neighboring sensors before it makes the moving decision. Based on the queried results, the virtual ruler moves towards the sensor that have least measurement times. The moving track of the enhanced moving strategy is shown in Fig. 3.10, where the floor is uniformly covered by the virtual ruler and all the sensors are visited.

3.3 Performance evaluation

We evaluate our virtual ruler approach in a $20m \times 20m$ square area where 50 sensors are randomly deployed. Two configurations of the deployed area are used in our simulation: i) 20 obstructions are randomly positioned (Fig. 3.11); ii) the square area is configured into a real indoor environment where rooms are separated from each

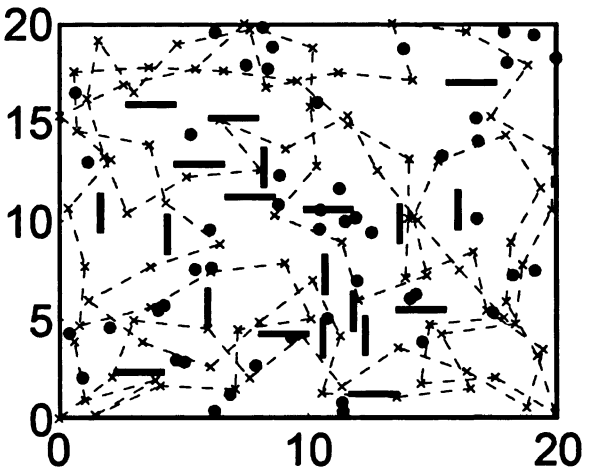


Figure 3.11 Sensors deployment with randomly distributed obstructions

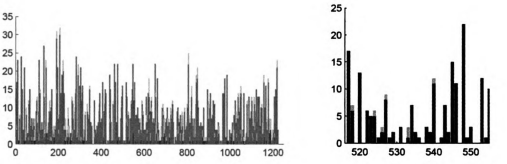
other by walls (Fig. 3.10). The virtual ruler moves around in the area and measures pairwise distances of static sensors within their measurable range. As illustrated in Fig. 3.10, the dashed lines show the trace of the virtual ruler that moves randomly throughout the region. The virtual ruler moves a total of 100 steps, and stops at each step to perform measurements. In this section, we evaluate both the distance measurement performance and the localization performance when combining the virtual ruler approach with the recursive approach.

3.3.1 Distance measurement performance of the mobile beacon approach

When the virtual ruler moves around in the deployed area, it can measure the distance between a pair of sensors multiple times from different perspectives. For the distance between the same pair of sensors, the virtual ruler may obtain different values, among which the correct value is mixed together with the incorrect ones. In order to statistically identify the correct value from the incorrect ones, we investigate

the characteristics of the distance measurement distribution through intensive simulations conducted in indoor environments. We record all the distance measurement values obtained by the virtual ruler and plot the distribution of measured values in Fig. 3.12, where Fig. 3.12(a) shows the full distribution of all the distance measurements and Fig. 3.12(b) shows an enlarged part of Fig. 3.12(a) for clear visualization. The height of each vertical bar represents the total number of distance measurements N between a pair of sensors. The vertical bar is further divided into two segments and the height of the the bottom segment represents the number of measurements k_{max} of the value d_{max}, which has the largest number of measurements among all the values observed by the virtual ruler. The sum of the number of measurements of all other values is represented as the length of the top bar. The bottom bar is colored in gray if it is the correct value; otherwise the bar is colored in black. Fig. 3.12 shows that the majority of bottom bars are painted in gray, which means the correct distance measurements tend to be observed by the virtual ruler more frequently than incorrect ones. Therefore, among all the measured distance values to the same pair of sensors, we can select the one with the highest observed frequency as the correct distance measurement.

We compare the distance measurement of the mobile-assisted approach [31], virtual ruler approach, virtual ruler approach with frequency analysis in both the indoor environment and random obstruction environment. The comparison results are shown in Table 3.2 and Table 3.3 respectively. The comparison shows that the virtual ruler approach has lower percentage of incorrect distance measurements than the mobile-assisted approach. Moreover, the percentage of incorrect distance measurements of



(a) Full distribution (b) Enlarged part

Figure 3.12 Distance measurement distribution

the virtual ruler approach can be significantly reduced by frequency analysis.

From Fig. 3.12, we can find a few counter examples where incorrect distance measurements fall to the bottom bars that are painted in black. However, the bar that contains the correct distance measurement at the bottom tends to be high. This observation motivates us to further exclude incorrect distance measurement by imposing a threshold to distance measurement. When the total number of distance measurements falls below the threshold, we regard the distance measurements as incorrect ones. By using the threshold strategy, the percentage of incorrect distance measurement is further reduced, as shown in Table 3.2 and 3.3.

3.3.2 Localization performance by combining the mobile beacon distance measurement with recursive approaches

The observations above shows that a distance measurement can be evaluated through two metrics: i) the total number of measurements to the same distance between pairwise sensors; ii) among all the measurements, the proportion of the value that

Table 3.2 Distance measurement comparison in an indoor floor environment

distance measurement approach	mobile-assisted	virtual ruler with one time measurement	virtual ruler with frequency analysis	virtual ruler with threshold strategy
total number of measurements	486	658	658	201
the number of incorrect measurements	129	146	100	2
percentage of incorrect measurements	26.54%	22.19%	15.2%	1%

Table 3.3 Distance measurement comparison in an area with randomly distributed obstructions

distance measurement approach	mobile-assisted	virtual ruler with one time measurement	virtual ruler with frequency analysis	virtual ruler with threshold strategy
total number of measurements	579	740	740	253
the number of incorrect measurements	159	166	104	3
percentage of incorrect measurements	27.46%	22.43%	14.05%	1.2%

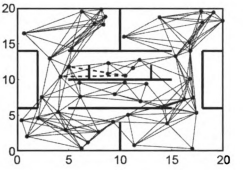


Figure 3.13 Distance measurement subset selected by virtual ruler

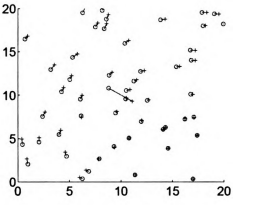


Figure 3.14 Applying virtual ruler to the recursive approach

has the largest contribution to the total number of measurements. Based on the two metrics, a confidence value can be assigned to each distance measurement such that we can rank distance measurements according to their confidences. With the ranked distance measurements, the virtual ruler distance measurement can be readily combined with the recursive approach to exclude incorrect distance measurements by giving a higher priority to a distance measurement with higher confidence. We then evaluate the combination of the virtual ruler approach and the recursive approach in the indoor floor environment. Besides the distance measurement errors incurred by the obstructions, we add Gaussian distributed random error to distance measurement. The simulation result is shown in Fig. 3.13 and Fig. 3.14. Fig. 3.13 shows that the virtual ruler successfully excludes most incorrect distance measurements and only two incorrect ones (painted with two bold dashed lines) are involved in the localization. In Fig. 3.14, the circles represent the true positions of sensors and lines represent the localization errors between true positions and the estimated positions.

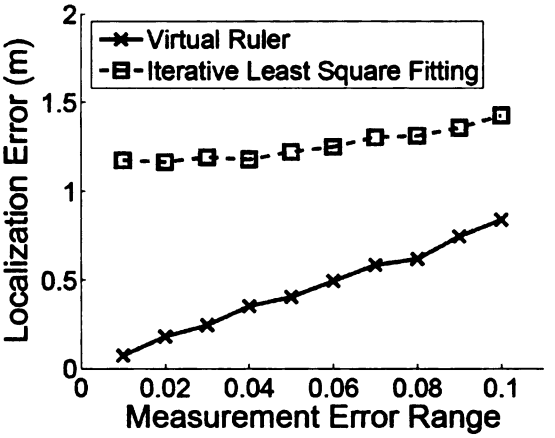


Figure 3.15 Compare the virtual ruler approach and iterative least squares fitting

We further compare the virtual ruler approach with the iterative least squares fitting algorithm in [32] that filters out incorrect distance measurements by iteratively applying least squares fitting. In each iteration of the iterative least squares fitting approach, the estimated distances between a sensor and static beacons are calculated based on the sensor's position estimated by least squares fitting. The distance measurement that differs most from its estimated value is excluded in the next iteration. Fig. 3.15 shows the comparison result in the indoor environment. We can see that the virtual ruler distance measurement combined with the recursive approach outperforms the iterative least squares fitting algorithm. The iterative least squares fitting algorithm fits all the measured distances equally such that the final estimated location is the averaged result of all the measurement. However, the least squares fitting algorithm does not guarantee that the estimated result will favor the correct distance measurements. If the estimated location favors the incorrect distance measurements, the correct distance measurement will be filtered in the iteration.

3.4 Summary

It is a challenging task to locate sensors in an indoor environment because the multi-path effects will incur large errors in distance measurements. The incorrect distance measurements, once mixed together with correct distance measurements, are difficult for localization algorithms to identify and exclude. In this chapter, we proposed to filter out the incorrect distance measurements in the first step of distance measurement. By using mobile beacons to measure distances between pairwise sensors from multiple perspectives, our proposed virtual ruler based distance measurement can statistically identify incorrect distance measurements, which provides a good basis for indoor localization algorithms. Especially, the virtual ruler based distance measurement can be further combined with the recursive approach such that distance measurements with higher confidence are selected with higher priority. The performance evaluation shows that the virtual ruler based distance measurement can achieve better localization results than previous mobile-assisted approaches.

CHAPTER 4

Locating Sensors in Complicated Environments with the Upper Bound Approach

4.1 Motivation

In the previous chapter, we use the mobile virtual ruler equipped with ultrasound transceivers to filter incorrect distance measurements between pairwise nodes. However, this approach utilizes the unique character of ultrasound based distance measurement, i.e. incorrect distance measurements have finite values and can be excluded through statistical analysis. As a result, the virtual ruler approach cannot be applied to some sensor network applications that prefer the radio based distance measurement to the ultrasound based distance measurement because the former has lower cost and smaller dimensions. The radio based distance measurement estimates distances between pairwise sensors from received radio signal strength (RSS), which is based on the phenomenon that radio signals attenuate during their transmission.

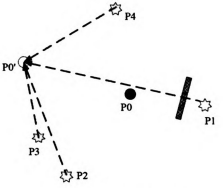


Figure 4.1 RSS-based distance measurement in an obstructed environment

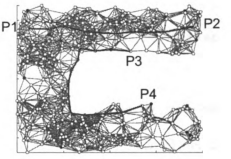


Figure 4.2 Multihop distance measurements in a C shape area

The RSS-based distance measurement often has large errors when radio transmission paths are blocked by obstructions, because the radio signals can be weakened by obstructions. An example is shown in Figure 4.1, where the RSS-based distance measurement between node P1 and P0 is much larger than its true value.

Because in-network distance measurements usually have short range, multihop approaches have been proposed to infer distances between any pairs of sensors, which approximate the length of the shortest path to the Euclidean distance between a pair of node. However, when sensors are deployed in a cave area, the distance inferred from multihop approaches will have large errors. An example is shown in Figure 4.2, where the length of the shortest path between node P3 and P4 is much longer than the Euclidean distance between node P3 and P4.

When distance measurements between pairwise sensors have small errors, sensor' positions can be accurately located by the maximum likelihood estimation (MLE) algorithm. MLE algorithm assumes that each measured distance is an approximation of the corresponding true distance. Thus, the optimal position for a sensor node is

found by minimizing the differences between the measured distances and the distances calculated from the assumed position of the node. More formally, this idea can be described as follows:

Let $\{P_i, 1 \leq i \leq n\}$ be the set of beacon nodes. Let P_0 be the node whose position is unkown, and $\widehat{P}_0$ be its estimated position. For each beacon P_i, let d_i be its measured distance to P_0, and $|P_0P_i|$ be its estimated Euclidean distance to node P_0. To minimize the difference between the measured distances and the estimated ones, we can refer to the optimization problem below, which is a typical least squares fitting problem.

$$\widehat{P}_0 = \arg\min_{P_0} \sum_{i=1}^{n} (|P_0P_i| - d_i)^2$$

Since the above objective function computes the aggregated difference between measured distances and the estimated ones, small errors in individual distance measurement will be averaged out, and as a result will not affect the accuracy of final estimated position significantly.

MLE algorithm, however, may fail to locate sensor accurately in complicated environments including obstructed or concave areas, where distance measurements based on the RSS approach or multihop approaches have large errors. In this case, a single distance with a large error can corrupt the final localization results. An example is shown in Figure 4.1, the estimation result P0' is pushed far away from its true position P0 due to the incorrect distance measurement P1P0.

In this chapter, we propose to use the upper bound algorithm to locate sensors in

complicated environments. The upper bound algorithm is based on the observation that incorrect distance measurements, either inferred from the RSS approach or the multihop approach, are always larger than their true value. Consequently, measured distances can be always viewed as the upper bound of the true values between pairwise sensors. By enforcing the upper bound constraints to the MLE algorithm, a sensors's estimated position can be tightly confined in the small area intersected by correct distance measurements. In the following discussion, we detailed how to apply the upper bound algorithm to the RSS-based approaches and the multihop approach respectively.

4.2 Apply the upper bound algorithm to RSS-based approaches

We have shown that RSS-based distance measurements may have large errors in obstructed environments and can corrupt the final localization results. To facilitate our discussion, we categorize distance measurement errors into two groups: *micro-error* refers to the case where the distance measurement is only slightly affected by environments, such as atmospheric conditions, and the error is within a tolerable extent; *macro-error* refers to the case when radio signals are severely distorted by environments, such as obstructions, and measured distances are significantly different from their true values.

In order to achieve accurate localization results in obstructed environments, we apply the upper bound algorithm to RSS-based approaches, which is named as Mul-

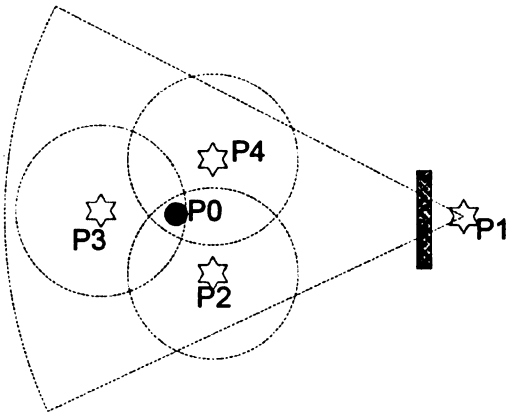


Figure 4.3 MRTP approach

tiscale Radio Transmission Power (MRTP) approach. Unlike the range-based approaches, which translate a distance from the Received Signal Strength (RSS)[83], the MRTP approach gradually increases a beacon's transmission range by increasing the scales of its transmission power, and the upper bound of the distances between a beacon and other sensors are determined by the minimal audible radio signals received by the sensors. By using the discrete scale levels of transmission ranges to estimate distance between sensors, we reduce estimation errors caused by the fluctuation in radio signals. Furthermore, unlike the range-based approaches, where measured distances are used as approximation for the true distances, In MRTP approach, measured distances only serve as upper bounds for the true distances that define a set of feasible locations for sensors. By avoiding approximating true distances with the measured distances, the MRTP approach will be resilient to the macro-error distance measurement, which usually is significantly different from the true distance.

Two possible techniques can be used to obtain the distance upper bound con-

straints between beacons and a sensor. The first approach is to quantize the RSS into multiple scales. Each RSS scale is mapped to a distance, which can be obtained through empirical test. When a sensor receives radio signals from a beacon, it compares measured RSS with empirical data. The distance upper bound constraints can be obtained by finding out the smallest RSS scale which is larger than measured RSS. The second approach is proposed when sensor nodes are incapable of measuring RSS. The main idea is to gradually increase the radio transmission power of sensors, until the radio packets can be "heard" by some receivers or the maximum transmission power is reached. Since each scale of transmission power corresponds to certain transmission range, given that a receiver receives a radio signal from a sender, it is able to infer, based on the scale of received signal, within which range it is from the sender. To be more precise, let P_0 be the receiver with unknown position, and P_1 be the sender. When P_0 receives a signal from P_1 at power scale level of m_1, the following inequality will hold for receiver P_0, i.e., $|P_0P_1| \leq f(m_1)$, where $|P_0P_1|$ stands for the Euclidean distance from receiver P_0 to sender P_1, and $f(m_1)$ is a function that maps the transmission power scale level of signals into its transmission range, which can be determined using empirical data. The above case can be generalized to multiple beacons, in which a different inequality constraint is introduced for every beacon P_i. Let $\{m_1, m_2, \ldots m_r\}$ be the set of signal levels that P_0 has received from beacons $\{P_1, P_2, \ldots P_r\}$. Thus, the following set of inequalities will hold for receiver P_0, i.e. $|P_0P_i| \leq f(m_i)$ $(1 \leq i \leq r)$.

However, the above set of inequality only determines the area that receiver will stay in. In order to pin down the exact location, we need to further decide which

position among the confined area is more likely to be correct than other positions. To solve this problem, we follow the Centroid approach, i.e., when a receiver detected a signal from a sender, it must be close to the sender. Thus, within the determined area, we believe the position that is closest to all 'audible' senders is most desirable one. This simple idea can be formulated into the following optimization problem, i.e.,

$$\widehat{P_0} = \arg\min_{P_0} \sum_{i=1}^{r} |P_0 P_i|^2 \tag{4.1}$$

$$\text{subject to } |P_0 P_i| \leq f(m_i) \; \forall 1 \leq i \leq r$$

The above optimization problem is a typical quadratic constraint and quadratic programming (QCQP) problem and can be solved efficiently using second order cone programming algorithms[84].

Compared to the range-free approach with uniform transmission power, the MRTP approach is more accurate, especially when the receiver is close to sensors. It is interesting to note that the Centroid approach can be viewed as a special case of Equation (4.1) when all the upper bounds $f(m_i) = \infty$. This result indicates that the centroid approach can be viewed as a MRTP approach under the assumption that the distance information is extremely inaccurate. However, in practice, by scanning the transmission power from low to maximum, distance measurement with reasonable accuracy can be achieved, at least for the beacon nodes that are close to the receivers. Under that circumstance, we will expect the proposed MRTP algorithm to provide more accurate estimation of a sensor's location than the range-free approach.

Compared to the MLE approach, the MRTP approach is more tolerant to large errors in macro-error distance measurements. This is because the MLE approach treats

each estimated distance as an approximation of the true distance and applies the maximum likelihood estimation to find a position such that all estimated distances are fitted well. In the case of obstructions that block between beacons and receivers, the estimated distances can be significantly far away from the true distances, which can severely degrade the accuracy in location estimation if we fit all measured distances equally.

In contrast, in our proposed MRTP approach for location recovery, we never treat the estimated distance (i.e., $f(m_i)$) as a good approximation of the true distance. Instead, an estimated distance only provides the upper bound as to which range the receiver will be away from the sender. Due to the competition between different constraints and the objective function, only certain constraints are effective to determine the receiver's position and the rest inequalities are ignored. To better illustrate this point, an example is shown in Figure 4.3, in which an obstruction blocks between P_1 and the receiver P_0. As a result, more transmission power is required to send the radio packets from P_1 to P_0, which results that the measured upper bound is much larger than the actual distance between P_0 and P_1. For the other three beacons P_2 P_3 and P_4, since there is no block of obstructions, more accurate upper bound estimation between them and the receiver can be achieved. Since the distance between the receiver P_0 and each sender is less than the estimated upper bound, for each sender, we draw a circle to represent the feasible area that receiver P_0 can stay. The final admissible area for P_0 is the intersection of all circles. As indicated in Figure 4.3, because of the loose upper bound estimation for the distance between beacon P_1 and P_0, the circle for P_1 has no impact in determining the final intersection. This

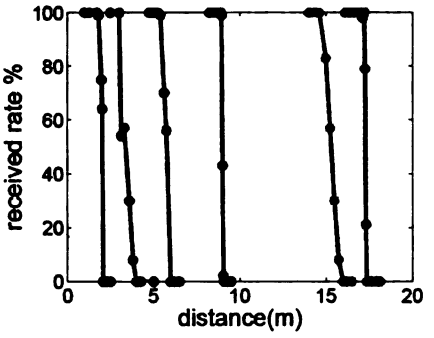


Figure 4.4 Packet received rate under different transmission distances

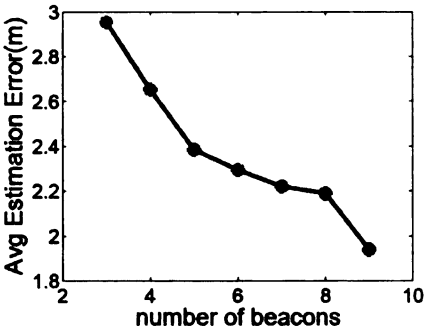


Figure 4.5 Small scale experiment results

analysis indicates that the proposed algorithm is able to filter out certain erroneous upper bound estimation, thus is more robust than the range-based approaches in an environment where multiple obstructions exist and correct distance estimations are difficult to obtain.

4.2.1 Experiment of the MRTP approach

To investigate the feasibility and performance of the MRTP approach in a real deployed environment, we conduct a serial of experiments, where sensors are deployed in a basket ball court. One sensor is attached to a laptop to accept inputs and display the localization results. The sensors used in the experiments are MICA2 sensor motes, the product from Crossbow company equipped with Chipcon CC1000 radio transceiver with the capability to adjust transmission power from -20 to 10dBm[85][86].

Range experiment on open area

To test the transmission ranges under different radio power scales in an open area environment, we put two sensors in the hallway, with one as the sender and the other as the receiver. To facilitate our measurement in a small area, only the sender is equipped with antenna. The sender is attached to the serial interface board of MIB510, through which commands are sent to the sender to set the scale of radio transmission power and start sending radio packets. The receiver counts the number of received packets for each transmission and saves it to the EPROM, which is read out later through the serial interface board. The ratio between the number of received packets and the number of sent packets is the probability for any radio packet to be successfully received within the experimental distance under certain transmission power. We sent 100 packets for each transmission, and repeated this experiment 20 times for different distances between the sender and the receiver. The whole process above was repeated for different transmission power scales. Figure 4.4 plots the received packet rate versus the distances between the receiver and the sender for different radio transmission power scales. Clearly, for each transmission power scale, there exists a sharp critical distance within which almost 100% of packets are successfully received; beyond the distance, almost no packet is received. We conclude that a clear mapping between the transmission power scale and its corresponding transmission range is available by seeking those critical points in the plot.

Evaluate the MRTP approach in a small scale experiment

To test the accuracy of the MRTP approach, we conducted a measurement experiment on a small scale physical sensor network. We deployed 9 beacons in a open area, whose positions were determined by a pseudo random generator. A sensor with unknown position was randomly placed in the same area. Through the laptop attached to the sensor, the"start to send" command is sent to each beacon, which sent out a serial of radio packets with different transmission power scales. The sensor found out the receivable packets with minimal transmission power scale, which was mapped to the transmission range based on the empirical data we collected in the experiment before. Based on the upper bound constraints from multiple beacons (from 3 to 9), the estimated position of the sensor is computed through our MRTP algorithm. The distance between the real position and the estimated position was recorded as the estimation error. The experiment was repeated 100 times by placing the unknown sensor to different locations. The estimation error averaged over 100 experiments was used as the final result.

Figure 4.5 shows that the average estimation error is reduced given more beacons are involved in the computation, and the estimation accuracy less than 2 meters can be achieved under the current MICA2 hardware support. We expect more accurate estimation if the scale of the transmission power can be more accurately adjusted.

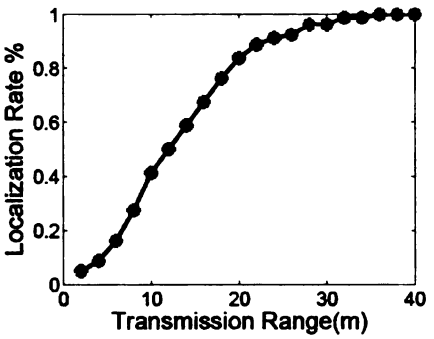


Figure 4.6 Localization rate of Centroid approach is improved when transmission range is increased

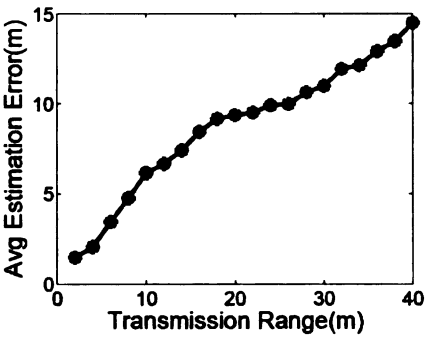


Figure 4.7 Average estimation error is increased when transmission range become larger

4.2.2 Compare the MLE, Centroid and MRTP approaches in large scale simulations

In the simulations below, we compare the performance of three methods, including the MLE, the Centroid, and the MRTP approaches. Radio signal is assumed to be the only method for determining the relative positions of sensor nodes. Nodes are randomly deployed in a 100m by 100m square area. The simulation is repeated multiple times with different number of nodes and beacons. The metrics defined below are used in our simulation:

- **Estimation Error:** the Euclidean distance between the estimated location of a sensor node and its actual position. For the i^{th} unknown node, it is defined as $\sigma_i = ((\widehat{x}_i - x_i)^2 + (\widehat{y}_i - y_i)^2)^{1/2}$, where $(\widehat{x}_i, \widehat{y}_i)$ is the estimated location and (x_i, y_i) is the real one.

- **Average Estimation Error:** the overall accuracy of a localization algorithm. It is defined as $\sigma = \sum_{i=1}^{N} \sigma_i / N$, where N is the total number of unknown nodes.

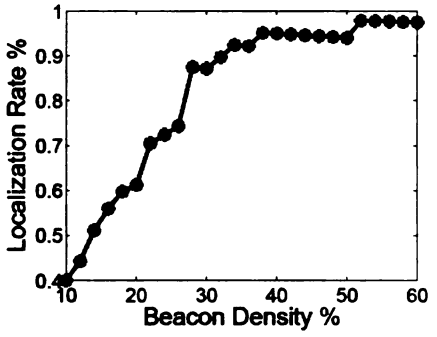


Figure 4.8 Localization rate of Centroid approach is improved when beacons are increased

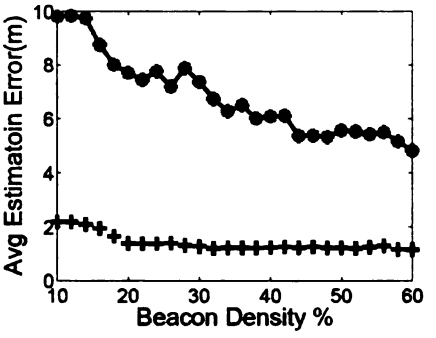


Figure 4.9 Average estimation error v.s. beacon density

- **Median Estimation Error:** the median value of ordered Estimation Errors of all estimated sensors.
- **Localization Rate:** the ratio of successfully located nodes to the total number of nodes.
- **Radio Transmission Range:** the maximum distance to which the radio signal can be propagated from a beacon.
- **Beacon Density:** the number of beacons per unit area.

Performance in open flat area

We compare the performance of centroid, MLE and MRTP algorithms in an open flat area, where the radio signal propagation is slightly affected by the atmospheric conditions such as temperature. This is related to the case where estimated distance can be *micro-erroneous*, but not macro-erroneous. We first fix Beacon Density and only vary the Radio Transmission Range. The simulation result in Figure 4.6 shows that

Localization Rate of Centroid algorithm is improved as Radio Transmission Range increases. This is because a larger radio transmission range will broaden the coverage area of beacons, and as a result, they become visible to more sensor nodes. Since for the Centroid algorithm to successfully locate the position of a sensor node, at least one beacon is required to be visible to the node, improving the visibility of beacons will help the Centroid algorithm to locate more sensor nodes. However, as shown in Figure 4.7, a larger Radio Transmission Range leads to an increase in the Average Estimation Error. This is because the measurement granularity of the Centroid approach is equal to the Radio Transmission Range. Thus, a larger Radio Transmission Range will result in a coarser estimation of positions for sensor nodes. Based on the above analysis for the Centroid algorithm, given a fixed number of beacons, improving Localization Rate will degrade the estimation accuracy. The above dilemma is due to the uniform transmission range used by the Centroid algorithm. In a random deployment, some sensor nodes are close to beacons, while others stay far away from the beacons. For the nodes with multiple beacons in their vicinity, a shorter radio transmission range is more preferable for higher accuracy. For other nodes far away from beacons, a larger radio transmission range is required for them to be located. The only way to improve both metrics in Centroid approach is to increase the Beacon Density, as proposed in [87]. Figure 4.8 and Figure 4.9 shows that both the Localization Rate and the estimation accuracy are improved as the number of beacons increases.

By using the non-uniform radio transmission range in each beacon, our MRTP algorithm successfully conciliates the contradiction between the Localization Rate

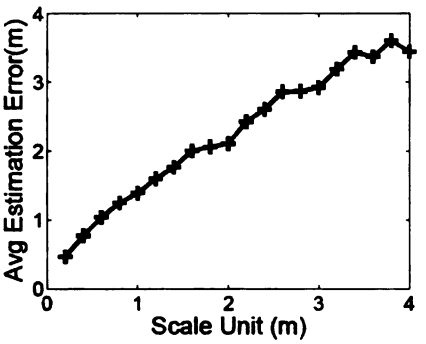


Figure 4.10 Estimation error vs. scale unit of MRTP

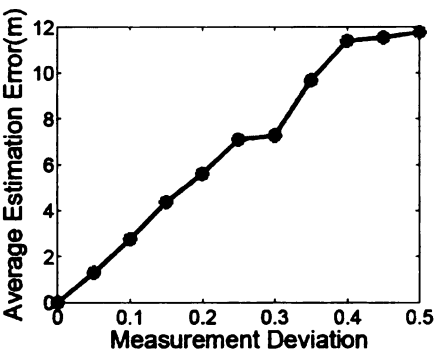


Figure 4.11 Average estimation error vs. measurement deviation of MLE

and the estimation accuracy. As illustrated before, the estimated location for an unknown node is confined to the intersection between multiple circles. When a sensor node is close to multiple beacons, an accurate estimation can be achieved since the intersection area is tightly bounded. By increasing the range of radio transmission of beacons, we are able to reach the sensor nodes that are far away from all beacons, thus their locations can still be computed. Figure 4.9 shows that the MRTP approach achieves significantly improved accuracy than the Centroid algorithm in estimating locations. It also indicates that the localization accuracy tends to saturate when a certain threshold of beacon density is reached. Our simulation also shows that all unknown nodes can be located when beacons' maximum transmission range is increased to a certain value. The MRTP algorithm outperforms the Centroid algorithm because the multi-scale radio transmission range provides more distance knowledge than the fixed radio transmission range.

Below we compare the performance of MRTP and MLE approaches in open flat areas, where the accuracy of distance measurement is critical to the accuracy of both

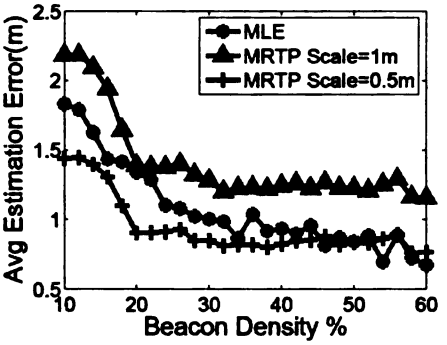


Figure 4.12 Impact of beacon density on average estimation error

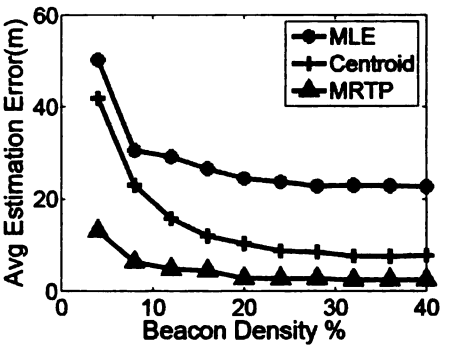


Figure 4.13 Performance comparison in an obstructed environment

localization estimations. We define the metrics below to represent the accuracy of distance measurement of MRTP and RSS, respectively.

- **Scale Unit:** the Scale Unit determines the length of increment in the transmission range when a beacon's transmission power scale is escalated to the next adjacent level. Here, we assume that the increment in transmission range is uniformly distributed.

- **Distance Measurement Standard Deviation (DMSD):** To simulate the error in distance measurement of the RSS that is caused by the radio's irregularity, a normal distribution is used to model the noise in measured distance, with a mean at the true distance m and a standard deviation $\sigma \bullet m$, where coefficient σ is the noise factor.

Figure 4.10 and Figure 4.11 shows that the Average Estimation Error increases when the Scale Unit and the coefficient σ of the DMSD increase. The simulation shows MRTP and MLE achieve the same accuracy when the Scale Unit is 1m and

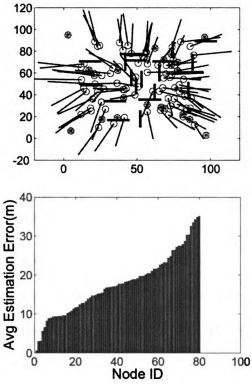


Figure 4.14 Performance of the MLE approach

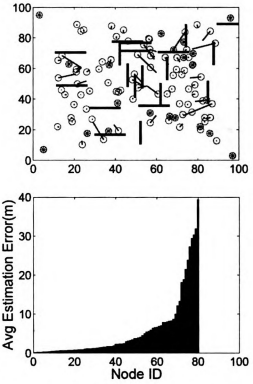


Figure 4.15 Performance of the MRTP approach

the coefficient is 0.1, which means the MRTP approach with Scale Unit of 1m has the same localization accuracy as the MLE approach whose Standard Deviation is 10% of the measured distance.

Figure 4.12 shows that the Average Estimation Error decreases, for both the MRTP and MLE approaches, as the number of beacons increases. Note that according to Figure 4.12, a steady decrease in the Average Estimation Error is observed for MLE, while the performance of MRTP appears to saturate when the number of beacons exceeds a certain threshold. This is because the MRTP approach treats the estimated distance as an upper bound of the true distance. This assumption is no

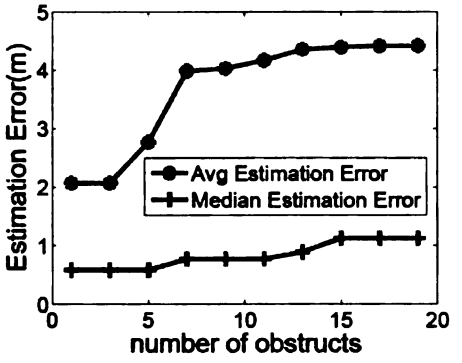


Figure 4.16 Impact of obstruction on estimation error

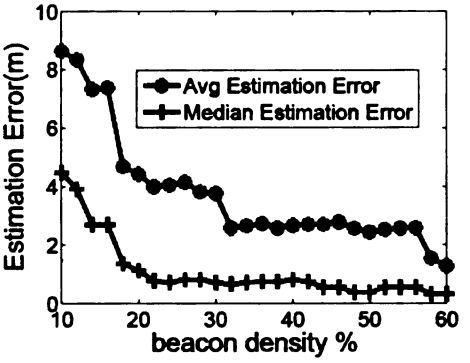


Figure 4.17 Impact of beacon density on estimation error

longer true when the true distance is within the range of the random noise in distance measurement. In contrast, since the MLE approach treats the measured distance as approximation of true distance, the noise in distance measurement is averaged out through solving the optimization problem in Equation (4.1). However, as we will see later in this chapter, we trade the accuracy of location estimation with its fault-tolerance. From the analysis above, we can conclude that: in the open flat area with fixed number of beacons, the MRTP approach can achieve more accurate estimation result than Centroid approach, and the estimation accuracy of both the MRTP and the MLE approaches are determined by the basic distance measurement accuracy.

Performance in an obstruction abundant environment

The simulation shows that in the flat open area, where the distance measurement is slightly affected by some random factors, the location estimation accuracy of both the MRTP and the MLE approaches is determined by the basic distance measurement accuracy. In such a case, the MRTP approach can be viewed as a multilevel quantized the MLE approach. It appears that the MRTP approach is different from the MLE

approach only in the format of distance measurement. In this section, we consider a realistic setup, in which multiple obstructions are placed to severely deteriorate the accuracy of distance measurements. In other words, the distance measurements can be macro-erroneous under this environment. The performance comparison of the MLE, the Centroid, and the MRTP approaches is shown in Figure 4.13, which illustrates that MRTP outperforms MLE in an obstruction abundant environment.

Figure 4.14 and Figure 4.15 show an example of comparing the MLE and the MRTP approaches, where each gray bar represents an obstruction, and each solid line represents the Estimation Error. The distribution of Estimation Errors is also shown at the bottom of corresponding graph, which plots the Estimation Errors of all nodes in the increasing order of the errors. The comparison demonstrates that almost all the estimation results of the MLE approach are severely corrupted, while more than half of the sensor nodes are located with high accuracy in the MRTP approach. Some of the nodes in the MRTP approach cannot estimate their positions accurately because almost all the beacons are invisible to those nodes; therefore the estimated node is not tightly constrained in a small area. Figure 4.16 shows that both the Average Estimation Error and the Median Estimation Error increase as the number of obstructions increases. This is because more beacons are hidden behind the obstructions. As shown in Figure 4.17, by increasing the Beacon Density in the deployed area, the estimation accuracy can be improved because more beacons are seen by the unknown nodes.

4.3 Apply the upper bound algorithm to multihop approaches

In this section, we further discuss how to apply the upper bound algorithm to multihop approaches. Multihop approaches have been proposed to overcome the limitation of the short-range distance measurement in sensor networks, which suggests to infer distances between any pair of sensors (including beacons) by approximating the lengths of the shortest paths to the Euclidean distances. Assisted by the multihop approaches, distances between any sensor to all beacons are available. This makes it possible to locate all sensors with a few beacons.

The localization accuracy of multihop based approaches are built on the basis that the Euclidean distances between pairwise sensors can be well approximated by the lengths of the shortest paths. Such an approximation is achievable only when the shortest paths are close to straight lines, which requires sensor nodes are uniformly and densely distributed in a convex area. When sensors are deployed in concave areas, the lengths of the shortest paths may not reflect the Euclidean distances correctly, because the shortest paths between some pairwise sensors have to detour along the concave areas (for instance C shape as shown in Fig. 4.2). To achieve accurate localization results in concave areas, we apply the upper bound algorithm to multihop approaches, which is defined as the improved multihop approach (i-multihop approach). We detail the i-multihop approach as follows.

4.3.1 Improved multihop approach

In this section we introduce our improved multihop (***i*-Multihop**) approach and explain how it can be extended to concave areas. To facilitate our discussion, we define symbols in the table below.

Symbol	Defination
$\mathbf{p}$	position of the sensor to be estimated
$\|\mathbf{p} - \mathbf{p}_i\|$	Euclidean distance calculated from a sensor's position $\mathbf{p}$ to a beacon's position $\mathbf{p}_i$
$\widehat{d}_i$	measured distance between a senor and the ith beacon
d_i	true Euclidean distance between a sensor and the ith beacon

Before we detail our i-Multihop algorithm, we will review the original multihop approach and investigate why it fails in concave areas.

Multihop algorithm: distance fitting approach

The key idea of multihop approaches is to discover a sensor network's geometry structure from its communication network topology. In multihop approaches, a sensor network is viewed as a connected graph $G = (V, E)$, where V is the vertex set representing sensors and E is the edge set representing links between a pair of sensors which are within radio transmission range. Multihop approaches infer the distance between a pair of sensors by approximating the length of the shortest path to the Euclidean distance. the length of the shortest path between vertex m and n is calculated as $L_{mn} = \sum l_i$, where l_i are the lengths of intermediate edges included in the shortest path. The value of l_i can be inferred from RSS which attenuates exponentially when the transmission distance is increased.

In the cases where RSS value is not available, multihop approaches infer the length of the shortest path from the average length per hop, which can be sampled by beacons as follows.

1. Distances D_{ij} between any pair of beacons can be evaluated from their known coordinates.

2. The number of hops H_{ij} of the shortest path between pairs of beacons can be inferred from the Dijkstra or Distance Vector algorithm.

3. The average length per hop to the *ith* beacon can be calculated as

$$h_i = \frac{\sum_{j \in V_m} D_{ij}}{\sum_{j \in V_m} H_{ij}},$$

   where V_m is the beacon set.

When the average length per hop is available, the length of the shortest path from a sensor to the *ith* beacon can be calculated as

$$L_i = h_i \times H_i,$$

where H_i is the number of hops of the shortest path from the sensor to the *ith* beacon.

The accuracy of multihop approaches is built on the assumption that the shortest path between a pair of sensors is close to a straight line, which is possible as long as the following assumptions hold:

1. Sensors and beacons are densely and uniformly distributed.

2. The network is maintained as a connected graph.

3. The deployed area has a convex shape.

The uniform and dense distribution of sensors can be achieved through a carefully controlled deployment. How to maintain *k-connectivity* of a network by selecting a proper transmission power is well studied in [88][89]. Therefore, it is not difficult to hold the first two assumptions. However, the third assumption cannot be guaranteed in practice, since the shapes of deployed areas are often out of human being's control.

As we pointed out before, the concave shape has severe impact on distance estimation of multihop approaches. Although a good approximation between the length of the shortest path and the Euclidean distance can still be achieved in some scenario (for instance P_1P_2 shown in Fig. 4.2), the lengths of the shortest paths may differ significantly from the Euclidean distances between pairwise nodes. This is because the shortest paths may be distorted by the concave area and cannot be close to a straight line. Such an example is shown as P_3P_4 in Fig. 4.2. To distinguish the distorted distance estimation from the rest, we divide the distances estimated by the multihop approaches into two categories: one is *incorrect* distance measurements which are distorted by the concave shape, the other is *correct* distance measurements which are not affected by the concave shape. Previous work[21][27] has shown that the localization results of MDS and mltilateration are severely corrupted by the large errors of *incorrect* distance measurements.

As we discussed before, to offset the inaccuracy of distance measurements, the maximum likelihood estimation (MLE) uses the least squares fitting to estimate sensors' positions by minimizing the difference between the calculated distances and

measured distances, which can only tolerant small measurement errors and cannot accurately locate sensors when large measurement errors are presented.

Multihop algorithm: the four nearest beacons

To eliminate the impact of concave shapes, it is proposed in [26] which uses the 4 nearest beacons instead of all of them. The intuition is that the shortest path from a sensor to the nearest beacon may be less affected by concave shapes. To facilitate our presentation, we denote this algorithm as the n-Multihop algorithm. The n-Multihop algorithm has two potential limitations.

1. The shortest path to the nearest beacon does not necessarily mean it is not affected by the concave shape. An example is shown in Fig. 4.2 where the path P_1P_2 is longer than P_3P_4, while the former is less affected by the concave shape and more close to its Euclidean distance.

2. By only using 4 beacons, some of the good distance measurements are eliminated from the localization, and therefore the redundancy of available beacons is sacrificed.

Now we start to detail our i-Multihop algorithm as below.

i-Multihop algorithm: upper bound approach

To facilitate the discussion, we first assume the in-network distance measurements between immediate neighbors are accurate, thus the mismatch between the shortest paths and straight lines connecting pairwise sensors is the only source of the distance

measurement error. We will relax this assumption in later discussion. Based on this assumption, we can have the following observation:

Observation 1: All the measured distances $\widehat{d_i}$ are no less than their true value d_i ($\widehat{d_i} \geq d_i$), because the length of the shortest path is always longer than the Euclidean distance of the straight line connecting pairwise nodes. Especially, the *incorrect* distance $\widehat{d'}_i$ distorted by the concave shape is much larger than its true value d'_i ($\widehat{d'}_i \gg d'_i$), because the shortest path deviates significantly from the straight line.

Based on the observation above, we model sensor localization in concave areas as below.

Model 1: Given a network graph $G = (V_m \bigcup V_n, E_s \bigcup E_t)$, the vertex set V_m defines the beacons set; the vertex set V_n defines the sensor set whose coordinates are unknown; the edge set E_s defines all the *correct* distance measurements $\widehat{d_i} \geq d_i$; and E_t defines all the *incorrect* distance measurements $\widehat{d'}_i \gg d'_i$. The sensor localization is to filter out the *incorrect* distance measurements E_t and recover coordinates of the vertex set V_n under the constraints of *correct* measurements E_s and beacon set V_m.

The challenge is how to recognize the *incorrect* distance measurements from the rest in the model where *incorrect* distance measurements are mixed together with *correct* distance measurements, which is impossible to be achieved by observing an individual distance measurement alone. However, we show that it is possible to eliminate the impact of *incorrect* distance measurements from the final localization result when multiple distance measurements are available. Instead of fitting distance

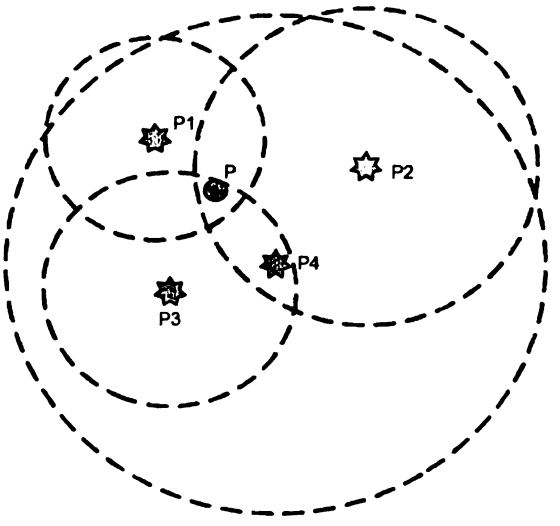


Figure 4.18 Sensor P is constrained in the intersection of the circular regions P_1, P_2 and P_3

measurements, we use upper bound constraints to locate sensors as below.

$$\widehat{\mathbf{p}} = \arg\min_{\mathbf{p}} \sum |\mathbf{p} - \mathbf{p}_i|^2 \tag{4.2}$$

$$\text{subject to } |\mathbf{p} - \mathbf{p}_i| \leq \widehat{d}_i$$

Remark 1 *The algorithm described in Eqn (4.2) can filter out incorrect distance measurements and achieve accurate localization results if and only if the number of correct distances is no less than 3.*

Based on our assumption $\widehat{d}_i \geq d_i$ we have distance measurement error $\delta_i = \widehat{d}_i - d_i \geq 0$. Without losing generality, for all the distances measurements between the senor and beacons, we assume $\delta_1 \leq \delta_2 \leq \ldots \leq \delta_m \ll \delta_{m+1} \leq \delta_{m+2} \ldots \leq \delta_n$. Here, δ_i $(1 \leq i \leq m)$ are small errors of *correct* distance measurements, and δ_i $(m+1 \leq i \leq n)$ are large errors of *incorrect* distance measurements.

Since $\widehat{d}_i \geq d_i$, we have $\mathbf{p} \in C_i$, where C_i is the circular region with origin $\mathbf{p}_i$ and radius $\widehat{d}_i$. We define the area of C_i as S_i, which represents the uncertainty of the

estimated position $\widehat{p}$. Based on the knowledge of $\widehat{d_i} \geq d_i$, we have $|\mathbf{p} - \mathbf{p}_i| \leq \widehat{d_i}$. The probability of estimated position $\mathbf{p}$ follows the uniform distribution:

$$p(\mathbf{p}) = \begin{cases} 1/S_i & \text{, if } |\mathbf{p} - \mathbf{p}_i| \leq \widehat{d_i}; \\ 0 & \text{, otherwise.} \end{cases}$$

When S_i becomes smaller, the probability density of $p(\mathbf{p})$ is increased. Thus, the uncertainty of the estimated position $\widehat{\mathbf{p}}$ is decreased. When multiple distance measurements are available, the true position of node $\mathbf{p}$ should be in the intersection of all circular regions C_i, i.e. $\mathbf{p} \in I = \bigcap_{1 \leq i \leq n} C_i$, and the probability of $\mathbf{p}$ follows:

$$p(\mathbf{p}) = \begin{cases} 1/S(I) & \text{, if } \mathbf{p} \in I; \\ 0 & \text{, otherwise.} \end{cases}$$

Here, the area $S(I)$ of intersection region I represents the uncertainty of the final estimation result. If $\delta_i \to 0$, $S(I) \to 0$, the estimated position $\widehat{\mathbf{p}}$ can be accurately pinpointed to its true position $\mathbf{p}$.

Let $S_t = \bigcap_{1 \leq i \leq m} C_i$. We have $S(I) \leq S_t$, which means the uncertainty will not be increased when *incorrect* distance constraints are added in the localization. Therefore, the *incorrect* distance measurements will not deteriorate the final localization result. The fundamental reason why the upper bound approach can tolerate the incorrect distance measurement is that its assumption $\widehat{d_i} \geq d_i$ is consistent with the observation $\widehat{d_i} \gg d_i$, while the assumption $\widehat{d_i} \approx d_i$ of the original distance fitting algorithm is inconsistent with the observation $\widehat{d_i} \gg d_i$.

An example of upper bound approach is shown in Fig. 4.18, where the distance

measurement between P and P_4 is much longer than its true value, which results in the large circle constraint. However, the *incorrect* distance measurement between P and P_4 has no impact on the final estimation result, since sensor P is tightly constrained by the constrained circular regions of P_1, P_2 and P_3.

The previous work[24][4] also suggests to use the upper bound constraints to locate sensors. Instead of using the local optimization where only distance to immediate neighboring beacons are used, the work [24] uses the global optimization of semidefinite programming. However, all approaches suffer the same problem, which requires that beacons are placed on the outside boundary of deployed area. Otherwise, the estimated positions will collapse toward the center. Such a phenomenon have been observed in previous work[25][26]. Below is a formalized description of the problem.

Problem 1 *Given beacons $\boldsymbol{p}_i$, there exists a polygon region P with all the vertices of $\boldsymbol{p}_i$. if sensor $\boldsymbol{p}$ is not within the polygon P ($\boldsymbol{p} \notin P$), the sensor's position $\widehat{\boldsymbol{p}}$ estimated by the upper bound approach will collapse toward the P.*

An example is shown in Fig. 4.19, where $\mathbf{P} \notin \triangle \mathbf{P_1P_2P_3}$. The sensor is constrained by a large intersection area, thus has high uncertainty. Position Pe estimated by the upper bound algorithm of Eqn (4.2) is attracted towards the $\triangle \mathbf{P_1P_2P_3}$ and deviates from its true position. This is contrast to previous example in Fig. 4.18, where sensor $\mathbf{P}$ is tightly constrained in a small intersection because beacons are distributed around the sensor.

To solve the collapse problem, we propose our solution as below.

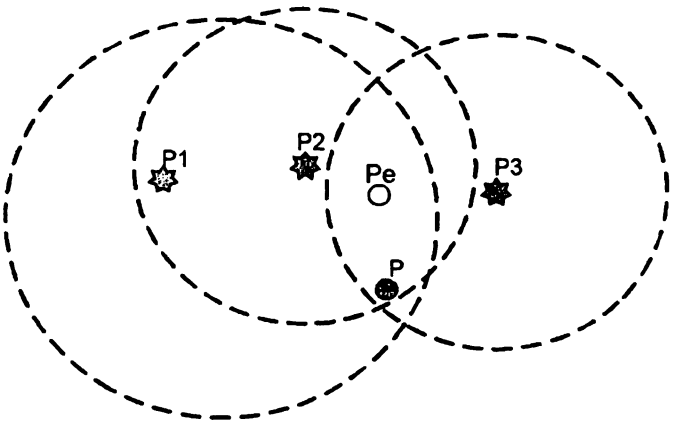


Figure 4.19 Collapsed result of the upper bound approach

i-Multihop algorithm: hybrid approach

In this section, we assume that sensors are densely distributed, thus the correct distance measurements, which are not affected by the concave shape, are close to their true values. Based on this assumption, we have the observation below.

Observation 2: All the measured distances $\widehat{d_i}$ are no less than its true value d_i ($\widehat{d_i} \geq d_i$). For all the *correct* distance measurements $\widehat{d_i}$, we have $\widehat{d_i} \approx d_i$. For *incorrect* distance measurements, we have ($\widehat{d'}_i \gg d'_i$).

Based on the Observation 2, we model the sensor localization as below.

Model 2: Given a network graph $G = (V_m \bigcup V_n, E_s \bigcup E_t)$, the vertex set V_m defines the beacons set; the vertex set V_n defines the position unknown sensor set; the edge set E_s defines all the correct distance measurements $\widehat{d_i} \geq d_i$ & $\widehat{d_i} \approx d_i$; and E_t defines all the incorrect distance measurements $\widehat{d'}_i \gg d'_i$. The sensor localization is to recover coordinates of vertex set V_n by filtering out the *incorrect* distance measurements E_t and fitting the *correct* distance measurements.

Based on the Model 2, we propose the localization algorithm below.

$$\widehat{\mathbf{p}} = \arg\min_{\mathbf{p}} \sum (|\mathbf{p} - \mathbf{p}_i| - d_i)^2 \tag{4.3}$$

$$\text{subject to } |\mathbf{p} - \mathbf{p}_i| \leq \widehat{d}_i$$

Remark 2: The algorithm described in Eqn (4.3) can filter out the incorrect distance measurements, and the final localization results will not collapse even when the sensor is not contained in the polygon formed by beacons.

The hybrid algorithm described in Eqn (4.3) combines the advantage of upper bound constraints and distance fitting. First, it uses the upper bound constraints to filter out the impact of *incorrect* distance measurements and pinpoint the estimated position to the intersection constrained by correct distance measurements. Second, it uses the distance fitting to fit *correct* distance measurements, which pushes the estimated position $\widehat{\mathbf{p}}$ toward its true position $\mathbf{p}$ and the final estimated position is not affected by the layout of beacons.

To facilitate the optimization computing, the algorithm in Eqn (4.3) can be simplified as below.

Since $|\mathbf{p} - \mathbf{p}_i| \leq \widehat{d}_i$, we have $|\mathbf{p} - \mathbf{p}_i| - \widehat{d}_i \leq 0$, thus $\widehat{d}_i - |\mathbf{p} - \mathbf{p}_i| \geq 0$. The objective function (4.3) can be simplified as:

$$\widehat{\mathbf{p}} = \arg\min_{\mathbf{p}} \sum (\widehat{d}_i - |\mathbf{p} - \mathbf{p}_i|),$$

which is equivalent to

$$\widehat{\mathbf{p}} = \arg\min_{\mathbf{p}} - \sum (|\mathbf{p} - \mathbf{p}_i|).$$

Since $|\mathbf{p} - \mathbf{p}_i| \geq 0$, we can rewrite the objective function as:

$$\widehat{\mathbf{p}} = \arg\min_{\mathbf{p}} - \sum (|\mathbf{p} - \mathbf{p}_i|)^2$$

The final objective function is described as below.

$$\widehat{\mathbf{p}} = \arg\min_{\mathbf{p}} - \sum (|\mathbf{p} - \mathbf{p}_i|)^2$$

$$\text{subject to } |\mathbf{p} - \mathbf{p}_i| \leq \widehat{d}_i,$$

which is equivalent to the objective function below.

$$\widehat{\mathbf{p}} = \arg\max_{\mathbf{p}} \sum (|\mathbf{p} - \mathbf{p}_i|)^2 \tag{4.4}$$

$$\text{subject to } |\mathbf{p} - \mathbf{p}_i| \leq \widehat{d}_i$$

The intuition of the algorithm above is that the estimated position is pushed far away from beacons, thus towards the outside constrained boundaries. The final optimization result is that the estimated position is limited in the intersection area of circular constrained regions (upper bound constraints) and pushed towards the constrained boundaries (distance fitting), which is the intent of the hybrid approach.

i-Multihop algorithm: final version

In this section, we will relax the assumption that the in-network distance measurements between immediate neighbors are accurate. Without this assumption, distances estimated from the lengths of the shortest paths will have two error sources: one is the measurement error between immediate neighbors, the other is incurred when the shortest path is not close to a straight line. Under this circumstance, it is possible that some of the distance measurements are less than their true values. This

happens in *correct* distance measurements where the shortest path is very close to a straight line and the measured distance of each segment is less than its true value, such that the length of the shortest path is less than the Euclidean distance due to the error accumulation of each segment. Based on this analysis, we have the following observation.

Observation 3: All the *correct* distance measurements $\widehat{d_i}$ is close to (either larger or less than) its true value d_i ($\widehat{d_i} \approx d_i$). All the *incorrect* distance measurements $\widehat{d_i}$ is larger than its true value d_i ($\widehat{d'}_i \gg d'_i$).

Based on this observation, the sensor localization is remodeled as below.

Model 3: Given a network graph $G = (V_m \bigcup V_n, E_s \bigcup E_t)$, the vertex set V_m defines the beacons set; the vertex set V_n defines the position unknown sensor set; the edge set E_s defines all the *correct* distance measurements $\widehat{d_i} \approx d_i$; and E_t defines all the *incorrect* distance measurements $\widehat{d'}_i \gg d'_i$. The sensor localization is to recover coordinates of vertex set by filtering out the *incorrect* distance measurements E_t and fitting the *correct* distance measurements.

The hybrid approach with the upper bound constraints will not work in the model 3, because *incorrect* distance measurements may be less than their true values such that the constrained circular regions cannot intersect with each other. In such a case, no feasible position exists to satisfy all the upper bound constraints and the objective function will not find out a suitable solution.

To solve the problem above, we add slack variables ε_i to the hybrid approach to get our final version of the i-Multihop algorithm.

$$\widehat{\mathbf{p}} = \arg\min_{\mathbf{p}} \sum (\widehat{d_i} + \varepsilon_i - |\mathbf{p} - \mathbf{p}_i|)^2 + k \sum \varepsilon_i \tag{4.5}$$

$$\text{subject to } |\mathbf{p} - \mathbf{p}_i| \leq \widehat{d_i} + \varepsilon_i,$$

where k is the weight coefficient which is set to a large value (10^6 in our computation). Due to the large value of the weight coefficient k, the second apart $k \sum \varepsilon_i$ has much higher priority to be minimized than the first part, which means the slack variable ε_i has higher priority to be minimized than the difference between calculated distance $|\mathbf{p} - \mathbf{p}_i|$ and the measured distance $\widehat{d_i}$.

The intuition behind the objective function (4.5) can be described as follows. For those distance measurements which are less than their true values, the slack variables ε_i can increase the upper bound to the minimum extent such that the summary of the measured distance $\widehat{d_i}$ and the slack variable ε_i is greater than the true value d_i. The consequence is that all the circular region constraints intersect into an non-empty set which contains a feasible solution for the objective function. Through this way, the problem is transformed to Model 2 where all the upper bound constraints are larger than the true distances. Therefore, we can use the similar approach as Model 2 to locate sensors by filtering out *incorrect* distance measurements and fitting *correct* distance measurements.

Again, we simplify the objective function (4.5) to facilitate our optimization computation as below.

$$\widehat{\mathbf{p}} = \arg\min_{\mathbf{p}} - \sum (|\mathbf{p} - \mathbf{p}_i|)^2 + k \sum \varepsilon_i \tag{4.6}$$

$$\text{subject to } |\mathbf{p} - \mathbf{p}_i| \leq \widehat{d}_i + \varepsilon_i$$

Average length per hop

In the case where distance measurements between immediate neighbors are unavailable, multihop algorithm estimate distances by multiplying the number of hops of the shortest path with the average length per hop. Here, the average length per hop to the *ith* beacon can be sampled as below.

$$h_i = \frac{\sum_{j \in V_m} D_{ij}}{\sum_{j \in V_m} H_{ij}},$$

where D_{ij} is the Euclidean distance from the jth beacon to the ith beacon, and H_{ij} is the number of hops from the jth beacon to the ith beacon.

However, the average length per hop calculated as above can be severely affected by the concave shapes. Because the shortest path between beacons may also be distorted by the concave shape and deviate far away from a straight line, the actual length L_{ij} of the shortest path will be much longer than the Euclidean distance D_{ij} calculated from the coordinates of the beacons. Therefore, the true value of the average length per hop L_{ij}/H_{ij} is much larger than the value estimated from D_{ij}/H_{ij}. This will make the average value estimated by $\sum_{j \in V_m} D_{ij} / \sum_{j \in V_m} H_{ij}$ less than the actual one $\sum_{j \in V_m} L_{ij} / \sum_{j \in V_m} H_{ij}$ because some of the distances D_{ij} are much less than the lengths of L_{ij}.

To solve the problem above, we need to filter out the pairwise beacon distances which are distorted by concave shapes, which is achievable by reusing the *i*-Multihop algorithm above. Since the calculation of the average length per hop is to infer dis-

tances from coordinates, it is a reversed process of localization algorithm which infers coordinates from distances. Therefore, we can use similar idea as i-Multihop algorithm to filter out distorted pairwise beacon distances and calculate correct average length per hop as below.

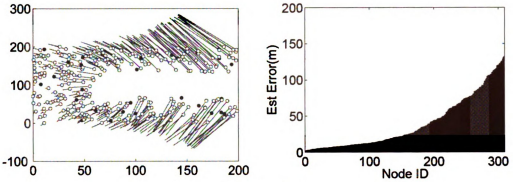
$$\widehat{l} = \arg\min_{l} \sum (lh_i + \varepsilon_i - |\mathbf{p}_i - \mathbf{p}|) + k \sum \varepsilon_i$$

$$\text{subject to } |\mathbf{p}_i - \mathbf{p}| \leq lh_i + \varepsilon_i,$$

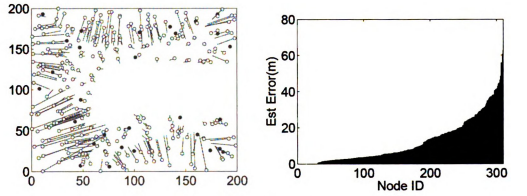
where l is the average length per hop, h_i is the number of hops of the shortest path between $\mathbf{p}$ and $\mathbf{p}_i$, ε_i is the slack variable, and k is the weight coefficient. The intuitive explanation of the objective function is to find the optimal value of the per hop's average length l which can minimize the difference between the calculated distance $|\mathbf{p}_i - \mathbf{p}|$ and the summary of the measured distance lh_i and the slack variable ε_i under the upper bound constraints. Similar to the i-Multihop algorithm, with the help of the upper bound constraints, only measured distances which are close to their true Euclidean distances will be involved in the optimization, and the *incorrect* distance measurements which are much larger than the true values are filtered out. To facilitate the optimization computation, we simplify the objective function above to the following linear optimization.

$$\widehat{l} = \arg\min_{l} (\sum h_i) l + k \sum \varepsilon_i \tag{4.7}$$

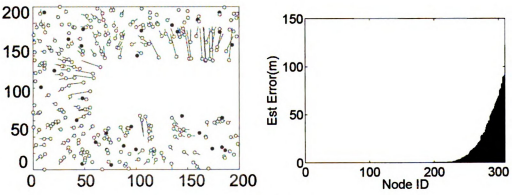
$$\text{subject to } |\mathbf{p}_i - \mathbf{p}| \leq lh_i + \varepsilon_i$$



(a) Distance Fitting



(b) Upper Bound



(c) Hybrid

Figure 4.20 Comparison of the distance fitting, upper bound and hybrid approaches in C shape configuration

4.3.2 Performance evaluation

We compare the i-Multihop algorithm with the original Multihop algorithms and n-Multihop algorithm in this section. Since all these three algorithms use the same beacon message flooding or Distance Vector algorithm to compute the number of hops along the shortest paths, the communication cost of the three is the same. Therefore, we can ignore the details of message communication and focus on the character of their geometry calculation. Such an abstraction can help us to evaluate their performance in Matlab simulation, where a sensor network is described as a network graph with vertices representing sensor nodes and edges representing the measurable distances between immediate neighbors.

To investigate the impact of concave shapes on the performance of the multihop approaches, we use three basic configurations. In the first configuration, 400 nodes were randomly deployed in a $200 \times 200m^2$ square area which has the convex shape. In the second configuration, A portion of sensors in the square area were moved out and the square shape of the network topology is transformed to the C shape as shown in Fig. 4.2. In the third configuration, we transform the network topology to the S shape.

The following metrics are used in our evaluation.

- **Transmission range** R: the maximum radio transmission range of sensor nodes.

- **Estimation error** μ_i: the distance between the estimated position and true position of sensor node i, and $\mu_i = |\widehat{\mathbf{p}}_i - \mathbf{p}_i|$.

- **Average estimation error** $\widehat{\mu}$: the average value of the estimation error μ_i and $\widehat{\mu} = \sum \mu_i / N$, where N is the total number of sensors.

Comparison of distance fitting, upper bound and hybrid approaches

We compare the performance of distance fitting, upper bound and hybrid approaches in the C shape configuration. In this comparison, we assume distance measurements between immediate neighbors are accurate enough, thus the deviation of the shortest path from the straight line is the only error source of distance measurement. The comparison result is shown in Fig. 4.20, where circles represent the true positions of sensors (solid circles for beacons and empty circles for sensors), and the lines represent the estimation error μ_i. The distribution of estimation error μ_i is also described by the bar graph on the right side, where the sensors are ordered by their estimation error μ_i. The comparison shows that the distance fitting approach(Fig. 4.20(a)) has the worst performance, because it tries to fit the distances to all the beacons while some of them are severely distorted by the C shape. Fig. 4.20(b) shows that the performance is improved significantly by the upper bound approach, which uses the distance upper bound to filter out the impact of distorted distance measurements. The performance is further improved by the hybrid approach (Fig. 4.20(c)), which solves the problem that the upper bound approach may have large estimation errors when all the beacons are located in one side of a sensor.

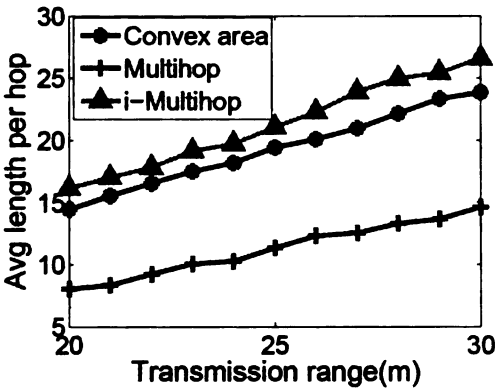


Figure 4.21 Average length per hop

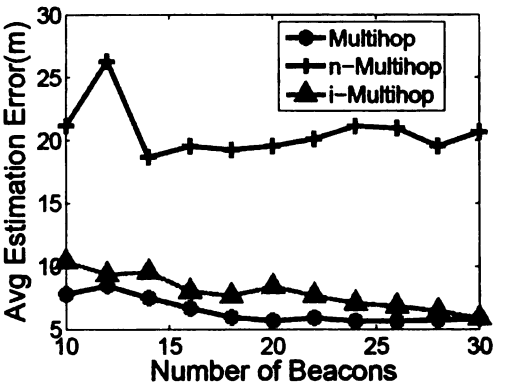


Figure 4.22 Square shape configuration

Average length per hop

When distances between immediate neighbors are not available, we can sample the average length per hop from beacons. However, as we discussed above, the average length per hop calculated in the original multihop algorithm will also be affected by concave shapes because the shortest paths between beacons may deviate far away from straight lines. To eliminate the effect of concave shapes, we use the upper bound constraints in i-Multihop algorithm to filter out the shortest paths which are distorted severely by concave shapes. We evaluate the estimation accuracy of the average length per hop as follows. First, the average length per hop is estimated in the square shape configuration using the original multihop algorithm. The experiment is repeated multiple times with different radio transmission ranges. For each radio transmission range, the square shape configuration is transformed to the C shape configuration by removing out some sensor nodes, and the average length per hop is estimated again by i-Multihop algorithm and multihop algorithm respectively. Since

the C shape is the subset of the square shape, they should share similar topology properties including the average length per hop. Fig. 4.21 shows that the average length per hop calculated by the i-Multihop algorithm is closer to the average length per hop of the square shape configuration than the one calculated by the multihop algorithm. This demonstrates that the i-Mulithop algorithm can recover the average length per hop correctly even in concave shapes.

A notable thing on the average length per hop of the shortest path is that it is different from the average distance between immediate neighbors. The reason is explained as follows. To minimize the total number of hops between two sensors, each hop of the shortest path is stretched to its maximum value. The consequence is that the average length per hop is increased with the radio's maximum transmission range, as shown in Fig. 4.21. In the experiment, we repeat the test by varying the radio transmission range while keeping all other network configuration the same as each other, thus the distances between immediate neighbors are the same in each test. However, the experiment shows that the average length per hop is increased when the maximum radio transmission range becomes longer.

Impact of concave shapes

In this section, we focus the performance comparison on the connectivity-based multihop algorithms, where the distances between immediate neighbors are not available and the average length per hop is sampled from beacons. To investigate the impact of concave shapes on the performance of multihop algorithms, we compare the multihop algorithm based on distance fitting, n-Multihop algorithm and i-Multihop algorithm

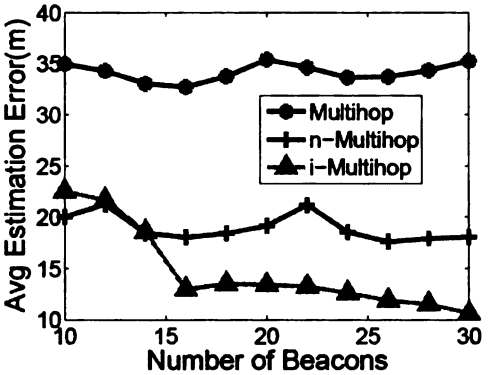


Figure 4.23 C shape configuration

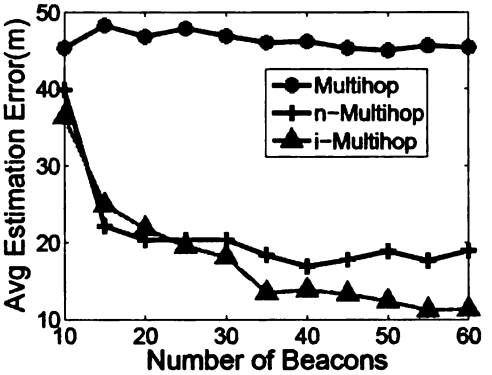


Figure 4.24 S shape configuration

in square shape, C shape and S shape configurations.

Fig. 4.22 shows the performance comparison of the three algorithms in the square shape configuration. The Multihop and the i-Multihop algorithms have similar performance, while the performance of the n-Multihop algorithm is much worse than the other two algorithms. The n-Multihop algorithm has the worst perform because in connectivity-based multihop algorithm, the distance estimated from the nearest beacon does not guarantee it is the best estimation which is closest to the true Euclidean distance. The performance of the n-Multihop algorithm becomes worse when the maximum transmission range becomes much longer than the average distance between immediate neighbors. As we discussed above, the average length per hop h is different from the average distance d between immediate neighbors, and the former is strongly related to the maximum radio transmission range R. When the density of deployed sensors is fixed, choosing large transmission range R will improve the network connectivity, which is helpful to sensor localization since the network becomes more rigid with higher connectivity. However, the large transmission range R will

lead to the consequence that the average length per hop h is much larger than the average distance d between immediate neighbors. If we estimate distances from the nearest beacon, it is possible that the beacon is within one or two hops range. If it is within one hop range, the true distance to the beacon is close to the average distance d between immediate neighbors. The consequence is that the true distance is less than a single average length per hop h, which is smallest measurable unit in connectivity-based Multihop algorithm. This will cause relative large errors from the sensor to nearby beacons. Such a distance estimation error imposes a limitation on the positioning accuracy of the n-Multihop algorithm. On the other hand, instead of fitting distances to the nearest beacons, the i-Multihop algorithm tries to fit the distance measurements which are closest to their true Euclidean distances, such that the final positioning result of a sensor is closer to its true location.

Fig. 4.23 shows the performance comparison of the three algorithms in the C shape configuration. The Mulihop algorithm performs worst, while the i-Multihop algorithm is the best of the three. The localization accuracy of the i-Multihop algorithm is improved significantly when the number of beacons are increased from 10 to 16, and it eventually converges to a fixed value when the number of beacons is continuously increased. There exists a critical point because in C shape configuration, when the number of beacons exceeds certain value, most of sensors can have three beacons which are connected by the shortest paths that are close to straight lines. The Mulithop algorithm has the worst performance because some of the distance estimation are distorted by the C shape. The n-Multihop algorithm does not perform as well as the i-Multihop algorithm because its localization accuracy is upper bounded

by the granularity of the transmission range, as we discussed above.

Fig. 4.24 shows the performance comparison of the three algorithms in the S shape configuration, which is more concave than the C shape and some of the distances estimated by the shortest paths deviate further from their true Euclidean distances. The comparison shows that the Multihop algorithm performs much worse than the n-Multihop algorithm and i-Multihop algorithm, and the i-Multihop algorithm has the best performance. In the S shape configuration, the performance of the i-Multihop algorithm is increased significantly when the number of beacons is increased to 30, and after that, it eventually converges to a fixed value. The critical point of the number of beacons is larger than the C shape because more beacons are necessary for all sensor to have at least three beacons connected by the close-to-straight-line shortest paths.

From the comparison above we can conclude that the i-Multihop algorithm performs best in the three algorithms, and it can locate sensors in concave environments with positioning accuracy comparable to that of convex environments if the number of beacons reaches the threshold.

4.3.3 Iterative approaches

The simulations above show that certain minimum number of beacons are required for i-Multihop algorithm to achieve sufficient localization accuracy in concave areas. More beacons are demanded when deployed areas become more complicated. This is because a sensor can accurately locate itself only when it is connected to at least three beacons by the close-to-straight-line shortest paths. If only a few beacons are

deployed in a concave area such as C shape or S shape, it is possible that some of sensors are connected to less than three beacons by the close-to-straight-line shortest paths, which results in large estimation errors. In this section, we show that high accuracy can be achieved with less beacons by iteratively applying the i-Multihop algorithm.

In the iterative approach, a few beacons are deployed as initial beacons. Due to the small number of initial beacons, they may be "visible" to only a small part of sensors through the close-to-straight-line shortest paths. Those small portion of sensors can accurately locate themselves by referring to the initial beacons. After that, those sensors with accurately determined positions "convert" themselves as new beacons by advertising beacon signals. It is possible that the newly added beacons are connected through the close-to-straight-line paths to some sensors which do not have sufficient initial beacons before. By utilizing the beacon signals sent from newly added beacons, those sensors previously with inaccurate estimation results can refine their positions and achieve accurate positioning results. The whole process are recursively repeated until all sensors are accurately located or no more beacons are added.

The challenging of applying the i-Multihop algorithm into the iterative process is how to identify "good" candidates which can accurately locate themselves. In other words, we need to estimate how accurate a sensor can locate itself before we can iteratively apply the i-Multihop algorithm. Due to the absence of global view of the entire network, a sensor cannot judge if it is connected to at least three beacons through the close-to-straight-line paths. Thus, we cannot identify "good" candidates by simply counting the distances estimated from close-to-straight-line paths. In the following

discussion, we propose the upper bound approach to estimate sensors' positioning accuracy.

Estimate positioning accuracy

As we discussed in Section 4.3.1, suppose a sensor $\mathbf{p}$ is within circular region constraints C_i with origin $\mathbf{p}_i$ and radius $\widehat{d}_i$. Here $\mathbf{p}_i$ are beacons' positions and $\widehat{d}_i$ are estimated distances from $\mathbf{p}$ to $\mathbf{p}_i$. Let $S_t = \bigcap C_i$, which represents the intersection area of all constrained regions C_i. We notice that the position of the sensor can be pinpointed more accurately when the area of S_t becomes smaller. On the other hand, if less than three distance measurements are *correct* distance estimations, the intersection area S_t tends to be large. This observation shows that we can identify good candidate for new beacons by checking the size of the intersection area S_t.

However, it is difficult to accurately calculate the area of the intersection S_t due to its irregular shape. Instead, we use the radius of the intersection S_t to estimate the positioning accuracy. As shown in Fig. 4.25, the intersection's radius d_u is defined as the maximum distance between the estimated position $\widehat{\mathbf{p}}$ to any other point $\mathbf{p}_x$ within the intersection area and can be calculated as below.

$$d_u = \max_{\mathbf{p}} |\widehat{\mathbf{p}} - \mathbf{p}_x| \tag{4.8}$$

$$\text{subject to } |\mathbf{p}_x - \mathbf{p}_i| \leq \widehat{d}_i$$

Here, $\widehat{\mathbf{p}}$ is the position estimated by our i-Multihop algorithm, $\mathbf{p}_i$ are beacons' positions and $\widehat{d}_i$ are measured distances. By the definition of d_u, we have $d_u \geq d_e$, where

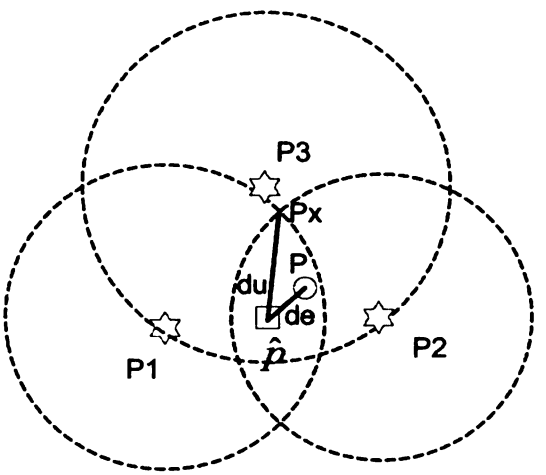


Figure 4.25 By its definition, radius d_u is larger than estimation error d_e

d_e is the estimation error between estimated position $\widehat{\mathbf{p}}$ and the true position $\mathbf{p}$. In other words, d_u is the upper bound of the estimation error d_e and can be used to estimate the positioning accuracy. Sensors with small estimation errors are identified if they have small estimation radius d_u.

Apply i-Multihop algorithm iteratively

Based on the radius d_u, we can find out sensors with high positioning accuracy, which makes it possible to apply i-Multihop algorithm iteratively in concave areas. In the iterative approach, positions of beacon nodes are broadcast through beacon signals. Beacon signals have counters which are increased by the lengths of hops when they are forwarded between neighboring sensors. The length of the shortest path from a sensor to a beacon can be found out from the minimum counter value among all the received beacon signals sent out by that beacon. Therefore, each sensor can learn beacons' positions and the lengths of the shortest paths to those beacons. Sensors

keep listening to beacon signals and refine their positions each time when new beacon signals are received. At the same time, sensors' position accuracy is estimated by their radiuses d_u. When the radius of a sensor is less than the threshold, it will advertise itself as a new beacon, whose beacon signals can be utilized by other sensor to refine their estimated positions. The positioning processes are repeated until all sensors are accurately located or no more beacons are added. Note the iterative process can be implemented by sensors' localized algorithm which consists of beacon signals listening and position refining. Therefore, it can be implemented in a fully distributed fashion.

Implement iterative i-Multihop approach in a distributed fashion

The pseudo-algorithm of the iterative i-Multihop approach is described in Algorithm 1. In the iterative algorithm, if a sensor is a beacon, it exits the localization algorithm after it sends out beacon packets. If the sensor is not a beacon, the sensor keeps listening to beacon packets. After it receives new beacon packets, the sensor recalculates its location together with the estimation error. If the estimation error falls below the threshold of becoming beacons, the sensor will announce itself as a new beacon and send out beacon packets. The entire algorithm terminates when no new beacons are added into the system. This algorithm is fully distributed because it can be finished by individual sensor without global coordination. The localized operations involve three simple steps: 1)keep listening to beacon packets; 2)update the estimated location and estimation accuracy; and 3)announce itself as a new beacon if the estimation accuracy falls below the threshold.

```
while true do
    if the sensor is a beacon then
        send out beacon packets which contain the sensor's coordinate;
        break;
    end
    keep listening to beacon packets for a time period;
    if beacon packets received then
        Update the sensor's position with the newly received beacon packets;
    else
        break;
    end
    Estimate the upper bound of localization error du;
    if du ≤ beacon_threshold then
        set the sensor as a beacon;
    end
end
```

Algorithm 1: Iterative i-Multihop algorithm

Performance of iterative i-Multihop algorithm

We evaluate the performance of the iterative i-Multihop algorithm as follows. In the evaluation, 314 nodes are deployed in a C shape area with only 4 initial beacons are deployed at the four corners. We assume that distances between neighboring sensors are measurable, thus the mismatch between the shortest path and the straight line is the main source of the distance measurement error. Fig. 4.26 shows both the average estimation error and the median estimation error are decreased along the iterative process when more and more beacons are involved in the localization process. We also note that the median estimation error is always smaller than the the average estimation error. This is because a small portion of sensors have much larger errors than the rest of sensors. To further illustrate the iterative i-Multihop algorithm, an example is shown in Fig. 4.27. At the beginning(Fig.4.27(a)) when only the four initial beacons are used, a number of sensors have large errors because they

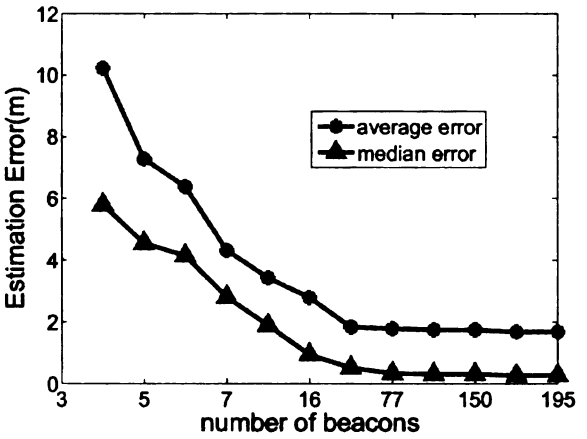
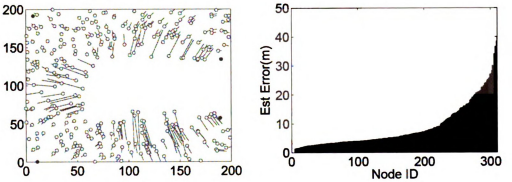


Figure 4.26 Positioning accuracy is improved along the iterative process

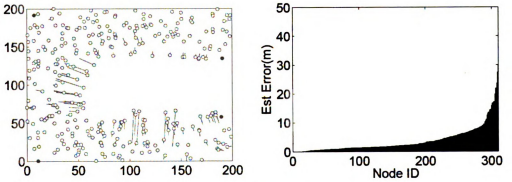
do not have three beacons visible through the close-to-straight-line shortest paths. Fig. 4.27(b) shows the intermediate status of the iterative process, where some sensors improve their positioning accuracy by referring to newly added beacons. Fig. 4.27(c) shows the final stage of the iterative process, where majority of sensors can locate themselves accurately. We notice that there are a few sensors which can not locate themselves accurately in the final stage. This is because those sensors do not have good beacons' layout even with the help of iterative approach. Those sensors with large estimation errors can be identified by their radius d_u, thus we can notify upper layer location-aided applications when sensors have large estimation errors and the positioning results are unreliable.

The iterative i-Multihop algorithm differs from previously proposed iterative approaches in the following aspects:

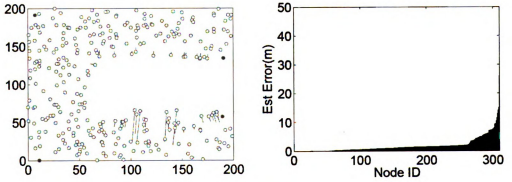
- It does not require that initial beacons are adjacent to each other, which is



(a) Initial stage



(b) Intermediate stage



(c) Final stage

Figure 4.27 Demo of iterative i-Multihop algorithm

an implicit assumption for other iterative approaches to initiate the iterative process.

- For beacons to be propagated to entire areas, previous iterative approaches usually require dense and uniform sensor distribution. Otherwise, the newly added beacons cannot approach some sensors and the whole iterative process is interrupted. On the contrary, in the iterative i-Multihop algorithm, all sensors can be located as long as they form a connected network. This is because sensors' positions are first estimated from initial beacons, and then refined by newly joined beacons. Here, the iterative strategy are mainly used to improve the positioning accuracy.

- If the positioning accuracy cannot be improved by the iterative process due to awkward beacon layout, the positioning error can be estimated by the radius d_u, which can be sent to upper layer applications together with the positioning data. With the notification of positioning accuracy, the upper layer location-aided applications can utilize the location information more intelligently by prudently dealing with the sensors with large estimation errors.

- A potential problem of previous iterative approach is that the positioning errors may accumulate along the iterative process. This is because the newly joined beacon are not as accurate as initial beacons and may have large estimation errors, especially the flip-over errors discussed in [17]. The estimation errors of newly joined beacons may accumulate in the following localization process and the final results are severely corrupted. The accumulative errors are minimized

in iterative i-Multihop algorithm because of two reasons: (1) we use the radius d_u to estimate positioning accuracy and beacons are only converted from sensors which can accurately locate themselves; (2) in the iterative i-Multihop algorithm, the positioning accuracy is consistently increased because sensors' estimated positions are updated by new beacon signals only when their estimation accuracy are improved, i.e. smaller radius d_u can be achieved.

4.4 Summary

In this chapter, we propose the upper bound approach to locate sensors in complicated environments including obstructed and concave areas, where both the RSS-based approaches and the multihop approaches may have large errors in distance measurements. The upper bound approach is based on the observation that incorrect distance measurements inferred from the RSS-based approaches and multihop approaches are always larger than their true values. Therefore, we can regard distance measurements as the upper bound of their true values and confine estimated positions to the small areas intersected by correct distance measurements. We evaluate the upper bound approach in both obstructed environments and concave areas with intensive simulation tests, which show that the upper bound approach is an effective solution to accurately locate sensors in complicated environments.

CHAPTER 5

Packet Routing in Wireless Sensor Networks

5.1 Motivation

Reporting sensed data to a base station is the primary function of a sensor network. A huge volume of data can be generated by numerous sensors, which incurs high communication cost if all the raw data is sent to the base station. To solve this problem, Directed Diffusion [90] and TinyDB/TAG [91] suggest a query/response mechanism. In this mechanism, the base station query named data through interest flooding, and only the data matching the interest is reported. Such a query/response mechanism is inspired by the novel concept of *data-centric communication* [92] in which data is named by attributes and the communication primitive is a form of query: which data has the named attributes? Here, the name of sensed data is the main concern rather than the identity of the sensor. To further reduce the communication costs incurred by the interest flooding, the *data-centric storage* [93][94] proposes to store data by name at nodes, which follows the rule that all data with the same general

name are stored at the same node. Queries for data with a specific name can therefore be sent directly to the node without reliance on interest flooding. The data-centric storage can be readily realized if point-to-point routing is available in sensor networks, which helps sensing nodes to store data to hosting nodes, and the base station to retrieve data from hosting nodes. Wireless sensor networks, when evolving their capability from simple sensing and reporting to complicated in-network storage [95] and in-network process [96], require intensive coordination among intelligent sensors. Such a coordination necessitates point-to-point routing between any pair of sensors. Due to its fundamental role, the research on point-to-point routing for a wireless network, initiated more than a decade ago [97], is still an active and ongoing area [98][99][100] in which many challenging problems need to be addressed.

First, point-to-point routing cannot be directly ported from wired networks, which achieve the scalability through hierarchical architecture and address aggregation. The hierarchical architecture aggregates routers into autonomous systems. Each router in an autonomous system is attached with several physical networks. Hosts in the same physical network share high-order bits of addresses as their network prefixes. The aggregation of both routers and addresses helps to minimize routing states maintained in each individual router because: (1) internal routers of an autonomous system only need to maintain routing information of physical networks within the same system; (2) autonomous systems, which are interconnected through border gateway routers, are viewed as single entities in the backbone inter-domain routing. The abstraction of autonomous systems helps to minimize routing states in gateway routers since the internal details of autonomous systems are transparent to the backbone routing.

Despite their success in wired networks, it is difficult to apply the hierarchical architecture and address aggregation to wireless sensor networks. First, wireless sensor networks consist of randomly deployed nodes which communicate with each other through radio channels. As a result, a purely flat network topology is naturally formed because randomly deployed nodes can only communicate with their immediate neighbors within the radio transmission range. It is difficult to organize a wireless sensor network topology into a hierarchical structure unless long haul wireless links can be easily added between remote nodes. Second, the address aggregation is meaningless in a flat network topology, since no central routers in the upper tier can be attached by a group of nodes which share high-order bits of addresses as group prefixes. Without the aid of hierarchical architecture and address aggregation, it is prohibitive to implement the table-driven shortest path routing in a wireless sensor network, which requires per-destination states maintained by individual nodes. When the network scales to thousands of nodes, the large size routing tables containing thousands of entries cannot be affordable to resource-constrained sensors.

The conflict between the large size network with a random structure and the small routing states affordable to sensors raises fundamental challenges to point-to-point routing in a wireless sensor network. To address this problem, location aware routing (LAR) has been proposed to forward packets in a wireless network according to nodes' geographic positions [6][7]. In LAR, a packet will be greedily forwarded to the next neighbor which is geographically closer to the destination, and finally delivered to the destination after consecutive hop by hop forwarding. LAR is promising in that packet routing is realized through a *localized* algorithm that solely relies on the positions of

the destination, the current node, and its immediate neighbors. The positions of a small set of neighbors compose a sensor's routing states, which can be easily fit to the sensor's limited memory.

The location aware routing assumes that a packet can be moved closer to the destination in the network topology when it is moved geographically closer to the destination in the Euclidean space. This assumption is based on the observation that the topological structure of a wireless network can be approximated by its geographic structure. Because a wireless node can only communicate with its neighbors within the maximum radio transmission range, pairwise nodes may have a short communication path in the network topology if they are geographically closer to each other in the Euclidean space, i.e. the hop count distance between pairwise nodes is proportional to their Euclidean distance. This observation is correct in an ideal wireless network model where sensors are uniformly distributed in an open flat area and communicate with neighbors through wireless channels of perfect reception. However, this ideal model oversimplified the spatial complexity of a realistic wireless sensor network that has *complicated topological structure* and *irregular wireless radio communication patterns* when deployed in complicated environments.

Due to the discrepancy between a wireless network's complex spatial characteristics and its oversimplified geographic description, the location aware routing may fail to deliver a packet or forward a packet along a suboptimal routing path. For example, a packet may be trapped in a local minimum where none of the neighbors is closer to the destination. A packet may also be forwarded along a route consisting of long distance hops with low quality wireless channels. Numerous approaches have

been proposed to recover the location aware routing from local minimum[7] or find the proper forwarding advance without sacrificing channel quality[57][43][58]. Constrained by the inaccurate geographic model on a network topology, these workaround solutions cannot guarantee a packet to be efficiently forwarded along the optimal routing path.

In this chapter, we aim to improve the routing performance of the greedy forwarding with two steps. First, we propose the topology aware routing to solve the problem of local minimum. The topology aware routing encodes hop count distances between pairwise nodes into nodes' virtual coordinates. Based on the precise hop count distance comparison, the greedy forwarding can always find the next hop that is one hop closer to the destination, and achieve guaranteed packet delivery with consecutive hop-by-hop forwarding. Second, we propose the ETX distance based greedy forwarding to find the optimal routing path comprising high quality links. The ETX distance based greedy forwarding embeds a wireless sensor network into a Euclidean space where nodes' virtual distance is equal to the number of expected transmissions for a packet to be successfully delivered between the pairwise nodes. Because the virtual distance directly reflects the end-to-end communication channel quality, the greedy forwarding can guide a packet along the optimal routing path which has the shortest virtual distance.

In the following discussion, we first detail the spatial complexity of wireless sensor networks and how it affect the routing performance in Section 5.2. We further show how topology aware routing solves the local minimum problem in Section 5.3. After that, we show that the end to end routing performance can be improved by ETX

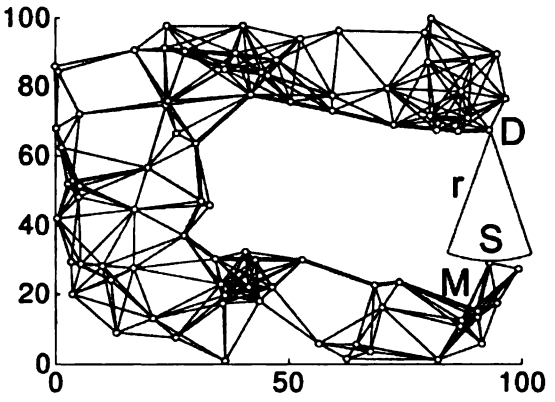


Figure 5.1 Inconsistence between the topological structure and geographic structure

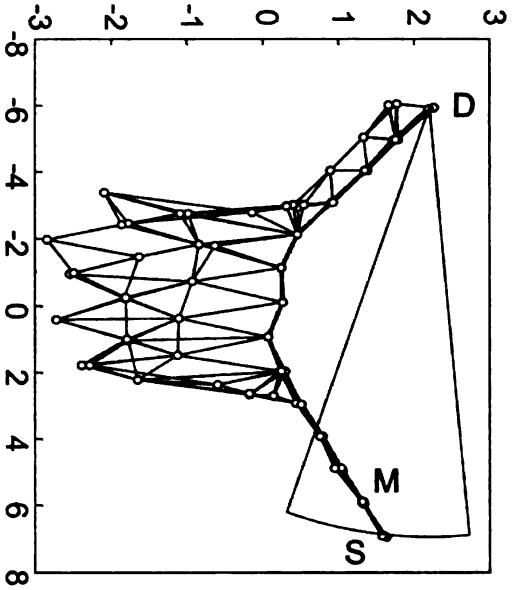


Figure 5.2 Consistence between the topological structure and virtual geometric structure

distance based greedy forwarding in Section 5.4. We evaluate the topology aware routing in Section 5.5 and ETX distance based greedy forwarding in Section 5.6 . After that, we summarize this chapter in in Section 5.7.

5.2 Spatial complexity of a wireless sensor network

Because radio signals are susceptible to environmental interference, a wireless sensor network often demonstrates complex spatial characteristics in a complicated environment, which include complicated network topologies and irregular radio communication channels.

5.2.1 Spatial complexity of a wireless network topology

Location aware routing (LAR) assumes that a packet will be one hop closer to its destination in each forwarding and finally delivered to the destination after consecutive hop by hop forwarding. This assumption, however, can only hold when nodes are uniformly distributed in an open flat area. When sensors are deployed in complicated environments, radio communication links may be blocked by obstructions and the network topology will have a concave shape. In such a case, the greedy forwarding may be trapped in the local minimum, where no neighbor is geographically closer to the destination. An example of local minimum is shown in Fig. 5.1, where node S cannot find any neighbor that is geographically closer to destination D.

Numerous recovery schemes have been proposed to solve the local minimum problem by exploring the geographic characteristics of a wireless network[7][49] or resorting to small range message flooding[8]. The recovery schemes usually have higher computation complexity than the simple greedy forwarding or may not work as a universal solution to handle various cases due to the spatial complexity of a wireless network. For example, the right-hand rule based perimeter routing[7] has suggested to route a packet along the counter clockwise direction when a packet is trapped into a local minimum. For the specific case in Fig. 5.1, the perimeter routing cannot route a packet through the local minimum M since the counter clockwise routing path is blocked by the obstruction.

In this chapter, we solve the problem of local minimum by encoding hop count distances to nodes virtual coordinates, such that hop count distances between pairwise

nodes can be precisely recovered from their small dimensional coordinates. Based on the precise hop count distances comparison between neighboring nodes, greedy forwarding can find the exact neighbor that is one hop closer to the destination. We show such an coordinate assignment mechanism can be approached through network embedding. An example of network embedding is shown in Fig. 5.2, where the same network topology of Fig. 5.1 is embedded into a 2-dimensional Euclidean space. Here, the same network topology, i.e. nodes' adjacent relationships are the same, is expressed in two different manners: Fig. 5.1 lays out nodes according to their geographic coordinates from which the inferred Euclidean distances represent the geographic distances between pairwise nodes; Fig. 5.2 lays out nodes according to their virtual coordinates embedded from hop count distance metric space, and Euclidean distances inferred from virtual coordinates represent hop count distances between pairwise nodes.

Because hop count distances of a network topology are directly reflected by Euclidean distances in Fig. 5.2, precise hop count distance comparison between neighboring nodes to a destination can be achieved from the comparison of their Euclidean distances to the destination. Based on the precise comparison, greedy forwarding can find the right neighbor that is one hop closer to the destination. As shown in Fig. 5.2, node S can find the right neighbor M, which is one hop closer to the destination D, because the hop count distances from node S and M to destination D are consistent with their Euclidean distances to D. This is in contrast to Fig. 5.1 , where node S has longer hop count distance to D than node M while its Euclidean distance to destination D is shorter.

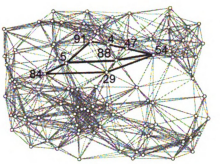


Figure 5.3 Long distance radio links of location aware routing

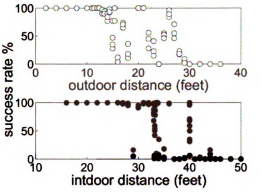


Figure 5.4 Packet reception between pairwise wireless nodes

5.2.2 Spatial complexity of wireless channels

The location aware routing uses a simple connectivity model to describe the wireless channels between pairwise nodes, i.e. pairwise nodes have the perfect reception channel if they are within the maximum transmission range of radio signals. In a realistic wireless network, neighboring nodes are often connected through unreliable wireless channels where packets may be lost due to the transmission error of radio signals. It is normal that packet loss rate is increased with the transmission range because the radio signals attenuate during their transmission, which leads to low signal to noise ratio (SNR). Since the location aware routing greedily select the next hop which is closest to the destination and therefore furthest to the sender, the location aware routing tends to include long distance hops in the routing path which are often unreliable and have high packet loss rate. An example is shown in Fig 5.3. Based on the greedy forwarding policy of the location aware routing, a packet is forwarded along the routing path $84 \rightarrow 29 \rightarrow 54$, which may have higher packet loss rate and

lower throughput than the routing path of $84 \rightarrow 5 \rightarrow 91 \rightarrow 4 \rightarrow 47 \rightarrow 54$ consisting of more while shorter intermediate links.

Several approaches[43][57][40] have been proposed to balance the forwarding distance and radio link quality, which can be divided into two categories:

1. define a threshold to exclude low quality radio links.

2. define a new metric which can be maximized under the constraints of both forwarding distance and radio link quality.

The irregular radio signal transmission pattern, however, makes it difficult to improve the performance of the location aware routing through these two strategies.

For the first strategy, it is difficult to determine a proper threshold value which can maximize the end-to-end routing performance. Fig. 5.4 shows the packet reception between pairwise nodes that we measured on the MICA2 sensor platform, which demonstrates that the perfect radio channel with 100% reception only exists between transceivers within a short distance. If the threshold is aggressively set to only include links with 100% packet reception, we may have a disconnected network or a routing path comprising excessive intermediate nodes, which increases both the processing cost and delay. We use an example to further explain how the threshold values affect the end-to-end routing performance. As shown in Fig. 5.3, the routing path $84 \rightarrow 5 \rightarrow 88 \rightarrow 54$ selected by the threshold of 85% packet reception rate outperforms the routing path $84 \rightarrow 5 \rightarrow 91 \rightarrow 4 \rightarrow 47 \rightarrow 54$ selected by the threshold of 100% packet reception rate, because the former uses less intermediate nodes in the packet forwarding with slightly inferior links. Because a proper threshold to determine the

optimal routing path may vary from different pairwise nodes, the location aware routing with a constant threshold cannot provide a universal solution to find the optimal routing path between any pairwise nodes.

Instead of simply excluding low quality radio links below a certain constant threshold, the second strategy selects radio links by optimizing the forwarding advance and quality of radio links simultaneously. For example, the energy-efficient forwarding[57] chooses the next hop which can maximize the product of the packet reception rate (PRR) and the distance traversed towards the destination. This strategy can achieve good routing performance when i) nodes are uniformly distributed; ii) the packet reception rate of the wireless channels can be explicitly modeled by the transmission distance. However, in an obstructed environment, radio signals have complex transmission patterns because the signal strength may be either strengthened or weakened due to multipathing or shading effect such that the packet reception rate is less correlated to the transmission distance and therefore difficult to model. Fig. 5.4 shows pairwise transceivers has different packet reception rates in outdoor and indoor environments.

Even if the location aware routing can find the optimal tradeoff between the forwarding advance and the link quality for individual hops, it may fail to find the path with the optimal end-to-end routing performance. Because both the advance distance and PRR are local metrics, the greedy forwarding may lead to a local minimum and fail to find the globally optimal path, which often happens in a network topology with complex spatial characteristics.

In this chapter, we try to find the optimal routing path by embedding a wireless

sensor network into a Euclidean space where nodes' virtual distance is equal to the number of expected transmissions (ETX) for a packet to be successfully delivered between pairwise nodes. Because the virtual distance directly reflects the end-to-end communication channel quality, the greedy forwarding can guide a packet along the optimal routing path which has the shortest virtual distance.

In the discussion below, we describe how the problem of local minimum is solved by topology aware routing. We further show that the end to end routing performance can be improved by ETX distance based greedy forwarding.

5.3 Topology aware routing

In this section, we show that greedy forwarding can achieve the same routing performance as the shortest path routing as long as hop count distances between neighboring nodes can be precisely compared. Next, we illustrate how to use the multidimensional scaling(MDS) to embed a network topology to a low dimensional Euclidean space in which the hop count distances between pairwise nodes can be accurately recovered. We also show how to extend the topology aware routing based on multidimensional scaling (TAR-MDS) to a distributed fashion through beacon sampling. The comparison between topology aware routing, beacon vector routing, and logical coordinate routing is discussed in the end of the section.

Before we proceed to the detailed description of TAR, we clarify the objectives of our proposed TAR as below:

1. We target to improve point-to-point routing performance of a wireless sensor

network comprising a large number of randomly deployed stationary nodes. This covers the main category of sensor networks which have limited dynamic characters due to nodes' failures.

2. We focus on improving routing success rate of greedy forwarding without reliance on any recovery schemes. As we discussed before, greedy forwarding is a viable routing approach to deliver packets in a wireless sensor network based on small routing states. Our objective is to reduce the chances of resorting to recovery schemes, which are regarded as auxiliary solutions to greedy forwarding and often more costly.

5.3.1 Greedy forwarding v.s. the shortest path routing

Proposition 1 *The greedy forwarding will route a packet from a source s to a destination d along the shortest path if hop count distances to the destination between neighboring nodes can be precisely compared in a connected network.*

Proof: Let $\delta(i,d)$ be the hop count distance between node i and destination d. Let $N(i)$ be the set of all the immediate neighbors of node i. Since the hop count distances between neighboring nodes can be precisely compared, the greedy forwarding can find node $j \in N(i)$ such that $\delta(i,d) - \delta(j,d) = 1$, i.e. node j is the neighbor which is one hop closer to the destination d than node i.

1. When $\delta(s,d) = 1$, source s will directly route a packet to destination d because destination d is the only neighbor which is one hop closer to itself:

Since $\delta(s,d) - \delta(j,d) = 1$, we have $1 - \delta(j,d) = 1$, i.e. $\delta(j,d) = 0$. As a result, $j = d$.

2. Assume that when $\delta(s,d) \leq k$, the greedy forwarding will route a packet along the shortest path. We prove that when $\delta(s,d) = k+1$, the greedy forwarding will route a packet along the shortest path:

   The greedy forwarding will forward a packet from source s to its neighbor m such that $\delta(s,d) - \delta(m,d) = 1$, i.e. $k + 1 - \delta(m,d) = 1$. As a result, we have $\delta(m,d) = k$. Based on the assumption, the greedy forwarding will route a packet along the shortest path to destination d, which means the number of hops for the greedy forwarding to deliver the packet from node m to d is k. Consequently, the number of hops for the greedy forwarding to deliver the packet from node s to d is $k+1$, which equals $\delta(s,d)$, the hop count distance of the shortest path between node s and d. Therefore, we can conclude the greedy forwarding will route a packet along the shortest path when $\delta(s,d) = k+1$. □

5.3.2 Embed network topologies to low dimensional Euclidean spaces

Proposition 1 shows that the greedy forwarding can achieve the same routing performance as the shortest path routing if a network topology can be accurately expressed by routing states, i.e. hop count distances between pairwise nodes can be precisely recovered from their local routing states. An naive approach to infer hop count distances between pairwise nodes from their local routing states is to maintain per-destination state in each node. The per-destination state maintained by node i in a network of

size N can be viewed as a N-dimensional virtual coordinate:

$$\mathbf{x}_i = [x_{i1}, x_{i2}, \ldots, x_{iN}]^t.$$

Here, x_{ij} is the hop count distance from node i to node j. Based on per-destination states, the hop count distance between any pair of node m and d can be easily obtained as x_{md}. Consequently, the hop count distances from neighboring node m and n to a destination d can be precisely compared by evaluating $x_{md} - x_{nd}$, which provides sufficient support for greedy forwarding to achieve the same routing performance as the shortest path routing.

High routing performance can be easily achieved based on the N-dimensional per-destination routing states. The challenge of an optimal routing design for wireless sensor networks is how to achieve high routing performance based on small constant size routing states, which can be reduced as how to accurately express a network topology by small routing states in an efficient fashion. A viable approach to the efficient expression of a network topology is to "losslessly compress" N-dimensional per-destination states to low dimensional routing states from which hop count distances between pairwise nodes can still be precisely recovered. This compression problem can be generalized as an embedding problem, which embeds a N-dimensional hop count distance metric space into a M-dimensional Euclidean space, where $M \ll N$. An embedding problem can be formalized as follows.

Let (X, δ) defines a metric space. Here X is a set and δ is a metric which defines a distance function between elements in X. For elements $\mathbf{x}_i, \mathbf{x}_j, \mathbf{x}_k \in X$, the distance function δ satisfies (1) symmetry: $\delta(\mathbf{x}_i, \mathbf{x}_j) = \delta(\mathbf{x}_j, \mathbf{x}_i)$; (2) positive

definiteness: $\delta(\mathbf{x}_i, \mathbf{x}_j) \geq 0$ and $\delta(\mathbf{x}_i, \mathbf{x}_j) = 0$ iff $i = j$; (3) triangular inequality: $\delta(\mathbf{x}_i, \mathbf{x}_k) + \delta(\mathbf{x}_k, \mathbf{x}_j) \geq \delta(\mathbf{x}_i, \mathbf{x}_j)$.

Let (P, d) defines the Euclidean space. Here P is a set of points mapped from elements in the set X and d is a metric which defines the function of 2-norm Euclidean distances between pairwise points in P. For $\mathbf{p}_i, \mathbf{p}_j \in P$, we have $d(\mathbf{p}_i, \mathbf{p}_j) = |\mathbf{p}_i - \mathbf{p}_j|$.

Definition 1 *An embedding of metric space (X, δ) into an Euclidean space (P, d) is a mapping $\phi : X \rightarrow P$ such that*

1. $\boldsymbol{p}_i = \phi(\boldsymbol{x}_i)$;

2. $\delta(\boldsymbol{x}_i, \boldsymbol{x}_j) = d(\phi(\boldsymbol{x}_i), \phi(\boldsymbol{x}_j)) = d(\boldsymbol{p}_i, \boldsymbol{p}_j) = |\boldsymbol{p}_i - \boldsymbol{p}_j|$

Remark 2 *For a network topology, the node set with the hop count distance function defines a metric space which can be embedded into an Euclidean space, because hop count distance function satisfied symmetry, positive definiteness and triangular inequality.*

Embedding a network topology into an Euclidean space can be intuitively explained as given hop count distances between pairwise nodes in a network topology, finding nodes' coordinates in a M-dimensional Euclidean space such that the hop count distances can be inferred from the 2-norm Euclidean distance of the mapped space. The objective of the embedding is to find the minimal M such that $\delta(\mathbf{x}_i, \mathbf{x}_j) = d(\phi(\mathbf{x}_i), \phi(\mathbf{x}_j))$, i.e. to embed a hop count distance metric space into the lowest dimensional Euclidean space in which hop count distances between pairwise nodes can still be precisely preserved.

Instead of an exact embedding, a network topology can be approximately embedded into an Euclidean space by relaxing condition 2 as $\delta(\mathbf{x}_i, \mathbf{x}_j) \approx d(\phi(\mathbf{x}_i), \phi(\mathbf{x}_j))$. Here, we slightly sacrifice the embedding accuracy to achieve lower dimensionality of the embedded Euclidean space and therefore smaller routing states. In summary, we define an efficient expression of a network topology as below:

Definition 2 *An efficient expression of a network topology is to embed the network topology into a M-dimensional Euclidean space such that:*

1. *M is minimized;*

2. *difference between hop count distances of the network topology and Euclidean distances of the embedded space are minimized:*

$$\min \sum_{\boldsymbol{x}_i, \boldsymbol{x}_j \in X} (\delta(\boldsymbol{x}_i, \boldsymbol{x}_j) - d(\phi(\boldsymbol{x}_i), \phi(\boldsymbol{x}_j))^2.$$

The double minimums in the definition above cannot be achieved simultaneously. The contradiction between the accuracy of the embedding and the small dimensionality of the embedded space reflects the inherent tradeoff between the accuracy of a network expression and the size of the expression. In this chapter, we seek to achieve an embedding accurate enough to support high routing performance at the low dimensionality suitable to resource-constrained nodes. In the following discussion, we show how to use multidimensional scaling (MDS) to realize this efficient expression.

For simplicity, we use short notations defined in Table 5.1 in following discussions.

Table 5.1 Notation list

Notation	Definition
i	node in a wireless sensor network
P	node set of a wireless sensor network, i.e. $i \in P$
k	beacon node which broadcast beacon messages to a network such that hop count distances from all other nodes to a beacon node are available
K	beacon node set
δ_{ij}	hop count distance between pairwise node i and j of a network topology
d_{ij}	Euclidean distance between node i and j in the embedded space

5.3.3 Embed network topologies through the multidimensional scaling

We use the multidimensional scaling (MDS) to find the optimal embedding which can efficiently compress routing states while accurately preserve hop count distances. MDS is a set of dimensionality reduction techniques which discover meaningful low dimensional structures hidden in their high dimensional observations. The MDS can be generalized as assigning coordinates to data points such that Euclidean distances estimated from the coordinates can best fit measured distances:

$$\widehat{P} = \arg\min_{P} \sum_{i,j \in P} (\delta_{ij} - d_{ij})^2 \tag{5.1}$$

We use a short deduction below to show how MDS can be used to embed a hop count distance metric space into a Euclidean space. According to the embedding defined in Definition 1, we have

$$\delta_{ij}^2 = |\mathbf{p}_i - \mathbf{p}_j|^2 = (\mathbf{p}_i - \mathbf{p}_j)^t(\mathbf{p}_i - \mathbf{p}_j) = \mathbf{p}_i^t\mathbf{p}_i + \mathbf{p}_j^t\mathbf{p}_j - 2\mathbf{p}_i^t\mathbf{p}_j$$

By shifting matrix P to the center, nodes' coordinates $\mathbf{p}_i$ and $\mathbf{p}_j$ can be expressed as the function of hop count distance δ_{ij}. Please refer to [101] for the complete intermediate deduction steps.

$$\mathbf{p}_i^t\mathbf{p}_j = \frac{1}{2}(-\delta_{ij}^2 + \frac{1}{n}\sum_{j=1}^{n}\delta_{ij}^2 + \frac{1}{n}\sum_{i=1}^{n}\delta_{ij}^2 - \frac{1}{n^2}\sum_{i=1}^{n}\sum_{j=1}^{n}\delta_{ij}^2)$$

$$= f(\delta_{ij}^2)$$

$$[\mathbf{p}_i^t\mathbf{p}_j] = [f(\delta_{ij}^2)] \tag{5.2}$$

$$\text{Let } [f(\delta_{ij}^2)] = F, \text{we have } P^tP = F \tag{5.3}$$

Because F is symmetric, it can be decomposed through singular value decomposition (SVD) as:

$$P^tP = F = V\Sigma V^t$$

$$P^t = V\Sigma^{1/2}$$

$$[\mathbf{p}_1, \mathbf{p}_2, \ldots, \mathbf{p}_N]^t = [\mathbf{v}_1, \mathbf{v}_2, \ldots, \mathbf{v}_N] \begin{bmatrix} \sigma_1 & & & \\ & \sigma_2 & & \\ & & \ddots & \\ & & & \sigma_N \end{bmatrix}^{1/2}$$

Here, $\sigma_1 \geq \sigma_2 \geq \ldots \geq \sigma_r \geq \sigma_{r+1} = \sigma_{r+2} = \ldots \sigma_N = 0$ are the rank-ordered set of singular values. Let $\sigma_i^{1/2} \equiv \lambda_i$. We have

$$[\mathbf{p}_1, \mathbf{p}_2, \ldots, \mathbf{p}_N] = [\lambda_1\mathbf{v}_1, \lambda_2\mathbf{v}_2, \ldots, \lambda_N\mathbf{v}_N]^t. \tag{5.4}$$

From Eqn (5.4), we have node i's N-dimensional coordinate

$$\mathbf{p}_i = [\lambda_1\mathbf{v}_{1i}, \lambda_2\mathbf{v}_{2i}, \ldots, \lambda_N\mathbf{v}_{Ni}]^t$$

such that $\delta_{ij} = |\mathbf{p}_i - \mathbf{p}_j|$ for pairwise nodes i and j. In the discussion below, we show how to reduce the dimensionality of $\mathbf{p}_i$ to realize the *exact embedding* and *approximate embedding*.

For exact embedding

$$\text{Let } \mathbf{p}_i' = [\lambda_1 \mathbf{v}_{1i}, \lambda_2 \mathbf{v}_{2i}, \ldots, \lambda_r \mathbf{v}_{ri}]^t.$$

It can be easily verified that $\delta_{ij} = |\mathbf{p}_i - \mathbf{p}_j| = |\mathbf{p}_i' - \mathbf{p}_j'|$ since $\lambda_{r+1} = \lambda_{r+2} = \ldots = \lambda_N = 0$. As a result, we construct a r-dimensional coordinate for node i such that $r < N$ and hop count distances between pairwise nodes can be exactly inferred from r-dimensional coordinates.

For approximate embedding

$$\text{Let } \mathbf{p}_i'' = [\lambda_1 \mathbf{v}_{1i}, \lambda_2 \mathbf{v}_{2i}, \ldots, \lambda_m \mathbf{v}_{mi}]^t, \ m < r.$$

The difference between the hop count distance and the Euclidean distance estimated from m-dimensional coordinates is:

$$\begin{aligned} \delta_{ij}^2 - |\mathbf{p}_i'' - \mathbf{p}_j''|^2 =& |\mathbf{p}_i' - \mathbf{p}_j'|^2 - |\mathbf{p}_i'' - \mathbf{p}_j''|^2 \\ =& \sum_{m+1 \le k \le r} \lambda_k^2 (\mathbf{v}_{ki} - \mathbf{v}_{kj})^2 \end{aligned} \tag{5.5}$$

Based on Eqn (5.5), we have $|\mathbf{p}_i'' - \mathbf{p}_j''| \to \delta_{ij}$ when $m \to r$. This reflects the tradeoff between the accuracy of network expression and the size of network expression. Moreover, if rank-ordered singular value λ_i quickly converge to zeros after the first several most important ones, we have $|\mathbf{p}_i'' - \mathbf{p}_j''| \approx \delta_{ij}$ for a relative small m. This is the case for network embedding and is verified in our performance evaluations. Here, MDS not only reveals the inherent tradeoff between the accuracy and size of a network

expression, but also provide a viable approach to efficiently approximate a network topology. The approximation of the embedding can be quantified as distortion *stress* defined as below.

$$\theta = \frac{\sum_{i,j \in P} (\delta_{ij} - d_{ij})^2}{\sum_{i,j \in P} \delta_{ij}^2}$$

The MDS is based on principal component analysis(PCA)[101] which can be intuitively explained by the example in Fig. 5.5, where a set of points are distributed in a 2-dimensional XY space. For points $\mathbf{p_1}$ and $\mathbf{p_2}$, their Euclidean distance can be calculated as $|\mathbf{p_1 p_2}| = \sqrt{(\mathbf{p}_{1x} - \mathbf{p}_{2x})^2 + (\mathbf{p}_{1y} - \mathbf{p}_{2y})^2}$. If we rotate the XY space to X'Y' space, the same distance can be calculated as $|\mathbf{p_1 p_2}| = \sqrt{(\mathbf{p}_{1x'} - \mathbf{p}_{2x'})^2 + (\mathbf{p}_{1y'} - \mathbf{p}_{2y'})^2}$. Here $(\mathbf{p}_{1y'} - \mathbf{p}_{2y'})^2 \approx 0$ because all points are distributed close to X' axis. Consequently, we have $|\mathbf{p_1 p_2}| \approx \mathbf{p}_{2x'} - \mathbf{p}_{1x'}$, i.e. the distance between points $\mathbf{p_1}$ and $\mathbf{p_2}$ in a 2-dimensional space can be approximated by their 1-dimensional coordinates. This example shows that by changing the view of the same data set, distances information can be well approximated in a lower dimensionality.

In practice, we can use the following steps to use MDS embedding to assign coordinates to nodes in a wireless sensor network.

1. Obtain the network topology by base station, which floods topology detection packets to an entire network and collects the connectivity information between neighboring nodes based on nodes' responses forwarded back along the reversed paths of flooding.

2. Use Dijkstra algorithm to compute hop count distances between pairwise nodes

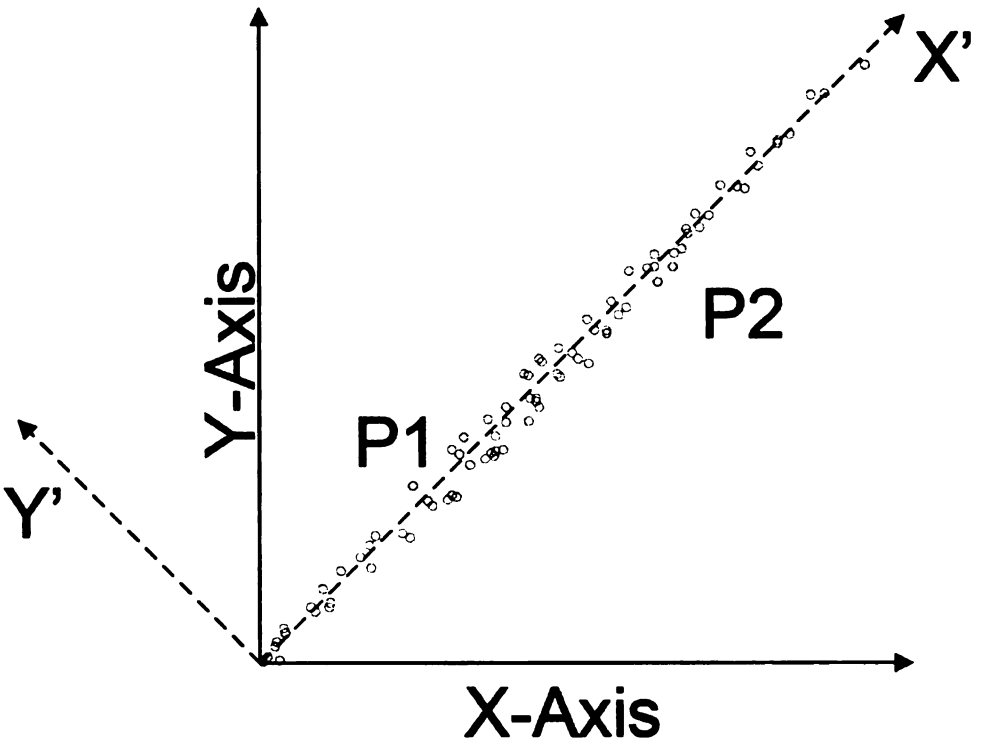


Figure 5.5 Principle component analysis

in the network topology.

3. Based on hop count distances between pairwise nodes, use MDS to embed the network topology into an Euclidean space where each node is assigned a coordinate.

4. Base station sends coordinates to corresponding nodes.

The algorithms above incurs massive communication messages to detect network topologies and uses a centralized computation to assign nodes' coordinates. In the following section, we investigate how to embed a network topology in a distributed fashion.

5.3.4 Embed network topologies in a distributed fashion

In this section, we show how to embed a network topology in a distributed fashion by sampling a portion of nodes in the network. We randomly select M nodes as reference points (beacons) from a network of size N . Each beacon k floods a beacon message to the network which contains a hop counter initialized as zero. The hop counter is increased by one when the message is forwarded to a next hop. By finding out the smallest hop counter among all the received beacon messages, a node can infer its hop count distance to beacon k. Based on the received messages sent out by all M beacons, node i can construct its hop count distance vector as below.

$$\mathbf{x}_i = [x_{i1}, x_{i2}, \ldots, x_{iM}]^t$$

Here, x_{ij} is the hop count distance from node i to beacon j.

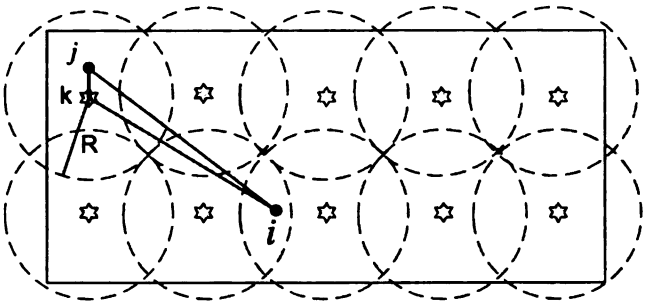


Figure 5.6 Beacon coverage with radius R

In order to reduce the communication cost, we only measure hop count distances from a node to M beacons instead of all the N nodes of a network. When sufficient beacons are uniformly distributed, it is possible to infer characteristics of a network topology through beacon sampling. As a result, we can achieve reasonable embedding accuracy based on partial observations, i.e. hop count distance measurements to a set of beacons instead of an entire network. To illustrate the relationship between embedding accuracy and size of beacon set, we start with some definitions below.

Definition 3 *Node i's minimum beacon distance l_i is defined as the hop count distance from node i to the nearest beacon, i.e. $l_i = \min_{j \in K} \delta_{ij}$, where K is the beacon set.*

Definition 4 *Beacon coverage radius R is defined as the maximum l_i for all nodes of a network, i.e. $R = \max_{i \in P} l_i$, where P is the node set of a network.*

The intuition of the definition above is that all the deployed area is covered by the union of circular regions centered at beacons with radius R as shown in Fig. 5.6. Accordingly, any node can find a beacon within the range of hop count distance R.

Proposition 2 *By only measuring hop count distances to a set of beacons, a network topology can be approximately embedded into a space in which the distance estimation error is bounded by beacon coverage radius R.*

Proof: If we can measure hop count distances to all the nodes of a network of size N, the network can be exactly embedded into a hop count distance metric space (X, δ) such that 1) node i's per-destination state is described by its coordinate as $\mathbf{x}_i = [x_{i1}, x_{i2}, \ldots, x_{iN}]$; 2) hop count distance between node i and j can be accurately inferred as $\delta_{ij} = \delta(\mathbf{x}_i, \mathbf{x}_j) = x_{ij}$.

If hop count distances are only available to a set of beacons of size M, the network can be approximately embedded into a M-dimensional space (X', δ') where node i's coordinate is described as:

$$\mathbf{x}'_i = [x'_{i1}, x'_{i2}, \ldots, x'_{iM}]^t.$$

Here x'_{ij} is the hop count distance from node i to the jth beacon. The distance function δ' is defined as

$$\delta'(\mathbf{x}'_i, \mathbf{x}'_j) = \delta_{ik}, \text{ where } k = \arg\min_{r \in K} \delta_{jr}.$$

Intuitively, we find the nearest beacon k to node j and use hop count distance δ_{ik} to estimate distance between pairwise nodes i and j. The distance estimation error in the embedded space is:

$$|\delta_{ij} - \delta(\mathbf{x}'_i, \mathbf{x}'_j)| = |\delta_{ij} - \delta_{ik}|.$$

Because hop count distance defines a metric which satisfies that triangle inequality,

as shown in Fig. 5.6, we have

$$|\delta_{ij} - \delta_{ik}| \leq \delta_{jk} \leq R. \quad \square$$

From Proposition 2, we can conclude that a dense and uniform beacon distribution is helpful to minimize the distance estimation error since such a beacon distribution can achieve small beacon coverage radius R. Increasing beacon density, however, will lead to higher dimensionality of the embedded space if we directly assign node coordinates in which each dimension is the hop count distance to one of beacons. In order to achieve sufficient embedding accuracy while preserving low dimensionality of the embedded space, we use the same strategy of above section to reduce the dimensionality through two steps. In the first step, the low dimensional embedded space is learned from beacon set such that each beacon is assigned a virtual coordinate. In the second step, non-beacon nodes' coordinates are calculated by fitting hop count distances to all the beacons. The details of these two step are discussed below.

Step one: a designated leading beacon collects the hop count distance vectors from all other beacons and construct beacons' hop count matrix X as below.

$$X = [\mathbf{x}_1, \mathbf{x}_2, \ldots, \mathbf{x}_m], \text{ where } \mathbf{x}_i \in K.$$

Based on the matrix X, We use the MDS to embed the hop count distance metric space (X, δ) into a low-dimensional Euclidean space (P, d) such that each beacon is assigned a virtual coordinate $\mathbf{k}_i$.

Step two: we use least square fitting to embed a non-beacon node into the low-dimensional Euclidean space such that the differences between hop count distances

and the corresponding Euclidean distances from the node to all beacons are minimized.

$$\widehat{\mathbf{p}}_i = \arg\min_{\mathbf{p_i}} \sum_{j=1}^{M} (|\delta_{ij} - |\mathbf{p}_i - \mathbf{k}_j|)^2.$$

Consequently, node i's virtual coordinate is the optimal result of the object function above which can best fit the estimated Euclidean distances d_{ij} to the hop count distances δ_{ij} from node i to all the beacons.

The *distributed multidimensional scaling* (DMDS) approach proposed above embeds nodes to a low-dimensional Euclidean space by fitting hop count distances to a set of beacons, which is all the observed network topological information available to us. The performance evaluation in next section shows that the hop count distance between any pairwise nodes can be accurately inferred from the virtual coordinates, despite the fact that the virtual coordinated are constructed from the partial observations on beacons. The topological characteristics of an entire network can be sampled from beacons because of two reasons:

1. Randomly selected beacons are uniformly distributed in the network, which makes them good candidates to represent the structure of a network topology.

2. High redundancy is shared between neighboring nodes. As we discussed above, neighboring nodes have similar hop count distances to the third node due to the triangular inequality. Wealthy topological information can be preserved by only using hop count distances to a few beacons because the hop count distance to other nodes close to beacons are often redundant.

5.4 ETX distance based greedy forwarding

Both topology aware routing and location aware routing assume that wireless channels between neighboring nodes have perfect reception, i.e. packets can always be successfully delivered. Based on this assumption, the shortest path between the source and the destination is the optimal routing path that requires the least number of packet forwarding. However, radio signals attenuate in transmission and are susceptible to environmental interference, which may lead to corrupted packets. In such a case, packets need to be retransmitted before they can be successfully delivered. Since the topology aware routing or the location aware routing aims to find the shortest path, each individual hop has long transmission range and low quality. Consequently, the greedy forwarding will fail to find the optimal routing path comprising high quality links.

In this section, we further improve the end-to-end routing performance of the greedy forwarding by improving the expression accuracy of a wireless sensor network. Unlike the simple geographic model where the communication route is approximated by the geographic path, we embed a wireless sensor network into a Euclidean space where nodes' virtual distance is equal to the number of expected transmissions (ETX) for a packet to be successfully delivered from a source to a destination. Because the virtual distance directly reflects the end-to-end communication channel quality, the greedy forwarding can guide a packet along the optimal routing path which has the shortest virtual distance. Here we use the number of expected transmissions instead of the hop count distance as the routing metrics to improve the routing performance

because the former has the following properties: i) it reflects the underlying wireless channel quality between neighboring nodes; ii) it also reflects the end-to-end channel quality from a source to a destination.

In the follow discussion, we first show how to evaluate the quality of wireless channels. After that, we describe the ETX distance based greedy forwarding.

5.4.1 Evaulation of underlying wireless channel

The communication quality and cost of a wireless channel can be evaluated by various metrics such as packet reception ratio, transmission delay, and throughput. A detailed comparison of three link-quality metrics - expected transmission count (ETX)[60], per-hop round trip time (RTT)[61], and per-hop packet pair delay - has been conducted in [62], which concludes that the ETX metric has the best performance in a static wireless sensor network. In this chapter, we show that ETX is an ideal metric to define the virtual distance to support the greedy forwarding to achieve optimal end-to-end routing performance.

To route data over unreliable wireless channels, hop-by-hop recovery is usually preferred over end-to-end recovery[68]. The hop-by-hop recovery is realized by acknowledging received packets and retransmitting lost packets. For neighboring nodes i and j in a route path, the receiver j will send back an acknowledgment to sender i when a packet is correctly delivered; and the sender i will retransmit a packet if it has not received the acknowledgment within a certain time period after a packet transmission. Assume that the packet loss rate from node i to j is P_{ij} and the packet loss rate from node j to node i is P_{ji}. The probability of a successful packet trans-

mission is $(1 - P_{ij})(1 - P_{ji})$, and the expected number of retransmissions defined as the expected transmission count metric in [60] between node i and j is:

$$ETX(i,j) = 1/(1 - P_{ij})(1 - P_{ji}).$$

Assume that a pairwise node p_1 and p_n has the routing path l comprising intermediate nodes $p_2, p_3, \ldots, p_{n-1}$; we have the expected transmission count of the routing path l as:

$$ETX(l) = \sum_{i=1}^{n-1} ETX(p_i, p_{i+1})$$

It has been proposed in [60] to incorporate the ETX into the on-demand routing such as DSR to find the optimal routing path between pairwise wireless nodes. In the combined approach, the source broadcasts route probing message to an entire network. The routing paths can be discovered when the destination sends back the response messages along the reversed paths of the probing message. Among all the paths connecting source and destination, the optimal routing path can be determined with the minimal ETX.

In this section, we propose to combine the ETX metric with the greedy forwarding such that the optimal routing path can be found without reliance on the frequently broadcast route probing messages.

5.4.2 Greedy forwarding based on the ETX distance

We define the ETX virtual distance between pairwise nodes $\mathbf{x}_i$ and $\mathbf{x}_j$ as the minimal ETX among all the routing paths connecting $\mathbf{x}_i$ and $\mathbf{x}_j$, i.e.

$$\delta(\mathbf{x}_i, \mathbf{x}_j) = \min_{l_i \in L} ETX(l_i),$$

where L is the set of routing paths connecting nodes $\mathbf{x}_i$ and $\mathbf{x}_j$. In this section, we assume that ETX distance between pairwise nodes in a wireless sensor network can be inferred from their virtual coordinates.

The greedy forwarding can achieve optimal end-to-end routing performance based on ETX distance comparison between neighboring nodes. In order to achieve that, we need to assign nodes virtual coordinates from which ETX distances can be accurately recovered. We can apply the same MDS embedding technique proposed in the topology aware routing by simply replacing the hop count distance metric with the ETX distance metric.

Based on the comparison of the ETX distances between neighboring nodes, the greedy forwarding can determine the next hop as follows: suppose a packet need to be forwarded to the destination $\mathbf{x}_k$. Let node $\mathbf{x}_i$ be the intermediate node with the routing packet and set N define all the neighbors of node $\mathbf{x}_i$. The next forwarding hop can be selected from the neighbor set N as:

$$\widehat{\mathbf{x}}_j = \arg \min_{\mathbf{x}_j \in N} (\delta(\mathbf{x}_i, \mathbf{x}_j) + \delta(\mathbf{x}_j, \mathbf{x}_k)), \tag{5.6}$$

i.e. the packet is greedily forwarded to the next hop $\mathbf{x}_j$ which minimizes the summary of the ETX distances $\delta(\mathbf{x}_i, \mathbf{x}_j)$ and $\delta(\mathbf{x}_j, \mathbf{x}_k)$.

5.5 Performance evaluation of topology aware routing

To evaluate topology aware routing (TAR), we compare the TAR, location aware routing (LAR), logical coordinate routing (LCR), and beacon vector routing (BVR) in this section through intensive simulation. Our evaluation focuses on the effectiveness of the constructed coordinates to support the greedy forwarding, i. e. the routing success rate of greedy forwarding given a node coordinate assignment of a network. To clearly evaluate the effectiveness of various coordinate assignment schemes, we test the routing success rate only based on the greedy forwarding and do not resort to any recovery solutions to the local minimum. Before proceeding to the detailed performance comparison, we brief LCR and BVR below and summarize the difference between TAR, LCR and BVR.

5.5.1 Difference between topology aware routing, beacon vector routing and logical coordinate routing

Previous work of beacon vector routing[8] and logical coordinate routing (LCR)[55] is similar to our proposed virtual coordinate approach in that nodes' coordinates are constructed from the hop count distances between pairwise nodes. Both BVR and LCR directly assign node coordinates based on their hop count distances to a set of beacons:

$$\mathbf{p}_i = [p_{i1}, p_{i2}, \ldots, p_{iM}],$$

where p_{ik}($1 \leq k \leq M$) is the hop count distance from node i to beacon k. Based on nodes' coordinates, LCR calculates the distance between pairwise nodes (i, j) as:

$$d(i,j) = |\mathbf{p}_i - \mathbf{p}_j| = (\sum_{k=1}^{m}(p_{ik} - p_{jk})^2)^{1/2}.$$

The BVR uses a different approach to estimate distance from node coordinates as following. Suppose d is the destination node, the distance from node s to node d is calculated as below:

$$\delta_k(s,d) = A\delta_k^+(s,d) + \delta_k^-(s,d),$$

where

$$\delta_k^+(s,d) = \sum_{i \in C_k(d)} \max(\mathbf{p}_{si} - \mathbf{p}_{di}, 0);$$
$$\delta_k^-(s,d) = \sum_{i \in C_k(d)} \max(\mathbf{p}_{di} - \mathbf{p}_{si}, 0).$$

Here, $C_k(d)$ is the set of the k closest beacons to destination d. $\delta_k^+(s,d)$ is the sum of differences when the selected beacons are closer to destination d than to node s. $\delta_k^-(s,d)$ is the sum of differences when the selected beacons are farther to the destination d than to the node s. The next hop j is chosen such that the $\delta_k(j,d)$ is minimized among all neighbors. The intuition behind BVR is to select nearest beacon set $C_k(d)$ to the destination d and greedily forward a packet towards the beacon set $C_k(d)$. Since beacons in $C_k(d)$ are close to the destination d, the packet is forwarded along the right direction. BVR improves LCR in that only the k ($k < M$) closest beacons (routing beacons) to destination d other than all the M beacons are involved in the distance comparison. As a result, BVR only needs to maintain hop count distances to k routing beacons as the destination address in a packet's head and therefore uses less bytes than LCR.

Our proposed topology aware routing (TAR) differs from BVR and LCR in two aspects:

1. TAR assigns nodes' coordinates by minimizing the differences between estimated Euclidean distances and measured hop count distances. In contrast, the distance estimated by LCR and BVR has less geometric relationship with the hop count distances between pairwise nodes. As a result, TAR can compare hop count distances between neighboring nodes more precisely and therefore achieve better routing performance of greedy forwarding.

2. In BVR and LCR, the dimensionality of nodes' coordinates is equal to the size of beacon set. As a result, increasing size of beacon set to sample more characteristics of a network topology will inevitably increase dimensionality of nodes' coordinates, which leads to larger size of routing states. In contrast, TAR uses MDS to effectively reduce dimensionality of nodes' coordinates such that wealthy topological information can be efficiently encoded into low-dimensional coordinates.

Since the fundamental differences between various coordinate assignments are their capabilities to extract the topological characteristics from a wireless sensor network, our evaluation focuses on the geometric characteristics of the network layer where the network topology has major impact on the routing success rate. We assume the physical layer and MAC layer of communications links have the same impacts on compared routing protocols. To isolate the interference of the physical and MAC layer to our routing performance comparison, we make the assumption

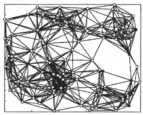


Figure 5.7 Square layout

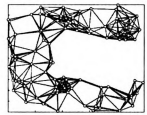


Figure 5.8 C layout

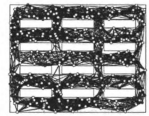


Figure 5.9 Street layout

that the communication links are reliable and can always deliver a packet between neighboring nodes. Such an abstraction can help us focus on how network topologies affect the greedy forwarding routing and how well network topologies are described by coordinate assignment solutions.

5.5.2 Simulation configuration and evaluation metrics

We use three configurations to simulate three representative scenarios of wireless sensor network deployments. The first scenario is that wireless nodes are deployed in an open flat area, which is simulated by a square shape network topology as shown in Fig. 5.7. The second scenario is that wireless nodes are deployed along a long path such as a river or valley, which often consists of segments of C shapes or S shapes. We generate the C shape network topology to simulate this scenario as shown in Fig. 5.8. The third scenario is that wireless nodes are deployed in streets of an urban area, where radio communications are blocked by buildings. We generate a square shape where multiple rectangles were positioned to simulate buildings as shown in Fig. 5.9.

To compare the performances of TAR, LAR, BVR and LCR, we repeate the following experiment in the same network topology configuration for each routing

scheme. In each experiment, the same randomly selected 1000 pairwise nodes are used as sources and destinations to test greedy forwarding routing based on nodes coordinates assigned by one of the routing schemes. We count the total number of routings which fail to deliver a packet from the source to the destination due to the local minimum. The percentage of failed routings (Routing Failure Rate) is used to measure the effectiveness of the routing scheme. All the metrics used to evaluate the routing performance are summarized below.

- **Routing Failure Rate**: The percentage that a packet cannot be delivered by the greedy forwarding from the source to the destination.
- **Average Routing Length**: The average number of hops that a packet need to be forwarded from a source to a destination.
- **Radio Transmission Range**: The maximum range that the wireless radio signals can be transmitted by a node.
- **Beacon Density**: the number of beacons used in a network.
- **Virtual Coordinate Dimensionality**: the number of elements used in a node's virtual coordinate.

5.5.3 Performance comparison between LAR and TAR-MDS

We first compare the performance between the location aware routing (LAR) and the topology aware routing using the multidimensional scaling approach (TAR-MDS). In order to set up a fair play, we intentionally embed network topologies into *two*

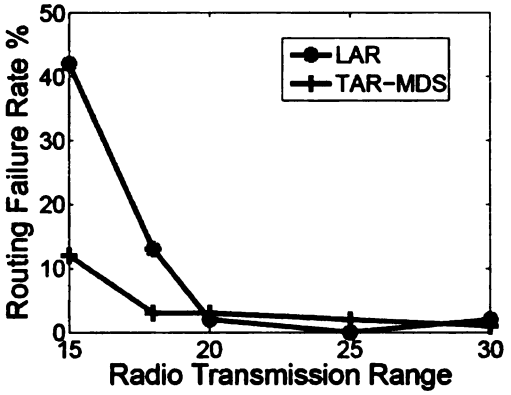


Figure 5.10 Routing failure rate in a square shape network

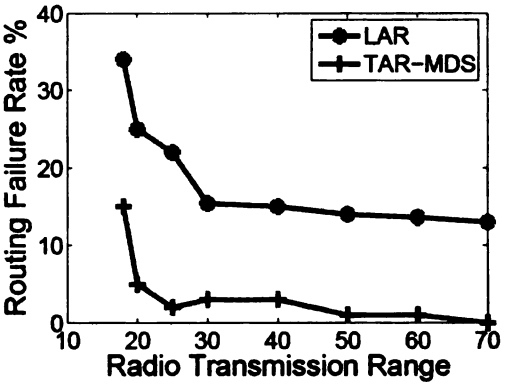


Figure 5.11 Routing failure rate in a C shape network

dimensional Euclidean spaces through MDS. Consequently, both routing approaches have the same costs in terms of the size of routing states maintained by individual nodes and the length of the destination address in a packet head.

Fig. 5.10 shows that TAR-MDS has much lower routing failure rates than LAR in the square configuration when the transmission range is short. This is because voids exist in a sparse network topology which leads to the mismatch between the lengths of the shortest path and the geographic distance, i. e. the shortest path has to detour around the voids and deviate away from the straight line. Due to the mismatch between the lengths of the shortest path and the geographic distance, a node may have a neighbor which is one hop closer to the destination but geographically farther to the destination.

Fig. 5.10 also shows that both LAR and TAR has similar routing failure rates when nodes' radio transmission range is long. This is because voids disappear in a dense network topology and the lengths of the shortest paths in the network topology can be

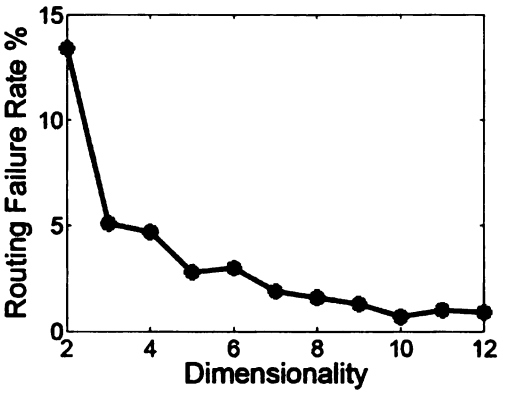


Figure 5.12 Impact of virtual coordinates' dimensionality on routing failure rate

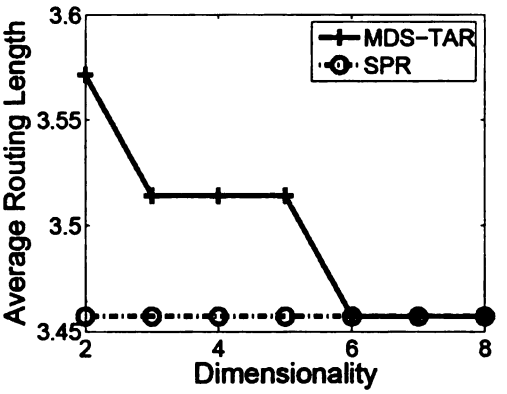


Figure 5.13 Routing quality of MDS-TAR

well approximated by the geographic distances. However, in a concave network such as C shape network, the TAR-MDS always has lower routing failure rates than the LAR, which is shown in Fig. 5.11. This is because in the C shape network topology, the shortest paths between pairwise nodes have to detour along the C shape and deviate away from the straight line directly connecting sources and destinations.

5.5.4 Impact of virtual coordinates' dimensionality in TAR-MDS

Fig. 5.12 shows that the routing failure rate of TAR-MDS can be significantly decreased if we increase the dimensionality of node coordinates from 2 to 6. After that, the routing failure rate eventually converges to zero. Fig 5.13 shows that TAR has the same average number of hops per routing as the shortest path routing (SPR) when the dimensionality is increased to 6. The routing failure rate decreases with the increase of virtual coordinates' dimensionality because higher dimensional virtual coordinates preserve higher fidelity of the network topology in the embedding. On

the other hand, a low routing failure rate can be achieved at a fairly low dimensionality (6 in this experiment) because the MDS is an effective dimensionality reduction technique which can accurately embed a network topology into a low dimensional Euclidean space.

Despite its effectiveness, the TAR-MDS algorithm is a centralized approach which relies on the global view of an entire network. In the following section, we evaluate the distributed version of the TAR, i.e. topology aware routing using distributed multidimensional scaling(TAR-DMDS), which uses less communication costs and is more suitable to the distributed wireless sensor network.

5.5.5 Impact of beacon set size on the routing performance

We investigate the impact of beacon set size on the routing performance of TAR-DMDS, BVR and LCR. All the three approaches share the similarity in that nodes' coordinates are constructed based on hop count distances to a set of beacons. Fig. 5.14 shows the relationship between the routing failure rates and the size of beacon set in C network topology. This figure shows that a small number of beacons (close to 4) is insufficient to represent the characteristics of the entire network topologies and a certain number of beacons (more than 20 in these two configurations) are required in order to achieve relatively low routing failure rates.

The discussion above shows that certain number of beacons is required to sample sufficient topological information from a network topology. We further investigate the relationship between the number of required beacons and the size of sampled network. We generate C shape network topologies at various size of 400, 800 and 1600 nodes.

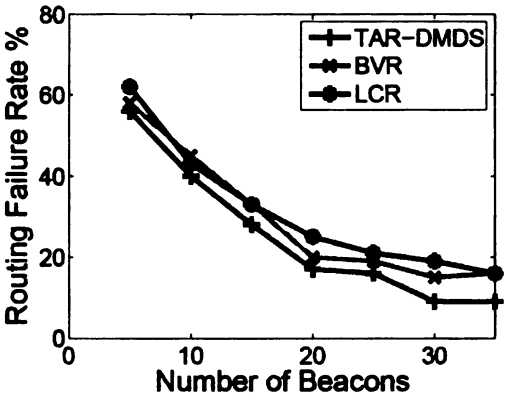


Figure 5.14 Routing failure rate v.s. number of beacons in C shape network

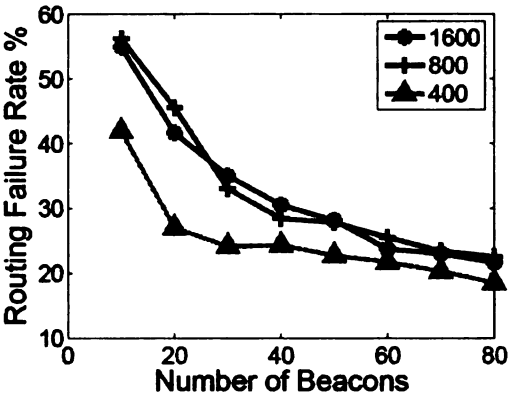


Figure 5.15 Scalability of topology aware routing

All the three networks have the same node density and are similar to each other in topological structure. We test the routing failure rate at different beacon sample size in all three networks. The result is shown in Fig. 5.15, which illustrates that their routing failure rates are close to each other when the number of beacons reaches a certain value (60 in our test). Based on this test, we conclude that the required size of sampling beacons mainly depends on the complexity of network topology instead of the total number of nodes, and the number of required beacons does not scale with the size of a network.

5.5.6 Impact of node coordinates' dimensionality in TAR-DMDS

The performance evaluation above shows that a certain number of beacons are required to achieve low routing failure rate. Large size of beacon set, however, will inevitably lead to high dimensionality of node coordinates and therefore large size routing states and long destination addresses in packets' heads. BVR uses routing beacons to reduce the length of destination addresses in packets' heads. Our proposed

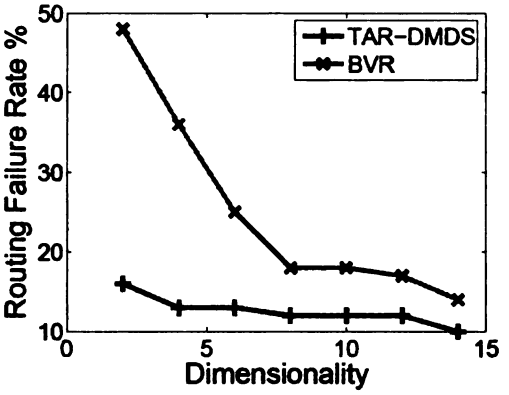


Figure 5.16 Impact of dimensionality on routing failure rate

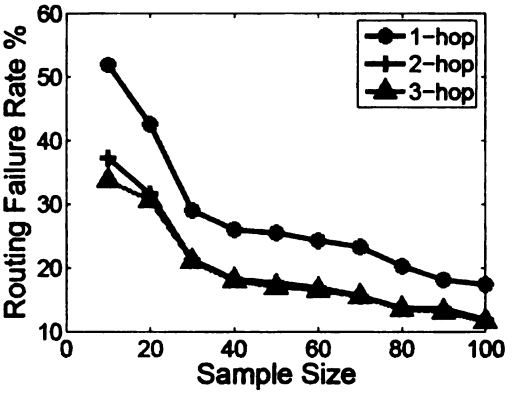


Figure 5.17 Routing failure rate v.s. sample size under different neighborhood scope

TAR-DMDS can achieve both small size routing states and short length of destination address by utilizing the dimension reduction technique. In this comparison, we view the number of routing beacons of BVR as the coordinates' dimensionality. We compare routing failure rates of TAR-DMDS and BVR with different coordinates' dimensionalities. Fig. 5.16 shows that the TAR-DMDS has lower routing failure rate than BVR in various dimensionalities. Especially in the low dimensionality such as 2, the routing failure rate of TAR-DMDS is much lower than that of BVR. The TAR-DMDS outperforms in low dimensionality because of two reasons: (1) the virtual coordinates of TAR-DMDS are constructed by directly fitting the hop count distances, which sets up a good basis for precise hop count distance comparison between neighboring nodes; (2) BVR only uses routing beacons in distance comparison, while TAR-DMDS embeds nodes' coordinates from the hop count distances to all available beacons and encodes more topological information into coordinates.

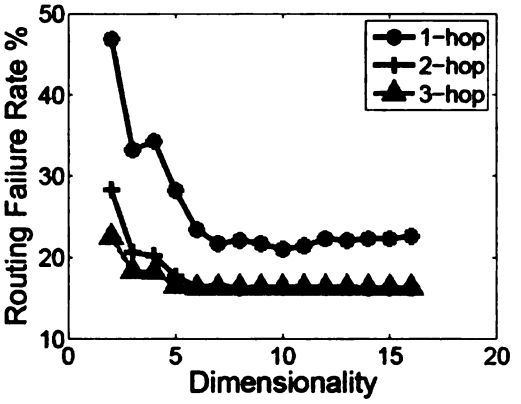


Figure 5.18 Routing failure rate v.s. dimensionality under different neighborhood scope

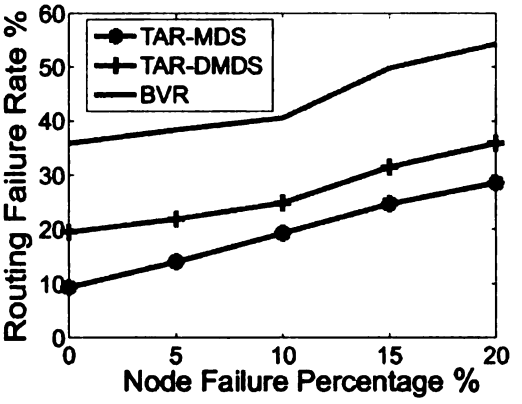


Figure 5.19 Robustness of topology aware routing

5.5.7 Routing performance based on neighbors' coordinates within two hops scope

The performance evaluation above shows that the topology aware routing can achieve high routing success rate at low dimensional virtual coordinates. This intrigues us to further investigate the routing performance when a node can learn coordinates which are two hops away. Due to the low dimensionality of virtual coordinates, the routing status can still be maintained at small size even a node preserve all the coordinates of nodes within two hops. The only cost here is the extra communication messages to exchange coordinates between two hop away nodes.

The results of performance evaluation are shown in Fig. 5.17 and Fig. 5.18. We can see that the same routing failure rates can be achieved at smaller sample size or lower virtual dimensionality when a node's routing scope is expanded to two hops. However, the performance improvement is not obvious when the scope is expanded from 2 hops to 3 hops.

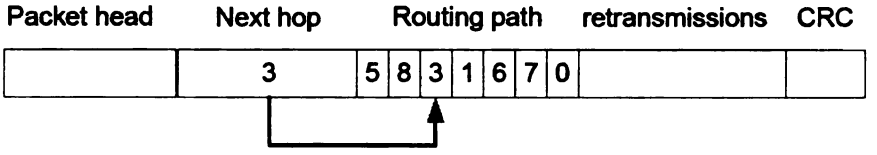


Figure 5.20 Packet format in path driven routing

5.5.8 Robustness of topology aware routing

In order to evaluate the robustness of topology aware routing, we use the street layout network topology as the testbed. We simulate nodes' failure by taking all the edges from failed nodes to their neighbors such that packets cannot be forwarded from/to failed nodes. When TAR-MDS, TAR-DMDS and BVR are used for routing packets, certain percentage of failed nodes are randomly selected from the network. We vary the percentage of failed nodes from 5% to 20% to investigate the resilience of topology aware routing to the node failures.

Fig. 5.19 shows that the routing failure rates of both TAR-MDS and TAR-DMDS are increased when the node failure rate increases. The TAR-MDS and TAR-DMDS fail to delivery packets because the routing paths discovered by TAR-MDS and TAR-DMDS are broken due to nodes' failures. However, we can observe that TAR-MDS and TAR-DMDS do show certain resilience to nodes' failures based on the relative small slopes of the two curves, which have the similar slopes as that of BVR. The performance evaluation shows that TAR-MDS and TAR-DMDS have the capabilities to tolerate nodes' failures by bypassing failed nodes.

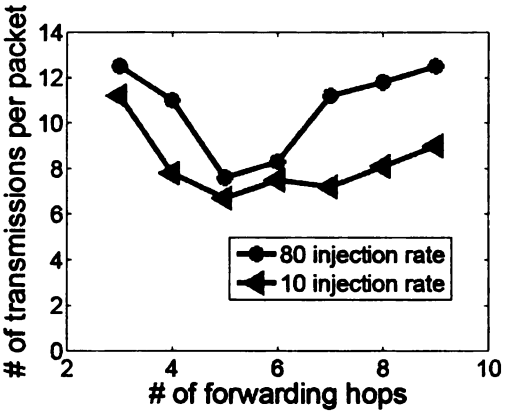


Figure 5.21 Experiment of mutlihop forwarding

5.6 Performance evaluation of ETX distance based greedy forwarding

We evaluate the ETX distance based greedy forwarding through a small scale experiment based on MICA2 platform and a large scale simulation based on TOSSIM/TYTHON[102]. TOSSIM is a bit level simulator which shares the same TinyOS code with the MICA2 platform. We first evaluate the packet transmission in multihop forwarding on the MICA2 platform. After that, we compare the greedy forwarding based on the ETX distance with the location aware routing in the TOSSIM simulator. Besides the metric used in Section 5.5, the following metric is also used evaluation.

Number of transmissions per packet: The number of transmissions per packet is the total number of transmissions, including retransmissions required for a packet to be forwarded from the source to the destination.

5.6.1 Evaluate packet transmission in MICA2 platform

Because the MAC layer of the current MICA2 sensor platform does not acknowledge packets successfully received, we implement the ACK-retransmission mechanism in the network layer. When an intermediate node receives a message, it will send an acknowledgment back to the previous hop before forwarding the message to the next hop. All the received messages are buffered in a queue and retransmitted until they are acknowledged by the next hop or the maximal retransmission threshold is reached. We implement a path driven routing in the network layer to evaluate packet transmission performance along different forwarding paths. The packet format is shown in Fig. 5.20, where the routing path field contains the sensor IDs of all the intermediate forwarding nodes. The next hop field is a pointer to the routing path field. The value of the next hop field is increased by one when a packet is forwarded along one hop. Based on the next hop field, we can find the next forwarding hop by looking up the routing path field. The retransmissions field records the total number of transmissions (including retransmissions) of the forwarded packet. Consequently, the number of transmissions required to route a packet from the source to the destination is available at the end of the routing path.

We arrange all the sensor nodes in a straight line and attach one laptop to the first node (source) and the other laptop to the last node (destination). A packet is injected to the routing path from the laptop attached to the source and the total number of packet retransmission is read at the laptop attached to the destination. By fixing the distance between the source and destination, we forward packet along

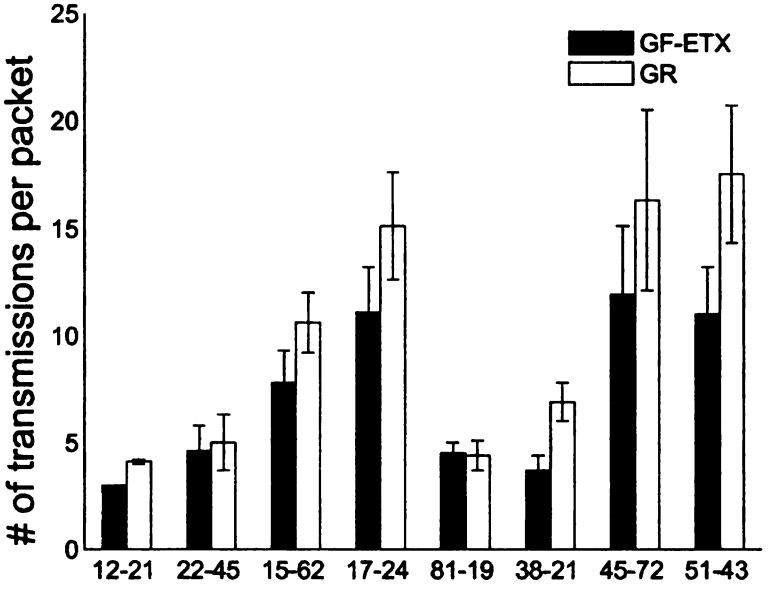


Figure 5.22 Number of transmissions between pairwise nodes

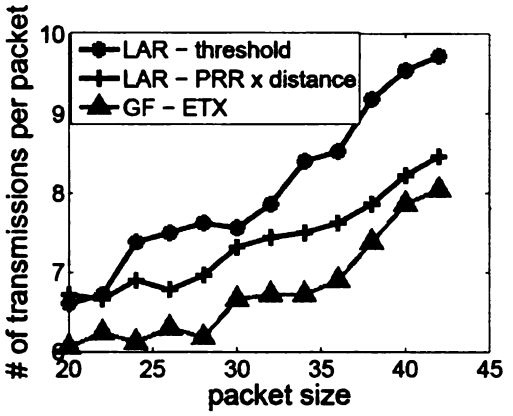


Figure 5.23 Number of transmissions under different packet sizes

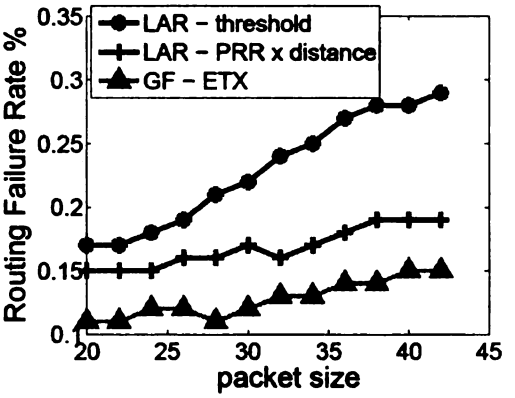


Figure 5.24 Packet failure rate under different packet size

different routing paths which consist of different number of intermediate forwarding nodes. We also test the multihop forwarding with different packet injection rates. The experimental result is shown in Fig. 5.21, which illustrates several properties of multihop forwarding in wireless channels: i) the high packet transmission rate will increase the average number of transmissions due to the channel collision of packet transmission; ii) the optimal number of forwarding hops exists for a pairwise nodes with a fixed transmission distance. Because a routing path with a small number of forwarding hops contains long distance low quality links, a packet has to be retransmitted multiple times in each individual link. On the other hand, a routing path with too many forwarding hops will lead to excessive packet forwarding. Besides, the densely distributed intermediate nodes have higher packet collision probability. We use the ETX distance to measure the optimal routing path between pairwise nodes. The ETX distance based greedy forwarding is further evaluated in the TOSSIM simulator.

5.6.2 Evaluate the ETX distance based greedy forwarding in TOSSIM

TOSSIM is a bit level simulator which can accurately simulate the radio transmission channels between wireless sensors. We use the lossy radio model which assumes each bit of the transmission packet has the probability of p to be flipped. The probability of bit flipping is measured from the real experiment. When a packet is received, the correctness is verified/recovered by forward error correction (FEC) code. A packet will be dropped if it cannot be recovered by the FEC verifying code. We use the TYTHON to control the simulated nodes in TOSSIM, which act as the two laptops

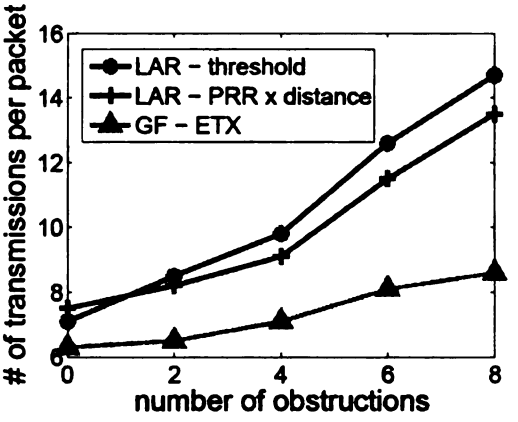


Figure 5.25 Number of transmissions under different obstructions

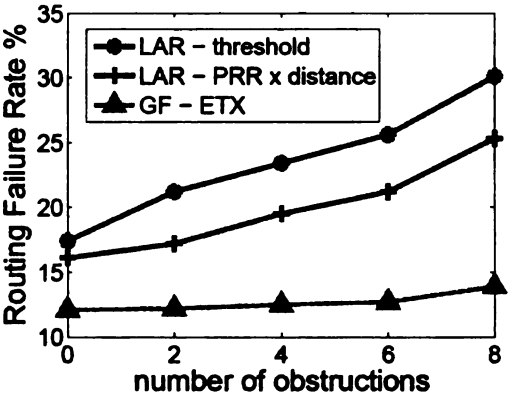


Figure 5.26 Packet failure rate under different obstructions

attached to the source and the destination. The pairwise source and destination are randomly selected from 100 nodes deployed in a 125 by 125 feet square area. The testing packet is injected to the simulated network through TYTHON which is forwarded along different routing paths computed from different routing algorithms. The average number of packet transmissions and the packet delivery ratio is logged by TYTHON. In order to minimize the interference between radio links, we inject the testing packets in a sequential order with a time interval between consecutive packets.

We first evaluate the greedy forwarding on 8 pairwise nodes randomly selected. We use both greedy forwarding based on ETX distance (GF-ETX) and the location aware routing (LAR) to forward packets between each pairwise nodes and record the average number of transmissions. Fig. 5.22 shows that the GF-ETX uses less number of transmissions to deliver packets from a source to a destination.

We further compare the GF-ETX, the location aware routing with threshold (LAR-threshold) and the location aware routing based on the product of the packet

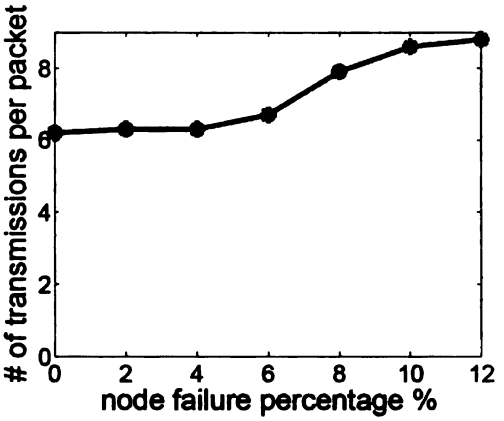


Figure 5.27 Number of transmissions under different failure percentage

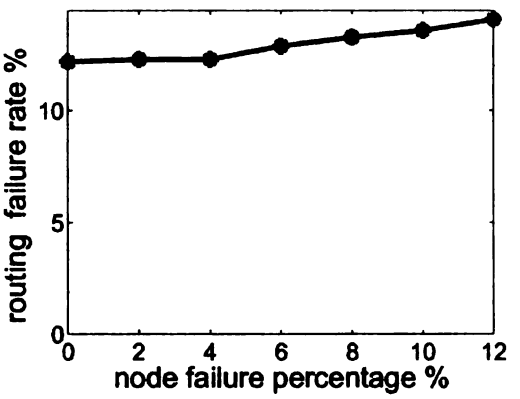


Figure 5.28 Routing failure rate under different failure percentage

reception rate and the forwarding distance (LAR-PRR x distance). Fig. 5.23 and Fig. 5.24 show that both the number of packet transmissions and the packet failure rate are increased with the increment of packet size. Moreover, the location aware routings are more susceptible to the packet size and have high average number of packet transmissions for larger size packets. This is because both the routing approaches tend to select low quality routing links with high bit flipping error and multiple bit errors may happen in long length packets which cannot be recovered by FEC code.

We compare the GF-ETX with the location aware routing in complex deployed environment by adding obstructions into the deployed area, which increases the spatial complexity of the network topology. Fig. 5.25 and Fig. 5.26 show that the greedy forwarding based on ETX distance is less affected by the interference from obstructions and can always learn the optimal routing paths in a complicated network topology. Because the ETX distance is a global metric which defines the end-to-end channel

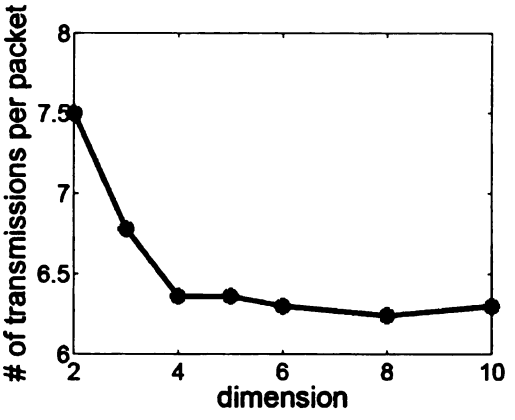


Figure 5.29 Number of transmissions under different dimensionality

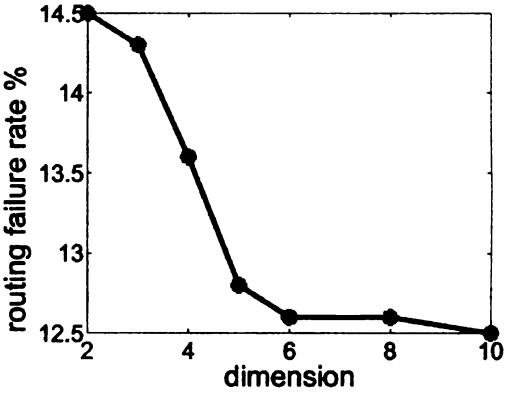


Figure 5.30 Routing failure rate under different dimensionality

quality between pairwise nodes, the greedy forwarding based on ETX distance can foresee the affection of obstructions and guide packet to bypass the obstruction along an optimal route.

We further evaluate how network dynamics, such as node failure, affect the routing performance of the ETX distance based greedy forwarding. We simulate nodes' failures by temporarily turning off nodes in TOSSIM. To keep the constant number of active nodes, a certain percentage of nodes keep turning on/off periodically. We vary the percentage of failed nodes from 2% to 12% to investigate the resilience of ETX distance based greedy forwarding to the node failures. Fig. 5.27 and Fig. 5.28 show that the impact of network dynamics on the routing performance is limited. This is because in a densely deployed wireless sensor network, multiple routing paths exist between pairwise nodes such that the greedy forwarding can successfully find a backup route by walking around failure nodes.

We evaluate the coding efficiency of ETX-embedding by varying the dimensional-

ity of the embedded space. Fig. 5.29 and Fig. 5.30 show that the routing performance is increased when the dimensionality is increased from 2 to 6, and converges after the dimensionality is greater than 6. The evaluation shows that the ETX-embedding can efficiently encode a wireless sensor network topology into small size virtual coordinates such that good routing performance can be achieved under low dimensionality embedded space.

5.7 Summary

Packet routing in wireless sensor networks is a challenging problem because data often needs to be routed in a purely flat network topology consisting of thousands of nodes which have limited resources to preserve routing states. Location aware routing shows its potential in that it uses simple greedy forwarding to deliver packets based on small size routing states. However, location aware routing uses the ideal geographic model that often oversimplified the spatial complexity of a wireless sensor network that are deployed in complicated environments. In this chapter, we propose to use graph embedding to achieve optimal end to end routing performance in complicated environment with small routing states. We first solve the local minimum problem by embedding a network topology to a Euclidean space where hop count distances can be recovered from node virtual coordinates. Based on the accurate hop count distance comparison between neighboring nodes, the greedy forward can find the right neighbor that is one hop closer to the destination and finally deliver the packet through consecutive hop-by-hop forwarding. We further show that the routing quality

can be improved by embedding the network topology to a Euclidean space where the number of expected transmissions (ETX) can be recovered from nodes' virtual coordinates. Guided by the ETX distance, the greedy forwarding can find the optimal routing path with the least number of transmissions to successfully deliver a packet from the source to the destination. We evaluate our proposed approaches in both simulations and experiments, which shows they can improve the routing quality in terms of the routing success rate and routing costs.

CHAPTER 6

Reliable Data Transmission in Wireless Sensor Networks

6.1 Motivation

Radio signals can be reflected and scattered by obstructions in complicated environments, which leads to unreliable wireless channels and packet loss in transmission. It is necessary to retransmit lost packets to ensure reliable data transmission in lossy wireless channels. In this chapter, we investigate how to realize the reliable data transmission with minimal overhead and maximal throughput. Two basic mechanisms have been widely used to achieve reliable data transmission over lossy channels. In time-out mechanism, a sender waits for an acknowledgement (ACK) from a receiver after a packet is sent out. If the ACK is not received within a certain time, the packet will be retransmitted. The time-out mechanism may degrade throughput of wireless sensor networks because **i**) its stop-and-wait nature reduces nodes' transmission rate; **ii**) the ACK packets consumes network bandwidth; **iii**) lost ACKs incur unnecessary retransmission. The extra overhead incurred by ACK has been partially mitigated

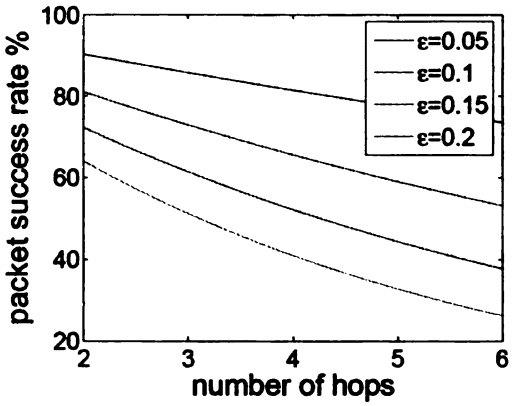


Figure 6.1 Packet success rate under different lossy channels

in RBC [69] that piggybacks grouped ACKs into forwarding data. The sequence based mechanism is the other approach to achieve reliable data transmission and has been successfully applied in the sliding-window algorithm of TCP protocol. However, originally designed for end-to-end protocols, the sequence based mechanism is not robust and incurs extra overhead when directly ported to hop-by-hop recovery [68]. In this chapter, we analyze the problems incurred by the sequence based mechanism in hop-by-hop recovery and propose the receiver-centric protocol to overcome those problems.

We organize the chapter as follows. We first define the problem of reliable data transmission in Section 6.2. After that, we detail the lost packet recovery of the receiver-centric protocol in Section 6.3. We evaluate the receiver-centric protocol in Section 6.4 and summarize the chapter in Section 6.5.

6.2 Problem definition

In general, sensed data is loss-tolerant because packets containing the reported data have few correlations among each other and meaningful information can be inferred from partially received packets. For better understanding of monitored events, it is more important to capture the total number of unique reports rather than to reliably deliver each individual packet. Nevertheless, it is necessary to retransmit lost packets in unreliable wireless channels to maximize the channel throughput and improve energy efficiency. We use a simple numerical analysis to illustrate this point. Assuming that the packet loss rate of a wireless channel is ε. The packet success rate after n-hop forwarding will be $(1-\varepsilon)^n$. We plot the packet success rate under different ε and n in Figure 6.1, which shows that more than 40% packet will be lost after 6−hop forwarding when channel loss rate $\varepsilon = 0.1$. This can seriously degrade channel throughput and waste transmission energy since many packets are lost in the middle of forwarding.

The time-out mechanism has been proposed to retransmit lost packets, in which a sender will wait for an acknowledgement after it sends out a packet to a receiver. If the sender does not receive the acknowledgement from the receiver within a certain period, it will retransmit the packet. The lost packet may be retransmitted multiple times until reaching a certain threshold. The time-out mechanism is simple and can be easily implemented. However, it incurs several problems when applied to resource-constrained sensor networks.

1. the ACKs incur extra overhead that cannot be ignored in bandwidth-limited

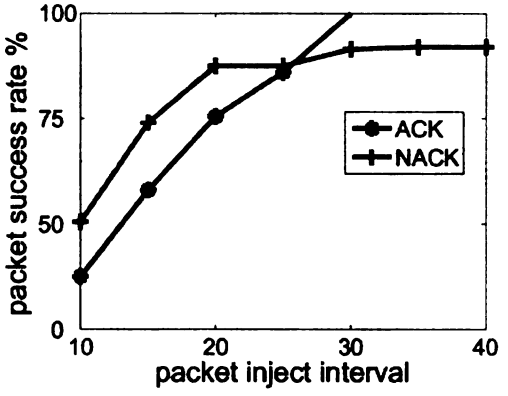


Figure 6.2 Packet success rate under different packet inject intervals

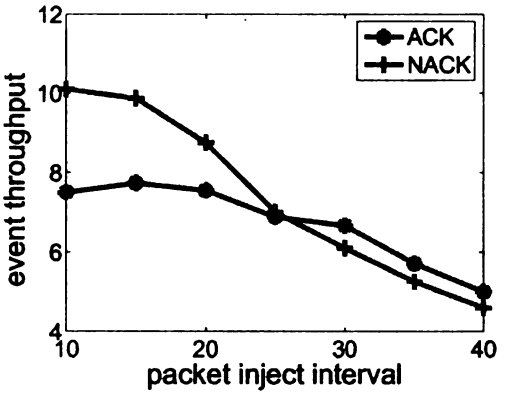


Figure 6.3 Event throughput under different packet inject intervals

wireless channels. In order to send back one bit information of an ACK, the receiver need to switch its receiving mode to transmitting mode, synchronize with the original sender by sending a serial of sync bytes, and finally transmit the ACK data. All this requires a minimum of $10-15$ bytes to just send the ACK. Since the data packets in sensor networks usually have small size (maximum 29 bytes in TinyOS 1.0 implementation), the extra overhead of ACK packets cannot be ignored.

2. the ACK itself my be lost due to the unreliable nature of wireless channels. This incurs unnecessary packet retransmission. The duplicated packets may be retransmitted in each forwarding and compete for bandwidth resources with data packets.

3. packets may be lost because of channel congestion, in which packets are corrupted in collisions or dropped by the overflowed buffer in the receiver. However, the time-out mechanism can not distinguish channel loss from channel conges-

tion and will blindly retransmit packets when ACKs are not received. This will intense channel congestion if all senders keep retransmitting lost packets.

To investigate how ACKs affect data transmission in sensor networks, we conduct experiments on a pair of MICA2 sensor motes. In our experiments, 200 packets are sent between a pair of sensors with different sending rates. In the first group of experiment, packets are transmitted with the time-out retransmission mechanism by enabling ACKs from the receiver to the sender. In the second group of experiment, we disable ACKs and do not use any retransmission mechanism. The comparisons are shown in Figure 6.2 and Figure 6.3, which illustrates that disabling ACKs can help sensors to achieve higher packet success rate and transmission throughput when packets are sending at small interval and therefore hight speed.

To retransmit lost packets while eliminating negative effect incurred by ACKs, we propose the receiver-centric protocol that can completely address the problem discussed above. First, the receiver-centric protocol is a sequence-based retransmission mechanism. It detect lost packets by checking the continuous received sequence number, which does not require to send ACKs. Second, the sequence numbers are notified to the sender implicitly by overhearing, which does not require extra messages. Third, the retransmission decision is decided by the receiver, which can easily detect channel congestion by checking its own buffer. Therefore, the retransmission will not mistakenly initiated due to channel congestion. Details of the receiver-centric protocol are discussed as the following section.

6.3 Reliable data transmission with the receiver-centric protocol

The receiver-centric protocol uses the sequence-based mechanism to detect and retransmit lost packets, which is similar to the sliding window mechanism used by the TCP protocol. To detect lost packets between a sender and a receiver, the sequence-based mechanism labeled all the packets from the sender with continuous sequences, and lost packets can be detected by the receiver if it receives packets with discontinued sequences. Based on discontinued sequences, the receiver can request the sender to retransmit the lost packets with missing sequences. The sequence-based mechanism, originally designed for the sliding window algorithm of the end-to-end TCP protocol, works between a pair of transceivers to ensure a highly reliable, strick in order packet transmission. Different from the TCP protocol, the receiver-centric protocol apply the sequence-based mechanism in hop-by-hop recovery to achieve high throughput and energy efficient data transmission from multiple sources to a sink. This difference leads to different design principles and implementation details, which we details as follows.

6.3.1 Streaming data transmission form sources to a sink

As we discussed the before, the receiver-centric protocol is designed to maximum throughput from sources to a sink. To receive a large volume of unique packets within a short period has higher priority than to ensure reliable transmission of individual packets. Based on this principle, we design the receiver-centric protocol to stream

packet transmission that will not be interrupted by the lost packet retransmission. We first show that continuous packet forwarding can be interrupted by the time-out mechanism and the sequence-based mechanism directly ported from the TCP protocol. After that, we discuss how the continuous data stream is maintained in the receiver-centric protocol.

In the time-out mechanism, a sender will wait for the ACK for a certain period after it sends a packet to a receiver. If the ACK is not received, the sender will resend the packets. This stop-and-wait mechanism reduce the channel transmission throughput. Moreover, ACKs may be lost, which incurs unnecessary retransmission. In this case, duplicated packets waste channel bandwidth resources.

The sequence-based mechanism can speed up packet forwarding because no ACKs are sent back from the receiver and the sender. All the network bandwidth can be used for data forwarding. However, the sequence based mechanism relies on strictly in-order sequence to detect lost packets, which may incur extra overhead and interrupt packet forwarding stream. As illustrated in Figure 6.4, when the source sends out packets to the sink through multihop forwarding, the source labels packets with continuously increased sequences. When node C receives packets 4 an 5 while lose packets 3, it will request node B to resend packets 3. At this moment, packets 5 and 6 have to be hold by node C and cannot be sent to node D. Otherwise, node D will also detect packet 3 is missing and request node C to retransmit. Here all the subsequent nodes have to wait until node C to recover the single lost packet 3. Therefore, the data forwarding streaming is interrupted by lost packet recovery. The situation may become worse if some packets cannot be recovered due to buffer over-

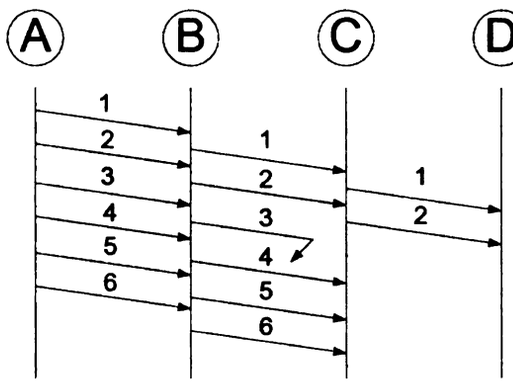


Figure 6.4 In TCP protocol, node B stops forwarding packet 4, 5, and 6 when packet 3 is lost

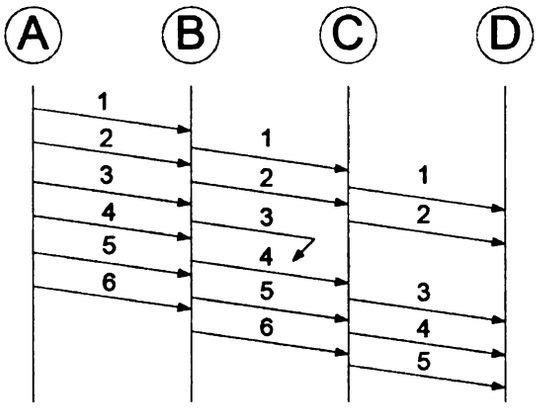


Figure 6.5 In receiver-centric protocol, node B continues to forward packet 4, 5, and 6 even when packet 3 is lost

flow. Those lost packets will always be detected by intermediate forwarding nodes and incur frequently lost requests for un-recoverable packets.

The sequence-based mechanism cannot maintain a continuous data forwarding stream because it relies on strictly continuous sequences globally maintained between the source and the sink. Therefore, an interruption happened to any intermediate node will stop data forwarding of the entire path. To solve this problem, we use the localized numbering mechanism to relabel each packets at each forwarding. In this mechanism, when a node receives packets, it will re-label packets with a continuously increased sequences maintained in the local variable. An example is shown in Figure 6.5, where each node independently maintain a local sequence number that is increased with received packets. Node C can detect lost packet 3 based on discontinued sequences. However, node C can still continue forwarding packets 4 and 5 to node D with no problems since all the packets are re-labeled by node C with new sequences. The packets forwarding between the pair of node B and C and the pair

of C and D is isolated by sequence renumbering. Therefore, packet interruption in one hop will not affect the continuous forwarding of the entire path.

6.3.2 Request lost packets with overhearing

To ensure continuous data packet forwarding, the receiver-centric protocol does not use dedicated messages to request lost packets from a sender. Instead, the lost sequences are piggybacked to data packets and the sender can be notified through overhearing. This mechanism uses the the broadcast nature of radio channels and the character of multihop forwarding, where packets forwarded by an intermediate node can always be overheard by its predecessor. Therefore, it can be used to signaling the predecessor with lost sequences. The consequence is that a virtual back-channel is created along the reverse path of data forwarding, in which a receiver can send lost sequences to its previous sender. The extra overhead is one more byte needs to attached to the data packet, which is more economical to use a dedicated message with more than 10 bytes length to notify the sender. More importantly, all the bandwidth can be dedicated to data packet forwarding when extra request messages are avoided.

6.3.3 Recover lost packets with O(1)time complexity

The receiver-centric protocol aims to provide high throughput of data forwarding by minimizing the overhead incurred by lost packet recovery. This is achieved by slightly modifying the queue management with an extra virtual head. In the receiver-centric protocol, each forwarding sensor maintains a buffer that operates as a normal queue, i.e. a received packet enters into the tail of the queue, and the packet at the head of

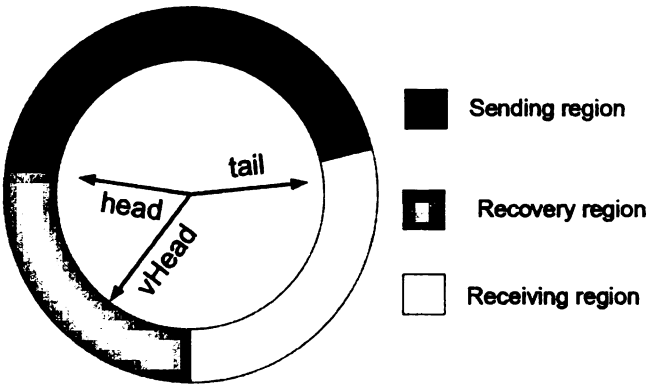


Figure 6.6 The receiver-centric protocol divides the packet buffers into three regions: the sending region, the recovery region, and the receiving region

the queue is sent out. In the receiver-centric protocol, the queue is divided into three regions: the sending region, the receiving region and the recovery region. As shown in Figure 6.6, where the sending region contains all the packets that wait to be sent, the recovery region contains all the sent packets, and the receiving region contains empty buffers that wait for new packets. Here, the sending region works as an normal queue which sends out packets at the *head* and receives packets at the *tail*. However, when a packet is sent out, it will be temporarily moved from the sending region to the recovery region, which might be used to recover lost packets as follows.

When the sender is notified by the receiver with missing sequences, the lost packets can be recovered from the recovery region. To achieve that, We use an extra pointer named *vHead* which can be temporarily pointed to the buffer containing the lost packets. When the lost packet is resent, the normal *head* will be continue used for packet forwarding. Here *vHead* is used to lookup and resend the lost packet. In the receiver-centric protocol, to look up lost packets in the recovery region can be finished in O(1) time without scan the entire region. This is achieved by renumbering

a packet based on the the index of the buffer containing that packet. Because the local sequence variable is increased simultaneously with the queue movement, i.e. the sequence number will be increased by 1 when a new packet enters into the queue, the sequence number s assigned to the new packet is correlated with the index i of the buffer containing that packet. We have $i = s\ mode N$, where N is the buffer size. Therefore, the index of a lost packet can be easily computed from its sequence with O(1) time.

6.3.4 Implement sequence-based recovery in TinyOS

We have implemented the sequence-based recovery mechanism in TinyOS 1.13. A simplified program of the sequence-based recovery mechanism is illustrated in Figure 6.7, which consists of three parts: the overhearing of lost sequences in *ReceiveMsg.receive*(), the lost packet retransmission in *QueueServicTask*(), and the close of the retransmission in *SendMsg.send*().

First, an intermediate node acquires the lost sequence through overhearing packets sent by the next hop. if the field *LostSeq* of the overheard packet contains non-null value, the node will set $bRecover == TRUE$ and $vHead = lostSeq\ mod\ MSG\_QUEUE\_SIZE$, i.e. point the $vHead$ to the index of the buffer contain the lost packet.

The Task QueueServicTask runs in the background, which repeatedly sends packets at the $Head$ of the queue ($sendingMsg = \&MsgBuf[Head]$). However, if $bRevoer$ is $TRUE$, which means the intermediate node receives a lost sequence, The *QueueServiceTask* will retransmit the packet pointed by the $vHead$

```
event TOS_MsgPtr
ReceiveMsg.receive[uint8_t id](TOS_MsgPtr pMsg){
  if( pMsg_data->src == parent
      && pMsg_data->lostSeq != NULL )
    vHead = pMsg_data->lostSeq % MSG_QUEUE_SIZE;
    bRecover = TRUE;
    return &tmpBuf;
  }
  return mForward(pMsg);
}

task void QueueServiceTask(){
  TOS_MsgPtr sendingMsg;
  {bool bRecover;
  if(bRecover){
    sendingMsg = & MsgBuf[vHead];
  else
    sendingMsg = & MsgBuf[Head];
  pMsg_data = (Msg_data *)sendingMsg->data;
  call SendMsg.send(sendingMsg->addr,
           sizeof(Msg_data), sendingMsg);
  return;
}

event result_t
SendMsg.sendDone(TOS_MsgPtr pMsg, bool success){
  if(pMsg == &MsgBuf[head]){
    head = (head + 1) % MSG_QUEUE_SIZE;
  }
  if(pMsg == &MsgBuf[vHead]){
    vHead = EMPTY;
    post QueueServiceTask();
    return SUCCESS;
  }
}
```

Figure 6.7 Sequence-based retransmission algorithm

```
typedef struct SF_CMD{
  uint8_t src;
  uint8_t CMD;
  bool bACK;
  uint8_t power;
  uint16_t injectRate;
  uint8_t num_children;
} SF_CMD;
```

Figure 6.8 Control message format

$(sendingMsg = \&MsgBuf[vHead])$.

When the intermediate node finishes sending a packet, it uses the $SendMsg.SendDone()$ to maintain the queue. If the sent packet is from the $Head$ of the queue, the $Head$ is moved to the next buffer unit. If the sent packet is from the $vHead$, which means the sent packet is retransmitted, both the $vHead$ and the $bRecover$ are cleared.

6.4 Performance evaluation

We implement the receiver-centric protocol in TinyOS and evaluate the performance of the receiver-centric protocol in both MICA2 and Tmote sensor motes. The receiver-centric protocol is evaluated by comparing with the *time-out retransmission enabled with acknowledgments (ACK)*, and the *best-effort mechanism without acknowledgments (NACK)*. We use the linear topology to evaluate the sequence-based retransmission mechanism. the following metrics are used in our evaluation.

- **event throughput**: the event throughput is defined as the total number of unique packets received at the receiver per second.

- **energy efficiency**: the energy efficiency is defined as the total number of bytes

of data divided by the total number bytes that are used to transmit the data.

- **packet inject interval**: the packet inject interval is the time period between two consecutive packet sending, which determines the sending speed of a sender.
- **buffer size**: the buffer size is length of transmission queue, i.e. is the maximum number of packets that a forwarding node can hold.
- **data size**: the data size is the number of bytes in a packet that is used to contain sensed data.

6.4.1 Experimental design and configuration

In order to evaluate the transmission throughput of different approaches, we have sources to keep sending packets, which can be forwarded by intermediate nodes and collected by the sink. The sink is connected to a laptop where the control terminal can **i**) control the experiment; **ii**) collect and analyze received packets. The control terminal is implemented with Java and communicates with the sink through the SerialForwarder. Our system consists of two parts, the management part and the data transmission part. In the management part, the control message is broadcast from the sink to the entire network, which initializes the network with desired configuration parameters including the radio transmission power, the buffer size, and the packet data size.

The sink also uses the control messages to trigger sources to start sending packets. The format of the control message is shown in Figure 6.8, which contains several fields including the packet inject rate, transmission power, ACK field, and packet

size. By setting those field with proper values, sources can be initiated with different parameters for each testing. It is possible that control messages may be lost during the broadcast, which results that **i)** some sources may not be triggered; **ii)** intermediate nodes may not be initiated with the proper transmission power and buffer size. We use two strategies to solve this problem. First, all the packets received by the sink are forwarded to the laptop through the serial cable. The packets contain the source ID that generates the packets. By checking all the received source IDs, the inactive sources that do not send packets can be identified and re-triggered. Since all the configuration parameters, such as the transmission power and buffer size, are included in the control messages, sources can be initialized with correct parameters as long as it is triggered by the control messages. Second, sources can pass the configuration parameters to intermediate nodes through data packets. When sources generating data packets, they will initialize packets with the transmission power and buffer size. Therefore, intermediate nodes can reset the configuration parameters with the same values as long as they receive packets from sources.

Since we only evaluate the communication performance of a sensor network at the link layer, the routing function at the network layer is not included in our program. Instead, we use fixed routing path in our test. This can be achieved by statically assigning routing table when sensor motes are reprogrammed. For the network topologies used in our evaluation, we only need to define the child/parent relationship in the routing table, such that a intermediate node can receive packets from its children and forward packets to its parent.

Assisted by the management part, we can control the testbed and evaluate dif-

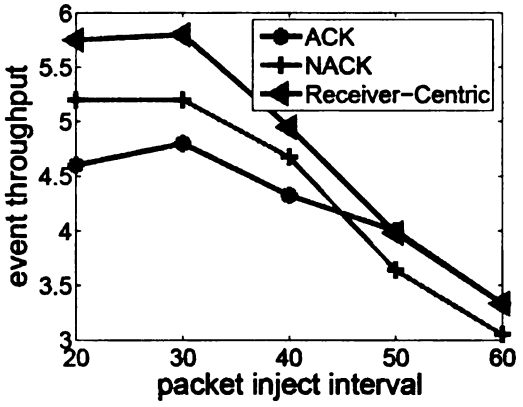


Figure 6.9 Lost packet recovery under packet inject intervals

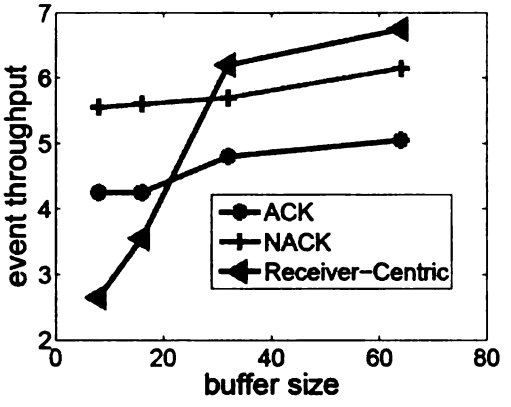


Figure 6.10 Lost packet recovery under different buffer sizes

ferent approaches in data transmission part. After we reprogrammed sensor motes with one approach, we conduct multiple tests with different settings of packet inject rate, buffer size and packet size. This is achieved by the control message broadcast from the sink. We detail our performance evaluation under various settings in the discussion below.

6.4.2 Lost packet recovery under different packet inject interval

We compar event throughput of the receiver-centric, the ACK, and the best-effort (NACK) approaches with a two hop network topology, where the sender continuously sends 200 packets to the receiver. All the packets received by the receiver are forwarded to the control terminal running in the laptop. The control terminal filters out duplicated packets based on the packet ID assigned by the sender. We evaluate the three approaches at different packet inject intervals. The comparison results are shown in Figure 6.9. The evaluation shows that the ACK mechanism has higher event throughput than the NACK approach under large packet inject interval. This

is because the ACk appproach can recover lost packets and therefore collect more packets at the receiver. However, when the packet inject interval becomes smaller, the ACK mechanism has lower event throughput than the best-effort mechanism. This is because of two reasons. First, when packets are sent at high speed with small inject interval, the ACK messages will compete bandwidth resources with data packets in the saturated wireless channel. Therefore, data packets will have smaller bandwidth and lower throughput. Second, ACKs may be lost due to the unreliable wireless channel. This makes the sender falsely believe that packets have been lost and unnecessarily retransmit duplicated packets. The duplicated packets consumes extra bandwidth resources and lower the throughput of data packets.

The evaluation above shows the inherent tradeoff between the benefit and overhead incurred by ACKs. We argue that it is necessary to maintain packet recovery at the link layer of a sensor network because **i)** Packets lost in the middle of a forwarding path wastes energy; **ii)** lost packets reduce event throughput. We therefore propose the receiver-centric protocol to recover lost packets with small overhead. Figure 6.9 shows that the receiver-centric protocol outperforms the ACK mechanism and the best-effort mechanism at both high packet inject rates and low packet inject rates. The receiver-centric protocol improves event throughput because **i)** it recovers lost packets and **ii)** packets are recovered at small overhead and only when necessary.

6.4.3 Lost packet recovery under different buffer sizes

We evaluate how the buffer size affect event throughput of the receiver-centric protocol. We fix the packet inject interval at 32 per millisecond, and vary the buffer

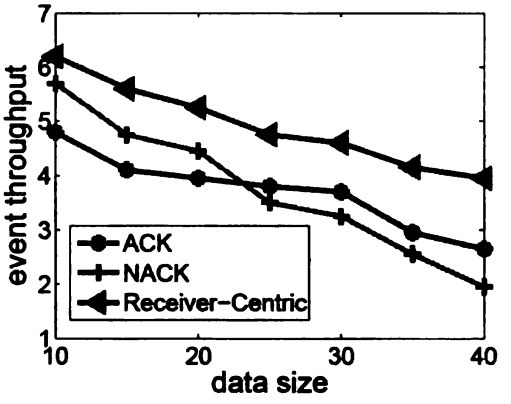


Figure 6.11 Lost packet recovery under different data sizes

size of all the forwarding nodes from 4 to 64. We test all the three approaches at different buffer sizes. The evaluation results of Figure 6.10 shows that the receiver-centric protocol performs worse when the buffer size is small, and outperforms other approaches when the buffer size is larger than 32. The receiver-centric protocol have better performance with larger buffer size because more buffer space can be allocated to the recovery region, which increase the possibility of recovering lost packets.

6.4.4 Lost packet recovery under different packet size

The event throughput can be affected by the packet size because a larger packet size consumes more bandwidth and is easier to be lost during the transmission. We evaluate how packet size affect the event throughput by varying the length of data field from 10 to 40 bytes. The default maximum data length is 29 bytes in TinyOS 1.0, we modify the system configuration to increase the maximum data length to 40 bytes. The comparison results of Figure 6.11 shows that the event throughput of all the three approaches are reduced when the packet size become larger. However, the NACk

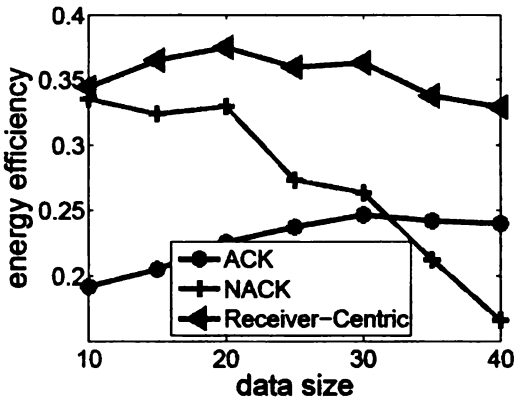


Figure 6.12 Energy efficiency under different data sizes

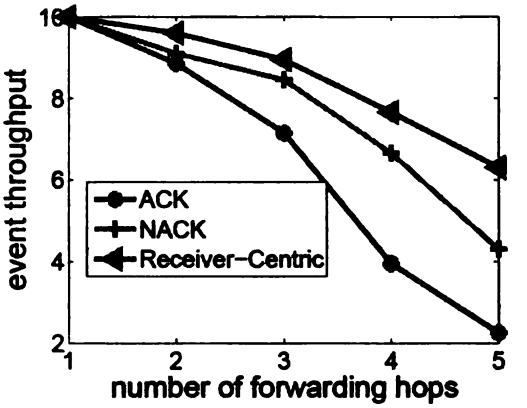


Figure 6.13 Lost packet recovery under different hops

approach has the lowest event throughput when data length is larger than 25 bytes. This because when packet size become larger, more packets will be lost during the transmission. These lot packets will not be received in the NACK approach because it has no recovery mechanisms. However, some of the lost packets can be recovered by the receiver-centric approach and the ACK approach, and therefore achieve higher event throughput. Among all the three approaches, the receiver-centric approach has the highest event throughput at different data lengths, because packets are recovered with with small overhead.

6.4.5 Energy efficiency of lost packet recovery

We evaluate energy efficiency of the receiver-centric approach at different data sizes from 10 bytes to 40 bytes. The evaluation results of Figure 6.12 shows that both the energy efficiency of both the receiver-centric and the ACK approaches are slowly increased when the data size becomes larger. The packet with larger data size is more energy efficient, because each packet has the fixed length of packet head and

the ratio of data field is increased when it has larger length. However, more packets may be lost during the transmission when data size become larger, which reduces the volume of data received at the sink. This offset the benefit of increased data length. Figure 6.12 also shows that the energy efficiency of the NACK approach drops fast when the data size become larger. This is because lost packets can not be recovered in the NACK approach and reduce the number of received data packets at the sink.

6.4.6 Lost packet recovery under different number of hops

We evaluate how the length of forwarding paths affect the event throughput by varying the number of forwarding nodes from 1 to 5. Figure 6.13 shows that the event throughput of the three approaches are reduced when the forwarding path become longer. particularly, the event throughput of the ACK approach and the NACK approach are severely affected by the number of forwarding hops. The ACK approach can be severely affected by the number of forwarding hops, because channel collision in the same forwarding path is intensified when the path contains more hops. The NACK approach has much lower event throughput when the path contains more forwarding hops, because more packets are lost in a longer forwarding path. In contrast, the receiver-centric approach outperforms the other two approaches because **i**) it recovers the lost packets and **ii**) the recovery process does not intensify in-path channel collision.

6.5 Summary

We propose the receiver-centric protocol to realize reliable data transmission with maximal throughput in sensor networks. Aiming at improving the event throughput instead of reliably transmitting each individual packets, the receiver centric approach optimizes the sensor network transport protocol as follows: i) it maintains the continuous data stream forwarding by minimizing the overhead incurred by control messages; ii) the virtual backward channel is created by utilizing the broadcast nature of wireless channels. We evaluate the effectiveness of the receiver-centric protocol through experimental comparison conducted in the MICA2 testbed, which shows that the receiver-centric protocol outperforms the hop-by-hop recovery and the best effort approach when events are reported with high sending rate through a densely deployed sensor network.

CHAPTER 7

Channel Access Scheduling in Wireless Sensor Networks

7.1 Motivation

It is often necessary to densely deploy sensors in complicated environments such that neighboring sensors can communicate with each other through short radio links, which are less susceptible to environmental interference. However, channel interference can easily happen among densely deployed sensors. Particularly, when huge volume of data need to be reported within a short time, channel interference among neighboring sensors can be intensified and become a serious problem. To solve this problem, several TDMA-based MAC protocols [11][12] have been proposed to assign time slots for channel access between neighboring sensors. These approaches either require a global view of an entire network topology to assign time slots [11] or incur extra message exchange within two-hop scopes [12]. Moreover, idling time slots reduces channel utilization. Because of their flexibility and scalability, major well-known MAC protocols like S-MAC [75], T-MAC [76], and B-MAC [77] adopt the contention-

based CSMA mechanism. However, channel contention may still happen in CSMA mechanisms when multiple neighboring nodes send data packet at high speed. In this chapter, we extend the receiver-centric protocol to reduce channel contention of CSMA mechanisms.

The receiver-centric protocol operates as an overlay of CSMA MAC layer and integrates the functions of both reliable data transmission and scheduling of channel access (Figure 7.1). What distinguish the receiver-centric protocol from other approaches is its fully cross layer design to maximize the system efficiency. The receiver-centric protocol utilizes the tree-based topology, the unique data transmission pattern presented in sensor networks, to assist packet retransmission and channel scheduling. The tree-based topology, naturally formed in sensed data collection, has the hierarchical structure and can be readily reused to manage data transmission. In the receiver-centric protocol, each intermediate node in the tree topology is viewed as a parent, which can receive data from multiple sources of children. The parent manages data retransmission and channel access of its children. By utilizing the tree-based structure of data collection that is unique in sensor networks, the receiver-centric protocol improve the performance of CSMA that is originally designed for general media access control.

We organize the paper as follows. We first define the problem of channel access scheduling in Section 7.2. After that, we detailed the channel access scheduling of the receiver-centric protocol in Section 7.3. We evaluate the receiver-centric protocol in Section 7.4 and summarize the chapter in Section 7.5.

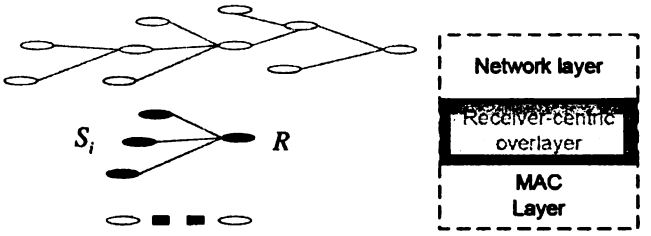


Figure 7.1 Receiver-centric protocol operates between the application layer and MAC layer, which utilizes the tree-based structure to schedule channel access

7.2 Problem definition

Data collection in wireless sensor networks always incurs multiple sensors to simultaneously forward data packets using radio channels of the same frequency. Multiple access control (MAC) is necessary to schedule neighboring sensors to use the shared channels. Because MAC protocols have critical impact on senor network performance, numerous approaches have been proposed that can be divided into three categories: CSMA, TDMA, and hybrid solutions of CSMA and TDMA.

Due to its simplicity, flexibility, and robustness, CSMA are widely adopted as major MAC protocols [75] [76] [77] in sensor networks. CSMA uses caring sensing multiple access mechanism to schedule neighboring sensors to access shared channels. Each senor listens to a channel and only sends out packets when it is idling. It is possible that two sensors listen to the same idling channel and send out packets simultaneously, which incurs collision and corrupt packets. To solve this problem, the back-off mechanism is proposed in which a sender may back-off random time before it resend a packet. A sender may keep resending a packets multiple times when a group of senders try to send out packets at a high transmission rate. This is the case when

large volume of bursting data need to be collected to a sink within a short period, which incurs intensive collisions and reduces network throughput.

To reduce channel collision in sensor networks, several approaches have been proposed to use TDMA to schedule channel access. By carefully assigning time slots to neighboring sensors that compete for the same channel, collision can be avoided because each sensor access the channel at different time slots. However, extra overhead is incurred by the TDMA mechanism to reduce channel collisions. First, it is not a trivial problem to assign time slots in a sensor network such that sensors will not interfere with each other during their transmission. It is NP-hard to find an assignment that is completely interference-free. Even for a close-optimal solution, either a global view of an entire network topology is required, or volume of message exchanges within two hop scopes are necessitated. Second, TDMA requires strict time synchronization that incurs frequent signaling, and timing drift may require a sensor network to re-assign time slots. Third, time slots are statically assigned to sensors in TDMA such that it is not flexible to topology changing, channel varying, and different data transmission patterns. In this case, collisions may happen due to changed network topology or channel conditions, or channel may be wasted due to the low utilization of idling channels.

To combine the advantages of CSMA and TDMA, hybrid solutions have been proposed. Z-MAC suggests to use CSMA in low data transmission rate and switch to TDMA when data transmission rate is high. Funneling-MAC proposes to apply TDMA in sensors that are deployed close to the sink and use CSMA in rest sensors that are far away from the sink. Funneling-MAC retains the flexibility and easy

deployment of CSMA while focuses on solving channel collision in the area that is close to the sink.

The receiver-centric approach is different from all the approaches above in that we are not replace the CSMA but wrap the CSMA with an overlay. The receiver-centric protocol works between the application layer and the MAC layer and realizes the application-aware channel access scheduling, i.e. the tree-based transmission pattern is incorporated into the channel access scheduling. By scheduling data transmission in the overlay, the collision in the MAC layer can be reduced. The receiver-centric protocol is efficient because its cross-layer design effectively utilizes data transmission in the application layer for channel access in the MAC layer. The receiver-centric protocol is flexible and easy to be deployed because it is a localized algorithm that does not require global view of a network topology or massive signaling. We detailed the design, implementation, and evaluation of the receiver-centric protocol in the following sections.

7.3 Channel access scheduling with the receiver-centric protocol

To maximize the throughput of data transmission, we further reduce channel collisions by enforcing channel access scheduling to CSMA. Operating as an overlay, the receiver-centric protocol bridges the application layer and the MAC layer, such that channel access can be scheduled in MAC layer according to the data flow of the application layer. We detail the channel access scheduling of the receiver-centric protocol

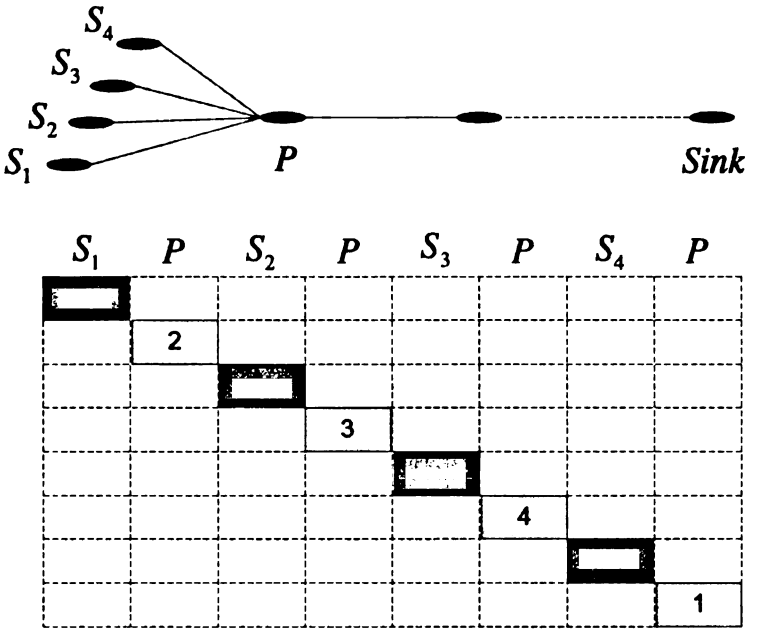


Figure 7.2 Receiver-centric protocol schedules time slots to different children through overhearing

as follows.

7.3.1 Basic idea

Instead of replacing CSMA, the receiver-centric protocol cooperates with CSMA to achieve channel access scheduling between neighboring sensors. As shown in Figure 7.1, the receiver-centric protocol operates as an overlay on CSMA. In this architecture, sensor still uses CSMA to access shared wireless channels. However, channel collisions can be effectively reduced since we reinforce the access scheduling in the receiver-centric overlay. Besides, this scheduling is efficient because it utilizes the unique data flow pattern presented in the tree-based sensor network topology. The receiver-centric protocol also minimizes the overhead of scheduling by reuse the data packet and does no incur extra signaling messages.

When collecting data from sources to a sink, the tree based topology will be naturally formed, where an intermediate node acts as a parent to receive packets from multiple children. Here, the parent-children structure is the basic unit of the tree-based network topology. We aim to realize channel access scheduling within this unit. As shown in Figure 7.2, where the parent receives packets from multiple sources of children and forwards packets towards the sink. Here, the parent is in the centralized position which can be utilized to schedule packet sending of its children. Moreover, this scheduling can reuse data packets through overhearing and does not incur extra signaling messages.

7.3.2 Channel access scheduling through overhearing

In the receiver-centric protocol, a delay δ is inserted into each packet sending, i.e. a sender will hold for a δ time period before it sends out the next packet. During this period, the sender waits for the scheduling signals from the parent. This is achieved through overhearing. When the parent forwards a packet to the next hop, one byte field is attached to the packet for scheduling. The parent set this scheduling field with the node ID of one of its children, which can be overheard by all its children. When a child finds that the overheard scheduling ID is equal to its own ID, the child will start to send a packet immediately after the parent finishes sending a packet. We use the round-robin scheduling to insure fairness among children, i.e. when the parent forwards a packet from child i, it will attach node ID $i + 1$ to the scheduling field, which prompts node $i + 1$ to send a packet after the parent finishes forwarding the packet from node i. The process of channel scheduling can be summarized as holding

and prompting. To reduce channel collision, all the children hold for a time period before sending their next packets, and one of children is prompted to start its packet when it overhears the scheduling ID from the parent.

Figure 7.2 illustrates the ideal case of the round-robin scheduling realized through the overhearing. When parent P forwards packet from child 1, the node ID 2 is attached to that packet, which can be overheard by all the children. Only node 2 that matches the scheduling ID will start to send its packet. The process is continued without incurring channel collisions.

7.3.3 Robust and flexible design

The example above shows an ideal case where all the nodes are perfectly scheduled by the receiver-centric protocol. However, real situation incurs more complicated cases, which includes channel collisions between adjacent units and channel idling within a scheduling unit.

Because the receiver-centric protocol only schedules channel access within the unit that contains the parent and all its children, channel collisions may still happen between adjacent units. We let this part of collisions to be handled by the underlying CSMA. The receiver-centric protocol cannot complete eliminate channel collisions in a sensor network. Instead, we seek to partially mitigate channel collisions with a light-weight solution that can be readily realized.

Because the receiver-centric protocol cooperates with the CSMA to realize the channel access, the communication channel can be efficiently utilized in a flexible fashion. A network topology may change due to node leaving or joining. These node

```
event TOS_MsgPtr
ReceiveMsg.receive[uint8_t id](TOS_MsgPtr pMsg){
  if(pMsg_data->src == parent &&
    pMsg_data->scheduleID == TOS_LOCAL_ADDRESS ){
    bSchedule = TRUE;
  }
}

event result_t
SendMsg.sendDone(TOS_MsgPtr pMsg, bool success){
  if (pMsg == &MsgBuf[head])
    head = (head + 1) % MSG_QUEUE_SIZE;
  if (pMsg == &MsgBuf[vHead])
    vHead = EMPTY;
  send_wait = 0;
  bSchedule = FALSE;
  call Timer_send.start(TIMER_ONE_SHOT, DELAY_SLOT);
  return SUCCESS;
}

event result_t Timer_send.fired(){
  if( send_wait < send_delay
    && bSchedule == FALSE ){
    send_wait = send_wait + DELAY_SLOT;
    return call Timer_send.start(TIMER_ONE_SHOT,
                                 DELAY_SLOT);
  }else
    post QueueServiceTask();
}
```

Figure 7.3 Channel access scheduling in the receiver-centric algorithm

activities can only temporarily affect the receiver-centric protocol and has little effect on its final performance. It is possible that node i suddenly crashes in the scheduling process. However, this will not stop the scheduling process because node $i+1$ will start to send packet after δ delay. Here packet sending of node $i+1$ is not triggered by the scheduling and some other child may also start to send packets, which results in channel collisions. We resort to CSMA to solve these collisions. After one of the child gain the channel access, the scheduling of the receiver-centric protocol will be resumed by attaching scheduling field with the node ID next to that child.

7.3.4 Implement the channel access scheduling in TinyOS

We implement the channel access scheduling in TinyOS 1.13. The simplified program of channel access scheduling is illustrated in Figure 7.3, which consists of three part: the scheduling part in function *ReceiveMsg.receive*(), the delay part in *SendMsg.SendDone*(), and the timer *Timer_send.fired*(). To enforce the channel access scheduling in CSMA, we insert a timer to the function SendMsg.SendDone(). When a packet is sent out, instead of immediately sending the next packet, the *QueueServiceTask* is delayed by the timer *Timer_send*. The *QueueServiceTask* can be started when **i**) the maximum delay has reached ($send\_wait < send\_delay$) or **ii**) the intermediate node is scheduled by its parent ($bSchedule == TRUE$). Here, the *bSchedule* can be set to $TRUE$ if the node overhears the $scheduleID$ is equal to itself ($scheduleID == TOS\_LOCAL\_ADDRESS$), as shown in function *ReceiveMsg.receive*().

7.4 Performance evaluation

We implement the receiver-centric protocol in TinyOS and evaluate the performance of the receiver-centric protocol in both MICA2 and Tmote sensor motes. The similar testbed, configuration, and metrics as the previous chapter is used to evaluate the channel access scheduling of the receiver-centric protocol. Particularly, we use the star topology and tree topology to evaluate the channel access scheduling mechanism.

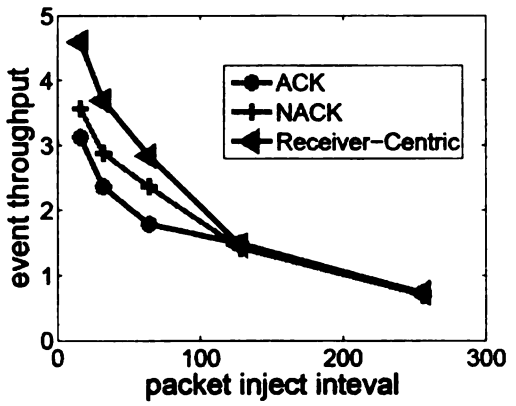


Figure 7.4 Event throughput of channel access scheduling under different inject intervals

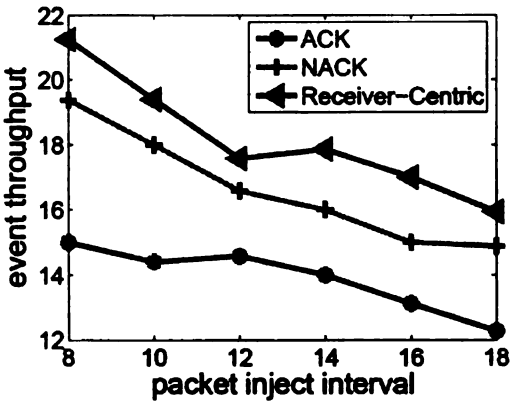


Figure 7.5 Event throughput of Tmote channel access scheduling under different inject intervals

7.4.1 Event throughput of channel access scheduling

We evaluate the event throughput of channel access scheduling as follows. We have two sources to send packets to a parent, which forwards received packets to the sink. Three approaches, including the receiver-centric with channel access scheduling, the ACK with CSMA, and the NACK with CSMA are evaluated in this test. We fist evaluate how the packet inject rate affect the event throughput by varying the packet inject interval from 16 millisecond to 256 millisecond. Figure 7.4 and Figure 7.5 show that when sources send out packet with small inject interval, the receiver-centric protocol outperforms both the ACK mechanism and the NACK mechanism with CSMA. Here, by enforcing the scheduling over CSMA, the receiver centric approach reduces the channel collision and improve event throughput.

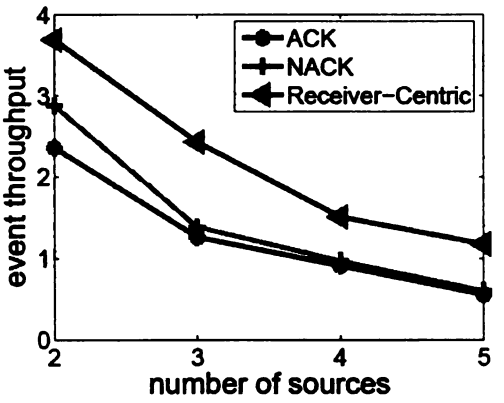


Figure 7.6 Channel access scheduling under different number of sources

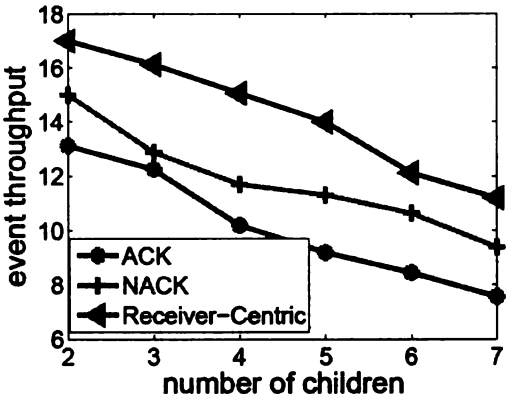


Figure 7.7 Tmote channel access scheduling under different number of sources

7.4.2 Channel access scheduling under different number of sources

We further evaluate the scheduling mechanism under different number of sources. We vary the number of sources from 2 to 5 and test the event throughput at the sink. Figure 7.6 and Figure 7.7 show that the event throughput is reduced with the increased number of sources. This because **i)** the bandwidth is divided and less bandwidth can be reserved by an individual source and **ii)** increasing the number of sources intensify the channel collisions, which results that more packets are lost during the transmission. Figure 7.6 and Figure 7.7 also show that the receiver-centric approach outperform the other two approaches at different number sources, because the scheduling mechanism reduces the channel collisions.

7.4.3 Channel access scheduling under different buffer sizes

We evaluate how the buffer size affect media access scheduling on MICA2 sensors by varying the buffer size from 16 to 64. Figure 7.8 shows that the event throughput of all the three approaches are increased when the buffer size is increased. The three

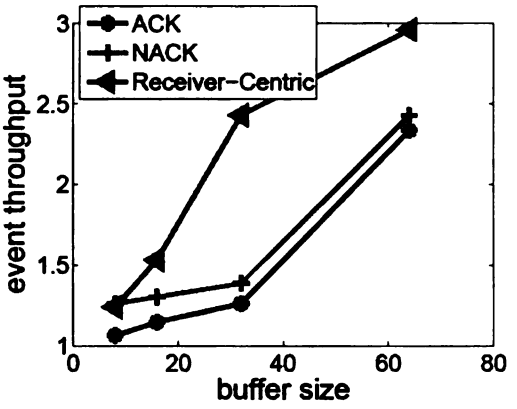


Figure 7.8 Event throughput of channel access scheduling under different buffer sizes

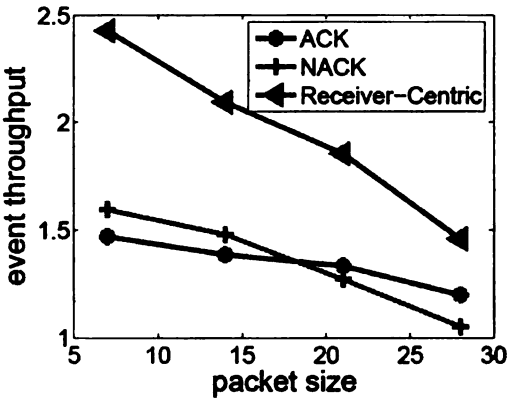


Figure 7.9 Event throughput of channel access scheduling under different packet sizes

approaches have close event throughput when the buffer size is small. However, the receiver-centric approach outperforms the other two approaches when the buffer become larger. All the three approaches have close and worst performance with small buffer size, because large number packets are dropped due to buffer overflow. When sensors have sufficient buffer size, they can achieve higher event throughput. Particip-ially, this event throughput can be further improved by channel access scheduling of the receiver-centric approach.

7.4.4 Channel access scheduling under different packet size

The effect of packet size on event throughput is shown in Figure 7.9. The event throughput of all the three approaches is decreased when the packet size is increased. This is because under the fixed network bandwidth, less number of packets can be transmitted within a certain period if the packet size becomes larger. Our test shows that the receiver-centric approach outperforms the other two approaches at different

packet sizes. The test also shows that the event throughput of NACK is higher than that of ACK when the packet size is small, while lower than that of ACK when the packet size becomes larger. This is because larger size packets are more easily lost in transmission, which significantly reduce event throughput of the NACK approach that has no recovery mechanisms.

7.4.5 Channel access scheduling in the tree topology

We evaluate the channel access scheduling in the tree topology consisting of 15 nodes. In this evaluation, all the leaf nodes work as sources and send packet to the root of sink. The channel access of leaf nodes is scheduled by their parents, which works as intermediate nodes and are also scheduled by upper layer parents. We vary the packet inject rate of sources from 8 millisecond to 20 millisecond and test the event throughput of the receiver-centric, the ACK, and the NACK approaches. The evaluation results in Figure 7.10 show that the receiver-centric approach outperforms the other two approaches at different packet inject intervals. Particularly, the channel access scheduling of the receiver-centric approach can improve 30% event throughput when packets are sent with small interval of 8 millisecond.

7.5 Summary

In this chapter, we aim to improve data transmission throughput in sensor networks by enforcing channel access scheduling to the CSMA. Our solution is flexible and scalable, and can be easily deployed in sensor networks. We have implemented the

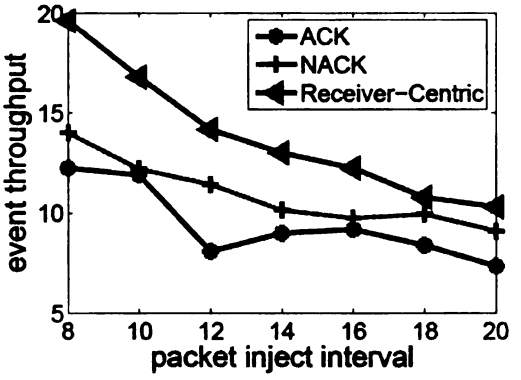


Figure 7.10 Scheduling in the tree topology

channel access scheduling in TinyOS 1.13 and evaluated their performance on both MICA2 sensors and Tmote sensors. Our evaluation shows that the channel access scheduling of receiver-centric protocol can significantly improve channel throughput of wireless sensor networks.

CHAPTER 8

Conclusion and Future Study

In this chapter, we conclude the work presented in this dissertation and discuss future study.

8.1 Conclusion

In the previous discussion, we demonstrate that wireless sensor networks will have irregular radio communication patterns or concave network topologies when deployed in complicated environments including obstructed or concave areas. We further demonstrate that system performance can be severely affected by obstructed and concave environments and special considerations are necessary to design wireless sensor networks for obstructed and concave environments. To build robust wireless sensor networks that can be adapted to obstructed and concave environments, we have developed a suit of approaches in sensor localization, packet routing, reliable data transmission, and channel access scheduling. We summarize our work as below.

Sensor localization provides basic service to many location aware applications and protocols. While many approaches have been proposed to locate sensors with lim-

ited available resources, few of them address the problem that distance measurements may have large errors when sensors are deployed into complicated environments. We try to accurately locate sensors that are deployed in complicated environments, where distance measurements have large errors because the radio or ultrasound signals may be reflected by obstructions. We have developed a group of algorithms to identify distance measurements with large errors, and to compute sensors' positions only based on correct measurements.

We first propose the virtual ruler approach that uses mobile beacons to eliminate erroneous distances in the measurement step. As discussed in Chapter 3, when the virtual ruler moves around in the deployed area, multiple distance measurements between the same pair of sensors can be obtained, among which erroneous measurements are mixed with the correct one. We show that it is possible to identify the correct distance measurement with statistical analysis. We use the small scale experiment to verify the distance mirror phenomenon that is utilized by the virtual ruler approach. After that, we evaluate the virtual ruler approach by the simulations with both indoor configurations and random configurations. We found that the virtual ruler approach can eliminate the majority of incorrect distance measurements when obstructions are not densely distributed.

We further propose the upper bound approach in Chapter 4 to eliminate erroneous distance measurements in the localization algorithms. The upper bound approach is based on the observation that the incorrect distance measurements inferred from radio signals or multihop approaches are always larger than its true value. Unlike previous approaches that use least squares fitting to fit all measured distances, the

upper bound approach views distance measurements as upper bound constraints, and locate sensors within the intersection area of all the circular constraints. The final localization result of the upper bound approach can be accurately pinpointed to the small areas confined by correct distance measurements and will not be affected by incorrect distance measurements with large errors. We conduct experiments to test the distance measurements with radio signals in an indoor basketball court. We further evaluate the upper bound approach in simulated environments with obstructions and concave shapes, which shows that the upper bound approach is effective to filter incorrect distance measurements and accurately locate sensors in obstructed and concave environments.

Packet routing is a challenging problem in wireless sensor networks because sensors only have limited memory space to store small routing states. Location aware routing has been widely suggested to realize point-to-point routing in sensor networks because it requires small routing states comprising sensors' positions. However, location aware routing cannot perform well in complicated environments because its ideal geographical model oversimplified the spatial complexity of wireless sensor networks. First, wireless sensor networks may have concave shapes where location aware routing may be trapped in the local minimum and fail to deliver a packet to a destination. To solve this problem, we propose the topology aware routing that embeds a sensor network into a Euclidean space such that the hop count distance are encoded to node virtual coordinates. Because hop count distances can be directly inferred from nodes' virtual coordinates, hop count distances to the destination can be accurately compared between neighboring nodes, which can assist the intermediate forwarding node

to find the right neighbor that is one hop closer to the destination. A packet can be finally delivered from the source to the destination through consecutive hop by hop forwarding. We evaluate the topology aware routing in various network topologies with concave shapes. Our evaluation shows that high route success rate can be achieved with small dimensional virtual coordinates.

Location aware routing also uses the simple disc model to describe the radio channel between neighboring nodes, i.e. a pair of nodes have perfect packet reception if they are within the radio transmission range. Based on this model, location aware routing aims to find the shortest path comprising the least number of forwarding hops from a source to a destination. However, in a realistic wireless sensor network, radio signals attenuate during their transmission and can be interfered by background noise, which leads to corrupted packets. a sender needs to retransmit the corrupted packets to ensure they are successfully delivered to the receiver. Packets may be retransmitted multiple times in a communication channel with weak radio signals. This often happens in the location aware routing that tends to select forwarding hops with long transmission range. The consequence is that excessive number of retransmissions are required to deliver a packet from the source to the destination along the routing path comprising low quality links. To achieve the optimal end-to-end routing performance, we further propose the ETX distance based greedy forwarding, which encodes the expected number of transmissions instead of the hop count distances to nodes virtual coordinates. With the help of the new embedding metric, the greedy forwarding can find the routing path that uses the least number of transmissions to successfully deliver a packet from the source to the destination. We evaluate our

ETX distance based greedy forwarding in TOSSIM emulator, which models the radio communication between pairwise nodes based on the empirical data measured from the MICA2 sensors. Our evaluation shows that our approach outperforms both the location aware routing and the shortest path routing in terms of the routing cost of total number of transmissions to deliver a packet from the source to the destination.

Due to the environmental interference, radio packets may be lost in transmission. Various retransmission mechanisms can be used to realize **reliable data transmission**. Most previous retransmission approaches fall into two categories: the sequence based mechanism and the time out mechanism. These two mechanisms are not optimal solutions for sensor networks when sensed data need to be transmitted within a short time at low cost, because i) they cannot ensure continuous data packet forwarding; or ii) the acknowledgement packets incur extra overhead. Aim to realize reliable data transmission with high throughput and low cost, we propose the receiver-centric protocol, which utilizes the broadcast nature of radio signals and the multiple hop forwarding of sensor networks. In the receiver centric protocol, lost packets are detected from discontinued sequence numbers, and the sender is notified by overhearing the lost sequences piggybacked to data packets. The receiver centric protocol ensures continuous data packet forwarding while incurs least overhead in lost packet retransmission. We evaluate the receiver centric protocol in an experimental testbed comprising MICA2 sensors, which shows that it can improve transmissions throughput while incurring minimal overheard.

To minimize interference of environment, sensors are usually densely deployed such neighboring nodes can communicate with each through short radio links. How-

ever, radio signals can be easily interfered with other other in a densely deployed sensor networks, which has severe impact on transmission throughput. CSMA instead of TDMA MAC protocols are widely used in practical sensor networks because the former can be easily deployed with low cost. However, channel contention can still happen in CSMA when bursting data need to be collected in a dense sensor network. We propose to improve the performance of CSMA by enforcing **channel access scheduling** on CSMA. We extend the receiver centric protocol to utilize the unique tree structure formed in sensed data collection. In a tree structure, multiple children report data to the same parent, where parent is in the central position and can be utilized to schedule channel access among multiple children. We implement the channel access scheduling in both MICA2 and Tmote sensors, and evaluate its performance in a real experimental testbed, which shows the transmission throughput can be significantly improved when compared to the CSMA protocol.

8.2 Future study

All our previous study aims to improve the performance of wireless sensor networks by centering around their geometric properties, which can only solve part of problems in building practical wireless sensor networks. The performance of wireless sensor networks can be improved from many other aspects, which we will explore in our future study. We list some possible directions as below.

Integrate with hierarchical design. Our previous study assumes that sensors can only communicate with neighbors within radio transmission range and form an ad

hoc network topology. This is a very basic form of wireless sensor networks. Recent development enables that certain powerful nodes called stargate can be deployed into sensor networks. Stargates nodes are more powerful than ordinary sensors and can communicate with each other with long range radio links. Stargate can be pointed as the head for a group of sensors that form a small ad hoc network. How to integrate the startgate with our proposed routing and MAC protocols to further improve the system performance will be explored in our future study.

Investigate temporal properties of wireless sensor networks. Most of our research are focused on the spatial properties of wireless sensor networks by exploiting their geometric characters. We assume that the network topology and the radio channels are relatively stationary. However, our experiments show that radio communication channels may change over time, which is difficult to model and can significantly affect experimental results. It is a challenging while practical task to design wireless sensor networks not only can adapt to complicated environment, but also can adapt to the changing environment. To realize this objective, intensive experiments need to be conducted to observe the temporal properties of wireless communications in sensor networks. Based on that, optimal design of wireless sensor networks can be achieved by incorporate those temporal properties.

Interconnect with other networks. Sensed data can be collected by various forms, and sensor networks can be interconnected with different type of networks. For example, sensor networks can be directly connected to Internet through a gateway, with which sensed data can be accessed by Internet users. Sensor networks can also be connected to vehicular networks and sensed data can be collected by mobile

vehicles. Sensor networks can also be interconnected to wireless mesh networks, where sensed data is collected through wireless mesh routers. Different interfaces need to be developed to interconnect sensor networks to different networks. It also necessary to develop effective algorithms to analyze and process the sensed data in the gateway and make them easily accessible by outside world.

Develop management platform for the testbed of wireless sensor networks. Our experiments show that it is a challenging task to manage volume of sensors to evaluate newly developed protocols and algorithms. In experiments, it is often necessary to have multiple sensors to start to send data simultaneously, and to stop the sending at the same time. It is also necessary to reconfigure and reprogram sensors with different protocols or algorithms while keeping the other evaluation configurations untouched. It will be time consuming or even impossible to manually realize these functions. We believe a testing platform with automated management functions can facilitate to evaluate new protocols and algorithms and further benefit future search on sensor networks.

BIBLIOGRAPHY

BIBLIOGRAPHY

[1] C. Wang and L. Xiao, "Locating sensors under limited measurement capabilities," *IEEE Network*, 2007.

[2] C. Wang, Y. Ding, and L. Xiao, "Virtual ruler: Mobile beacon based distance measurements for indoor sensor localization," in *MASS*, 2006.

[3] C. Wang and L. Xiao, "Locating sensors in concave areas," in *INFOCOM*, 2006.

[4] C. Wang, L. Xiao, and J. Rong, "Sensor localization in an obstructed environment," in *DCOSS*, 2005.

[5] C. Wang and L. Xiao, "Sensor localization in concave areas," *ACM Transactions on Sensor Networks*, vol. 4, iss. 1, Feb. 2008.

[6] P. Bose, P. Morin, I. Stojmenovic, and J. Urrutia, "Routing with guaranteed delivery in ad hoc wireless networks," in *ACM Int. Workshop on Discrete Algorithms and Methods for Mobile Computing and Communications*, 1999.

[7] B. Karp and H. T. Kung, "GPSR: Greedy Perimeter Stateless Routing for Wireless Networks," in *MobiCom*, 2000.

[8] R. Fonseca, S. Ratnasamy, J. Zhao, C. T. Ee, D. Culler, S. Shenker, and I. Stoica, "Beacon-vector routing: Scalable point-to-point routing in wireless sensor networks," in *NSDI*, 2005.

[9] C. Wang and L. Xiao, "Improving routing quality of greedy forwarding in wireless networks," in *IWQoS*, 2007.

[10] C. Wang, G. Zeng, and L. Xiao, "Optimizing end to end routing performance wireless sensor networks," in *DCOSS*, 2007.

[11] K. Arisha, M. Youssef, and M. Younis, "Energy-aware tdma based mac for sensor networks," in *IEEE Workshop on Integrated Management of Power Aware Communications Computing and Networking*, 2002.

[12] V. Rajendran, K. Obraczka, and J. J. Garcia-Luna-Aceves, "Energy-efficient collision-free medium access control for wireless sensor networks," in *SenSys*, (New York, NY, USA), pp. 181–192, ACM Press, 2003.

[13] D. Niculescu and B. Nath, "Ad hoc Positioning System (APS)," in *GLOBECOM*, 2001.

[14] T. He, C. Huang, B. M. Blum, J. A. Stankovic, and T. F. Abdelzaher, "Range-Free Localization Schemes in Large Scale Sensor Networks," in *MobiCom*, 2003.

[15] J. Albowicz, A. Chen, and L. Zhang, "Recursive Position Estimation in Sensor Networks," in *ICNP*, 2001.

[16] A. Savvides, C. Han, and M. B. Strivastava, "Dynamic fine-grained localization in ad-hoc networks of sensors," in *MobiCom*, 2001.

[17] D. Moore, J. Leonard, D. Rus, and S. Teller, "Robust Distributed Network Localization with Noisy Range Measurements," in *SenSys*, 2004.

[18] R. Nagpal, H. Shrobe, and J. Bachrach, "Organizing a Global Coordinate System from Local Information on an Ad Hoc Sensor Network," in *IPSN*, 2003.

[19] A. Savvides, H. Park, and M. B. Srivastava, "The n-hop multilateration primitive for node localization problems," in *Mobile Networks and Applications*, 2003.

[20] A. Savvides, H. Park, and M. B. Srivastava, "The bits and flops of the n-hop multilateration primitive for node localization problems," in *WSNA*, 2002.

[21] Y. Shang and W. Ruml, "Improved MDS-Based Localization," in *INFOCOM*, 2004.

[22] Y. Shang, W. Ruml, Y. Zhang, and M. P. J. Fromherz, "Localization from mere connectivity," in *MobiHoc*, 2003.

[23] X. Ji and H. Zha, "Sensor Positioning in Wireless Ad-hoc Sensor Networks with Multidimensional Scaling," in *INFOCOM*, 2004.

[24] L. Doherty, K. S. J. Pister, and L. E. Ghaoui, "Convex Position Estimation in Wireless Sensor Networks," in *INFOCOM*, 2001.

[25] P. Biswas and Y. Ye, "Semidefinite Programming for Ad Hoc Wireless Sensor Network Localization," in *IPSN*, 2004.

[26] Y. Shang, H. Shi, and A. Ahmed, "Performance tradeoffs of localization methods in ad-hoc sensor networks," in *MASS*, 2004.

[27] H. Lim and J. C. Hou, "Localization for anisotropic sensor networks," in *INFOCOM*, 2005.

[28] W. Zhao, M. Ammar, and E. Zegura, "A message ferrying approach for data delivery in sparse mobile ad hoc networks," in *MobiHoc*, 2004.

[29] M. M. B. Tariq, M. Ammar, and E. Zegura, "Message ferry route design for sparse ad hoc networks with mobile nodes," in *MobiHoc*, 2006.

[30] M. Sichitiu and V. Ramadurai, "Localization of wireless sensor networks with a mobile beacon," in *MASS*, 2004.

[31] N. B. Priyantha, H. Balakrishnan, E. Demaine, and S. Teller, "Mobile-assisted localization in wireless sensor networks," in *INFOCOM*, 2005.

[32] M. Kushwaha, K. Molnar, J. Sallai, P. Volgyesi, M. Maroti, and A. Ledeczi, "Sensor node localization using mobile acoustic beacons," in *MASS*, 2005.

[33] C. Perkins and P. Bhagwat, "Highly dynamic destination -sequenced distance-vector routing (DSDV) for mobile computers," in *SIGCOMM*, 1994.

[34] V. Park and M. Corson, "A highly adaptive distributed routing algorithm for mobile wireless networks," in *InfoCom*, 1997.

[35] D. B. Johnson and D. A. Maltz, "Dynamic source routing in ad-hoc wireless networks," *Mobile Computing*, 1996.

[36] C. Perkins and E. Royer, "Ad-hoc on-demand distance vector (AODV) routing," *Internet Draft*, 1998.

[37] K. Sundaresan and R. Sivakumar, "Routing in ad-hoc networks with MIMO links," in *ICNP*, 2005.

[38] J. Wu and F. Dai, "Broadcasting in ad hoc networks based on self-pruning," in *InfoCom*, 2003.

[39] Y. Ko and N. H. Vaidya, "Location-aided routing (LAR) in mobile ad hoc networks," in *MobiCom*, 1998.

[40] S. Wu and K. S. Candan, "Gper: Geographic power efficient routing in sensor networks," in *ICNP*, 2004.

[41] K. Seada, A. Helmy, and R. Govindan, "On the effect of localization errors on geographic face routing in sensor networks," in *IPSN*, 2004.

[42] F. Kuhn, R. Wattenhofer, Y. Zhang, and A. Zollinger, "Geometric adhoc routing: Of theory and practice," in *PODC*, 2003.

[43] S. Lee, B. Bhattacharjee, and S. Banerjee, "Efficient geographic routing in multihop wireless networks," in *MobiHoc*, 2005.

[44] X.-Y. Li, G. Calinescu, and P.-J. Wan, "Distributed construction of a planar spanner and routing for ad hoc wireless networks," in *INFOCOM*, 2003.

[45] G. Xing, C. Lu, R. Pless, and Q. Huang, "On greedy geographic routing algorithms in sensing covered networks," in *MobiHoc*, 2004.

[46] J. Bruck, J. Gao, and A. Jiang, "Localization and routing in sensor networks by local angle information," in *MobiHoc*, 2005.

[47] T. Melodia, D. Pompili, and I. Akyildiz, "On the interdependence of distributed topology control and geographical routing in ad hoc and sensor networks," *JSAC*, 2005.

[48] D. Kim and N. Maxemchuk, "Simple robotic routing in ad hoc networks," in *ICNP*, 2005.

[49] Q. Fang, J. Gao, and L. J. Guibas, "Locating and bypassing routing holes in sensor networks," in *InfoCom*, 2004.

[50] J. Newsome and D. Song, "Gem: Graph embedding for routing and data-centric storage in sensor networks without geographic information," in *SenSys*, 2003.

[51] A. Rao, S. Ratnasamy, C. Papadimitriou, S. Shenker, and I. Stoica, "Geographic routing without location information," in *MobiCom*, 2003.

[52] B. Leong, S. Mitra, and B. Liskov, "Path vector face routing: Geographic routing with local face information," in *ICNP*, 2005.

[53] M. Lim, A. Greenhalgh, J. ChesterField, and J. Crowcroft, "Landmark guided forwarding," in *ICNP 2005.*

[54] Y. Zhao, B. Li, Q. Zhang, Y. Chen, and W. Zhu, "Efficient hop id based routing for sparse ad hoc networks," in *ICNP*, 2005.

[55] Q. Cao and T. F. Abdelzaher, "A scalable logical coordinates framework for routing in wireless sensor networks," in *RTSS*, 2004.

[56] A. Caruso, S. Chessa, S. De, and A. Urpi, "GPS free coordinate assignment and routing in wireless sensor networks," in *InfoCom*, 2005.

[57] K. Seada, M. Zuniga, A. Helmy, and B. Krishnamachari, "Energy-efficient forwarding strategies for geographic routing in lossy wireless sensor networks," in *SenSys*, 2004.

[58] S. Wu and K. S. Candan, "GPER: Geographic power efficient routing in sensor networks," in *ICNP*, 2004.

[59] H. Zhang, A. Arora, and P. Sinha, "Learn on the fly: Data-driven link estimation and routing in sensor network backbones," in *InfoCom*, 2006.

[60] D. S. J. D. Couto, D. Aguayo, J. Bicket, and R. Morris, "A high-throughput path metric for multi-hop wireless routing," in *MobiCom*, 2003.

[61] A. Adya, P. Bahl, J. Padhye, A. Wolman, and L. Zhou, "A multi-radio unification protocol for ieee 802.11 wireless networks," in *Broadnets*, 2004.

[62] R. Draves, J. Padhye, and B. Zill, "Comparison of routing metrics for static multi-hop wireless networks," in *SIGCOMM*, 2004.

[63] A. Woo, T. Tong, and D. Culler, "Taming the underlying challenges of reliable multihop routing in sensor networks," in *SenSys*, 2003.

[64] J. Zhao and R. Govindan, "Understanding packet delivery performance in dense wireless sensor networks," in *SenSys*, 2003.

[65] T. S. E. Ng and H. Zhang, "Predicting internet network distance with coordinates-based approaches," in *InfoCom*, 2002.

[66] L. Tang and M. Crovella, "Virtual landmarks for the internet," in *IMC*, 2003.

[67] H. Lim, J. C. Hou, and C.-H. Choi, "Constructing internet coordinate system based on delay measurement," *IEEE/ACM Trasactions on Networking*, 2005.

[68] F. Stann and J. Heidemann, "RMST: Reliable data transport in sensor networks," in *SNPA*, 2003.

[69] H. Zhang, A. Arora, Y. ri Choi, and M. G. Gouda, "Reliable bursty convergecast in wireless sensor networks," in *MobiHoc*, 2005.

[70] C.-Y. Wan, A. T. Campbell, and L. Krishnamurthy, "PSFQ: a reliable transport protocol for wireless sensor networks," in *WSNA*, 2002.

[71] S.-J. Park, R. Vedantham, R. Sivakumar, and I. F. Akyildiz, "A scalable approach for reliable downstream data delivery in wireless sensor networks," in *MobiHoc*, 2004.

[72] Q. Cao, T. He, L. Fang, T. Abdelzaher, J. Stankovic, and S. Son, "Efficiency centric communication model for wireless sensor networks," in *INFOCOM*, 2006.

[73] I. Rhee, A. Warrier, M. Aia, and J. Min, "Z-MAC: a hybrid mac for wireless sensor networks," in *SenSys*, 2005.

[74] J. Li and G. Y. Lazarou, "A bit-map-assisted energy-efficient mac scheme for wireless sensor networks," in *IPSN*, (New York, NY, USA), pp. 55–60, ACM Press, 2004.

[75] W. Ye, J. Heidemann, and D. Estrin, "An energy-efficient mac protocol for wireless sensor networks," in *INFOCOM*, 2002.

[76] T. van Dam and K. Langendoen, "An adaptive energy-efficient mac protocol for wireless sensor networks," in *SenSys*, 2003.

[77] J. Polastre, J. Hill, and D. Culler, "Versatile low power media access for wireless sensor networks," in *SenSys*, 2004.

[78] G.-S. Ahn, S. G. Hong, E. Miluzzo, A. T. Campbell, and F. Cuomo, "Funneling-mac: a localized, sink-oriented mac for boosting fidelity in sensor networks," in *SenSys*, (New York, NY, USA), pp. 293–306, ACM Press, 2006.

[79] L. Girod and D. Estrin, "Robust Range Estimation Using Acoustic and Multimodal Sensing," in *IROS*, 2001.

[80] K. Whitehouse and D. Culler, "Calibration as parameter estimation in sensor networks," in *WSNA*, 2002.

[81] A. Smith, H. Balakrishnan, M. Goraczko, and N. Priyantha, "Tracking moving devices with the cricket location system," in *MobiSys*, 2004.

[82] J. Hill and D. Culler, "Mica: A wireless platform for deeply embedded networks," *IEEE Micro*, 2002.

[83] J. Hightower, R. Want, and G. Borriello, "SpotON: An Indoor 3D Location Sensing Technology Based on RF Signal Strength," *UW CSE technical report 00-02-02, University of Washington, Department of Computer Science and Engineering, Seattle, WA*, 2000.

[84] M. S. Lobo, L. Vandenberghe, S. Boyd, and H. Lebret, "Applications of Second-order Cone Programming," *Linear Algebra and its Application*, 1998.

[85] Crossbow, "MICA2 sensor datasheet," *Crossbow technology Inc. technical specification*, 2004.

[86] R. Want, A. Hopper, V. F. ao, and J. Gibbons, "The active badge location systemw," *ACM Transactions on Information Systems (TOIS)*, vol. 10, no. 1, 1992.

[87] N. Bulusu, J. Heidemann, and D. Estrin, "Adaptive Beacon Placement," in *ICDCS*, 2001.

[88] N. Li and J. C. Hou, "Flss: A fault-tolerant topology control algorithm for wireless networks," in *MobiCom*, 2004.

[89] X.-Y. Li, P.-J. Wan, Y. Wang, and C.-W. Yi, "Fault tolerant deployment and topology control for wireless ad hoc networks," in *MobiHoc*, 2003.

[90] C. Intanagonwiwat, R. Govindan, and D. Estrin, "Directed diffusion: A scalable and robust communication paradigm for sensor networks," in *MobiCom*, 2000.

[91] S. Madden, M. J. Franklin, J. M. Hellerstein, and W. Hong, "TAG: a tiny aggregation service for ad-hoc sensor networks," in *OSDI*, 2002.

[92] J. Heidemann, D. Estrin, R. Govindan, and S. Kumar, "Next century challenges: scalable coordination in sensor networks," in *MobiCom*, 1999.

[93] S. Shenker, S. Ratnasamy, B. Karp, R. Govindan, and D. Estrin, "Data-centric storage in sensornets," *SIGCOMM Comput. Commun. Rev.*, vol. 33, no. 1, pp. 137–142, 2003.

[94] C. T. Ee, S. Ratnasamy, and S. Shenker, "Practical data-centric storage," in *NSDI*, 2006.

[95] M. Li, D. Ganesan, and P. Shenoy, "PRESTO: Feedback-driven data management in sensor networks," in *NSDI*, 2006.

[96] S. Kapadia and B. Krishnamachari, "Comparative analysis of push-pull query strategies for wireless sensor networks," in *DCOSS*, 2006.

[97] C. Perkins and T. Watson, "Highly dynamic destination sequenced distance vector routing (DSDV) for mobile computers," in *SIGCOMM*, 1994.

[98] H. Frey and I. Stojmenovic, "On delivery guarantees of face and combined greedy-face routing in ad hoc and sensor networks," in *MobiCom*, (New York, NY, USA), pp. 390–401, ACM Press, 2006.

[99] Y.-J. Kim, R. Govindan, B. Karp, and S. Shenker, "Lazy cross-link removal for geographic routing," in *SenSys*, 2006.

[100] B. Leong, B. Liskov, and R. Morris, "Geographic routing without planarization," in *NSDI*, 2006.

[101] T. F. Cox and M. A. A. Cox, *Multidimensional Scaling.* Chapman and Hall, 1994.

[102] P. Levis, N. Lee, and M. W. D. Culler, "Tossim: Accurate and scalable simulation of entire tinyos applications," in *SenSys*, 2003.