Delft University of Technology
Master Thesis in Computer Science

GDE: A Distributed Gradient-Based Algorithm for Distance Estimation in Large-Scale Networks

Qingzhi Liu

embedded software

TU Delft Delft University of Technology

GDE: A Distributed Gradient-Based Algorithm for Distance Estimation in Large-Scale Networks

Embedded Software Section
Faculty of Electrical Engineering, Mathematics and Computer Science
Delft University of Technology
Mekelweg 4, 2628 CD Delft, The Netherlands

Qingzhi Liu
Q.Liu-1@student.tudelft.nl

15th August 2011

Author
Qingzhi Liu (Q.Liu-1@student.tudelft.nl)

Title
GDE: A Distributed Gradient-Based Algorithm
for Distance Estimation in Large-Scale Networks

MSc presentation
24th August 2011

Graduation Committee

Prof. Dr. Koen Langendoen (Chair)	Delft University of Technology
Dr. Stefan Dulman	Delft University of Technology
Dr. Alex Iosup	Delft University of Technology
Msc. Ir. Andrei Pruteanu	Delft University of Technology

Abstract

Today, wireless networks are connecting more and more devices around us. The scale of these systems demands for novel techniques to maintain availability for various services such as routing, localization, context detection, etc. Distance estimation is one of their most important building blocks. The majority of current algorithms presume knowledge about node position via systems such as GPS. While for some application scenarios this approach is feasible, for a lot of cases it suffers from frequent unavailability and high costs in terms of energy consumption. The main contribution of the thesis is the introduction of a novel distributed algorithm called *GDE* for the estimation of distances in large-scale wireless networks. It is based on a gossiping mechanism to estimate distances between nodes solely based on local interaction. We analyze the parameters that should be considered by real applications, and present mathematical models to compensate their influence for distance estimation. Three kinds of applications are shown in the thesis using the *GDE* algorithm, including cluster center detection, overlay shape construction, and routing. Finally, we introduce some more improvement methods for the *GDE* algorithm to increase the distance estimation accuracy. The evaluations by means of simulation show that *GDE* succeeds in estimating the distance between nodes in both static and mobile scenarios with considerably high accuracy for various parameter setups, such as varying node density, node speed, spatial node distribution, etc.

Preface

No matter how well prepared a person can be, life is always full of surprises and challenges. I found myself in many happy or difficult situations after I came to study at Delft University of Technology, the Netherlands. I sometimes felt hesitating and tired, but I always knew the dream is in the distance, the path is under my feet, and that this is the life I enjoy.

I am delighted that to have successfully accomplished my thesis. hat makes me happier is the guidance I received from my supervisor Dr. Stefan Dulman. I would like to express my sincere gratitude him. He gave me not only instructions for the thesis, but also the freedom to fully develop my ideas and research abilities. At the same time, he provided with me several inspiring thoughts at some important steps in my research, which are considerably valuable for the thesis.

I would like to extend my special thanks to Andrei Pruteanu, who gave me extensive help from the start to the finish of the work at my thesis. He always encouraged me to try my research ideas and at the same time, warned me that there could be problems on the way. He is patient to my words, no matter how bad my writing and English speaking are. Without his help, many ideas and experiments could of hardly been finished.

I would especially like to thank my parents, my father and mother back home in China. No matter how difficult the situations I was in, they always trusted and supported me. They made me confident to pursue on my life path.

I would also like to thank all my friends in the Netherlands. We had a lot of good times together.

Qingzhi Liu

Delft, The Netherlands
15th August 2011

Contents

Chapter 1

Introduction

Distance estimation is one of the most important building blocks for wireless sensor network applications, especially for the ones that need location information. This thesis presents a novel distributed algorithm called *GDE* for the estimation of distances in large-scale wireless networks.

1.1 Problem Statement

Nowadays sensors are more and more widely used due to advances in low-power micro-electronics and wireless network technology. They are used in many application domains, including environmental monitoring, smart buildings, target tracking, battlefield surveillance, etc. As the number and the diversity of sensors are both increasing substantially, wireless sensor networks are connecting more and more devices around us. However, along with ubiquity comes also the problem of tremendous scale. In recent years, the scientific research on large-scale sensor networks has increased significantly. The current approaches that enable services such as routing, data dissemination, context detection etc. require new, radical changes, in order to assure scalability. When location information is required, most of the current algorithms presume that node position information is available via systems such as GPS. Now, for some application scenarios, this approach is feasible (e.g., outdoors). Unfortunately, for most of the actual deployments, GPS is either not available (e.g., indoors) or it is inaccurate and most likely consumes a lot of power. At the same time, in large-scale networks the sensor nodes can not always be placed at desired locations to ensure for instance a given network connectivity. Most likely, nodes also exhibit some sort of mobility, which makes difficult to continuously update their position information. For all these reasons, there is a need for ways to estimate distances between nodes only based on local interaction with no additional information such as the one provided by GPS or classic localization techniques. The estimated distances can be used to compute location information, and thus enable many other applications on top.

1.2 Contributions

In this thesis, we introduce a *distributed gradient-based distance estimation algorithm* called *GDE*. It is designed for large-scale networks and uses simple, local interactions.

The main research questions we answer in the thesis are the following:

1. Is there an algorithm to accurately estimate distances in large-scale networks solely based on local interactions?
2. Can such an algorithm provide a good distance estimation even when nodes are mobile, the node density is not uniform and algorithm parameters are adaptive?
3. What are the possible application scenarios that can make use of such an algorithm?

GDE is based on a hop-count gradient that is smoothed to decreases the distance estimation error [17]. Additionally, we introduce two compensation methods for the error in order to improve the estimation accuracy. We further extend the original algorithm to compensate for the influence on the accuracy of the distance estimation algorithm by various network properties that occur in real application conditions. It includes varying the node density, the node speed, the spatial node distribution, the average multicast ratio, the communication round length, the shape of the transmission range, etc. We present a mathematical analysis as well as models for the influence of these parameters. The thesis is a theoretical study on the feasibility of such an approach for large-scale, dynamic systems. We validate our algorithms via simulations for various deployment setups, such as varying node density, different average node speed, realistic radio propagation models, etc.

The experimental results show that our algorithm performs well in terms of distance estimation accuracy. We use *GDE* for three application scenarios such as cluster center detection, overlay creation, and routing. According to the results, the applications using the *GDE* algorithm show much better results compared to the other distance estimation algorithms. Finally, we introduce some further improvements, and according to the preliminary tests, they can further improve the accuracy of the *GDE* algorithm.

1.3 Thesis Outline

The rest of the thesis is organized as following. Section 3 presents two existed distance estimation algorithms. Section 4 presents the basic *GDE* algorithm for a static, spatially uniform deployment. Section 5 presents the improved algorithm models for some realistic deployment parameters. An

in-depth experimental analysis by means of simulations of the algorithms for various parameter setups is presented in Section 6. Section 7 presents three kinds of applications where the *GDE* algorithm shows improved performance. Section 8 presents some further improvement methods for the accuracy of the *GDE* algorithm. We conclude the thesis and give insights about future work in Section 9.

Chapter 2

Related Work

Wireless sensor networks are used for a large number of applications, such as environmental monitoring [16, 32], structural monitoring [31], target tracking [3, 21], surveillance missions [8], etc. Distance estimation for wireless systems is one of the most important building blocks for various services such as routing [26], clustering [33], localization [8, 27] etc.

For the case of geographic routing [11], nodes use locations as their addresses, and forward packets (when possible) in a greedy manner towards the destination. However, this approach demands for the nodes to be aware of their location. While this is a feasible assumption in some settings (e.g., nodes have GPS [9] modules attached), there are many cases where node location information is either not available or it requires additional expensive equipments, such as indoors, underground, underwater, as so on [19]. For many clustering algorithms [6] that require node position information, again all nodes are assumed to be aware of their location. To optimize for power, the GPS module has to be turned on for a very short time during bootstrapping. This approach does make sense when the nodes are either static or there are small changes of the network topology. When nodes are always moving for example, the localization system has to be turned on continuously, thus leading to a high power consumption. Three main kinds of localization methods are discussed in [22], which are based on time of arrival (TOA), angle of arrival (AOA) or received signal strength (RSS) respectively. Some localization algorithms that need to compute the pair-wise distance between nodes are not suitable for large scale networks. There is a lot of work on the indoor localization [14, 24]. Besides GPS, other methods suitable in particular for indoor deployments, make use of the wireless signal strength [1]. The main problem that the class of algorithms is facing is the lack of accurate models for indoor radio propagation. Problems are caused by various sources such as multi-path effects, reflection, and fading effects. Most of the time, there is limited correlation between the received signal strength indication and the real distance between devices. StarDust

system [29] uses light to localize nodes in a centralized approach. The nodes use reflectors to estimate their positions. A localization algorithm called moving baseline localization (MBL) makes localization computation by estimating the node trajectories [20]. MBL algorithm needs some complicated information about the neighbor nodes, such as velocity.

Our *GDE* algorithm uses the gossiping-based protocol [2,4] as the communication approach. Gossiping-based communication strategy has many advantages [5], including inherent scalability, load-balancing, resilience to node and network failures, easiness of management, etc. Compared to flooding, gossiping-based protocols require each node to communicate and exchange data with one or a few neighboring nodes. Gossiping-based protocols can efficiently decrease the message complexity in both static and dynamic networks [12]. Thus they are successfully used in many applications, such as routing [7].

The algorithm [17] serves as a starting point for our work and builds a self-organized, global coordinate system on top of an ad-hoc wireless sensor network. It relies only on distributed, simple computations and local communication. The algorithm adapts to failures or additions of nodes. A remarkable aspect uncovered by the work is that greedy routing performs better when using virtual coordinates than when using the actual geographic coordinates in the presence of obstacles.

Chapter 3

Gradient Algorithms

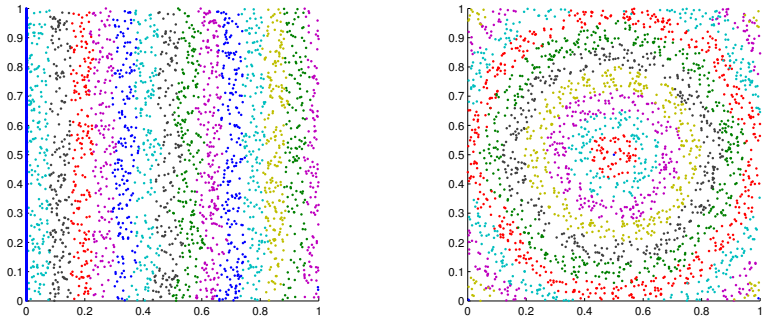
There are two existing distance estimation algorithms based on the gradients. In this chapter we briefly introduce the related work. The two existing algorithms are the basis of our *GDE* algorithm.

3.1 Hop-Count Gradient

Suppose all nodes are static and uniformly distributed in a deployment area, and the transmission radius of each node is r. The node density is defined as the average number of nodes within the transmission range of each node. Some nodes, called *seeds*, are placed at fixed positions, and start with a constant counter value. Each node, except the seeds, listens to its neighbors and selects the largest counter value, and then rewrites the local counter value decreased by one. The counter values of all nodes in the deployment area vary uniformly: the nodes closer to the seed nodes have larger counter values, and the nodes that are further away from the seed nodes have smaller counter values. We refer to this variation trend as a *gradient*. Figure 3.1(a) shows the resulting gradient when a straight line of seed nodes are vertically placed at the left side of the deployment area, and Figure 3.1(b) shows the resulting gradient when a seed node is placed at the center of the deployment area. In the figures, each node is colored according to its own counter value.

One can notice in Figure 3.1(a) and Figure 3.1(b) that the obtained gradient is not smoothed but looks rather like stairs. This is because each node computes the local counter value using only *the largest counter value* in its own transmission range, so based on this strategy, some adjacent nodes will have the same local counter values. We call the nodes with the same counter values a *wave* in the gradient. Because the waves depend on the minimum number of hops from the seeds, we call this gradient the *hop-count gradient* (HCG).

To create the hop-count gradient, the counter values should be disseminated in the network. Push and Pull algorithms can be used for data dis-



(a) (b)

Figure 3.1: (a) Hop-count gradient with a straight line of seed nodes at the left side of the square. (b) Hop-count gradient with a seed node at the center of the square.

Algorithm 1: Create Hop-Count Gradient with Push

```
1: counter                                          ▷ received counter
2: localCounter                                     ▷ local counter
3: {neighbors}                                      ▷ neighbor set

4: receive counter
5: if counter > localCounter
6:     set localCounter counter
7: end if
8: send (localCounter - 1) to {neighbors}
```

semination via gossiping [10]. Push proactively sends data into the network, while Pull reactively sends queries first into the network and then receives data from the network. There are two methods to disseminate the counter values and create the hop-count gradient.

The first approach is to use the Push method. The seed nodes start as the first gradient wave and send to the neighbors messages with a constant gradient counter value. Each recipient stores the counter value and sends messages to the neighbors with the counter decremented by one. In order to prevent the backward gradient waves, each node only stores the maximum counter value received and ignores the lower counter values. Algorithm 1 describes the main process of creating the hop-count gradient using Push method.

The second approach is to use the Pull method. The seed nodes initiate as the first gradient wave with a constant gradient counter value. All nodes except the seeds listen to the neighbor area and selects the largest gradient counter value, then rewrites the local counter value with the maximum

Algorithm 2: Create Hop-Count Gradient with Pull

1: *counter* ▷ received counter
2: *localCounter* ▷ local counter
3: {*neighbors*} ▷ neighbor set

4: **ask** *counter* **from** {*neighbors*}
5: **set** *localCounter* (**max**{**all** *counter*} - *1*)

received counter decreased by one. Compared with the Push method, Pull method does not need to select the received counter values in order to prevent the backward broadcasting. Algorithm 2 describes the main process of creating the hop-count gradient using the Pull method.

3.2 Smoothed Gradient

The distance estimation produced by the hop-count gradient is a multiple of the width of the gradient wave ω. It produces an average error of approximately 0.5ω [17]. To improve the accuracy, the local neighborhood gradient values can be used to compute the *smoothed gradient* (SMG). This section introduces the *SMG* algorithm. Based on the original *SMG* algorithm, we also present the computation of the smoothed gradient value of the node placed at the border of the gradient area.

3.2.1 Regular Nodes

Suppose the transmission radius of each node is r. To simplify the computation of the smoothed gradient, the communication area of every node is approximated to a square with side length $2r$. Figure 3.2 shows the transmission area of node i. By a we denote the integer hop-count gradient value of node i. α, β and γ are the ratios of three areas of various hop-count gradient waves as seen by node i. Because the nodes are uniformly distributed in the area, α, β and γ are supposed to be equal to the rates between the number of nodes in the adjacent three areas to the total number of nodes in the transmission range. Suppose the width of each wave ω equals the transmission range r of a node. In order to obtain the gradient value of the node, we normalize the length of the gradient wave to 1. The gradient distance of node i to the seed nodes is computed using Formulas 3.1, 3.2 and 3.3. P_α and P_β are the computed distance from node i to the line of seed nodes using parameter α and β respectively. G_α and G_β are the gradient values. $P_{\alpha\beta}$ is the average value of P_α and P_β, and $G_{\alpha\beta}$ is the gradient value.

$$\left.\begin{array}{c} P_\alpha = (a-1)r + x \\ x = r - 2r\alpha \end{array}\right\} \stackrel{r=1}{\Rightarrow} G_\alpha = P_\alpha = a - 2\alpha \tag{3.1}$$

$$\left.\begin{array}{l} P_\beta = ar - y \\ y = r - 2r\beta \end{array}\right\} \overset{r=1}{\Rightarrow} G_\beta = P_\beta = a + 2\beta - 1 \tag{3.2}$$

$$P_{\alpha\beta} = \frac{P_\alpha + P_\beta}{2} \overset{r=1}{\Rightarrow} G_{\alpha\beta} = P_{\alpha\beta} = a - \alpha + \beta - \frac{1}{2} \tag{3.3}$$

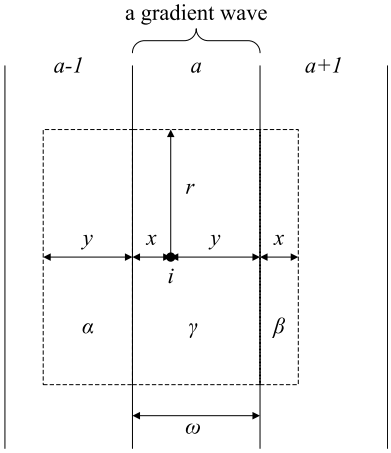


Figure 3.2: Adjacent hop-count gradient waves around node i.

Another approach to compute the smoothed gradient is to average the gradient values of the neighboring nodes [17]. The average gradient value is G_{avg} (in which n_l is the number of nodes in the transmission range of node i) and the smoothed gradient value is G_{SMG}. They are computed using Formula 3.4 and Formula 3.5 respectively.

$$\begin{aligned} G_{avg} &= \frac{n_l\alpha(a-1) + n_l(1-\alpha-\beta)a + n_l\beta(a+1)}{n_l} \\ &= a - \alpha + \beta \end{aligned} \tag{3.4}$$

$$G_{SMG} = G_{avg} - \frac{1}{2} = a - \alpha + \beta - \frac{1}{2} \tag{3.5}$$

3.2.2 Border Nodes

Formula 3.5 supposes that nodes in different hop-count gradient waves have different hop-count values. The nodes at the border of the hop-count gradient might discover nodes with non-initialized hop-count gradient values.

Nodes in the neighborhood of the seed nodes can discover others belonging to different gradient waves while still having the same hop-count values. In Figure 3.3(a), the line of seed nodes takes the first hop-count gradient value of 0. Then, according to the hop-count gradient algorithm, nodes in area α and γ have the same hop-count gradient value of 1. In Figure 3.3(b), gradient a is the last hop-count gradient wave, then the nodes in area β do not have a hop-count gradient value. Based on the formulas given in [17], we present methods to correctly compute the smoothed gradient values of the nodes placed at the border of the hop-count gradient.

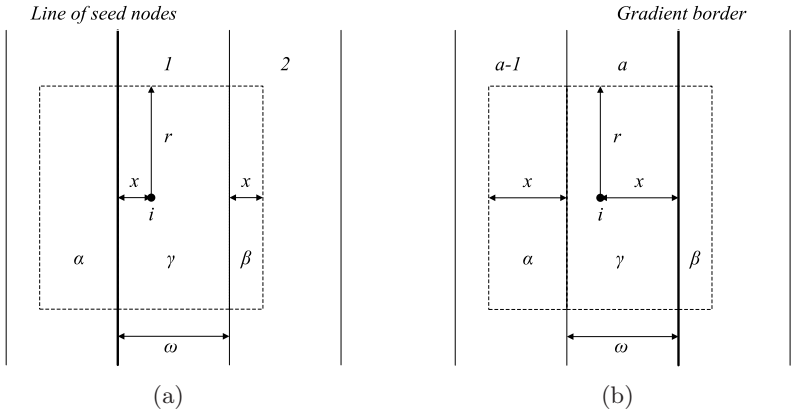


Figure 3.3: (a). Transmission range of node i when located in the neighbor area of the seed nodes. (b). Transmission range of node i when located near the border of the hop-count gradient.

If the node is in the first hop-count gradient wave, as shown in Figure 3.3(a), the smoothed gradient value can be computed using Formula 3.6. If the node is located in the last hop-count gradient wave, as shown in Figure 3.3(b), the smoothed gradient value can be computed using Formula 3.7. P_{first} and P_{last} are the computed distance from node i to the line of seed nodes in the first and last hop-count gradient wave respectively. G_{SMG}^{first} and G_{SMG}^{last} are the gradient values.

$$\left.\begin{array}{c} P_{first} = x \\ x = 2r\beta \end{array}\right\} \overset{r=1}{\Rightarrow} G_{SMG}^{first} = P_{first} = 2\beta \tag{3.6}$$

$$\left.\begin{array}{c} P_{last} = ar - x \\ x = 2r\alpha \end{array}\right\} \overset{r=1}{\Rightarrow} G_{SMG}^{last} = P_{last} = a - 2\alpha \tag{3.7}$$

Chapter 4

GDE Algorithm

GDE algorithm is based on the *HCG* and *SMG* algorithms that were introduced in Chapter 3. The basic *GDE* algorithm is designed for a static and spatially uniformly distributed deployment. It includes three basic computations: gradient width reduction, averaging of the reduced gradient width, and the overestimated width reduction. In this chapter we introduce the series of improvements that make *GDE* a suitable algorithm for a distributed static environment.

4.1 Gradient Width Reduction

Suppose node i has transmission range r and node k has the largest distance to node i among all the nodes in the transmission range of node i. The number of nodes in the area is not infinite, thus there is a high chance that node k is not exactly located on the border of the transmission range of node i, thus the distance between node k and node i is smaller than the transmission range r. This implies that the width of the hop-count gradient wave is smaller than r, and is referred as *width reduction of gradient waves.*

Figure 4.1 shows the phenomenon of width reduction of gradient waves. Suppose node n is located at the border p_{bu} of hop-count gradient wave $a-1$, and node m that is located at the border p_b of hop-count gradient a can just discover node n. Because node density is not infinite, there is a high chance that there is no node on the border p_{bu}, and the nodes that are closer to the border p_{bu} are on the line p_{wu}. So the real border of the hop-count gradient wave $a-1$ moves from p_{bu} to p_{wu}. Suppose node n is moved to position n'. The last node that can discover node n' has to move from position m to m'. The border of the hop-count gradient wave a changes from p_b to p_w, because of the movement of the previous border from p_{bu} to p_{wu}. The nodes positioned between p_b and p_w, such as node v, can no longer discover the nodes with hop-count gradient value $a-1$, so it updates the hop-count gradient value to $a+1$.

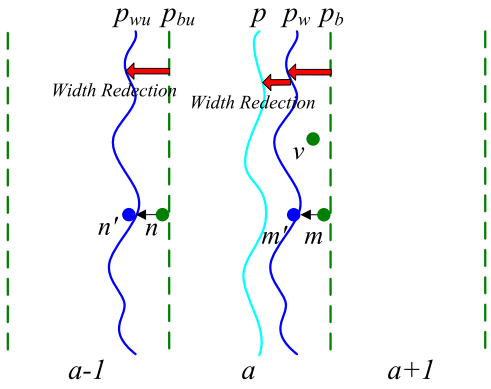


Figure 4.1: Width reduction of gradient waves.

At the same time, because of the nodes missing around border p_w, the nodes that are closer to the border p_w are on the line p. The real border of the hop-count gradient wave a moves from p_b to p. As a result, the width of the hop-count gradient wave becomes smaller. At the same time, the movement of the border always influences the border placements of all the next hop-count gradient waves, so the border relocation, such as from p_b to p in hop-count gradient wave a, becomes larger and larger.

Suppose the estimated distance corresponding to a communication hop is d_{hop}. Equation 4.1 (taken from [13]) shows the relation between the real estimated distance and the node density n_l.

$$d_{hop} \overset{r=1}{=} (1 + e^{-n_l} - \int_{-1}^{1} e^{-\frac{n_l}{\pi}(\arccos t - t\sqrt{1-t^2})} dt) \tag{4.1}$$

Let $\Delta = 1 - d_{hop}$ be the width reduction of each hop-count gradient wave. As explained, the border relocation has a cumulative characteristic and is related to the hop-count gradient value. The cumulative reduced width of the wave with hop-count gradient value a is $S_i = (a-1)\Delta$ and the gradient width reduction value can be computed using Formula 4.2.

$$G_{si} = G_{SMG} - S_i \tag{4.2}$$

4.2 Averaging of Reduced Gradient Width

The width reduction of the gradient wave introduced in Section 4.1 influences the result of the smoothed gradient. Each node in the one hop-count gradient wave could be placed at two positions. The first place is shown in Figure 4.2(a), where node n is not in the reduced width area of the gradient

wave. The second place is shown in Figure 4.2(b), where node n is in the reduced width area of the gradient wave.

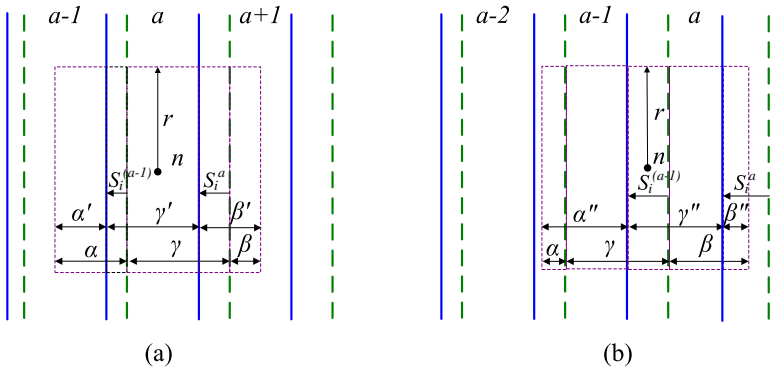


Figure 4.2: (a). Average gradient of node n outside the reduced width area of the gradient wave. (b). Average gradient of node n inside the reduced width area of the gradient wave.

In the first case, as shown by Figure 4.2(a), the percentages of discovered nodes in α, β, γ become α', β', γ', as shown in Equation 4.3.

$$\begin{aligned}
\alpha^{'} &= \frac{2r\alpha - S_i^{(a-1)}}{2r} \overset{r=1}{=} \frac{2\alpha - (a-1)\Delta}{2} \\
&= \alpha - \frac{1}{2}(a-1)\Delta \\
\beta^{'} &= \frac{2r\beta + S_i^{a}}{2r} \overset{r=1}{=} \frac{2\beta + a\Delta}{2} \\
&= \beta + \frac{1}{2}a\Delta \\
\gamma^{'} &= \frac{2r\gamma + S_i^{(a-1)} - S_i^{a}}{2r} \overset{r=1}{=} \frac{2\gamma + (a-1)\Delta - a\Delta}{2} \\
&= \gamma - \frac{1}{2}\Delta
\end{aligned} \tag{4.3}$$

In the second case, as shown by Figure 4.2(b), the percentages of discovered nodes in α, β, γ become α'', β'', γ'', as shown in Equation 4.4.

$$
\begin{aligned}
\alpha^{''} &= \frac{2r\alpha + (r - S_i^a)}{2r} \overset{r=1}{=} \frac{2\alpha + (1 - a\Delta)}{2} \\
&= \alpha + \frac{1}{2} - \frac{1}{2}a\Delta \\
\beta^{''} &= \frac{2r\beta - (r - S_i^{(a+1)})}{2r} \overset{r=1}{=} \frac{2\beta - [1 - (a+1)\Delta]}{2} \\
&= \beta - \frac{1}{2} + \frac{1}{2}(a+1)\Delta \\
\gamma^{''} &= \frac{2r\gamma - (r - S_i^a) + (r - S_i^{(a+1)})}{2r} \\
&\overset{r=1}{=} \frac{2\gamma - (1 - a\Delta) + [1 - (a+1)\Delta]}{2} \\
&= \gamma - \frac{1}{2}\Delta
\end{aligned}
\tag{4.4}
$$

If a node is outside the reduced width area, the average gradient value G_{avg}^{out} and the smoothed gradient value G_{si}^{out} are shown using Formula 4.5 and 4.6. If the node is inside the reduced width area, the average gradient value is G_{avg}^{in} and the smoothed gradient value becomes G_{si}^{in} as shown using Formula 4.7 and 4.8.

$$
\begin{aligned}
\mathrm{G}_{avg}^{out} &= \frac{n_l\alpha^{'}(a-1) + n_l\gamma^{'}a + n_l\beta^{'}(a+1)}{n_l} \\
&= a - 2\alpha + \frac{1}{2} + (a\Delta - \frac{1}{2}\Delta)
\end{aligned}
\tag{4.5}
$$

$$
\begin{aligned}
G_{si}^{\mathrm{out}} &= G_{avg}^{out} - \frac{1}{2} - (a-1)\Delta - \frac{1}{2}\Delta \\
&= G_{SMG} - S_i - \frac{1}{2}\Delta
\end{aligned}
\tag{4.6}
$$

$$
\begin{aligned}
\mathrm{G}_{avg}^{in} &= \frac{n_l\alpha^{''}(a-1) + n_l\gamma^{''}a + n_l\beta^{''}(a+1)}{n_l} \\
&= a - 2\alpha - \frac{1}{2} + a\Delta - \frac{1}{2}\Delta
\end{aligned}
\tag{4.7}
$$

$$
\begin{aligned}
G_{si}^{in} &= G_{avg}^{in} - \frac{1}{2} - (a-1)\Delta - \frac{1}{2}\Delta + 1 \\
&= G_{SMG} - S_i - \frac{1}{2}\Delta + 1
\end{aligned}
\tag{4.8}
$$

Due to the fact that nodes do not have position information and cannot tell whether they are inside or outside the reduced width area, we compute the expected average value of the gradients G_{si}^{out} and G_{si}^{in}. Let R_{in} be the percentage of the reduced width of one hop-count gradient wave compared to the transmission range r, as shown in Formula 4.9. Let R_{out} be the percentage of the not-reduced width of one hop-count gradient wave compared to the transmission range r, as shown in Formula 4.10. Then the gradient value G_{esi} is shown using Formula 4.11.

$$R_{in} = \frac{(r - rd_{hop})}{r} \overset{r=1}{=} 1 - d_{hop} \tag{4.9}$$

$$R_{out} = 1 - R_{in} \overset{r=1}{=} d_{hop} \tag{4.10}$$

$$G_{esi} = G_{si}^{out} R_{out} + G_{si}^{in} R_{in} \tag{4.11}$$

4.3 Overestimated Width Reduction

The gradient computation presented in Section 4.1 uses the expected average value d_{hop}. This expected average value will overestimate the width reduction of gradient waves. Suppose the gradient wave border moves as shown by Figure 4.3(a). If the border moves from p_b to p, and node n is located in the area $(p + d_{hop})^+$, which means node n cannot discover the nodes on the border p, then the computed hop distance d_{hop} is the expected width of hop-count gradient wave. The real border that moves to p can not be a straight line, but a curved line like p_r. Node m' may move to the position of node m, to be inside the transmission range of the node n. As a result, node n can discover some of the nodes on the border p. This implies that the reduced width Δ is overestimated.

Assuming the shape of the gradient waves is a set of parallel lines, the node density is ρ and the transmission range of each node is r. The probability that k nodes are located in area h is shown using Formula 4.12 [17]. For example, in Figure 4.3(b), node n has d_{hop} as the real estimated distance, which is introduced in Formula 4.1. Let p be the gradient border using d_{hop}, p_d be the real border of the hop-count gradient, X be the overestimated reduced width, and A be the overlap area of the overestimated reduced width area and the transmission range. It can be observed that the probability that there are no nodes in the area A equals the probability that the gradient border p will have a shift larger than x, as shown using Formula 4.13. Therefore, the probability distribution function of the overestimated reduced width X is shown using Formula 4.14.

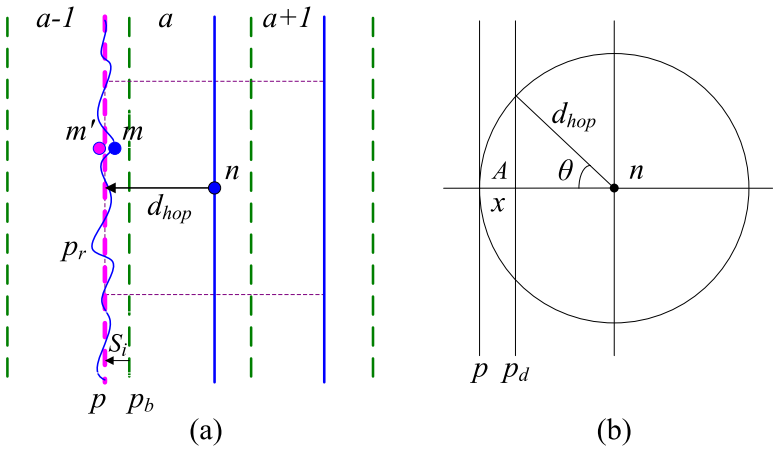


Figure 4.3: (a). Overestimation of the gradient wave width reduction. (b). Expected overestimated reduced wave width.

$$\Pr(k \in h) = \frac{(\rho h)^k}{k!} e^{-\rho h} \tag{4.12}$$

$$\Pr(0 \in A) = \Pr(X > x) = e^{-\rho A} \tag{4.13}$$

$$F(x) = \Pr(X \leq x) = 1 - e^{-\rho A} \tag{4.14}$$

The size of area A can be computed using Formula 4.15 and the probability density function of X is shown by Formula 4.16. The expected overestimated reduced width Δr of one hop-count gradient wave can be written as Formula 4.17. Let a be the hop-count gradient value of a node. The gradient value for each node is computed as shown by Formula 4.18.

$$\begin{aligned} A &= 2[\pi r^2 \frac{\arccos\frac{r-x}{r}}{2\pi} - \frac{(r-x)\sqrt{r^2-(r-x)^2}}{2}] \\ &\overset{r=1}{=} \arccos(1-x) - (1-x)\sqrt{2x-x^2} \end{aligned} \tag{4.15}$$

$$\begin{aligned} f(x) &= \mathrm{F}'(x) = (1 - e^{-\rho A})dx \\ &= \frac{\rho - 2x^2 + 4x - 1}{e^{(x-1)\sqrt{2x-x^2}+\rho \arccos(1-x)}\sqrt{2x-x^2}} \end{aligned} \tag{4.16}$$

$$\begin{aligned}\Delta r = E(x) &= \int_0^1 x f(x) dx \\ &= \int_0^1 x \frac{\rho - 2x^2 + 4x - 1}{e^{(x-1)\sqrt{2x-x^2}+\rho \arccos(1-x)}\sqrt{2x-x^2}} dx \end{aligned} \tag{4.17}$$

$$G_{GDE} = G_{si}^{out} R_{out} + G_{si}^{in} R_{in} + (a-1) d_{hop} \Delta r \tag{4.18}$$

4.4 Error Reference Value

This section introduces an error reference value, which can be used as a benchmark for the testing results of the distance estimation errors.

If node density ρ tends to $+\infty$, then Δ and Δr, which are presented in Section 4.1 and Section 4.3 respectively, converge to 0, as shown by Equations 4.19 and 4.20, which means the gradient width reduction and the gradient width overestimation problems disappear for large node densities.

$$\lim_{\rho \to +\infty} \Delta \overset{r=1}{=} 1 - \lim_{\rho \to +\infty} d_{hop} = 0 \tag{4.19}$$

$$\begin{aligned} &\lim_{\rho \to +\infty} F(x) \overset{r=1}{=} \lim_{\rho \to +\infty} (1 - e^{-\rho a}) = 1 \\ &\Rightarrow \lim_{\rho \to +\infty} f(x) = 0 \Rightarrow \lim_{\rho \to +\infty} \Delta r = 0 \end{aligned} \tag{4.20}$$

The number of nodes n in the area of size h follows a *Poisson* distribution with parameter λ (λ=ρh). If node density ρ and λ goes to $+\infty$, then the probability of n nodes in the area of size h tends to follow a *Normal* distribution with a mean λ and a deviation of λ, as shown using Formula 4.21. Therefore, if node density ρ goes to $+\infty$, the number of nodes in various hop-count gradient waves of one transmission range also obeys a *Normal* distribution, as shown in Formula 4.22, in which n_α and n_β are the number of nodes in sections α and β, and S_α and S_β are the area size of α and β.

$$n \sim P(\lambda) = \frac{\lambda^n}{n!} e^{-\lambda} \overset{\lambda \to +\infty}{\Rightarrow} n \sim N(\lambda, \lambda) \tag{4.21}$$

$$\begin{aligned} n_\alpha &\sim N(\rho S_\alpha, \rho S_\alpha) \\ n_\beta &\sim N(\rho S_\beta, \rho S_\beta) \end{aligned} \tag{4.22}$$

The linear combination of *Normal* distributions is also a *Normal* distribution. The computed gradient value using Formula 4.18 follows a *Normal* distribution as shown using Formula 4.23, in which n_l is the average number of nodes in the transmission range of a node.

$$\begin{aligned}\lim_{\rho\to+\infty} G_{GDE} &= \lim_{\rho\to+\infty}(G_{si}^{out}R_{out} + G_{si}^{in}R_{in}) \\ &= \lim_{\rho\to+\infty}(a - \alpha + \beta - \frac{1}{2}) \\ &= \lim_{\rho\to+\infty}(a - \frac{n_\alpha}{n_l} + \frac{n_\beta}{n_l} - \frac{1}{2}) \\ &\sim N(a - \frac{\rho S_\alpha}{n_l} + \frac{\rho S_\beta}{n_l} - \frac{1}{2}, \frac{\rho(S_\alpha + S_\beta)}{n_l^2})\end{aligned} \tag{4.23}$$

If nodes are distributed on a grid and node density goes to infinity, then by using Formula 4.24 the gradient is computed as G_{GDE}^{g}. If nodes in the area are uniformly distributed and node density goes to infinity, then using Formula 4.25 the gradient is computed as G_{GDE}^{u}.

$$G_{GDE}^{g} \overset{\rho\to+\infty}{=} a - \frac{\rho S_\alpha}{n_l} + \frac{\rho S_\beta}{n_l} - \frac{1}{2} \tag{4.24}$$

$$G_{GDE}^{u} \overset{\rho\to+\infty}{=} a - \frac{n_\alpha}{n_l} + \frac{n_\beta}{n_l} - \frac{1}{2} \tag{4.25}$$

If all nodes are uniformly distributed, the error becomes ΔG, which is the absolute difference value between the gradient that is computed when nodes are distributed on a grid and the gradient computed when nodes are uniformly distributed, as shown using Formula 4.26, in which S_l is the area size of the transmission range.

$$\begin{aligned}\Delta\mathrm{G} &= G_{GDE}^{u} - G_{GDE}^{g} \\ &= (a - \frac{n_\alpha}{n_l} + \frac{n_\beta}{n_l} - \frac{1}{2}) - (a - \frac{S_\alpha}{S_l} + \frac{S_\beta}{S_l} - \frac{1}{2}) \\ &= (\frac{S_\alpha}{S_l} - \frac{n_\alpha}{n_l}) + (-\frac{S_\beta}{S_l} + \frac{n_\beta}{n_l})\end{aligned} \tag{4.26}$$

Formula 4.22 shows the probability distribution of the number of nodes n_α and n_β, while ΔG is the linear combination of the *Normal* distributions of n_α and n_β. The error ΔG also follows a *Normal* distribution, as shown using Formula 4.27.

$$\begin{aligned}\Delta \mathrm{G} &\sim N(\frac{S_\alpha}{S_l} - \frac{\rho S_\alpha}{n_l}, \frac{\rho S_\alpha}{n_l^2}) + N(-\frac{S_\beta}{S_l} + \frac{\rho S_\beta}{n_l}, \frac{\rho S_\beta}{n_l^2}) \\ &\overset{n_l=S_l\rho}{\sim} N(0, \frac{\rho S_\alpha}{n_l^2}) + N(0, \frac{\rho S_\beta}{n_l^2}) \sim N(0, \frac{\rho(S_\alpha + S_\beta)}{n_l^2}) \\ &\overset{S_\alpha+S_\beta=\frac{1}{2}S_l}{\sim} N(0, \frac{1}{2n_l}) \end{aligned} \tag{4.27}$$

The average absolute error for all nodes is the expected value for $|\Delta G|$. The mean value of ΔG is 0 and the probability distribution of ΔG is symmetrical to $x = 0$. To compute the expectation value of $|\Delta G|$, we first compute the expected value in [0, +), then double the result as the expected value of $|\Delta G|$. Let ΔG^+ be the function of ΔG within function domain [0, +). Z is the standard $Normal$ distribution of ΔG within function domain [0, +). u and δ are the mean value and the standard deviation of the Normal distribution ΔG. The expected value of Z is presented by Formula 4.28.

$$\begin{aligned} Z &= \frac{\Delta \mathrm{G}^+ - u}{\delta} \overset{u=0}{=} \frac{\Delta \mathrm{G}^+}{\delta} \\ f(z) &= \frac{1}{\sqrt{2\pi}} e^{-\frac{z^2}{2}} \\ E(Z) &= \frac{1}{\sqrt{2\pi}} \int_0^{+\infty} z e^{-\frac{z^2}{2}} dz = \frac{1}{\sqrt{2\pi}} \end{aligned} \tag{4.28}$$

ΔG is the linear function of $E(Z)$, therefore the expected absolute error of distance estimation can be computed using Formula 4.29.

$$\begin{aligned} E(|\Delta G|) &= 2E(\delta Z) = 2\delta \mathrm{E(Z)} \\ &= \frac{2}{\sqrt{2n_l}} \frac{1}{\sqrt{2\pi}} = \frac{1}{\sqrt{\pi n_l}} \end{aligned} \tag{4.29}$$

It can be seen that as the node density goes to infinity, the error goes to zero. In reality, the node density can not go to infinity. This result is the error of distance estimation in the ideal case, which can be used as a benchmark for the experimental results.

Chapter 5

GDE Algorithm Parameters

So far we have discussed about algorithms in more or less idealized conditions. However, for real deployments, wireless systems have to cope with many problems such as node mobility, communication failures, non-uniform node distributions, influence of different communication paradigms (unicast vs. multicast), etc. In this chapter we analyze these aspects and present methods to make the *GDE* algorithm adapt to various parameters.

5.1 Approximation Error

The formulas of *GDE* algorithm that compute the gradient values suppose that all nodes are uniformly distributed in the area and the transmission area of each node is a square. The formulas that compute the overestimated width reduction of the gradient in Section 4.3 consider the distribution of nodes, but the computations still presume that the shape of the hop-count gradient is a set of parallel lines. In reality, the transmission range of the node is more like a circle. And most of the time only one or a small number of nodes can be set as the seed nodes, then the shape of the hop-count gradient waves could be a circle with the seed node at the center. In the previous computations, transmission range is approximated to square and parallel gradient waves are linearized to parallel lines. The approximation from nonlinear conditions to the linear conditions would introduces errors. This section explains the error introduced by the use of parallel gradient shape instead of circle gradient shape. At the same time, we analyze some of the characteristics of the nonlinear conditions, and present an approach to repair the error introduced by the condition that the circle transmission range is in the circle gradient waves.

5.1.1 Circle Gradient

If the hop-count gradient shape is circle instead of parallel lines, the overestimated gradient shrink compensation value Δr that is computed as shown in Section 4.3 still holds but the size of the overlapping area is different. In Figure 5.1, o is the place of the seed node, n is a node with real transmission radius d_{hop}, K is one circle border of the hop-count gradient wave without considering the overestimated wave width reduction, $K^{'}$ is the border considering the overestimated wave width reduction. If the circle gradient border K and the compensated border $K^{'}$ has overlapping areas with the transmission range of node n, which are represented by areas a_1 and a_2. If there is a line of seed nodes vertically placed at position o, then the gradient border K will become p and the compensated border $K^{'}$ will become p_d. It can be found that the overlapping area of the gradient borders p, p_d and the transmission range of node n is almost equal to the overlapping areas a_1 and a_2 which are created by the circle borders K, $K^{'}$ and the transmission range. With the increasing of the radius of the gradient wave circle, the circle gradient wave lines K and $K^{'}$ will be more and more close to the parallel wave lines p and p_d. The difference of the areas between the circle waves and the parallel waves will become smaller and smaller. If the radius of the circle gradient wave is infinity, then the gradient border can be seen as parallel lines. If the radius of the circle gradient wave is small, then the overestimated shrink compensation value Δr will be overestimated.

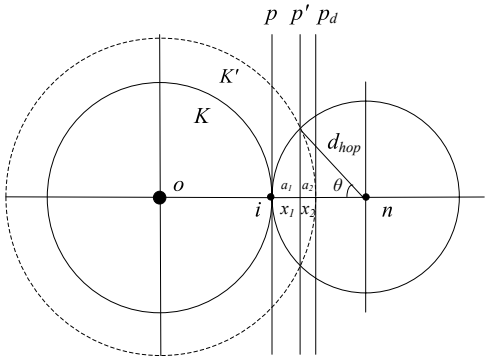


Figure 5.1: Node n with circle transmission range in circle gradient waves.

We present the detailed computation method for the over estimated width reduction of the circle gradient waves. As shown by Figure 5.1, a_1 and a_2 are the two parts of the overlapped areas, x_1 and x_2 are the width of the two overlapped areas, $p^{'}$ is a line used to split the overlapping area into a_1 and a_2. Area A, which is computed by Formula 4.15, is composed of the area a_1 and a_2. Suppose the length x is the sum of length x_1 and x_2, R is the radius of the circle gradient wave K, r is the transmission range which equals d_{hop}. Then the relation between x_1 and x, a_1 and x_1, a_2 and x_1 can

be computed using Formulas 5.1, 5.2 and 5.3. When the transmission range r is normalized to 1, the radius R can be replaced by the gradient value G_i of node i. Since the gradient difference between node n and node i is 1, we use $G_i = G_n - 1$ to compute the gradient value of node i, in which G_n is the gradient value of node n.

$$\left.\begin{array}{c}(R+x)^2-(R+x_1)^2=r^2-(r-x_1)^2 \Rightarrow \frac{R}{r}=\frac{x_1}{x_2} \\ x_1+x_2=x\end{array}\right\}$$
$$\Rightarrow x_1=\frac{R}{R+r}x \overset{r=1,R=G_i}{\Rightarrow} x_1=\frac{G_i}{G_i+1}x \tag{5.1}$$

$$a_1 \overset{r=1}{=} \arccos(1-x_1)-(1-x_1)\sqrt{2x_1-x_1^2} \tag{5.2}$$

$$\begin{aligned}a_2 &= 2[\pi(R+x)^2\frac{\arccos\frac{R+x_1}{R+x}}{2\pi}-\frac{(R+x_1)\sqrt{(R+x)^2-(R+x_1)^2}}{2}] \\ &= (R+x)^2\arccos\frac{R+x_1}{R+x}-(R+x_1)\sqrt{x^2-x_1^2+2Rx-2Rx_1} \\ &\overset{r=1,R=G_i}{=} (G_i+x)^2\arccos\frac{G_i+x_1}{G_i+x}-(G_i+x_1)\sqrt{x^2-x_1^2+2G_ix-2G_ix_1}\end{aligned} \tag{5.3}$$

$$\begin{aligned}A &= a_1+a_2 \\ &= \arccos(1-\frac{G_i}{G_i+1}x) \\ &\quad -(1-\frac{G_i}{G_i+1}x)\sqrt{\frac{2G_i}{G_i+1}x-(\frac{G_i}{G_i+1}x)^2} \\ &\quad +(G_i+x)^2\arccos\frac{G_i+\frac{G_i}{G_i+1}x}{G_i+x} \\ &\quad -(G_i+\frac{G_i}{G_i+1}x)\sqrt{x^2-(\frac{G_i}{G_i+1}x)^2+2G_ix-2G_i\frac{G_i}{G_i+1}x} \\ &= \phi(x)\end{aligned} \tag{5.4}$$

The area A is the function with parameter x, which is shown in the Formula 5.4 as $\phi(x)$. Then the expectation of the overestimated width reduction length can be computed in Formula 5.5. So if there is only one node as the seed node and the hop-count number is very small, then Formula 5.5 is more accurate than Formula 4.17 to compute the overestimated width reduction length of the gradient waves. As the radius of the circle gradient wave increases, the circle gradient waves will become more and more like parallel

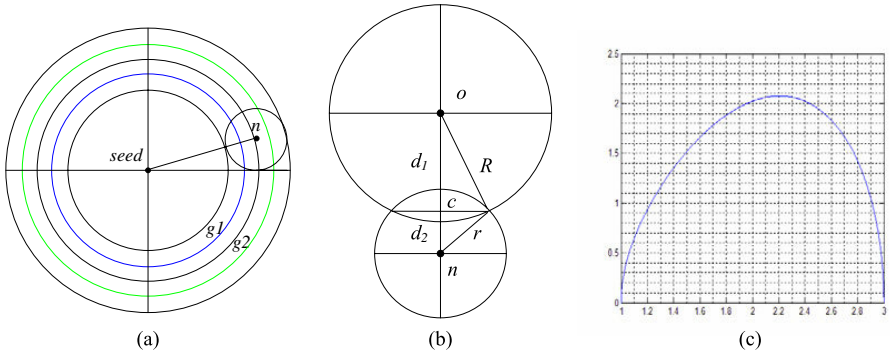
gradient waves, then the improvement on the accuracy using the formula introduced in this section will decrease.

$$\Delta r = E(x) = \int_0^1 xf(x)dx = \int_0^1 x(1 - e^{-\rho\phi(x)})'dx \tag{5.5}$$

5.1.2 Circle Transmission Range

The formulas provided in the previous section suppose the transmission range to be square to simplify the computation. Formula 5.5 recomputes the overestimation of the width reduction. The error of the linear approximation of the square transmission range in the other formulas can also introduce error in the results of distance estimation. Here, we present a method to take advantage of the nonlinear characteristic of circle gradient wave and circle transmission range to repair the error of the distance estimation.

Suppose the gradient waves are circle shapes and the transmission range of each node is also a circle as shown in the Figure 5.2(a), in which g_1 and g_2 are two gradient lines that have the same distance to node n. It can be found that the length of the crossed arcs of two gradient lines g_1 and g_2 and the transmission range of node n are different.



(a) (b) (c)

Figure 5.2: (a). The circle transmission range of node n with two circle gradients g_1 and g_2. (b). The circle transmission range with a circle gradient. (c). The relation between the radius of the gradient circle and the length of the crossed arc of the gradient circle and the transmission range.

Now we present the computation of the length of the crossed arc of the gradient line and the transmission range of node n. As shown in Figure 5.2(b), o is the seed node and n is a node placed in the gradient, the circle with radius R is one of the gradient circles, the circle with radius r is the transmission range of node n, d_1 is the distance from o to line c, d_2 is the distance from n to line c. Suppose d is the distance between the seed node o and the node n, l is the length of the crossed arc of the gradient

circle and the transmission range. Then the length of the crossed arc l of the gradient line and the transmission range of node n can be computed as shown by Formula 5.6.

$$\left.\begin{array}{c} r^2 - d_2^2 = R^2 - d_1^2 \\ d_1 + d_2 = d \end{array}\right\} \Rightarrow l = 2R \arccos \frac{R^2 - r^2 + d^2}{2Rd} \quad (d - r \leq R \leq d + r) \tag{5.6}$$

It can be seen from Formula 5.6, that if the transmission range r and the distance d are fixed values, then the length of arc l is the function of radius R. Figure 5.2(c) shows the relation between the radius of the gradient circle and the length of the crossed arc. In Figure 5.2(c), we fix the value of r to 1 and d to 2, and the x coordinate is the radius of the gradient circle and y coordinate is the length of the crossed arc of the gradient circle and the transmission range. It can be found that if two gradient circles have the same distance to node n, then the gradient circle with larger radius R has larger arc value l. It means that, in the same node distribution area, the probability that the nodes on the gradient circle with smaller radius is more likely to be not discovered than on the gradient circle with larger radius.

Suppose there are two nodes n_{close} and n_{far}, and both nodes n_{close} and n_{far} are in the transmission range of node n. Node n_{close} is the node that is closest to the seed node among the nodes in the transmission range of node n, and node n_{far} is the node that is the furthest to the seed node among the nodes in the transmission range of node n. Then n_{close} is on the gradient circle with smaller radius, while n_{far} is on the gradient circle with larger radius. So according to the Formula 5.6, the probability that the node n_{close} is not discovered by node n is larger than the probability that node n_{far} is not discovered by node n. This conclusion also means that, from the probabilistic point of view, the distance from node n to node n_{far} is larger than the distance from the node n to the node n_{close}, since the theoretical closest node n_{close} is more likely not be discovered than the theoretical furthest node n_{far}.

Algorithm 3 presents the detail steps to repair the error of the circle transmission range in the circle gradients. Suppose the gradient values become larger and larger from the seed nodes. G_{GDE} is the GDE gradient value of node n. G_{min} is the minimum gradient value of the neighbors, which means the node is closest to the seed node among all neighbors of node n. G_{max} is the maximum gradient value of the neighbors, which means the node is furthest to the seed node among all neighbors of node n. Suppose that the gradient difference between G_{GDE} and G_{max} is statistically larger than the difference between G_{GDE} and G_{min}. If the gradient difference between G_{GDE} and G_{min} is statistically larger than the difference between G_{GDE} and G_{max}, then we recompute the gradient value of node n to G_{GDE}^{repair}

Algorithm 3: Repair Error Statistically

1: G_{GDE} ▷ Local gradient value
2: G_{min} ▷ Minimum gradient value in neighbors
3: G_{max} ▷ Maximum gradient value in neighbors

4: **if** $|G_{GDE} - G_{min}| > |G_{max} - G_{GDE}|$
5: **set** G_{GDE}^{repair} $[(G_{min} + G_{max})/2]$
6: **end if**

The algorithm provides a method to repair the distance estimation value from the statistical point of view. It is true that, because of the random distribution of nodes in the area, there exists the possibility that the node distance estimation is correct but $[|G_{GDE} - G_{min}| > |G_{max} - G_{GDE}|]$. Then the Algorithm 3 will introduce error to the gradient value. Statistically, the error introduced is always smaller than the error decreased by Algorithm 3. At the same time, it can be found that if the distance d is much larger than the radius r, the gradient shape is almost like parallel lines, the difference between the arc lengths that is closer and further to the seed node does not exist. So Algorithm 3 only works for gradients built for small network diameters.

5.2 Node Mobility

In Section 4, we introduce *GDE* algorithm in the static environment. However, in most cases of the application, nodes are dynamic. In this section, we present a probability distribution function of nodes in an area, and then provides a method to compensate the influence of mobile nodes to the distance estimation of *GDE* algorithm.

5.2.1 Node Distribution

As shown in Formula 4.12, suppose the nodes are static and uniformly distributed in the area, then the probability that k nodes are in the area of size h follows the Poisson distribution. If the nodes are dynamic, intuitively, for every discrete moment in time, the probability that k nodes are in the area of size h also follows the same Poisson distribution. We present the probability density distribution of the number of nodes in the transmission range of each node in dynamic.

Suppose all nodes are dynamic. Node m enters the transmission range of node i according to a Poisson process with intensity λ. Node m stays in the transmission range of node i for T_m time. The times T_m of all nodes are independent. The following formulas present the proof that the number of

nodes $R(t)$ in the transmission range of node i at any time t has a Poisson distribution.

Suppose S_m is the time point that the node m enters the transmission range of node i. T_m is the time that the node m stays in the transmission range of node i. m is the node that goes into the transmission range of node i, and the value of m is from 1 to $N(t)$, in which $N(t)$ represents the largest node number that has going into the transmission range of node i before time t. Then the number of nodes $R(t)$ in the transmission range of node i at any time t can be described using Formula 5.7.

$$\mathrm{R(t)} = \sum\nolimits_{m=1}^{N(t)} 1_{\{S_m+T_m \geq t\}} \tag{5.7}$$

According to the law of total probability, the number of nodes $R(t)$ in the transmission range of node i at any time t can be described using Formula 5.8.

$$P[R(t) = q] = \sum\nolimits_{n=0}^{\infty} P[R(t) = q|N(t) = n]P[N(t) = n] \tag{5.8}$$

Suppose W_t is the time point that the node enters the transmission range of node i from time 0 to time t. According to the characteristic of Poisson distribution, the time W_t is uniformly distributed in the range [0, t] [30]. The conditional probability in Formula 5.8 can be written as Formula 5.9.

$$\begin{aligned} P[R(t) = q|N(t) = n] &= P[\sum\nolimits_{m=1}^{n} 1_{\{S_m+T_m \geq t\}} = q|N(t) = n] \\ &= P[\sum\nolimits_{m=1}^{n} 1_{\{W_m+T_m \geq t\}} = q] \end{aligned} \tag{5.9}$$

Now we compute the probability that $W_m + T_m \geq t$. The probability distribution function is $F_T(x) = P[T \leq x]$. Then the result can be shown in Formula 5.10. p is the probability that that $W_m + T_m \geq t$.

$$\begin{aligned} p &= P[W_m + T_m \geq t] \\ &= \int_0^t P[W_m + T_m \geq t|W_m = w]\frac{dP[W_m \leq w]}{dw}dw \\ &= \frac{1}{t}\int_0^t P[T_m \geq t - w]dw = \frac{1}{t}\int_0^t (1 - P[T \leq t - w])dw \\ &= \frac{1}{t}\int_0^t (1 - F_T(t - w))dw = \frac{1}{t}\int_0^t (1 - F_T(x))dx \end{aligned} \tag{5.10}$$

Then the conditional probability of Formula 5.9 can be written as Formula 5.11.

$$P[R(t) = q|N(t) = n] = \left(\begin{array}{c} n \\ q \end{array} \right) p^q (1-p)^{n-q} \tag{5.11}$$

Finally, we put Formula 5.11 back into Formula 5.8, then we get the result Formula 5.12.

$$\begin{aligned} P[R(t) = q] &= \sum\nolimits_{n=0}^{\infty} P[R(t) = q|N(t) = n] P[N(t) = n] \\ &= \sum_{n=q}^{\infty} \frac{n!}{q!(n-q)!} p^q (1-p)^{n-q} \frac{(\lambda t)^n}{n!} e^{-\lambda t} \\ &= \frac{(p\lambda t)^q}{q!} e^{-p\lambda t} \end{aligned} \tag{5.12}$$

It can be seen from Formula 5.12, that the number of nodes $R(t)$ in the transmission range of node i at any time t follows a Poisson distribution. At the same time, it can be seen that the Poisson process of the result not only depends on the intensity of the incoming nodes, but also depends on the distribution function of $W_m + T_m$, which is shown by Formula 5.10. The distribution function of the time that the nodes stay in the transmission range of another node is decided by the moving model, therefore further research on the moving model and the staying time in the transmission range can more clearly show the distribution of the number of nodes in the transmission range. Further insight on the moving model is not the research point of the thesis. In this thesis, we use the Formula 4.12 to model the distribution of mobile nodes.

5.2.2 Node Mobility Compensation

In the previous sections we described an algorithm for the case when the nodes are static. In this section, we introduce a compensation strategy for the case in which the nodes are dynamic (i.e., we consider the Random Walk mobility model assuming a discrete time model).

In Figure 5.3(a), p_b is the border of a hop-count gradient wave. Node m can travel from position p to p' and p'' in one time step. The last node that can discover the border p_b changes from node n to node n', which enlarges the width of the hop-count gradient wave in the gradient. A strategy to compensate the enlarged width in the distance estimation value is needed.

We use the expected moving distance of one node in a given time step E_{dc} to compensate the enlarged width value, which equals the movement distance from p_b to p''. The translation of the hop-count gradient into x and y coordinates is shown by Figure 5.3(b), in which y coordinate is parallel to the line of the seed nodes.

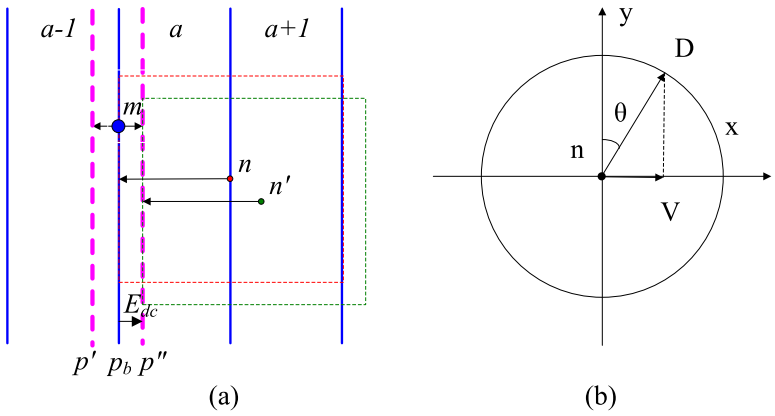


Figure 5.3: (a). Influence of dynamic node to the border of the hop-count gradient waves. (b). Expected moving distance of the gradient wave border.

When nodes move with various speeds, the average expected moving distance for a node in one time step is D which makes an angle θ with the y axis (V being the projection on it). Due to the fact that all the possible movement directions are symmetrical with respect to the x coordinate, only the movement for $y > 0$ is considered (θ ranging from $-\pi/2$ to $\pi/2$). The probability density function of V is given by Equation 5.13. Since we need the absolute expected moving distance of one node, the expected value is integrated from 0 to D as shown by Equation 5.14.

$$\left.\begin{array}{c}\theta \sim U(-\frac{\pi}{2}, \frac{\pi}{2}) \\ V = D\sin\theta\end{array}\right\} \Rightarrow f(v) \overset{-D\leq v<D}{=} \frac{1}{\pi\sqrt{D^2 - v^2}} \tag{5.13}$$

$$E_{dc} = E(v) = \int_0^D v f(v) dv = \int_0^D \frac{v}{\pi\sqrt{D^2 - v^2}} dv \tag{5.14}$$

The enlarged width of each gradient wave is accumulating from one hop-count gradient to the next. For a node with hop-count gradient value a and transmission range r, the dynamic compensation value C_d and the gradient value $G_{GDE}^{comp\,mob}$ are given by Equations 5.15 and 5.16.

$$C_d = [(a-1)d_{hop}E_{dc}]/r \tag{5.15}$$

$$G_{GDE}^{comp\,mob} = G_{GDE} + C_d \tag{5.16}$$

The computations above suppose that all nodes have the same constant speed D for different moving directions. In practical applications, all nodes

move with different speeds. Here we present two methods to compensate the speed.

The first method supposes that we know the average expected speed of the nodes $E(D)$. Then replace the average expected moving distance for a node in one time step D with $E(D)$ in the Formula 5.13, and all the other computations are the same as in Formulas 5.14, 5.15 and 5.16.

The second method supposes that we know more detailed conditions about the node speeds, such as the probability density function of all nodes. Then the computation of the expected movement distance can be used to compensate the movement of nodes. Suppose θ is the moving direction of the node, X is the function of θ, and θ is uniformly distributed from $-\pi/2$ to $\pi/2$, then the probability density function of X is shown in Formula 5.17. Suppose Y is the speed of the node with uniform distribution from 0 to D. The probability density function of Y can be computed in Formula 5.18. Suppose V is the moving distance to the positive direction of x. Then the relation between V, X and Y is shown in Formula 5.19, in which X and Y are mutual independent with each other [28]. And the probability density function of V can be computed using Formula 5.20. Then compute the average expectation value of v as Formula 5.21.

$$\left.\begin{array}{c}\theta \sim U(-\frac{\pi}{2}, \frac{\pi}{2}) \\ X = \sin\theta\end{array}\right\} \Rightarrow f(x) = \left\{\begin{array}{c}\frac{1}{\pi\sqrt{1-x^2}}, -1 < x < 1 \\ 0, others\end{array}\right. \tag{5.17}$$

$$Y \sim U(0, D) \Rightarrow f(y) = \left\{\begin{array}{c}\frac{1}{D}, 0 < y < D \\ 0, others\end{array}\right. \tag{5.18}$$

$$V = Y\sin\theta = XY \tag{5.19}$$

$$\begin{aligned} f(v) &= \int_{-\infty}^{+\infty} \frac{1}{|x|} f(x, \frac{v}{x}) dx = \int_{-\infty}^{+\infty} \frac{1}{|x|} f_X(x) f_Y(\frac{v}{x}) dx \\ &= \int_{v/D}^{1} \frac{1}{|x|} \frac{1}{\pi\sqrt{1-x^2}} \frac{1}{D} dx \\ &= -\frac{1}{\pi D} In \frac{D - \sqrt{D^2 - v^2}}{v}, 0 < v < D \end{aligned} \tag{5.20}$$

$$\begin{aligned} E_{dc} = E(v) &= \int_0^D v f(v) dv \\ &= -\frac{1}{\pi D} \int_0^D v In \frac{D - \sqrt{D^2 - v^2}}{v} dv \end{aligned} \tag{5.21}$$

The formulas above present the process of using the speed distribution function to compute the compensation value. In the computation we suppose the moving direction parameter θ follows the uniform distribution, and the moving speed parameter Y also follows the uniform distribution. In applications, the distribution functions of some parameters could be different. Even if the distribution functions are different, the computation process provided here is still the same.

5.3 Multicast Communication

GDE algorithm does not require for each node to communicate with all its neighbors. Multicast can be used to decrease the number of messages in the network. Suppose the nodes are uniformly distributed in the area. Each node only selects a fraction of its neighbors to communicate with. The difference compared to the original network is that the new network has an average lower node density. According to the formulas in Section 4, *GDE* can estimate distance for networks with different node densities. Therefore, each node can use multicast to communicate with their neighbors. The equations of *GDE* algorithm introduced in Section 4 can still be used, while the only difference is that node density n_l needs to be adjusted with the percentage of neighborhood size considered.

At the same time, according to the design, *GDE* uses the gossiping communication mechanism, which determines the *GDE* algorithm to have a lower message complexity compared with flooding. Suppose a fixed network with n nodes. If a single message needs to be spread in a network using randomized gossip, the round times of communication converges to $O(logn)$ and the message complexity converges to $O(n \cdot logn)$ [23]. If each node in the network broadcasts the message to the other nodes in the network, the worst-case of the message complexity becomes $O(n^2)$, which is much larger than the gossip-based information dissemination.

5.4 Round Length

GDE assumes a discrete time model, with un-synchronized communication rounds of size P. Each node acts once in each round, in a different time slot. The rounds are not synchronized and are being used to mimic the asynchronous nature of communication. Suppose all nodes follow a Random Walk mobility model, the transmission range is r and the expected moving distance at one time step is E_{dc}. We model the motion segments with a Markov Chain in order to compute the expected moving distance (see Figure 5.4). In the model we only consider nodes moving in one dimension (i.e., towards the next hop-count gradient). Suppose that state j, $j = 0, ..., N$, is the state that describes a node moving forward for a distance

$j \cdot E_{dc}$. p_j represents the probability that the node moves further from the original position, while q_j the probability of the node to move closer to the original position. Let N be the maximum number of moving steps in one round of length P. Let r_0 be the probability that a node does not move out of the initial state and is equal to $\frac{1}{2}$, and let p_0 be the probability that a node moves out of the initial state and is equal to $\frac{1}{2}$. In the inner processes, the backward and forward motion probabilities of a node in one time step are the same, so the node moves to the next state or to the previous state with $p_j = q_j = \frac{1}{2}$. For the final state N, due to the fact that there are nodes that can move to the previous state or start a new computation round, let r_N be the probability that a node starts a new round (equal to $\frac{1}{2}$), and q_N be the probability that a node moves backward (equal to $\frac{1}{2}$). Due to the fact that there is a large number of nodes, the probability of each steady state can be used as the probability that a node stays in that state for P steps. Considering the mathematical properties of the Random Walk [30] , the model that describes the steady states π_j $(0 \leq j \leq N)$ is given by Equation 5.22. One can notice that all states converge to the same probabilities.

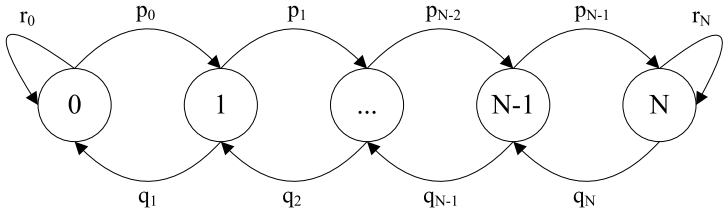


Figure 5.4: Moving states of each time step of a round in the Markov Chain.

$$\begin{aligned}
&\pi_0 = r_0\pi_0 + q_1\pi_1 \overset{r_0=q_1=\frac{1}{2}}{\Rightarrow} \pi_0 = \pi_1 \\
&\pi_N = p_{N-1}\pi_{N-1} + r_N\pi_N \overset{p_{N-1}=r_N=\frac{1}{2}}{\Rightarrow} \pi_N = \pi_{N-1} \\
&\pi_j = \frac{\prod_{m=0}^{j-1}\frac{p_m}{q_{m+1}}}{1+\sum_{j=1}^{N}\prod_{m=0}^{j-1}\frac{p_m}{q_{m+1}}} \overset{p=q=\frac{1}{2}}{=} \frac{1}{1+N} \quad (1 \leq j < N)
\end{aligned} \tag{5.22}$$

The node in state 0 does not affect the gradient of nodes in the next hop-count gradient. For this reason, we ignore state 0. For each state in the Markov Chain, it represents the expected moving distance from the original place. For state j, the moving distance is $j \cdot E_{dc}$. Because every steady state has the same probability as shown by Equation 5.22, the expected value for the moving distance of all states can be computed by Equation 5.23. It can be found that if the length of the round becomes 1, Equation 5.23 is the same as Equation 5.14. The compensation value for the round length is given in Equation 5.24, and the computation for the gradient value is given in Equation 5.25.

$$E_{dc}^{p} = \frac{\sum_{j=1}^{P} jE_{dc}}{P} = \frac{(1+P)E_{dc}}{2} \tag{5.23}$$

$$C_{d}^{p} = [(a-1)d_{hop}E_{dc}^{p}]/r \tag{5.24}$$

$$G_{GDE}^{comp\,round} = G_{GDE} + C_{d}^{p} \tag{5.25}$$

This section only uses Random Walk as mobility model. For the other movement model, such as SLAW [15], the probabilities of the states on the Marcov Chain take different values. But the analysis and computation process are the same, so we do not present the detail results.

5.5 Realistic Model of Transmission Range

In the previous sections, the transmission range uses the unit disk model. In real applications, the transmission range is not a circle disk, and has rather different shapes according to the application conditions. In this section, we present a compensation strategy to allow *GDE* algorithm work with irregular transmission ranges.

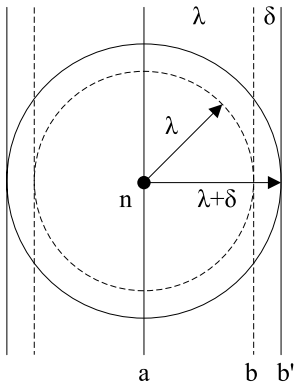


Figure 5.5: New width of the hop-count gradient wave under the influence of irregular transmission range.

We suppose the transmission distance of the node does not take a constant value r, but has a Normal distribution. To create the hop-count gradient, each node discovers the largest gradient value in its transmission range. So the hop-count gradient value of each node depends on the furthest transmission range the node can has. If the transmission distance follows a Normal

distribution $N(\lambda, \delta^2)$, the mean value of the transmission distance λ is not the furthest distance a node can transmit. We use the average furthest transmission distance $\lambda + \delta$ as the new transmission radius. In the formulas of the previous sections, we use the mean value λ as the width of the wave of the hop-count gradient to compute the gradient value. For every wave of the hop-count gradient, the distance is under-estimated with a value of δ.

As shown in Figure 5.5, the transmission range of node n follows Normal distribution $N(\lambda, \delta^2)$. Suppose node n is on the border of the hop-count gradient wave a. The next hop-count wave border changes from b to $b^{'}$, because of the Normal distribution of transmission distance. If the node is in between gradient border a and b, then the under-estimated distance caused by the previous hop-count gradient waves D_p can be computed as Formula 5.26, in which λ is the average transmission radius of each node and δ is the standard deviation of the transmission radius. If the node is between the gradient border b and $b^{'}$, then the under-estimation distance caused by its own hop-count gradient wave can be computed as Formula 5.27. The computation of the gradient value is given in Equation 5.28.

$$D_p = (G_{HCG} - 1)\lambda \tag{5.26}$$

$$D_l = \frac{\frac{\delta^2}{2}}{(\lambda + \delta)} = \frac{\delta^2}{2(\lambda + \delta)} \tag{5.27}$$

$$G_{GDE}^{comp\,trans} = G_{GDE} + [D_p + D_l]/r \tag{5.28}$$

This thesis is mainly about the theoretical research on the algorithm for distance estimation for the large-scale sensor networks. So here we do not make much discussion and analysis on the shape of the transmission range of sensor nodes in various indoor and outdoor deployments. The method provided here can be extended to compensate for other kinds of the irregular shapes of the transmission range.

5.6 Integrated Parameters

For real deployments, the node distribution is not uniform. Furthermore, node speed, the neighborhood interaction and the communication periods may vary. Therefore, we introduce an algorithm that can compute the gradient value for nodes within various deployment conditions as shown by Algorithm 4.

Each node communicates with parts of the neighbors to computes its own gradient values as shown in Section 4, and propagates gradient information

Algorithm 4: Compute Gradient by Propagation

1: **select** $\{SN\}$
2: **compute** D_{hop}, E_{sc}, E_{dc}^{p}
3: $E_c = (1 - D_{hop}) + E_{sc} + E_{dc}^{p}$
4: **for each element** $\{SN\}$
5: **if** $G_{HCG\_i} = (G_{HCG} - 1)$
6: **sum** $(PE_{c\_i} + E_{c\_i})$
7: **end if**
8: **end for**
9: $PE_c =$**average**$(PE_{c\_i} + E_{c\_i})$
10: $G_{GDE}^{adp} = G_{SMG} - PE_c$

to the nodes belonging to the next hop-count gradient wave. The propagated gradient information includes average values of the gradient reduced width value Δ, the static compensation value E_{sc} and the dynamic compensation value E_{dc}. Each node uses the received width reduction values and compensation values to compute its own gradient value.

For Algorithm 4, G_{HCG} is the hop-count gradient value, $\{SN\}$ is the subset of the neighbors to communicate with, D_{hop} is the real one hop distance, E_{sc} is the compensation for the static case which equals Δr as shown in Equation 4.17, E_{dc}^{p} is the compensation for the mobile case given by Equation 5.23. E_c is the sum of reduced width $(1 - D_{hop})$, E_{sc}, and E_{dc}. PE_c is the sum of E_c for the nodes in all the previous hop-count gradients. PE_{c_i}, E_{c_i} and G_{HCG_i} are the PE_c, E_c and G_{HCG} values of node i in the neighbor subset $\{SN\}$. G_{SMG} is the gradient value computed using the simple smooth algorithm introduced in Section 3.2 and G_{GDE}^{adp} is the final computed gradient value.

Chapter 6

GDE Algorithm Analysis

To validate our algorithms, we run simulations in Matlab. The nodes are placed randomly in a square area of 1000×1000 meters. The transmission range r of each node is 80 meters. The node density n_l is the average number of nodes within the transmission range of each node. The seed nodes are placed on the left border of the square. The resulting hop-count gradient looks like parallel waves. For the dynamic case, each node moves for a certain distance at every second. We run experiment 20 times for each testing point. Each node makes a distance estimation from itself to the line of seed nodes. The error for each node is computed as $\varphi = \frac{|\xi - \theta|}{r}$, in which ξ is the estimated distance to the line of seed nodes using different algorithms, θ is the real distance towards the line of seed nodes, r is the transmission range of each node and φ is the error normalized to the transmission range r. The hop-count gradient (HCG) and smoothed gradient (SMG) algorithms compute the gradient values G_{HCG} and G_{SMG}, and the estimated distance is computed by $\xi_{HCG} = G_{HCG} \cdot d_{hop}$ and $\xi_{SMG} = G_{SMG} \cdot d_{hop}$ respectively. The gradient-based distance estimation (GDE) algorithm uses algorithms presented in the previous sections to compute the gradient value G_{GDE}, and the estimated distance is computed by $\xi_{GDE} = G_{GDE} \cdot r$.

6.1 Node Density

With the first set of tests we want to find what are the effect of node density and network diameter to the accuracy of the algorithms. All nodes are static and uniformly distributed, and broadcast locally to the neighbors.

The qualitative analysis about the average absolute error of the distance estimation of *HCG*, *SMG* and *GDE* algorithms is presented in Figure 6.1 for various node densities and network diameters. On the x axis we see the node density and, on the y axis the network diameter. The z coordinate is the average absolute error. The results show that the *HCG* has almost no decrease of the error after node density larger than 15. The distance estim-

ation error using *SMG* and *GDE* algorithms decreases when node density increases. The error of *GDE* is always smaller than the one of *SMG*. When node density is constant, and the network diameter increases, the errors of both *SMG* and *GDE* have small increase but by a small factor. This is mainly caused by the cumulative effect of the distance estimation error of the nodes in the previous hop-count waves. Still, the error of *GDE* is much smaller than the error of *HCG*. Finally, the increasing error rate of *GDE* is smaller than the one of *SMG*.

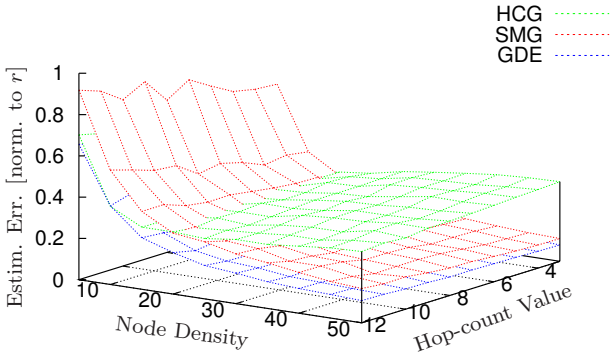


Figure 6.1: Absolute error for various node densities and hop-count gradient values.

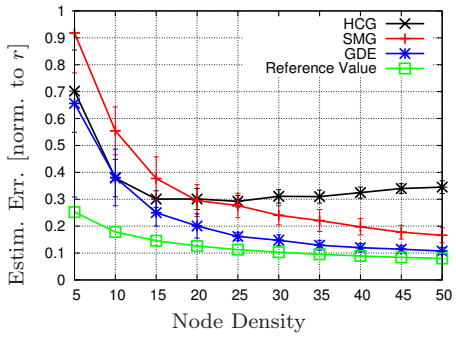


Figure 6.2: Absolute error for various node densities.

The quantitative analysis about the average absolute error of the three algorithms is presented in Figure 6.2 for various node densities. The network diameter is 12. The *GDE* method reduces the average error of *HCG* and *SMG* significantly for node density higher than 15. The standard deviation of the error for both *SMG* and *GDE* decrease when the average node density increases. The standard deviation of the estimation error for *GDE* is always

smaller than the one of *SMG* for node density higher than 15. For an average node density 30 the average estimation error for *HCG*, *SMG* and *GDE* are 0.31, 0.24 and 0.15. While the standard deviation is 0.025, 0.036 and 0.020 respectively. Figure 6.2 also shows out the error reference value according to Formula 4.29. As the node density increases, the error of *GDE* algorithm tends closer to the error reference value.

6.2 Node Speed

The second set of experiments is intended to find the effects of various node speed levels and node densities to the accuracy of the algorithms. Nodes are uniformly distributed, and broadcast to the neighbors. Their movements follow the Random Walk model, and the network diameter is 12.

The qualitative analysis about the average absolute error of the distance estimation of *HCG*, *SMG* and *GDE* algorithms is presented in Figure 6.3 for various node speeds and node densities. The x axis shows the node density, the y axis the node speed, and the z axis the average absolute error. On the y axis, the speed is increased in step of 2 m/s. The estimation error of *GDE* decreases as node density increases. On the other hand, it increases slightly when the average node speed increases. Also, when the node speed increases, the error of *HCG* and *SMG* increases also. For node speeds higher than 8 m/s, the error of *SMG* increases greatly and is even larger than the one of *HCG*. This is mainly caused by the fact that *SMG* is designed for the static environment and it does not consider the influence of mobile nodes.

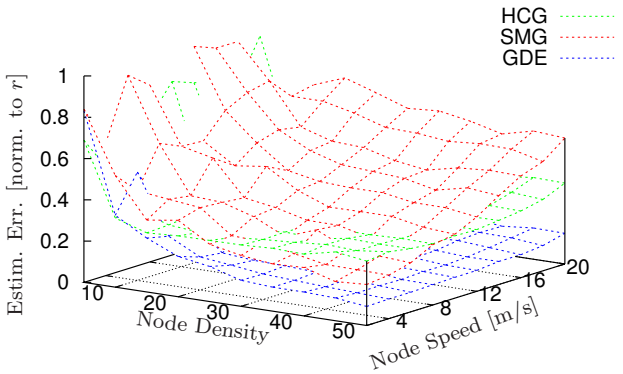


Figure 6.3: Absolute error for various node densities and node speeds.

The quantitative analysis about the average absolute error of the three algorithms is presented in Figure 6.4 for various node speeds. Node density is set to 50. The network diameter is 12. As the speed increases, the error

of *GDE* increases slightly, but *SMG* increases significantly. When the node speed is 20 m/s, the average error of distance estimation by *GDE* is 0.148, while the error levels for *HCG* and *SMG* are 0.391, 0.610 respectively. The standard deviation for *HCG*, *SMG* and *GDE* are 0.011, 0.019 and 0.010 respectively. The test results show that the *GDE* algorithm has a good performance in a dynamic environment. *HCG* and *SMG* are not suitable in a dynamic environment especially at high node speed.

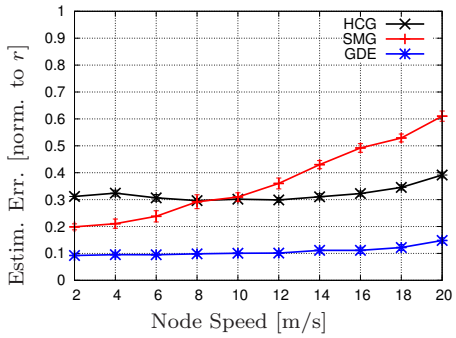


Figure 6.4: Absolute error for various node speed levels.

We also test using another kind of movement model, SLAW [15]. Figure 6.5 shows the error of the three algorithms for various node densities. In the experiment, all nodes are uniformly distributed in the area, the movement distance per second is 1/40 of the transmission radius. *GDE* takes the best performance with node density 15. The error of *GDE* decreases from 0.286 of node density 20 to 0.092 of node density 50. The standard deviation of *GDE* generally is also smaller than *HCG* and *SMG*. The test results show that the *GDE* algorithm can also work well in a dynamic environment with SLAW movement model.

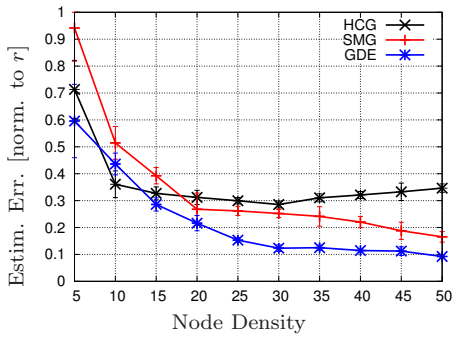


Figure 6.5: Absolute error for various node densities with SLAW movement model.

6.3 Circle Gradient Waves

In the real application, usually only a small number of seed nodes can be deployed in the area. In such condition, for each seed node, the gradient created becomes a circle. As shown in Figure 6.6, a seed node is fixed at the position (0, 0). It can be seen that the gradient shape becomes circular arcs instead of parallel lines. In this section, we test whether the *GDE* algorithm exhibits good performance given that there is only one seed node. At the same time, as shown in the Section 5.1, Algorithm 3 provides a method to take advantage of the characteristic of circle gradient waves. We also want to know whether Algorithm 3 can improve the accuracy of the *GDE* algorithm.

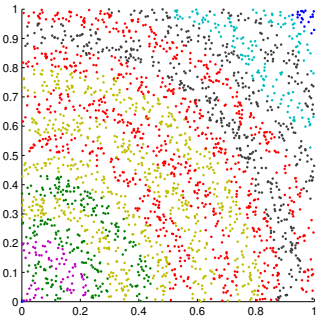


Figure 6.6: Hop-count gradient waves with one seed node at position (0, 0).

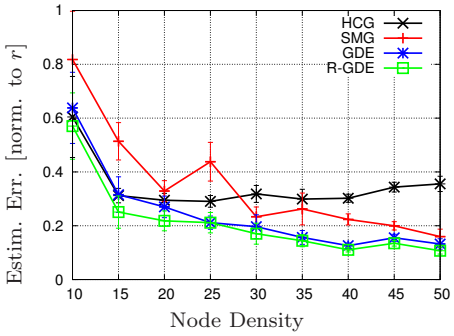


Figure 6.7: Absolute error for various node densities.

In the experiment, only one seed node is fixed at position (0, 0). All the nodes are uniformly distributed with Random Walk model, and the movement distance per second is 1/40 of the transmission radius r. Figure 6.7 shows the testing result. It can be seen that as the node density increases from 15 to 50, the error of *GDE* decreases from 0.315 to 0.132. *HCG* almost has no improvement as the node density increases, and *SMG* only improves from 0.514 to 0.159. At the same time, it can be found that *R-GDE*, which

represents the *GDE* algorithm using Algorithm 3 in the Section 5.1, has the best result accuracy from node density 10 to 50. This means that Algorithm 3 in the Section 5.1 can increase the accuracy of distance estimation for the circle gradient shape, although the improvement is not that large. Therefore, *GDE* can work well even if there are only a small number of seed nodes. *GDE* algorithm not only can make distance estimation to a line of seed nodes, but also can make distance estimation to only one seed node.

6.4 Spatial Node Distribution

In this section, we check the influence of different spatial node distribution to the accuracy of the algorithms. Nodes follow again a Random Walk mobility pattern with speed of 2 m/s. The seed nodes are placed on the left border of the square. This time the placement of the nodes is not uniformly distributed. Instead, the y coordinates of nodes are uniformly distributed from 0 to 1000 meters, while the x coordinates of nodes are distributed following a normal distribution $|N(0, 0.5) \cdot 1000|$, with maximum value 1000 and minimum value 0. So the field that is closer to the seed nodes has higher node density, and the field that is further to the seed nodes has lower node density. The *GDE* gradient is computed using the algorithm shown in Section 5.6. Figure 6.8 shows the distribution of nodes in the area where the hop-count gradient is depicted by various colors.

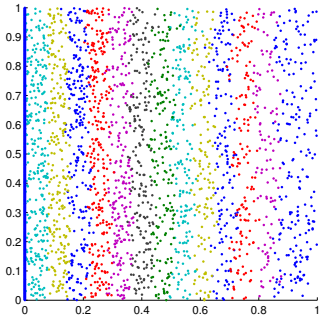


Figure 6.8: Hop-count gradient for Normal node distribution.

In Figure 6.9, on the x axis is depicted the node density, on the y axis is the average distance estimation error. As shown by the graph, the distance estimation errors of *HCG* and *SMG* increase significantly. For *GDE*, the average error of distance estimation is quite low at around 0.089 when the node density is around 80. For a density of 20, the error is 0.150. So *GDE* is also suitable for deployments with non-uniform node distributions.

To further test the *GDE* algorithm, we consider a second experiment, in which the distribution of the nodes in the area is different. All the other

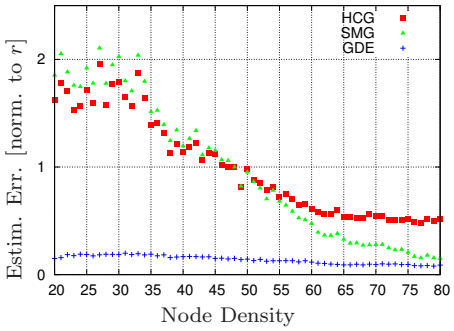


Figure 6.9: Absolute error for Normal node distribution.

parameters are the same as the first spatial node distribution test. In Figure 6.10, generally there are four fields with higher node density. The centers of the four fields are (0.25, 0.25), (0.75, 0.25), (0.75, 0.25), (0.75, 0.75), in which all the coordinates are normalized from 1000 meters to 1 unit. The node distribution follows a Normal distribution with standard deviation 0.1.

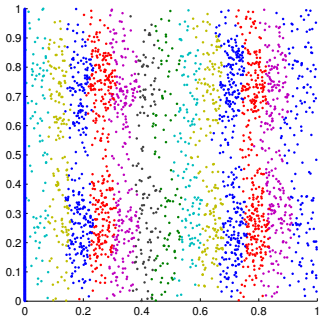


Figure 6.10: Hop-count gradient for Normal node distribution with four high node density fields.

Figure 6.11 shows the testing results. In the figure, the x axis is the distance to the line of seed nodes, y axis is the average distance estimation error. It can be seen that the errors of *HCG* and *SMG* increase significantly from the place that is closer to the seed line to the place that is further to the seed line. For *GDE*, the average error of distance estimation is quite low compared with the other two algorithms. At the distance of 800 meters, the error is only 0.196 of the transmission range. It can also be noticed that generally there are two peak values in the results of *GDE* algorithm. This is mainly because that, along the dimension from the place that is closer to the seed line to the place that is further to the seed line, there are two fields with Normal distribution nodes. So *GDE* shows good performance given non-uniform spatial node distributions.

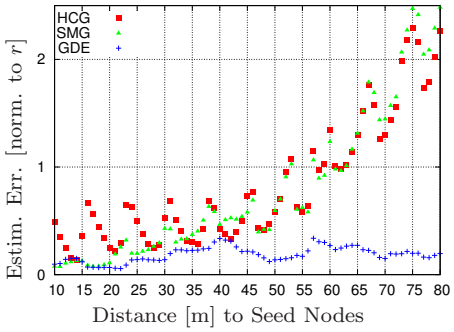


Figure 6.11: Absolute error for Normal node distribution with four high node density fields.

6.5 Multicast Percentage

For wireless systems where nodes follow duty-cycled sleep schedules, nodes that transmit can only be heard by a subset of their neighbors. The percentage of nodes that take part determines the speed of information diffusion. For this experiment, nodes are uniformly distributed with an average density of 50. The network diameter is 12. The nodes take Random Walk with speed of 2 m/s.

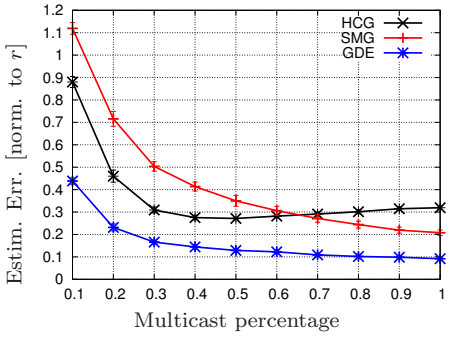


Figure 6.12: Absolute distance estimation error for various multicast percentages.

In Figure 6.12, the x coordinate shows the multicast percentage, while the y coordinate shows the error level. Each node randomly selects a subset of its neighbors at each time step. The multicast percentage varies from 0.1 to 1.0. The error of both *SMG* and *GDE* algorithms decreases significantly. *GDE* always computes a better estimate than *SMG*. This is shown by both the average value and the standard deviation. The estimation error of *HCG* decreases for the multicast percentage between 0.1 and 0.5 and shows almost no improvement afterwards. For an average multicast percentage of 0.6, the error and the standard deviation of *GDE* are 0.122 and 0.009 respectively,

and the values are almost the same as the results of Section 6.1 with node density 30. This experiments shows that even when nodes are not able to talk to all their neighbors, the *GDE* still shows good performance.

6.6 Communication Round Length

The *GDE* algorithm uses a discrete time model, with un-synchronized communication rounds of size P. In this section, we test the influence of the length of the communication round to the accuracy of distance estimation. For this experiment, nodes are uniformly distributed, with an average node density of 50. They use broadcasting to communicate with their neighbors, and the network diameter is 12. The nodes take Random Walk with speed of 2 m/s.

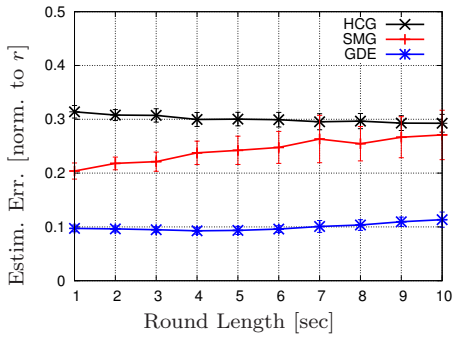


Figure 6.13: Absolute distance estimation error for various round lengths.

In Figure 6.13, the x axis is the round length in seconds, while the y axis is the estimation error. As the round length increases, the error and standard deviation of *SMG* increases significantly. On the other hand, the error and standard deviation of *GDE* shows a much smaller increases from 0.097 and 0.006 to 0.113 and 0.014. The test results show that *GDE* algorithm offers good distance estimation without requiring very frequent communication with neighbor nodes.

6.7 Realistic Model of Transmission Range

In the theoretical analysis, we assume the transmission range of the node is a circle disk. But in the real application, the shape of the transmission area depends on many factors, and the shape can hardly be a circle. In Section 5.5, we present a method to compensate the influence of the transmission range of the node that is not a circle disk. To test our algorithm, we make a simple transmission range model, which assumes that the transmission distance of the node changes according to a Normal distribution. The

transmission circle of the node is sliced to 36 pieces evenly. For each slice, the transmission distance follows a Normal distribution. The mean value equals the theoretical transmission range r. The maximum transmission range is limited to the twice the length of the transmission range. Figure 6.14 shows an example of the transmission range of the node, in which the blue circle is the expected circle transmission range, and the red shape is the realistic model of transmission range.

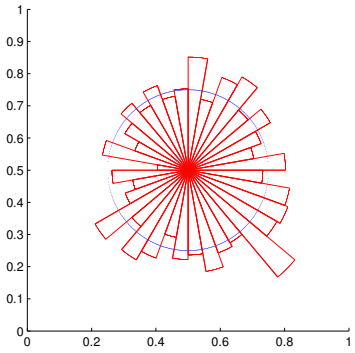


Figure 6.14: A demonstration of the realistic model of transmission range.

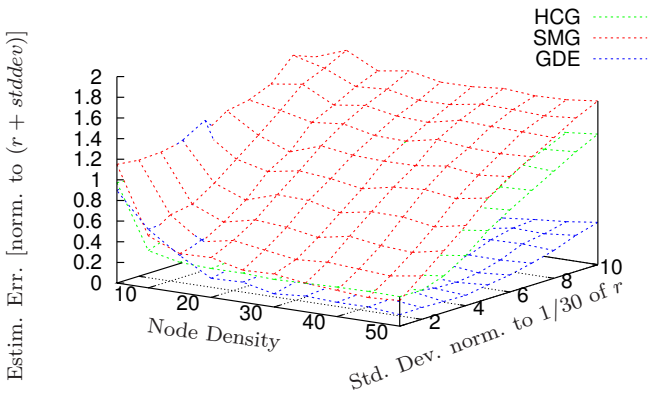


Figure 6.15: Absolute distance estimation error with realistic transmission range model for various node densities and transmission distance standard deviations.

In the experiment, the transmission distance of each range slice for every node changes at every second. The standard deviation *stddev* is normalized to 1/30 of the transmission range. All the nodes are uniformly distributed with Random Walk model, and the movement distance per second is 1/40 of

the transmission radius. Figure 6.15 shows the relation between the error of the distance estimation, the node density and the value of the standard deviation. It can been seen that as the standard deviation increases from 1/30 of the transmission range to 1/3 of the transmission range, the errors of *HCG* and *SMG* increase significantly. And the error of *SMG* is even larger than *HCG*. *GDE* algorithm generally keeps a low error of distance estimation, but still as the standard deviation increases the error also increases.

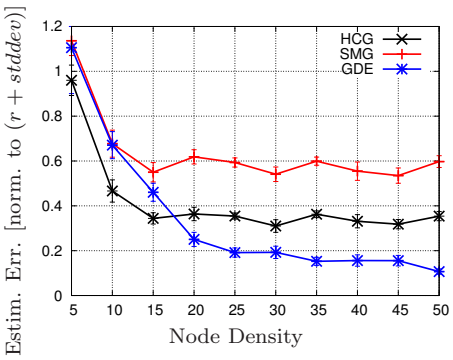


Figure 6.16: Absolute distance estimation error for various node densities with realistic transmission range model of 1/10 transmission distance standard deviation.

Figure 6.16 shows the performance of *GDE* algorithm with standard deviation of 1/10 the transmission range. It can be seen that from node density 20 to 50, the error of *GDE* decreases from 0.250 to 0.107. At the same time, it keeps a small standard deviation from 0.032 to 0.012. The results show that the *GDE* algorithm can work in the condition that the shape of the transmission range is not a circle.

6.8 Adaptive Behavior

In the experiments of the previous sections, we test the *GDE* algorithm under the influence of different environment parameters. In real deployments though, nodes are not uniformly distributed, node speed varies, the local interaction is described by a multicast communication, etc, which means the affects of all these parameters will show on the *GDE* algorithm at the same time. In this section, we test the *GDE* algorithm with all the parameters together.

In the first test, nodes are distributed following a Normal distribution, which is the same as the first experiment in the Section 6.4. The mobility model is Random Walk but node speed follows also a Normal distribution with mean value of 2 m/s. The multicast percentage depends on the local density. Nodes interact with a minimum 15 nodes to a maximum 30 nodes. The random variable of the number of successful transmissions for each

node in each round follows a Binomial distribution. For each neighbor, the probability of making successful communication has a probability of $p_s = (0.2 \cdot n_l + 15)/n_l$, in which n_l is the node density. The communication round length P changes according to the node density as $P = |(p_s \cdot n_l)/15 - 1| \cdot 9 + 1$. In this experiment, we suppose that each node gossips with their neighbors according to the computed fixed round. The node that is selected as the gossiping target should always acknowledge, even if the node is still in the sleeping period of the communication round.

Figure 6.17 shows the experiment results. The x axis shows the distance to the seed nodes while y axis shows the error normalized to the transmission range r. It can be seen that, as the distance to the seed nodes increases, the distance estimation error also increases. When the distance is larger than 300 meters, the estimation error of *SMG* is even larger than that of *HCG*. The estimation error of the two algorithms increases significantly from 300 to 800 meters. As expected, *GDE* generally shows small distance estimation errors, which only increases from 0.120 to 0.456.

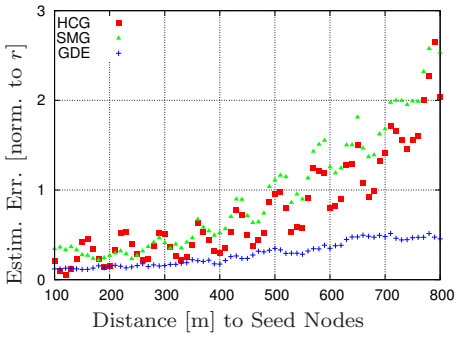


Figure 6.17: Absolute distance estimation error for various parameters.

In the second experiment, the parameters become extreme. The node distribution is the same as the second experiment in the Section 6.4, which generally has four high node density fields in the area. The mobility model is still Random Walk and node speed follows a Normal distribution with mean value of 2 m/s. The multicast percentage depends on the local density. The number of successfully communicated neighbors is within a minimum 15 nodes to a maximum 30 nodes. For each neighbor, the probability of make successful communication has a probability of $p_s = (0.2 \cdot n_l + 15)/n_l + q, (15 \leq n_s \leq 30)$, in which n_l is the number of nodes in the local transmission range and q is a Normal distribution factor with $N(0, 0.1)$. The factor q simulates the random changes in the communication, which randomly affects the successful rate of the communication. Therefore the variable of the number of successful transmissions generally follows a Binomial distribution. The communication round length changes according to the local node density and also follows a Normal distribution, with $N(m, 0.5)$, in which m is the mean

value of the communication round length, $m = |(30 - (0.2 \cdot n_l + 15))/15| \cdot 3$. At the same time, this experiment requires that, if a node is in the sleeping period of the communication round, it does not make gossiping communication to the neighbors and does not make acknowledgement message, even if some other nodes selects it as the communication target. The shape of the transmission range of each node changes for every second, which is the same as in the Section 6.7.

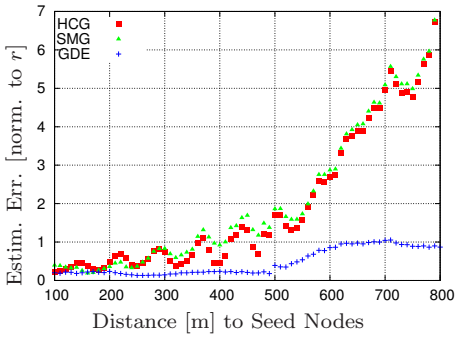


Figure 6.18: Absolute distance estimation error for various parameters.

Figure 6.18 shows the experiment results. It can be seen that, as the distance to the seed nodes increases, the distance estimation error also increases. *GDE* algorithm takes the best performance, whose error grows from 0.200 of 100 meters to 0.868 of 800 meters. It should be noticed that from a real distance of 500 meters to 600 meters, the error takes the maximum increasing rate. At 490 meters, the error is only 0.179, while the error increases to 0.856 at 600 meters. After 600 meters, the increasing rate even takes a minus value. Some further investigation could be made on the large increasing rate to further decrease the distance estimation error. Generally, according to the two experiments, *GDE* has basic adaptive capability, and can make acceptable distance estimation results in the environment with integrated parameters.

Chapter 7

Applications of GDE

Many applications need accurate distance estimation or coordination position information. In this chapter, we provide three kinds of applications where the distance estimation algorithm *GDE* can be used, including cluster center detection, overlay shape construction and routing. Generally, in the applications, the created gradient of *GDE* algorithm can be used to obtain the direction information, and the distance estimation of *GDE* algorithm can be used to create a coordination system.

7.1 Cluster Center Detection

Some algorithms such as ASH [25] can create clusters of nodes in the large-scale networks. One question is how to find the node that is closest to the center of the cluster. The gradient created by *GDE* algorithm can be used to find the center of a cluster.

In this experiment, we suppose all nodes in the area are set to two different fields. The first field is the square in the center of the whole area. All the nodes in the square field form a cluster. The second field is the place that is not in the square. Figure 7.1(a) shows the two fields in the area. In this application, we need to find the node that is closest to the center of the cluster. The center of the cluster is defined as the average value of the coordinate vectors of all nodes in the specified cluster.

In the initialization step, suppose each node knows which cluster it belongs to. If a node can discover some nodes in a different cluster, then itself is set as the seed node. Each node uses the gradient algorithms presented in the previous sections. The nodes with the lowest gradient value are selected as the center of the cluster. Figure 7.1(b), Figure 7.1(c) and Figure 7.1(d) show the selected center nodes, using *HCG*, *SMG* and *GDE* algorithms respectively. In the picture, the bigger nodes are the selected center nodes, and the green cross mark is the expected real center.

To evaluate the performance of cluster center detection by different al-

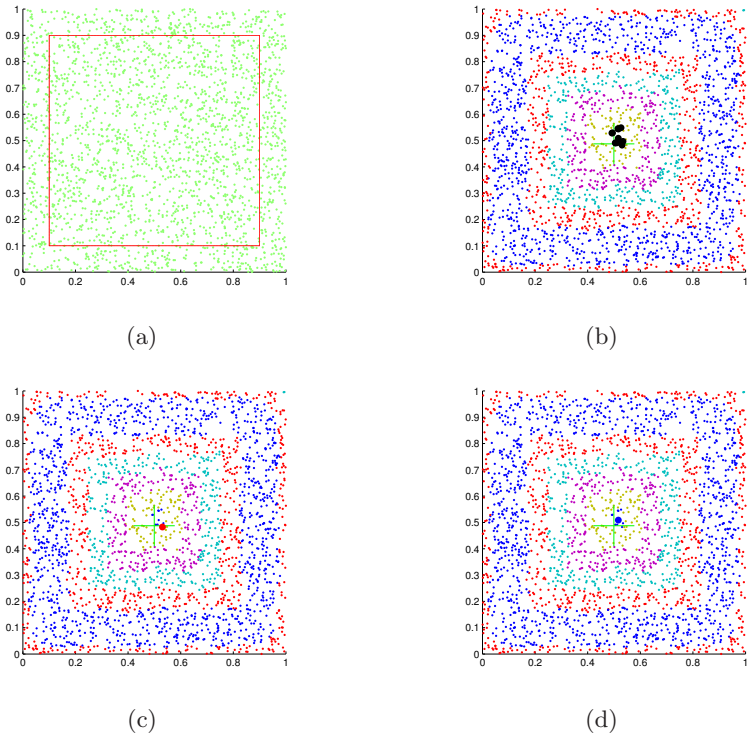


Figure 7.1: (a). Two partitioned fields in the area. (b). Selected center nodes of the cluster with the highest gradient value by algorithm *HCG*. (c). Selected center nodes of the cluster with the highest gradient value by algorithm *SMG*. (d). Selected center nodes of the cluster with the highest gradient value by algorithm *GDE*.

gorithms, we set two kinds of statistical indicators. The first one is the average distance from the selected center nodes to the real center of the cluster. The second one is the number of selected center nodes. In the experiments, the nodes are uniformly distributed in a square area. The side length of the square is normalized to 1. The cluster field is the square with length of 0.8 in the center of the whole area as shown in Figure 7.1(a). The transmission radius of the node is 0.08. The evaluations are made in both static and dynamic environments. In the dynamic testing, the nodes follow a Random Walk model and the movement distance per second is 0.002.

Figure 7.2 shows the average distance to the real center of the cluster using three algorithms. It can be seen that for node density 15 to 25, *SMG* has the best accuracy. And from node density 30 to 50, *GDE* has the best accuracy. At node density 45 and 50 testing point, the error of *HCG* is even smaller than *SMG*. Figure 7.3 shows the number of center nodes selected by the algorithms. It can be found that *SMG* and *GDE* always select less or

equal than 2 nodes as the center nodes, while *HCG* selects more than 10 nodes as the center nodes. In the dynamic environment, the testing results are shown in Figure 7.4 and Figure 7.5. Still *GDE* takes the best accuracy and selects only one or two nodes as the center nodes.

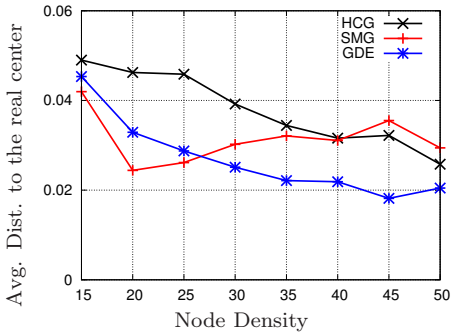


Figure 7.2: Average distance from the selected center nodes to the real center with static nodes.

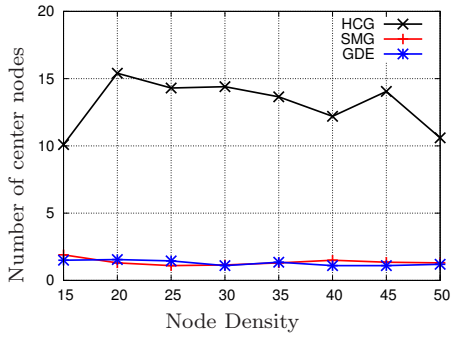


Figure 7.3: Number of selected center nodes.

The reason that *HCG* takes the worst accuracy and selects many nodes as the center nodes is because of its low resolution. The nodes in the same hop-count gradient wave take the same gradient value, then there are many nodes taking the same highest gradient value. *SMG* and *GDE* both have higher resolutions. So both algorithms have better results. Although *GDE* is based on the smoothed gradient created by *SMG*, *GDE* make many compensation computations, which in advance smoothed the gradient of *SMG*. Therefore, from node density 30, *GDE* has much better accuracy than *SMG* in both static and dynamic situations. Because the node density can not be infinity, there is a high probability that there is no node at the position that is computed as the center. So even using *GDE* algorithm, there may be more than one nodes that are selected as the center nodes.

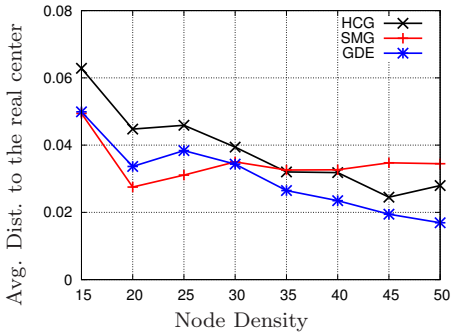


Figure 7.4: Average distance from the selected center nodes to the real center with dynamic nodes.

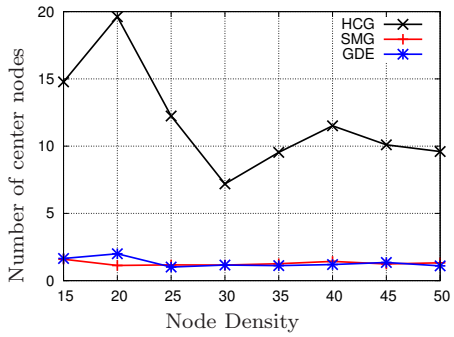


Figure 7.5: Number of selected center nodes.

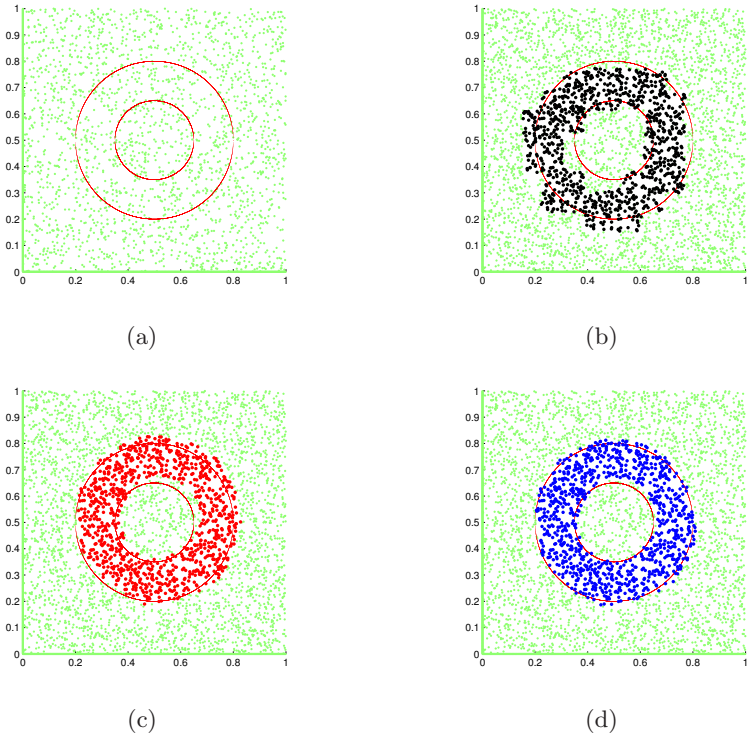
7.2 Overlay Shape Construction

Since using the distance estimation algorithm *GDE*, nodes can compute the distance to some fixed nodes with known positions, then the estimated distances to more than one fixed positions can be used to create a coordination system. According the coordinate values, each node in the network can be marked with required colors, and all the colored nodes can construct a specified overlay shape. The application of programmable self-assembly strategy is shown in [18]. It can assemble autonomous nodes into global shapes, in which the *GDE* algorithm can be used. This section presents the process of creating an overlay shape using distance estimation algorithms.

7.2.1 Constructed by Two Seed Lines

In the experiment, we suppose all nodes are uniformly distributed in a square area. To create a coordination system, two lines of seed nodes are set vertically and horizontally at the border of the square. Each node computes the distances to the two lines, then the two distances can be used as the

coordination values. Using the created coordination system, any shape can be created. Figure 7.6(a) shows the nodes distribution on the material and the shape to be created. Figure 7.6(b), Figure 7.6(c), Figure 7.6(d) show the created shape using *HCG*, *SMG* and *GDE* algorithm respectively.



(a) (b)

(c) (d)

Figure 7.6: (a). The border of the overlay shape to be created. (b). Created shape using *HCG*. (c). Created shape using *SMG*. (d). Created shape using *GDE*.

In order to evaluate the performance of different algorithms to create a specific shape, we set two kinds of statistical indicators. The first one is the number of nodes outside the desired shape. If the node is not in the shape, but it is computed as in the shape using the algorithm, then it is counted as an error node. At the same time, if the node is in the shape, but it is not computed as in the shape using the algorithm, then it is also counted as an error node. And then count the total number of error nodes. The second method is to compute the average distance of all error nodes to the desired shape. The distance is defined as the shortest distance from the node to the border of the shape. In the experiments, the nodes are uniformly distributed in a square area. The length of the side of the square area is normalized 1. The standard shape is constructed by two concentric circles. The center of the circles is (0.5, 0.5). The smaller circle has the radius 0.15. The bigger circle has the radius 0.3. The standard shape is shown in Figure 7.6(a). The

transmission radius of the node is 0.08. We take the experiments in both static and dynamic environments. In the dynamic testing, the nodes follow a Random Walk model and the movement distance per second is 0.003.

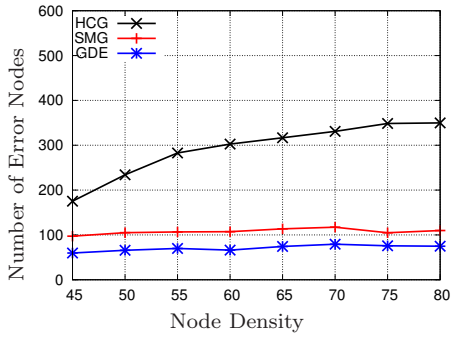


Figure 7.7: Number of error nodes in the shape for static nodes.

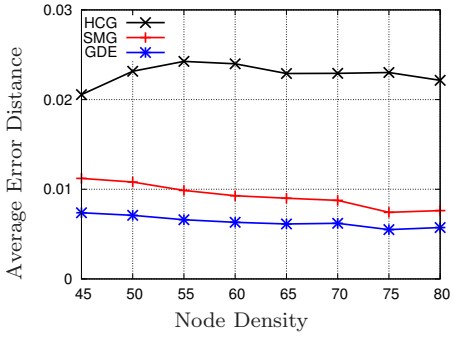


Figure 7.8: Average error distance of error nodes in the shape for static nodes.

In the static environment, as shown in Figure 7.7 and Figure 7.8, *GDE* can not only has the minimum number of error nodes, but also has the minimum distance to the correct shape. From node density 45 to 80, the number of error nodes of *SMG* and *GDE* almost do not increase, But *HCG* increases from 175 to 345. The reason is that *HCG* has very low resolution, which can hardly compute the correct position of each node. At the same time, it can also be found that as the node density increase from 45 to 80, the average error distance of *SMG* and *GDE* decreases a little. This is because if the node density increases the accuracy of distance estimation of the nodes also increases.

GDE also works very well in the dynamic environment as shown by Figure 7.9 and Figure 7.10. The variation trend of the results is almost the same as in the static environment. From node density 45 to 80, the number of error nodes of *GDE* changes from 60 to 75 in the static situation, while

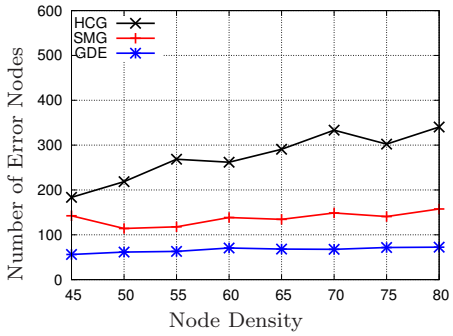


Figure 7.9: Number of error nodes in the shape for mobile nodes.

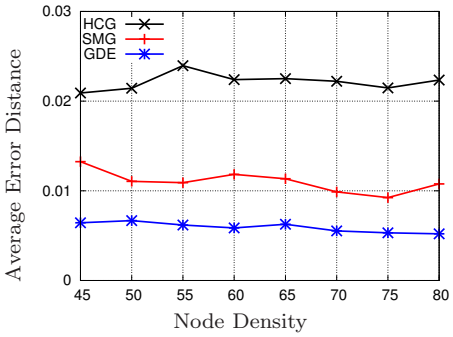


Figure 7.10: Average error distance of error nodes in the shape for mobile nodes.

from 56 to 72 in the dynamic situation. And the average error distance of *GDE* changes from 0.007 to 0.006 in the static situation, while from 0.006 to 0.005 in the dynamic situation. Therefore it can be seen that *GDE* algorithm can be used for overlay shape construction, and works both well in static and dynamic environments.

7.2.2 Constructed by Two Seeds

In the previous discussion and testing, in order to create a coordination system, two lines of seed nodes are placed at the borders of the area. But in real applications, it is not feasible to create a line of seed nodes. Instead, we can only use a small number of nodes as seeds. Even if using a small number of seed nodes, the coordination system can also be created using the *GDE* algorithm. Using the estimated position information, the overlay shape can also be constructed.

To create the coordination system, the minimum number of seed nodes that are needed in a square area is two. And the two seed nodes should be placed at some known positions. Then any node and the two seed nodes can

create a triangle, in which the lengths of the three edges can be computed and the positions of the two seed nodes are known. Using this triangle, the position values of the nodes can be computed.

Suppose side length of the square area is 1. In the experiment, we put two nodes at the corner of the square (0, 0) and (0, 1) as the seed nodes. Then each node computes the distances to the two seed nodes and the coordinate values. As shown in Figure 7.11, node n is in the area with two seed nodes at the corner. Suppose the position of the two seed nodes are known, and the distance of the two seed nodes are l. Use the *GDE* algorithm, the distance from the seed nodes to node n can be computed as d_1 and d_2. Then it is easy to compute the coordinate values of node n using Formula 7.1.

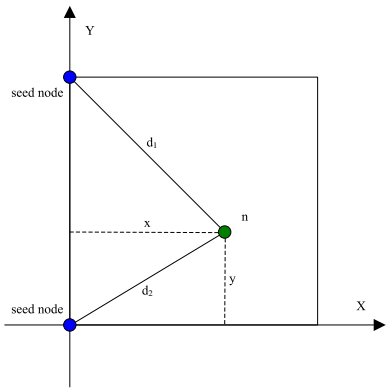


Figure 7.11: Use the distance to the seed nodes to compute the coordinate values of node n.

$$\left.\begin{array}{c} x^2 + (l-y)^2 = d_1^2 \\ x^2 + y^2 = d_2^2 \end{array}\right\} \Rightarrow \left\{\begin{array}{c} x = \sqrt{d_2^2 - y^2} \\ y = \frac{d_2^2 - d_1^2 + l^2}{2l} \end{array}\right. \tag{7.1}$$

With the computed coordinate values, we create the shape shown in Figure 7.6(a). And we still compute the number of error nodes and the average error distance. We only test for all nodes in dynamic. All nodes are uniformly distributed in a square area. The length of the side of the square area is 1. The standard shape is constructed by two concentric circles. The center of the circles is (0.5, 0.5). The smaller circle has the radius 0.15. The bigger circle has the radius 0.3. The transmission radius of the node is 0.08. The nodes follow a Random Walk model and the movement distance per second is 0.002. The two seed nodes are at the corners of the square (0, 0) and (0, 1). Figure 7.12 shows the number of error nodes with only two seed nodes at the corners. Figure 7.13 shows the average error distance to the right shape. It can be seen that as the node density increases from 5 to 80, the number of error nodes of *HCG* increases from 71 to 381. The

number of error nodes of *GDE* only increases from 42 to 75, which is the lowest of all three algorithms. *GDE* also shows the best performance in the average distance error. The error of distance of *GDE* changes from 0.025 of node density 15 to 0.006 of node density 80. The testing results of *GDE*, from node density 45 to 80, are almost the same as the results obtained in the Section 7.2. This means that we can also construct the overlay shape accurately by using only two seed nodes.

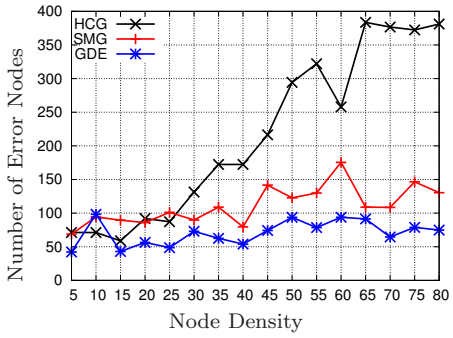


Figure 7.12: Number of error nodes in the shape with two seed nodes.

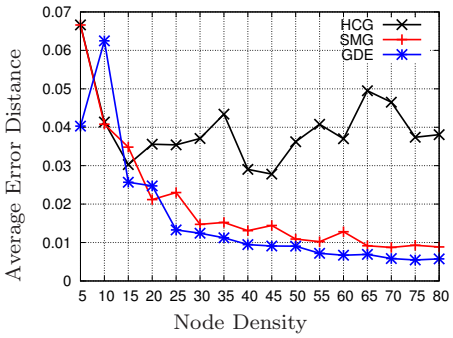
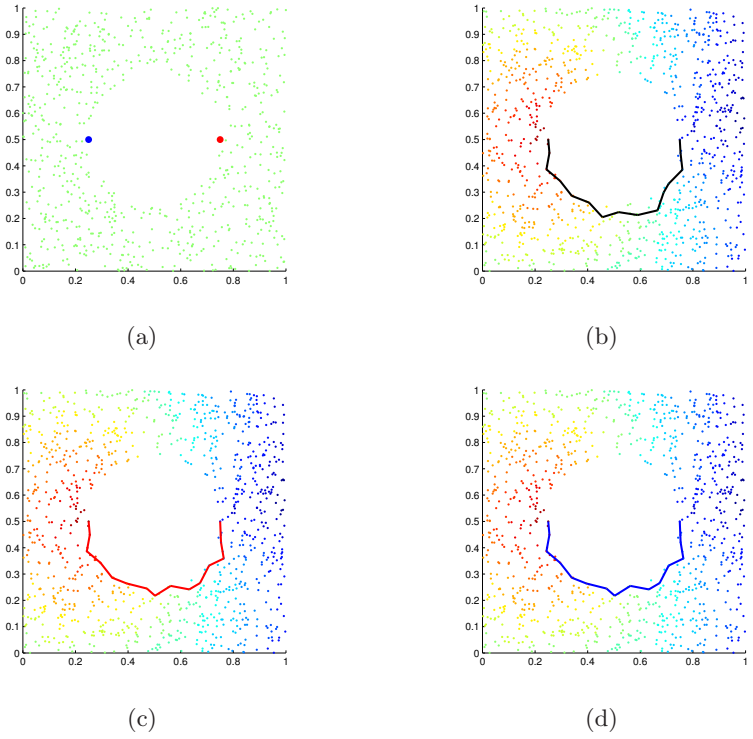


Figure 7.13: Average error distance of error nodes in the shape with two seed nodes.

In real applications, the positions of the seed nodes may be different than the ones in the experiment. The process of calculating the coordinate values of the nodes is the same. It should also be mentioned here that, according to the paper [17], the position the seed nodes can also influence the accuracy of the distance estimation, which could be further researched for the overlay shape construction.

7.3 Routing

The gradient value can be used to make distance estimation. At the same time, the gradient itself also includes some direction information, which can be used as routing information to find the target nodes. This section presents a demonstration of the ability of using the gradient values to make routing from one node to another node in a network.



(a) (b) (c) (d)

Figure 7.14: (a). The source and the destination nodes in the network. (b). Route from the source node to the destination node using *HCG*. (c). Route from the source node to the destination node using *SMG*. (d). Route from the source node to the destination node using *GDE*.

Suppose there are many nodes in the area, and for some reason there is no node in the middle of the area (e.g., in a stadium). Figure 7.14(a) shows the distribution of nodes, in which the blue node is the target node, and the red node is the source node. We use the gradient to create a route from the source node to the target node in the network. First, we set the target node as the seed node, and then a gradient can be created in the network. Any node that wants to send information to the seed node just selects the node with the largest gradient value as the next hop in its own transmission range, and send messages to that node. Because the seed node holds the largest gradient value in the network, the messages will finally reach the seed

node. Figure 7.14(b), Figure 7.14(c), and Figure 7.14(d) show the route from the source node to the target node using *HCG*, *SMG* and *GDE* algorithm respectively. In this experiment, the node density is 20, and all nodes are static. It can be seen that even if there is a hole in the middle of the area, the gradient can be created, and the route from the source node to the target node can be found. It can also be found that *GDE* and *SMG* selects the same route. *HCG* selects a different route but the number of hops is almost the same as the hop number of *GDE* and *SMG*. This is because *SMG* and *GDE* both are based on the hop-count gradient created by *HCG*. *GDE* algorithm is more accurate than *SMG* and *HCG* in the distance estimation. But the routing strategy here is to find the shortest path in the simulation, so the next hop node is always selected among the nodes in the previous hop-count gradient wave. Therefore, there is not that much difference for all three algorithms to select the route. But if the routing strategy demands some position information, the results of the three algorithms may become different. This experiment presents an idea about the application using the gradient-based algorithms for routing. Detailed routing strategy in the application using *GDE* algorithm is a subject of future work.

Chapter 8

Accuracy Improvement of GDE

The *GDE* algorithm presents a method of using simple gossiping to create a gradient and estimate distance between nodes. This chapter provides some improvement methods that can further increase the accuracy of the *GDE* algorithm. According to the testing results, the improvement methods have some progress in the accuracy of the *GDE* algorithm.

8.1 Multilayers

Suppose nodes are uniformly distributed in the area. Each node is randomly given a number s_i $(i = 1, ..., N)$. All the nodes with the same number s_i can construct a new network, in which the nodes are uniformly distributed. The nodes with the same number s_i use the *GDE* algorithm to estimate distances in the new network, and the average distance estimation error is e_i. According to the computation introduced in Section 4.4, the error of the distance estimation using *GDE* algorithm follows a Normal distribution with mean value 0 as the node density tends to infinity. It can be reasoned that the average value of the distance estimation error e_i, $i = 1, ..., N$, tends to 0. Therefore, rearranging the nodes in the area into many layers, and calculating the average value of the distance estimation in each layer may improve the accuracy of the distance estimation results.

As shown in Figure 8.1, each node is randomly assigned to a layer according to a initial constant probability. Then each layer creates a new network but with lower node density. According to the results of the experiments in the previous sections, lower node density produces higher error. Although for each layer, the accuracy becomes lower, the average value of all the distance estimation values of all layers is closer to the correct value.

In the experiment, nodes are dynamic and uniformly distributed in an square with side length 1. The transmission radius of each node is 0.08.

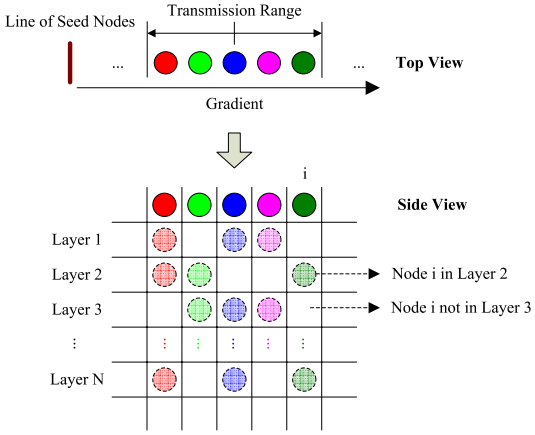


Figure 8.1: Multilayers of nodes in the network.

The nodes follow a Random Walk model and the movement distance per second is 0.002. the network is sliced to 10 layers and for each layer the node density is 25. If the node density of the realistic network is n_l, then the probability that a node is selected in a layer is $25/n_l$. Figure 8.2 shows the testing results of the *GDE* algorithm and the improved *GDE* using the multilayer method. In Figure 8.2, the *Multilayer-GDE* represents the *GDE* algorithm with the multilayer method. It can be seen that there is a little improvement to the basic *GDE* algorithm. From node density 30 to 50, the error of the original *GDE* algorithm is from 0.155 to 0.097, and the error of *GDE* algorithm using multilayer is from 0.149 to 0.098.

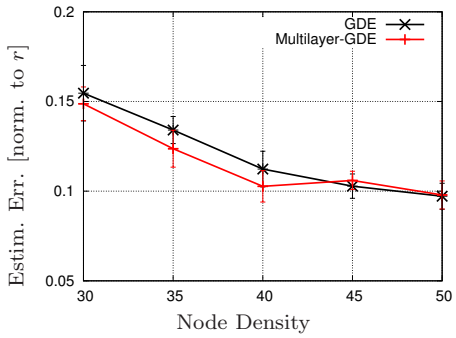


Figure 8.2: Absolute distance estimation error using *GDE* algorithm with multilayers improvement method.

Some questions are not answered in the multilayer method. In the experiment, the number of layers is set to 10, and the node density of each layer has a constant value of 25. Obviously, how to arrange the number of layers and what is the best node density that can be used in each layers could be further investigated.

8.2 Recalibrated Transmission Range

In Section 3.2, the transmission range is supposed to be a square to simplify the computation. But obviously this kind of simplification introduces some distance estimation error. This section presents the computation of recalibrating the transmission range from square to a circle.

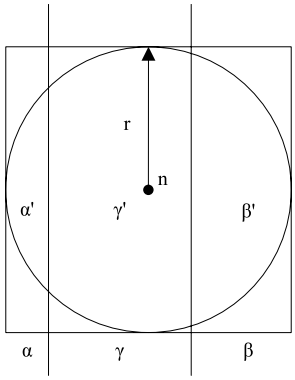


Figure 8.3: The realistic transmission range and the theoretical transmission range.

As shown in Figure 8.3, in the ideal case, node n has a transmission range of a circle. Suppose S_α is the size of the area of α, $S_{\alpha'}$ is the size of the area of $\alpha^{'}$. The size of both area can be computed using Formula 8.1 and Formula 8.2. Then using Formula 8.3 the relation equation between $\alpha^{'}$ and α can be computed.

$$S_\alpha = S_r = \alpha(2r)^2 = 4r^2\alpha \tag{8.1}$$

$$\begin{aligned} S_{\alpha'} = S_s - S_t &= \frac{2\arccos\frac{r-2r\alpha}{r}}{2\pi}\pi r^2 - \frac{2\sqrt{r^2-(r-2r\alpha)^2}(r-2r\alpha)}{2} \\ &\overset{r=1}{=} \arccos(1-2\alpha) - 2(1-2\alpha)\sqrt{\alpha-\alpha^2} \end{aligned} \tag{8.2}$$

$$\alpha^{'} = \frac{S_{\alpha'}}{\pi r^2} = \frac{1}{\pi}\arccos(1-2\alpha) - \frac{2}{\pi}(1-2\alpha)\sqrt{\alpha-\alpha^2} \tag{8.3}$$

In the simulation the transmission area is $\alpha^{'}$ of the circle transmission range, but in the computation, the area is supposed to be α. Using Formula 8.3, the detected area $\alpha^{'}$ can be recomputed to α.

In this experiment, the nodes are dynamic and uniformly distributed in an square with side length 1. The transmission radius of each node is 0.08. The

nodes follow a Random Walk model and the movement distance per second is 0.002. Figure 8.4 shows the improved *GDE* algorithm using the recalibrated transmission range. In Figure 8.4, the *Recalibrated-GDE* represents the *GDE* algorithm with the recalibrated transmission range improvement method. It can be observed that the recalibration can improve the basic *GDE* algorithm. From node density 30 to 50, the error of the original *GDE* algorithm is from 0.155 to 0.097, and the error of the improved *GDE* algorithm using recalibrated transmission range is from 0.152 to 0.091.

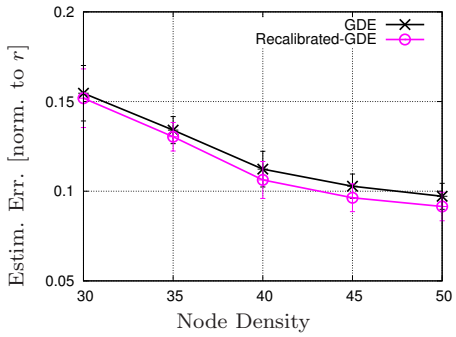


Figure 8.4: Absolute distance estimation error using *GDE* algorithm with recalibrated transmission range improvement method.

However, there is also a problem for this method. This method is based on the computation of the size of the area, which suppose the nodes are uniformly distributed in the area. If the node density is not uniformly distributed, the recalibration computation may be different.

8.3 Fixed Length Calibration

GDE algorithm presents a basic method to create the distance estimation. In a practical application, some additional methods can be introduced to further increase the accuracy of the algorithm. The most used method, which has also been applied in some other localization techniques, such as GPS, is to take a fixed length to recalibrate the final distance estimation results.

In the experiment, we use two lines of seed nodes positioned at known places to calibrate the final distance estimation results. As shown by Figure 8.5, we set two line of seed nodes vertically at the two opposite borders of the square area. The length of the two lines of seed nodes is L. Using the *GDE* distance estimation algorithm, the distance between node n and the two lines of seed nodes can be computed as L_f and L_b, which represent the estimated distance to the forward and backward seed line respectively. The average error of the two distance estimation values L_f and L_b can be computed by Formula 8.4. For each distance estimation value, the result

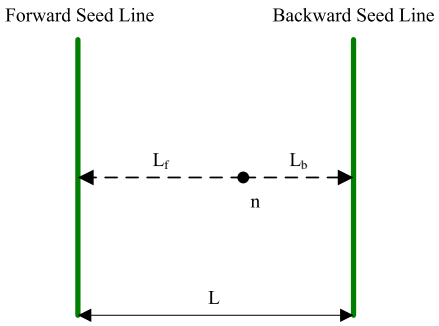


Figure 8.5: Use fixed length L to calibrate the distance estimation result.

can be calibrated using Formula 8.5 and Formula 8.6, in which L_{cf} and L_{cb} are the final two distance estimation results to the forward and backward seed lines.

$$C_L = \frac{L - (L_f + L_b)}{2} \tag{8.4}$$

$$L_{cf} = L_f + C_L \tag{8.5}$$

$$L_{cb} = L_b + C_L \tag{8.6}$$

For this experiment, the nodes are mobile and uniformly distributed in an square with side length 1. The transmission radius of each node is 0.08. The nodes take a Random Walk model and the moving distance per second is 0.002.

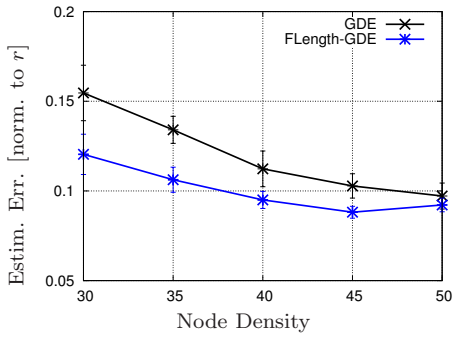


Figure 8.6: Absolute distance estimation error using *GDE* algorithm with fixed length calibration improvement method.

It can be seen from the Figure 8.6 that the fixed length calibration method can improve the accuracy of *GDE* algorithm. In the figure, the *FLength-GDE* represents the *GDE* algorithm with the fixed length calibration improvement method. From node density 30 to 50, the error of the original *GDE* algorithm is from 0.155 to 0.097, and the error of the improved *GDE* algorithm using fixed length calibration is from 0.120 to 0.092.

In a practical application, it is hard to implement a line of seed nodes. To solve the problem, the method introduced in Section 7.2 can be used. Some seed nodes can be placed at some specific positions, then the computed coordinate values can be used to calibrate the results. However, the positions of the seed nodes can influence the accuracy of the distance estimation, and influence the calibration results. So how to place the seed nodes to achieve the best result could be further investigated.

8.4 One Side Repair

There are some characteristics that can be used to further improve the *GDE* algorithm. This section presents a method to further improve the accuracy of *GDE* algorithm.

For example in Figure 8.7, the gradient step is supposed to be 1. The hop-count gradient difference between area γ and α is always 1. To create a hop-count gradient, each node finds the maximum gradient value in its transmission range, and then decreases gradient by one as its own gradient value. If node density of the network is infinity, the hop-count gradient difference between area γ and β is also 1. In reality, node density can not be infinity, which means that there is a probability that there is no node in area β. The hop-count gradient difference between area γ and β becomes 0, which is smaller than the gradient difference between areas γ and α.

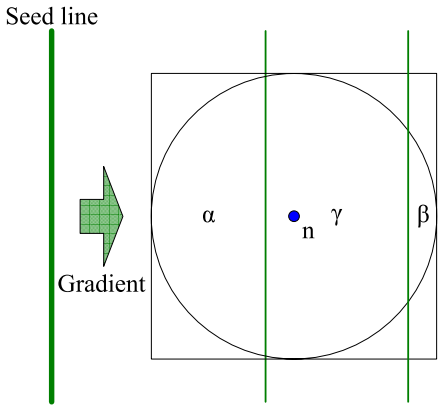


Figure 8.7: Repairing of the gradient by calculating the gradient differences between the two sides.

Suppose the maximum gradient difference from node n to the nodes in area α is G_{fd}. $Max(G_\alpha)$ is the maximum gradient value in the area α. G_n is the hop-count gradient value of node n. Then G_{fd} identically equals to the gradient step value G_{step} as shown in Formula 8.7.

$$G_{fd} = |Max(G_\alpha) - G_n| \equiv G_{step} \tag{8.7}$$

Suppose the maximum gradient difference from node n to the nodes in area β is G_{bd}. $Min(G_\beta)$ is the minimum hop-count gradient value in the area β. Then G_{bd} equals to the gradient step value G_{step} or 0 as shown in Formula 8.8.

$$G_{bd} = |G_n - Min(G_\beta)| = \begin{cases} G_{step}, n_\beta \neq 0 \\ 0, n_\beta = 0 \end{cases} \tag{8.8}$$

$$G_{fd} \geq G_{bd} \tag{8.9}$$

Based on Formula 8.7 and Formula 8.8, it can be seen that the hop-count gradient difference between areas α and γ is always larger or equal than the hop-count gradient difference between areas β and γ as shown in Formula 8.9.

However, the computation of smoothed gradient counts the sum of the hop-count gradient values of all the neighbor nodes. If the sum of the hop-count gradient values in area α is larger than the sum of the hop-count gradient values in area γ, then the final computed smoothed gradient value will bias to the hop-count gradient value in area α. As shown in Formula 8.9, the hop-count gradient difference between areas α and γ is always larger or equal to the hop-count gradient difference between areas β and γ. Therefore, the final computed smoothed gradient value will also bias to the hop-count gradient value in area α. *GDE* algorithm is based on the smoothed gradient, so this phenomenon could introduce an error to the final distance estimation results of *GDE* algorithm.

Suppose node n has gradient value G_{GDE}. G_{GDE}^{min} and G_{GDE}^{max} are the minimum and maximum gradient values in the transmission range of the node. ΔG_α is the maximum gradient value difference between node n and area α, and ΔG_β is the maximum gradient value difference between node n and area β. If $\Delta G_\alpha > \Delta G_\beta$, then the repairing of the gradient value of node n is $[G_{GDE}^{min} + G_{GDE}^{max}]/2$.

In the experiment, the nodes are dynamic and uniformly distributed in an square with side length 1. The transmission radius of each node is 0.08. The nodes follow a Random Walk model and the movement distance per second is 0.002. Figure 8.8 shows the testing results. In the figure, the *Side-GDE*

represents the *GDE* algorithm with the one side repairing improvement method. It can be seen that this improvement algorithm decreases the error of the *GDE* algorithm. From node density 40, the error of this algorithm is better that the original *GDE* algorithm. At node density 40, the error of the algorithm using the one-side repair method is 0.100, and at node density 50, the error of the algorithm using the one-side repair method is 0.083.

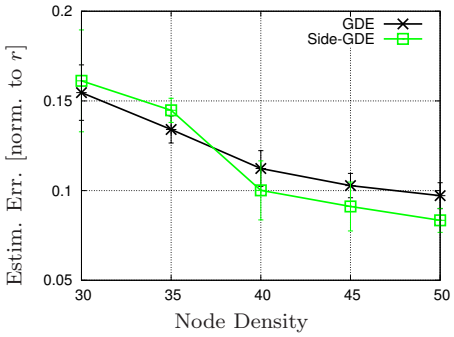


Figure 8.8: Absolute distance estimation error using *GDE* algorithm with one side repair improvement method.

The one side repair method can improve the *GDE* algorithm. Finding out the range that the method can make improvement is a subject of further work. At the same time, we repair the gradient value of node n to $[G^{min}_{GDE} + G^{max}_{GDE}]/2$, which could be further investigated to find out whether there is a more accurate method for repairing.

8.5 Integrated Test

In this section, we combine all the four improvement methods to test the influence to the accuracy of the *GDE* algorithm.

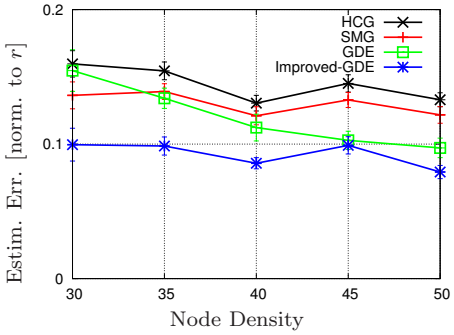


Figure 8.9: Absolute distance estimation error using *GDE* algorithm with all the improvement methods.

In the experiment, the nodes are mobile and uniformly distributed in an square with side length 1. The transmission radius of each node is 0.08. The nodes follow a Random Walk model and the movement distance per second is 0.002. Figure 8.9 shows the testing results of the improved *GDE* algorithm. It can be seen that from node density 30 to 50, the error of the basic *GDE* algorithm changes from 0.155 to 0.097, while the error of the improved *GDE* algorithm changes from 0.010 to 0.079. So the improved *GDE* algorithm has a higher accuracy than the basic *GDE* algorithm.

The testing simply combined all the methods together. Although the four methods all can improve the accuracy of the *GDE* algorithm respectively, some questions raised in each method are not answered. At the same time, how do the improvement methods influence each other is unknown, and how to combine of the methods to achieve the best result is also another unanswered question.

Chapter 9

Conclusions and Future Work

9.1 Conclusions

Distance estimation in wireless networks is of tremendous importance for a lot of application domains. The majority of the algorithms that require position information presume knowledge about node position via systems such as GPS. While this approach is feasible for some application scenarios, for a lot of cases it suffers from frequent unavailability and high costs in terms of energy consumption.

For this reason, in this thesis we introduce a novel distributed algorithm called *GDE* for the estimation of distances in large-scale wireless networks. It is based on a gossiping mechanism to estimate distances between nodes using solely local interactions. The *GDE* algorithm also considers the influence of real application conditions, including node mobility, non-uniform spatial node distribution, multicast percentage, communication round length, transmission range, etc. The model and the theoretical evaluation by means of simulation show that *GDE* succeeds in estimating the distances between nodes in both static and dynamic scenarios with considerably high accuracy, even under the influence of different kinds of environment parameters. At the same time, we introduce some practical applications using *GDE* algorithm, including cluster center detection, overlay shape construction, and routing. The testing to the applications shows that *GDE* can be successfully used with considerably high accuracy. Finally, some further improvement methods are introduced and tested, which can increase the accuracy of the *GDE* algorithm.

9.2 Future Work

GDE can successfully be utilized for distance estimations in large-scale networks. The algorithm is presented from a theoretical point of view, and still has many details that could be further investigated. Generally, there are four main future research details.

Firstly, the thesis already presents methods to compensate for the errors caused by various parameters. The models used in the tests are all based on theoretical conditions. So further investigation should extend the tests to more realistic deployment conditions, such as real human mobility models, real communication models, etc.

Secondly, the accuracy of the *GDE* algorithm could be further improved. Section 8 presents some methods to improve the accuracy of the *GDE* algorithm. There are still some unanswered questions for each method.

Thirdly, the influence of the real transmission range of nodes to the accuracy of *GDE* is an investigation point. There is a very basic foundation for the *GDE* algorithm, which requests the transmission range to be a circle or at least to be generally symmetric. This can only happen in an ideal environment. Section 5.5 presents a method to compensate the influence of the irregular transmission range. The mathematical model of the transmission range still assumes that the average values of the transmission distance on each direction are the same. In the real application, such as in the buildings, the shape of the transmission range is sometimes unpredictable. It is even impossible to guarantee that the transmission range is generally symmetrical. How to make the distance estimation algorithm work better with irregular transmission ranges is a subject of future work.

Finally, it is shown that the *GDE* algorithm produces higher accuracy for higher node density. In some practical applications, the network is sparse. Using multi-hop communication instead of gossiping can increase the number of communicated partners of each node, but this may also increase the message complexity and introduce other problems. How to make *GDE* work better in a low node density area could also be further investigated.

Bibliography

[1] A. Awad, T. Frunzke, and F. Dressler. Adaptive distance estimation and localization in wsn using rssi measures. 2007.

[2] K.P. Birman, M. Hayden, O. Ozkasap, Z. Xiao, M. Budiu, and Y. Minsky. Bimodal multicast. *ACM Transactions on Computer Systems (TOCS)*, 17(2):41–88, 1999.

[3] W.P. Chen, J.C. Hou, and L. Sha. Dynamic clustering for acoustic target tracking in wireless sensor networks. *Mobile Computing, IEEE Transactions on*, 3(3):258–271, 2004.

[4] P.T. Eugster, R. Guerraoui, S.B. Handurukande, P. Kouznetsov, and A.M. Kermarrec. Lightweight probabilistic broadcast. *ACM Transactions on Computer Systems (TOCS)*, 21(4):341–374, 2003.

[5] R. Friedman, D. Gavidia, L. Rodrigues, A.C. Viana, and S. Voulgaris. Gossiping on manets: the beauty and the beast. *ACM SIGOPS Operating Systems Review*, 41(5):67–74, 2007.

[6] G. Gupta and M. Younis. Load-balanced clustering of wireless sensor networks. In *Communications, 2003. ICC'03. IEEE International Conference on*, volume 3, pages 1848–1852. IEEE, 2003.

[7] Z.J. Haas, J.Y. Halpern, and L. Li. Gossip-based ad hoc routing. In *INFOCOM 2002. Twenty-First Annual Joint Conference of the IEEE Computer and Communications Societies. Proceedings. IEEE*, volume 3, pages 1707–1716. IEEE, 2002.

[8] T. He and all. Energy-efficient surveillance system using wireless sensor networks. In *Proc. of the 2nd International Conf. on Mobile Systems*. ACM, 2004.

[9] B. Hofmann-Wellenhof, H. Lichtenegger, and J. Collins. Global positioning system. theory and practice. *Global Positioning System. Theory and practice., by Hofmann-Wellenhof, B.; Lichtenegger, H.; Collins, J.. Springer, Wien (Austria), 1993, 347 p., ISBN 3-211-82477-4, Price DM 79.00. ISBN 0-387-82477-4 (USA).*, 1, 1993.

[10] T. Imielinski, S. Viswanathan, and BR Badrinath. Data on air: Organization and access. *Knowledge and Data Engineering, IEEE Transactions on*, 9(3):353–372, 1997.

[11] B. Karp and H.T. Kung. Gpsr: greedy perimeter stateless routing for wireless networks. In *Proc. of the 6th annual Intern. Conf. on Mobile Comp. and Netw.*, pages 243–254. ACM, 2000.

[12] D. Kempe, A. Dobra, and J. Gehrke. Gossip-based computation of aggregate information. 2003.

[13] L. Kleinrock and J. Silvester. Optimum transmission radii for packet radio networks or why six is a magic number. In *Proc. of the IEEE National Telecom. Conference*, volume 4, pages 1–4, 1978.

[14] L. Klingbeil and T. Wark. A wireless sensor network for real-time indoor localisation and motion monitoring. In *Proceedings of the 7th international conference on Information processing in sensor networks*, pages 39–50. IEEE Computer Society, 2008.

[15] K. Lee, S. Hong, S.J. Kim, I. Rhee, and S. Chong. Slaw: A mobility model for human walks. 2009.

[16] M. Li and Y. Liu. Underground structure monitoring with wireless sensor networks. In *Proceedings of the 6th international conference on Information processing in sensor networks*, pages 69–78. ACM, 2007.

[17] R. Nagpal, H. Shrobe, and J. Bachrach. Organizing a global coordinate system from local information on an ad hoc sensor network. In *Information Processing in Sensor Networks*, pages 553–553. Springer, 2003.

[18] R. Nagpal, G.J. Sussman, and H. Abelson. *Programmable self-assembly: constructing global shape using biologically-inspired local interactions and origami mathematics*. Massachusetts Institute of Technology-Artificial Intelligence Laboratory, 2001.

[19] L. Ojeda and J. Borenstein. Personal dead-reckoning system for gps-denied environments. In *Safety, Security and Rescue Robotics, 2007. SSRR 2007. IEEE International Workshop on*, pages 1–6. IEEE, 2007.

[20] J. Park, E.D. Demaine, and S. Teller. Moving-baseline localization. In *Proceedings of the 7th international conference on Information processing in sensor networks*, pages 15–26. IEEE Computer Society, 2008.

[21] S. Pattem, S. Poduri, and B. Krishnamachari. Energy-quality tradeoffs for target tracking in wireless sensor networks. In *Information Processing in Sensor Networks*, pages 553–553. Springer, 2003.

[22] N. Patwari, J.N. Ash, S. Kyperountas, A.O. Hero III, R.L. Moses, and N.S. Correal. Locating the nodes: cooperative localization in wireless sensor networks. *Signal Processing Magazine, IEEE*, 22(4):54–69, 2005.

[23] B. Pittel. On spreading a rumor. *SIAM Journal on Applied Mathematics*, pages 213–223, 1987.

[24] N.B. Priyantha, A. Chakraborty, and H. Balakrishnan. The cricket location-support system. In *Proc. of the 6th Annual Intern. Conf. on Mobile Comp. and Netw.*, pages 32–43. ACM, 2000.

[25] A. Pruteanu, S. Dulman, and K. Langendoen. Ash: Tackling node mobility in large-scale networks. In *Self-Adaptive and Self-Organizing Systems (SASO), 2010 4th IEEE International Conference on*, pages 144–153. IEEE, 2010.

[26] A. Rao and all. Geographic routing without location information. In *Proc. of the 9th Annual Intern. Conference on Mobile Computing and Netw.*, pages 96–108. ACM, 2003.

[27] M. Rudafshani and S. Datta. Localization in wireless sensor networks. In *Proceedings of the 6th international conference on Information processing in sensor networks*, pages 51–60. ACM, 2007.

[28] Zhou Sheng, Shiqian Xie, and Chengyi Pan. *Probability Theory and Mathematical Statistics*. Beijing, China: Higher Education Press, 2002.

[29] R. Stoleru and all. Stardust: a flexible architecture for passive localization in wireless sensor networks. In *Proc. of the 4th Intern. Conf. on Embedded Networked Sensor Sys.*, pages 57–70. ACM, 2006.

[30] P. Van Mieghem. *Performance analysis of communications networks and systems*. Cambridge Univ Pr, 2006.

[31] N. Xu, S. Rangwala, K.K. Chintalapudi, D. Ganesan, A. Broad, R. Govindan, and D. Estrin. A wireless sensor network for structural monitoring. In *Proceedings of the 2nd international conference on Embedded networked sensor systems*, pages 13–24. ACM, 2004.

[32] Z. Yang, M. Li, and Y. Liu. Sea depth measurement with restricted floating sensors. In *Real-Time Systems Symposium, 2007. RTSS 2007. 28th IEEE International*, pages 469–478. IEEE, 2007.

[33] O. Younis and S. Fahmy. Heed: a hybrid, energy-efficient, distributed clustering approach for ad hoc sensor networks. *IEEE Transactions on Mobile Computing*, pages 366–379, 2004.