Computer Engineering Department
Faculty of Engineering
Deanery of Higher Studies
Islamic University – Gaza
Palestine

Intelligent and Distributed Localization of Nodes in Wireless Sensor Networks

Amina Yusif Al-Sallut

Supervisor

Prof. Ibrahim S. I. Abuhaiba

A Thesis Submitted in Partial Fulfillment of the Requirements for the Degree of Master of Science in Computer Engineering

1433H (2012)

Dedication

I dedicate this thesis to……

My brother, God bless his soul…

My great parents, for instilling the importance of hard work and higher education, and their continuous prayers for me...

My dear husband, for his understanding, patience, and great support along the way……..

My sister, brothers, and kids, may you also be motivated and encouraged to reach your dreams……....

Acknowledgment

I feel greatly privileged to express thanks to all the people who helped me to complete the project successfully.

I would like to thank Prof. Ibrahim S. I. Abuhaiba, Professor of Computer Engineering Department at the IUG, for giving me the honor of his supervision, for his great understanding and support, and for providing me encouragement and guidance throughout my work.

Table of Contents

Chapter 1

Chapter 2

Chapter 3

List of Figures

List of Tables

List of Abbreviations

`

APS	Ad-hoc Positioning System
AOA	Angle-Of-Arrival
BMU	Best Matching Unit
DLSOM	Distributed Localization using Self Organizing Maps
DV-HOP	Distance Vector-HOP
GPS	Global Positioning System
MIT	Massachusetts Institute of Technology
MLP	Multi-Layer Perceptron
MCL	Monte Carlo Localization
MCB	Monte Carlo Localization Boxed
MDS-MAP	Multidimensional Scaling Map
PVA	Position-Velocity-Acceleration
RSSI	Received-Signal-Strength-Indicator
RNN	Recurrent Neural Networks
SOM	Self Organizing Map
SOFM	Self Organizing Feature Map
SVM	Support Vector Machine
TDOA	Time-Differential-Of-Arrival
TOA	Time-Of-Arrival
WSN	Wireless Sensor Networks

تحديد ذكي وموزع لمواقع نقاط الاتصال في شبكات الاستشعار اللاسلكية

أمينة يوسف السلوت

ملخص

في شبكات الاستشعار اللاسلكية، اتخذت مسألة تحديد مواقع العقد اللاسلكية مساحة واسعة للبحث. إن معظم التطبيقات بحاجة إلى معرفة مواقع عقد الاستشعار لأسباب تتعلق بالتوجيه الأمثل والسريع للبيانات. في هذا البحث، تم اقتراح خوارزمية موزعة جديدة باستخدام SOMs لتحديد موقع عقدة لاسلكية في شبكة استشعار لاسلكية.

تصنف الخوارزمية المقترحة كخوارزمية خالية من النطاق حيث تستخدم فقط معلومات الاتصال بين العقد دون الحاجة لقياس وقت الوصول أو قوة الإشارة كما تتطلب الخوارزميات المستندة إلى النطاق. وتستفيد الخوارزمية من المعلومات المستقبلة من العقد المجاورة ومن المواقع المعروفة لبعض العقد الثابتة لحساب المواقع التقديرية للعقد. تتكون الخوارزمية من مرحلتين رئيسيتين: مرحلة التهيئة: ويتم فيها حساب المواقع التقديرية الأولية التي تغذي SOMs، ومرحلة التعلم: والتي تستخدم SOMs لحساب المواقع الفعلية لعقد الاستشعار.

باستخدام المعلومات المستقبلة من العقد المجاورة في المرحلة الأولى خفضت الخوارزمية من وقت تعلم SOMs وعدد مرات التكرار للوصول إلى التقارب بشكل ملحوظ. من ناحية أخرى، بدءاً ببيانات حقيقية بدلاً من البيانات العشوائية رفعت الخوارزمية دقة النتائج إلى الحد الأقصى. وعلاوة على ذلك، فإن التنفيذ الموزع للخوارزمية خفف الضغط على العقد اللاسلكية، التي تتميز بقدرات محدودة وطاقة منخفضة.

تم تنفيذ الخوارزمية المقترحة باستخدام برنامج MATLAB وجربت بنشر عدد مختلف من العقد في منطقة محددة مع نطاقات اتصال مختلفة. وتم التحقق من أداء الخوارزمية بعدد من نماذج المحاكاة والتي أحرزت دقة جيدة جداً. وعلاوة على ذلك، فإن الخوارزمية قد أثبتت فعاليتها مع انخفاض متوسط الخطأ وعدد مرات التكرار المطلوبة بالمقارنة مع الخوارزميات الحديثة ذات الصلة.

Intelligent and Distributed Localization of Nodes in Wireless Sensor Networks

Amina Yusif Al-Sallut

Abstract

In wireless sensor networks, the issue of wireless nodes localization has taken a wide area of research. Most applications need to know position of sensor nodes for reasons of optimal and fast data routing. In this research, a new distributed localization algorithm based on Self Organizing Maps (SOMs) is proposed to determine the location of a wireless node in a wireless sensor network.

The proposed algorithm is classified as a range-free algorithm which uses only the connectivity information between nodes without the need to measure the time of arrival or signal strength as the range-based algorithms require. It utilizes the neighborhood information and the well-known anchors' positions to calculate the estimated locations of nodes. Our algorithm is made up of two main stages: the initialization stage, in which the initial estimated locations of nodes are calculated to be fed to the SOMs, and the learning stage, in which SOMs are used to calculate the physical locations of sensor nodes.

By using the neighborhood information at the first stage, the algorithm has reduced the SOM learning time and the number of iterations to the convergence significantly. On the other hand, starting with real beneficial data rather than random data maximized the accuracy of the resulted locations. Furthermore, the distributed implementation of the algorithm highly alleviated the pressure on the wireless nodes, which are characterized with low power and limited capabilities.

The proposed algorithm has been implemented using MATLAB software and experimented by deploying different number of nodes in a specific area with different communication radio ranges. Extensive simulations evidently verified the performance of the algorithm and achieved a very good accuracy. Moreover, the algorithm proved its effectiveness with the low average error and number of iterations needed in comparison with other recent related algorithms.

Keywords

Wireless Sensor Networks, Localization, Self Organizing Maps, Anchor Nodes

Chapter 1

Introduction

1.1 Sensor Networks

Sensor networks are dense wireless networks of small, low-cost sensors, which collect and disseminate environmental data. Wireless sensor networks facilitate monitoring and controlling of physical environments from remote locations with better accuracy [1]. They have applications in a variety of fields such as environmental monitoring, military purposes and gathering sensing information in inhospitable locations [2-5]. Sensor nodes have various energy and computational constraints because of their inexpensive nature and ad-hoc method of deployment. Considerable research has been focused at overcoming these deficiencies through more energy efficient routing, localization algorithms and system design [6, 7]. Figure 1.1 [8] demonstrates a typical wireless sensor network deployed in a specific area.

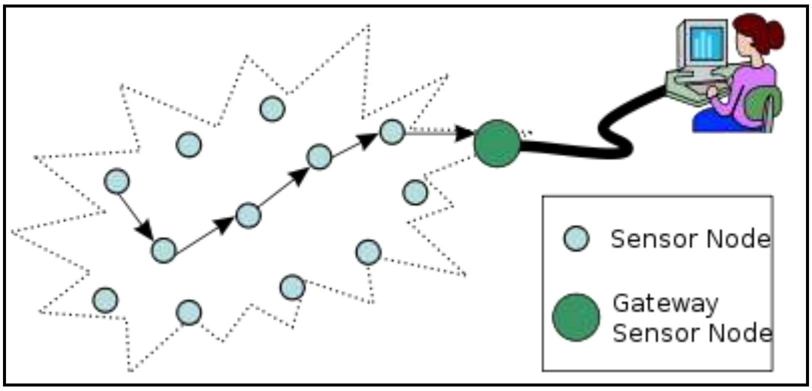


Figure 1.1 A typical Wireless Sensor Network.

1.2 Topic Area

A wireless sensor network [9] is usually a relatively large-scale network of inexpensive, energy efficient devices [10]. For a node in a WSN, awareness of its location and maybe the location of some other nodes is crucial for a successful operation. As a case in point, routing data in sensor networks requires a fine cooperation among nodes in order to use small amount of energy and to deliver data as fast as possible. A node can choose a proper way to the destination, usually a sink node, if it knows geographic location of itself and its neighbor nodes [11].

Furthermore, most applications of sensor networks need to know position of sensor nodes. For instance, a jungle watching WSN [12] must find out and report the location of a probable fire.

Using GPS (Global Positioning System)[13] devices is the simplest way to determine the location of a sensor node; nevertheless, because of some trait of GPS devices which are in contrary with sensor networks demands, using them in all sensor nodes is not justifiable. These traits include relatively high cost, high weight, and debatable accuracy of GPS equipment in some situations. To overcome GPS Limitations, many localization techniques have been developed for sensor networks which don’t depend on the GPS devices merely. In these localization methods, a few nodes, called anchors or seeds, is equipped with GPS devises and help other nodes to determine their position.

Many algorithms have been proposed for localization of static WSNs [14-16]. Nodes in static WSNs do not have movement; in consequences, if a node of these networks could estimate its location once, it won’t have to repeat localization process again. Nodes in mobile WSNs may move by external agent like wind, animal’s movements, stream of a river, or by internal movement agents like wheels and continuous track [17, 18]. Mobility has two contrary effects on localization process. In one hand, as previous works [19] indicates, mobility can help localization of static sensor networks. In that, more nodes can get information from mobile anchor nodes. On the other hand, mobile sensor networks may suffer from rapidly changing situation which leads to less validation time for observed information.

In general, the proposed localization algorithms concentrated on static WSNs due to the high importance of this issue. Under mobility conditions, the static localization algorithm is supposed to be applicable with some superficial modifications and periodic mobility parameters tracking.

1.3 Research Question

Localization is a fundamental task in wireless ad-hoc networks. We consider the problem of locating and orienting a network of unattended sensor nodes that have been deployed in an area at unknown locations. In a location related system, the acquisition of objects' locations is the critical step for the effective and smooth working procedures. The basic concept is to deploy a large number of low-cost, self-powered sensor nodes that acquire and process data. The sensor nodes may include one or more acoustic microphones as well as seismic, magnetic, or imaging sensors. A typical sensor network objective is to detect, track, and classify objects or events in the neighborhood of the network [20].

We consider location estimation in networks where a small proportion of devices, called reference devices or anchors, have a priori information about their coordinates. All devices, regardless of their absolute coordinate knowledge, estimate the range between themselves and their neighboring devices. Such location estimation is called relative location because the range estimates collected are predominantly between pairs of devices of which neither has absolute coordinate knowledge.

We intend to implement a range-free localization algorithm with the consideration of power limitations of sensor nodes, the need for accurate results, and the time required to execute the algorithm and get results.

1.4 Thesis Contribution

In this research, a new enhanced SOM-based localization algorithm is being proposed due to the importance of the localization process and the limitations of the wireless sensor nodes (hardware and power limits). The proposed technique is supposed to be faster and more accurate over the previous similar algorithms.

The main contribution of this research is the utilization of neighboring nodes' information to be used in the first stage of the learning process in the SOM, rather than random initialization. Also, a modified SOM algorithm will be used instead of the classical SOM algorithm (i.e. the updating formula). Furthermore, the localization algorithm will be implemented in a distributed manner rather than centralized.

With these specifications, the algorithm will be faster, more accurate, and significantly decrease the overhead on the sensor nodes.

1.5 Outline of Rest of Thesis

The rest of thesis is organized as follows: Chapter 2 briefly describes some of the most related works that have been done in the wireless nodes localization area. The advantages and drawbacks of every work have been mentioned and discussed. In chapter 3, the required theoretical and experimental materials that have been used in this research are described.

Chapter 4 describes accurately the proposed technique, algorithm steps and the whole system methodology used to localize the wireless nodes in a given wireless network. In chapter 5, the experimentation environment, the experiment simulations and results are demonstrated and analyzed. Comparisons to other related and recent works have been done and also analyzed to verify the proposed algorithm. Finally, the concluding remarks and future work have been listed in chapter 6, conclusion.

Chapter 2

Related Work

2.1 Localization Algorithms

Recently, mobile ad-hoc network localization has received attention from many researchers [21]. Many algorithms and solutions have been presented so far. These algorithms are ranging from simple to complicated schemes, but they can be categorized as range-based and range-free algorithms. Range-free algorithms utilize only connectivity information and the number of hops between nodes. The others utilize the distance measured between nodes by either using the Time-Of-Arrival (TOA) [22], Time-Differential-Of-Arrival (TDOA) [23], Angle-Of-Arrival (AOA) [24], or Received-Signal-Strength-Indicator (RSSI) [25] technologies. However, they usually need extra hardware to achieve such measurement. When calculating the absolute location, most schemes need at least three anchors (nodes that are equipped with Global Positioning System or know their location in advance).

Range-free algorithms are widely used due to the observable advantages over the range-based algorithms especially the conservation in power consumption in wireless devices. Many trends have utilized the artificial neural networks in the localization process. Different types of neural networks have been used in these algorithms. One of the most recent used neural networks are the Self Organizing Maps (SOMs). The use of SOMs showed its effectiveness in the localization process over other algorithms.

2.2 Range-based Localization Algorithms

The traditional ranging methods based on received signal strength (RSSI), time of arrival (TOA), angle of arrival (AOA), time difference of arrival (TDOA), etc. have several shortcomings from the point of view of the sensor networks. RSSI is usually very unpredictable since the received signal power is a complex function of the propagation environment. Hence, radios in sensors will need to be well calibrated otherwise sensors may exhibit significant variation in power to distance mapping.

TOA using acoustic ranging will require an additional ultrasound source. TOA and RSSI are affected by measurement as well as non-line of sight errors. TDOA is not very practical for a distributed implementation. AOA sensing will require either an antenna array or several ultra-sound receivers [20].

The Active Badge Location System [26] is often credited as one of the earliest implementations of an indoor sensor network used to localize a mobile node [27]. Although this system, utilizing infrared, was only capable of localizing the room that the mobile node was located in, many other systems based on this concept have been proposed. The Bat system [28, 29], much like the Active-Badge System, also utilizes a network of sensors. This system features a central controller that emits a query which the mobile node responds to with an ultrasonic pulse. This pulse is picked up by a network of receivers at varying times due to their locations. These times can be used to compute the distances and hence the location of the mobile node. Researchers at MIT (Massachusetts Institute of Technology) have utilized similar concepts from the Bat System in their Cricket sensors, using a more decentralized structure. This system requires less of a support infrastructure than the Bat system. The Cricket Location System [30] uses a hybrid approach consisting of an Extended Kalman filter, Least Square Minimization to reset the Kalman filter, and Outlier Rejection to eliminate bad distance readings. Other researchers at MIT have proposed localization by exploiting properties of robust quadrilaterals to localize an ad-hoc collection of sensors without the use of beacons [31].

It is also possible to localize optically as shown in the HiBall head tracking system [32]. Arrays of LEDs flash synchronously, and cameras capture the position of these LEDs. The system utilizes information about the geometry of the system and computes the position. Localization using signal strength of RF signals has been studied extensively, [33-36] are all examples of methods that were devised using this approach.

Monte Carlo Localization (MCL) was also one of the first practical methods for localization of mobile WSNs. Sequential Monte Carlo method had used for localization of mobile robots previously [37]. Hu and Evans adapted this technique for sensor networks and proposed a practical method [38] for localization of mobile sensor networks. In Sequential Monte Carlo methods, the current state of a system can be obtained by using its current observations and its posterior state. In MCL, the time is divided into discrete intervals. A sensor node moves during a time interval and localizes at the beginning of the next time interval. The main idea of Monte Carlo Localization Boxed (MCB) method proposed by Baggio and Langendoen [39] is to limit the area which the samples are drawn in MCL. Unlike the MCL, MCB uses the information obtained from anchor nodes both before and after generation of samples. This can lead to faster and more efficient sample generation.

All of these approaches complain from the use of range-based (signal strength) and connectivity information, that are hard and slow to be collected, need hardware support in the network nodes, and are rapidly changed especially in mobile networks as well.

2.3 Range-free Localization Algorithms

Some other schemes are range-free and use connectivity information only. One of the first examples of such a technique is the “GPS-Less” [40] positioning system, where nodes use a centroid approach to estimate their position by averaging the coordinates of nearby anchor nodes. DV-HOP (Distance Vector-HOP) is a typical range-free algorithm [41], where anchor nodes flood the network with message beacons that are used by each node to determine the minimal hop count distances. Using an estimate of the average hop length, this information is used to obtain distance values and perform multi-lateration. A similar approach is proposed in [42] and in [43] as an Ad-hoc Positioning System (APS). It uses distance-vector forwarding technique to get the minimum hop count from a node to heard anchors. By using corrections calculated by anchors (average hop-distance between anchors), nodes estimate their location by using lateration (triangulation) method.

Besides DV-HOP, some other algorithms seem to be more complicated, but have better accuracy. The Multidimensional Scaling Map (MDS-MAP) proposed by Shang et al. [44] is an example. MDS-MAP is originated from a data analytical technique by displaying distance-like data in geometrical visualization. It computes the shortest paths between all pairs of nodes to build a distance matrix and then applies the classical Multidimensional Scaling (MDS) to this matrix to retain the first two largest eigenvalue and eigenvector to a 2D relative map. After that, with three given anchors, it transforms the relative map into an absolute map based on anchors' absolute location. There are some variances of MDS-MAP such as centralized method: MDS-MAP(C), and distributed one: MDS-MAP (P). But, in the distributed method, to get the absolute location, nodes need global information about the subnetwork's map that contains at least three anchors.

2.3.1 Localization using Neural Networks

Neural networks have not been used extensively in this area. There has been some research conducted by Chenna et al in [45]. However, Chenna et al, restricted themselves to comparing Recurrent Neural Networks (RNN) to the Kalman Filter. In [46], the authors showed that an MLP (Multi-Layer Perceptron) neural network can be used for localization, and that its performance exceeds that of the Position-Velocity (PV) and Position-Velocity-Acceleration (PVA) variants of the Kalman Filter. Tran and Nguyen [47] proposed a new localization scheme based on Support Vector Machine (SVM). The authors have contributed another machine learning method to the localization problem, and proved the upper bound error of this method.

These first approaches that used neural networks in the localization process in wireless sensor networks showed that it is a promising area and may lead to faster and more accurate localization.

2.3.2 SOM-based Localization Algorithms

Regarding the localization based on Self-Organizing Maps, some researchers have employed SOM directly or with some modification. The SOM projection technique from the input space to the plane defined by the lattice of neurons property has been widely exploited in many applications for data analysis and visualization of large data sets [48, 49]. More recently, SOMs have been used to implement localization schemes for mobile robots in unknown environments [50, 51]. The SOM, initially trained with information collected by on-board sensors during the exploration phase, is then used as a virtual map to translate new sensor readings into grid positions or to recognize different environments. The method presented by Giorgetti [52] employed the classical SOM to the localization. This method uses centralized implementation and requires thousands of learning steps in convergence of network topology. The authors also realize that this method is good for small and medium size networks of up to 100 nodes. S. Asakura et al. proposed a distributed localization scheme [53] based on SOM. Hu and Lee [54] also proposed another version of distributed localization based on SOM. In this work, the authors employed a deduced SOM version [55]. But, this method still needs too many iterations (at least 4000) to make the topology to be converged with a relatively low accuracy.

In another work [56], the authors use SOM to track a mobile robot with the utilization of surrounding environments from readings of sensor data. In the work presented by Ertin and Priddy [57], another version of SOM was used to implement the localization in wireless sensor networks. Their model is based on the assumption that network nodes can sense a common phenomena (e.g. acoustic or seismic) at synchronized timesteps. A further assumption is that the correlation between sensor readings is a function only of the distance between nodes. Under these conditions, sensor readings from all the nodes are first accumulated to form the training set, and then, after the SOM model has been trained, are used to sort the nodes according to their proximity to a set of virtual sensors placed on a regular grid.

These SOM-based algorithms led to acceptable error range and accuracy, but still need a huge number of iterations to approach convergence. Also, they used the classical SOM update algorithm, which may be not suitable in some cases, and applicable only to small networks.

This research will also use the Self Organizing Maps in the localization process, but will try to enhance the existing techniques and try to get benefit from the connectivity and neighboring nodes' information to start the learning stage with meaningful initialization, and thus fasten the process. Moreover, the SOM update function will be modified to be suitable for different network topologies as well as different network sizes.

Chapter 3

Background

3.1 Wireless Sensor Networks

Previously, sensor networks consisted of small number of sensor nodes that were wired to a central processing station. However, nowadays, the focus is more on wireless, distributed, sensing nodes. When the exact location of a particular phenomenon is unknown, distributed sensing allows for closer placement to the phenomenon than a single sensor would permit [58]. Also, in many cases, multiple sensor nodes are required to overcome environmental obstacles like obstructions, line of sight constraints etc. In most cases, the environment to be monitored does not have an existing infrastructure for either energy or communication. It becomes imperative for sensor nodes to survive on small, finite sources of energy and communicate through a wireless communication channel. Another requirement for sensor networks would be distributed processing capability. This is necessary since communication is a major consumer of energy. A centralized system would mean that some of the sensors would need to communicate over long distances that lead to even more energy depletion. Hence, it would be a good idea to process locally as much information as possible in order to minimize the total number of bits transmitted.

3.1.1 Challenges in WSNs

In spite of the diverse applications, sensor networks pose a number of unique technical challenges due to the following factors [20]:

- Ad hoc deployment: Most sensor nodes are deployed in regions, which have no infrastructure at all. A typical way of deployment in a forest would be tossing the sensor nodes from an airplane. In such a situation, it is up to the nodes to identify its connectivity and distribution.
- Unattended operation: In most cases, once deployed, sensor networks have no human intervention. Hence the nodes themselves are responsible for reconfiguration in case of any changes.

- Not tethered: The sensor nodes are not connected to any energy source. There is only a finite source of energy, which must be optimally used for processing and communication. An interesting fact is that communication dominates processing in energy consumption. Thus, in order to make optimal use of energy, communication should be minimized as much as possible.
- Dynamic changes: It is required that a sensor network system be adaptable to changing connectivity (for e.g., due to addition of more nodes, failure of nodes etc.) as well as changing environmental stimuli.

Thus, unlike traditional networks, where the focus is on maximizing channel throughput or minimizing node deployment, the major consideration in a sensor network is to extend the system lifetime as well as the system robustness [59].

3.1.2 Sensor Networks Applications

Sensor networks may consist of many different types of sensors as discussed in [60] such as seismic, low sampling rate magnetic, thermal, visual, infrared, acoustic and radar, which are able to monitor wide variety of ambient conditions that include the following:

- Temperature
- Humidity
- vehicular movement
- lightning condition
- pressure
- soil makeup
- noise levels
- the presence or absence of certain kinds of objects
- mechanical stress levels on attached objects
- The current characteristics such as speed, direction and size of an object.

Sensor nodes can be used for continuous sensing, event detection, event ID, location sensing, and local control of actuators. The concept of micro-sensing and wireless connection of these nodes promises many new application areas. The applications are categorized into military, environment, health, home and other commercial areas. It is possible to expand this classification with more categories such as space exploration, chemical processing and disaster relief. In fact, due to the pervasive nature of micro sensors, sensor networks have the potential to revolutionize the very way we understand and construct complex physical system [61].

3.2 Localization

In sensor networks, nodes are deployed into an unplanned infrastructure where there is no a priori knowledge of location. The problem of estimating spatial-coordinates of the node is referred to as localization. An immediate solution, which comes to mind, is GPS or the Global Positioning System. The different approaches to the localization problem have been studied in [62-64]. However, there are some strong factors against the usage of GPS. For one, GPS can work only outdoors. Secondly, GPS receivers are expensive and not suitable in the construction of small cheap sensor nodes. A third factor is that it cannot work in the presence of any obstruction like dense foliage etc. Thus, sensor nodes would need to have other means of establishing their positions and organizing themselves into a co-ordinate system without relying on an existing infrastructure. Most of the proposed localization techniques today, depend on recursive trilateration/multilateration techniques [59]. One way of considering sensor networks is taking the network to be organized as a hierarchy with the nodes in the upper level being more complex and already knowing their location through some technique (say, through GPS). These nodes then act as beacons by transmitting their position periodically. The nodes, which have not yet inferred their position, listen to broadcasts from these beacons and use the information from beacons with low message loss to calculate its own position. A simple technique would be to calculate its position as the centroid of all the locations it has obtained. This is called as proximity based localization. It is quite possible that all nodes do not have access to the beacons. In this case, the nodes which have obtained their position through proximity based localization themselves, act as beacons to the other nodes. This process is called iterative multilateration. As can be guessed, iterative multilateration leads to accumulation of localization error. Thus, trilateration is a geometric principle which allows us to find a location if its distance from three already-known locations.

The same principle is extended to three-dimensional space. In this case, spheres instead of circles are used and four spheres would be needed. When a localization technique using beacons is used, an important question would be how many initial beacons to deploy. Too many beacons would result in self-interference among the beacons while too less number of beacons would mean that many of the nodes would have to depend on iterative multilateration.

3.2.1 Localization Techniques

Localization can be classified as fine-grained, which refers to the methods based on timing/signal strength and coarse-grained, which refers to the techniques based on proximity to a reference point. In [65], an over-view of the various localization techniques is given.

Examples of fine-grained localization are:

- Timing: The distance between the receiver node and a reference point is determined by the time of flight of the communication signal.
- Signal strength: As a signal propagates, attenuation takes place proportional to the distance traveled. This fact is made use of to calculate the distance.
- Signal pattern matching: In this method, the coverage area is pre-scanned with transmitting signals. A central system assigns a unique signature for each square in the location grid. The system matches a transmitting signal from a mobile transmitter with the pre-constructed database and arrives at the correct location. But pre-generating the database goes against the idea of ad hoc deployment.
- Directionality: Here, the angle of each reference point with respect to the mobile node in some reference frame is used to determine the location.

Examples of coarse-grained localization are proximity based localization as described earlier. [65] Proposes a localization system which is RF-based, receiver-based, ad hoc, responsive, low-energy consuming and adaptive. RF-based transceivers would be more inexpensive and smaller compared to GPS-receivers. Also in an infrastructure less environment, the deployment would be ad hoc and the nodes should be able to adapt themselves to available reference points.

Locating objects in two (e.g., surface of the earth) or three dimensions (e.g., space) from the knowledge of locations of some distinguished nodes, called anchors, has been the central problem in navigation. Anchors can know location of a node from its distances and/or angles. What distinguishes the localization problem in sensor networks from the navigation problem is the following: due to spatial expanse of a sensor network, not every sensor will have the required number of anchors for ranging; to be cost effective, fewer anchors are desired.

3.3 Self Organizing Maps (SOMs)

A self-organizing map (SOM) [66] or self-organizing feature map (SOFM) is a type of artificial neural network [67] that is trained using unsupervised learning to produce a low-dimensional (typically two-dimensional), discretized representation of the input space of the training samples, called a map. Self-organizing maps are different from other artificial neural networks in the sense that they use a neighborhood function to preserve the topological properties of the input space.

This makes SOMs useful for visualizing low-dimensional views of high-dimensional data, akin to multidimensional scaling [68]. The model was first described as an artificial neural network by the Finnish professor Teuvo Kohonen, and is sometimes called a Kohonen map [69].

Self Organizing Map (SOM) by Teuvo Kohonen provides a data visualization technique which helps to understand high dimensional data by reducing the dimensions of data to a map. SOM also represents clustering concept by grouping similar data together. Therefore it can be said that SOM reduces data dimensions and displays similarities among data.

With SOM, clustering is performed by having several units compete for the current object. Once the data have been entered into the system, the network of artificial neurons is trained by providing information about inputs. The weight vector of the unit that is closest to the current object becomes the winning or active unit. During the training stage, the values for the input variables are gradually adjusted in an attempt to preserve neighborhood relationships that exist within the input data set. As it gets closer to the input object, the weights of the winning unit are adjusted as well as its neighbors.

Teuvo Kohonen writes "The SOM is a new, effective software tool for the visualization of high-dimensional data. It converts complex, nonlinear statistical relationships between high-dimensional data items into simple geometric relationships on a low-dimensional display. As it thereby compresses information while preserving the most important topological and metric relationships of the primary data items on the display, it may also be thought to produce some kind of abstractions."

3.3.1 Reducing Data Dimensions and Data Similarity

Unlike other learning technique in neural networks, training a SOM requires no target vector. A SOM learns to classify the training data without any external supervision. Figure 3.1 [70] illustrates a SOM and how an input vector *x* is connected to the nodes (neurons) of the SOM.

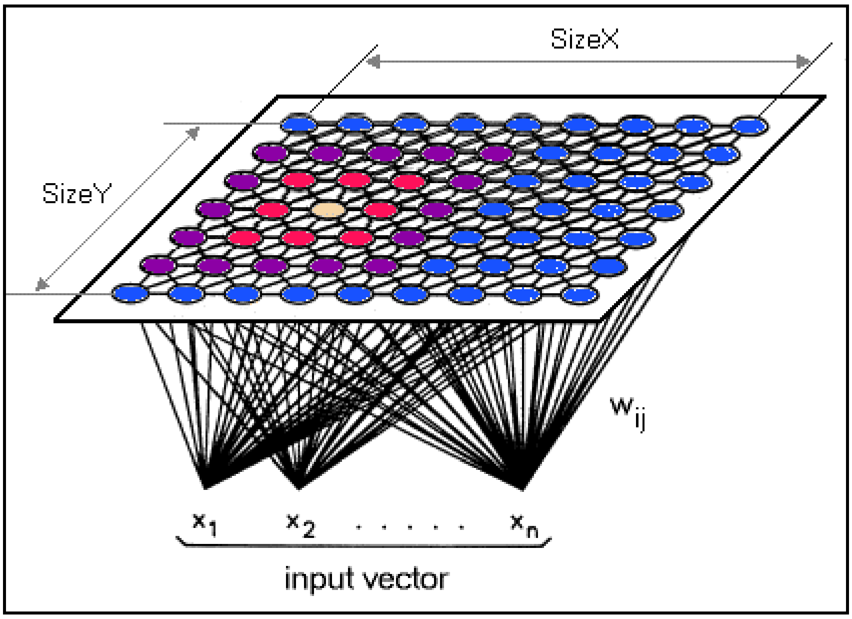


Figure 3.1 A typical SOM with input vector connection.

Every node is connected to the input the same way, and no nodes are connected to each other.

Getting the Best Matching Unit is done by running through all weight vectors and calculating the distance from each weight to the sample vector. The weight with the shortest distance is the winner. There are numerous ways to determine the distance; however, the most commonly used method is the *Euclidean Distance* and/or *Consine Distance*.

3.3.2 SOM Algorithm

Each data from data set recognizes themselves by competing for representation. SOM mapping steps starts from initializing the weight vectors. From there a sample vector is selected randomly and the map of weight vectors is searched to find which weight best represents that sample. Each weight vector has neighboring weights that are close to it. The weight that is chosen is rewarded by being able to become more like that randomly selected sample vector. The neighbors of that weight are also rewarded by being able to become more like the chosen sample vector. From this step the number of neighbors and how much each weight can learn decreases over time. This whole process [70, 71] is repeated a large number of times, usually more than 1000 times.

- In sum, learning occurs in several steps and over many iterations:

1. Each node's weights are initialized.
2. A vector is chosen at random from the set of training data.
3. Every node is examined to calculate which one's weights are most like the input vector. The winning node is commonly known as the Best Matching Unit (*BMU*).
4. Then the neighborhood of the *BMU* is calculated. The amount of neighbors decreases over time.
5. The winning weight is rewarded with becoming more like the sample vector. The neighbors also become more like the sample vector. The closer a node is

to the *BMU*, the more its weights get altered and the farther away the neighbor is from the *BMU*, the less it learns.

6. Repeat step 2 for *N* iterations.

- Calculating the Best Matching Unit:

Calculating the *BMU* is done according to the Euclidean distance among the node's weights (w_1, w_2, …, w_n) and the input vector's values (x_1, x_2, … , x_n). This gives a good measurement of how similar the two sets of data are to each other.

$$dist = \sqrt{\sum_{i=0}^{n}(x_i - w_i)^2} \qquad (3.1)$$

where *n* is the dimension of the weight vectors and the input vector.

- Determining the *BMU* Neighborhood
  - ○ Size of the neighborhood: an *exponential decay* function($\sigma(t)$) is used that shrinks at each iteration until eventually the neighborhood is just the *BMU* itself.

$$\sigma(t) = \sigma_0 \exp\left(\frac{-t}{\lambda}\right) \qquad (3.2)$$

Where *t* is the time step, σ_0 is the initial value and λ is a time constant.

  - ○ Effect of location within the neighborhood: The neighborhood is defined by a gaussian curve so that nodes that are closer are influenced more than farther nodes.

$$\Theta(t) = \exp\left(-\frac{dist^2}{2\sigma^2(t)}\right) \qquad (3.3)$$

where $\Theta(t)$ is the neighborhood function used in the updating formula.

Figure 3.2 [71] illustrates the changes of weights on a given weight vector w_j by applying the neighborhood function.

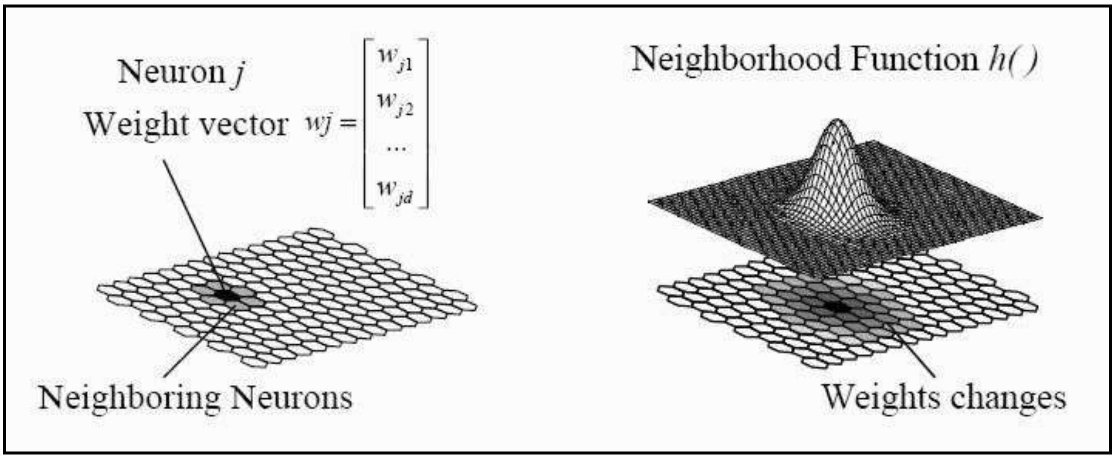

Neuron j
Weight vector $wj = \begin{bmatrix} w_{j1} \\ w_{j2} \\ \ldots \\ w_{jd} \end{bmatrix}$
Neighborhood Function $h(\,)$
Neighboring Neurons
Weights changes

Figure 3.2 The weights changes by applying the neighborhood function.

- Modifying Nodes' Weights

The new weight for a node is the old weight, plus a fraction (L) of the difference between the old weight and the input vector and adjusted (Θ) based on distance from the *BMU*.

$$w(t+1) = w(t) + \Theta(t)L(t)(x(t) - w(t)) \qquad (3.4)$$

The learning rate, L, is also an exponential *decay* function. This ensures that the SOM will converge.

$$L(t) = L_0 \exp\left(-\frac{t}{\lambda}\right) \qquad (3.5)$$

where L_0 is the initial value of the learning rate, λ represents a time constant, and t is the time step.

Chapter 4

Proposed Technique

4.1 The Proposed Algorithm

In this chapter, the proposed algorithm used to localize the nodes' positions in a wireless network will be introduced and described in details.

We named the algorithm (DLSOM), which stands for (Distributed Localization using Self Organizing Maps). The algorithm is divided into two main stages, the initialization stage and the SOM learning and converging stage.

4.1.1 Anchors Number and Placement

The accuracy of the estimated positions is highly affected by the number of anchor nodes and their distribution in the sensor field. Although various algorithms use the location information of anchors differently, the number and placement of anchors affect the accuracies of localization algorithms to a certain extent. Substantial amount of anchors are required to maintain the accuracy for distributed algorithms based on multilateration, in which nodes estimate their positions as the average of the received positions from anchors and neighbors. Theoretically, more anchors bring higher location accuracy. However, too many anchors cause high energy consumption and calculation complexity.

Many previous studies found that the optimal number of anchors to be selected in most distributions of wireless sensor networks ranges from 3 to 6. In multilateration-based algorithms, like our algorithm, using 4 anchors gives satisfying accuracy with very slight difference of using more anchors, taking into account the calculation complexity and energy consumption.

In [72], the number of anchors is optimized through a simulation with Matlab. The localization algorithm was simulated with different number of anchors (3, 4, 6, 8, 10, 20, and 50). The correction quality of multilateration stagnates when more than six anchors are used. To keep low complexity and energy consumption, four is chosen as the optimal number of anchors. Also, in the experiments of [73], they tried from 3 to 10 anchors on the anisotropic networks and found that 4 anchors usually give the best result. Another benefit of using just 4 anchors for multilateration is that the communication cost is much lower than using many anchors. The delay in getting the distance information is also smaller. The information locality enables the method to scale to large-size networks.

For the selection of anchors' positions, also many studies showed that the performance is better when anchors are uniformly distributed along the perimeter of the network. The nodes with high correlations with other nodes should be selected [74-76]. In [73], experiments showed that placing 4 anchors randomly gives slightly worse solutions than using the 4 outer anchors (at the network perimeter). Using the 4 inner anchors (at the four centers) gives the worst solution.

In this research, we tried to select the anchors at the perimeter and at the four centers of the network. Extensive simulations using the two distributions showed that selecting the anchors at the network perimeter gives better accuracy and less estimation error.

This can be interpreted as: the nodes at the perimeter of the network are more correlated to the unknown nodes. Moreover, the neighbors of these anchors can estimate their locations more accurately based on the anchor position with regard to the network (as will be explained later in the algorithm methodology and equations). Thus, four anchors distributed at the network perimeter are selected in the simulations of our algorithm.

4.1.2 The Initialization Stage

In this stage, the wireless network nodes try to estimate their locations using the well-known positions of the anchor nodes. The anchor nodes are very small number of nodes that know their locations in advance (equipped with GPS).

Each node in the network can estimate its initial location according to the following steps:

1- Each anchor node broadcasts its location to only the neighbors of that node. An example is illustrated in Figure 4.1:

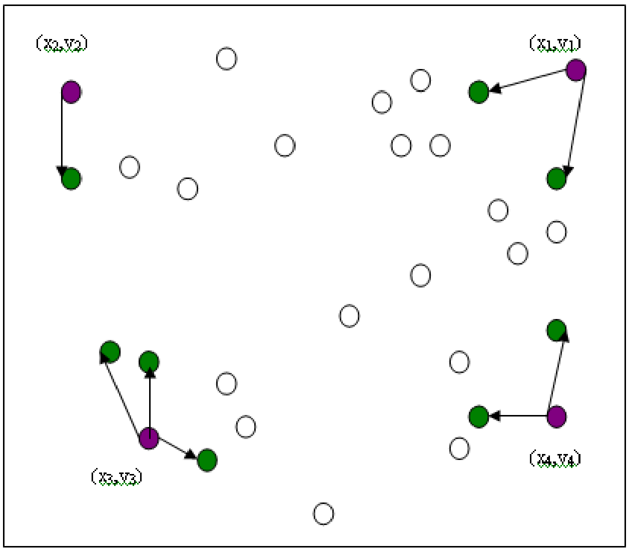


Figure 4.1 Anchors broadcast locations to neighbors.

2- According to the location of that node with regard to the network topology (top-right, top-left, bottom-right, bottom-left), the neighboring nodes can estimate their initial locations, as illustrated in Figure 4.2.

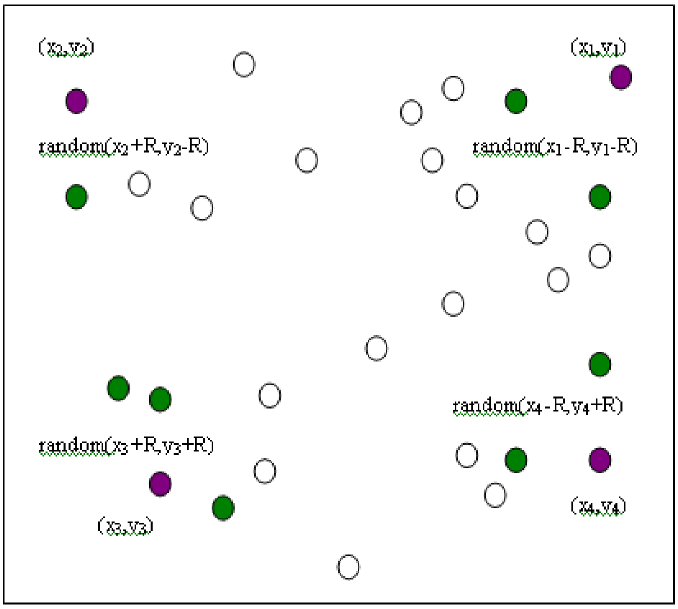


Figure 4.2 Anchors' Neighbors estimated locations.

3- Now, the neighboring nodes also broadcast their estimated locations resulted from step 2 to their neighbors inside the network as Figure 4.3 illustrates.

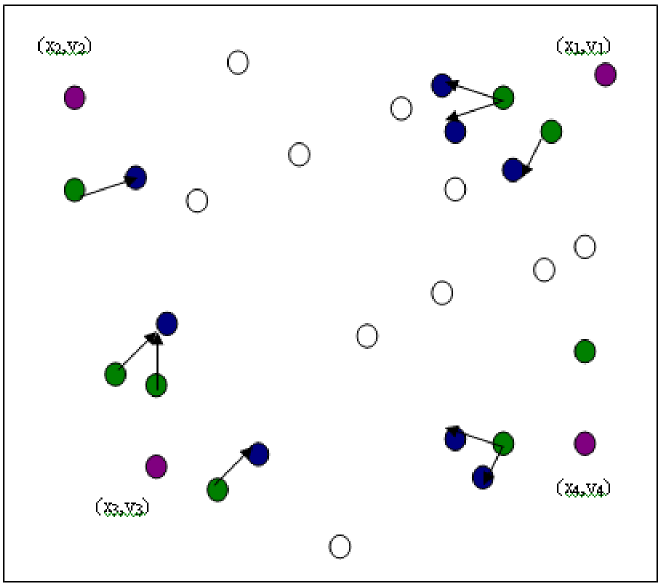


Figure 4.3 Neighbors broadcast their estimated location.

4- As shown in Figure 4.4, each node can estimate its location by averaging the estimated locations received from its neighbors.

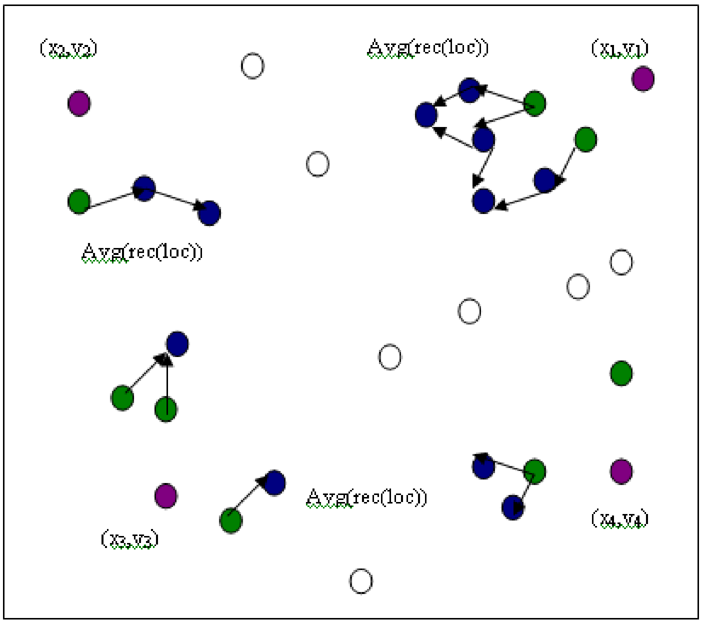


Figure 4.4 Neighbors estimate locations by averaging received locations.

5- Finally, the unconnected nodes, like the red nodes illustrated in Figure 4.5, i.e. have no neighbors can estimate their locations randomly.

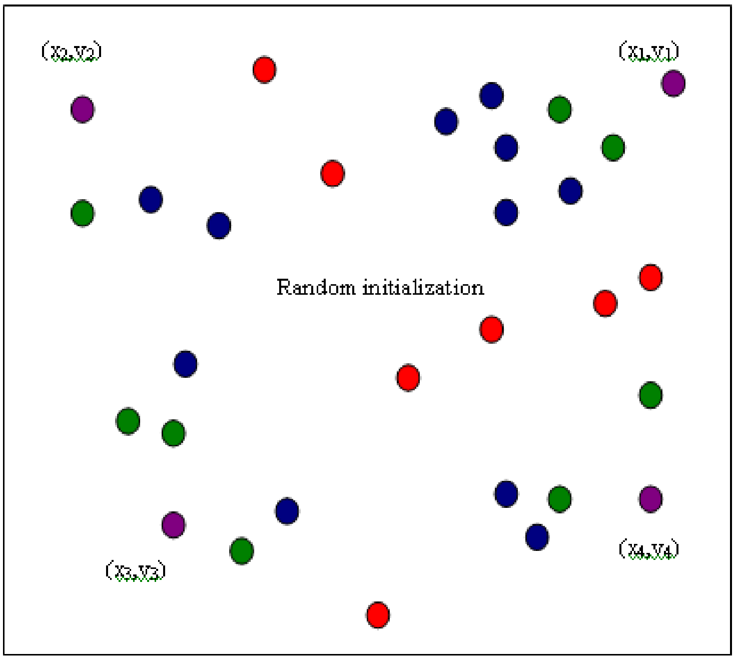


Figure 4.5 Unconnected nodes estimate their locations randomly.

6- All the estimated initial locations can now be fed to the next stage of the algorithm, the learning stage.

4.1.3 The Learning Stage

To begin the iterations of the learning stage, the estimated locations resulted from the previous stage must be exchanged through the network. This can be done according to the following steps:

1- Each node forwards its estimated location to all of its neighbors. Now, each node plays as the input vector and the winning neuron for the region formed by its neighbors. This is done instead of finding the Best Matching Unit (*BMU*) for each node to update its location and the neighboring locations as well. Due to the distributed scheme of the algorithm, the step of finding the *BMU* can be skipped, and consider each node as the *BMU* for the set of its neighbors to do perform the calculations and updates.

2- Based on SOM updating formulas, the neighboring nodes of each node update their weights (locations) using that node as the winning neuron.

3- At the end of the previous step, each node transmits its neighbor location updates to all of its neighbors. On the other hand, it also receives its location updates from its neighbors.

4- Finally, each node calculates its new estimated location by averaging its current location and the received updates from its neighbors.

This process is repeated *T* time (The total number of iterations). The general steps of the Initialization and learning stages are shown in Figure 4.6:

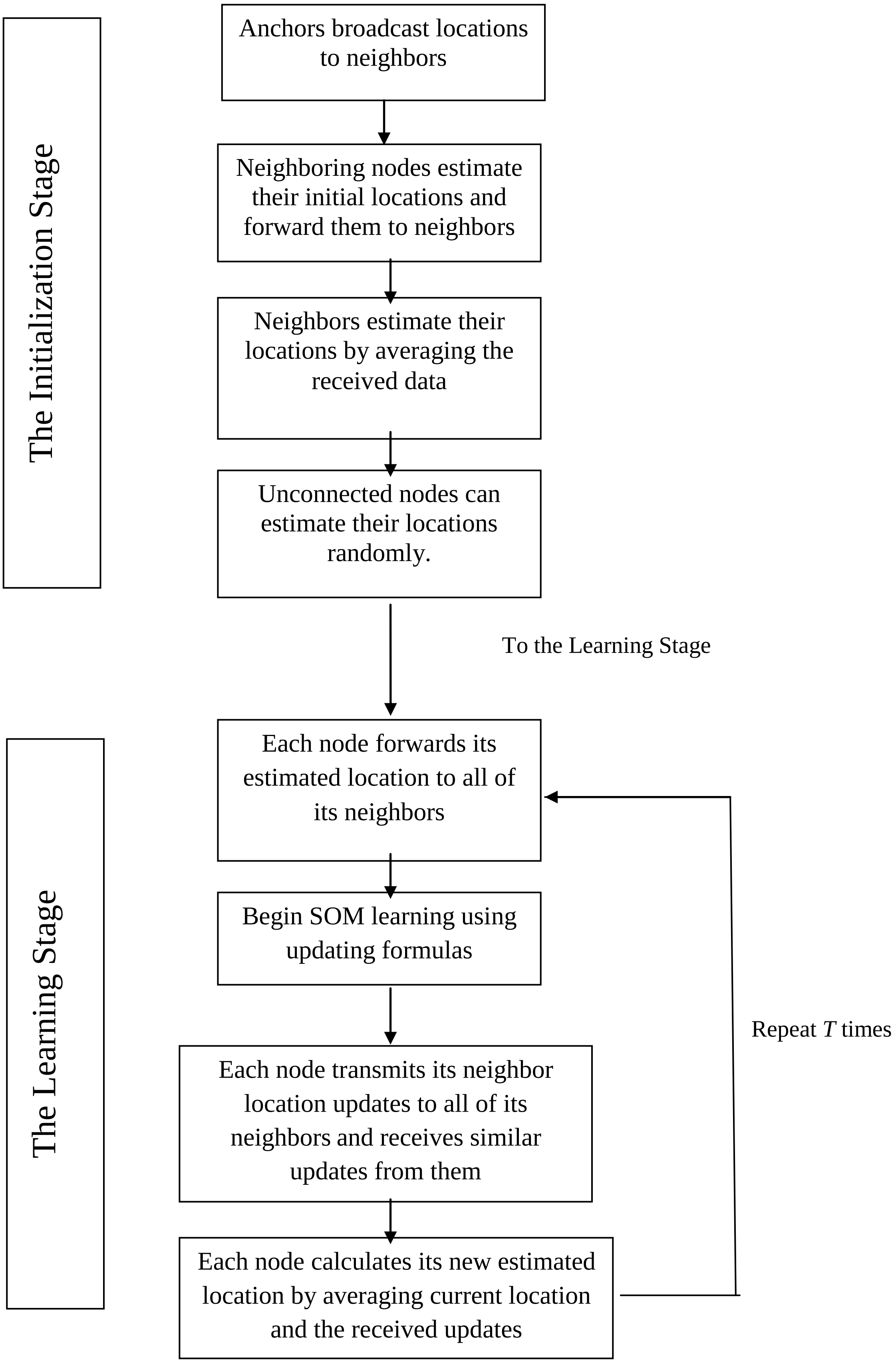

The Initialization Stage
Anchors broadcast locations to neighbors
Neighboring nodes estimate their initial locations and forward them to neighbors
Neighbors estimate their locations by averaging the received data
Unconnected nodes can estimate their locations randomly.
To the Learning Stage
The Learning Stage
Each node forwards its estimated location to all of its neighbors
Begin SOM learning using updating formulas
Each node transmits its neighbor location updates to all of its neighbors and receives similar updates from them
Each node calculates its new estimated location by averaging current location and the received updates
Repeat *T* times

Figure 4.6 The block diagram of the DLSOM algorithm.

4.2 DLSOM Mathematical Equations and Methodology

As described in the previous section, the DLSOM algorithm is divided into two main stages. Given a wireless network with randomly distributed nodes (*N* nodes) in a specified area ($L \times L$) and a determined radio range (*R*) (also called a communication range) in which the nodes can maximally communicate with others, the anchor nodes are selected to be at the four ends (the perimeter) of the network topology (can select anchors randomly or at the four centers, but this selection technique makes the algorithm easier and more accurate).

In the initialization stage, the anchors begin to broadcast their well-known locations to their neighbors (nodes within their communication range), as well as the position of these anchors with regard to the network topology (top-right, top-left, bottom-right, bottom-left). This information can be usefully used by the neighbors to estimate their initial locations. The neighboring nodes to the anchor (top-right) will estimate their locations using the following equation:

For all nodes *i*, where $i = \{1, 2, 3,\ldots.., N_1\}$, N_1 is the number of the top-right anchor neighbors:

$$(x_i , y_i) = (\text{random } [x_{anch1}\text{-}R,\ x_{anch1}],\ \text{random } [y_{anch1}\text{-}R,\ y_{anch1}]) \qquad (4.1)$$

where (x_i , y_i) are the coordinates of the *i*-th neighboring node to the first anchor (top-right), *R* is the communication range, and (x_{anch1}, y_{anch1}) is the well-known location of the first anchor.

The *x*-coordinate of the first anchor (top-right) is supposed to be the maximum *x* value for the whole network nodes. The neighboring nodes of this anchor can take advantage from this information and estimate their *x*-coordinate as a random value in the period of $[x_{anch1}\text{-}R, x_{anch1}]$.

Also, the *y*-coordinate of the first anchor (top-right) is supposed to be the maximum *y* value for the whole network nodes. The neighboring nodes of this anchor can take advantage from this information and estimate their *y*-coordinate as a random value in the period of $[y_{anch1}\text{-}R, y_{anch1}]$.

Similarly, the neighboring nodes to the second anchor (top-left) estimate their locations according to (4.2):

For all nodes j, where j = {1, 2, 3,….., N_2}, N_2 is the number of the top-left anchor neighbors:

$$(x_{\mathrm{j}}\, ,\, y_{\mathrm{j}}) = (\text{random } [x_{anch2},\, x_{anch2}{+}R],\ \text{random } [y_{anch2}{-}R,\, y_{anch2}]) \qquad (4.2)$$

where (x_j , y_j) are the coordinates of the j-th neighboring node to the second anchor (top-left), and (x_{anch2}, y_{anch2}) is the well-known location of the second anchor.

The x-coordinate of the second anchor (top-left) is supposed to be the minimum x value for the whole network nodes. The neighboring nodes of this anchor can take advantage from this information and estimate their x-coordinate as a random value in the period of [x_{anch2}, x_{anch2}+R].

Also, the y-coordinate of the second anchor (top-left) is supposed to be the maximum y value for the whole network nodes. The neighboring nodes of this anchor can take advantage from this information and estimate their y-coordinate as a random value in the period of [y_{anch2}-R, y_{anch2}].

The neighboring nodes to the third anchor (bottom-right) estimate their locations according to (4.3):

For all nodes k, where k = {1, 2, 3,….., N_3}, N_3 is the number of the bottom-right anchor neighbors:

$$(x_k\, ,\, y_k) = (\text{random } [x_{anch3}{-}R,\, x_{anch3}],\ \text{random } [y_{anch3},\, y_{anch3}{+}R]) \qquad (4.3)$$

where (x_k , y_k) are the coordinates of the k-th neighboring node to the third anchor (bottom-right), and (x_{anch3}, y_{anch3}) is the well-known location of the third anchor.

The x-coordinate of the third anchor (bottom-right) is supposed to be the maximum x value for the whole network nodes. The neighboring nodes of this anchor can take advantage from this information and estimate their x-coordinate as a random value in the period of [x_{anch3}-R, x_{anch3}].

Also, the y-coordinate of the third anchor (bottom-right) is supposed to be the minimum y value for the whole network nodes. The neighboring nodes of this anchor can take advantage from this information and estimate their *y*-coordinate as a random value in the period of [y_{anch3}, y_{anch3}+R].

Finally, the neighboring nodes to the fourth anchor (bottom-left) estimate their locations according to (4.4):

For all nodes *z*, where z = {1, 2, 3,….., N_4}, N_4 is the number of the bottom-left anchor neighbors:

$$(x_z\,,\,y_z) = (\text{random }[x_{anch4},\,x_{anch4}{+}R],\ \text{random }[y_{anch4},\,y_{anch4}{+}R]) \qquad (4.4)$$

where (x_z , y_z) are the coordinates of the *z*-th neighboring node to the fourth anchor (bottom-left), and (x_{anch4}, y_{anch4}) is the well-known location of the fourth anchor.

The *x*-coordinate of the fourth anchor (bottom-left) is supposed to be the minimum *x* value for the whole network nodes. The neighboring nodes of this anchor can take advantage from this information and estimate their *x*-coordinate as a random value in the period of [x_{anch4}, x_{anch4}+R].

Also, the *y*-coordinate of the fourth anchor (bottom-left) is supposed to be the minimum *y* value for the whole network nodes. The neighboring nodes of this anchor can take advantage from this information and estimate their *y*-coordinate as a random value in the period of [y_{anch4}, y_{anch4}+R].

Now, the estimated initial locations of the anchors' neighbors are transmitted to the unknown neighboring nodes. These neighboring nodes will actually receive many estimated locations (according to the number of neighbors). Each node estimates its initial location by averaging the received locations according to the following equation:

For all nodes *v*, where v = {1, 2, 3, ….., N_5}, N_5 is the number of inner nodes that received estimated locations from neighbors:

$$(x_v\,,\,y_v) = \left(\frac{\sum_{i=1}^{N_v} x_i}{N_v} \;,\; \frac{\sum_{i=1}^{N_v} y_i}{N_v} \right) \qquad (4.5)$$

where (x_v , y_v) are the coordinates of the v-th node, and N_v is the number of node's v neighbors.

As a final step of the initialization stage, the rest of the network nodes that did not receive any estimated location, due to its wide distance of the network nodes, i.e. out of range of any network node will estimate their initial locations randomly (no knowledge of any neighboring information).

For all nodes s, $s = \{1, 2, 3, \ldots.., N_6\}$, N_6 is the number of the remaining unconnected nodes:

$$(x_s , y_s) = \text{random}() \tag{4.6}$$

where (x_s , y_s) are the coordinates of the s-th node.

After completing the initialization stage, each node forwards its estimated location to all of its neighbors in preparation to the learning stage.

Before going into the learning stage SOM formulas, let us formulate the mathematical notations that will be used in these formulas and equations. The actual locations of the wireless nodes are denoted by w_i^a ($i = 1, 2, 3, \ldots, N$) and the estimated locations are denoted by w_i^e ($i = 1, 2, 3, \ldots, N$).

As each node forwarded its estimated initial location to all of its neighbors, it also knows the estimated locations of its neighbors, denoted as $w_{i,j}^e$ ($j = 1, 2, 3, \ldots, N_i$) where N_i is the number of neighboring nodes to the node with location w_i^e .

Now, the node with the location w_i^e plays as the winning neuron to the region formed by the neighbors of that node. Instead of finding the *BMU* for each node, this calculation step is skipped. Because each node knows its neighbors (within communication range) and due to the distributed scheme of the algorithm, each node can be considered as the *BMU* to the set of its neighbors and performs the location updates of its neighbors.

The following steps represent the learning stage of the algorithm:

1. Based on SOM, the neighboring nodes of w_i^e will update their weights (locations) according to the following updating formula:

$$w_{i,j}^e\ (t+1) = w_{i,j}^e\ (t) + \Delta\ (t) \qquad (4.7)$$

where Δ (t) is calculated using:

$$\Delta\ (t) = \alpha\ (t)\ (\, w_i^e\ -\ w_{i,j}^e\ (t)) \qquad (4.8)$$

And α (t) is the learning rate exponential decay function calculated using:

$$\alpha\ (t) =\ \exp\ (\frac{-(t+1)}{T}) \qquad (4.9)$$

where t is t-th time step of the total T learning steps.

2. The node with location w_i^e now transmits its neighbor location updates to all of its neighbors, and on the other hand, receives the same updates from its neighbors as $w_{j,i}^e$ (j = 1, 2, 3, …, N_i).

3. At the end of the step, the node with location w_i^e calculates its new estimated location according to the following equation:

$$w_i^e\ =\ \frac{1}{N_i+1}\ (\sum_{j=1}^{N_i}\ w_{j,i}^e\ +\ w_i^e\ ) \qquad (4.10)$$

This is done by averaging the current location and the received updates from neighbors.

4. Now, it re-forwards its new estimated location to its neighbors.

This learning process (steps 1-4) is repeated T times. The number of iterations can be varied till convergence (reaching stable results).

The algorithm pseudocode, initialization stage flowchart, and learning stage flowchart are illustrated in Figure 4.7, Figure 4.8, and Figure 4.9, respectively.

```
Algorithm 4.1: DLSOM
Purpose: Wireless Node Localization
Input: $N$: No. of Nodes, $R$: Communication Radio Range
Output: $w_i^e$, $i$=1,…., $N$: Estimated Nodes' Locations
Procedure:
% Initialization
for all anchors anch1, anch2, anch3, anch4 do
    Broadcast location to neighbors
end for
for all nodes  i=1: N1 do  % N1, number of anch1 neighbors
    Receive anch1 location ($x_{anch1}$, $y_{anch1}$)
   % anch1 is top-right; estimate location based on the equation:
    $(x_i , y_i)$ = (random [$x_{anch1}$-$R$, $x_{anch1}$], random [$y_{anch1}$-$R$, $y_{anch1}$])
end for

for all nodes j=1: N2  do   % N2, number of anch2 neighbors
    Receive anch2 location ($x_{anch2}$, $y_{anch2}$)
   % anch2 is top-left; estimate location based on the equation:
    $(x_j , y_j)$ = (random [$x_{anch2}$, $x_{anch2}$+$R$], random [$y_{anch2}$-$R$, $y_{anch2}$])
end for

for all nodes k=1: N3  do   % N3, number of anch3 neighbors
    Receive anch3 location ($x_{anch3}$, $y_{anch3}$)
   % anch3 is bottom-right; estimate location based on the equation:
    $(x_k , y_k)$ = (random [$x_{anch3}$-$R$, $x_{anch3}$], random [$y_{anch3}$, $y_{anch3}$+$R$])
end for
```

Figure 4.7 DLSOM pseudocode, cont.

```
for all nodes $z=1 :N4$    do    % $N4$, number of anch4 neighbors
    Receive anch4 location ($x_{anch4}$, $y_{anch4}$)
    % anch4 is bottom-left; estimate location based on the equation:
    $(x_z , y_z) = (\text{random } [x_{anch4}, x_{anch4}+R], \text{random } [y_{anch4}, y_{anch4}+R])$
end for

for all nodes $m=1:M$    do   % $M= (N1+N2+N3+N4)$
    Transmit estimated locations to neighbors
end for

for all nodes  $v = 1: N_5$    do    % $N5$,  number of inner nodes that have neighbors
    Receive estimated locations from neighbors
    % estimate location based on the equation:
    $(x_v , y_v) = \left( \frac{\sum_{i=1}^{N_v} x_i}{N_v} , \frac{\sum_{i=1}^{N_v} y_i}{N_v} \right)$
end for

for all nodes  $s = 1: N_6$    do  % $N_6$ , number of the remaining unconnected nodes
    $(x_s , y_s) = \text{random } ()$
end for
```

Figure 4.7 DLSOM pseudocode, cont.

```
for all nodes $i$ do
    $w_i^e = (x_i , y_i)$
end for

% SOM Learning
for $t$ =1:$T$ do
    $\alpha (t) = \exp (\frac{-(t+1)}{T})$
    for all nodes $i$ do
        for all nodes $j=1:N_i$ do    % $N_i$ , number of $w_i^e$ neighbors
            $\Delta (t) = \alpha (t) ( w_i^e - w_{i,j}^e (t))$
            $w_{i,j}^e (t+1) = w_{i,j}^e (t) + \Delta (t)$
        end for

    for all nodes $i$ do
        Transmit updated locations to neighbors
        Receive updated locations from neighbors
        $w_i^e = \frac{1}{N_i + 1} ( \sum_{j=1}^{N_i} w_{j,i}^e + w_i^e )$
    end for
end for
```

Figure 4.7 DLSOM pseudocode.

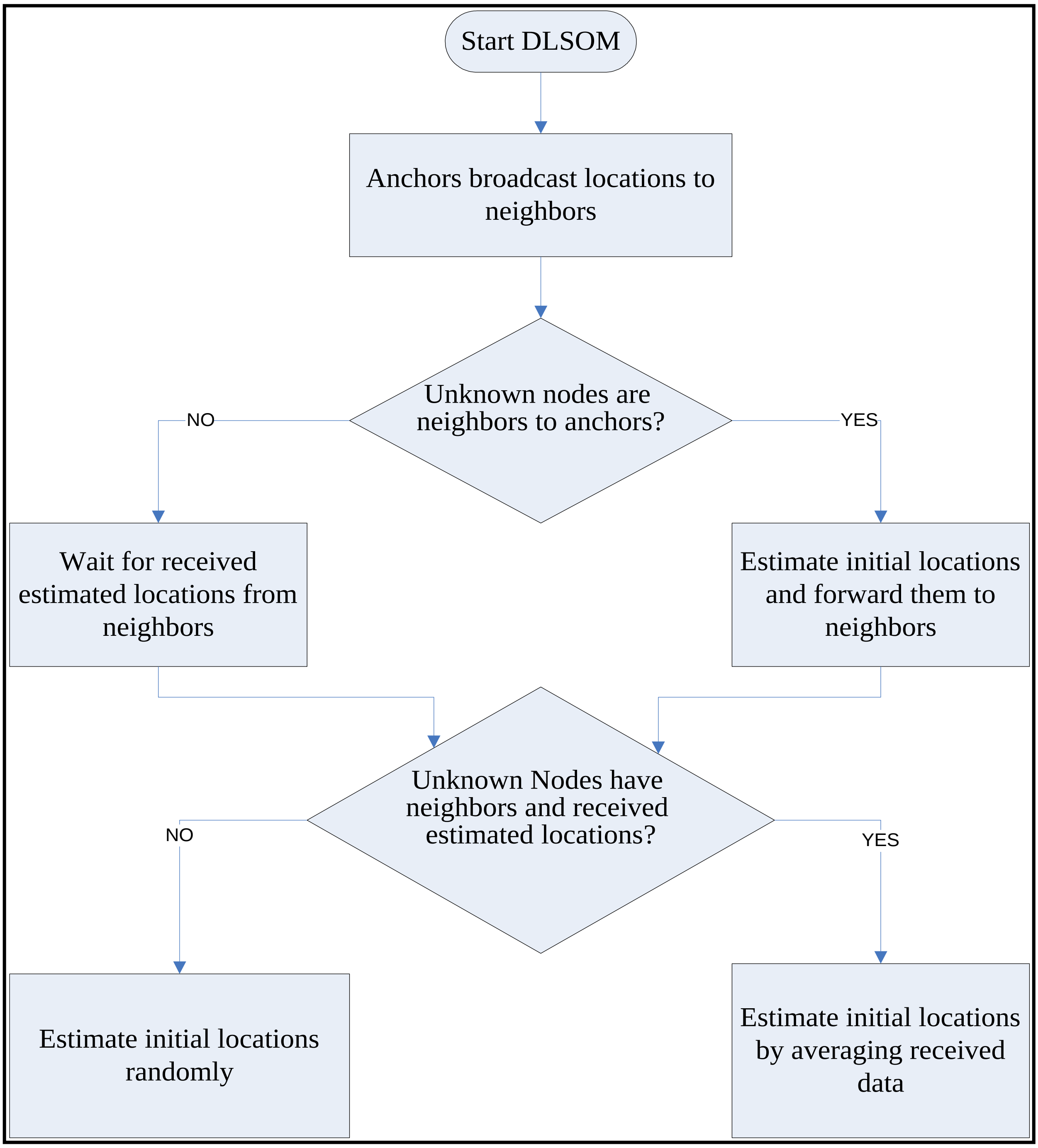

Start DLSOM
Anchors broadcast locations to neighbors
Unknown nodes are neighbors to anchors?
NO
YES
Wait for received estimated locations from neighbors
Estimate initial locations and forward them to neighbors
Unknown Nodes have neighbors and received estimated locations?
NO
YES
Estimate initial locations randomly
Estimate initial locations by averaging received data

Figure 4.8 The Initialization Stage Flowchart.

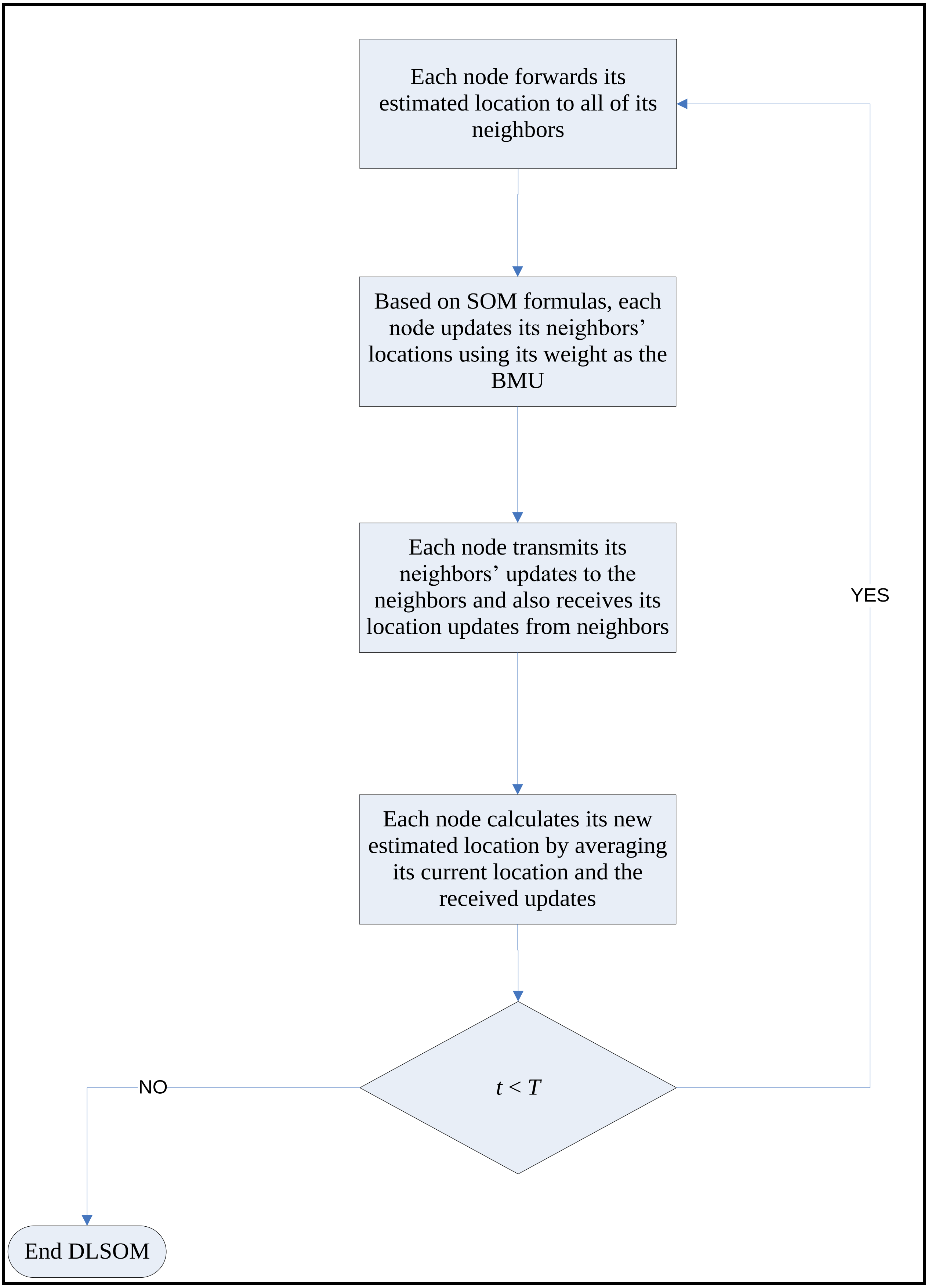

Each node forwards its estimated location to all of its neighbors
Based on SOM formulas, each node updates its neighbors' locations using its weight as the BMU
Each node transmits its neighbors' updates to the neighbors and also receives its location updates from neighbors
Each node calculates its new estimated location by averaging its current location and the received updates
t < T
YES
NO
End DLSOM

Figure 4.9 The Learning Stage Flowchart.

Chapter 5

Experimentation and Results

5.1 Experimental Environment

The DLSOM algorithm is implemented and executed using Matlab 7.9 Software. The source code is run on a desktop PC with Intel Pentium 4.0, 2.6 GHz CPU and 512 KB RAM. The flow of the program is as follows:

1- Wireless network topology generation and deployment: The network was generated randomly based on the following varying parameters:

(a) Number of Nodes (*N*): varied from 10 to 100 nodes.

(b) Communication Range (*R*): varied from 1 km to 4 km

(c) Deployment Area (*L*×*L*): 10 km× 10 km is used.

2- Initialization Stage: The initial estimated locations of nodes were calculated according to the following parameters:

(a) Four anchors are selected at the perimeter of the network topology (top-right, bottom-right, top-left, bottom-left).

3- SOM Learning Stage: The final estimated locations were calculated according to the following SOM parameters:

(a) Maximum Number of Iterations (*T*): varied from 25 to 50.

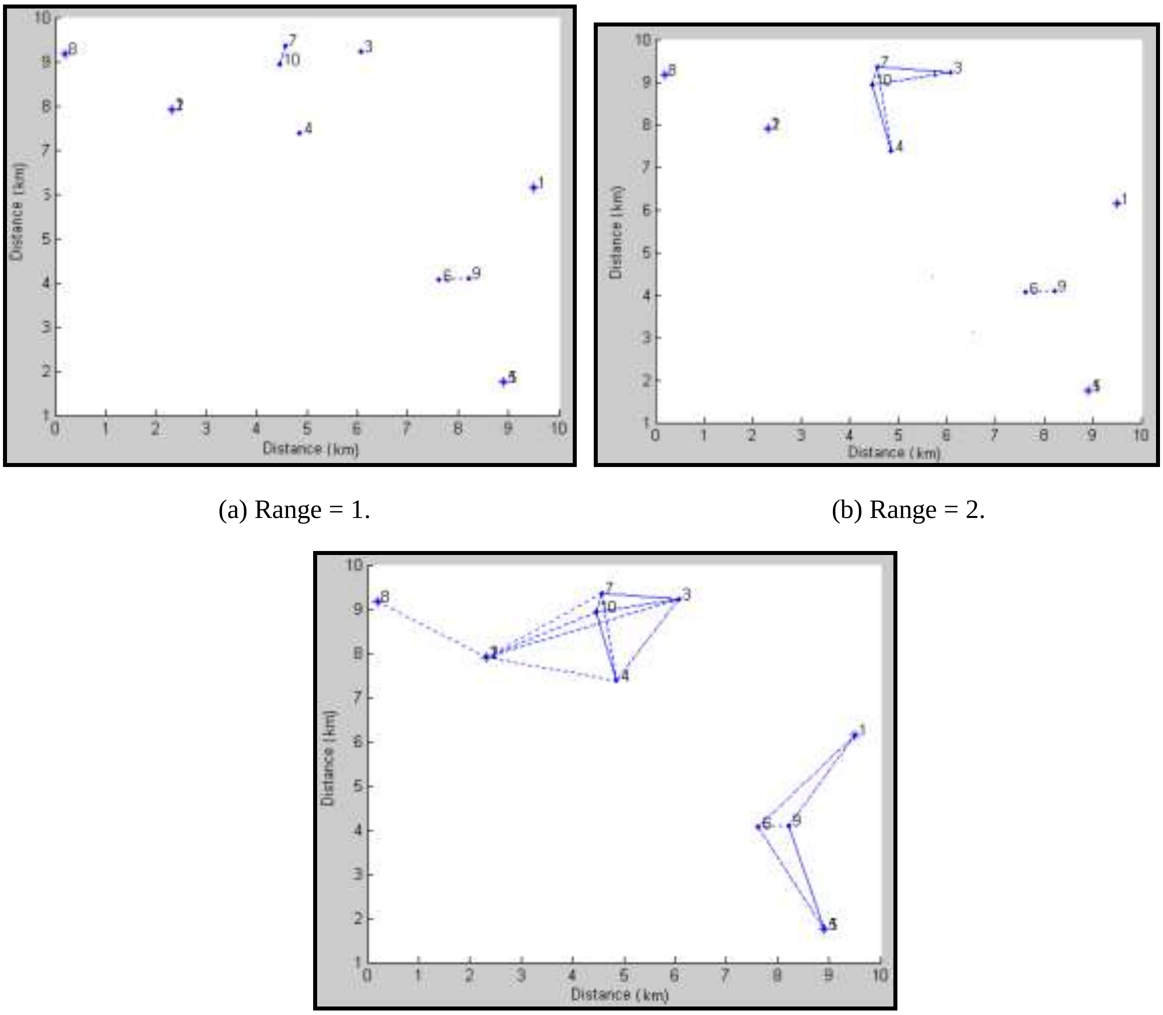
(b) Learning Rate (α): a decay exponential function of the current iteration and the maximum number of iterations is used.

(c) Weight Updating Formula (Δ): is calculated using the learning rate and the estimated vectors.

5.2 Simulation Results

5.2.1 A 10-Nodes Wireless Network

The wireless network is generated randomly with 10 wireless nodes in an area of 10km ×10km. Figure 5.1 shows the actual network nodes' locations with different communication ranges, and the lines connecting the nodes represent the neighborhood relations between nodes based on the communication range given to the network. In Figure 5.1 (a), (b), and (c), the ranges are 1km, 2km, and 4km, respectively**.**



(a) Range = 1.

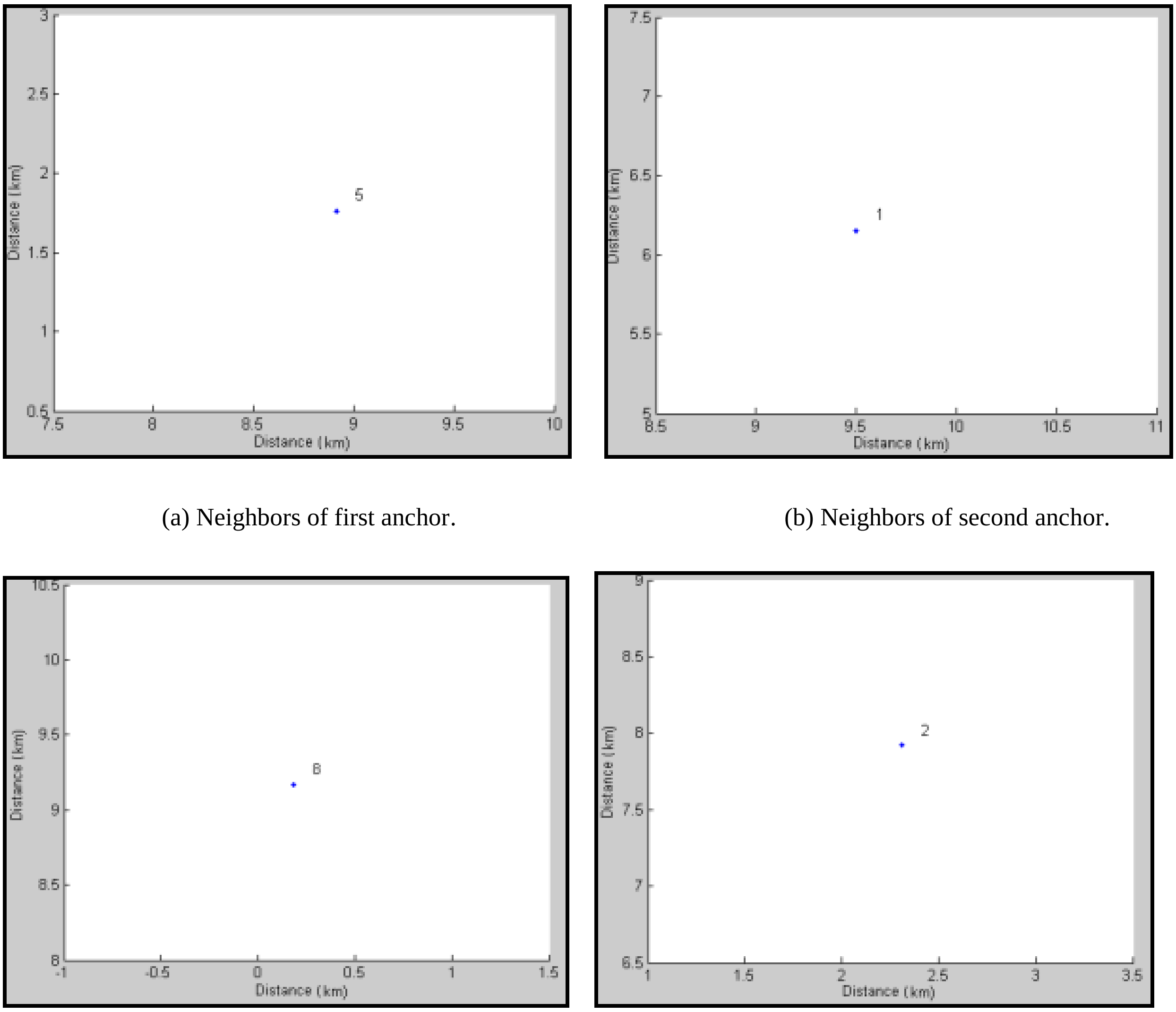
(b) Range = 2.

(c) Range = 4.

Figure 5.1 Actual Wireless Networks (N=10).

The anchors are selected to be nodes 1 (top-right), 2 (bottom-left), 5 (bottom-right), and 8 (top-left).

For the communication range 1, Figure 5.2 (a), (b), (c), and (d) show the estimated locations for the neighbors of these anchors. As observed in the actual network, due to the low connectivity, the anchors almost have no neighbors.



(a) Neighbors of first anchor.

(b) Neighbors of second anchor.

(c) Neighbors of third anchor.

(d) Neighbors of fourth anchor.

Figure 5.2 Estimated Locations of anchors' neighbors (N=10, R=1).

For this communication range, most of the nodes have no neighbors, this, of course, will force the nodes to estimate their locations randomly (no neighborhood information is available).

After the initialization stage for all the network nodes is complete, each node has an initial estimated location as illustrated in Figure 5.3 These locations are to be used in the SOM learning as initial weights.

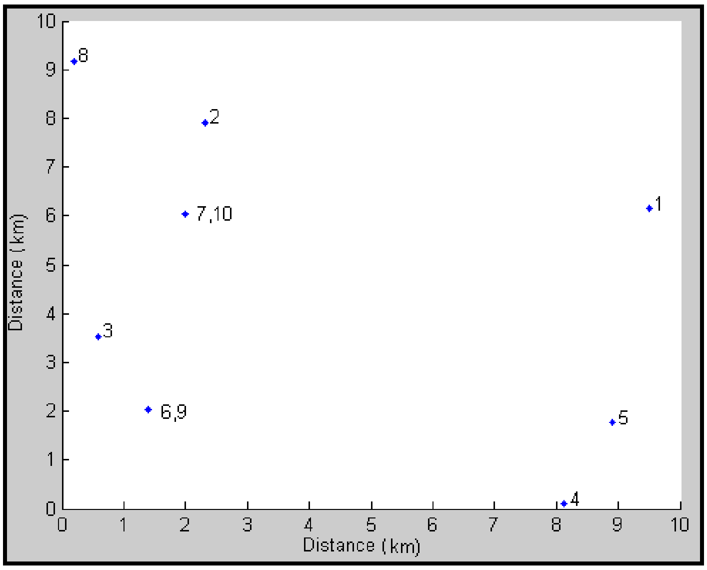


Figure 5.3 The estimated locations for the network nodes (N=10, R=1).

The degree of divergence between the actual network and the network of initial estimated locations (shown in figure 5.3) is very high. Almost all the network nodes have estimated their locations randomly.

As a result of the SOM learning, the final estimated locations (weights) are shown in Figure 5.4:

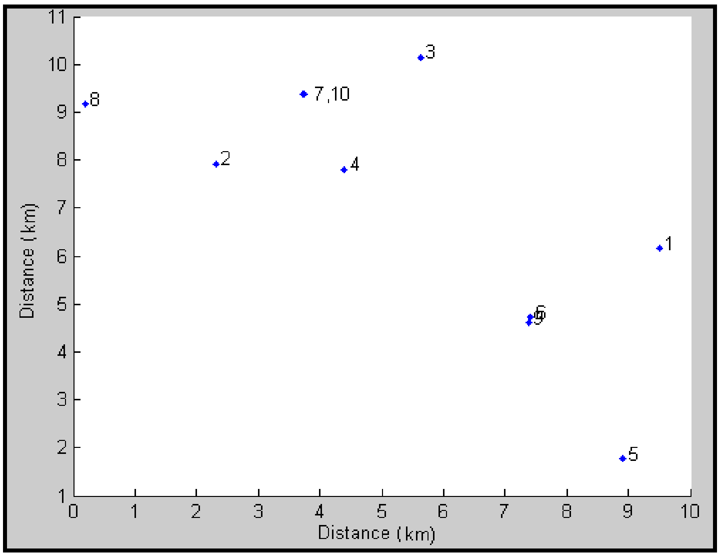
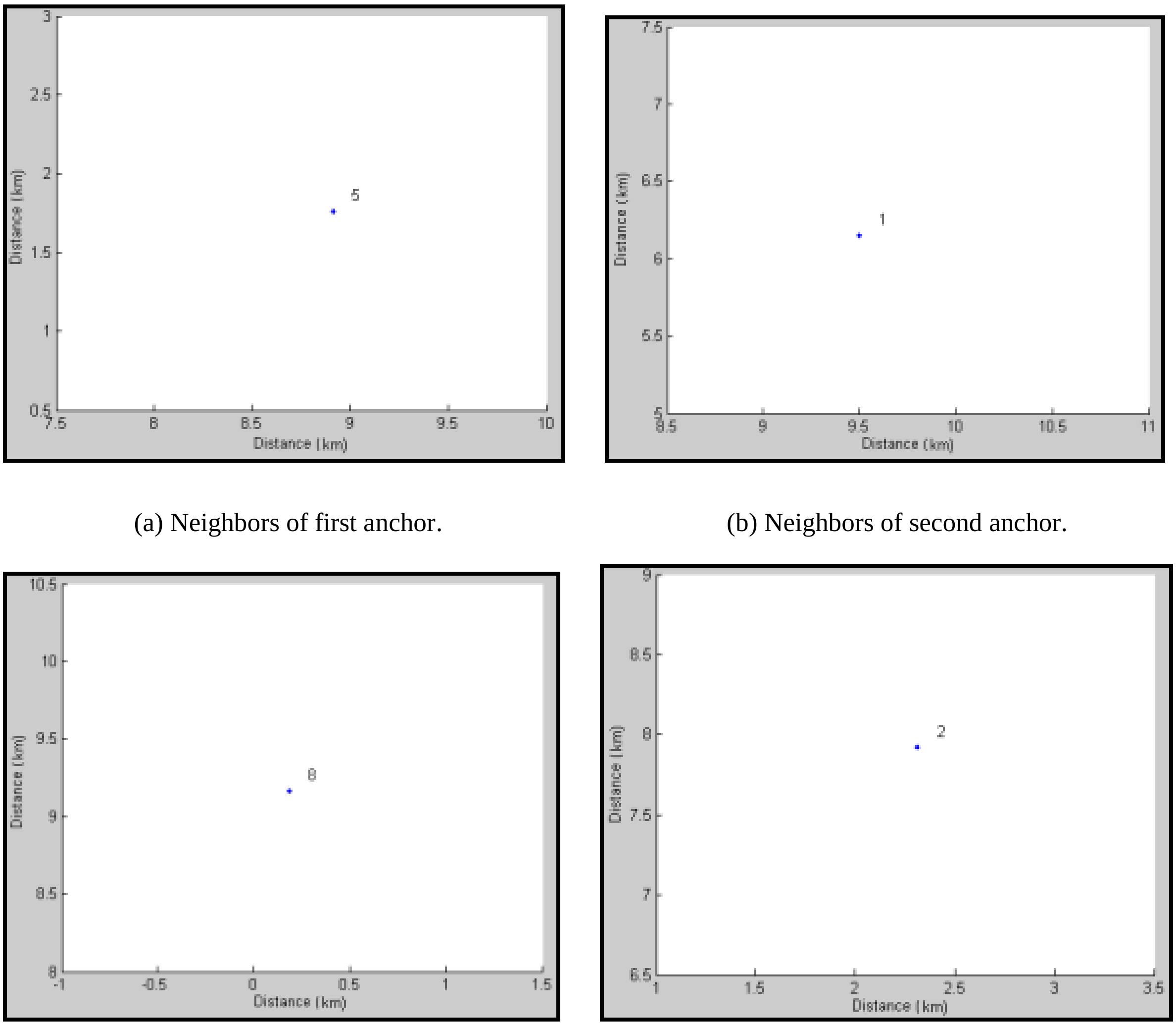


Figure 5.4 The resulted estimated locations for the network nodes (N=10, R=1).

Obviously, the accuracy of the result is not very high. For example the nodes 6 and 9 shared the same location, and node 3 has an estimated location with a considerable difference of the actual one. Also, the number of iterations needed to reach convergence (stable results) is considerably high (took about 50 iterations).

For the communication range 2, Figure 5.5 (a), (b), (c), and (d) shows the estimated locations for the neighbors of the four anchors. As observed in the actual network, due to the low connectivity and low number of nodes, the anchors almost have no neighbors.



(a) Neighbors of first anchor.

(b) Neighbors of second anchor.

(c) Neighbors of third anchor.

(d) Neighbors of fourth anchor.

Figure 5.5 Estimated Locations of anchors' neighbors (N=10, R=2).

After the initialization stage for all the network nodes is complete, each node has an initial estimated location as illustrated in Figure 5.6 These locations are to be used in the SOM learning as initial weights.

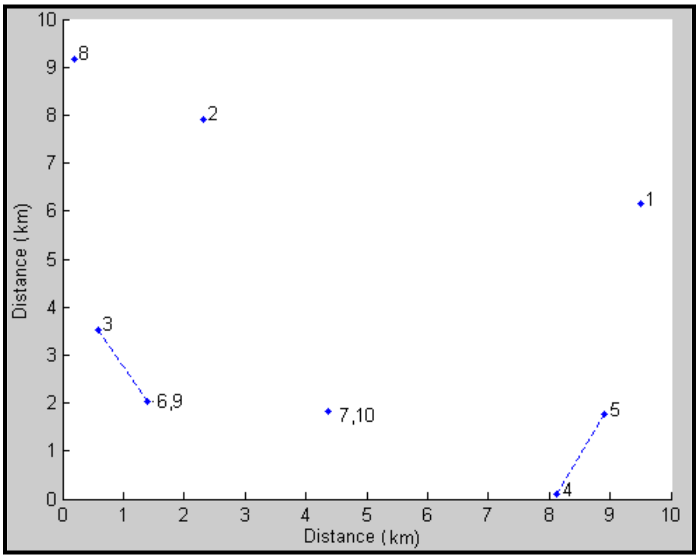


Figure 5.6 The estimated locations for the network nodes (N=10, R=2).

Here, the degree of divergence between the actual network and the initial estimations network is much smaller. This is because some of the inner nodes have neighbors and got advantage of the estimated locations.

As a result of the SOM learning, the final estimated locations (weights) are shown in Figure 5.7:

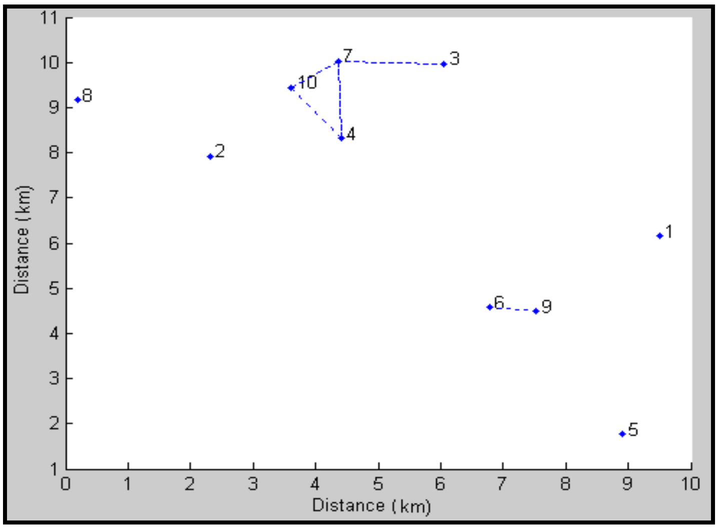
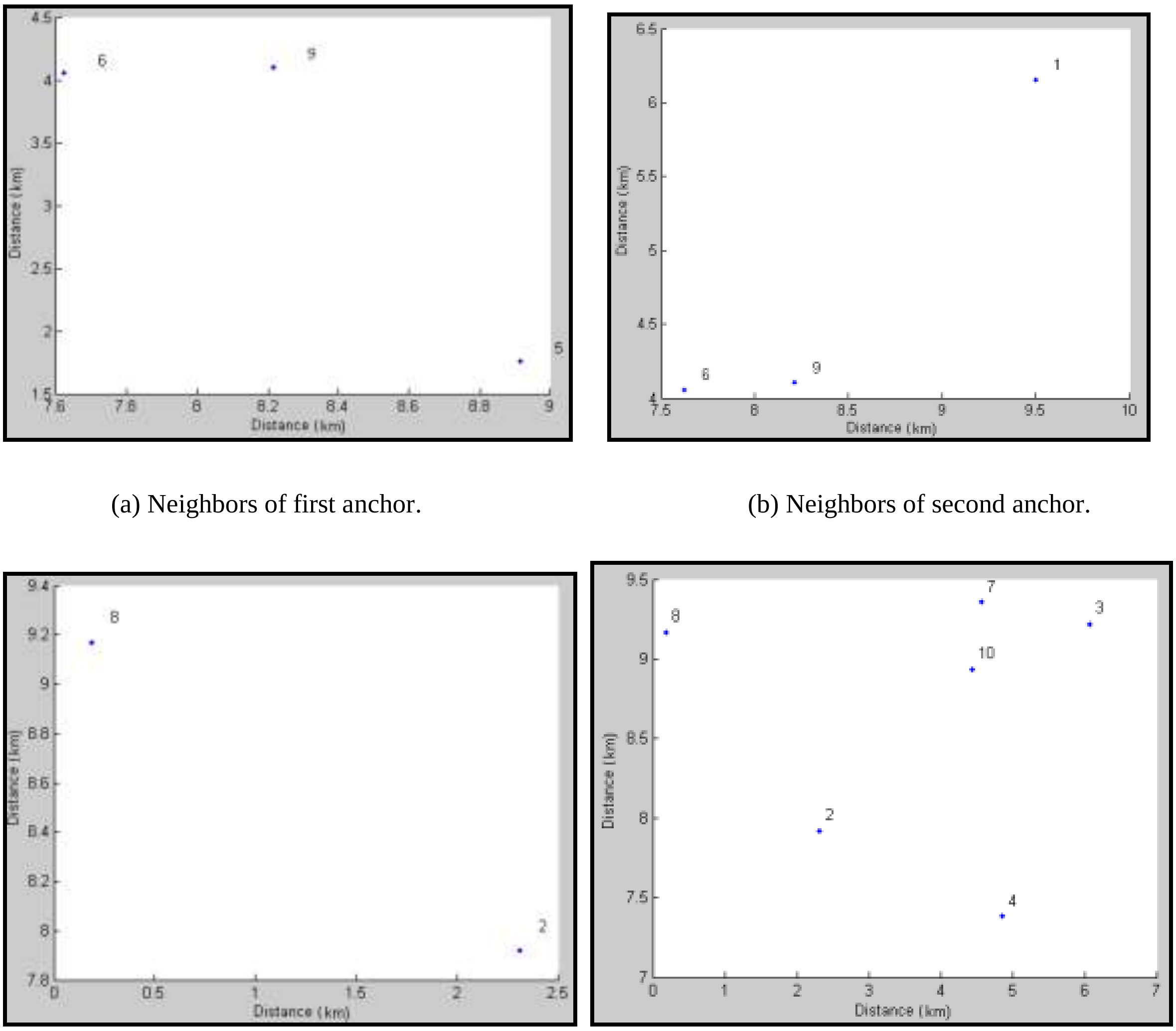


Figure 5.7 The resulted estimated locations for the network nodes (N=10, R=2).

Obviously, the accuracy of the result is also much higher (about twice), and also, the number of iterations needed to reach convergence (stable results) is lower (took about 35 iterations).

For the communication range 4, Figure 5.8 (a), (b), (c), and (d) shows the estimated locations for the neighbors of the four anchors. As observed in the actual network, all anchors have neighbors. These neighbors estimate their locations as shown in the following figures.



(a) Neighbors of first anchor.

(b) Neighbors of second anchor.

(c) Neighbors of third anchor.

(d) Neighbors of fourth anchor.

Figure 5.8 Estimated Locations of anchors' neighbors (N=10, R=4).

For this communication range, most of the nodes have neighbors, this, of course, will help the nodes to estimate their locations based on neighborhood information and totally get benefit from the proposed algorithm.

After the initialization stage for all the network nodes is complete, each node has an initial estimated location as illustrated in Figure 5.9 These locations are to be used in the SOM learning as initial weights.

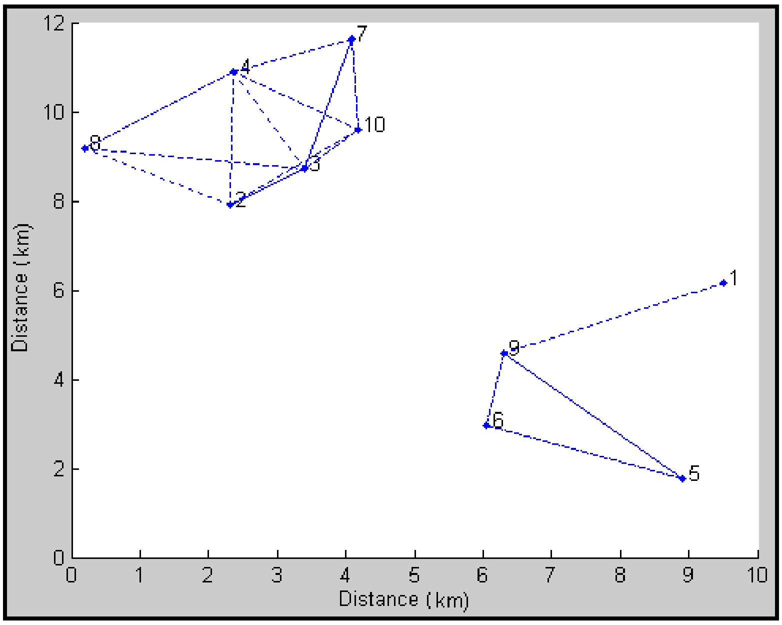


Figure 5.9 The estimated locations for the network nodes (N=10, R=4).

The degree of divergence between the actual network and the initial estimations network is considerably small. As a result of the SOM learning, the final estimated locations (weights) are shown in Figure 5.10:

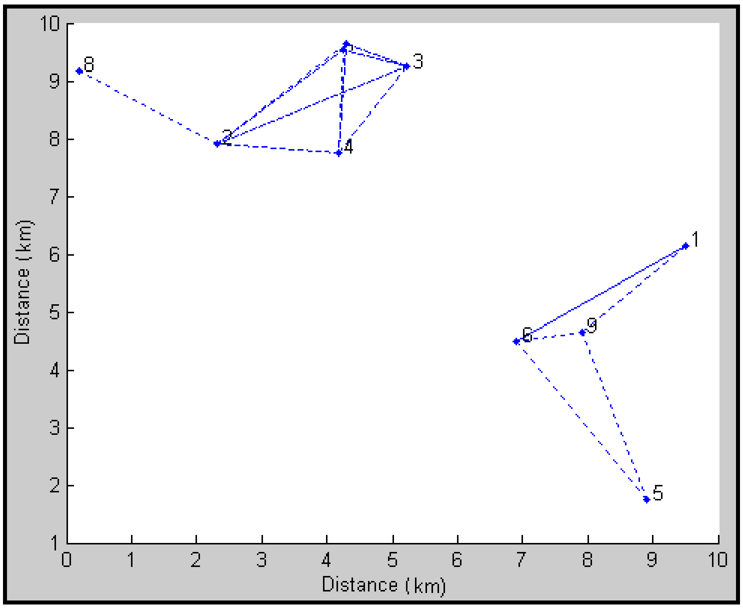
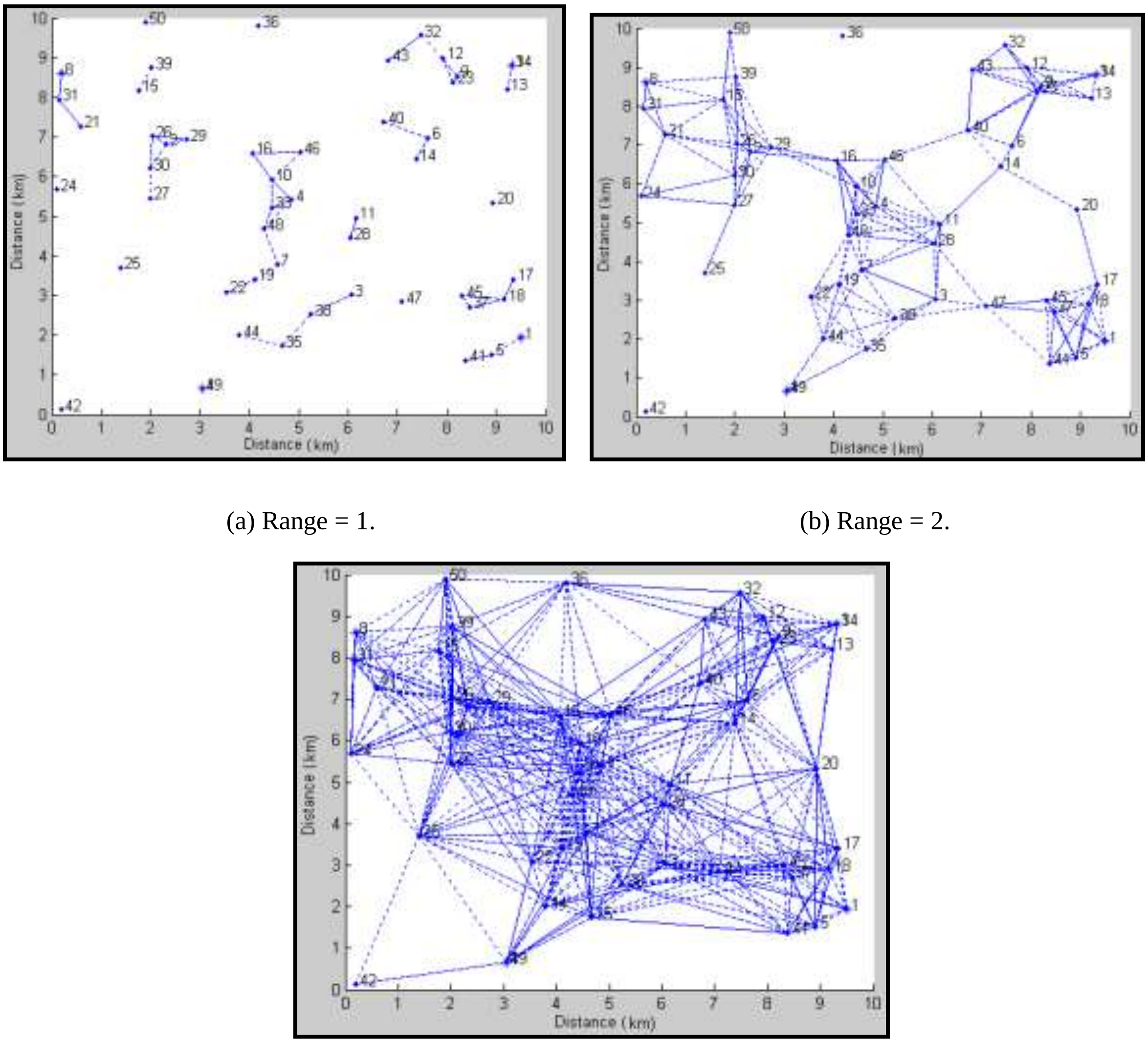


Figure 5.10 The resulted estimated locations for the network nodes (N=10, R=4).

The accuracy of the result is considerably high. This can be observed from the previous figure, in which most of the nodes have estimated locations with low difference of the actual locations. Also, the number of iterations needed to reach convergence (stable results) is very low (took about 25 iterations).

5.2.2 A 50-Nodes Wireless Network

The wireless network is generated randomly with 50 wireless nodes in an area of 10km ×10km. Figure 5.11 shows the actual network nodes' locations with different communication ranges, and the lines connecting the nodes represent the neighborhood relations between nodes based on the communication range given to the network. In Figure 5.11 (a), (b), and (c), the ranges are 1km, 2km, and 4km, respectively.



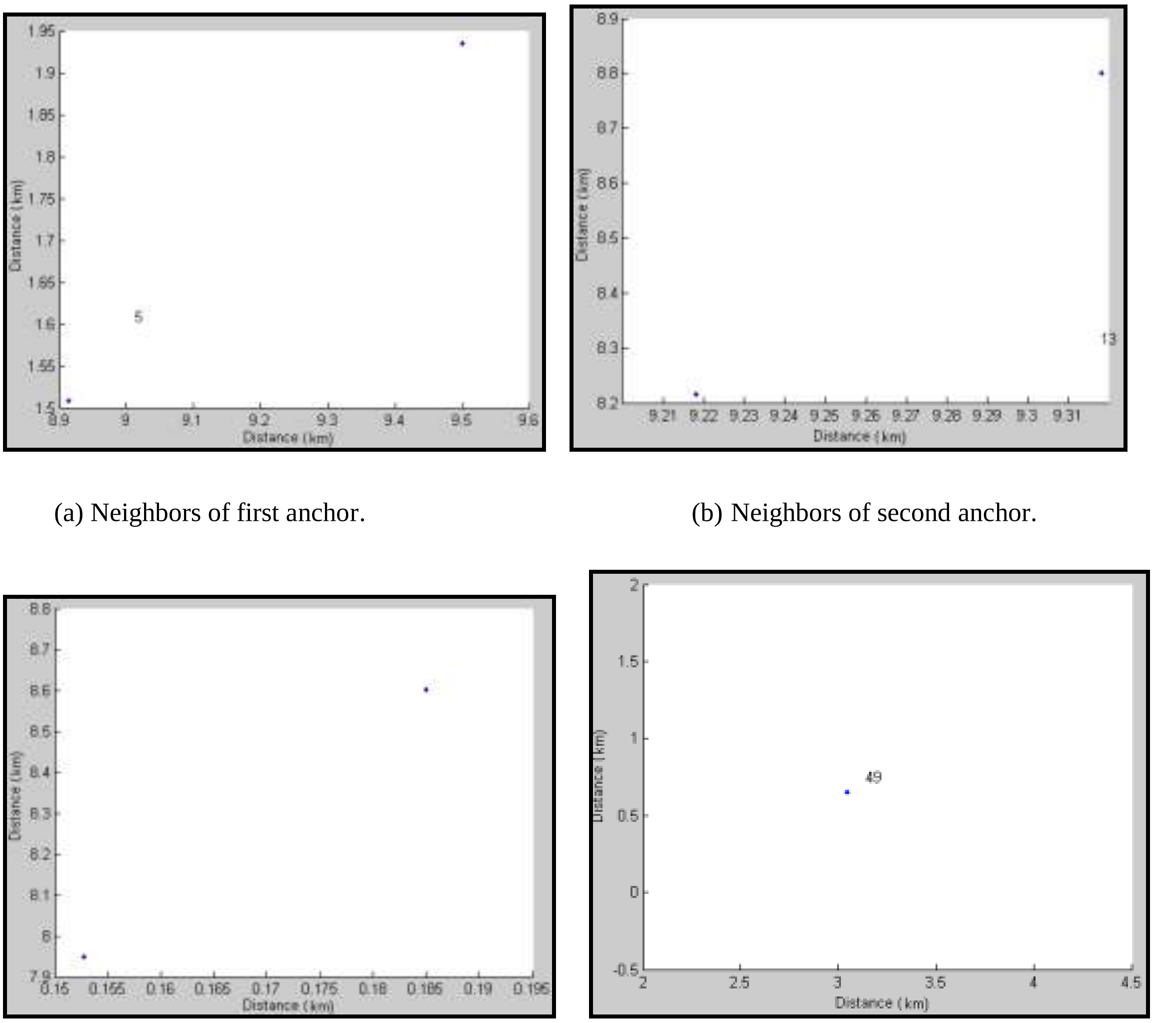
(a) Range = 1.

(b) Range = 2.

(c) Range = 4.

Figure 5.11 Actual Wireless Networks (N=50).

The anchors are selected to be nodes 1 (bottom-right), 8 (top-left), 34 (top-right), and 49 (bottom-left). For the communication range 1, Figure 5.12 (a), (b), (c), and (d) shows the estimated locations for the neighbors of these anchors. As observed in the actual network, the anchors almost have no neighbors due to low radio range.



(a) Neighbors of first anchor.

(b) Neighbors of second anchor.

(c) Neighbors of third anchor.

(d) Neighbors of fourth anchor.

Figure 5.12 Estimated Locations of anchors' neighbors (N=50, R=1).

After the initialization stage for all the network nodes is complete, each node has an initial estimated location as illustrated in Figure 5.13 These locations are to be used in the SOM learning as initial weights.

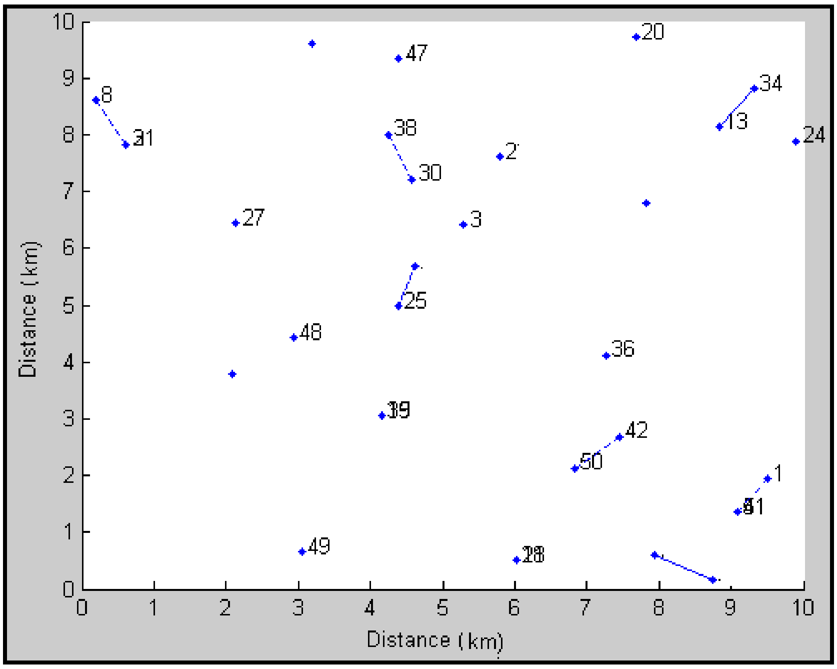


Figure 5.13 The estimated locations for the network nodes (N=50, R=1).

Most of the nodes estimated their locations randomly due to low connectivity. Also, some of the inner nodes that have neighbors shared the same estimated location. As a result of the SOM learning, the final estimated locations (weights) are shown in Figure 5.14:

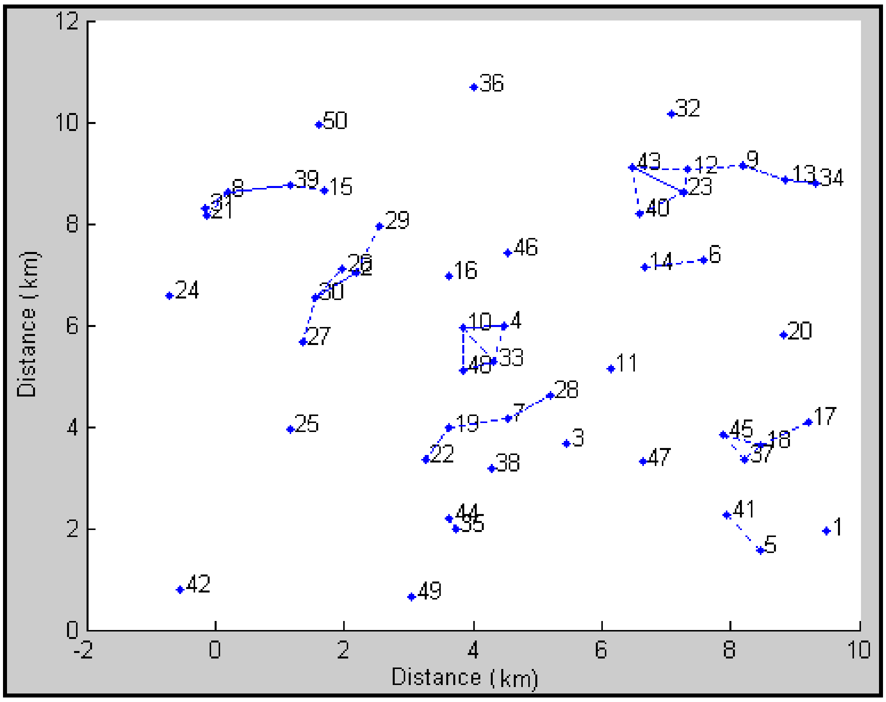
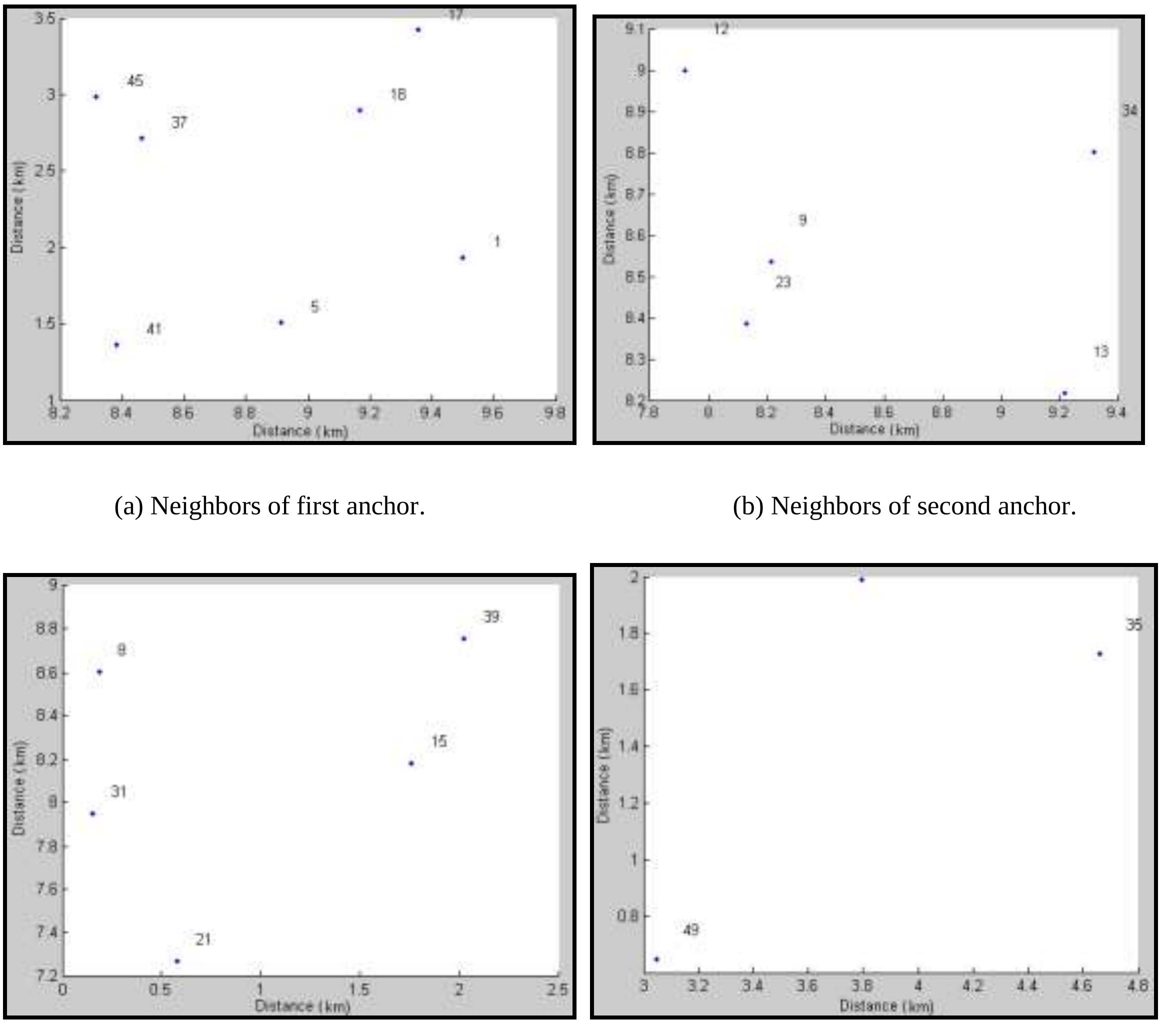


Figure 5.14 The resulted estimated locations for the network nodes (N=50, R=1).

The accuracy of the result is relatively low. As shown in the previous figure, the final estimated locations for most of the nodes have noticeable difference of the actual locations. Also, the number of iterations needed to reach convergence (stable results) is relatively high (In contrast with the 10-Node wireless network with R = 4). Here, It took about 35 iterations to converge.

For the communication range 2, Figure 5.15 (a), (b), (c), and (d) shows the estimated locations for the neighbors of the four anchors. All four anchors have considerable number of neighbors that can estimate their locations easily.



(a) Neighbors of first anchor.

(b) Neighbors of second anchor.

(c) Neighbors of third anchor.

(d) Neighbors of fourth anchor.

Figure 5.15 Estimated Locations of anchors' neighbors (N=50, R=2).

After the initialization stage for all the network nodes is complete, each node has an initial estimated location as illustrated in Figure 5.16 These locations are to be used in the SOM learning as initial weights.

For this communication range, most of the nodes have neighbors, this, of course, will help the nodes to estimate their locations based on neighborhood information and totally get benefit from the proposed algorithm.

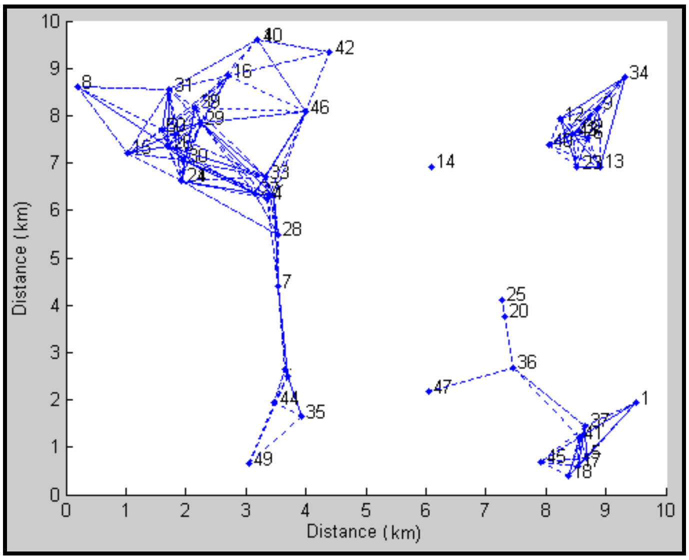


Figure 5.16 The estimated locations for the network nodes (N=50, R=2).

As a result of the SOM learning, the final estimated locations (weights) are shown in Figure 5.17:

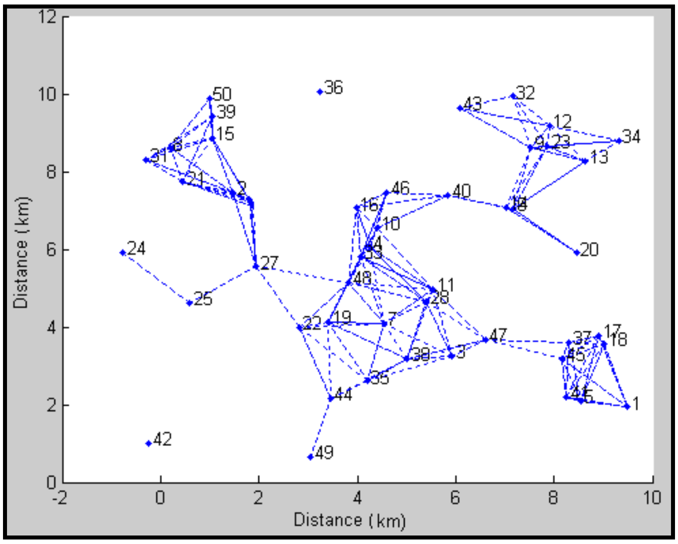
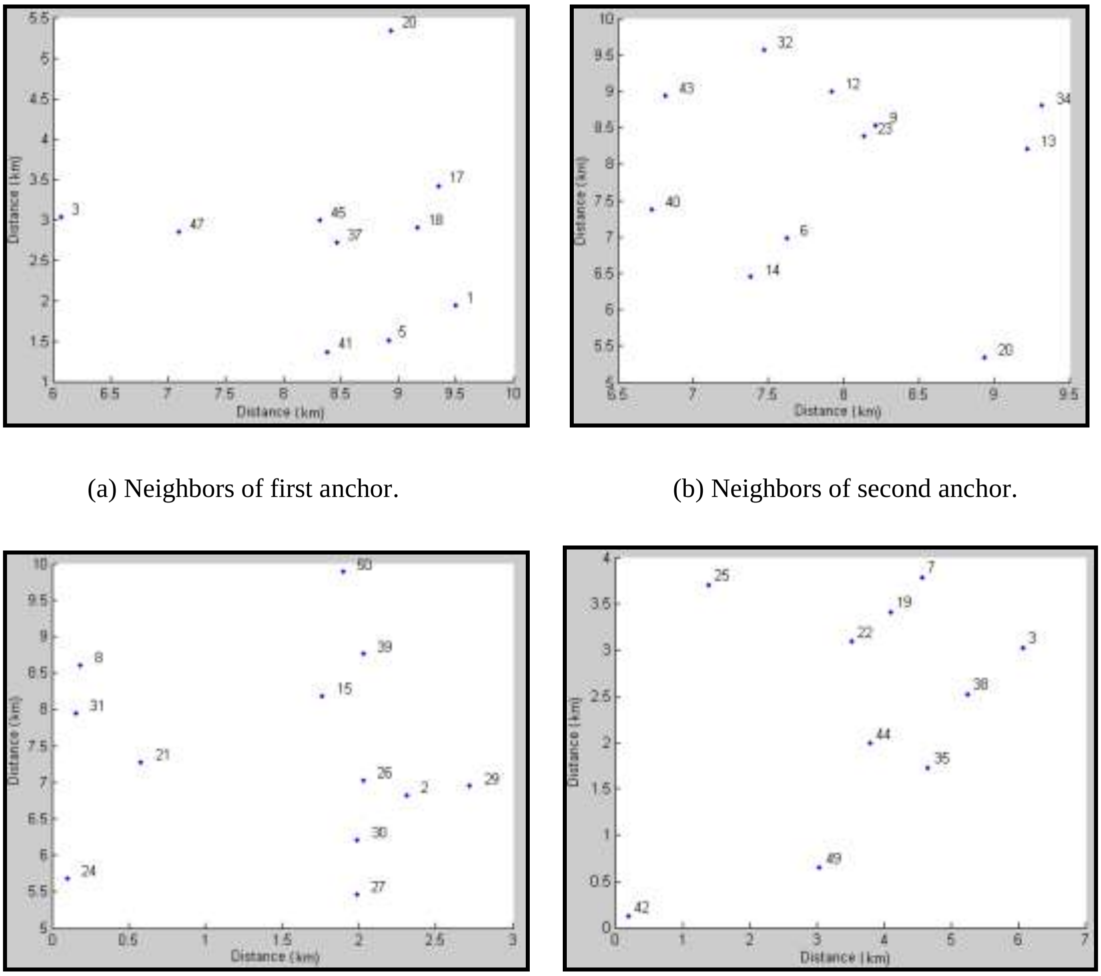


Figure 5.17 The resulted estimated locations for the network nodes (N=50, R=2).

The accuracy of the result is higher, and also, the number of iterations needed to reach convergence (stable results) is much lower (took about 25 iterations). This is due to the higher connectivity and the correlation between nodes.

For the communication range 4, Figure 5.18 (a), (b), (c), and (d) shows the estimated locations for the neighbors of the four anchors. As can be shown in the actual network, all anchors have large number of neighbors that can estimate their locations based on received anchors' locations.



(a) Neighbors of first anchor.

(b) Neighbors of second anchor.

(c) Neighbors of third anchor.

(d) Neighbors of fourth anchor.

Figure 5.18 Estimated Locations of anchors' neighbors (N=50, R=4).

After the initialization stage for all the network nodes is complete, each node has an initial estimated location as illustrated in Figure 5.19 These locations are to be used in the SOM learning as initial weights.

For this communication range, all of the nodes have neighbors, this, of course, will help the nodes to estimate their locations based on neighborhood information and totally get benefit from the proposed algorithm.

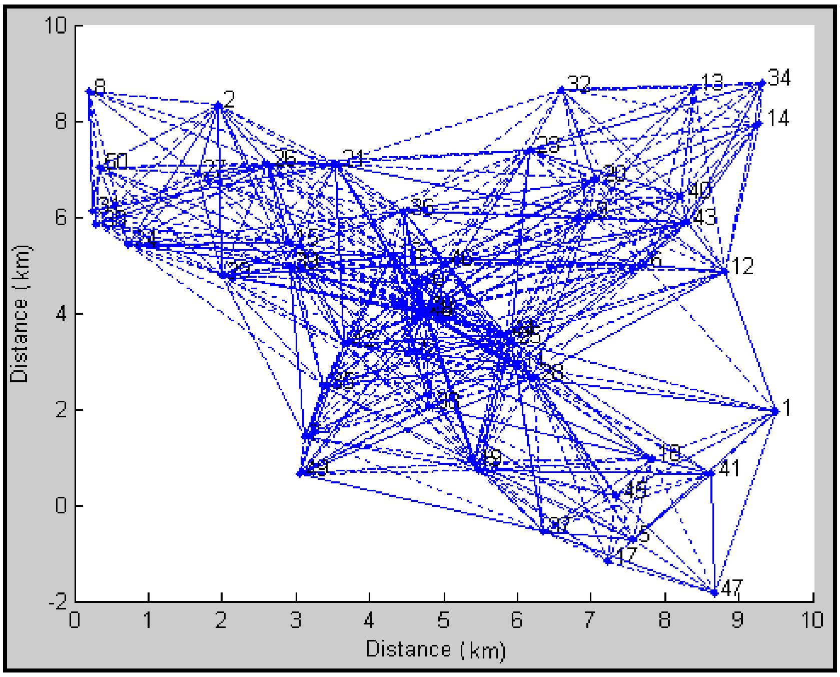


Figure 5.19 The estimated locations for the network nodes (N=50, R=4)

.

But, because of the existence of the averaging step during the initialization stage, and because of the similarity of neighbors for some nodes, the final estimated locations for these nodes will be –nearly- the same. This will lead to the agglomeration noticed in the previous figure.

This problem occurred in most of multiteraion-based methods, in which a node can estimate its location by averaging the received locations. This is due the accumulation of the error in previous estimations.

As a result of the SOM learning, the final estimated locations (weights) are shown in Figure 5.20:

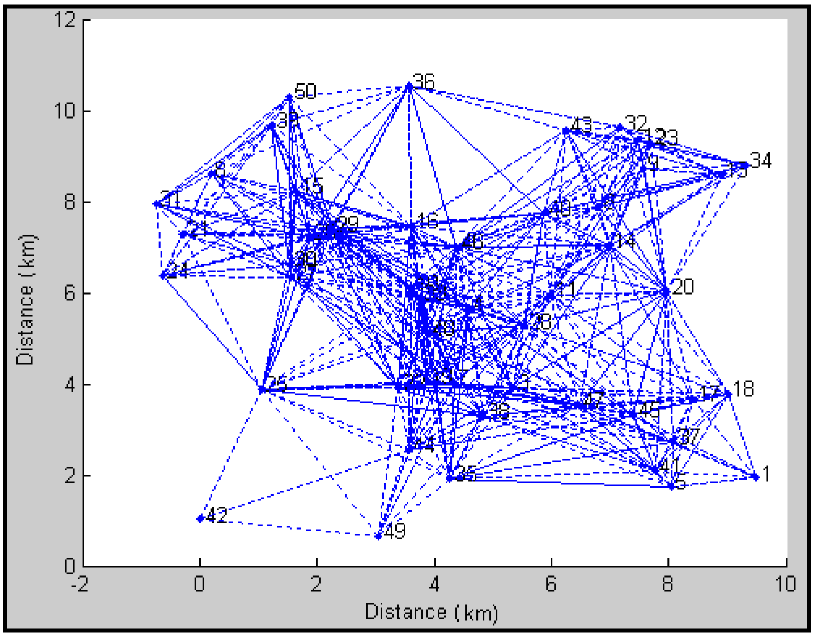


Figure 5.20 The resulted estimated locations for the network nodes (N=50, R=4).

Here, we noticed that the accuracy of the algorithm is lower than the previous case, contrary to the expectations. This can be explained due to the agglomeration happened during the initialization stage. The number of iterations is about 28.

5.2.3 A 100-Nodes wireless Network

The wireless network is generated randomly with 100 wireless nodes in an area of 10km×10km. Three networks with different communication ranges, 1km, 2km, and 4km have been used.

The anchors are selected to be nodes 8 (top-left), 18 (bottom-right), 24 (bottom-left), and 100 (top-right). For the communication range 1, anchors have low number of neighbors due to low radio range.

After the initialization stage for all the network nodes is complete, each node has an initial estimated location. These locations are to be used in the SOM learning as initial weights. Some of the inner nodes, that have neighbors, estimate their locations based on received data from adjacent nodes. As a result of the SOM learning, the final estimated locations showed that the accuracy of the result is relatively low (many nodes shared the same location), and also, the number of iterations needed to reach convergence (stable results) is high (took about 45 iterations). Although the communication range is very low, the problem of agglomeration slightly occurred and - besides the low communication range- influenced the accuracy of the algorithm. This is due the large number of nodes deployed in a limited area.

For the communication range 2, most anchors have neighbors that can estimate their locations based on anchors' sent data. After the initialization stage for all the network nodes is complete, each node has an initial estimated location that can be used in the SOM learning as initial weights. The problem of agglomeration occurred; this is due to the large number of nodes.

As a result of the SOM learning, the final estimated locations (weights) showed that the accuracy of the result is higher than the previous case (the difference of estimated and actual locations of nodes decreased), and also, the number of iterations needed to reach convergence (stable results) is lower (took about 36 iterations).

For the communication range 4, anchors have high number of neighbors that can estimate their locations with taking advantage of received data. After the initialization stage for all the network nodes is complete, each node has an initial estimated location to be used in the SOM learning as initial weights.

Again, the agglomeration problem occurred with a high degree due to the high communication range and the large number of nodes. As a result of the SOM learning, the accuracy of the result in this case is relatively low (many nodes share the same location and have noticeable difference of the actual locations) and the number of iterations is about 40.

Table 5.1 summarizes the results of the three network types that have been used in simulations. The best case is when the number of nodes is 50 with radio range of 2, in which the average estimation error is about 0.228 whereas the worst case is when the number of nodes is 10 with radio range of 1, in which the average estimation error is about 0.45.

Table 5.1 The simulation results for different set of parameters.

Avg. Error	**Number of Iterations**	**Rradio Range**	**Number of Nodes**
0.4500046	50	1	10
0.341468	35	2	10
0.263515	25	4	10
0.401948	35	1	50
0.227984	25	2	50
0.295691	28	4	50
0.389915	45	1	100
0.276799	36	2	100
0.316757	40	4	100

5.2.4 Selecting the Anchors at the Four Centers

To see the effect of changing the four anchors placement on DLSOM, we tried to select the anchors at the four centers of the network. The following Table 5.2 shows the average error results by applying this change - placement - on the same experimental cases described earlier.

Table 5.2 The simulation results for selecting anchors at the four centers.

Avg. Error	**Radio Range**	**Number of Nodes**
0.4714	1	10
0.3176	2	10
0.683	4	10
0.721	1	50
0.5279	2	50
0.693	4	50
0.689	1	100
0.6567	2	100
0.812	4	100

As shown in the previous table, the average error by placing the anchors at the four centers is much larger in comparison with the results shown in Table 5.1. This can be explained as: the anchors at the perimeter of the network are more correlated to nodes than the four-center anchors. Moreover, the equations used to estimate the locations of anchors' neighbors - explained in chapter 4 - are more accurate; for all the anchors at the centers of the network, the estimation equation that can be used is:

$$(x\,,\,y) = (\text{random}\,[x_{anch}\text{-}R,\, x_{anch}+R],\ \text{random}\,[y_{anch}\text{-}R,\, y_{anch}+R]) \qquad (5.1)$$

where $(x\,,\,y)$ are the coordinates of all the anchors' neighbors, $(x_{anch},\, y_{anch})$ are the coordinates of any of the four anchors, and R is the communication range. Obviously, the period of randomization for location estimation is unified and larger in contrast with the four different periods used before.

Thus, based on this experiment results and previous studies, the anchors are selected at the perimeter of the network to increase the DLSOM accuracy.

5.3 Evaluation Parameters

The evaluation parameter that is used in this thesis is the average error between the estimated locations resulted from the DLSOM algorithm and the actual locations. This error is calculated according to the following equation:

$$Error(R) = \frac{1}{N}\sum_{i=1}^{N}\frac{\left|w_i^e - w_i^a\right|}{R} \qquad (5.2)$$

where R is the communication range, N is the number of nodes, w_i^e is the i-th node estimated location, and w_i^a is the i-th node actual location.

5.3.1 Performance of DLSOM

To evaluate the performance of DLSOM, the average error is calculated for each of the experimental cases. Figures 5.21, 5.22, and 5.23 show the relationship between the number of nodes and the average error calculated.

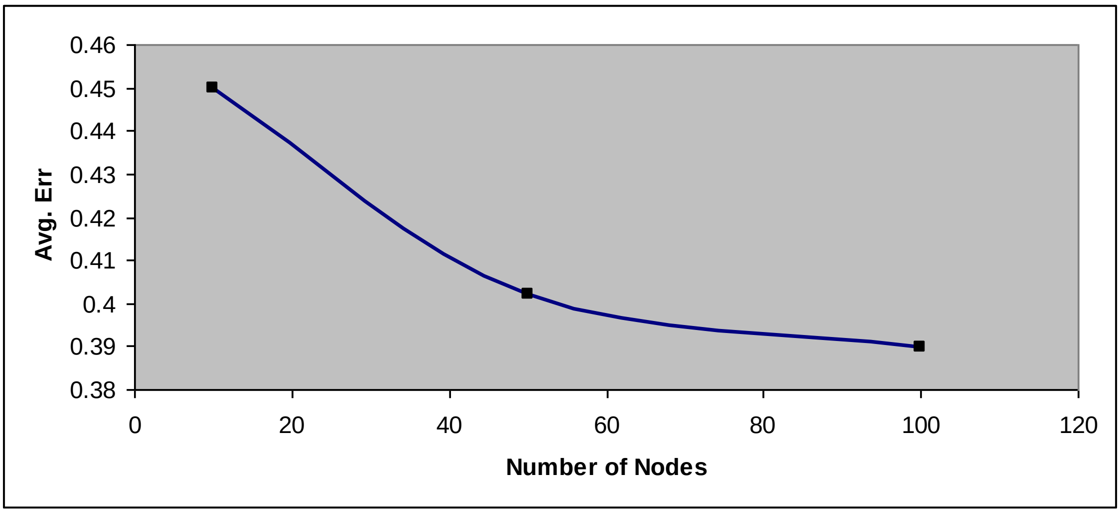


Figure 5.21 Number of network nodes Vs. Average error (R=1).

For radio range 1, by increasing the number of nodes, the average error is decreasing. This can be explained as: by increasing the number of nodes with a low radio range, the neighborhood information increases slightly, and thus, the DLSOM algorithm performs better.

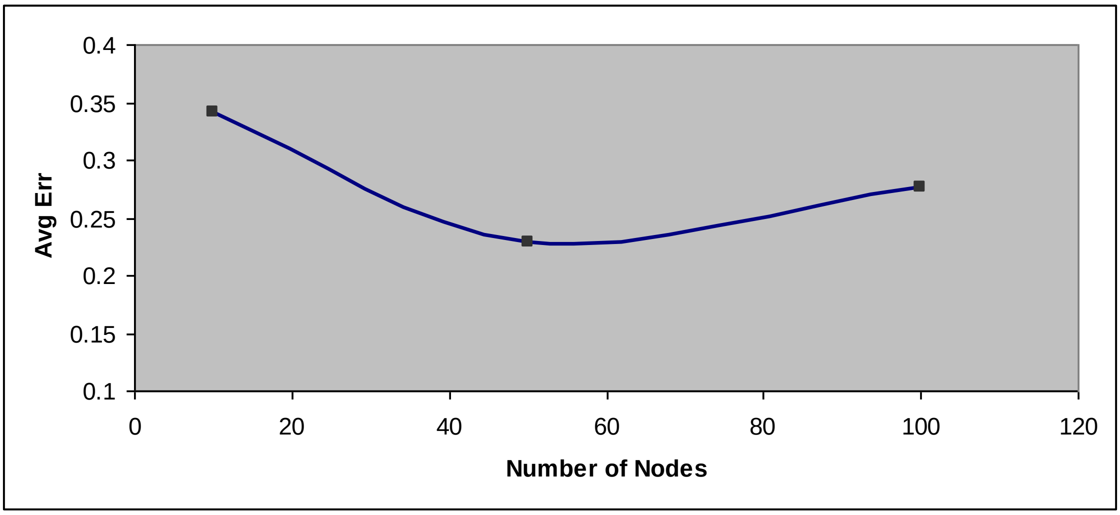


Figure 5.22 Number of network nodes Vs. Average error (R=2).

For radio range 2, by increasing the number of nodes, the average error is decreasing and then increasing. This can be explained as: by increasing the number of nodes with a medium radio range, the neighborhood information increases, and thus, the DLSOM algorithm performs better. But, by deploying a large number of nodes in a limited area with this radio range, the agglomeration problem – described earlier - occurs and the DLSOM performance decreases slightly.

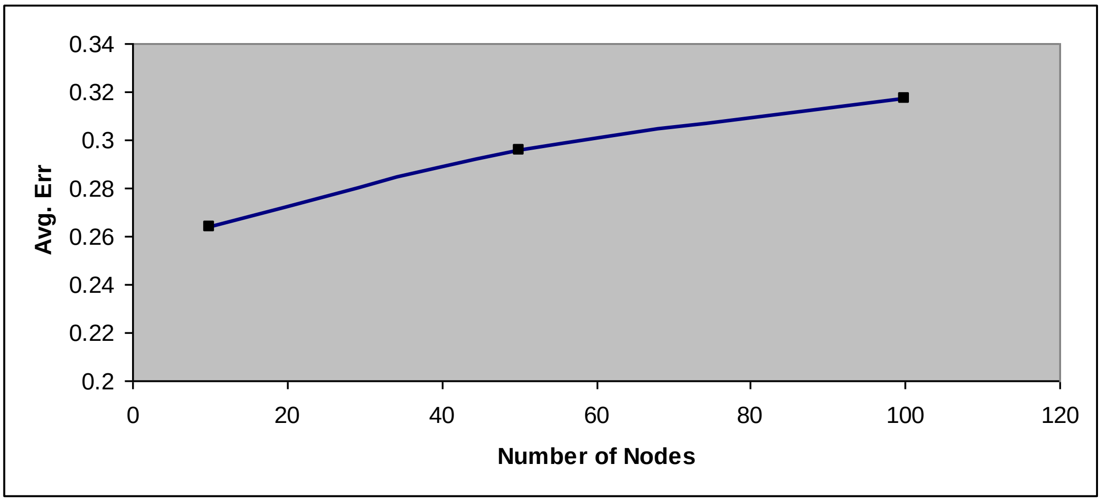


Figure 5.23 Number of network nodes Vs. Average error (R=4).

For radio range 4, by increasing the number of nodes, the average error is increasing. This can be explained as: by increasing the number of nodes with a large radio range in a limited area, the agglomeration problem occurs and the DLSOM performance decreases slightly.

In Figures 5.24, 5.25, and 5.26, the effect of radio range variance is illustrated:

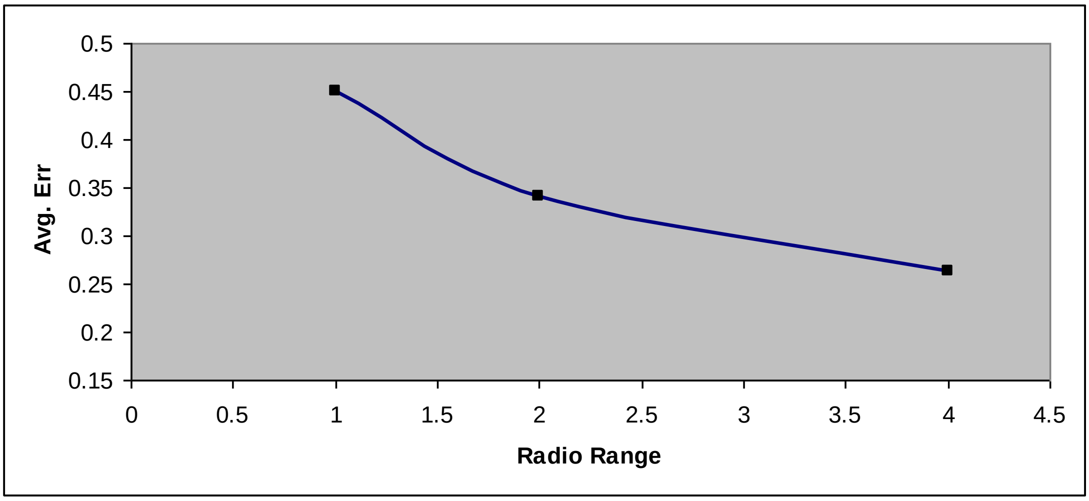


Figure 5.24 Radio range Vs. Average error (N=10).

For low number of nodes, by increasing radio range, the average error decreases. This is due to the neighborhood information support that increases with high radio ranges.

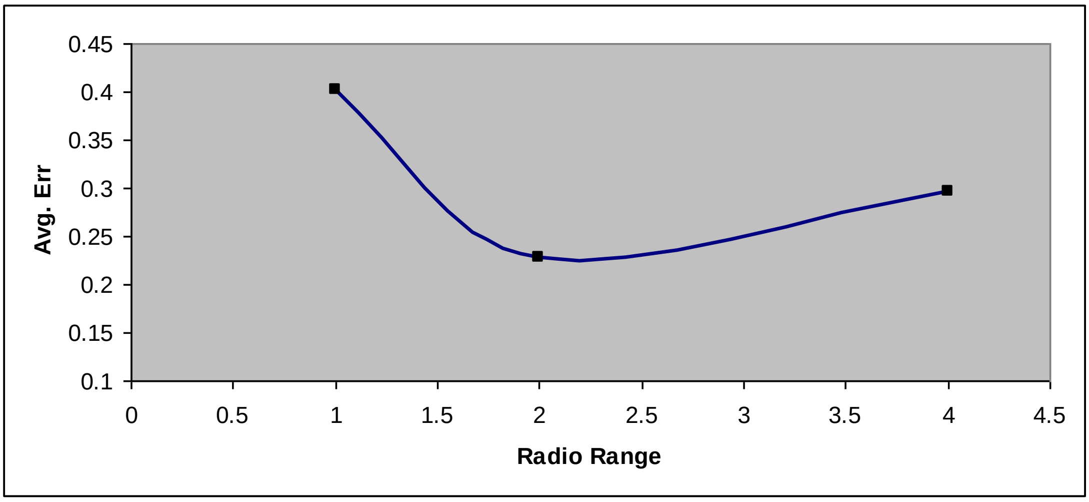


Figure 5.25 Radio range Vs. Average error (N=50).

For medium number of nodes, by increasing radio range, the average error decreases and then increases. This is – again - due to the occurrence of the agglomeration problem.

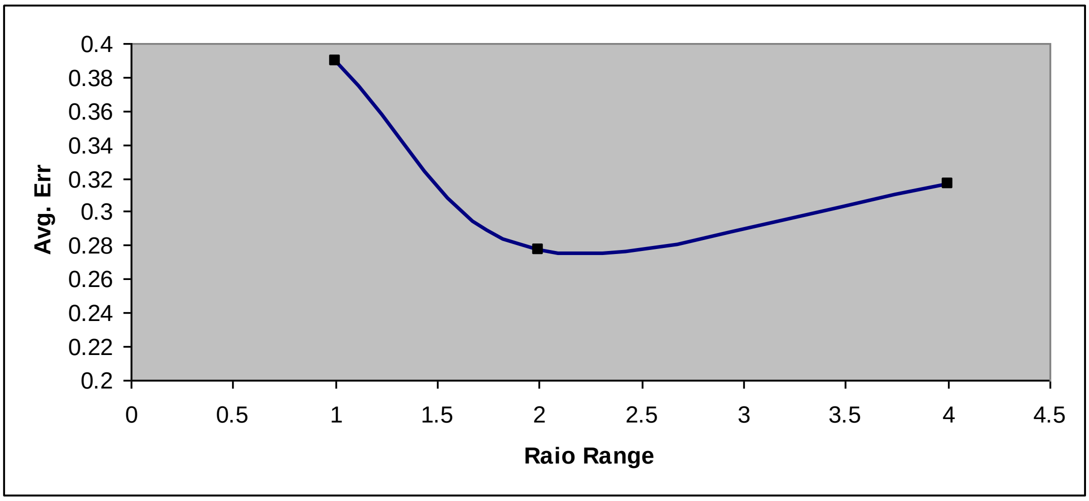


Figure 5.26 Radio range Vs. Average error (N=100).

Also, for large number of nodes, by increasing radio range, the average error decreases and then increases because of the occurrence of the agglomeration problem.

5.3.2 Time Analysis

To evaluate the speed of the algorithm, the time elapsed to execute the DLSOM algorithm has been measured for each of the simulation experimental cases.

Figures 5.27, 5.28, and 5.29 illustrate the relationship between the number of nodes and time elapsed to execute the algorithm:

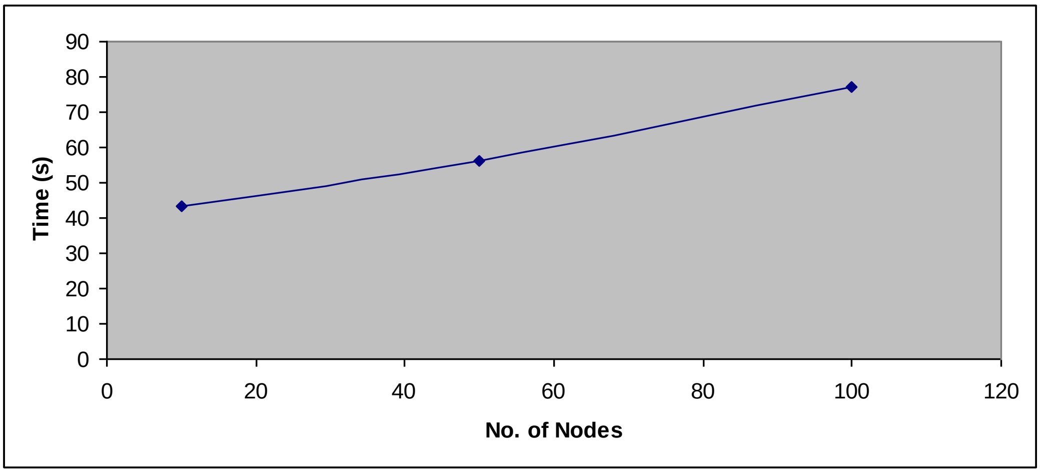


Figure 5.27 No. of Nodes Vs. Time (R=1).

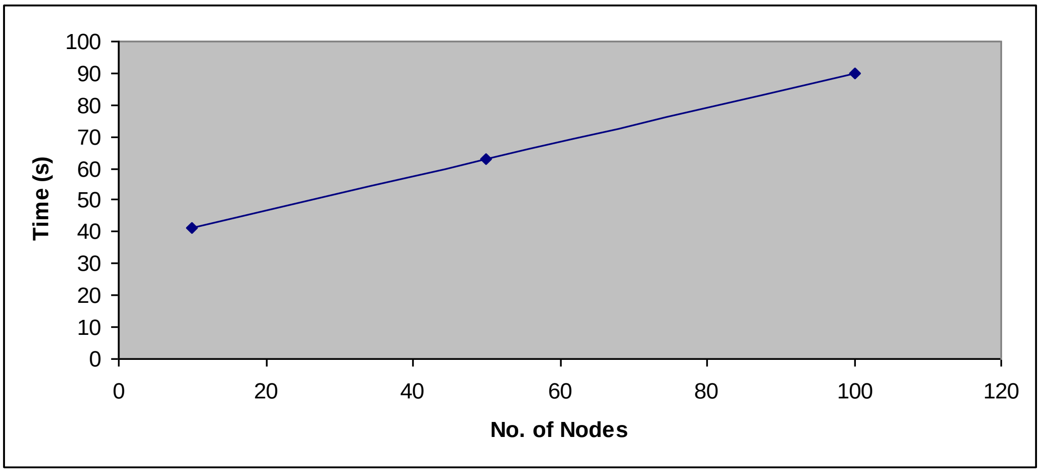


Figure 5.28 No. of Nodes Vs. Time (R=2).

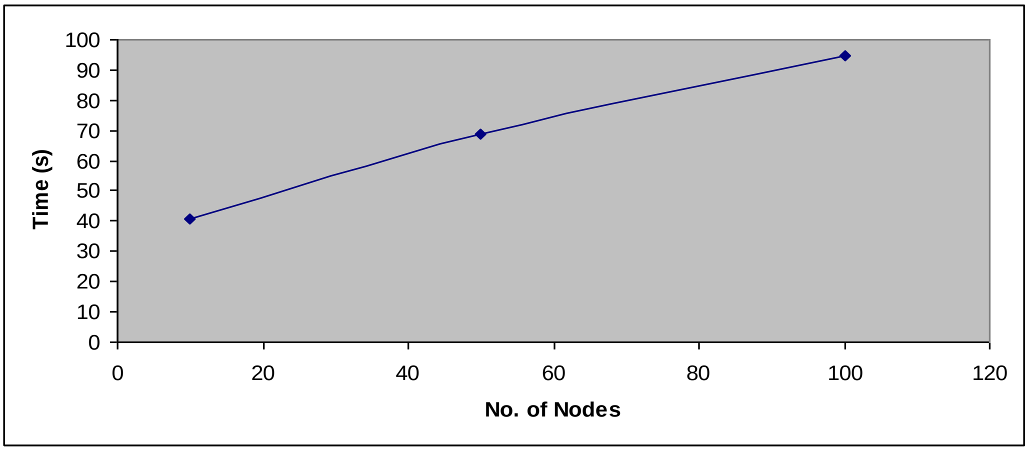


Figure 5.29 No. of Nodes Vs. Time (R=4).

As observed from the previous figures, the time needed to execute the DLSOM algorithm increases linearly by increasing the number of nodes. The time spent in the initialization stage is constant and short. This is because this stage executes for one time (not iterative) and the calculations are fast and simple.

Most of the time is spent in the learning stage (iterative process), in which the nodes transmit and receive location updates to neighbors at each iteration. This time, of course, increases by increasing the number of nodes.

For example, in Figure 5.27, for radio range 1, the time needed to execute the algorithm is about 56 seconds. About 12 seconds only are spent in the initialization stage and 44 seconds in the learning stage.

Figures 5.30, 5.31, and 5.32 illustrate the relationship between the radio range variance and the time elapsed to execute the algorithm.

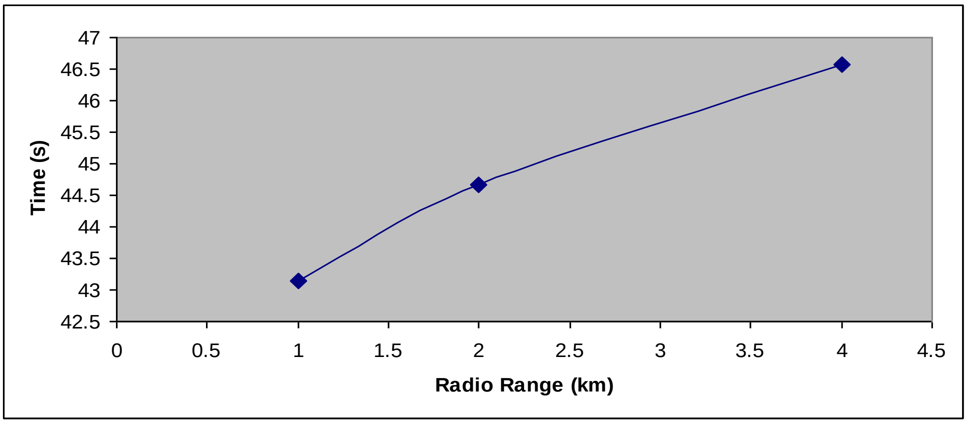


Figure 5.30 Radio Range Vs. Time (N=10).

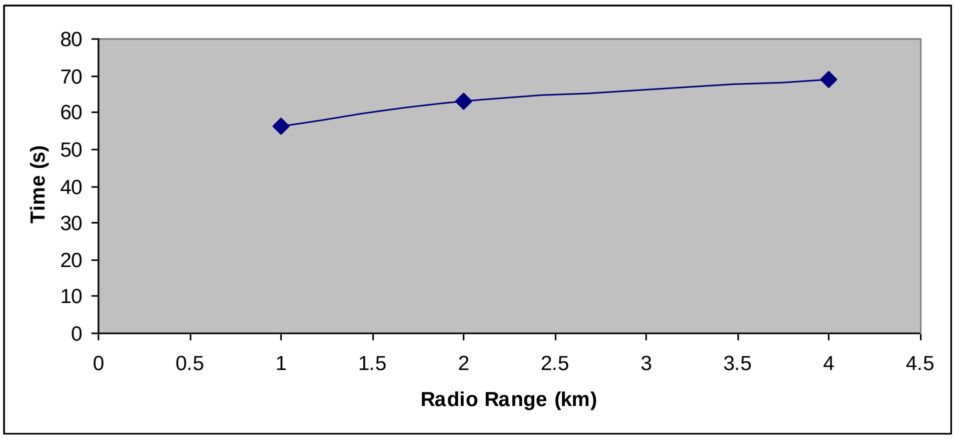


Figure 5.31 Radio Range Vs. Time (N=50).

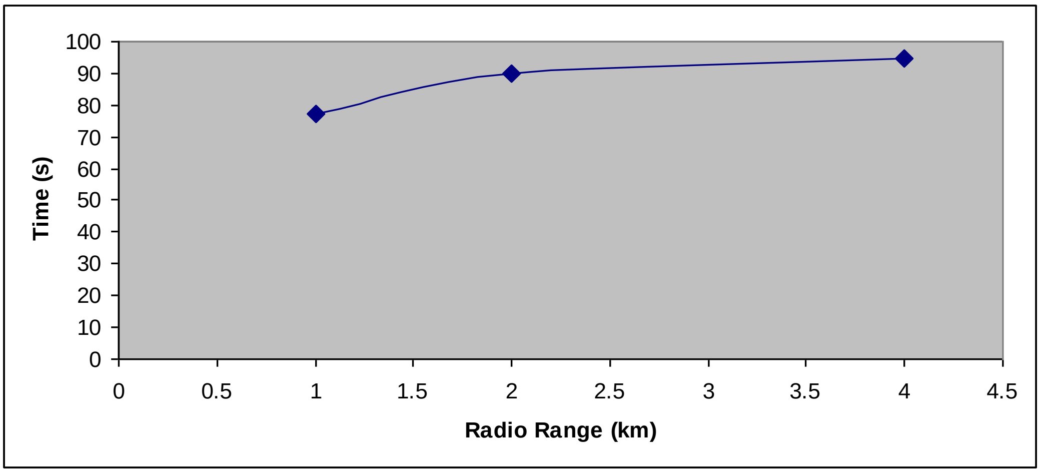


Figure 5.32 Radio Range Vs. Time (N=100).

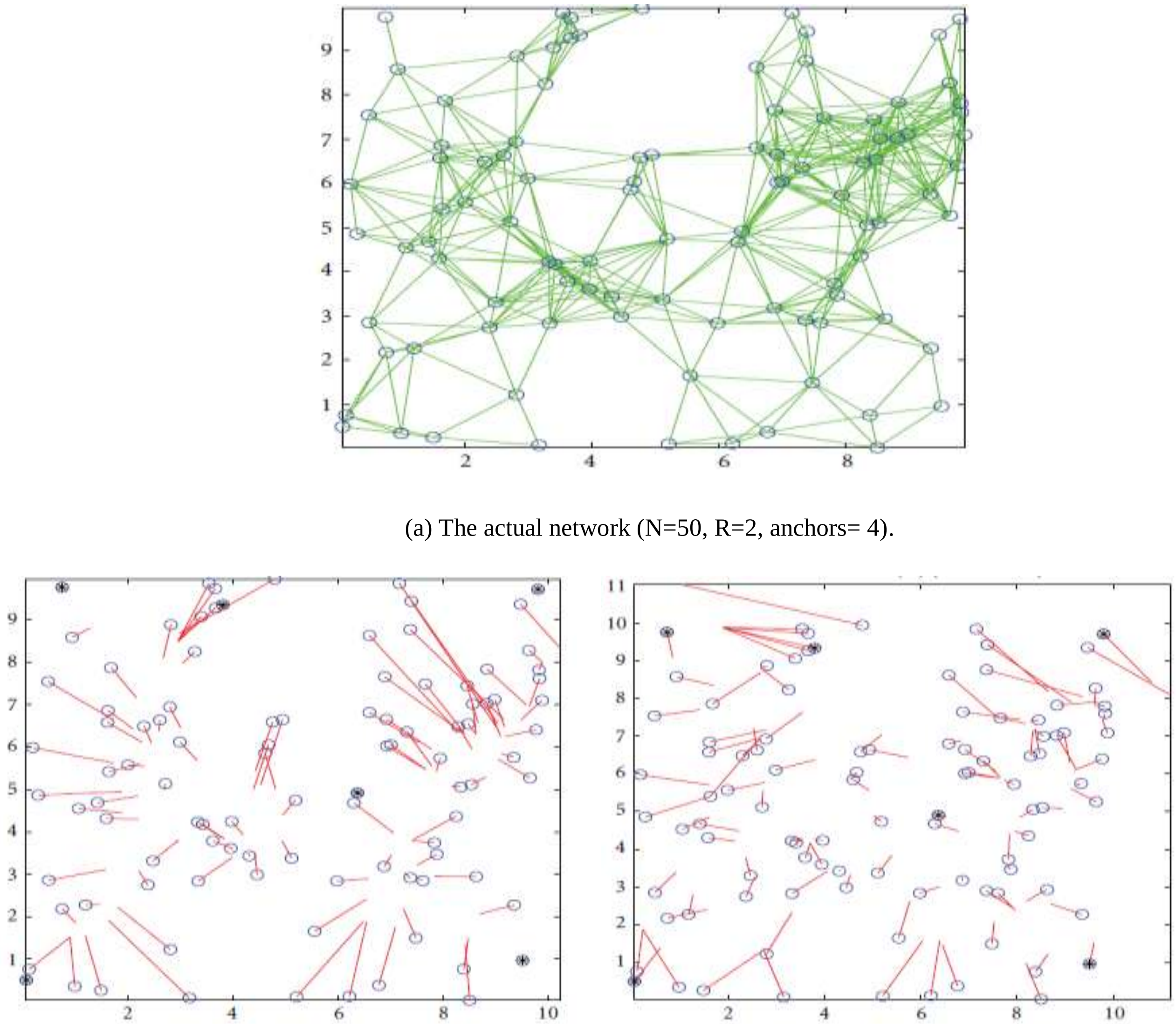
Also, as observed from the previous figures, the time needed to execute the DLSOM algorithm increases linearly by increasing the radio range. This is because by increasing the radio range, more transmission and receiving processes are done by the increment of neighboring nodes. These processes take most of the time.

In all previous figures, although the number of iterations decreases by number of nodes and radio range increment, the time spent increases. This slight increment does not affect the performance of the algorithm; DLSOM is considered to be fast in all cases.

5.3.3 Performance Comparisons

The DLSOM algorithm is compared to many similar algorithms that are used to localize the wireless nodes in an ad-hoc wireless network. The comparable parameter chosen is the average error and some expressive figures in a unified set of simulation parameters.

For 50 wireless nodes randomly deployed in an area of 10km×10km with a radio range of 2km and four anchors selected at the network perimeter, Figure 5.33 [57] illustrates the actual network and the different estimated networks resulted from applying the SOM and MDS-MAP, respectively.



(a) The actual network (N=50, R=2, anchors= 4).

(b) SOM.

(c) MDS-MAP.

Figure 5.33 Resulted estimated networks by applying SOM and MDS-MAP (N=50, R=2).

The following Table 5.3 summarizes the average error calculated for each of the previous estimated networks. These results are the average of 30 different trials of the same set of parameters.

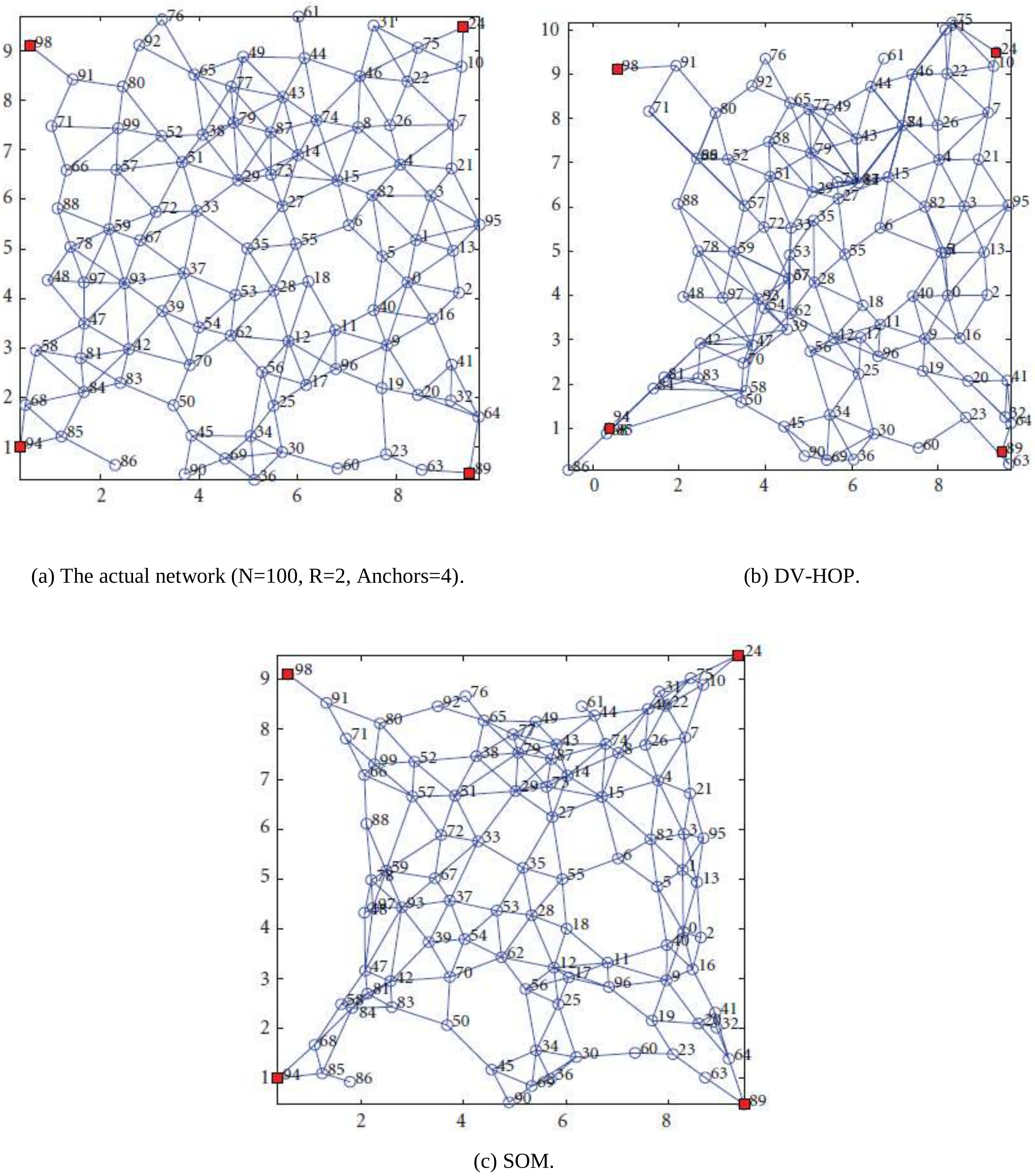
Table 5.3 The average error for different algorithms (N=50, R=2).

Avg. Error	**Radio Range**	**No. of Nodes**	**Algorithm**
0.46	2	50	**SOM**
0.38	2	50	**MDS-MAP**
0.27	2	50	**DLSOM**

Obviously, the DLSOM algorithm has the least average error over the other algorithms with a noticeable variance. It is worth mentioning that especially in this case – 50 Nodes and range of 2 - , the DLSOM performs the best.

Also, comparing the resulted estimated networks by applying the SOM and the MDS-MAP algorithms to the one resulted by applying the DLSOM algorithm in a unified set of simulation parameters, the DLSOM network is observed to be the most similar to the original network (actual network shown in Figure 5.11 (b) and estimated using DLSOM is shown in Figure 5.17).

In Figure 5.34 [54], 100 wireless nodes are randomly deployed in an area of 10km×10km with a radio range of 2km and four anchors selected at the ends of the network. The actual network and the different estimated networks resulted from applying DV-HOP and SOM are illustrated in Figure 5.34 (a), (b) and (c), respectively.



(a) The actual network (N=100, R=2, Anchors=4).

(b) DV-HOP.

(c) SOM.

Figure 5.34 Resulted estimated networks by applying DV-HOP and SOM (N=100, R=2).

The following Table 5.4 summarizes the average error calculated for each of the previous estimated networks. These results are the average of 50 different trials of the same set of parameters.

Table 5.4 The average error for different algorithms (N=100,R=2).

Avg. Error	Radio Range	No. of Nodes	Algorithm
0.50	2	100	**DV-HOP**
0.35	2	100	**SOM**
0.30	2	100	**DLSOM**

Again, the DLSOM algorithm has the least average error over the other algorithms with a noticeable variance. And, comparing the resulted estimated networks by applying the SOM and the DV-HOP algorithms to the one resulted by applying the DLSOM algorithm in a unified set of simulation parameters, the DLSOM network is observed to be the most similar to the original network.

Figures 5.35 and 5.36 show the actual and estimated networks by applying DLSOM in the same set of the previous parameters:

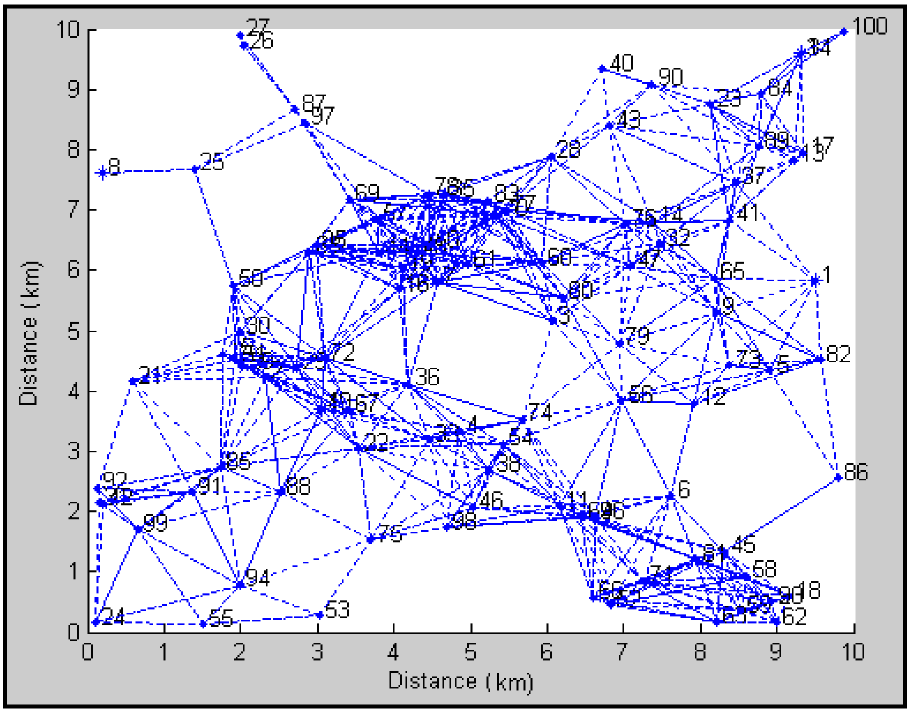


Figure 5.35 Actual Network (N=100, R=2).

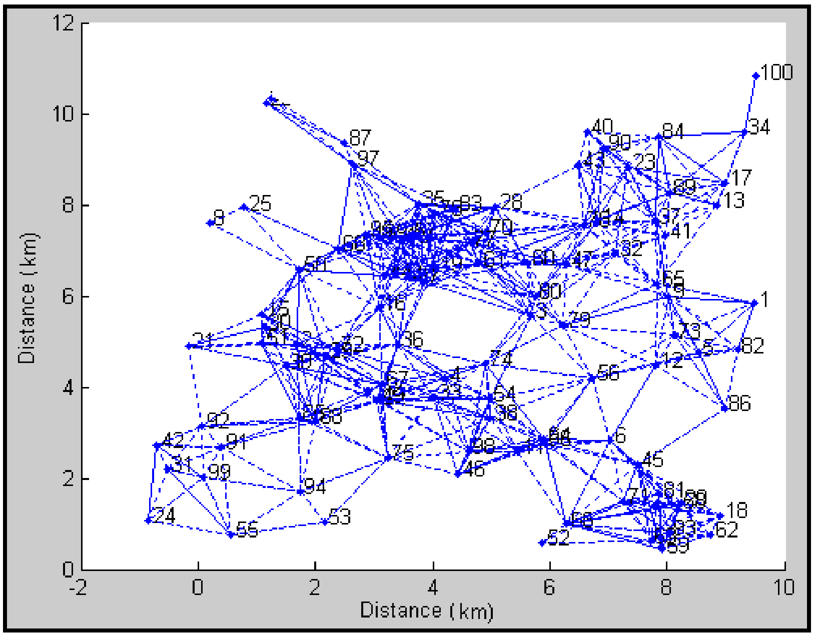
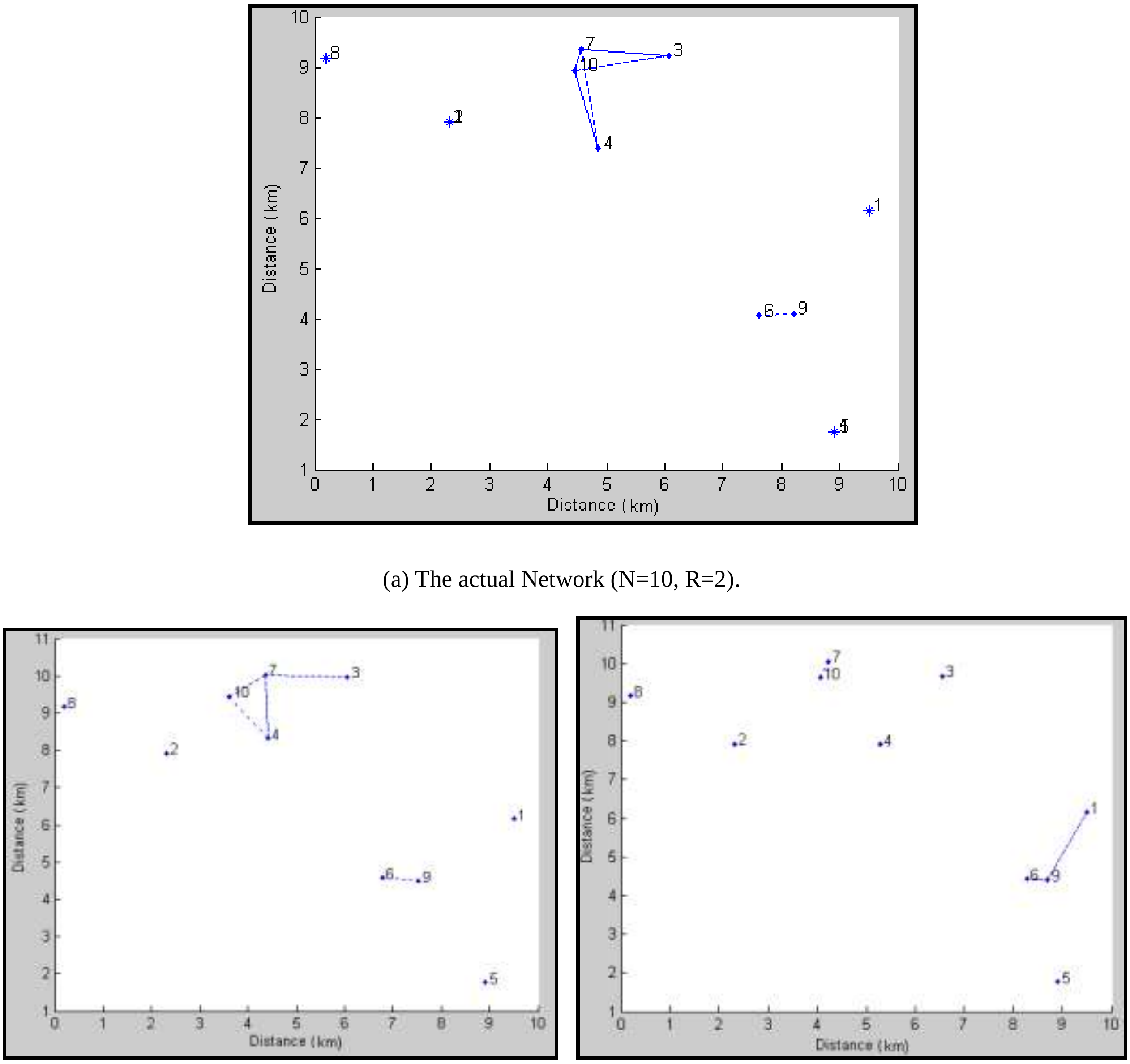


Figure 5.36 Resulted estimated Network by DLSOM (N=100, R=2).

For low radio ranges, low number of nodes, and irregular network topology (random), some algorithms proved their high performance over the others. In [52], a classical SOM localization algorithm is used and produced accurate results in comparison with the others. Here, we will prove that in these situations our algorithm performs better.

We have implemented the classical SOM algorithm used in [52] on the same hardware/software environment that we used to implement our algorithm. Figure 5.37 illustrates the actual network and the resulted estimated networks by applying the classical SOM and the DLSOM algorithms.

The original network is a 10 wireless nodes deployed randomly in an area of 10km×10km with a radio range of 2km.



(a) The actual Network (N=10, R=2).

(b) DLSOM.

(c) Localized SOM.

Figure 5.37 Performance comparison by applying DLSOM and Localized SOM.

The figure shows that the DLSOM performs better with the similarity to the original network and with the less average error calculated for both algorithms (showed in Table 5.5).

Moreover, the classical SOM algorithm is centralized while the DLSOM is distributed. Hence, the wireless network nodes' computation overhead is reduced significantly, and also the number of iterations took in the DLSOM is about 35 which is not mentioned in comparison to the thousand iterations took by the classical localized SOM.

The following Table 5.5 summarizes the average error calculated for DLSOM and Localized SOM of the previous estimated networks. These results are the average of 50 different trials of the same set of parameters.

Table 5.5 The average error for DLSOM and Localized SOM (N=10, R=2).

Avg. Error	**Radio Range**	**No. of Nodes**	**Algorithm**
0.36	2	10	**DLSOM**
0.38	2	10	**Localized SOM**

With about 2% less percentage error of DLSOM in comparison with localized SOM, DLSOM approved its effectiveness even in WSNs with low number of nodes.

Chapter 6

Conclusion

6.1 Summary and Concluding Remarks

In this thesis, a Self Organizing Maps (SOMs) - based distributed localization algorithm is proposed (DLSOM). The main objective of this algorithm is to calculate the locations of nodes in wireless sensor networks.

The intelligent SOM neural networks are selected due to their multiple characteristics over other types of neural networks. One of the most important characteristics of SOMs is their unsupervised training fashion, in which no reward or cost functions are needed. The other feature is that SOMs provides a technique for representation of multidimensional data into much lower-dimensional spaces. Moreover, the arrangement of neurons into a grid increases the accuracy of the results.

The proposed localization algorithm aimed to get benefit from the neighborhood information that can be gathered fast and easy by the wireless nodes (each node knows its neighbors based on the communication radio range). Thus, no communication overhead occurs and the usage of this information in the initialization stage of the algorithm significantly helped SOMs to begin with useful initial data to be used in the learning stage. Hence, the learning time and the number of iterations took by SOMs to reach stable results have been noticeably decreased. Also, the accuracy of the results increased and the output locations are the most approximate in comparison with the previous related algorithms.

The experimentation and simulation results proved the effectiveness of the DLSOM algorithm in different simulation parameters. The criterion that has been calculated to evaluate and compare the performance of the algorithm over other algorithms is the average error.

The average error of the DLSOM is found to be the least in most cases in comparison with some related algorithms that proved their accuracy in this field. The accuracy of DLSOM which is directly related to the average error is also found to be the most. The total average error (calculated by averaging errors using all experiments) is about

30%. This percentage is much lower than the other algorithms. Moreover, the number of iterations needed in DLSOM ranges from 25 to 50. This number is significantly much smaller in comparison with the hundreds to thousands iterations needed by other algorithms which reduces the learning time substantially.

Experimentations showed that the case in which the DLSOM performs the best is with the set of parameters (50 wireless nodes and radio range of 2). The performance gets worse and worse by increasing the number of nodes in a limited area and also increasing the radio range with this large number of nodes. But in all cases, the algorithms proved its effectiveness over the previous contributions either in low or high number of nodes. Referring to the best case mentioned before, this can be considered as the most important advantage of the algorithm because most of the real wireless networks deployed in a limited area (10km×10km) usually contains no more 50 nodes with a communication range of 2km as an average.

6.2 Recommendations and Future Work

During the experimentations and validation of our proposed algorithm, the problem of agglomeration has been occurred in some cases, in which the nodes have approximately the same initial estimated location due to the averaging step in the initialization stage. These cases include the high number of network nodes deployed in a limited area and have a relatively high communication radio range.

This may be solved by selecting the nearest three - or more (based on the total number of nodes) - neighbors to be averaged to get the estimated location. In this way, the nodes that have the same set of neighbors will be enforced to choose the nearest subset and hence will get a distinct estimated location.

Another suggested solution is to replace the averaging process step with a more complex and distinguishing mathematical process, such as trilateration method used in GPS systems to locate some wireless node, in which three known-location nodes are used to estimate the unknown node location with a series of complex mathematical equations. On the other hand, trilateration could be more accurate to calculate the initial estimated locations of wireless nodes.

In this thesis research, we investigated a proposed distributed localization algorithm that has been applied on static wireless sensor networks, in which the nodes have no movement.

For mobile wireless networks, in which the nodes move with specific parameters (motion speed and motion direction), the proposed localization algorithm can be modified to be applied on these networks. This can be done by refreshing the set of nodes' neighbors periodically and use the most updated neighbors list in the learning stage to get the most approximate location of the node.

The period of refreshing is determined based on the motion speed of the wireless nodes in the network.

In mobile networks, the accuracy of the algorithm is supposed to be less than static networks due to the continuous movement and different neighborhood information that change rapidly, and hence influence the principle of the algorithm.

References

[1] H. Karl and A. Willig, "Protocols and Architectures for Wireless Sensor Networks", West Sussex, England, John Wiley and Sons Ltd, 2005.

[2] D. Culler, D. Estrin, and M. Srivastava, “Overview of Sensor Networks”, *IEEE Computer Journal*, vol. 37, pp. 41 - 49, Aug. 2004.

[3] K. Martinez, J. K. Hart, and R. Ong, “Environmental sensor networks”, *IEEE Computer Journal*, vol. 37, pp. 50-56, Aug. 2004.

[4] A. Mainwaring, D. Culler, J. Polastre, R. Szewczyk, and J. Anderson, “Wireless sensor networks for habitat monitoring”, *in Proceedings of the 1st ACM International workshop on Wireless sensor networks and applications*, Atlanta, Georgia, USA, 2002, pp. 88-97.

[5] I. F. Akyildiz, D. Pompili and T. Melodia, “Underwater acoustic sensor networks: research challenges”, *Ad Hoc Networks*, vol. 3, pp. 257-279, May. 2005.

[6] Dargie, W. and Poellabauer, C., "Fundamentals of wireless sensor networks: theory and practice", West Sussex, England, John Wiley and Sons, 2007, ch.6, sec. 2, pp. 168–183, 191–192

[7] Sohraby, K., Minoli, D., Znati, T. "Wireless sensor networks: technology, protocols, and applications", West Sussex, England, John Wiley and Sons, 2007, ch.5, sec. 3, pp. 203–209

[8] (2011, May), Wireless Sensor Networks, [Online]. Available: http://en.wikipedia.org/wiki/Wireless_sensor_network

[9] K. Delin, “The Sensor Web: A Macro-Instrument for Coordinated Sensing”, *Sensors*, vol. 2, pp. 270-285, Jul. 2002.

[10] Yick, J., Mukherjee, B., Ghosal, D, ”Wireless sensor network survey", *Computer Networks*, vol. 52, no. 12, pp. 2292-2330, April. 2008

[11] B. Karp and H. Kung., “GPSR: greedy perimeter stateless routing for wireless networks”, *in Mobile computing and networking conference*, Boston, MA, USA, 2000, pp. 243-254.

[12] Hefeeda, M., Bagheri, M., “Wireless sensor networks for early detection of forest fires”, *in IEEE International Conference on Mobile Adhoc and Sensor Systems*, Pisa, 2008, pp. 1-6.

[13] Pace, S., Frost, G., Lachow, l., Frelinger, D., Fossum, D., Wassem, D.K., Pinto, M., “GPS history chronology and budgets”, in The global positioning system, Santa Monica, RAND Corporation, 1995, Appendix B

[14] Pandey, S., Agrawal, P., “A Survey on Localization Techniques for Wireless Networks”, *Journal of the Chinese Institute of Engineers*, vol. 29, no. 7, pp. 1125-1148, 2006

[15] Mo, L., Yunhao, L.: Rendered path, "Range-free localization in anisotropic sensor networks with holes”, *IEEE/ACM Trans*. Netw, vol. 18, no. 1, pp. 320-332, 2010

[16] Zhetao Li, Renfa Li, Yehua Wei and Tingrui Pei, "Survey of Localization Techniques in Wireless Sensor Networks", *Information Technology Journal*, vol. 9, pp. 1754-1757, Sep. 2010.

[17] Juang, P., Oki, H., Wang, Y., Martonosi, M., Peh, L.S., “Energy-efficient computing for wildlife tracking: Design tradeoffs and early experiences with ZebraNet”, *ACM*, vol. 37, pp. 96-107, 2002

[18] Yihan, L., Panwar, S.S., Shiwen, M., Burugupalli, S. Jong ha, L., “A mobile ad hoc biosensor network”, *in IEEE International Conference on Communications*, NY, USA, 2005, vol.2, pp. 1241 – 1245

[19] Peng, R., Sichitiu, M.L., “Localization of wireless sensor networks with a mobile beacon”, *in IEEE Conference on Mobile Ad-hoc and Sensor Systems*, NC, USA, 2004, pp. 174 – 183

[20] Nalini Vasudevan, "Node Localization in Wireless Ad-Hoc Sensor Networks", R.V. College of Engineering, Bangalore, *Tech. Rep*. 52, June. 2005

[21] R. Poovendran, C. L.Wang, and S. Roy, “Secure Localization and Time Synchronization for Wireless Sensor and Ad Hoc Networks”, *Springer*, 2007.

[22] B. H. Wellenhoff, H. Lichtenegger, and J. Collins, “Global Positioning System: Theory and Practice”, 5th edition, *Springer*, 2001.

[23] A. Savvides, C. C. Han, and M. B. Strivastava, “Dynamic finegrained localization in ad-hoc networks of sensors”, *in Mobile Computing and Networking conference*, New York, NY, USA , 2001, pp. 225-246.

[24] D. Niculescu and B. Nath, “Ad hoc positioning system (APS) using AOA”, *IEEE Computer*, vol. 3, pp. 1734 - 1743, April. 2003.

[25] N. Patwari, A. O. Hero III, M. Perkins, N. S. Correal, and R. J. O’Dea, “Relative location estimation in wireless sensor networks”, *IEEE Transactions on Signal Processing*, pp. 2137 - 2148, Aug. 2003.

[26] R. Want, A. Hopper, V. Falc˜ao, and J. Gibbons., “The active badge location system”, Cambridge University , *Technical Report*. 92.1, 1992.

[27] N. Priyantha., “Providing precise indoor location information to mobile devices”, M.S. thesis, Massachusetts Institute of Technology, MIT, 2001.

[28] A. Ward, A. Jones, and A. Hopper. “A new location technique for the active office”, *IEEE Personnel Communications*, vol. 4, pp. 42 - 47 , Oct. 1997.

[29] A. Harter, A. Hopper, P. Steggles, A. Ward, P. Webster., “The anatomy of a context-aware application”., *in Proceedings of the 5th Annual ACM/IEEE International Conference on Mobile Computing and Networking*, New York, NY, USA , 1999, pp. 798-815.

[30]. A. Smith, H. Balakrishnan, M. Goraczko, and N. Priyantha., “Tracking moving devices with the cricket location system”, *in ACM conference on Mobile systems*, New York, NY, USA, 2004.

[31] D. Moore, J. Leonard, D. Rus, and S. Teller., “Robust distributed network localization with noisy range measurements”, *ACM Proceedings of the 2nd international conference on Embedded networked sensor systems*, New York, NY, USA, 2004.

[32] Greg Welch, Gary Bishop, Leandra Vicci, Stephen Brumback, Kurtis Keller, and D’nardo Colucci., “High-performance wide-area optical tracking’, *Teleoper. Virtual Environ*, vol. 10, pp. 1-21, Feb. 2001.

[33] C. Alippi, A. Mottarella, and G. Vanini., “A RF map-based localization algorithm for indoor environments”, *IEEE International Symposium on Circuits and Systems*, vol. 1, pp. 652 - 655, May. 2005.

[34] J. Hightower, G. Borriello, and R. Want. SpotON, “An indoor 3D location sensing technology based on RF signal strength”, CSE, *Technical Report*, 2000.

[35] K. Lorincz and M. Welsh, “A robust decentralized approach to RF-based location tracking”, *Technical Report*. TR-04-04, 2004.

[36] L. M. Ni, Y. Liu, Y. C. Lau, and A. P. Patil. Landmarc, “Indoor location sensing using active RFID”, *in IEEE International Conference on Pervasive Computing and Communications*, 2003, pp. 407 - 415.

[37] Dellaert, F., Fox, D., Burgard, W., Thrun, S. “Monte Carlo Localization for Mobile Robots”, *IEEE International Conference on Robotics and Automation*, 1999, vol. 2, pp. 1322 - 1328.

[38] Hu, L., Evans, D., “Localization for mobile sensor networks”, *International Conference on Mobile Computing and Networking*, USA, 2004, pp. 1167-1180.

[39] Baggio, A., Langendoen, K., “Monte Carlo localization for mobile wireless sensor networks”, *Ad Hoc Networks*, vol. 6, pp. 718–733, July. 2007

[40] N. Bulusu, J. Heidemann, and D. Estrin. , “GPS-less low-cost outdoor localization for very small devices”, *IEEE Personal Communications Magazine*, vol. 7, pp. 28-34, Oct. 2000.

[41] D. Niculescu and B. Nath. “DV Based Positioning in Ad Hoc Networks”. *Journal of Telecommunication Systems*, vol. 22, pp. 267-280 , 2003.

[42] R. Nagpal, H. Shrobe, and J. Bachrach. , “Organizing a global coordinate system from local information on an ad hoc sensor network”, *In Proceedings of the 2nd international conference on Information processing in sensor networks (IPSN'03)*, Springer-Verlag, Berlin, Heidelberg, 2003, pp. 333-348.

[43] D. Nicolescu and B. Nath, “Ad-hoc positioning systems (APS),” *in Proceedings of IEEE Global Telecommunications Conference* , San Antonio, USA, 2001, vol. 5, pp. 2926 - 2931.

[44] Y. Shang,W. Ruml, Y. Zhang, and M. Fromherz, “Localization from connectivity in sensor networks,” *IEEE Transactions on Parallel and Distributed Systems*, vol. 15, pp. 961-974, 2004.

[45] S. K. Chenna, Y. Kr. Jain, H. Kapoor, R. S. Bapi, N. Yadaiah, A. Negi, V. Seshagiri Rao, and B. L. Deekshatulu, “State estimation and tracking problems: A comparison between Kalman filter and recurrent neural networks”, *International Conference on Neural Information Processing*, Berlin, 2004, pp. 275-281.

[46] A. Shareef, Yifeng Zhu, Mohamad Musavi, and Bingxin Shen., “Comparison of MLP Neural Networks and Kalman Filter for Localization in Wireless Sensor Networks", *in Proceedings of 19th IASTED International Conference on Parallel and Distributed Computing Systems*, November, 2007, Cambridge, Massachusetts, USA, pp. 323-330.

[47] D. A. Tran and T. Nguyen, “Localization in wireless sensor networks based on support vector machines”, *IEEE Transactions on Parallel and Distributed Systems*, vol. 7, pp. 1278-1290, 2008.

[48] S. Kaski, J. Kangas, and T. Kohonen., “Bibliography of self-organizing map (SOM)”, *Neural Computing Surveys*, vol. 3, pp. 1-156, 1998.

[49] M. Oja, S. Kaski, and T. Kohonen, ” Bibliography of self-organizing map (SOM)”, *Neural Computing Surveys*, vol. 2, pp. 14-167, 2003.

[50] J. Janet, R. Gutierrez, T. Chase, M. White, and J. Sutton., “Autonomous mobile robot global self-localization using Kohonen and region-feature neural networks”, *Journal of Robotic Systems*, vol. 3, pp. 504-509, 1997.

[51] U. Gerecke and N. Sharkey., “Quick and dirty localization for a lost robot. Computational Intelligence in Robotics and Automation”, *IEEE International Symposium*, 1999, vol. 2, pp. 15-26.

[52] G. Giorgetti, S. K. S. Gupta, and G. Manes, “Wireless localization using self-organizing maps”, *In Proceedings of the 6th international conference on Information processing in sensor networks (IPSN '07). ACM*, New York, NY, USA, 2007, pp. 293-302.

[53] S. Asakura, D. Umehara, and M. Kawai, “Distributed location estimation method for mobile terminals based on SOM algorithm", *IEICE Transactions on Communications*, vol. 85, pp. 1042-1050, 2002.

[54] J. Hu and G. Lee, “Distributed localization of wireless sensor networks using self-organizing maps”, *IEEE International Conference on Multisensor Fusion and Integration for Intelligent Systems*, Aug, 2008, pp. 284 - 289.

[55] J. Sum, C. S. Leung, L.W. Chan, and L. Xu, “Yet another algorithm which can generate topograph map,” *IEEE Transactions on Neural Networks*, vol. 6, pp. 548-559, 1997.

[56] J. A. Janet, R. Gutierrez, T. A. Chase, M. W. White, and J. C. Sutton III, “Autonomous mobile robot global self-localization using Kohonen and region-feature neural networks”, *Journal of Robotic Systems*, vol.14, pp. 263-282, 1997.

[57] E. Ertin and K. L. Priddy, “Self-localization of wireless sensor networks using self organizing maps”, *Intelligent Computing: Theory and Applications III*, vol. 5803, pp. 138-145, March. 2005.

[58] D. Estrin et al.," Instrumenting the world with wireless sensor networks", *Proceedings of the International Conference on Acoustics, Speech and Signal Processing (ICASSP 2001)*, Salt Lake City, Utah, May. 2001.

[59] N. Bulusu et al. , "Scalable Coordination for wireless sensor networks: Self- Configuring Localization Systems", *in Proceeding of the Sixth International Symposium on Communication Theory and Applications (ISCTA 2001),* Ambleside, Lake District, UK, July. 2001,.

[60] I.F. Akyildiz et al., "Wireless Sensor Networks : A Survey", Georgia Institute of Technology, Atlanta, *Computer Networks*, vol.38, pp. 393-422, 2002.

[61] D. Estrin et al., "Next Century Challenges: Scalable Coordination in Sensor Networks", *Proceedings of the Fifth Annual International Conference on Mobile Computing and Networks (MobiCOM ’99)*, Seattle, Washington, August, 1999, pp.263-270.

[62] X.Ma. ,"Location Related Issues in Mobile Network Systems", *AD HOC Wireless Networking*, Minnesota, Minneapolis, pp. 365 - 382, 2002.

[63] F.Koushanfar et al., "Location Discovery in Ad-Hoc Wireless Sensor Networks", *AD HOC Wireless Networking*, California, pp. 137 - 174, 2002.

[64] N. Dankwa , "An Evaluation of Transmit Power Levels for Node Localization on the Mica2 Sensor Mote", *New Haven*, 2004.

[65] N. Bulusu at al., "GPS-less Low Cost Outdoor Localization For Very Small Devices", *IEEE Personal Communications*, vol. 7, No. 5, pp. 28-34, Oct. 2000.

[66] T. Kohonen., “Self-organized formation of topologically correct feature maps”, *Biological Cybernetics*, vol.43, pp. 59-69, 1982.

[67] S. Haykin., “Neural Networks”, Prentice Hall, 2nd edition, 1998.

[68] T. Kohonen, “Self-Organizing Maps”, Springer, 3rd edition, 2001.

[69] Teuvo Kohonen and Timo Honkela, "Kohonen network", Scholarpedia, 2(1):1568, 2007.

[70] Jae-Wook Ahn and Sue Yeon Syn , (2012, February, 15), Self-Organizing Maps, [Online]. Available: http://www.sis.pitt.edu/~ssyn/som/som.html.

[71] Kohonen's Self Organizing Feature Maps, (2012, February, 16), [Online]. Available: http://www.ai-junkie.com/ann/som/som.html.

[72] Xinwei Wang A, Ole Bischoff A, Rainer Laur A, and Steffen Paul A, " Localization in Wireless Ad-hoc Sensor Networks using Multilateration with RSSI for Logistic Applications ", Proceedings of the Eurosensors XXIII conference, 2009, pp.461-464.

[73] Yi Shang , Hongchi Shi, Ahmed, A.A, " Performance study of localization methods for ad-hoc sensor networks", Mobile Ad-hoc and Sensor Systems, IEEE International Conference,Oct.,2004,pp.184 - 193

[74] Mehdi Vojdani , Mehdi Dehghan, " Localization in Anchor Less Wireless Sensor Network", International Association of Computer Science & Information Technology (IACSIT), vol.2, Jan, 2012

[75] Zhiheng Xie, Mingyi Hong, Hengchang Liu, Jingyuan Li, Kangyuan Zhu, and John Stankovic. "Quantitative uncertainty-based incremental localization and anchor selection in wireless sensor networks", In Proceedings of the 14th ACM international conference on Modeling, analysis and simulation of wireless and mobile systems (MSWiM '11). ACM, New York, NY, USA, 2011, pp. 417-426.

[76] Han G., Jiang J., Shu L., Xu Y., Wang F., "Localization Algorithms of Underwater Wireless Sensor Networks: A Survey", *Sensors*. Vol.12, no.2, pp.2026-2061, 2012.