MOBILITY MODELING AND TOPOLOGY PREDICTION IN COGNITIVE MOBILE NETWORKS

by

Abdullah Alshehri

A Thesis Submitted in Partial Fulfillment

of the Requirements for the Degree of

Master of Computer Science (MCS)

in

Business and Information Technology

University of Ontario Institute of Technology

Oshawa, Ontario, Canada

(July, 2012)

Abstract

The purpose of this thesis is to analyze a non-intrusive connectivity visualization method for OLSR-based MANET topology in different mobility models. The visualization relies on the local topology databases (neighborhood database and topology database) available in OLSR nodes in the network. Two different views are considered in this method: central view and nodal view. In the central view, the network topology is viewed from a control center which has access to the databases of all nodes, while on the other hand, the nodal visualization provides a picture of the network topology from individual nodes point of view. In this thesis, the full view of the network has been compared to the nodal view to calculate the error rate for topology discovery, based on the total numbers of active and undiscovered links.

The main contribution of this thesis is to analyze and improve the accuracy of coarse localization techniques under different mobility models, using the Force-directed algorithm to calculate the approximate location of the nodes. The localization information was gathered from layer-3 connectivity, utilizing anchor nodes that are equipped with GPS and other non-GPS nodes instead of using traditional methods that include received signal strength, time of arrival and angle of arrival. The approximate location information of the nodes derived from this technique has been compared with original node location in order to determine the accuracy of this technique. To improve the accuracy, several mobility prediction filters such as moving average filter, Kalman filter and low pass filter have been applied to the approximate location data. The simulation is done to calculate the error between the original location data and the coarse approximations, and the results shows that Moving Average provides the best results.

Acknowledgements

I wish to express my sincere thankful to my supervisor Dr. Shahram S. Heydari for his guidelines, motivation engorgement and patient. I am as well appreciative to Faizul Islam and Md. Mustafizur Rahman, for their help and support throughout my research.

I also can't describe how much I am grateful to my parents and brothers back home for their care, help and encouragement during my studies.

A special thank goes to all my friends for helping me during my stay in Canada.

Table of Contents

List of Figures

List of Tables

Chapter 1

Introduction

1.1 Background

Mobile Ad-hoc Network (MANET) consists of peer nodes with usually identical networking abilities, which are capable of acting as mobile routers and communicate directly without any infrastructure. MANETs are self-configuring and adaptive, to re-configuration when the network topology changed due to node mobility. Packets can be sent in multi-hops from the nodes at source to those at the destination with no need of fixed intermediary router nodes. Hence, they are not restricted in their deployment by any demand for such infrastructure and can be set up quickly in scenarios where setting up a communication backbone is not possible [1].

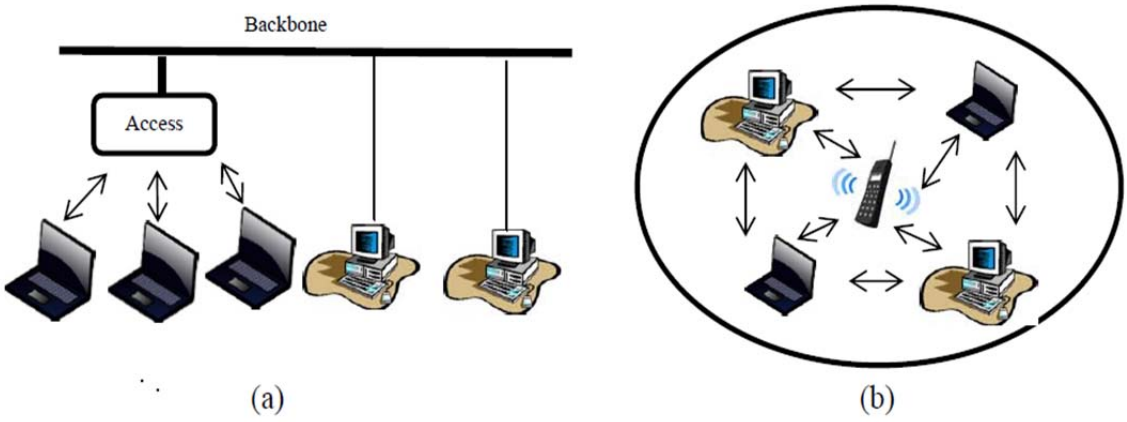


Figure 1-1 Network with (a) infrastructure and (b) without infrastructure

Some properties of mobile ad-hoc network are given below, as its advantages and restrictions.

Advantages:

- There is no need for an infrastructure
- Peer to Peer connectivity
- End-to-End connectivity is achieved by multi hop forwarding

Restrictions:

- Bandwidth is limited.
- Additional security problems in wireless network as compared to wired networks.
- Non-continuous energy source like batteries are used by many nodes in the network.

Despite those constraints, the applications of MANET are increasing, as the use of the number of portable devices is increasing. The major areas in which MANET has applications includes [2]

- Appliances level
- Personal Area network
- Battlefields
- Hazardous areas
- Disaster areas

At the appliance level, different portable devices such as laptops, tablets and cell phones can be connected with the help of ad-hoc network. All other appliances in a building can also exchange information between themselves. The interconnecting network between different mobile devices is termed as personal area network (PAN) and MANET, which often form the foundation of these types of networks.

In battlefields there is no communication infrastructure. MANET can help the devices to communicate with each other where no other source of communication can work. During floods and other disasters, MANET playing a key role in rescue operation, making possible the communication to distant and hard places which is normally impossible to access.

The performance of MANET depends on many parameters such as the following [3]:

1. The size of the network, i.e. the number of nodes
2. Connectivity of the network, which is the average number of neighbors for a node.
3. The rate of topology change in the network depending on the mobility of the nodes
4. Mobility pattern of the nodes
5. Traffic flow characteristics in the network

Situational Awareness

A Situational Awareness System (SAS) is an application in tactical MANET networks that gathers the information of the location and topology to produce an overview of the network [4]. In situations such as battlefield, fires and disaster relief, it is very important to keep informed the command unit of the networks status, surrounding conditions of the nodes and providing all information about the command center with location, connections, node energy levels and security. SAS is also a very useful application in various fields of life including personal communication networks, vehicular and ambient communications [6]. In particular, knowing the location and connectivity of the nodes would help the command center to make better decisions regarding the operations on the ground.

In SAS, the command center needs to know the status of the network in real time, which is referred as the "Common Operating Picture" (COP) [5]. This COP includes status, location, connectivity and other parameters of the network components to provide information to command

and control center about the status of the network. It is essential to keep the COP updated with changes and to provide a logical view of network configurations. In general, designing a SAS for tactical MANET is a challenging task. The common features of such networks include low bandwidth, limited energy and small size of the mobile nodes, low computational power and frequent changes in topology [7].

There are different methods for localization including the use of GPS information. However in certain scenarios, it may be impractical to use the GPS information for all nodes due to cost, or natural obstacles such as building, tunnel and forest which might weaken the signal of the GPS satellites. Thus, some non-GPS methods have been developed for localization.

Some of the localization methods used for tactical MANET includes [8]:

- Use of global positioning techniques
- Node connectivity
- Time of arrival (ToA)
- Direction of arrival (DoA)

The nodes which have the GPS facility, having information about location are known as anchors. One of the applications of situation awareness system is in cognitive network. Cognitive network is a network in which elements adapt themselves according to the network conditions through learning and reasoning. Here the central requirement policy is implemented, where the requirement of whole network is considered instead of individual nodes requirements. The information provided by SAS allows nodes in cognitive networks to incorporate the changing in the location. This makes it easy to configure the mobile nodes according to the movement of the nodes.

1.2 Related Work

The most common mobility models used for mobile ad hoc networks were discussed in [11] [12]. The authors examined different mobility models where the nodes were either dependent, as in group mobility models, or independent, as in random mobility models. They evaluated the performance of Ad Hoc networks under these mobility patterns in different network protocols. To validate wireless network protocols, a variety of methodologies were applied. These included the use of simulation and repeatable scenarios. Repeatable scenarios give the operator freedom to repeat testing of particular protocol with some variation of scenarios to gain better understanding of how the changes affect the performance. Simulations, on the other hand, allow the testing of one parameter at a time, keeping others constant, to study the effect of the changes in more detail.

Celestine and Gwang [10] designed an intelligent system, MANET using the cognitive behavior in tactical battlefield. The system studied the human behavior and how they react and make decision in dynamic topology. The human behavior was found to have properties such as making independent decisions in different situations which is the requirement in the MANET too. The intelligent system was able to discover the environment information to understand and learn the operational environment of the MANET. The system used OODA model [13], where the system could collect and observe data, adapt to environment changes, decide and act. This model designed in real time to respond and react to the new information.

The method proposed in this research is used to improve the localizations based on movement predictions in cognitive networks. Several algorithms have been developed in order to predict the movements of mobile users in fixed wireless networks [14] [15]. These algorithms are based on the previous history of mobile node and the future location is calculated on the basis of the user's movement history, i.e. the algorithms apply to different mobility models such as circle model,

track model and Markov chain model. The paths of the movements can be random or regular. A single person takes a particular path daily for going to work etc. These movement patterns help to improve the quality of service, for example, more optimal routing can be provided to the user for his fixed movement patterns. However, the drawback of this method is that it fails to predict varying behavior for different applications of MANETs and node's mobility.

In [16], an algorithm implemented to estimate the expiration time of the wireless link between two peered nodes. If the time of the expiration between two nodes is predicted, the route can be reconfigured before the link is broken between the two nodes. The timing information can be calculated based on the location information and the speed of the movement. GPS can be used to provide exact location information, if this is not the case, then approximate distances between nodes can be determined by measuring transmission power rate changes.

In [17], a Kalman filter was used to estimate the acceleration, position and velocity of the nodes. In Kalman filter an autoregressive model was used for the mobility of the mobile Nodes. For the cellular networks, the base station node was fixed while other nodes were mobile; as a result the autoregressive model gave a good estimation of the mobile tracking.

McDonald and Znabi [18] have shown that a link availability model that used probability could predict the future status of a wireless link. They introduced a method to estimate the availability of new links, if the link between two adjacent nodes expired. Link availability, prediction model was basically used the probability, if the link between nodes would remain active or not after certain period of time. The time of expiration of the link was calculated prior to the link expiration which helped to find an alternative link in case of link expiration.

The mobility in the network may cause network partitioning. If that happened, then the network became a combination of sub networks. The prediction of the mobility helped in preventing this type of network disruptions. The prediction methods in [19] use a low complexity data clustering algorithm for this purpose. This algorithm accurately determined the group and individual mobility patterns. The problem with the prediction method was that, it assumed the node velocities to be constant which was not a case in majority of MANETs.

Dersingh et al considered the full view and nodal view of network topology based on the OLSR connectivity and link information [6]. In full (centralized) view, the command and control center would collect information from all nodes to build the network topology. In nodal view, each node (candidate node) would individually build its own view of the topology based on OLSR information. Here In Chapter 2, it has been explained that an OLSR nodal has a partial view of the network topology. The authors calculated the topology discovery accuracy based on the total number of active links at each time step. Then they compared the nodal view to the full view to find out the total number of unrecovered links at each time step in each scenario. Their simulation included several static and mobile scenarios.

A visualization system named MANET- viewer was developed by Koyama et. Al. [20]. This system was developed for observing ad-hoc network behavior on the basis of a centralized approach, where nodes are dedicated for collection of data. These dedicated nodes were personal computers with wireless cards. The topology information of the network was gathered by these nodes. The performance of their visualization approach depends on status and characteristics of the network. Experiments showed that information collection took more time when the numbers of hopes were increased. The location information was not provided by the system, though it allowed manual addition of GPS coordinates.

Islam et al [21] proposed a network visualizer called Netviz based on applying a Force-directed algorithm [22] to the local connectivity databases in an OLSR MANET node. This approach would be discussed in detail in Chapter 2 and is the foundation for the work in this thesis. Netviz allows the users to set some anchor or landmark nodes with known GPS information used to determine the location of the other nodes based on connectivity generated from OLSR database. Simulation results in [21] were conducted based on the assumption that 25% of the nodes would have their real-time location information, while rest would use the Force-directed method to determine their approximate locations. However, the localization errors in this method were high and need to improve for any practical use.

1.1 Motivation and Objectives

The method in [21] had inadequate accuracy to determine the location of the nodes, and the impact of mobility was not fully explored in that works. The main purpose is to improve the performance by using the predictive and smoothing filtering techniques, and also explore the effect of mobility by simulating it under different mobility models.

In this research we study the performance of Force-Directed algorithm in order to improve the coarse location data and then apply different types of filters to improve the results. To check the results, simulation will be done in order to determine the better performing filter.

1.2 Methodology

The simulation environment consists of the NS-3 simulator for collecting OLSR and location data, then feeding this data into the NetViz visualizer and comparing the computed location information with the actual location data from NS-3. For the visualizing purpose, two files are generated:

- Topology File
- Locations File

For the topology file, topology map is built in the candidate node on the basis of the information received from the individual nodes. An application runs on the candidate node for the extraction of topology and location information from NS-3 OLSR databases. The following diagram shows the operation of the NetViz visualizer.

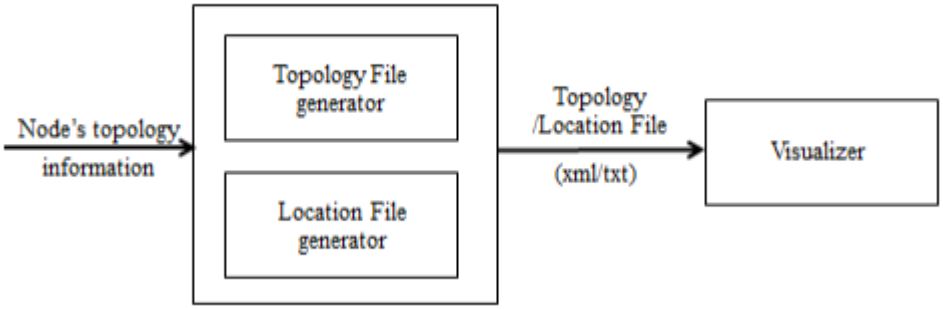


Figure 1-2 Visualization Process

The topology file contains the node adjacency information in xml or txt format, where the file is being updated during run time. The generator module has the access to the neighborhood database and the topology database of the candidate nodes.

The Topology file contains the number of active links within the networks at each time step. The total number of active links was calculated according to the links of each neighbour node as well as multipoint relay (MPR) nodes. More information about these nodes is provided in Chapter 2. Different mobility models were used to perceive the impact of node mobility on the performance of MANET.

Once the nodal coarse location information is determined, the following filters are applied to the approximated data of location and topology from the visualizer (NetViz):

- Moving average
- Kalman filter
- Low pass filter

The values are then compared with the NS-3 values, as well as the previous approximate values are also compared with original values; the process is shown in the below figure.

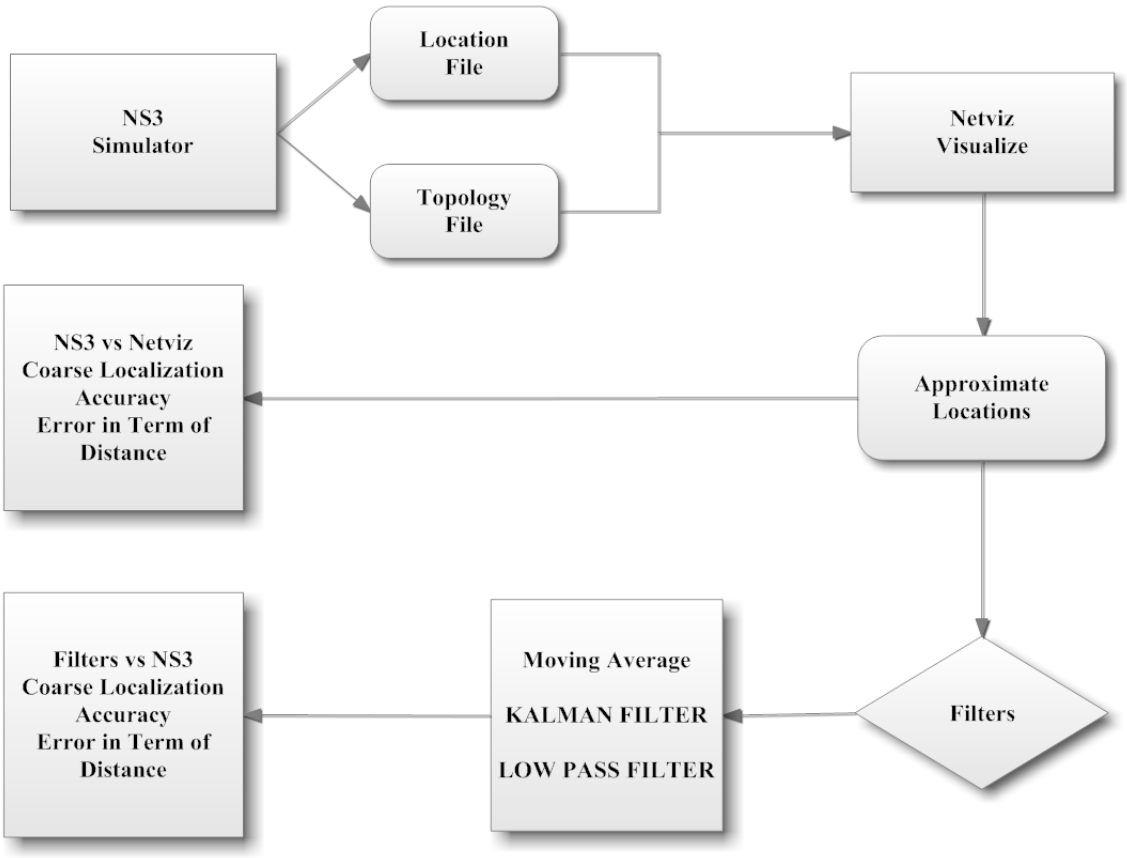


Figure 1-3 Coarse Locations Improvements Process

Below is the picture representation of the simulation demonstrating each step of the process: First of all, the script will be run on a command line to get the topology and locations file, the implementations of the mobility models were integrated with OLSR's script in [6] to show the movement of nodes with different mobility models.

The topology and location files were generated from NS-3 in every 2sec as the Figure 1-4 shows a portion of the files.

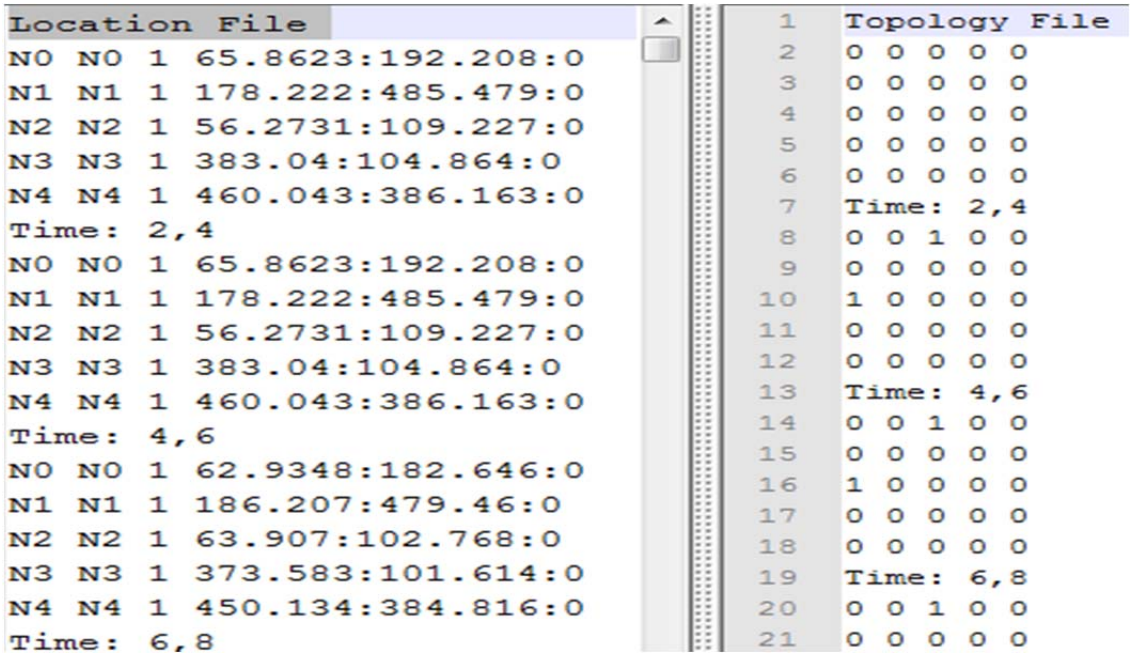


Figure 1-4 Topology File and Location File (NS-3)

Then those files generated from NS-3 are fed into NetViz.

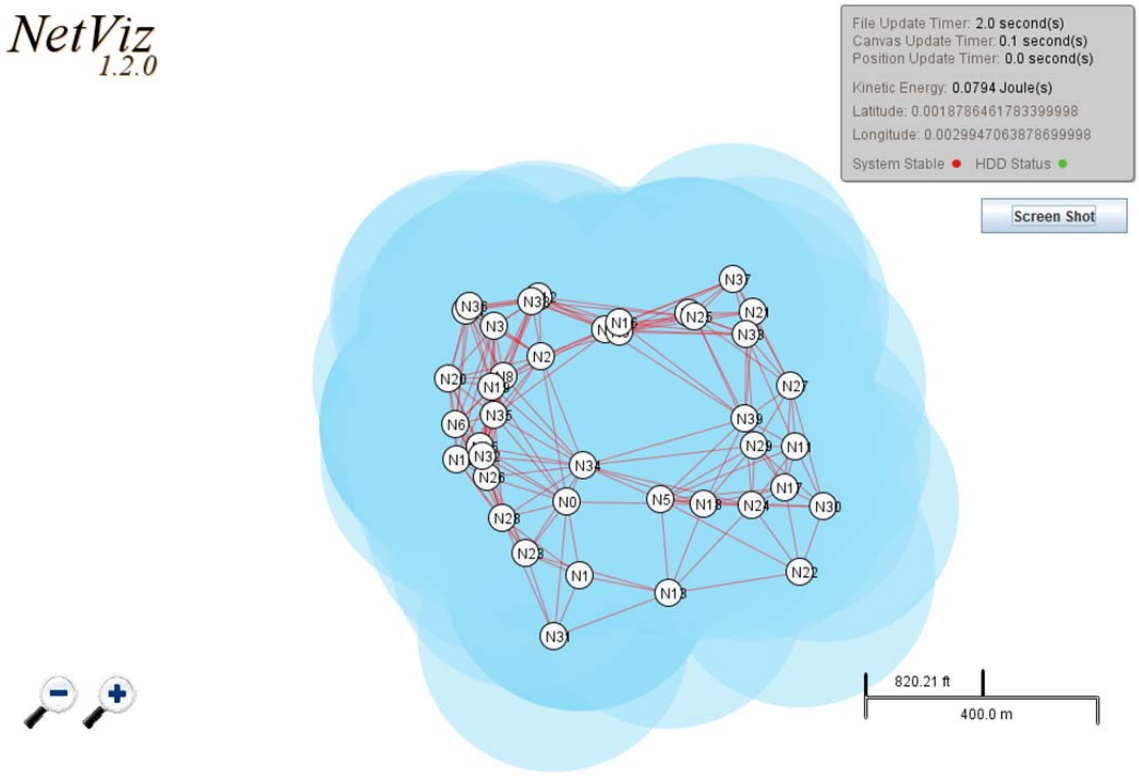


Figure 1-5 NetViz Visualizer

Finally the generated file from NetViz contains the approximate locations which then compared with the original locations. But the initial positions of the nodes remain same and throughout the process if there are anchor nodes than its position will also remain same.

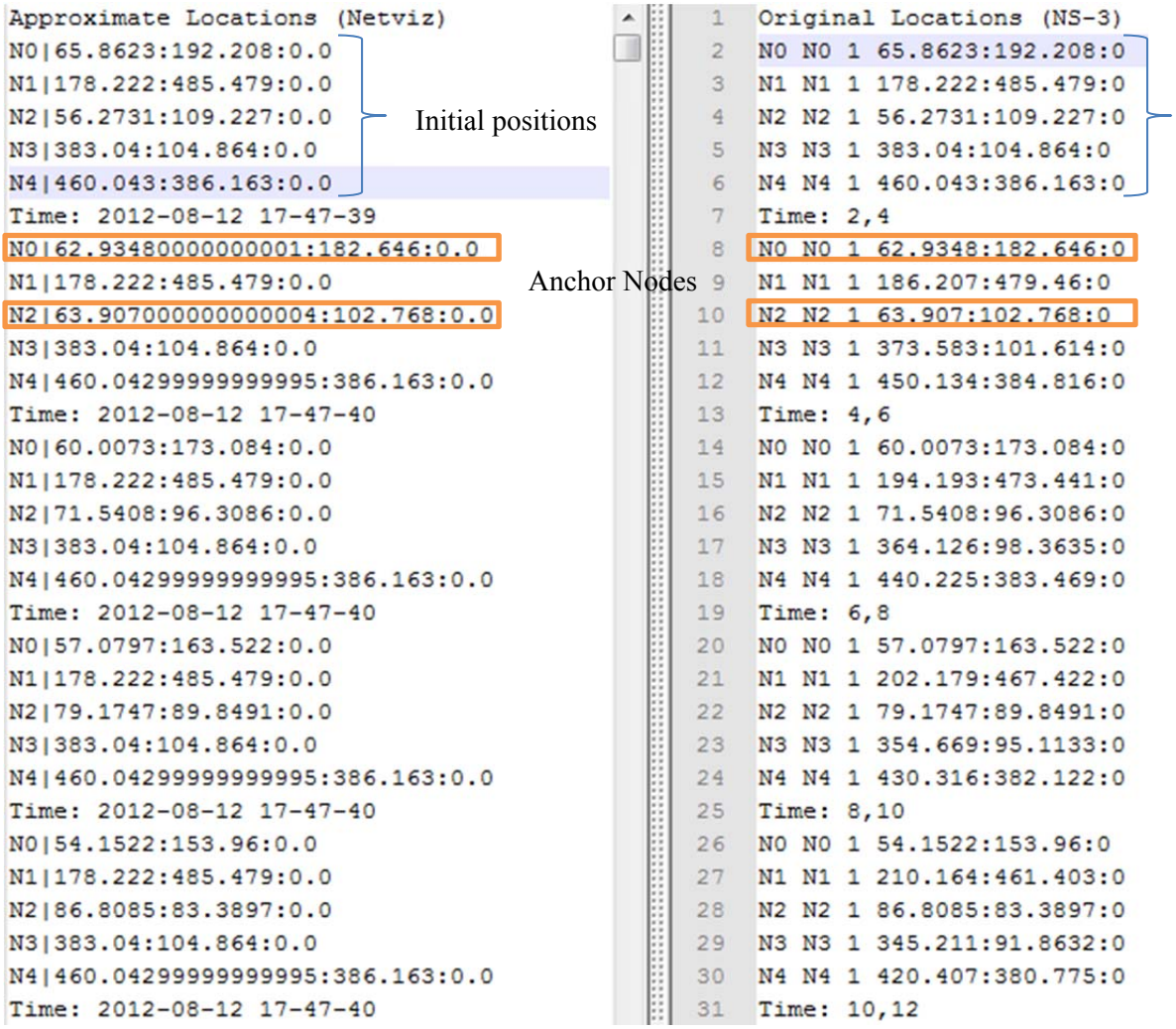


Figure 1-6 Approximate Locations and Original Locations Log_file

1.3 Thesis Outline

The rest of this thesis is organized as following:

The second chapter gives an overview of OLSR based on awareness system, coarse location, the impact of mobility models and the performance results for mobility models.

The third chapter describes the process that used to calculate the location and topology prediction, as well as it gives an overview about each prediction filter and how it was implentened.

The forth chapter will show the performance results of each filter and the coarse locations while Chapter 5 will cover the coculsion and future works.

Chapter 2

OLSR-based Situational Awareness Systems

The working group "MANET" of Internet Engineering Task Force (IETF) proposes two kinds of routing protocols for Mobile Ad hoc network – proactive and reactive routing protocol [1]. In reactive routing protocol such as AODV or DSR the routes are discovered on demand, i.e., there is no need to exchange control information in the absence of data traffic. On-Demand algorithms react to events such as the start of a data session or when a current session dies. In this approach the routes are updated only for the destination where the traffic intended to go. On the other hand, proactive routing protocol such as OLSR, periodically updates routes for all destinations even though there is no requirements of sending data traffic for the destinations. In general sense, proactive routing protocol introduce overhead in terms of bandwidth usage as control messages are sometimes exchanged unnecessarily, but the advantage of this routing protocol is that, it can quickly distribute network information over the network. The purpose of this chapter is to describe OLSR and some of the important ideas about connectivity visualization of OLSR-based MANET topology and coarse location.

2.1 OLSR

The OLSR (Optimized Link State Routing Protocol) is a table driven and proactive routing protocol [27]. It is designed for mobile wireless networks, e.g. mobile ad hoc network (MANET) and is specified in IETF RFC 3626. It is based on the link state routing but optimized the pure link state algorithm such as OSPF. The concept of multipoint relays (MPRs) is used in achieving the optimization. MPRs are some selected node within the network that is used to forward

broadcast message during the flooding process. The optimization, in fact, is achieved in the following ways:

I) Introduction of multipoint relays to reduce the size of control messages as nodes only broadcast a sub set of links with its neighbors, which are MPRs rather than broadcasting its all connected links to all nodes in the network, the process used in classical flooding mechanisms.

ii) Only multipoint relays are allowed to flood messages, which minimize network control message overhead substantially.

iii) In OLSR, only the partial link state information is distributed within the network. MPR nodes are responsible to report only the links between itself and its selector. Two important OLSR functionalities are neighbor sensing and topology discovery.

In OLSR, every node can detect its 1-hop and 2-hop neighbors based on the "Hello" message which is periodically broadcasted in the network by the nodes. This "Hello" message contains list of all the neighbors of a particular node. It also contains the status of the links to the neighbors. Thus, each node builds its own database of neighborhood, and two-hop neighborhood after analyzing the collected data of the "Hello" message. Any changes in the neighborhood are identified from these messages. All the 1-hop and 2-hop information are valid for a limited period of time, refreshed periodically. There can be four statuses of the connected links [28]:

- Symmetric: In this link, communication is possible in both ways. For example, if there are two nodes 'a' and 'b', then both nodes can send and transmit data to each other.
- Asymmetric: The communication takes place in one way. For example, from 'a' to 'b' or 'b' to 'a'.
- A Multi-point relays: In this case, the link must be symmetric and the node which has sent the hello message must select this node as a multipoint relay.

- Lost: If the link has been lost.

MPR nodes are selected by each node from the advertised list of 1-hop neighbors for that node. The selection algorithm ensures that, MPR nodes are selected for a node in such a way that the node can reach its entire two hop symmetric neighbors using the MPR node. However, MPR sets of a node may change, when 1-hop or 2-hop nodes of those nodes are changes. Each node maintains two sets of information – MPR set and MPR selector set. The MPR set defines a group of nodes which are chosen as MPRs for a particular node. On the other hand, MPR selector set contains all the neighbor nodes which have selected this node as a MPR. When a node from "MPR selector set" sends message for broadcasting purpose, MPR node broadcast this message [27].

Each node creates the topological information of the network based on the Topology Control (TC) message, which receives from the MPR nodes. The TC messages are flooded to all nodes in the network and take advantage of MPRs to reduce the number of retransmissions. It advertises the links between the MPR node and its MPR selectors.

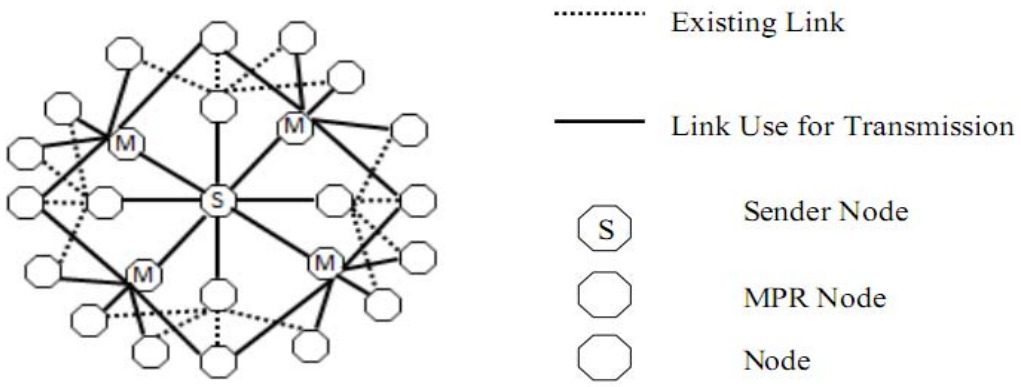


Figure 2-1 Flooding Mechanism MPR

In "Hello" message there is a field called Willingness, which is used in MPR selection process. This field decides whether a node wants to carry or forward traffic on behalf of other node. The value of this field can be an integer number between 0 and 7. The value 0 WILL_NEVER means a node does not wish to carry traffic for other nodes. This node will not be included in MPR set. If the value is 7 or WILL_ALWAYS, then the node will forward other nodes traffic. A node will include the entire neighbor node in its MPR set which has WILL_ALWAYS value set to 7. This is the beginning of the process and the node keeps adding its neighbors to the MPR set based on the reach ability to the 2-hop nodes which is based on the link set, the neighbor set, and 2-hop neighbor set until all the 2-hop neighbors are covered by at least the one of the MPRs.

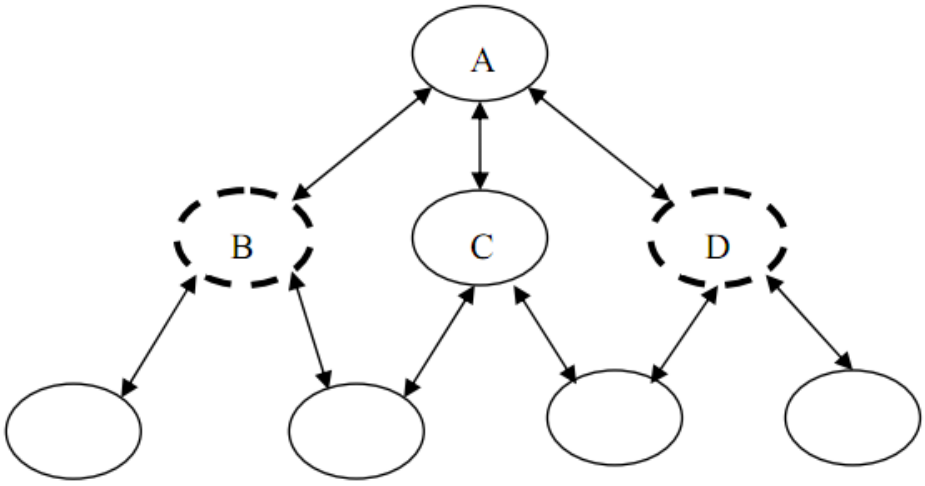


Figure 2-2 Node B and D Selected as MPR by A

However, willingness values can be changed based on node's condition. For example, node's Willingness value will be reduced if the battery life of a node is reduced. A typical OLSR node's Willingness value should be set to WILL_DEFAULT or 3.

Visualization of the network can be done in two ways [6].

- Centralized visualization and
- Nodal visualization.

The nodal visualization provides a picture or view of the network topology from individual nodes point of view. This view can be partial or complete depending on different time instances. Each of the nodes in the network builds a view of the network topology based on the messages it process, while for the centralized network view, It assumed that a central monitoring node can intercept all the message exchanged in the network or has access to the routes and topology databases of all nodes. Comparing to the nodal network discovery, converges in centralized network discovery is more speedy. However, active participation, for example, collecting status information by means of sending queries to the nodes may require in this case.

A combination of both approaches can be applied, if required in demanding network situational awareness, for example, tactical MANET.

The candidate node, which can be a general node or central monitoring node, performs the task of network visualization (nodal view or centralized view). The candidate node implements an application module. The purpose of this module is to collect the node's topology information, location information and then display the network topology graphically.

The topology file generator generates the topology file and the file is updated real-time when there is any change, which can be implemented in two ways. Firstly, generators have access to the neighborhood database and topology database of the nodes. The neighborhood database contains node's 1-hop and 2-hop neighbor nodes addresses. While On the other hand, the topology database contains the address of last hop node before the destination node and the address of the corresponding destination node in the network. Utilizing the information of these two databases,

entire network topology can be constructed. Secondly, topology file can be created by utilizing all the incoming and outgoing messages, which the topology generator's module can create from the information.

2.2 Situational Awareness Systems

One of the most important factors in Situational Awareness Systems is localization, which is a part of the real time common operating picture of the network. This COP includes status, location, connectivity and other parameters of network components to command and control center [6].

GPS is a method of localization, but it can't be present on each node, since it's not cost effective and there are hurdles in the communication with GPS satellite like building, tunnel and forest. Due to this fact, some non- GPS localization methods has been developed [8].

Received Signal Strength Indicator (RSSI): The methodology utilized by RSSI is based on the amount of power transmitted. As the name RSSI itself suggest to be a signal strength indicator, it measures the power of radio signal at the receiver end and the effective propagation loss based on the amount of power transmitted. In order to determine the distance, theoretical and empirical approaches are used as the previously calculated loss is known. In location discovery technique cost is considered to be a huge influential factor. Since the sensors available are already equipped with radio, RSSI reduce the cost factor significantly. Radio signals use multipath propagation for communication. Due to this factor RSSI performance is not as good as compared to the distance measuring method. This methodology is specifically used for RF Signals.

Time Based Methods (ToA, TDoA): The technique that this method uses is based on the time of arrival (ToA) or time-difference-of-arrival (TDoA). It calculates the time taken for the

propagation of signal during the communication between two nodes. Once the propagation time is found it can directly be converted to distance. This method can be used for any kind of signals such as infrared, ultrasound and RF Signals. Since any kind of signals can be used for communication, nodes which use acoustic signals may face difficulties to detect the accuracy of signals in hazard condition. The propagation of signals may be dampened due to its multi path propagation effects.

Angle-of-Arrival (AoA): AoA calculates the distance based on the angle of received signal and the position of the node is determined by geometric relation. The accuracy for this technique is higher than the RSSI, but it is not cost effective. In order to determine the precise condition of the node and determine the angle of received signal it requires additional hardware as compared to RSSI.

2.3 Coarse Locations

Coarse localization is a method for approximating the location of nodes in mobile ad-hoc networks. For the localization, information of the location of the network is gathered by the nodes based on layer-3 connectivity. In OLSR-based coarse localization, an area is considered in which there are N nodes and each node in the network has a communication device that establishes an ad-hoc network. An assumption is made that the total area of consideration is much greater than the radio range of single node. Due to this consideration, multi-hop communication will be adapted using OLSR [21]. The whole network is presented as a graph

$$G = (V, E)$$

Where

"V"represents the nodes in the network in terms of vertices

"E" represents unordered pair

If the nodes are considered at time" **t**" and the locations are represented as {L1,L2 …}. L as x and y coordinates. "R" represents the ranges of transmission by each node. The adjacency matrix is shown by A $\{a_{ij}\}$

$$a = \begin{cases} 0 & i = j \\ 1 & i \neq j \ and \ d_{ij} \leq R_i \\ 0 & i \neq j \ and \ d_{ij} \geq R_i \end{cases} \qquad (2-1)$$

Where

$$d_{ij} = \sqrt{((L_x^i - L_x^j)^2 + (L_y^i - L_y^j)^2} \qquad (2-2)$$

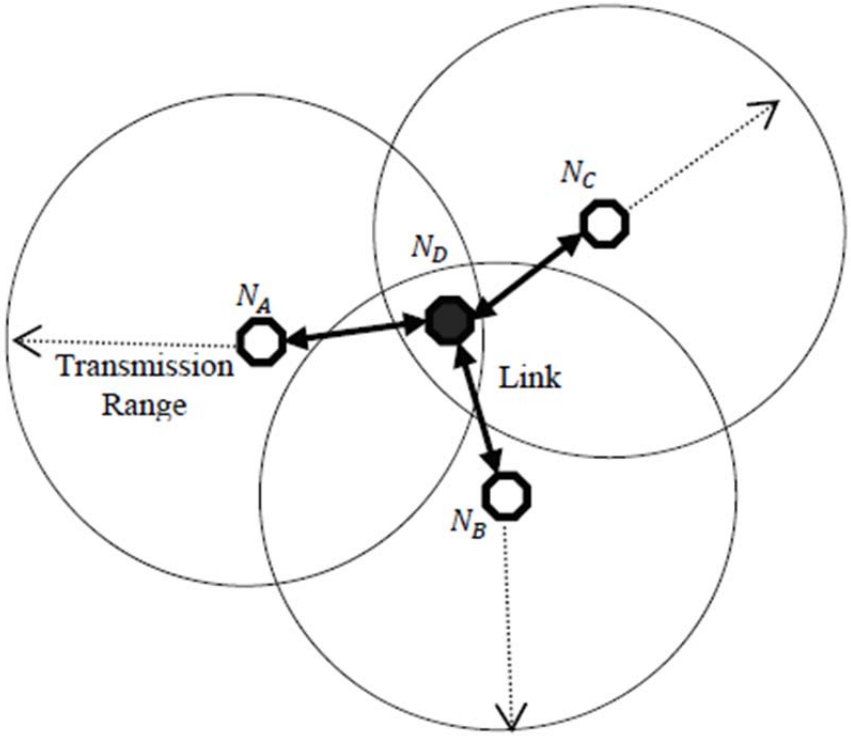


Figure 2-3 Coarse Localization Using Overlap of Adjacency Transmission Range

In Figure 2-3, the nodes are represented as small octagons. The circles surrounding the nodes represent the transmission range of each node. Once the adjacency network is established the residing zone of the non-GPS enabled node can be approximate. This method of location approximation is referred to as Coarse Localization.

S_i represents the area of transmission of node and S_i' represents the area outside the transmission.

$$L_i \, \varepsilon \left(\bigcap_{a_i j=1} S_j \right) \bigcap \left(\bigcap_{\substack{a_{ij}=0 \\ i \neq j}} S_i' \right)$$

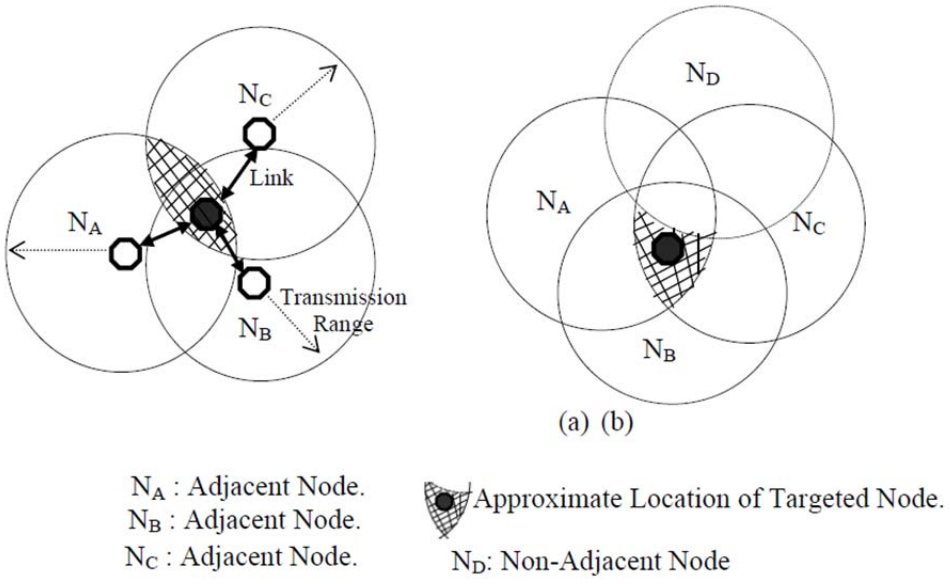


Figure 2-4 Coarse localization with Topology information

The coarse localization is basically a graph layout problem which can be solved using Force Direct algorithm. In force directed method, the graph is the physical model and nodes act as vertices in the network. In FD model, there are two components.

- Transferring the connectivity into a system of physical forces.
- Algorithm for minimizing the total energy of the system.

Kamada-Kawai FD model is a popular method to represent the system in form of physical forces [26].In this method, a spring model is used between adjacent two nodes based on hooks law. The energy in KK FD is expressed as below

$$E = \sum_{u,v \in V, u<v} \frac{1}{2} k_{uv} \left(|L_u - L_v| - l_{uv}\right)^2 \qquad (2-3)$$

Where

L_u and L_v are the locations of the nodes

u and v are calculated at each iteration

k_{uv} is the spring constant

A variant of this method in which the attractive force is modeled by Hook's law and repulsive force is modeled by coulomb's law is used because it provides a better balance of attractive and repulsive forces.

$$f_{xy}^{r} = \frac{k_e(Q1Q2)}{d^2} \qquad (2-4)$$

$$f_{x,y}^{a} = \frac{1}{2} k_s d \qquad (2-5)$$

This approach was implemented in [21] to provide coarse visualization of tactical MANETs. While localization based on the layer-3 connectivity information is bandwidth efficient (because only one-bit link status information per link needs to be exchanged), however the results indicated poor accuracy. In Chapter 3 we will present a method to improve the accuracy of this approach.

2.4 The Impact of Mobility

The performance of the ad-hoc OLSR network under different mobility patterns is discussed in this section.

A mobility model is a mathematical representation of the movement of mobile nodes (MN) in a network. It can be used to represent the movement of the MNs in the simulations. The change of the speed and the direction is illustrated with respect to the time.

There are two types of mobility models i.e. **independent node mobility models** and **group node mobility models**. In independent node mobility, the mobility of single node is considered while in group mobility, more than one node is considered. Various Independent node mobility models include

- Random walk mobility model (Random Walk MM)
- Random waypoint MM
- Random direction MM
- Gauss-Markov MM

The group mobility models include:

- Exponential correlated random MM
- Column MM
- Nomadic community MM
- Pursue MM
- Reference point group MM (RPGMM)

The random walk mobility model was first described by Einstein in [21]. A keen observation of the nature indicates that the movement of many particles is random in the nature. In this movement pattern, the new location is selected on the basis of the random speed and random direction from the previous location. The speed randomly selected from a given range. The direction of the movement is in 0 to 2л radians. In random walk, each step can be in terms of constant time movement or constant distance travelled.

The movement within a simulation boundary, it means that there is a bouncing effect from the boundary of the simulation area. After bouncing, the angle of return is determined by angle of strike. For the purpose of simulation, different dimensional walks (movement) have been developed which includes 1D, 2D, 3D and d-D. This type of movement is known as Brownian motion [11].

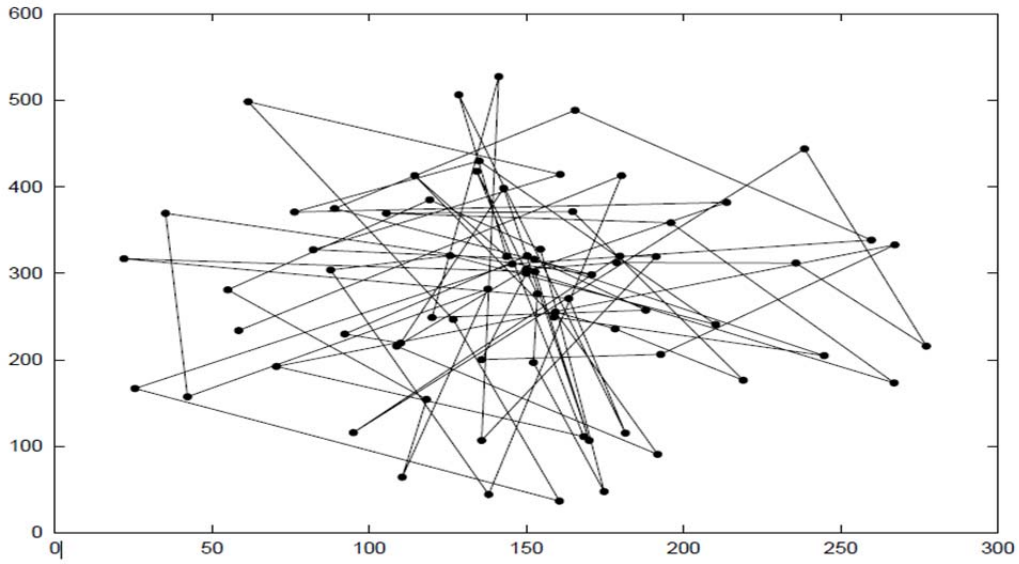


Figure 2-5 Random Walk 2D

Random Waypoint MM has a time of pause at every location of the way and then the direction and the speed is changed from it [12].Difference between the Random walk and the random waypoint is that, there is a pause time at each location of the movement. If the pause time is zero then the random waypoint will be as random walk [11].

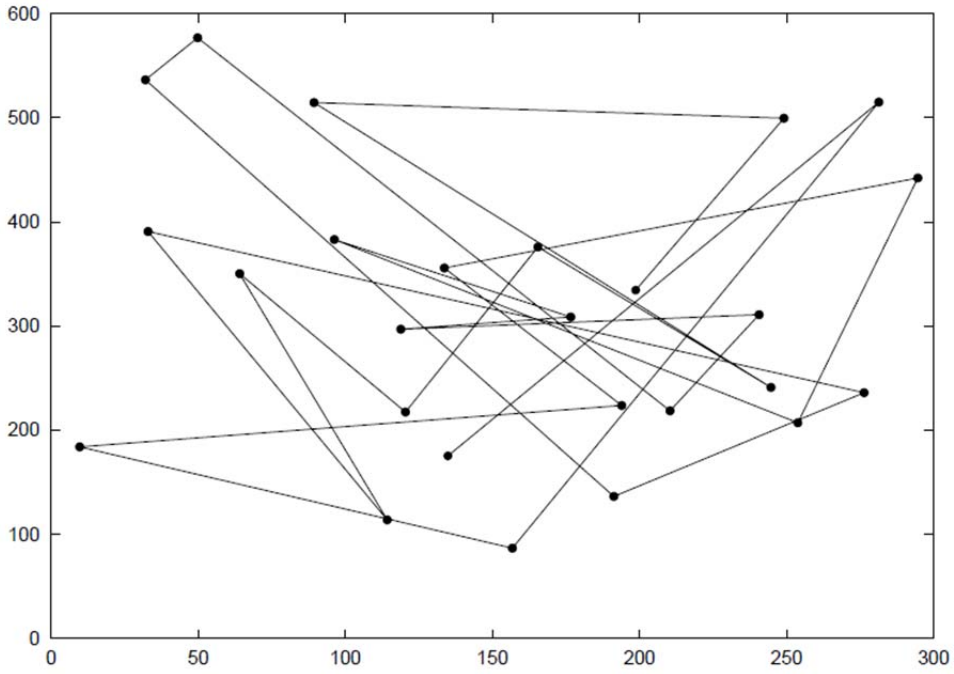


Figure 2-6 Random Way Point

The problem of clustering of nodes in a particular simulation area is known as density wave. This problem exists in random waypoint MM [11]. Density wave is generated due to bouncing of different nodes and gathering in a particular area. Due to this density, it's hard to analysis the mobility. In order to overcome the problem of density wave, the Mobility model of random direction was developed. The mobility node pauses for the specified time after reaching the simulation boundary. The modified random direction MM [12] is the newer version of MM. in this modification, the mobility nodes chooses random direction but it is not necessary to travel

towards the boundary instead the direction can be changed before reaching the boundary unlike Random Way point nodes will never reach to the simulation's boundary rather than they move to the center of the simulation.

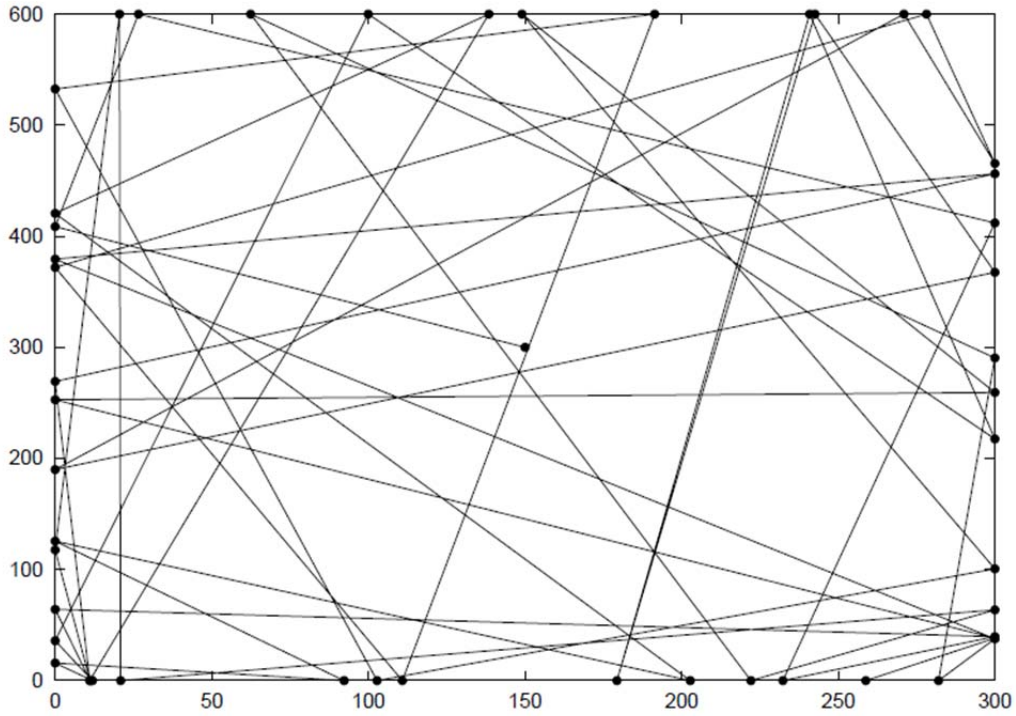


Figure 2-7 Random Direction 2D

The three models discussed so far were all random in nature, and there was no relation between the previous and current direction and speed.

The origin of the Gauss- Markov model [12] is for personal communication service networks (PCS), but it is used for the simulation of ad-hoc network as well. The speed and direction of the current time instant is assigned to the model initially and the next speed and direction is computed based on the initial speed, direction and a tuning parameter. Mean value of the speed and direction is included in the calculation. The value of tuning factor lies in between 0 to 1. If the

value of tuning parameter is set to zero, the selection of next speed and direction is completely random while if the tuning parameter is set to one, then a linear relationship is obtained.

$$v_t = \alpha v_{t-1} + (1-\alpha)v + \sqrt{(1-\alpha^2)}v_{xt-1} \qquad (2-6)$$

$$d_t = \alpha d_{t-1} + (1-\alpha)d + \sqrt{(1-\alpha^2)}d_{xt-1} \qquad (2-7)$$

v_t And d_t are the new speed and direction of the mobility nodes, t is the time interval and α is tuning parameters $0 \leq \alpha \leq 1$.

It is recommended to use appropriate value of tuning parameter for the mobility model which is gathered after visualization. The sharp turns encountered in random walk is controlled in Gauss-Markov with the help of past velocities affecting the present velocities.

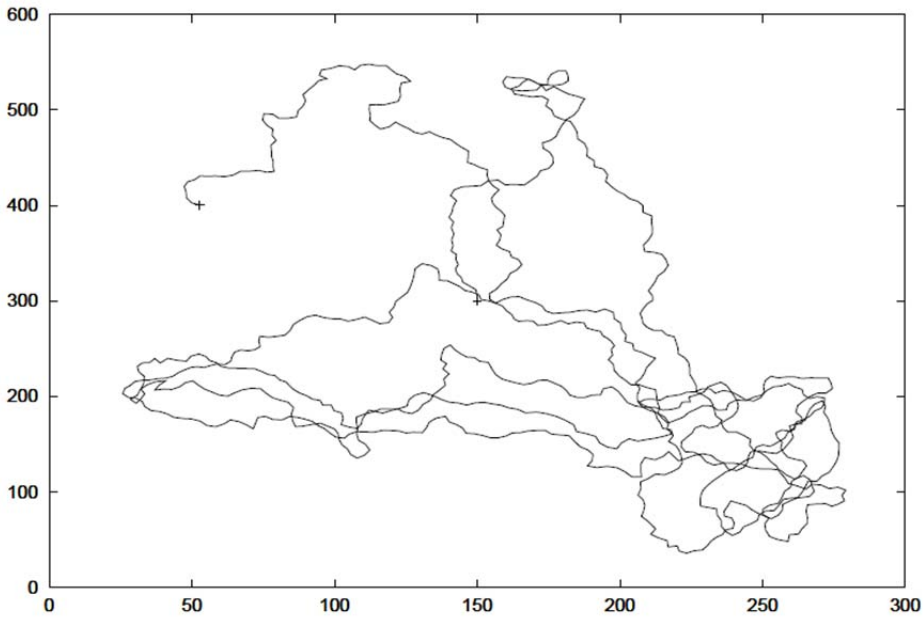


Figure 2-8 Gauss- Markov

In an ad-hoc network, there are situations where two or more different nodes movements are dependent on one another. Some examples include two people walking together on the street, travelling in the same car or a group of military vehicles moving toward the same assigned target. Group mobility models represent such cases.

In exponential correlated random mobility model [12], the motion function is used for the creation of mobility pattern. The next position of the group is identified with the help of previous position as it is done in other mobility model. Rate of change from previous position to new position is adjusted with a time factor τ. The equation of the new possible b that is motion function is given below [11]

$$b_{t+1} = b_t e^{-\frac{1}{\tau}} + (\sigma \sqrt{\left(1 - \left(e^{-\frac{1}{\tau}}\right)^2\right)}\, r \qquad (2-8)$$

Where b_{t+1} is the new location and b_t is the old position

r is the random Gaussian variable and σ is the variance.

The drawback of this mobility model is that it is not easy to create the motion, because the selection of $\sigma\ and\ \tau\ for\ a$ given motion pattern is difficult. Other mobility models overcome this problem.

Another group mobility model that is useful for searching purposes is the column Mobility model [11]. In this approach the movements are in a straight line and the update is used to allow separate MNs to follow each other.

$$new\ reference\ point = old\ reference\ point + advance\ vector \qquad (2-9)$$

The collective movement of mobility nodes from one point to another point is modeled by nomadic community mobility model. In nomadic community MM [11], there is an entity MM around which MNs moves.

Pursue mobility model searches for a target as the name shows [12]. The new reference point is the sum of the old reference point, advance vector, product of acceleration and difference of target and old position.

$$new\ reference\ point = old\ reference\ point + acceleration(target - old\ position) + advance\ vector \quad (2.10)$$

This model starts from an initial reference grid. The placement of reference point in the reference grid is done with the help of a relation that is they are placed a column pattern with different angles. The new reference point in this mobility model is the sums of old reference point and the advance vector, where the advance vector is a predefined offset that moves the reference grid.RPGMM is a generic method for handling group mobility (GM) [11]. It represents random motion of group of mobility nodes as well as individual nodes present within the group. There is a group motion vector, which determines the movement of the group. Group vector can be randomly chosen or predefined. The new position of the mobility nodes is determined with the following formula

$$new\ position = random\ vector + new\ reference\ point \qquad (2-11)$$

The example of RPGMM is avalanche rescue in which there are fixed paths to be followed and the follower can chose a random path in the chosen area.

2.4.1 Applications of Mobility Models

In random walk, there is a no dependency of current speed and direction on past speed and direction and each particle is free to select its direction and speed from the point of start. Most particles in nature studied in physics moves randomly which is modeled in random walk [12].

In random way point, there is a pause time at each stay of particle or node before it again starts moving in random direction with random speed. The main application of this model is a pedestrian which walks in random direction with random speed after reaching a destination and there is a pause time [12].

For the model of random direction, the direction of each particle changes randomly but this model is just for simulation purpose and practically, this is not possible since the distribution of people in the city can't be random [11].

Gauss Markov is a model in which there is a level of randomness in the speed and the direction which remains constant for a certain period of time. This model can be applied directly on the movement of the automobile. An automobile moves with a constant speed in a particular direction and the speed varies randomly dependent on the driver [11].

2.5 Topology Discovery Performance results for mobility models

2.5.1 Simulation Environment

For our simulations we used NS-3 simulator which is a discrete-event open-source network. It has intensive documentation through the internet for researcher's use. It is written in Python and C++ languages [24]. The simulations were run on the Linux operating system Ubuntu 10.10. All the scripts for the simulations were written in C++.

A number of different scenarios were run for different network sizes and mobility models. In addition to the existing mobility models in NS3, One Direction mobility model was also implemented in which the nodes were moving in one straight line. This mobility model is a good representative of vehicular ad hoc networks (VANET).

Using NS-3, the full view and nodal view were generated through accessing OLSR database to calculate the total number of active links. In order to calculate the accuracy of the OLSR-based topology discovery, we compare nodal view to full view to find the active links and undiscovered links.

If:

L = total undiscovered links

F = active links in full view

N= active links in nodal view

$$\text{Error Rate} = \frac{L}{F} = \frac{F-N}{F} \qquad (2-12)$$

Different scenarios were used with different number of nodes 5,10,20,30. Each scenario runs with several mobility models. The speed of Mobility nodes MNs is chosen in the range of 0 to 10 m/sec, while the pause time is in the range of 2 to 20 seconds except Random Walk and One Direction, where there is no pause time.

SIMULATIONS PARAMETERS	
Size of The Area	500*500m
Number of Nodes	5,10,20,30
Transmission Range	40 m
Wireless Model	IEEE 802.11 a
Delay model	Constant Speed Propagation Delay and Friss Propagation loss Model
Mobility model	Random Way Point Random Walk 2D Random Direction 2D Gauss Markov One Direction
Node Speed	3 m/s 10 m/s One Direction
Simulation Time	250sec
Step Time	2sec

Table 2-1 Simulation Parameters

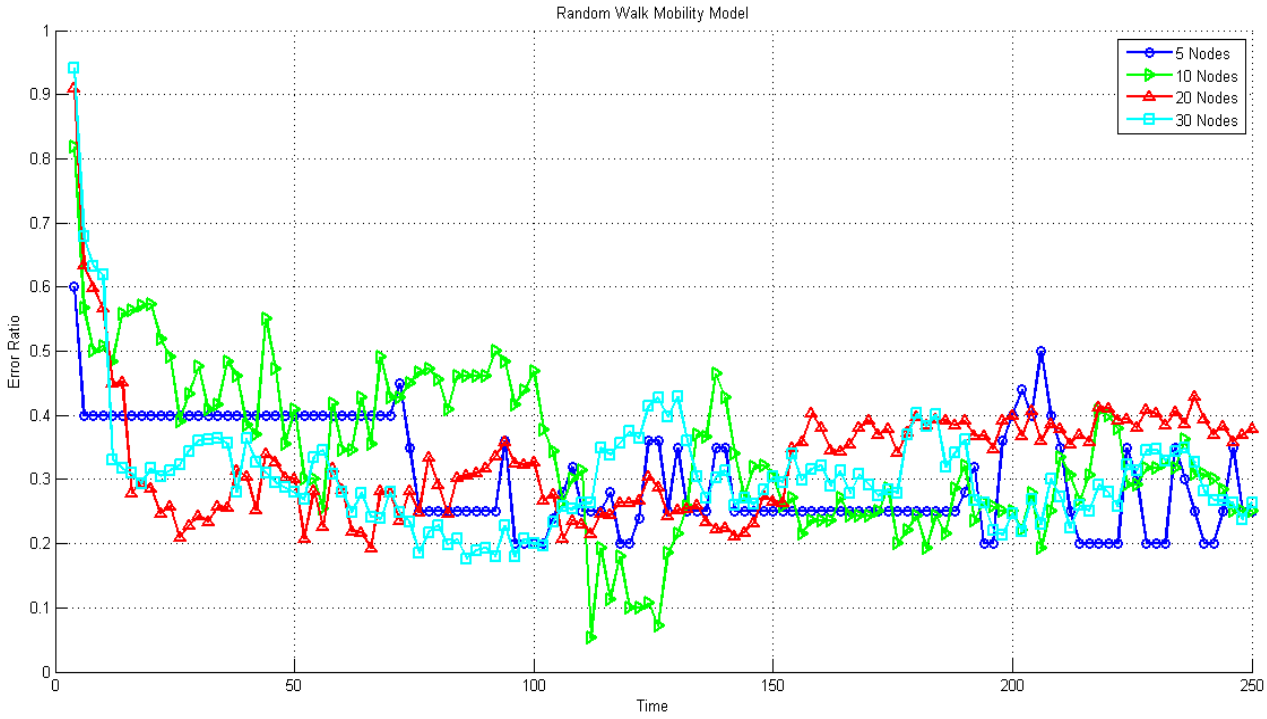


Figure 2-9 Average Error Rate for Random Walk 2D

In the above graph of random walk mobility model, the topology discovery error ratio is plotted versus the time. There are 4 lines in the graph. Each line is representing unique number of nodes in the network. The error ratio is constant for certain period of times in 5 nodes. The fluctuation increases in 10 nodes and with the increase of nodes to 20 and 30 nodes, the variations and the initial value of the error ratio is high due to the nodes start establishing links with others nodes.

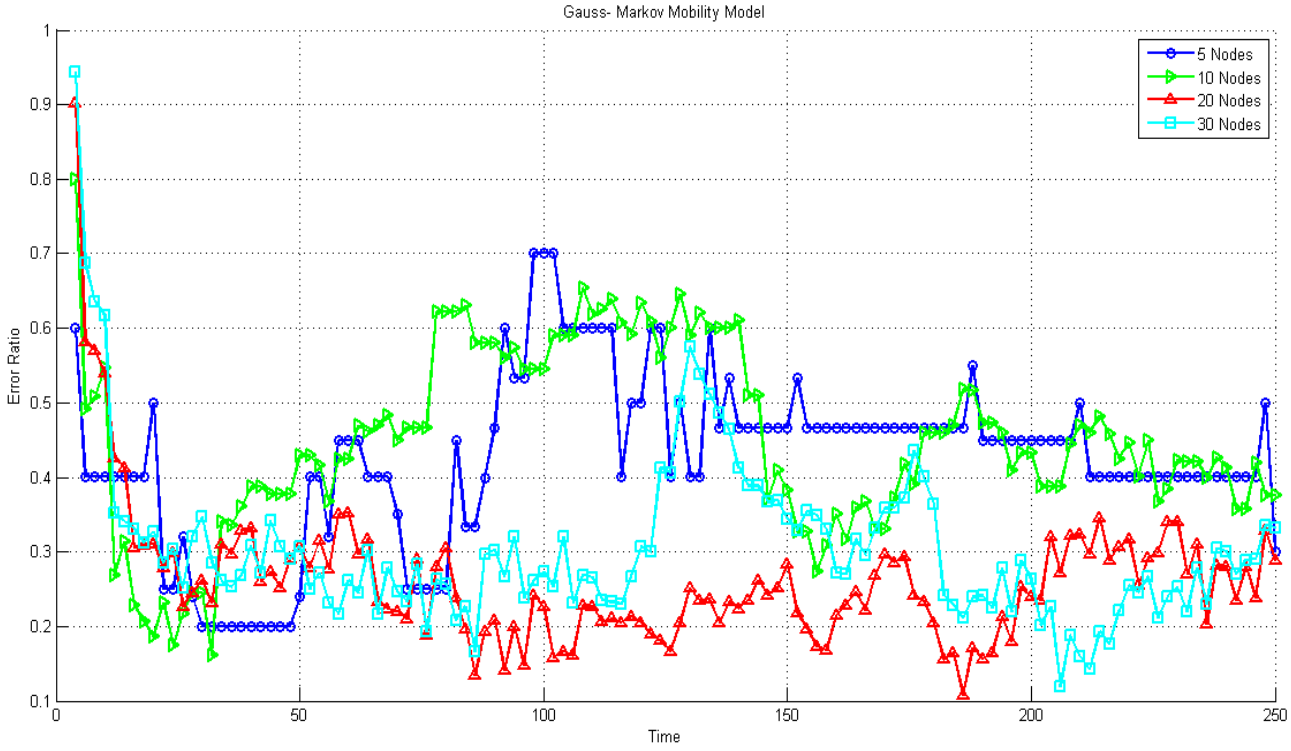


Figure 2-10 Average Error Rate for Gauss Markov

The above graph is related to gauss Markov mobility model. The error ratio for 5 nodes is low initially as compared to greater number of nodes, but more stable. With more nodes, the fluctuations are higher, because with the increase in the nodes, the random behavior of each node makes the error ratio more fluctuating.

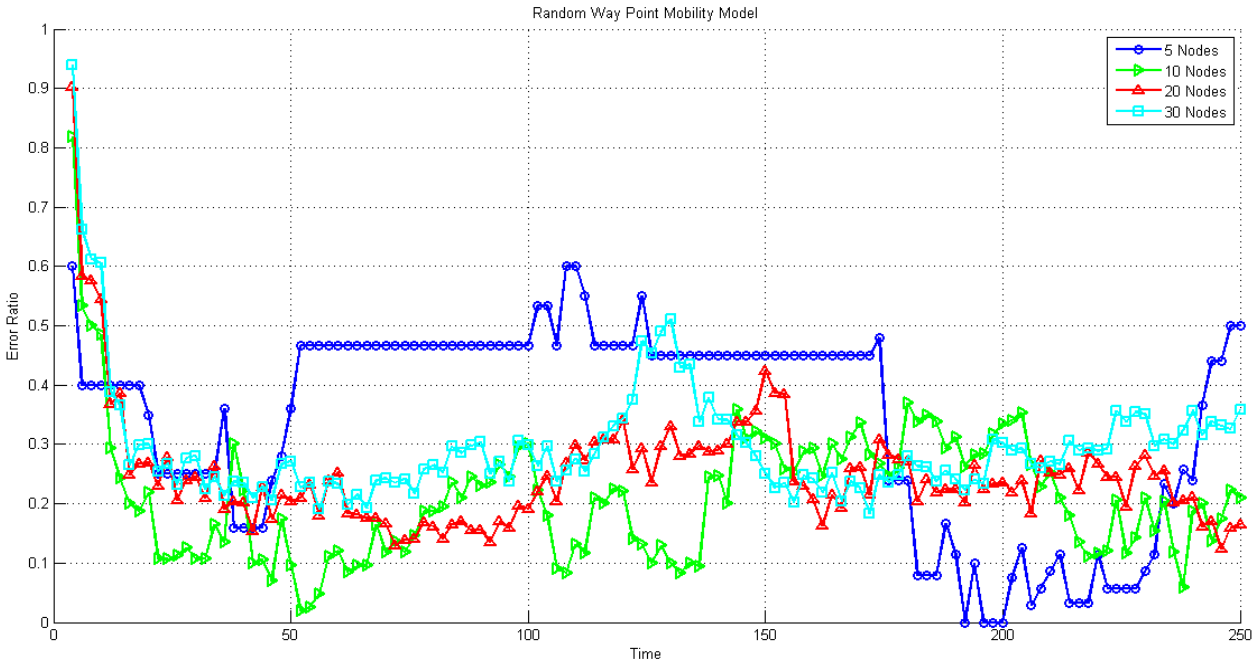


Figure 2-11 Average Error Rate for Random Way Point

In random way point, it's quite similar to random walk mobility model. With 5 nodes, the error ratio is high at start and then decreases for some time. It again increases and almost remains constant for a considerable period of time. After constant error ratio, it decreases and finally it again increases. The importance of the each link is higher in smaller networks so when a single link is missing, there is an abrupt change as visible in 5 nodes. The lowest error ratio is visible in the graph of 10 nodes and 20 and 30 nodes seem to follow the same path. The importance of the each link is higher in smaller networks so when a single link is missing, there is an abrupt change as visible in 5 nodes.

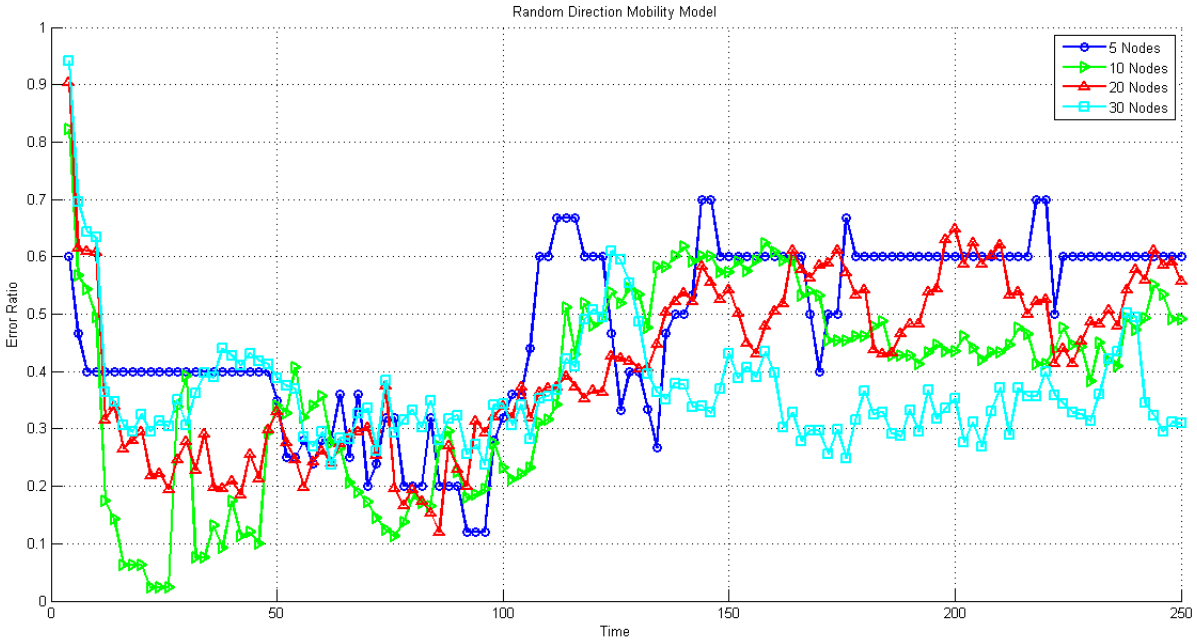


Figure 2-12 Average Error Rate for Random Direction 2D

In random direction mobility model, 5 nodes has minimum error ratio and 30 nodes has the highest error ratio. After the initial period, error ratios for the number of nodes decreases and remains sustained till the time scale reaches 100s. After 100s, the error ratio increases again because the nodes move till the reach the simulation's boundary so it might lose some connections. 5 nodes network has the highest error ratio and 30 nodes network has the lowest error ratio in the end of simulation.

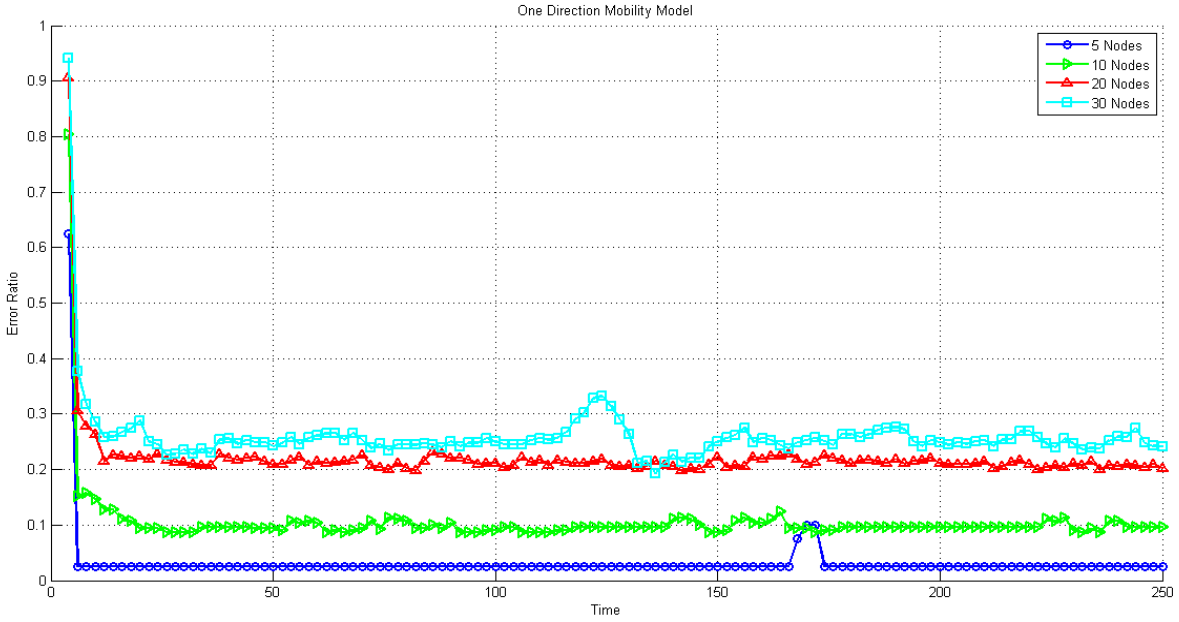


Figure 2-13 Average Error Rate for One Direction

In one direction mobility model, the plots of different number of nodes are considerably stable throughout the time span. The graph of 5 nodes has the least error ratio while the graph of 30 nodes seems to have highest error ratio as it is shown in Figure 2-9.

2.5.2 Analysis

At the beginning of the simulations the topology discovery error increased due to the OLSR delay in topology discovery at the start of establishing links among nodes. The nodes distributed arbitrary within the area so a node at the start of the simulation might have an active link or more to the nodes within its range. During the simulation time, each node starts moving and discovers the information about the networks topology. In all the MMs, the initial error ratio is high and the network with highest number of nodes (30 nodes) has the highest error ratio while least number of nodes network (5 nodes) has the minimum error ratio due to node density. For the all MMs, the error ratio will never become stable due to the permanent topology changes. In random walk, Gauss Markov, random way point, the pattern seems to be similar with some variations due to the difference in the method of movement while in Random Direction the error rate coverage after 100s.The pattern in one direction mobility model seems to be different from all other MMs since the nodes are allowed to move in only one direction which makes the error ratio considerably constant over the period of time.

Chapter 3

Topology and Locations Prediction

3.1 Mathematical Model for Location and Topology prediction

Analysis of simulation data in [21] indicates that the coarse location information contains random fluctuations due to the behaviour of the Force Directed (FD) algorithm. The nodes in the algorithm make unexpected bouncing, because they get pushed from other nodes, and this issue increases the localization error. Such bouncing does not happen in real MANETs, in which some degree of continuity of velocity direction and smooth transitions are expected. Here the objective is to use some estimation techniques to determine the correct location of the nodes based on the history of their movement and therefore, eliminate the incorrect bouncing and fluctuations resulting from the FD algorithm.

The first step is to calculate the direction and velocity at each time step to observe the behaviour of the node. The direction θ and velocity ϑ at each time step t >1 are computed as the following [25]:

Let's assume the coordinates $x_0, x_1, x_2, \ldots\ldots$ are the approximate locations of a given node. Then we can calculate the direction and speed of the movement as following:

$$\theta i = \begin{cases} \text{atan}\left(\dfrac{y_i - y_{i-1}}{x_i - x_{i-1}}\right) & if\ x_i > x_i - 1 \\ \text{atan}\left(\dfrac{y_i - y_{i-1}}{x_i - x_{i-1}}\right) & if\ x_i < x_i - 1 \\ \dfrac{1}{2}\ л & if\ x_i = x_i - 1\ and\ y_i > y_i - 1 \\ -\dfrac{1}{2}\ л & if\ x_i = x_i - 1\ and\ y_i \leq y_i - 1 \end{cases} \qquad (3-1)$$

$$\vartheta i = \frac{\sqrt{\left(x_i - x_{(i-1)}\right)^2 + \left(y_i - y_{(i-1)}\right)^2}}{T} \qquad (3-2)$$

Where T is the time step.

If there are any sudden changes, we filter the data by applying different filters such as Moving Average, Kalman Filter and Low Pass Filter. These filters will be discussed in Section 4.2. In this case we apply any changes on speed ϑ or directions θ, we have to calculate the new $\bar{\vartheta}\ and\ \bar{\theta}$ to get the new locations as below:

$$\begin{aligned} x_i &= x_{i-1} + \bar{\vartheta} * t_s * \cos\bar{\theta}_i\,, \quad \text{for } i = 2,3,\dots,n \\ y_i &= y_{i-1} + \bar{\vartheta} * t_s * \sin\bar{\theta}_i\,, \quad \text{for } i = 2,3,\dots,n \end{aligned} \qquad (3-3)$$

Once we gather the new locations from filters, we compare them to the original data taken from NS-3 to measure the improvement in localization accuracy.

Most of the filters that we have used are also used in different applications such as mathematical analysis (moving average) or signal processing (low pass filter) [29] [31].

Our process is shown in Figures 3-1

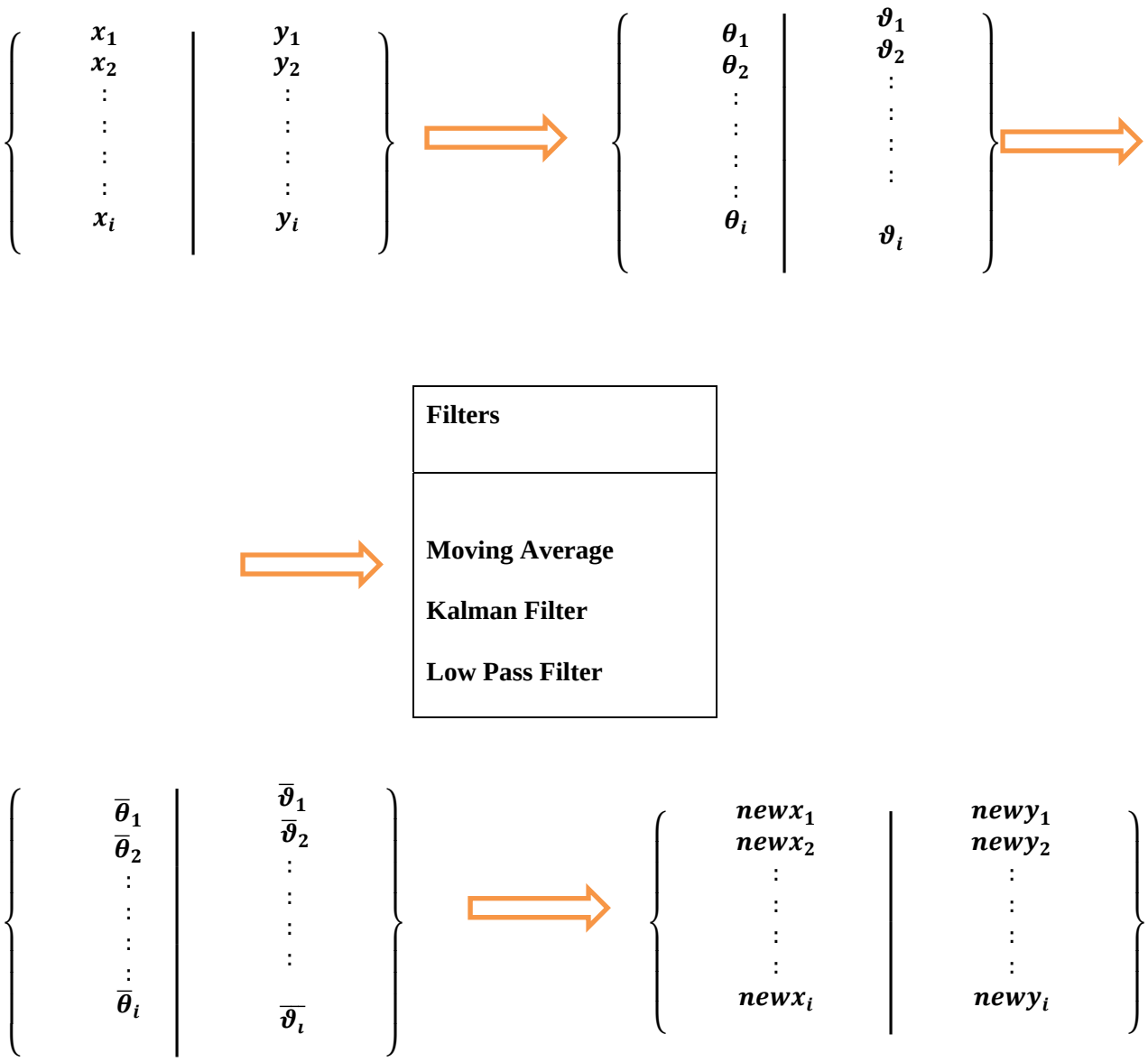


Figure 3-1 Algorithm Procedure

3.2 Prediction Filters

The most acknowledged problem in analyzing time series is to predict the future values of the discrete time series or signals. Predicting the future value of the signal at discrete time n is the main issue for the series in which some of the signal values are known:

$$x(1)\, x(2) \ldots \ldots . x(n - M)$$

Where "M" is any integer from 1 to n-1.

In the following, we review the three prediction filters used in this thesis and then explain how they were implemented in our approach.

3.2.1 Moving Average

Moving average is a method of predicting a new time series from an observed time series using its sequential values [29]:

- Two sided moving averages
- One sided moving averages

The difference between two-sided moving averages and one-sided moving average is that two sided moving averages are used for smoothing purposes, while one sided moving averages are solely used for the forecasting purpose. The methodology of moving average is very simple but it is a building block for more complicated tools. The main idea of moving average are those values which closely relate in terms of time are likely to be close, so taking the average value will remove some randomness from the data.

Normally many time series can be mathematically described as

$$x(t) = y(t) + e \qquad (3 - 4)$$

Where y (t) is a smooth time series and e is the zero mean error series. Smoothing is the method of finding the function y (t) from the x (t).

Following graph shows the noisy time series with the smoothly filtered time series.

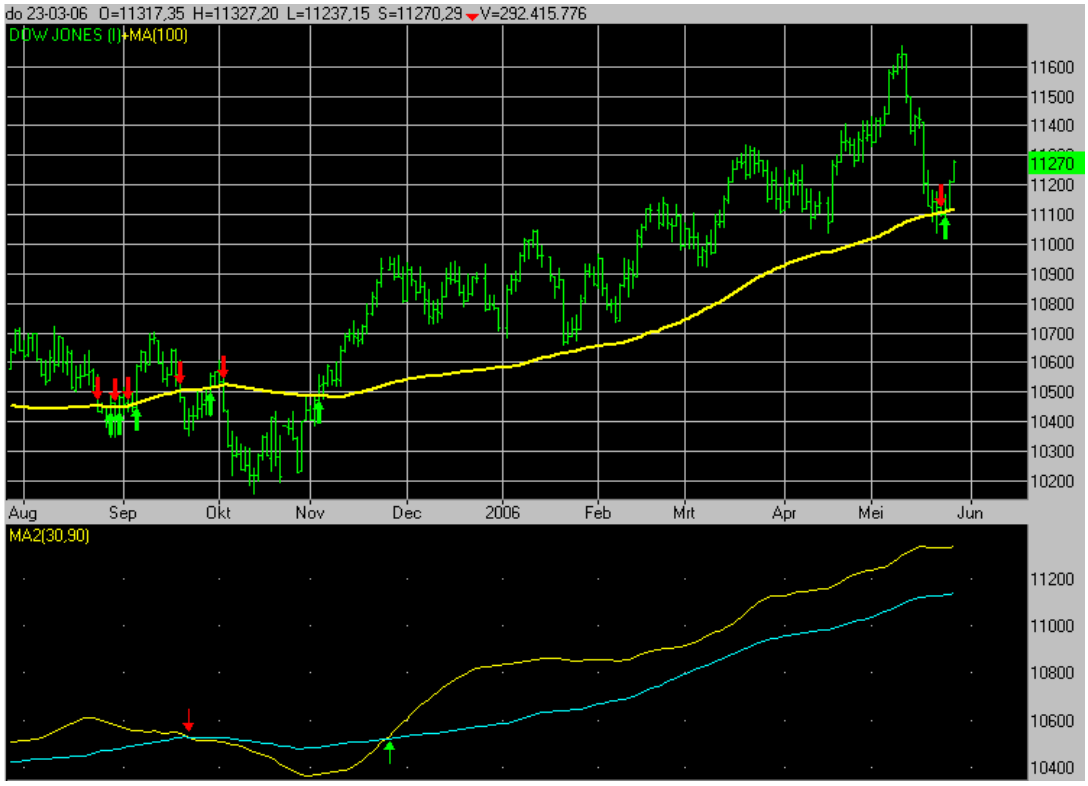


Figure 3-2 Noisy Time Series and Smooth Filter

In two sided moving average, the initial values and final values don't affect the smoothing. This causes problem in forecasting the most recent values. For the calculation of the average 2k+1 observations are used, that is why this is some time known as 2k+1 smoother.

To calculate the four term moving average, the resultant at time t will be

$$f_{t-0.5} = \frac{(y_{t-2}+y_{t-1}+y_t+y_{t+1})}{4} \qquad f_{t+0.5} = \frac{y_{t-1}+y_t+y_{t+1}+y_{t+2}}{4} \qquad (3-5)$$

Where t is time instant and y_t is the value of y at t time instant

In this type of moving average estimation, there are two terms on the right and one or two terms on left and one on the right of the reference point *t*. None of the above formula gives a value at t time instant, so an additional calculation is required as following:

$$f_t = \frac{\left\{\frac{(y_{t-2} + y_{t-1} + y_t + y_{t+1})}{4} + \frac{y_{t-1} + y_t + y_{t+1} + y_{t+2}}{4}\right\}}{2} \qquad (3-6)$$

The above moving average equation is also known as 2*4 moving average. The weight of the above moving average is 5 which correspond to the number of terms.

Generally for the number of terms m, the weight will be m+1. Weight is the effect of the number in an overall average. This is a double moving average estimation since one moving average is smoothened by a second moving average estimate.

The weighted moving average will be given as

$$f(t) = \sum_{j=-k}^{k} a_j * y_{t-j} \qquad (3-7)$$

The main advantage of weighted moving average is that it provides smoother estimation since each value in the series has some weight which can either enhance its effect or reduce in overall average.

In single sided moving average, the forecasting is done using the following formula:

$$y(t+h) = \frac{1}{k+1}\sum_{j=0}^{k} y_{t-j} \qquad (3-8)$$

3.2.2 Kalman Filters

Kalman filter is a well-known mathematical method for the purpose of predicting the time series. Kalman filter is an equation set with prediction and correction approaches [30]. Kalman filter tries to reduce the estimated error covariance. The procedure of estimation with kalman filter is to define a state, which is a discrete time value of a time series.

Suppose A is n x n characteristic matrix of the system, while B is n x l matrix which controls the input for the state x. The variable H is m x n matrix that relates the actual state to the measurement z. The mathematical equation of x_k is

$$x_k = Ax_{k-1} + Bu_{k-1} + w_{k-1} \qquad (3-9)$$

And the measurement z is defined as

$$z_k = Hx_k + v_k \qquad (3-10)$$

w_k and v_k are random, independent and normal distributions in which w_k is a process noise and v_k is measurement noise.

In Kalman filtering, estimation is made with the help of a feedback control. At a point of time, the state of the process is estimated and the feedback is taken from the varying measurements. The mathematical equations of the kalman filter falls into two categories:

- Time update equations
- Measurement update equations

The time update equation can be thought as a predictor equation and the measurement update equation can be thought as a corrector equation.

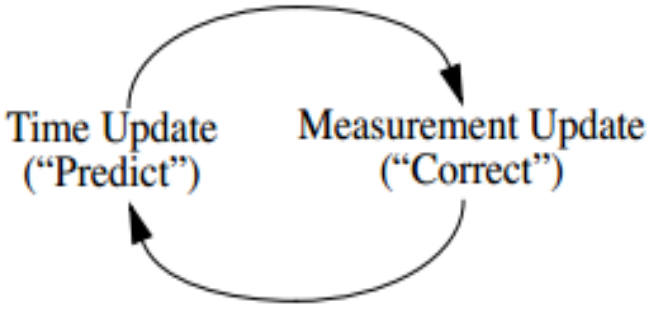


Figure 3-3 Time Update and Measurement Update Cycle

The diagram in Figure 3-4 shows the algorithm flow in terms of the equations. The state x_k is predicted on the basis of the previous state x_{k-1}. A and B are process variables and u_k is the unit state. Q and R are process noise covariance and measurement noise covariance. P is error covariance and K is the Kalman gain.

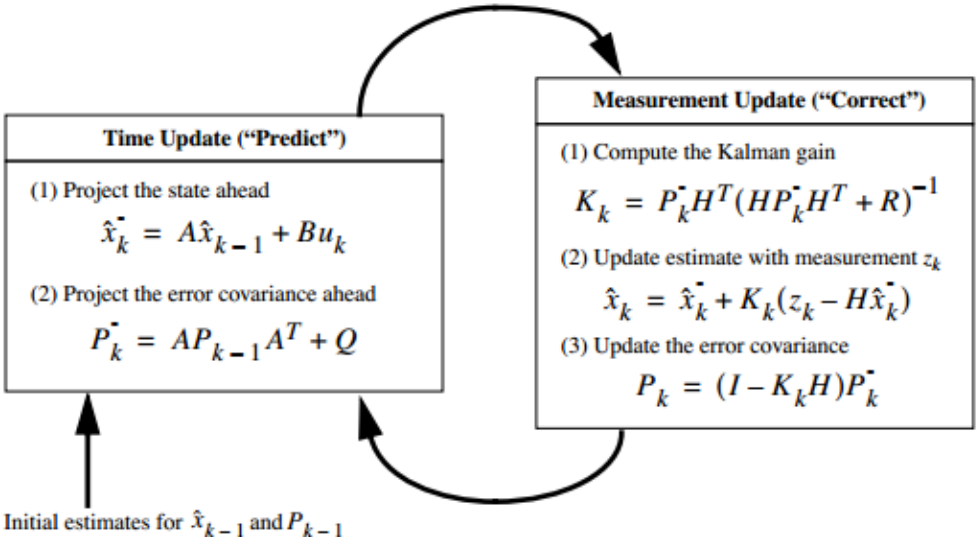


Figure 3-4 Algorithm Flow

Q is process noise and R is the measurement noise. The difference of measurement zk and Hxk is termed as measurement innovation and K is the gain matrix. K is calculated as each time instant for minimizing the estimate error covariance. The mathematical equation for gain is

$$k_k = \frac{P_k H^T}{H P_k H^T + R} \qquad (3-11)$$

Equation 3-11 shows the dependency of gain on estimate error covariance, H and R (measurement noise).

3.2.3 Low pass Filter

The Low pass filter has a smoothening effect on the signal and removes the high changing values from each time series [31]. The Low pass filter allows low frequencies in the signal to pass and stops high frequencies in the signal to create the smoothening effect.

In frequency domain for ideal low pass filter we have:

$$Y = 1 \quad if\ f < freq$$

$$Y = 0 \quad if\ f > freq \qquad (3.12)$$

Where Freq is the frequency limit below which frequencies are allowed to pass and above which frequencies are block by the filter. The response of a low pass filter is shown in Figure 3-5. At low frequencies, the gain of the filter is unity which means that input is equal to the output.

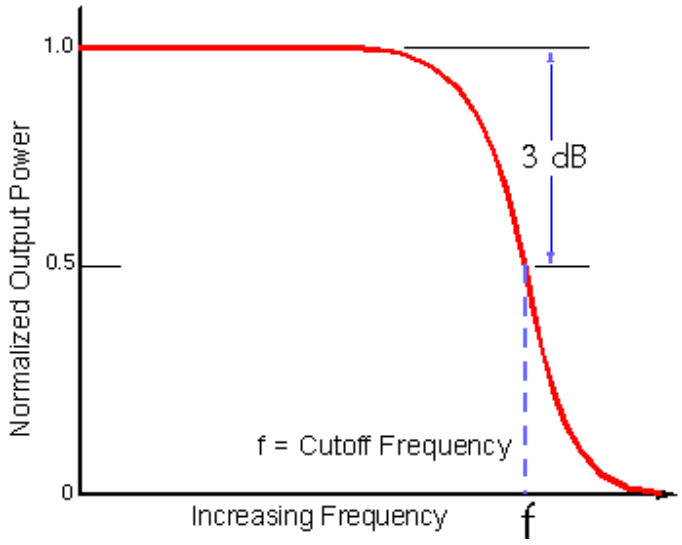


Figure 3-5 Low Pass Filter Response

In the figure above, 'f' is the cutoff frequency at which the magnitude decreases to half that is 0.5.

3.3 Implementation

In this work, some changes were made on Netviz application in synchronizing the time. A detailed description of these changes is presented in the Appendix A. Moreover, different methods were used for better approximation of the data stream available. Each method has different impact on the approximate locations and shows different results in each scenario.

The methods that were considered and implemented in MATLAB were

- Moving Average
- Kalman filter
- Low pass filter

3.3.1 Moving Average

Let's assume(x_i, y_i), where $i = 1,2, \dots, n$ and n is the number of positions in prediction algorithm. Similarly,(x_{vi}, y_{vi}), where $i = 1,2, \dots, n$, represents the coordinates of NetViz positions. The initial position is calculated by taking average of first k of NetViz positions so that the initial error (if any from NetViz visualizer) is minimized [29].

$$x_1 = \frac{\sum_{j=1}^{k} x_{vj}}{k} \qquad (3-13)$$

and

$$y_1 = \frac{\sum_{j=1}^{k} y_{vj}}{k} \qquad (3-14)$$

In the calculation of prediction algorithm, for each point to be calculated should follow another point, called "target point". For the predicted point(x_{i+1}, y_{i+1}), the target points(x_{ti}, y_{ti}), where$i = 1,2, \dots, (n-1)$, are calculated according to the following formula.

$$x_{ti} = \begin{cases} \frac{\sum_{j=i}^{i+n-k} x_{vj}}{n-k+1}, & \text{for } i = 1,2, \dots, (n-k+1) \\ \frac{\sum_{j=i}^{n} x_{vj}}{n-i+1}, & \text{for } i = (n-k+2), (n-k+3), \dots, n \end{cases} \qquad (3-15)$$

and

$$y_{ti} = \begin{cases} \frac{\sum_{j=i}^{i+n-k} y_{vj}}{k}, & \text{for } i = 1,2,\dots,(n-k+1) \\ \frac{\sum_{j=i}^{n} y_{vj}}{n-i+1}, & \text{for } i = (n-k+2),(n-k+3),\dots,n \end{cases} \qquad (3-16)$$

Here k is the number of points in a span which are considered to calculate the average. The span size will be k form $i = 1$ to$(n-k+1)$. From $i = (n-k+2)$ to n, the span size will reduce as the number of remaining points is reducing. The travel path d_i of each target point (x_{ti}, y_{ti}) is calculated as follows,

$$d_i = \begin{cases} 0, & \text{for } i = 1 \\ \sqrt{\left(x_{ti} - x_{t(i-1)}\right)^2 + \left(y_{ti} - y_{t(i-1)}\right)^2}, & \text{for } i = 2,3,\dots,n \end{cases} \qquad (3-17)$$

As mentioned before, the forced direct algorithm creates some unexpected bouncing nodes which cause higher error between the real and the approximate location information. In order to avoid this big jump of a node, the distance d_i is filtered and modified distance $\hat{d}_i$ is determined as follows.

$$\hat{d}_i = \begin{cases} d_i, & \text{if } d_i < d_{max} \\ d_{max}, & \text{if } d_i \geq d_{max} \end{cases} \qquad (3-18)$$

Where,

d_{max} is the threshold value.

In our simulation, the threshold value d_{max} takes the value of 50. The average velocity u, of travel of predicted point is a constant value and determined as shown below,

$$u = \frac{\frac{1}{n}\sum_{i=1}^{n} \hat{d}_i}{t_s} \qquad (3-19)$$

Where,

t_s , is the step time.

Then the angle α_i (for $i = 2,3, \dots, n$) of vector from (x_{i-1}, y_{i-1}) to (x_{ti}, y_{ti}) with respect to x-axis is determined. The angle α_i determines the direction of the node to move. Note that initial angle $\alpha_1 = 0$. The predicted points are then using the following formula.

$$\begin{aligned} x_i &= x_{i-1} + u * t_s * \cos \alpha_i , \quad \text{for } i = 2,3, \dots, n \\ y_i &= y_{i-1} + u * t_s * \sin \alpha_i , \quad \text{for } i = 2,3, \dots, n \end{aligned} \qquad (3-20)$$

3.3.2 Kalman Filter

We implemented the Kalman filter on the approximate location data we obtained from NetViz. The function parameters include system matrix A and the measurement noise, which was set to 0.1 [32]

The Kalman filter implementation in MATLAB works in a loop. For predictions, the following formula is used:

$$x_k = A * x_{k-1} \qquad (3-21)$$

Using the initial values for Matrix A from [32], and making adjustments through trial and error to get the best performance and x_{k-1} is named asx_{meas}in the algorithm which was given the value of

$$x_{meas} = [xkal_1 \quad 0 \quad ykal_1 \quad 0]' \qquad (3-22)$$

$xkal_1$ and $ykal_1$ are the initial values of the vector x and y which are the approximate location values from NetViz. Resultant x_{k-1} is a column vector of size 4*1.

The next step of the implementation in the loop is to find the covariance P which is defined as

$$P_k' = AP_k + A^T + Q \qquad (3-23)$$

P_k is the covariance at any time k. A^T is the transpose of the matrix A, and Q is the process noise which is defined in the starting as a product of scaling factor 0.1 and a matrix of ones and zeros of dimension 2*2.

```
Q=.1*[ones(2,2) zeros(2,2);zeros(2,2) ones(2,2)];
```

After the prediction step, the next step is to go into the series of correction equations as the algorithm of kalman filter demands. In correction stages, the gain of the kalman filter is defined with the help of the equation

$$k_k = \frac{P_k H^T}{H P_k H^T + R} \qquad (3-24)$$

Where H is defined as

$$H = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 \end{bmatrix}$$

And R is the measurement noise defined as constant 0.1

The Kalman gain tends to decrease as the loop execution increases since the predictions starts to come close to correct values. The value of output x_k is updated and corrected on the basis of the gain. The higher the value of gain, the greater will be the correction so the value of k needs to go down with the loop. $Z_k - Hx_k$ is the measurement innovation as defined. This factor also decreases with execution. The correction value of x_k becomes

$$x_k = x_k' + (z_k - Hx_k')K_k \qquad (3-25)$$

At the end of the loop, the value of covariance P_k is updated with the equation mentioned below

$$P_k = (1 - KkH)P_k' \qquad (3-26)$$

3.3.3 Low pass filter

In this case a digital filter is implemented. There are two types of digital filter. One is finite input response and other one is infinite input response. We used finite input response filter in which the denominator is unity and the response of the filter is defined by the coefficients on the filter [31] . The response of the filter is given as

$$h[n] = \frac{y[n]}{x[n]} \qquad (3-27)$$

We used the MATLAB function *fir1* to create a low-pass filter. Fir1 uses standard windowed, linear-phase FIR digital filter design.

It produces a vector b=[b0…..bN] such that :

$$y[n] = b_o x[n] + b_1 x[n-1] + \cdots + b_N x[n-N] \qquad (3-28)$$

$$= \sum_{i=0}^{N} b_i \, x[n-i]$$

The coefficients of the numerator are presented by b. Where y is the filtered result and x is the input. We then used the *filter* function to apply the above filter to our inputs. We defined the cut-off frequency to be the Nyquist-frequency (half the sampling rate).

The response of the filter and its order is defined in the command fir1 which basically determine the number of coefficient for the construction of vector y. if 1st order is selected, there will be two coefficients of vector b. Generally, for Nth order filter, there will be N+1 coefficients of vector b and hence of the filtered vector y. in MATLAB, as mentioned. Fir1 makes the filter coefficients and then filter() command implements the filter on the data to get the filtered data. We implemented the filter on the approximate values gathered from visualization and then after implementation of the filter, the results are compared with approximate values to see the effect of

the low pass filter individually. For different order of filter, the comparison gives different results.

Chapter 4

Performance Evaluation

MATLAB control system toolbox was used for implementation of the filters. There are two sets of simulations to evaluate the coarse locations and filters. We run the simulation on NS-3 to obtain the topology information (full view) and locations file, and then we fed into a visualizer. Node adjacency information and anchor locations are supplied to the Force Direct Algorithm to calculate the approximate locations of the non-anchored nodes.

The visualizer generates a log file for (x_{aprox} , y_{aprox}) coordinates at each time step. Then the approximate locations are compare to (x_{org} , y_{org}) generated by NS-3 to calculate the accuracy of coarse localization.

The error in this case is calculated as

$$distance(error) = \sqrt{\left(x_{org} - x_{aprox}\right)^2 + \left(y_{org} - y_{aprox_Y}\right)^2} \qquad (4-1)$$

We have calculated the direction and speed from the generated approximate locations as it is shown in this section. We applied our implemented filters Moving Average, Kalman Filter and Low Pass Filter to the (x_{aprox} , y_{aprox}) values, and then calculated the new localization errors using the equation 4.1 and compared the error with the original distance errors from netViz. Each filter has its own parameters. We change the parameters for each filter based on the mobility models as shown in table 4-2 using trial and error method to achieve the best result. In our simulations, we assumed that 25% of the Nodes would be anchors with known location information and the rest are non-anchored nodes whose locations are calculated by the Forced directed algorithm.

SIMULATIONS PARAMETERS	
Size of The Area	500*500m
Number of Nodes	30,40,50
Transmission Range	40 m
Wireless Model	IEEE 802.11 a
Delay model	Constant Speed Propagation Delay and Friss Propagation loss Model
Mobility model	Random Way Point Random Walk 2D Random Direction 2D Gauss Markov One Direction
Node Speed	3 m/s 10 m/s for One Direction
Simulation Time	1000sec
Step Time	2sec

Table 4-1 Simulations Parameters

4.1 Random Walk 2D

The parameters used in the filters simulations are based on trial and error method and the effects of different parameters were tested.

Num_Nodes	Anchored Nodes	Non-Anchored Nodes	Moving Average Parameters		Kalman Filter Parameters		Low Pass Filter Parameters	
30	8	22	*k*	5	Q	.1	h	20
			span	100	R	60		
40	10	30	*k*	5	Q	.1	h	20
			span	150	R	150		
50	13	37	*k*	5	Q	.1	h	50
			span	100	R	10		

Table 4-2 Random Walk 2D Parameters

Figure 4-1, Figure 4-2 and Figure 4-3 shows the errors in distance, the real and approximated movement of non-anchored node between coarse location (NetViz) and moving average for various network sizes. At the beginning of the simulation the error rate drops down exponentially and that with the help of setting initial position from NetViz. Moving average removes all the jumps on coarse locations and the movement became smoother. We took the average error rate for one scenario i.e. 30, 40 and 50 nodes, and the error rate at times could be as little as 10m. The

parameters for each scenario of the nodes are different, depends on the movement of each node where the node is independent of other nodes.

NetViz data and the result of the moving average are shown in the Figure 4-1. The graph of the figure shows the actual movement of nodes. The filters were applied on the NetViz data only, not on the original data. The moving average filter response is also shown in the second graph. From the original data which is only used as a reference, the error is calculation is done.

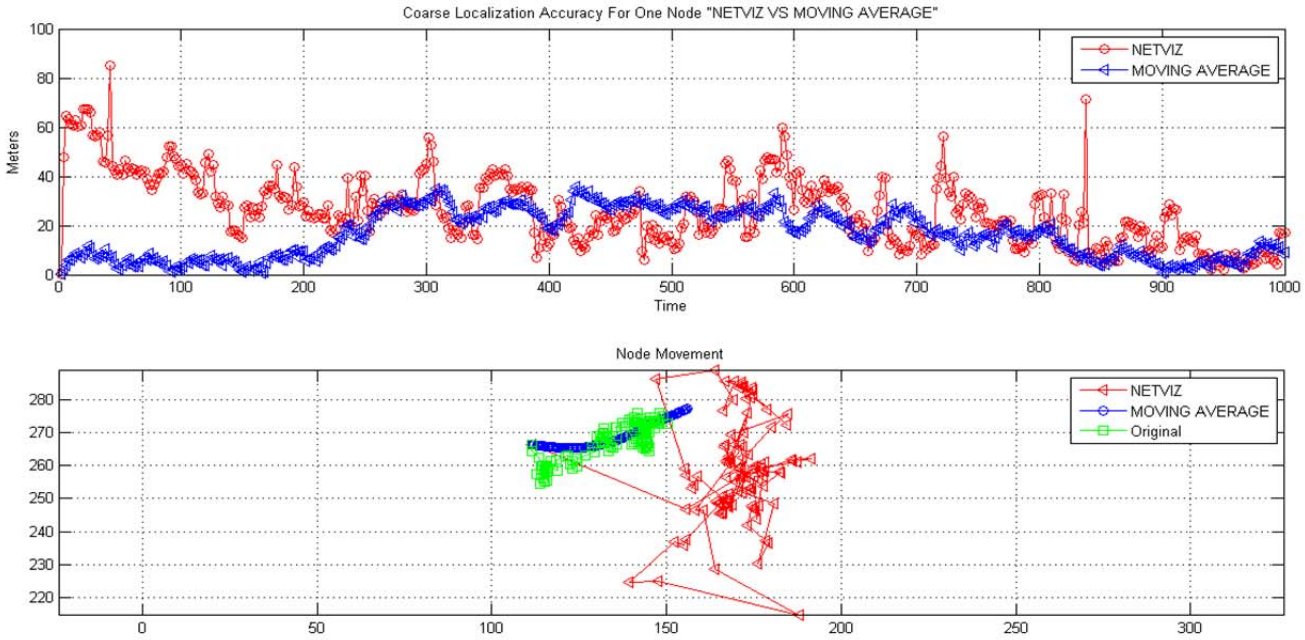


Figure 4-1 Moving Average vs. NetViz for One Scenario of 30 Nodes

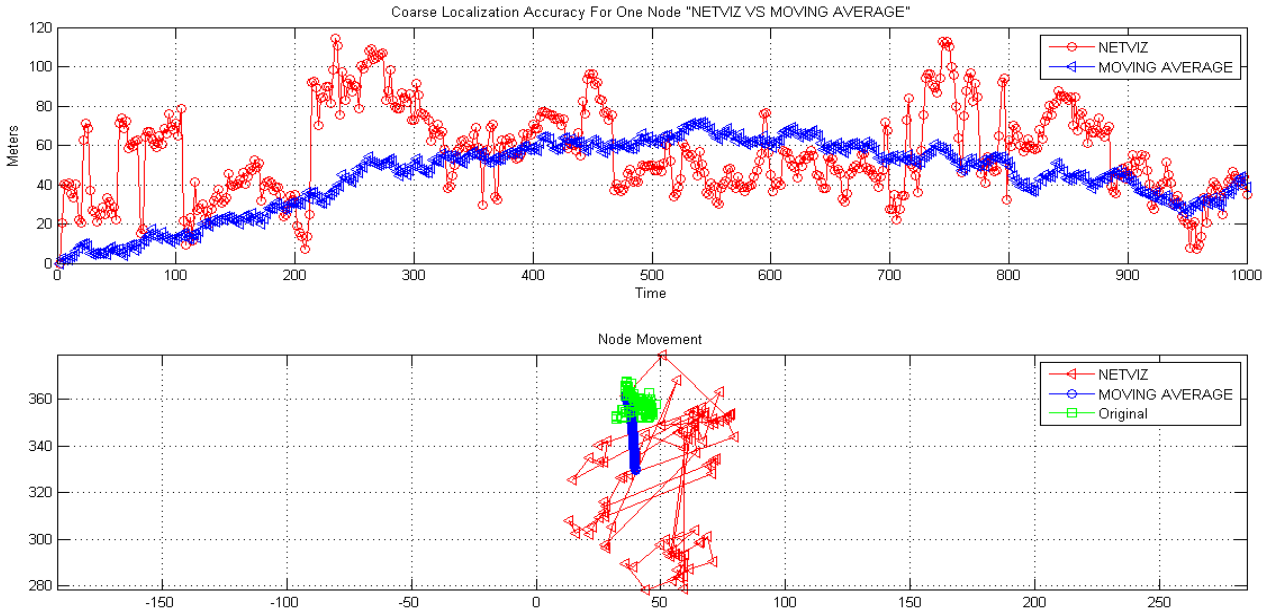


Figure 4-2 Error Moving Average vs. NetViz for One Scenario of 40 Nodes

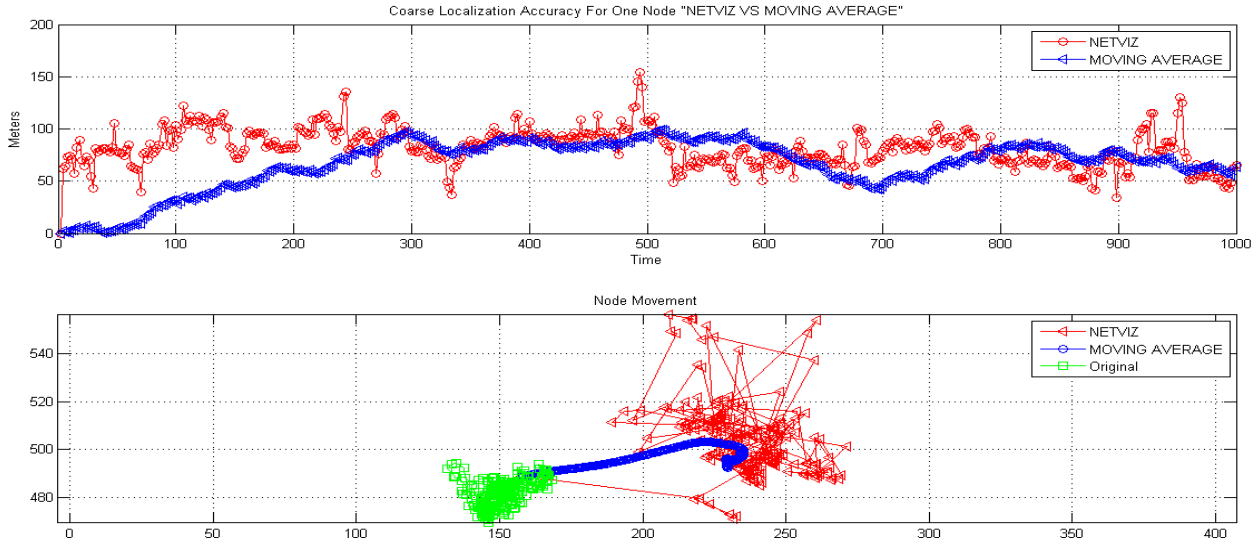


Figure 4-3 Error Rate Moving Average vs. NetViz for One Scenario of 50 Nodes

Figure 4-4, Figure 4-5 and Figure 4-6 show the error rate and nodal movement for Kalman Filter. By taking the average of the error rate of one scenario of 30, 40 and 50 nodes the error decreased over time to approximately 5m. Kalman Filter is not completely removing all the bouncing from the coarse location. Due to the large Euclidean distances between the approximate points Kalman Filter will not be able to remove all the noise unless points are close to each other.

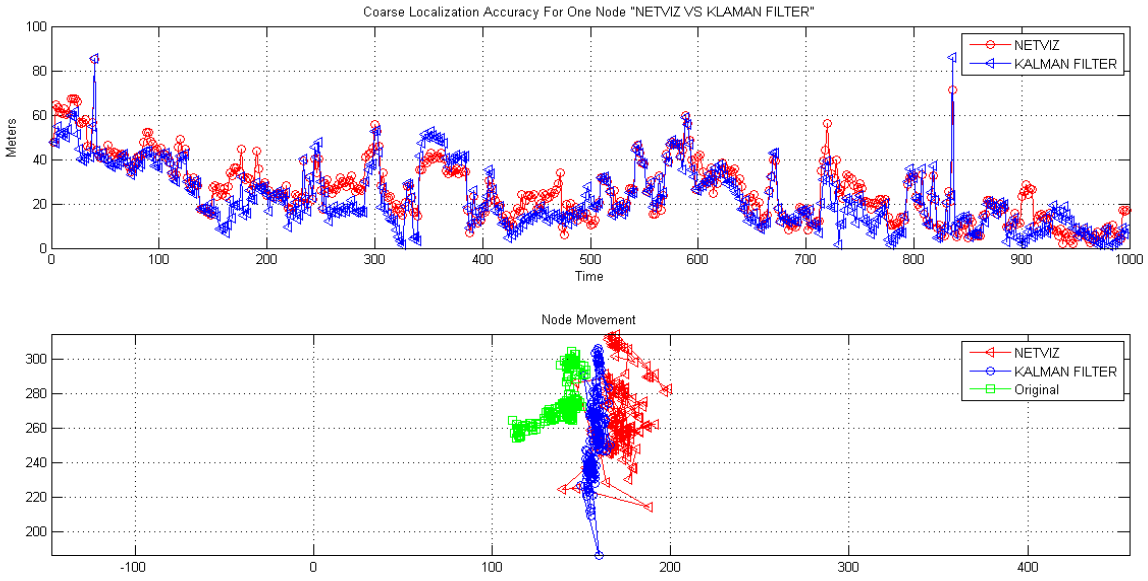


Figure 4-4 Error Rate Kalman Filter vs. NetViz for One Scenario of 30 Nodes

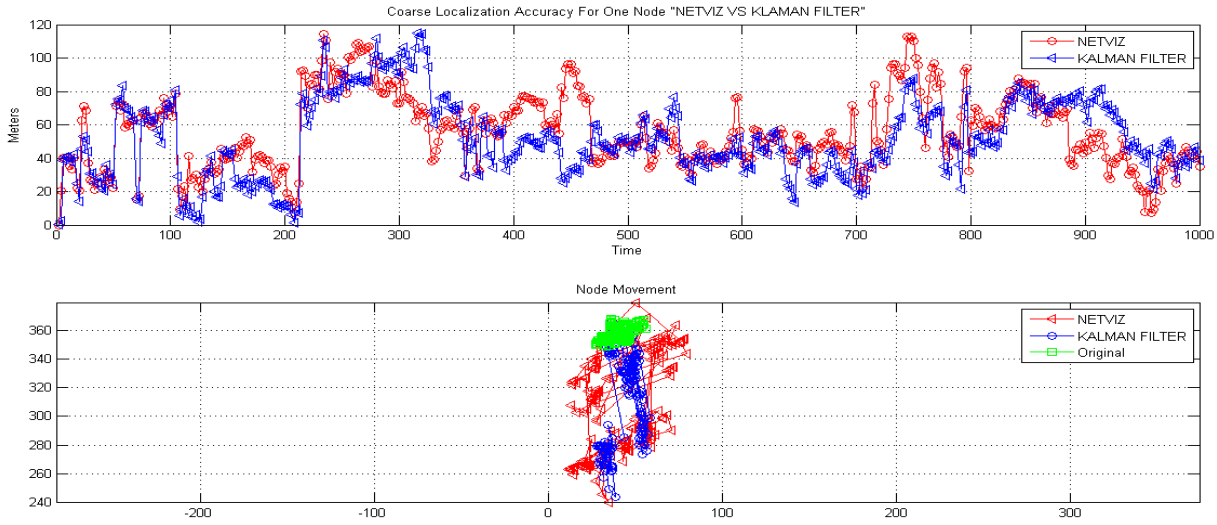


Figure 4-5 Kalman Filter vs. NetViz for One Scenario of 40 Nodes

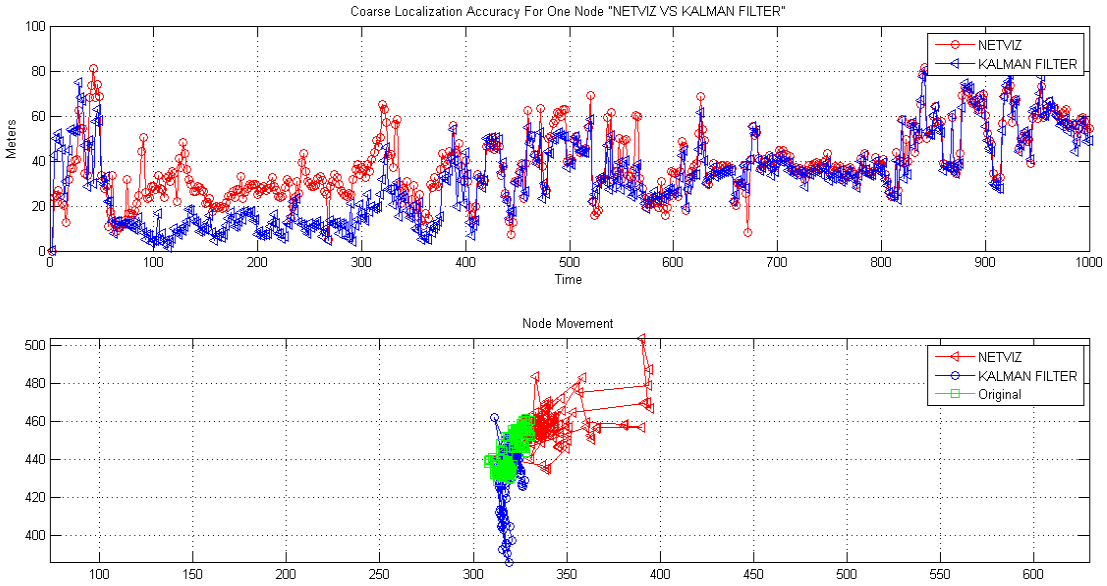


Figure 4-6 Error Rate Kalmam Filter vs. NetViz for One Scenario of 50

Figure 4-7, Figure 4-8 and Figure 4-9 present the error rate and nodal movement for the Low Pass Filter in Random Walk mobility scenario. Low Pass Filter is able to remove the jumps, and make the movement smoother. The average was taken between coarse location and Low Pass Filter and the result shows the error rate over time could go down to 3m. Changing the size of the window of the filter or bandwidth, (h) will change the initial position and movement of node.

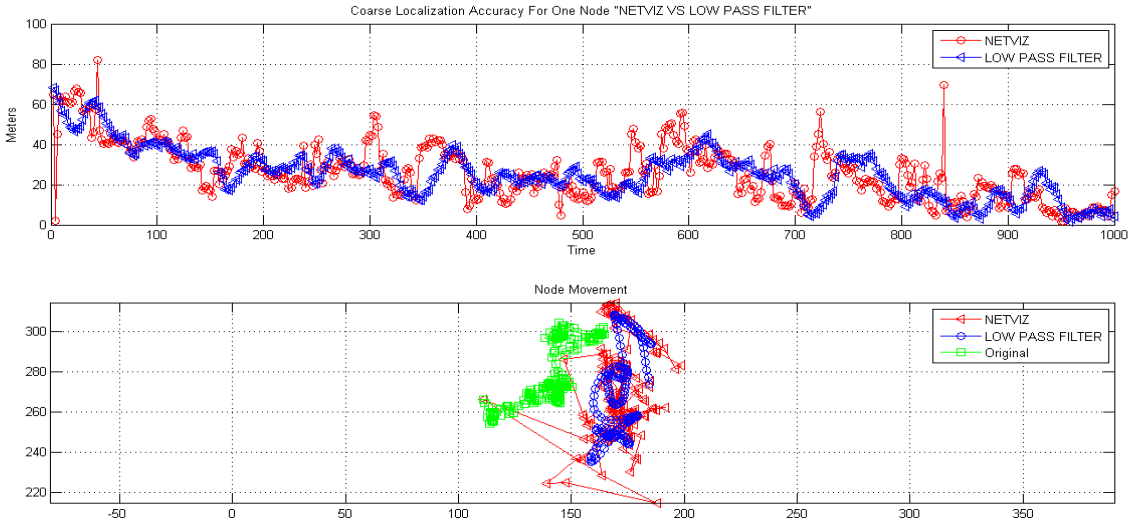


Figure 4-7 Error Rate Low Pass Filter vs. NetViz for One Scenario of 30 Nodes

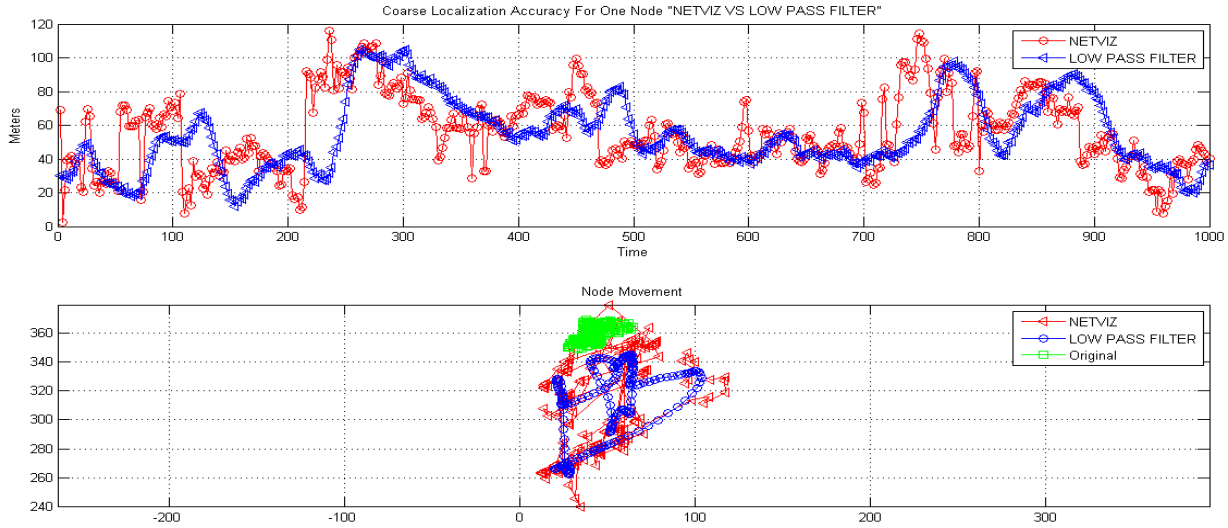


Figure 4-8 Error Rate Low Pass Filter vs. NetViz for One Scenario of 40 Nodes

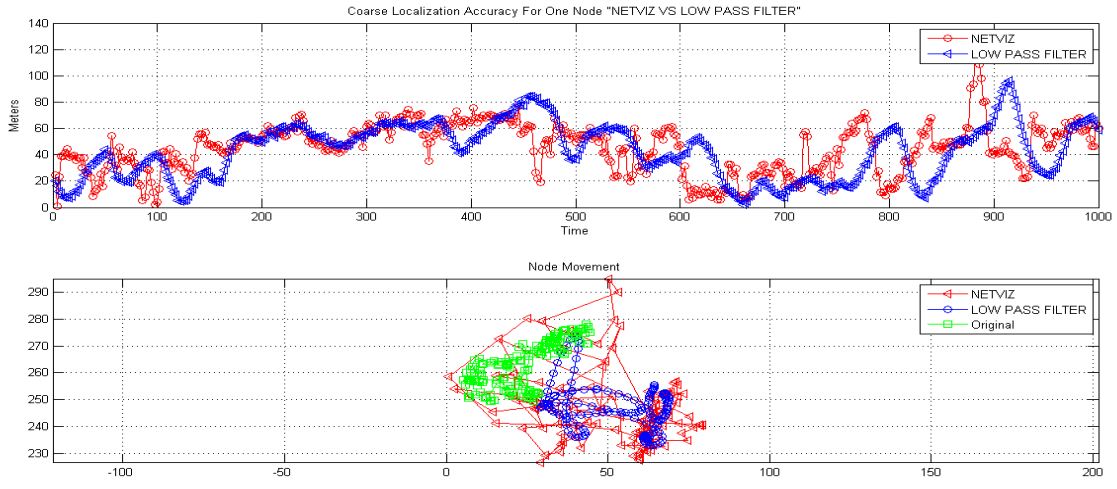


Figure 4-9 Error Rate Low Pass Filter vs. NetViz for One Scenario of 50 Nodes

We calculated the average error in distance for all Non-anchored nodes to show the total error for each coarse location, moving average, Kalman filter and low pass filter.

Let's assume

$ni: Node\ of\ i$

$Na: number\ of\ non-anchord\ node$

$ni: has\ location\ at\ (x_i, y_i)$

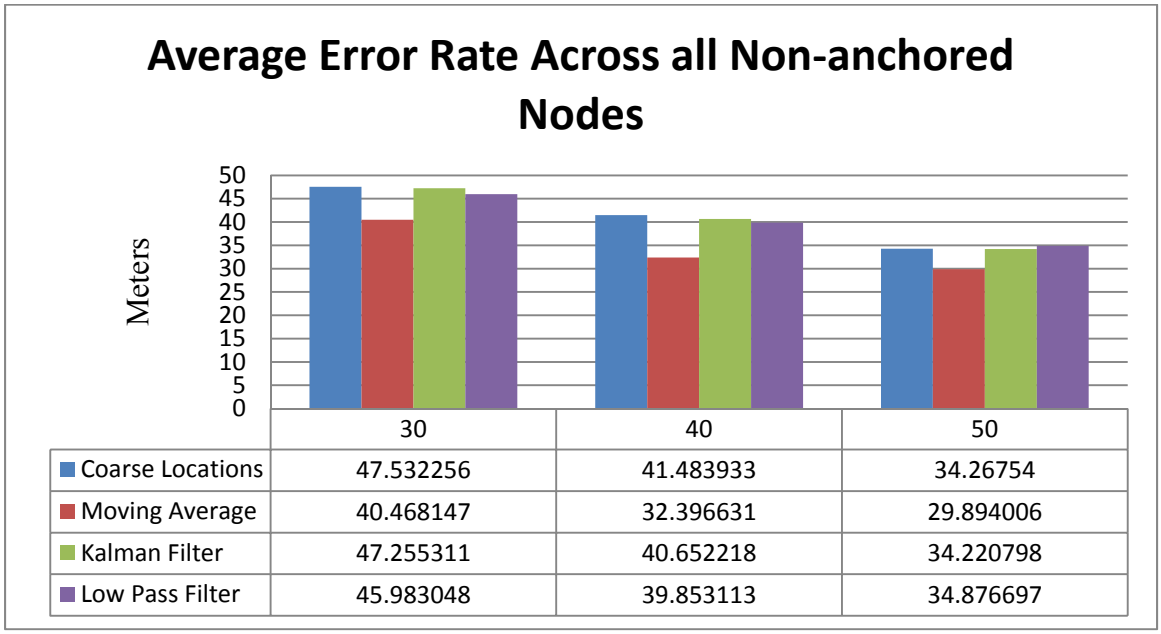
$nj: has\ location\ at\ (x_j, y_{ji})$

$$d_{ij} = \sqrt{(x_i - x_j)^2 + (y_i - y_j)^2} \qquad (4-2)$$

$$\sigma = \sum_{k=0}^{Na} dk \qquad (4-3)$$

$$\bar{\sigma} = \frac{\sigma}{Na} \qquad (4-4)$$

The results shows the impact of the parameters for a particular filter on each individual node when we calculated the error for one node the error dropped significantly, however the total average's error for the entire non-anchored nodes still decrease or increase like Low Pass Filter with 50 nodes. In order to have a better predictive method the filters should be applied to each node with different parameters as it is shown in Figure 4-10.



	30	40	50
Coarse Locations	47.532256	41.483933	34.26754
Moving Average	40.468147	32.396631	29.894006
Kalman Filter	47.255311	40.652218	34.220798
Low Pass Filter	45.983048	39.853113	34.876697

Figure 4-10 Random walk 2D

4.2 Gauss Markov

Num_Nodes	Anchored Nodes	Non-Anchored Nodes	Moving Average Parameters		Kalman Filter Parameters		Low Pass Filter Parameters	
30	8	22	k	5	Q	.1	h	1
			span	30	R	.1		
40	10	30	k	5	Q	.1	h	1
			span	30	R	.1		
50	13	37	k	5	Q	.1	h	1
			span	30	span	.1		

Table 4-3 Gauss Markov Parameters

Figure 4-11 shows the error rate and node's movement for coarse locations and moving average for Gauss Markov mobility model. Moving Average still gives a better estimation by removing the bounces generated by Force Direct Algorithm. The Figure shows one scenario of 30 nodes, where the average error rate of moving average vs. coarse locations decreased up to 6m. The initialization for nodes through NetViz helps to improve the results in the case of using moving average. From visualizing the node's movement, we notice that original positions are moving faster than approximate, hence it causes the high error ratio in both cases.

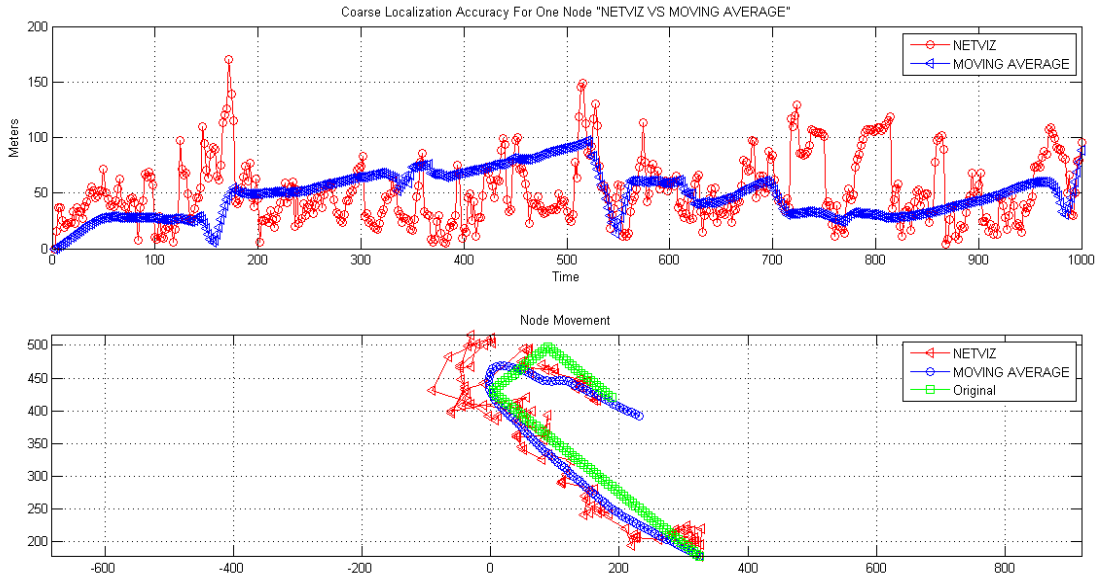


Figure 4-11 Moving Average vs. NetViz for One Scenario

As it is shown in Figure 4-12, the error rate between NetViz and Kalman filter for one node has no significant difference in the changes. The reason is, related to the force direct algorithm, where is a large Euclidean distances. If there were not jumps between positions Kalman Filter will perfume perfectly as it is coming on section 5.7, where we applied Kalman Filter on the top of moving average. Changing the measurement noise R will affect the new positions where it will be moving far apart from the approximate locations, due to the arbitrary movement of nodes as it is shown in Figure 4-13.

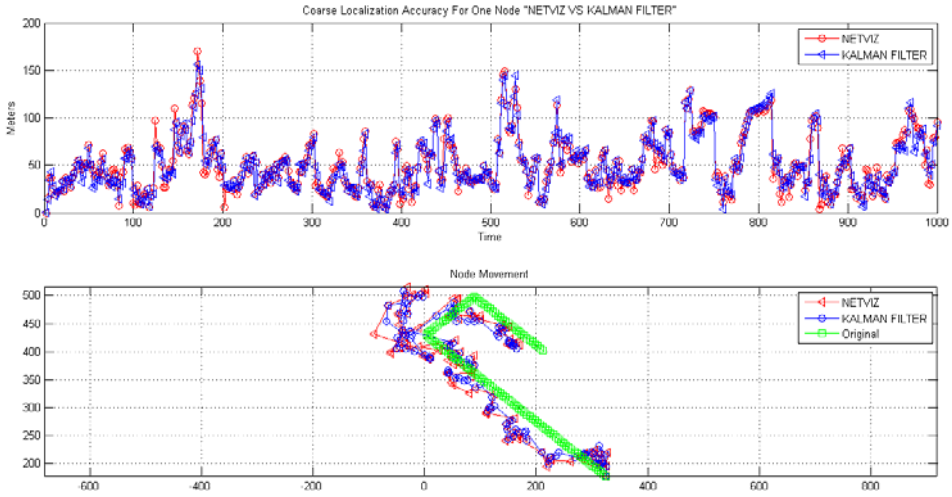


Figure 4-12 Kalman Filter vs. NetViz for One Scenario

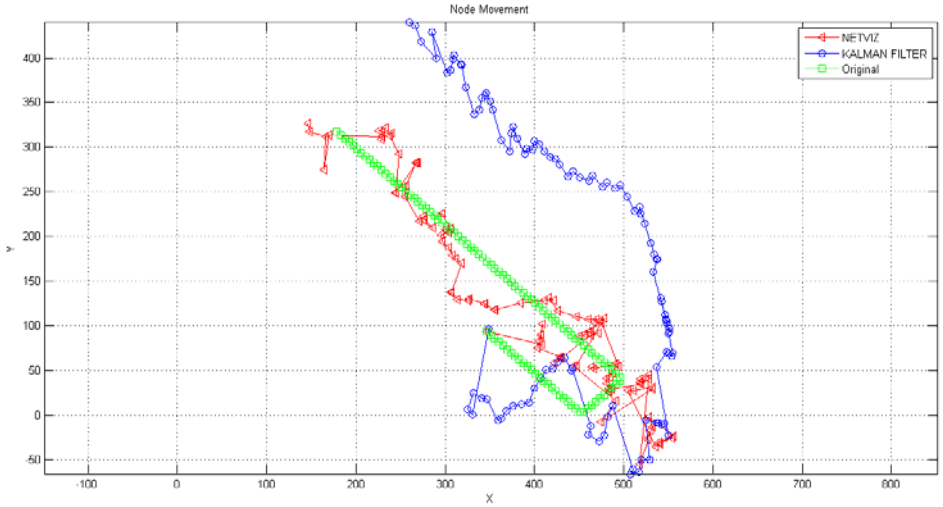


Figure 4-13 One Non-Anchored Movement Kalman Filter

We set the size of window (h) to 1 to remove the randomness jumps; however the average errors rate shows the coarse locations gives better results than Low pass Filter as it is shown in Figure 4-14. Due to jumps in the approximate locations when we increase the size of the window, the new locations generated by Low Pass Filer will be further away from the original data as it is shown on Figure 4-15.

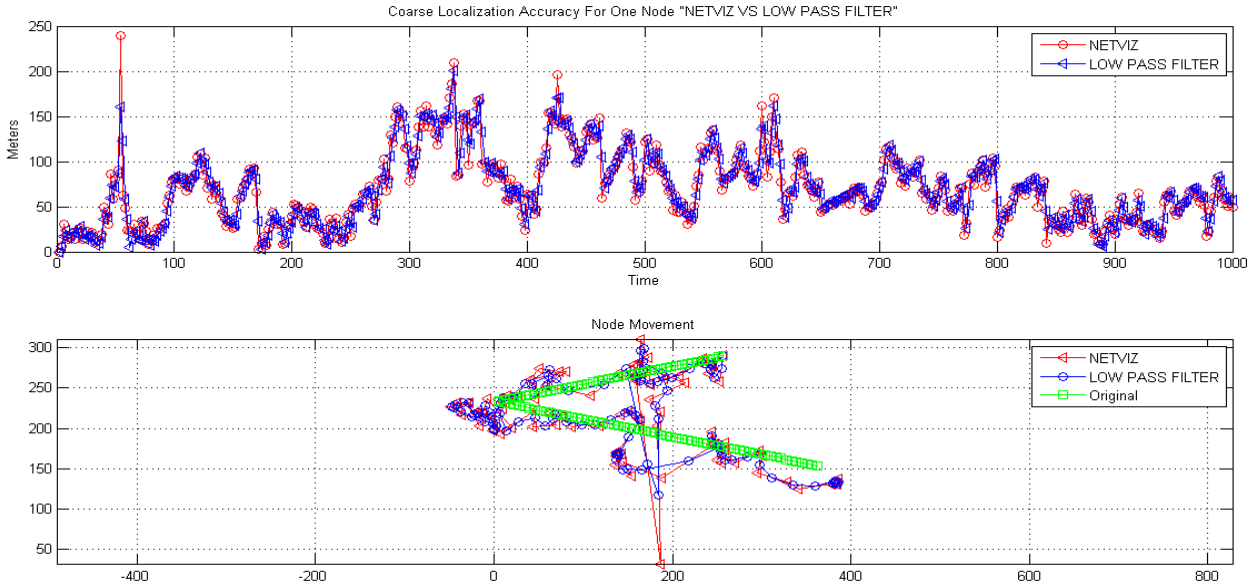


Figure 4-14 Low Pass Filter vs. NetViz for One Scenario

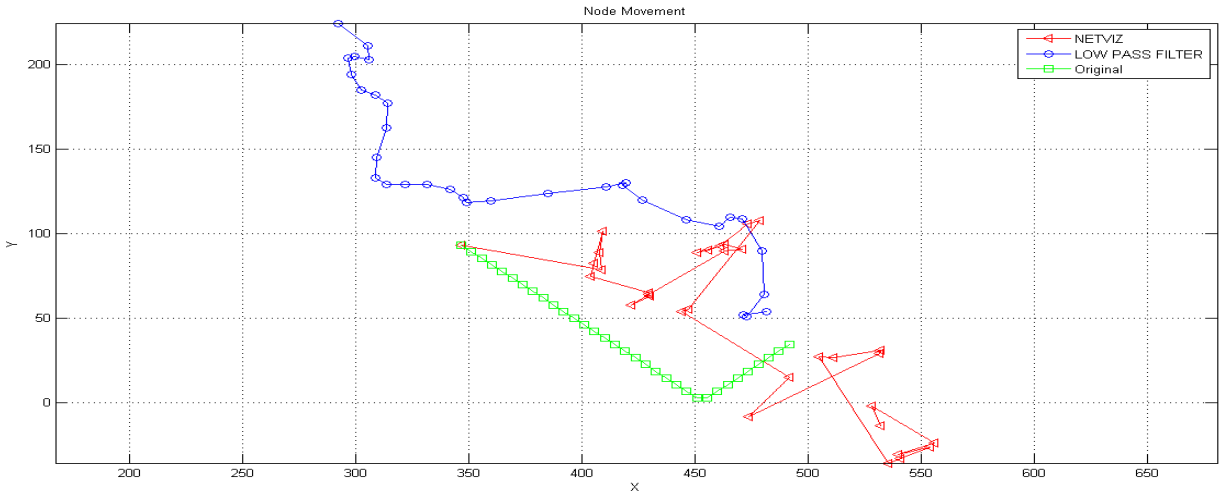
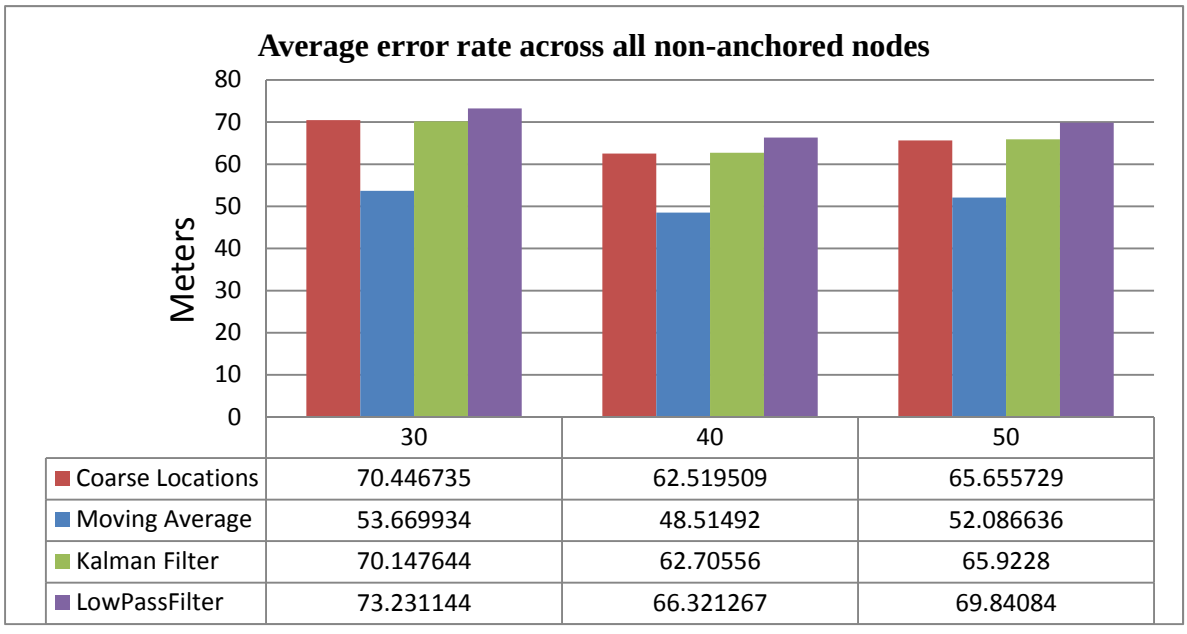


Figure 4-15 One Non-Anchored Movement Low Pass Filter

The total average rate for all the non-anchored nodes shows that moving average has the best results for the entire scenarios. Kalman Filter still kept close to the coarse locations while Low Pass Filter increased the error rate.



	30	40	50
Coarse Locations	70.446735	62.519509	65.655729
Moving Average	53.669934	48.51492	52.086636
Kalman Filter	70.147644	62.70556	65.9228
LowPassFilter	73.231144	66.321267	69.84084

Figure 4-16 Gauss Markov

4.3 Random Way Point

Num_Nodes	Anchored Nodes	Non-Anchored Nodes	Moving Average Parameters		Kalman Filter Parameters		Low Pass Filter Parameters	
30	8	22	*k*	5	Q	.1	h	1
			span	15	R	.006		
40	10	30	*k*	5	Q	.1	h	1
			span	10	R	.1		
50	13	37	*k*	5	Q	.1	h	1
			span	30	R	.2		

Table 4-4 Random Way Point Parameters

Moving average has different parameters for each mobility model depends on the way they move. The error rate for Random Way Point for each node decreased up to 10m comparing to the coarse locations. Figure 4-17 shows how the moving average removes most of the unwanted bouncing from the approximate locations. The entire scenarios for moving average shows obvious changes in the initializing of the simulation due to the setting of initial points through approximate locations.

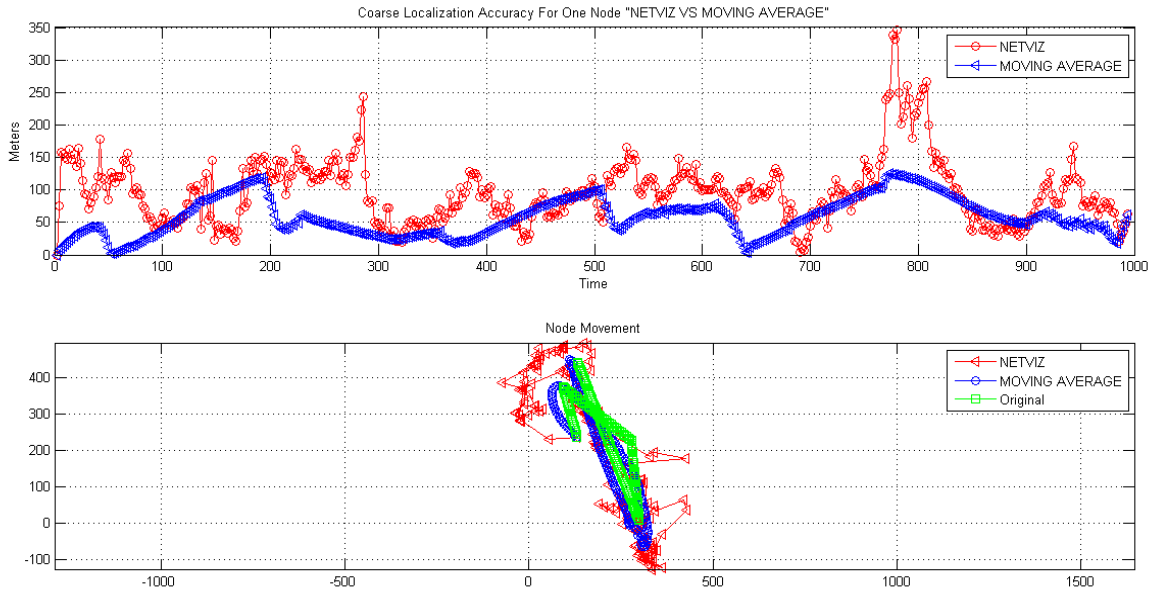


Figure 4-17 Moving Average vs. NetViz for One Scenario

Kalman filter decreased the error rate slightly from the coarse locations. In order to have a good prediction with the filter, there should not be jumps between the points. In Figure 4-18 the average of the error rate for Kalman filter decreased about 500cm comparing to the coarse location.

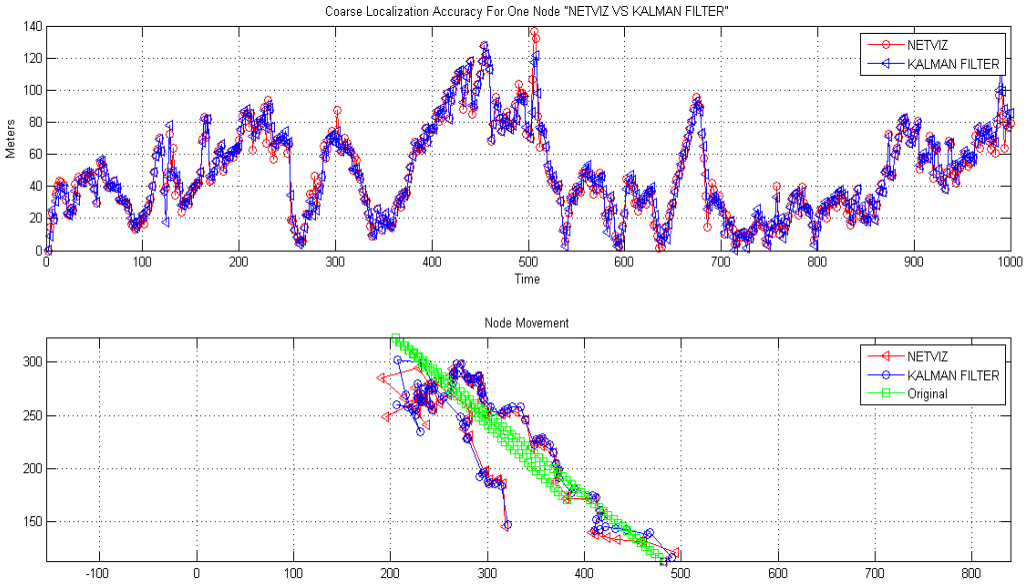


Figure 4-18 Kalman Filter vs. NetViz for One Scenario

The results in Figure 4-19 show that Low Pass Filter has no impact on the coarse locations. It just decreases or increases the error rate a slight. The filter will illustrate decent results if there were not distance between approximate points (jumps).

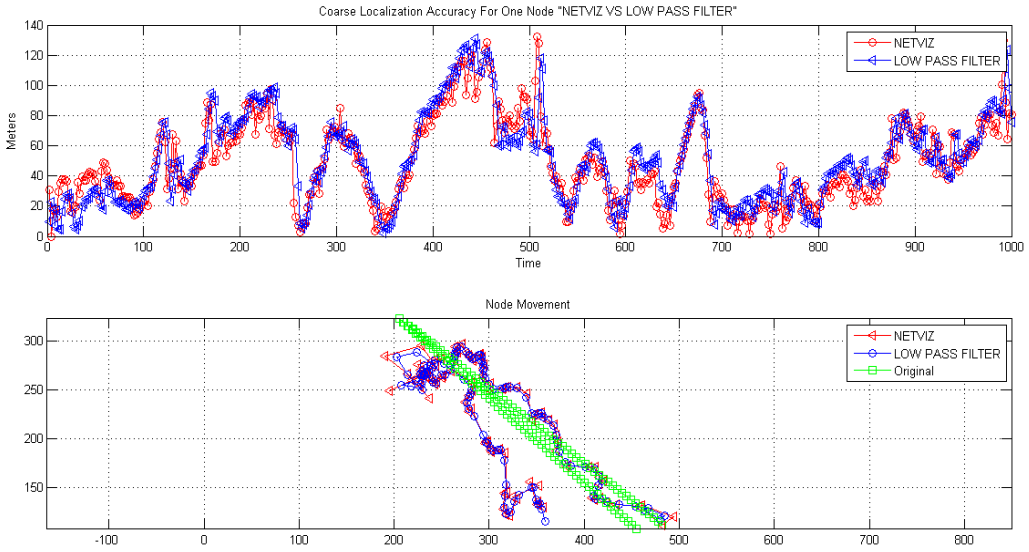
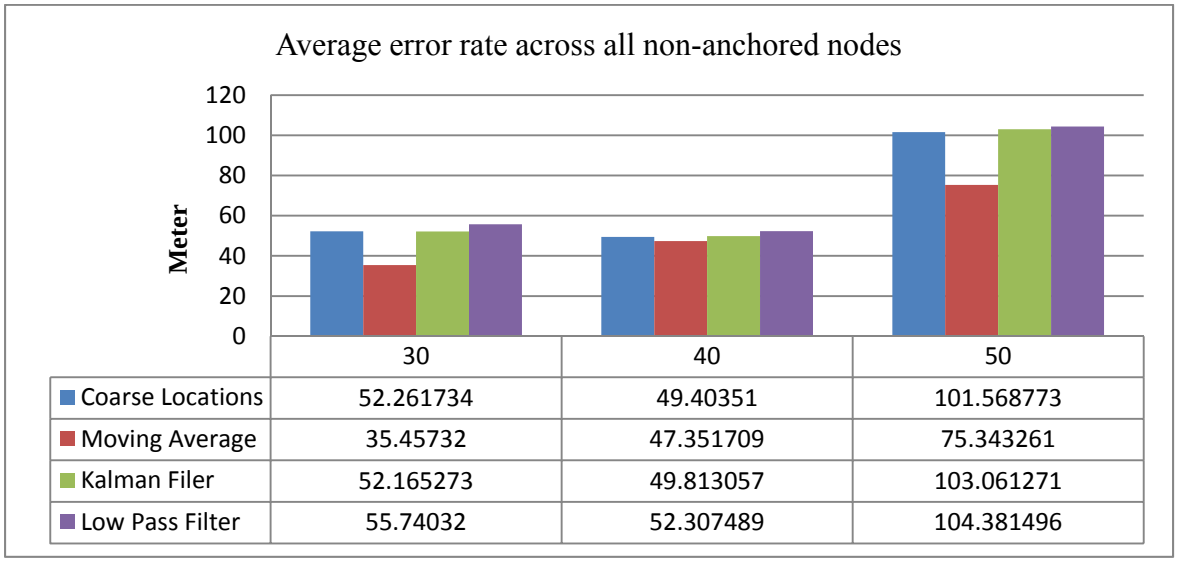


Figure 4-19 Low Pass Filter vs. NetViz for One Scenario

The average error rate across all the non-anchored nodes shows that moving average decreased the error up to 25m with 50 nodes, 3m with 40 nodes and up to 16m with 30 nodes. For kalman filter and low pass filter error rate is increased about 3m comparing to the coarse locations as it is shown in Figure 4-20.



	30	40	50
Coarse Locations	52.261734	49.40351	101.568773
Moving Average	35.45732	47.351709	75.343261
Kalman Filer	52.165273	49.813057	103.061271
Low Pass Filter	55.74032	52.307489	104.381496

Figure 4-20 Random Way Point

4.4 Random Direction 2D

Num_Nodes	Anchored Nodes	Non-Anchored Nodes	Moving Average Parameters		Kalman Filter Parameters		Low Pass Filter Parameters	
30	8	22	*k*	5	Q	.1	h	3
			span	15	R	.1		
40	10	30	*k*	5	Q	.1	h	4
			span	30	R	.1		
50	13	37	*k*	5	Q	.1	h	1
			span	30	R	.1		

Table 4-5 Random Direction 2D

Moving average, removed the unexpected bouncing from the approximate locations. Two factors were significant to decrease the error rate in the initialization of the simulation were taking the average of the first five points and initial points set through NetViz. Figure 4-21 shows the results for moving average and coarse locations, where the average error rate decreased up to 10m for each node with different parameters in Random Direction.

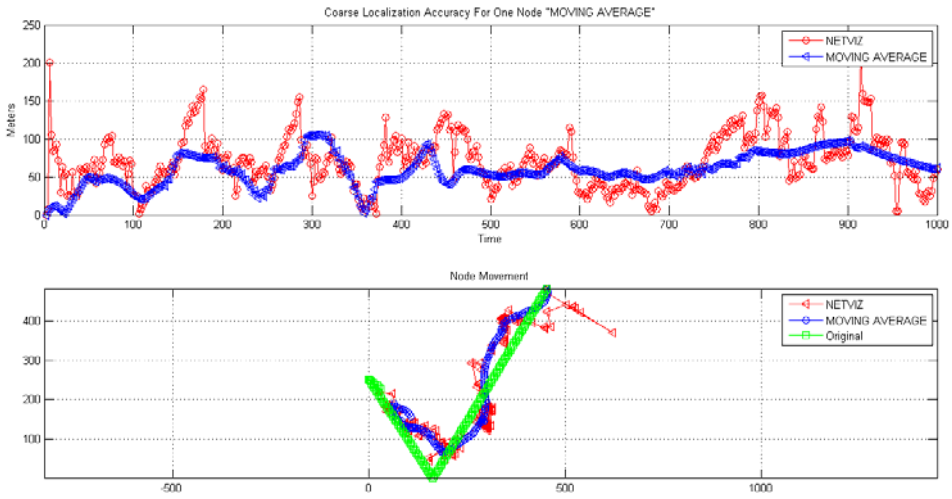


Figure 4-21 Moving Average vs. NetViz for One Scenario

Kalman Filter uses to remove the noise and give a good prediction however in the case where there are large Euclidean distances increasing the noise measurement R will increase the error between Kalman and the original positions. Figure 4-22 presents the difference in the distance between Kalman Filter and coarse locations in Random Direction, where the average error rate was about 300cm.

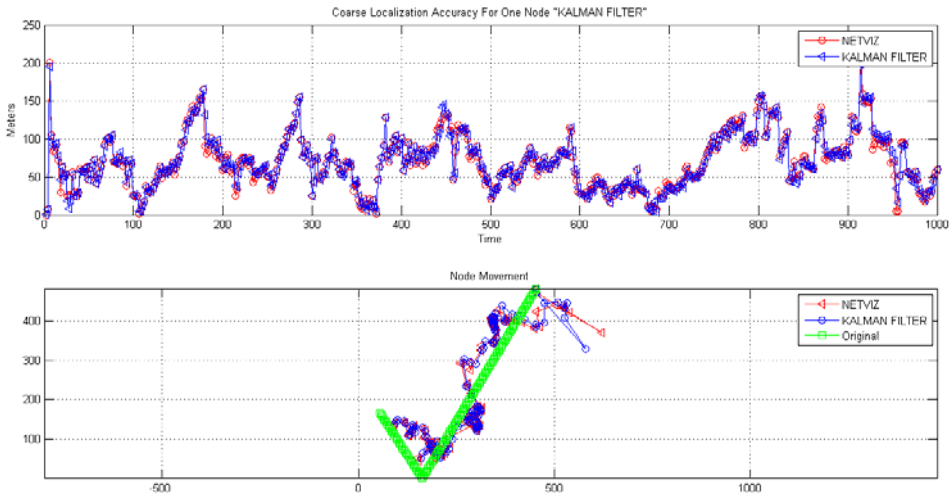


Figure 4-22 Kalman Filter vs. NetViz for One Scenario

Low Pass filter still has no effects on the coarse location for Random Direction. It just increases the error rate with a minor change. Figure 4-23 shows the error rate between the filter and coarse locations, where average error rate is increased about 2m comparing to original data.

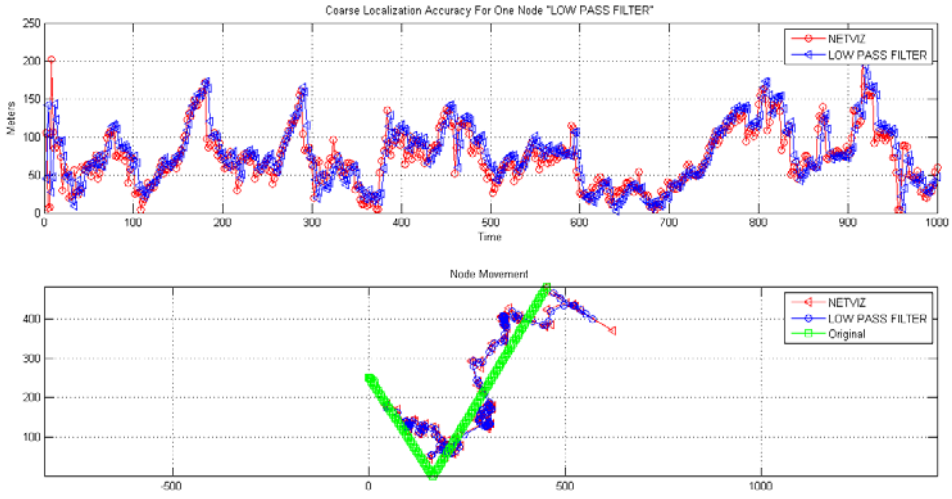
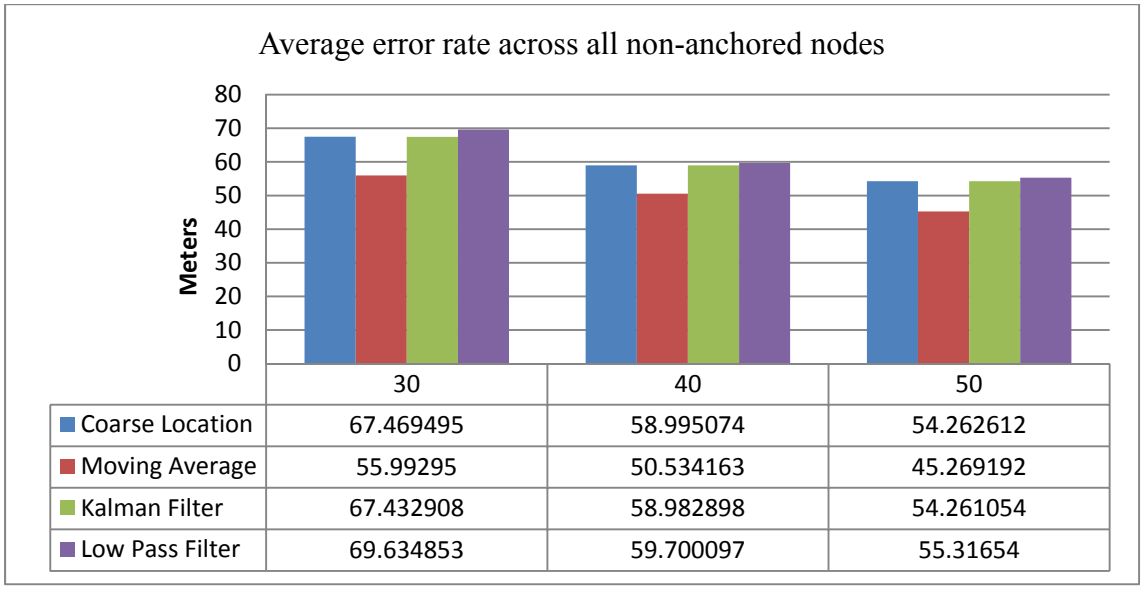


Figure 4-23 Low Pass Filter vs. NetViz for One Scenario

Figure 4-24 represents the error rate between the entire non-anchored nodes for Random Direction. Moving average still gives good results unlike low pass filter and kalman Filter where the results decreased or increased with different scenario.



	30	40	50
Coarse Location	67.469495	58.995074	54.262612
Moving Average	55.99295	50.534163	45.269192
Kalman Filter	67.432908	58.982898	54.261054
Low Pass Filter	69.634853	59.700097	55.31654

Figure 4-24 Random Direction 2D

4.5 One Direction

Num_Nodes	Anchored Nodes	Non-Anchored Nodes
30	8	22
40	10	30
50	13	37

Table 4-6 One Directions

For one direction mobility models the coarse locations shows a great results, where the error rates are between 15m to 25m as it is shown in Figure 4-25. The only different between the original data and the approximate is the nodes are located themselves in different positions however all the nodes move in the same route with some jumps along the path. It is hard to apply filters to the approximate locations due to the movement pattern where the jumps are being made on a straight line.

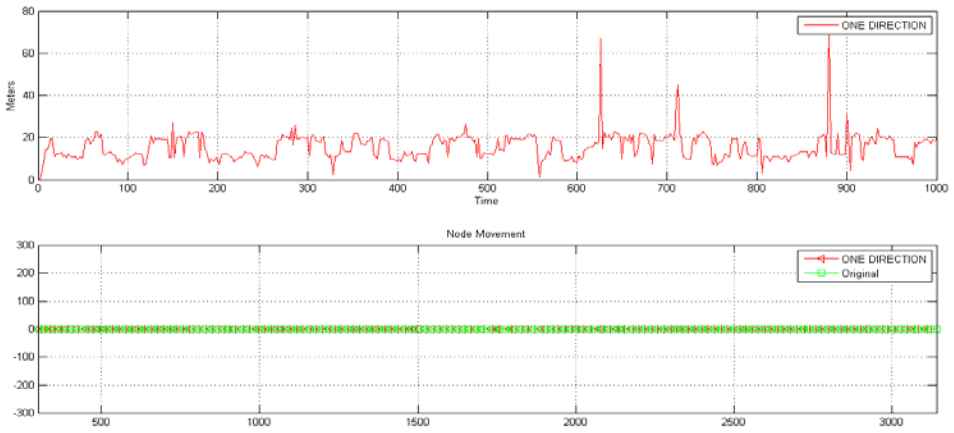


Figure 4-25 One Scenario for One Direction

Figure 4-26 shows the results for the total number of non-anchored nodes where the results of each scenario are between 15 to 23m for coarse locations.

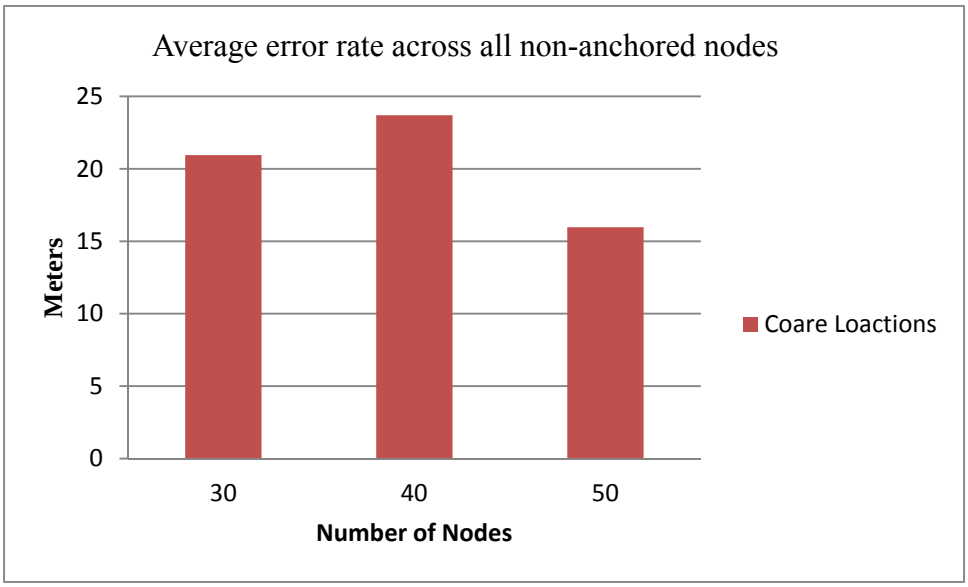


Figure 4-26 One Direction

Moving Average vs. Kalman Filter

As we discussed in sections 4.2, 4.3, 4.4, 4.5, we observed that Kalman filter was not efficiently performed due to the arbitrary movement of the node. Kalman filter and low pass filter worked only on one mobility model which is Random walk. Nodes in Random walk travel within a constant distance and time. There is no pause time hence the points are close to each other. Other mobility models they are moving in straight line until they reach a point to change the direction with no defined distance or time.

In order to get further improvement in the results, we applied Kalman filter to the moving average results, where the bounces had been removed. The results give significant improvement in the case we apply Kalman filter on each individual node however when we calculated the total average for all non-anchored still the results are not promising unless we change the parameters of Kalman filter for each node based on trial and error method.

4.5.1 Random Walk 2D

We change the parameters for A and R where A is responsible for system status and R is for the noise measurement. The default initial values for A was taken from the implementation of Random Kalaman Filter process [32], where A was set to

A= [1 0 0 0

0 1 0 0

0 0 1 0

0 0 0 1]

And R= 0.1

The filter used parameters from [32] and then adjusted them for each case by trial and error. 'A' helps to get the moving average locations to be close to the original data by changing the first value of the matrix. The error rate for one scenario of 40 nodes shows in Figure 4-27, where the error rate decreased for moving average about 3m and 7m in Kalman filter.

A= [14 0 0 0

0 1 0 0

0 0 1 0

0 0 0 1]

R = .006

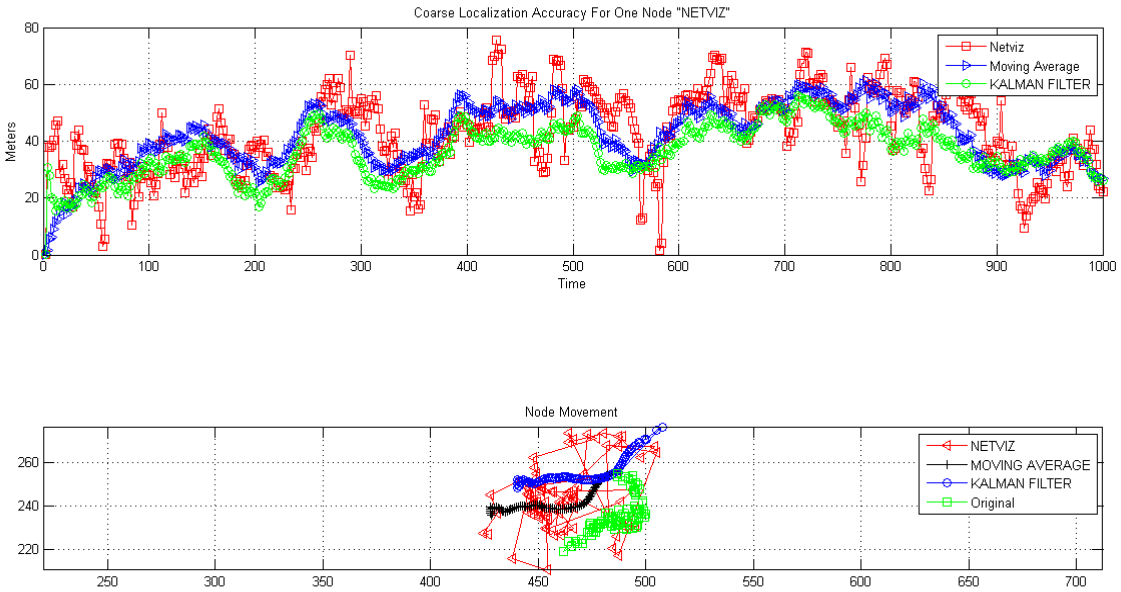


Figure 4-27 Kalman Filter vs. Moving Average

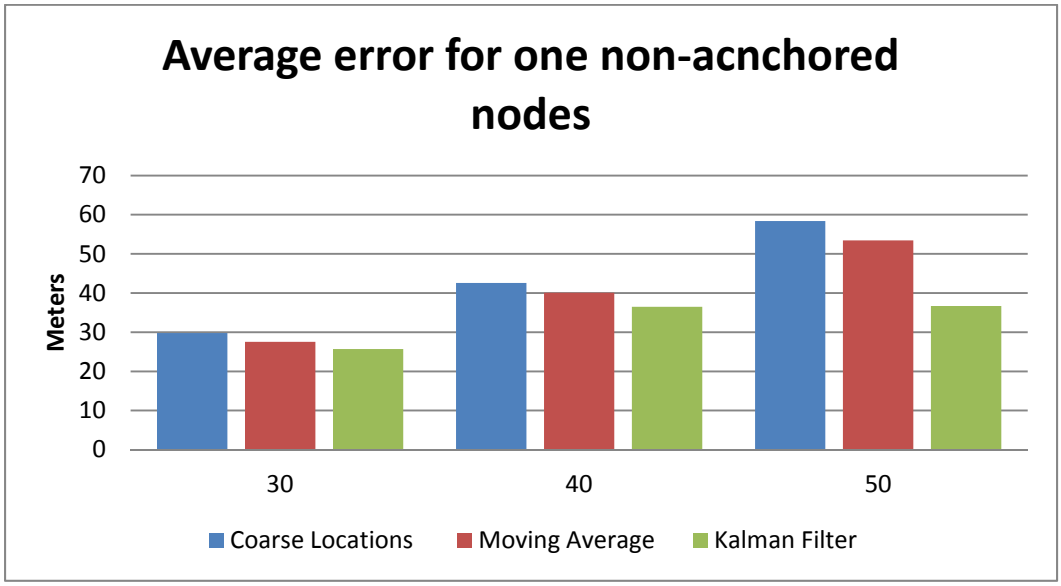


Figure 4-28 Random Walk 2D

4.5.2 Guess Markov

The parameters were used in Kalman filter for Guess Markov mobility model are

A =

$$A = \begin{bmatrix} 0.5 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

R=.006

Figure 4-29 shows the average error for Kalman filter where it decreased about 4m from the moving average.

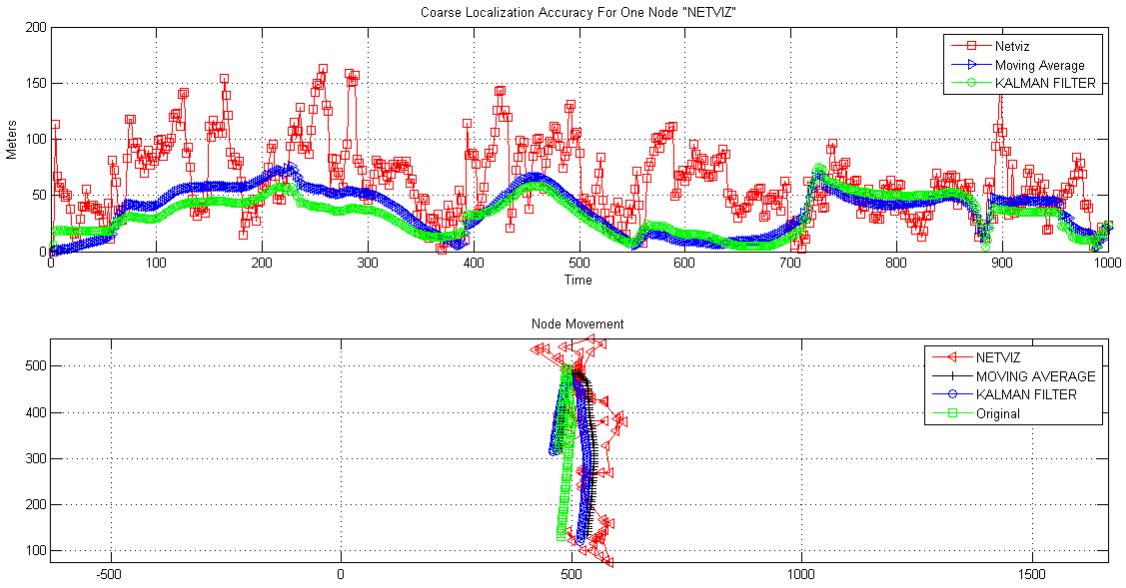


Figure 4-29 Kalman Filter vs. Moving Average

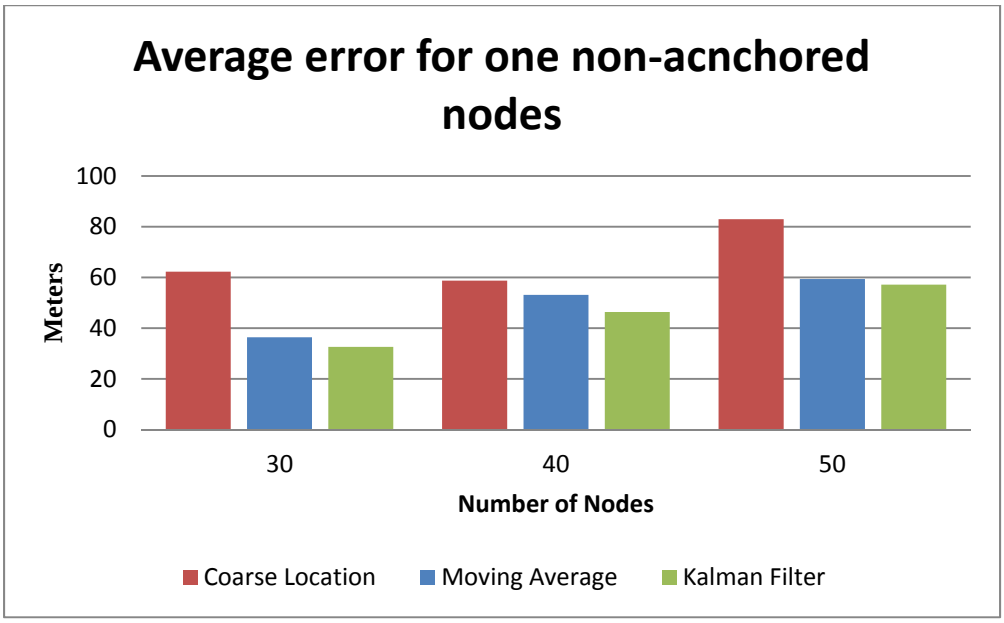


Figure 4-30 Gauss Markov

4.5.3 Random Way Point

The Figure 4-31 shows one scenario of 40 nodes, where kalman filter decreased the error rate of moving average about 5m. The parameters were used to get these results are

A =

A= [0.33 0 0 0

0 1 0 0

0 0 1 0

0 0 0 1]

R=.006

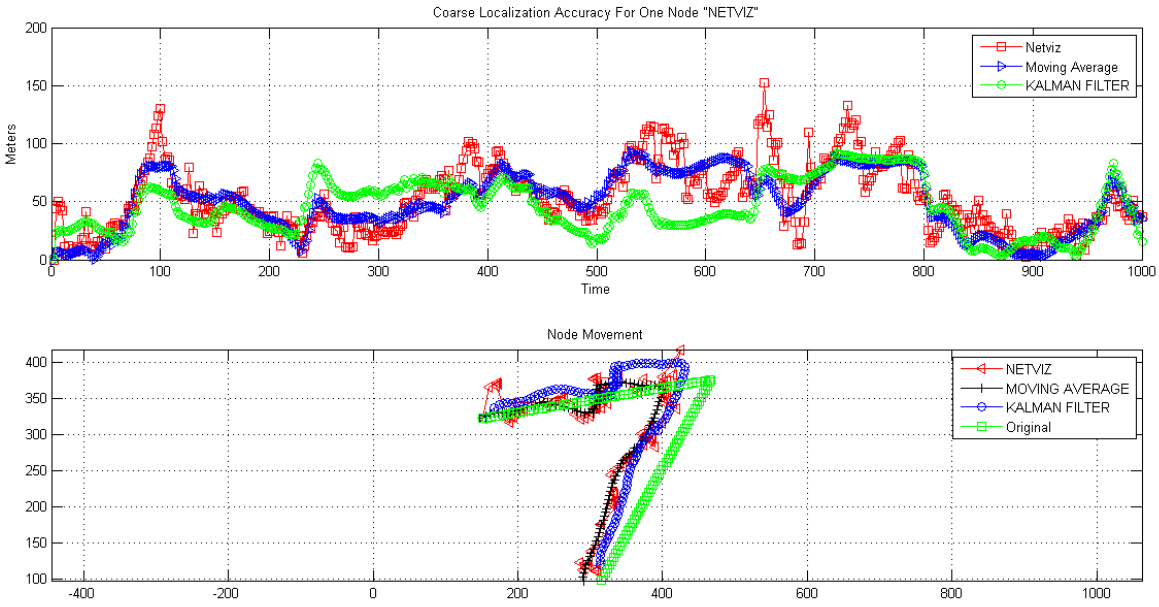


Figure 4-31 Kalman Filter vs. Moving Average

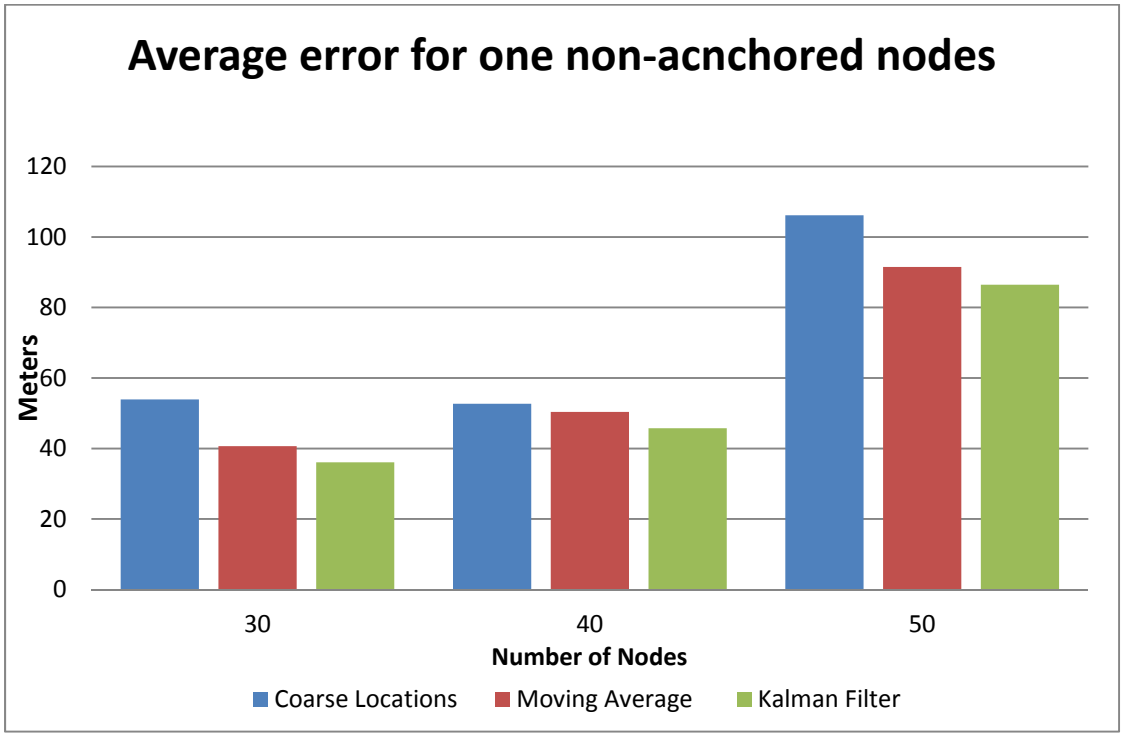


Figure 4-32 Random Way Point

4.5.4 Random Direction 2d

In order to have good results with applying Kalman filter to moving average, we should look into each node movement individually. The Figure 5-34 shows the result of one scenario for 40 nodes where error rate of Kalman filter decreased up to 3m comparing to moving average.

The Kalman parameters used

A= [0.33 0 0 0

0 1 0 0

0 0 1 0

0 0 0 1]

R=.009

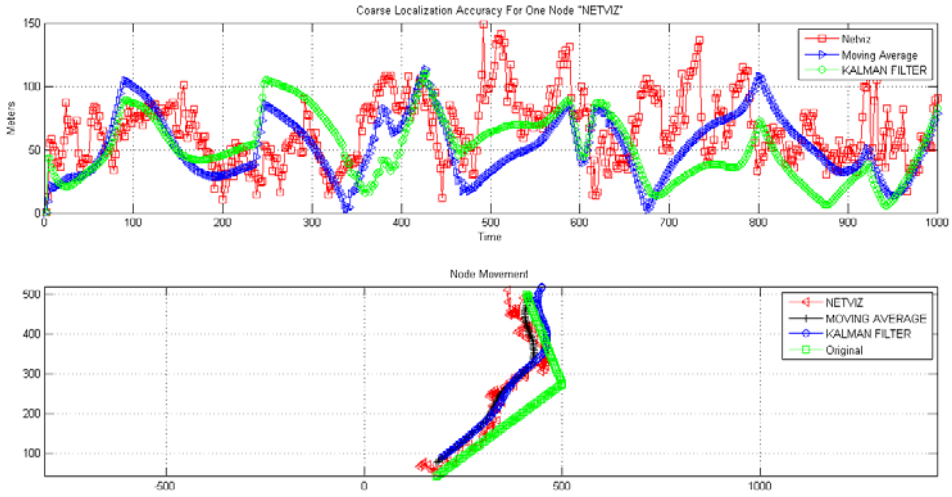


Figure 4-33 Kalman Filter vs. Moving Average

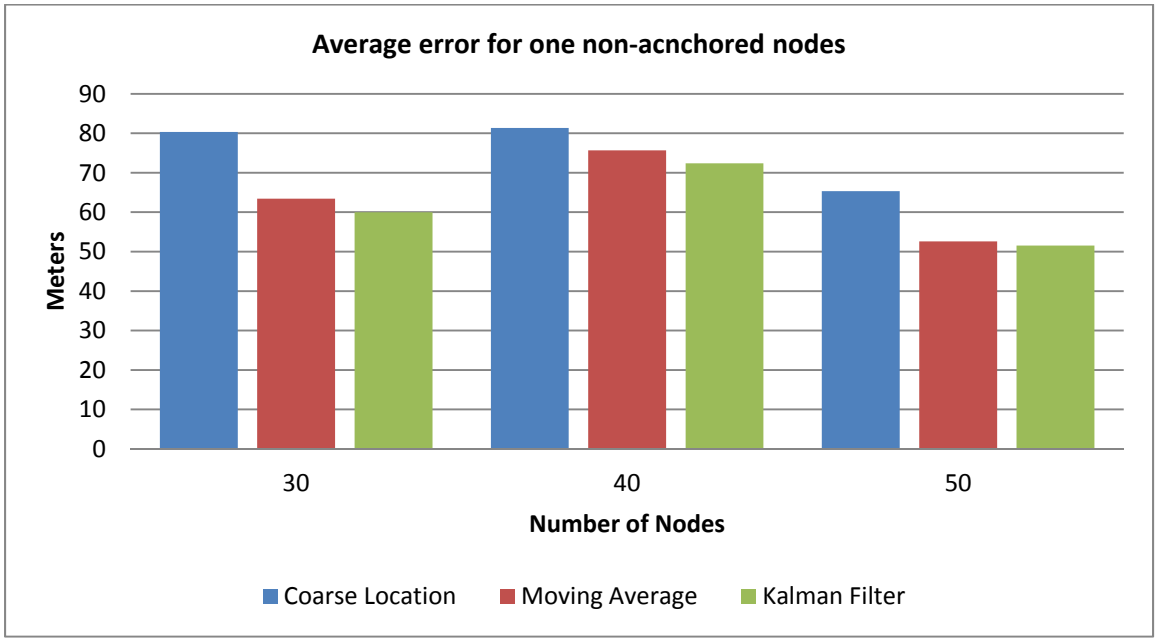


Figure 4-34 Random Direction 2D

4.6 Analysis

Moving Average filter was able to decrease the error rate for all the mobility models that were used whether the filter applies to each individual node or to the total number of non-anchored node.

The initialization of the approximate locations through the visualizer and taking the average of the first five points were useful to drop the initial error rate significantly. Moving average gave the best results among others filter for the all the scenarios due to the ability in removing all the unexpected bouncing of approximate locations.

Kalman filter shows good results for one mobility model which is Random Walk due to the node's movement where the node moves a constant distance and time without pause time hence the Euclidean distances between approximate points are close. In the other hand, Kalman filter decreased or increased the error rate a little for other mobility models in both cases one node and total number of non-anchored nodes due to the large Euclidean distances between approximate points.

Low pass filter gave decent results for Random Walk. It decreased the error rate for the scenario when it applied to one node however the error for the total number of non-anchored nodes was decreased with 30 and 40 nodes and increased with 50nodes. For other mobility models, the performance of the filter was not effective it just increased the error a slight due to large Euclidean distances and the filter doesn't allow the big values to be passed.

Applying Kalman filter on the top of moving average gave great results in the case where the error calculates for each individual node however the error still increased for the total number of non-anchored nodes comparing to the moving average.

Through the simulations and the experiments that were done, the results show that moving average was the best filters among others to improve the coarse locations. Using trial and error method, moving average will always decrease the error rate as long as the k is 5 and the span is 10 or more.

Kalman predicts the next value of data on the basis of previous value; due to large Euclidean distances the error is not satisfactory while moving average considers a bunch of data. Low pass filter seems to be worst simply because it doesn't allow big difference to reflect on the resultant data.

Chapter 5

Conclusion and Future Works

In the thesis, the objectives achieved were mobility prediction of the nodes and the determination of the topology. For the purpose of mobility prediction, different mobility models were examined and contrasted. The simulation section shows the analysis of different mobility models. For the localization of the network, coarse localization concept was examined which is based on OLSR protocol. In the determination of topology, there were two views, centralized and nodal view. Centralized view utilizes non-anchored nodes whose locations are calculated by the Forced directed algorithm. In simulations, location file was used which contains the anchor nodes responsible for situational awareness of the system. SAS played an important role in the localization since it helps in the generation of the location file in the module for the visualizer NS-3. The data generated from the visualizer were filtered by three filters as mentioned in the chapter 3. After the simulation, the best filter were found to be moving average as it produces better results compared to kalman and low pass filters. In the future work, Different localization techniques can be tested incorporating the nodal view. Furthermore, different filters can be explored to improve accuracy of the results.

Appendix A
Change to NetViz Application

Currently, the NetVis Converter works through the source topology and location data, converting them into NetVis format entry by entry, and outputting the results to a location where the NetVis visualizer can read them. The converter is supposed to output each entry in lockstep with the rate the visualizer consumes them because each entry is passed using the same file. However, it is possible for the converter to output twice, thus overwriting an entry before the visualizer can consume it.

This happens because the programs have no explicit synchronization instead rely on the Sleep function to provide an exact delay so that proper ordering of their operations is maintained. However, the Sleep function is fundamentally approximate, and defers to the OS scheduler as to when the thread should run again. This issue is exacerbated by the visualizer's use of threads with higher priority than those which perform the consumption of entries, which will pre-empt the consumption thread and defer its execution. This situation creates a race condition that leads to data loss.

To resolve this, the NetVis Converter has itself been converted to a preprocessor. A counter is used to suffix each output file from the converter. This provides an ordering that the NetVis visualizer can to consume the entries in the correct sequence without race conditions.

Additional modifications have been made to ensure the converter outputs a complete data set in the first entry files as initial positions, regardless of node markings (Anchor Nodes).

NetVis Converter.

The NetVis Converter is changed to a preprocessor

1. Every time a topology or location file is to be output, a counter is used as a suffix on the file extension.

a. Added an integer _fileCounter to the ConverterThread class (ConverterThread.java:41)

b. Added concatenation of the _fileCounter to the topology file name (ConverterThread.java:144)

c. Increment _fileCounter (ConverterThread.java:173)

d. Added concatenation of the _fileCounter to the partition data file name (ConverterThread.java:362)

e. Added concatenation of the _fileCounter to the full data file name (ConverterThread.java:393)

f. Increment _fileCounter (ConverterThread.java:420)

2. Sleeping at any time or showing animations are unnecessary

a. Commented out the Sleep function (ConverterThread.java:176)

b. Commented out the Sleep function and related branch structures (ConverterThread.java:195-197)

c. Commented out unnecessary catch statement relating to sleep (ConverterThread.java:251)

d. Commented out the Sleep function (ConverterThread.java:423)

e. Commented out the Sleep function and related branch structures (ConverterThread.java:459-461)

f. Commented out unnecessary catch statement relating to sleep (ConverterThread.java:560)

3. The converter should terminate when finished processing the data, rather than repeating.

a. Call the shutdown function after processing (ConverterThread.java:565)

b. Added a JOB_FINISHED event (ThreadEvent.java:29)

c. Add check to make sure we don't shutdown if we're already shutdown (ConverterThread.java:573)

d. Emit the JOB_FINISHED event on thread shutdown (ConverterThread.java:576)

e. Add a job counter to the Converter class (Converter.java:43)

f. Initialize the job counter to 0 (Converter.java:64)

g. Set the job counter to 2 for the topology and location processing jobs (Converter.java:147)

h. Create a handler for the JOB_FINISHED event, decrement the job counter, emit the CV_systemOff event when the job count goes to 0 (Converter.java:166-173)

4. The first set of partition data should be a complete set for initialization purposes.

a. The output statements have been moved out of the else body of the preceding branch statement, so that they will run even when the node hasn't been seen yet. (Converter.java:535-549)

b. An extra clause is added to the if statement that filters the values added to the partition data file. For the first file (_fileCounter equals 0), this ensures all values are added. (ConverterThread.java:539)

NetVis

1. Every time a new topology file is to be read, a file counter is incremented. The file path is determined by the counter value. This process stops with the last file in the sequence.

a. Added an integer _fileCounter to the CanvasThread class (CanvasThread.java:52)

b. Added a boolean _allFilesRead to the CanvasThread class (CanvasThread.java:53)

c. Initialize _fileCounter to 0 (CanvasThread.java:65)

d. Initialize _allFilesRead to false (CanvasThread.java:66)

e. Concatenate _fileCounter to the node topology path (CanvasThread.java:248)

f. Concatenate _fileCounter to the node file path (CanvasThread.java:256)

g. Concatenate _fileCounter to the node topology path (CanvasThread.java:821)

h. Once there are no more files, set _allFilesRead to true and revert to the last file (CanvasThread.java:823-828)

i. Concatenate _fileCounter to the node file path (CanvasThread.java:1007)

j. Once there are no more files, set _allFilesRead to true and revert to the last file (CanvasThread.java:1008-1014)

k. If more files remain, increment _fileCounter (CanvasThread.java:1050-1051)

l. Cease output after all the files have been read (CanvasThread.java:801)

References

[1] Internet Engineering Task Force. Mobile Ad-Hoc Networks (MANET) -WORKGROUP. [Online]. http://www.ietf.org/html.charters/manet-charter.html.

[2] J. Sun, "Mobile Ad-Hoc Networking: An Essential Technology for Pervasive Computing.," in International conference on Info-Net, 2001, pp. 316-321.

[3] J. Macker and M. S. Corson, "Mobile Ad-Hoc Networking (MANET): Routing Protocol Performance Issues and Evaluation Considerations," draft-ietf-manetissues-o1.txt, April, 1998.

[4] Dale Arden, Li Li, Paul Labbe, and Ying Ge, "self-Aware - Situational Aware : Integrated Handhelds for Dispersed Civil and Military Urban Operations," Inside GNSS, vol. 2, no. 2, pp. 34-45, March/April 2007.

[5] M. Gregoire and L. Beaudoin, "Visualization for Network Situational Awarness in Computer Network Defence," in Visualization aand the Common operational Picture. Meeting Proceeding RTO-MP-IST-043, Neuilly-sur-Seine, France, 2005,pp. 20-1-20-6.

[6] Anand Dersingh, Z M Faizul Islam, Shahram Shah Heydari, and Md. Mustafizur Rahman, "OLSR-based Topology Discovery for Mobile Situational Awarness System," in The 2nd International Conference on Ambient System, Networks and Technologies, Niagra Falls, Canada, 2011.

[7] M. Gerla, "From Battlefields to Urban Grids: New Research Challenges in Ad-Hoc Wireless networks ," Pervassive and Mobile Computing, vol. 1, pp. 77-93, 2005.

[8] Amitangshu Pal, "Localization Algorithms in Wireless Sensor Networks: Current Approaches and Future Challenges," Network Protocols and Algorithms, vol. 2, no.1, 2010.

[9] Thomas, R.W.; DaSilva, L.A.; MacKenzie, A.B.; , "Cognitive networks," New Frontiers in Dynamic Spectrum Access Networks, 2005. DySPAN 2005. 2005 First IEEE International Symposium on , vol., no., pp.352-360, 8-11 Nov. 2005

[10] Ntuen, Celestine A. ; Kim, Gwang M. ; Bowman, Elizabeth ; Iyer, Purush S, "An Agent-based Model Simulation of Multiple Collaborating Mobile Ad Hoc Networks (MANET), " in the 16th International Command and Control Research and Technology Symposium,Qu,Canada,2011

[11] T Camp, J Boleng, Vanessa Davies, "A Survey of Mobility Models for Ad Hoc Network Research," WIRELESS COMMUNICATIONS & MOBILE COMPUTING (WCMC): SPECIAL ISSUE ON MOBILE AD HOC NETWORKING: RESEARCH, TRENDS AND APPLICATIONS, vol. 2, pp. 483-502, 2002

[12] Bai, Fan; Helmy, Ahmed "A Survey of Mobility Models in Wireless Ad-hoc Networks,". Chapter 1 in Wireless Ad-Hoc Networks. Kluwer Academic. 2006.

[13] Boyd, J.R. 1987. An Organic Design for Command and Control. In Boyd, J.R. 1987. A

Discourse on Winning and Losing. Maxwell Air Force Base, AL: Air University Library Document No. M-U 43947 (Briefing slides)

[14] Liu, G. & Maguire Jr. (1996). A Class of Mobile Motion Prediction Algorithms for Wireless Mobile Computing and Communications. Mobile Networks and Applications Journal, Springer, 1(2), 113-121.

[15] Erbas, F., Steuer J., Kyamakya, K., Eggesieker, D. & Jobmann K. (2001). A Regular Path Recognition Method and Prediction of User Movements in Wireless Networks. Proceedings of the Vehicular Technology Conference (VTC' 2001),IEEE, 2672-2676.

[16] Lee, S-J., Su, W. & M. Gerla, (2001) "Wireless Ad Hoc Multicast Routing with Mobility Prediction", Mobile Networks and Applications, Vol. 6, 351-360.

[17] Z. Zaidi and B. Mark. Mobility estimation based on an autoregressivemodel. Submitted to IEEE Transactions on Vehicular Technology,Jan. 2004. (Pre-print) Available at URL: http://mason.gmu.edu/zzaidi.2004.

[18] McDonald A.B. & Znabi, T.F. (1999a). A path availability model for wireless ad hoc networks. Proc. IEEE WCNC'1999, New Orleans, USA, 35-40.

[19] Wang K., & Li, B. (2002). Group Mobility and Partition Prediction in Wireless Ad Hoc Networks. Proc. of the IEEE International Conference on Communications (ICC'2002).

[20] Akio Koyama, kenyu Kamakura, and Leonard Barolli, "MANET-viewer: AVisualization system for Mobile Ad-Hoc Networks," in Proceeding of the 7th International Conference on Advances in Mobile Computing and Multimedia (MoMM 2009 Workshop), 2009, pp. 452-457.

[21] Z M Faizul Islam, Mitchell Romanuik, Shahram Shah Heydari, and Md. Mazda Salmanian," OLSR-based Coarse Localization in Tactical MANET Situational Awareness Systems," IEEE International Conference on Communication, Ottawa, Canada, 2012.

[22] Alon Efrat, David Forrester, Anand Iyer, and G. Stephen Kobourov, "Force Directed Approaches to Sensor Localization," in 8th Workshop on Algorithm Engineering and Experiments, 2006, pp. 108-118.

[23] M. Sanchez and P. Manzoni. Anejos: A java based simulator for ad-hoc networks. Future Generation ComputerSystems, 17(5):573–583, 2001.

[24] (2011, December) Network simulator – NS-3. [Online]. http://www.nsman.org

[25] Xu Li, Nathalie Mitton, and David Simplot "Mobility prediction based neighborhooddiscovery in mobile Ad Hoc networks, " In Proceedings of the 10th international IFIP TC 6 conference on Networking - Volume Part I, Valencia, Spain,2011

[26] T. Kamada and S. Kawai, "An algoritm for Drawing General Undirected Graphs," Information Processing Letters, vol. 31, pp. 7-15, 1989.

[27] T. Clausen and P. Jacquet. Optimized Link State Routing protocol (OLSR), RFC 3626. [Online]. http://hipercom.inria.fr/olsr/rfc3626.txt

[28] Khandekar, A.; Bhushan, N.; Ji Tingfang; Vanghi, V.; , "LTE-Advanced: Heterogeneous networks," Wireless Conference (EW), 2010 European , vol., no., pp.978-982, 12-15 April 2010 doi: 10.1109/EW.2010.5483516.

[29] Rob J Hyndman “Moving averages” November 8,2009

[30] Greg Welch and Gary Bishop “ An introduction to the kalman filter” July 2006 http://www.cs.unc.edu/~welch/media/pdf/kalman_intro.pdf

[31] (2008, may) Digital Filters with MATLAB [online] http://www.mathworks.com/tagteam/55876_digfilt.pdf

[32] W.S. Harwin, N "Note on Klaman Filter, " March 25, 2009 http://www.personal.reading.ac.uk/~shshawin/dnb/notes_on_kalman_filter.pdf